

Technische Universität München

TUM School of Engineering and Design

**Simulation of Circuit Races for the Objective
Evaluation of Race Strategy Decisions**

Alexander Maximilian Heilmeyer, M.Sc.

Vollständiger Abdruck der von der TUM School of Engineering and Design der
Technischen Universität München zur Erlangung eines

Doktors der Ingenieurwissenschaften

genehmigten Dissertation.

Vorsitzender:

Prof. Dr. Constantinos Antoniou

Prüfer der Dissertation:

1. Prof. Dr.-Ing. Markus Lienkamp

2. Prof. Dr. Maximilian Schiffer

Die Dissertation wurde am 14. März 2022 bei der Technischen Universität München eingereicht
und durch die TUM School of Engineering and Design am 14. September 2022 angenommen.

“Just as in Formula One, the best strategists have won before they even get onto the field. If you are someone who loves intense racing, that can be a disappointment because winning looks too easy. But if you are interested in how you get to the position where winning looks easy, then you will find the sport more complex and subtle than any other yet invented.” Ross Brawn & Adam Parr [1, p. 86]

Preface

This thesis was written during my time as a Ph.D. student at the Institute of Automotive Technology at the Technical University of Munich in the context of the Roborace and Indy Autonomous Challenge research projects.

Special thanks go to my doctoral supervisor Prof. Dr.-Ing. Markus Lienkamp. He has always supported the research projects with his fullest strength and given my colleagues and me the necessary leap of faith. I owe him the opportunity to do my doctorate on this rather unusual topic, which I enjoyed very much.

Many thanks to Prof. Dr. Maximilian Schiffer for taking over the second examination of this dissertation and Prof. Dr. Antoniou for chairing the examination committee.

I would also like to express my deep gratitude to the mentor of my dissertation, Dr.-Ing. Michael Graf. He was the one who motivated me to work on this exciting topic at the beginning of my research period. He not only supported me in the first steps but also always stood by me with advice and critical questions.

Likewise, I would like to thank my former team lead Dr.-Ing. Johannes Betz and all my colleagues at the institute for the wonderful time and the excellent support in research and teaching. This is especially true for the colleagues in my research projects, with whom I have experienced many great but also many challenging moments. It will be hard to find such a motivated team again in this lifetime.

Thanks to the colleagues in the secretariat, the administration, and the workshops for their support in the daily work at the institute.

Furthermore, I would like to thank the students I had the privilege of mentoring on their theses and on whose results I have built for this thesis.

Last but not least, I would like to thank my wife, family, and friends for their ever-present support and patience during this dissertation project.

Munich, March 2022

Alexander Heilmeier

Contents

List of Abbreviations	III
Formula Symbols	V
1 Introduction	1
1.1 Race Strategy in Circuit Motorsport	1
1.2 Timekeeping in Circuit Motorsport	7
1.3 Short Overview of Popular European Circuit Racing Series	9
1.4 The FIA Formula 1 Championship	10
1.5 Motivation and Goal	12
2 State of the Art	13
2.1 Racing Line Generation	13
2.2 Lap Time Simulation	16
2.3 Race Simulation	19
2.3.1 Race Simulation Fundamentals	19
2.3.2 Probabilistic Influences in Race Simulations	20
2.4 Automation of Race Strategy Decisions	21
2.5 Derivation of the Research Questions	22
3 Methodology	25
3.1 Approach	25
3.2 Focus Area	27
4 Results	29
4.1 Databases	29
4.1.1 Race Track Database.....	29
4.1.2 Timing Database.....	30
4.2 Racing Line Generation	31
4.3 Lap Time Simulation	64
4.3.1 Quasi-Steady-State Lap Time Simulation	64
4.3.2 Revision of the Tire Force Calculation.....	76
4.4 Race Simulation	76

4.4.1	Deterministic Race Simulation	76
4.4.2	Parameter Determination for the Deterministic Race Simulation	86
4.4.3	Probabilistic Race Simulation	89
4.5	Automation of Race Strategy Decisions	112
4.5.1	Basic Strategy Optimization and Supervised Learning Virtual Strategy Engineer	112
4.5.2	Reinforcement Learning Virtual Strategy Engineer	146
4.6	Case Study	148
4.6.1	Racing Line Generation	149
4.6.2	Lap Time Simulation	150
4.6.3	Deterministic Race Simulation	153
4.6.4	Probabilistic Race Simulation	158
4.6.5	Race Strategy Evaluation	159
5	Discussion	167
5.1	Answers to the Research Questions	167
5.2	Critical Points	168
5.3	Outlook	172
6	Summary	173
	List of Figures	i
	List of Tables	iii
	Bibliography	v
	Prior Publications	xvii
	Supervised Student Theses	xix

List of Abbreviations

BSO	Basic Strategy Optimization
DRS	Drag Reduction System
DTM	Deutsche Tourenwagen Masters
F1	FIA Formula 1 Championship
FCY	Full-Course Yellow
FE	FIA Formula E Championship
FIA	Fédération Internationale de l'Automobile
GP	Grand Prix
LTS	Lap Time Simulation
RS	Race Simulation
SC	Safety Car
VSC	Virtual Safety Car
VSE	Virtual Strategy Engineer
WEC	FIA World Endurance Championship

Formula Symbols

Formula Symbol	Unit	Description
a_{tire}	lap	Tire age
a_y	m s^{-2}	Lateral acceleration
A	–	Action (reinforcement learning)
B_{fuel}	kg lap^{-1}	Fuel consumption
c_{tire}	–	Tire compound
$F_{x,\text{pot}}$	N	Tire force potential in longitudinal direction
F_z	N	Wheel load
$F_{z,0}$	N	Nominal wheel load
G	s	Return value (reinforcement learning)
i, j	–	Counting variables
k_0	s	Tire degradation model parameter (intercept)
$k_{1,\text{lin}}$	s lap^{-1}	Linear tire degradation model parameter (slope)
l	lap	Current lap in a race
l_{tot}	lap	Total number of laps in a race
$m_{\text{fuel,tot}}$	kg	Total fuel mass allowed to consume in a race
p	–	Rank position
p_{ref}	–	Reference rank position (reinforcement learning)
q	–	Action-value function (reinforcement learning)
r	–	Reward value (reinforcement learning)
r_{diffcomp}	–	Reward value depending on the usage of different compounds (reinforcement learning)
r_{laptime}	–	Reward value depending on lap time (reinforcement learning)
r_p	–	Reward value depending on rank position (reinforcement learning)
$r_{p,\text{final}}$	–	Reward value depending on final rank position (reinforcement learning)
r_{tot}	–	Total reward value of a race (reinforcement learning)

R	m	Corner radius
S_{mass}	s kg^{-1}	Mass sensitivity of the lap time
t_{base}	s	Basic lap time
t_{car}	s	Lap time increase due to car abilities
t_{driver}	s	Lap time increase due to driver abilities
t_{drs}	s	Lap time decrease due to DRS
t_{duel}	s	Lap time increase due to dueling drivers
t_{fuel}	s	Lap time increase due to fuel mass
$t_{\text{fuel,tot}}$	s	Total time loss in a race due to fuel mass
$t_{\text{gap,overtake}}$	s	Race time advantage required for an overtaking maneuver
t_{lap}	s	Lap time
$t_{\text{lap,ref}}$	s	Reference lap time (reinforcement learning)
$t_{\text{p,grid}}$	s pos^{-1}	Lap time increase due to grid position (first lap only)
t_{pit}	s	Total time loss of a pit stop
$t_{\text{pitdrive,inlap}}$	s	Lap time increase due to driving through the pit lane (in-lap only)
$t_{\text{pitdrive,outlap}}$	s	Lap time increase due to driving through the pit lane (out-lap only)
t_{Q}	s	Qualifying lap time
t_{race}	s	Race time
t_{racepace}	s	Race pace time delta
$t_{\text{race,tot}}$	s	Race duration
$t_{\text{standstill}}$	s	Lap time increase due to the start from a standstill (first lap only)
t_{tire}	s	Lap time increase due to tire degradation
v	m s^{-1}	Velocity
v_{c}	m s^{-1}	Corner speed
w	m	Track width
Z	–	Race state (reinforcement learning)
κ	rad m^{-1}	Curvature
μ_{w}	–	Modifier for the coefficient of friction due to weather conditions
μ_{x}	–	Coefficient of friction in longitudinal direction
$\mu_{\text{x},0}$	–	Coefficient of friction in longitudinal direction at nominal wheel load
π	–	Pursued policy (reinforcement learning)

1 Introduction

Only a few years after the invention of the automobile in the late 19th century, people started competing against each other by driving as fast as possible from town to town – motorsport was born. In the early years of motorsport, the focus was not necessarily on speed but rather on the technical reliability of the cars. For example, in the 1894 race from Paris to Rouen, only 15 of 102 starters reached their destination after 126 km [2, p. 30]. As a result of the public's growing interest in the races, automobile manufacturers themselves soon became involved in motorsport, recognizing its potential as a showcase for their products performing under extreme conditions. The statement “Win on Sunday, sell on Monday” originates from this period. In principle, nothing has changed to this day about the fact that motorsport is mostly a marketing activity, in addition to the development and testing of new technologies [3, p. 9]. For this reason, it is of great importance to perform well in the races to be able to promote successes in combination with the products of the respective brand. To be successful, the teams must exploit all opportunities that give them an advantage over their competitors. One of the ways to influence the outcome of races is race strategy, which is the thematic core of this thesis. In the following sections, an introduction to the background is given.

1.1 Race Strategy in Circuit Motorsport

Motorsport is a diverse field with many vehicle classes and racing series, each with its regulations. For land vehicles, a distinction can be made between driving along a given route and driving on circuits [3, p. 2]. In the former case, the participants usually drive from a start point A to an endpoint B, e.g., from Paris to Dakar in the original Dakar Rally. Thus, they pass most parts of the route only once. On the other hand, in circuit racing, the participants drive on a closed race track. The driver who crosses the finish line in the lead after the specified number of laps or elapsed time wins the race. Consequently, the goal of each race participant is to achieve the fastest possible lap time on average.

Entering and leaving a race track happens via a pit lane, usually located parallel to the start-finish straight. Figure 1.1 shows the integration of the pit lane into the layout of the Hockenheimring in southern Germany. However, the pit lane is not only relevant for accessing the race track. It also allows the drivers to make a pit stop at the end of each lap. During a pit stop, for example, the tires or the driver can be changed, or fuel can be refilled. All this impacts the rest of the race, which is why pit stops are a key element in circuit motorsport. As will become apparent in the following, race strategy is primarily based on the ability to influence the course of the race by choosing pit stops wisely.

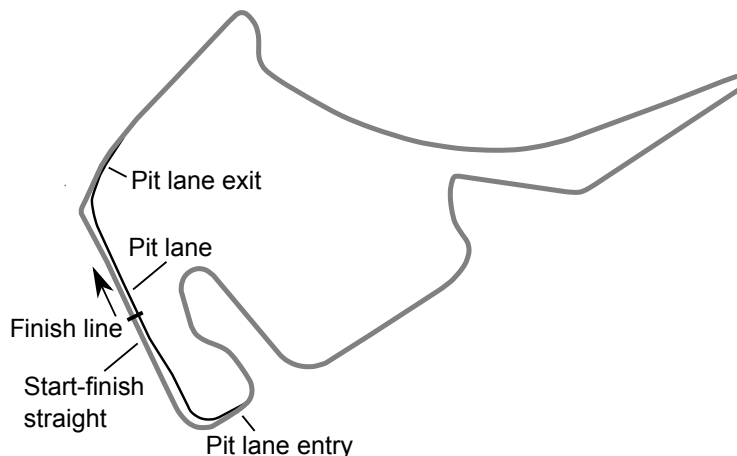


Figure 1.1: Layout of the Hockenheimring located in southern Germany based on [4].

The goal of race strategy is to finish the race in the best possible position, which (in the result) is mostly equivalent to finishing the race in the shortest possible race duration. Race strategy generally comprises the following aspects, the relevance of which may vary depending on the regulations of the respective racing series:

- Number and timing of pit stops
- Tires
- Fuel
- Yellow phases
- Tactical opportunities
- Driving style and vehicle setup

These aspects and the related backgrounds are presented in detail in the following.

Number and Timing of Pit Stops

Various aspects cause the necessity for pit stops. The most dominant one is to provide the race car with a fresh set of tires. Fresh tires allow the driver to achieve significantly faster lap times than a worn-out set since racing tires degrade quickly. With combustion-powered cars, another reason for pit stops is refueling. However, in many sprint racing series, refueling has been banned in recent years, primarily for safety reasons. Therefore, this aspect applies mainly to endurance racing. In electric racing series, the cars could theoretically be recharged during pit stops. However, since this would take a comparatively long time, in the first seasons of the FIA Formula E Championship (FE), for example, drivers simply swapped cars during a pit stop in the middle of the race. This has no longer been necessary since the 2018–2019 season due to an increased battery capacity in the cars. Further reasons for pit stops are repairing broken parts of the car, changing drivers, or changing the car's setup. The downside of pit stops is that the driver loses time relative to his opponents for two reasons. First, the car is stationary as long as the mechanics work on it. Second, driving through the pit lane is limited to a speed of 60 km h^{-1} to 80 km h^{-1} due to safety reasons while the other cars continue the race at full speed. [5]

Finding the best balance between the benefit and expense of making a pit stop is the major task of a race strategist. Generally speaking, a pit stop in lap l is beneficial if the sum of the lap times with a pit stop $t_{\text{lap|pitstop}}(i)$ in the remaining laps $l_{\text{tot}} - l$ plus the relative time loss due to the pit

stop t_{pit} is smaller than the sum of the lap times without a stop $t_{\text{lap|nopitstop}}(i)$ in the remaining laps as stated by

$$\sum_{i=l}^{l_{\text{tot}}} t_{\text{lap|pitstop}}(i) + t_{\text{pit}} < \sum_{i=l}^{l_{\text{tot}}} t_{\text{lap|nopitstop}}(i). \quad (1.1)$$

Consequently, a pit stop gets more appealing if a driver can drive significantly faster lap times afterward, e.g., because he is on worn-out tires or because the relative pit stop time loss is small, e.g., due to a yellow flag phase.

The general challenge is that future lap times are hard to estimate because they are subject to many influences. This ranges from minor influences such as the difficulty of predicting interactions between drivers with similar performance levels to significant influences such as yellow flag phases. On the one hand, this means that it is impossible to determine the optimal race strategy before a race, but that the team must continuously adapt the prepared strategy to the ever-changing race situation. On the other hand, this implies that the decision for a pit stop cannot be made separately for each race segment, but only in consideration of the overall strategy for the race since strategy decisions in later laps influence the future lap times.

Figure 1.2 illustrates the relations from Equation 1.1 using an example that deals with a pit stop near the end of a race. Since the example is limited to tire degradation, it is sufficient to consider the corresponding time loss instead of the overall lap times. The starting point is a driver who drives the 41st lap on tires with an age of 20 laps. In this example, the tires degrade linearly with 0.1 s lap^{-1} . Thus, the driver suffers a lap time loss due to tire degradation of 2.0 s on lap 41, of 2.1 s on lap 42, and so on. Adding up these values shows that starting from the end of the 40th lap until the end of the race after lap 55, the driver loses a total of 40.5 s due to tire degradation. If the driver instead enters the pits at the end of lap 40 and puts on fresh tires, this initially results in a time loss of 25 s at the end of lap 41. However, the degradation then starts from zero. Thus, the driver loses only 35.5 s until the end of the race, including the pit stop. This gives him an advantage of 5 s over the variant without a pit stop. This example is, of course, highly simplified since, for example, no interactions with other drivers on the track were taken into account.

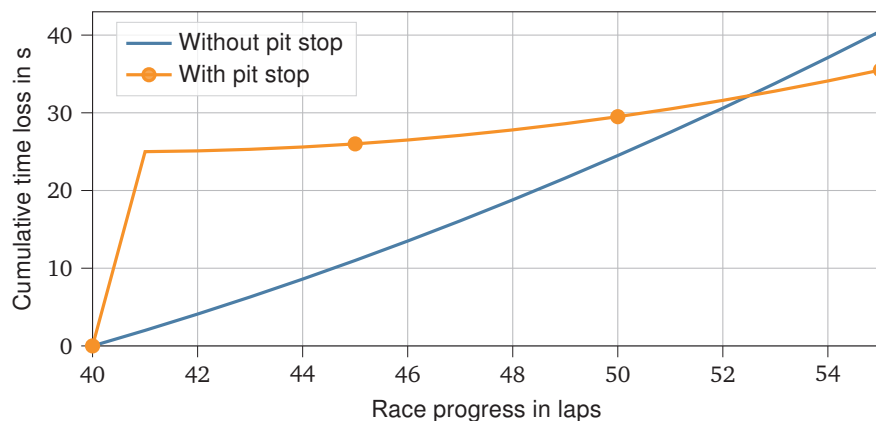


Figure 1.2: Comparison of the cumulative time losses due to tire degradation with and without making a pit stop in an example scenario.

Tires

It has already been discussed that a fresh set of tires enables faster lap times than a used set. Both performance and degradation behavior of racing tires depend primarily on the tire

compound used. In many racing series, there is more than one tire compound to choose from, resulting in various possible combinations for a race. For example, in the 2019 season of the FIA Formula 1 Championship (F1), the tire supplier provided three different compounds per race [6, art. 24.1]: *soft*, *medium*, and *hard*. These, in turn, were selected by the tire supplier from a total of five compounds available that season, depending on the track characteristics.

In general, a softer tire can transmit higher forces than a harder tire, allowing the driver to achieve faster lap times. The downside is that a softer tire degrades faster than a harder tire. This behavior is visualized in Figure 1.3 as an example. The intersection of the corresponding lines shows that a driver on a soft compound starts to lose time compared to a driver on a hard compound after approximately seventeen laps. Depending on how many laps remain until the end of the race and how much time is lost in the pits, it may be advantageous for the driver on soft tires to make another pit stop to replace them, as shown in the previous example.

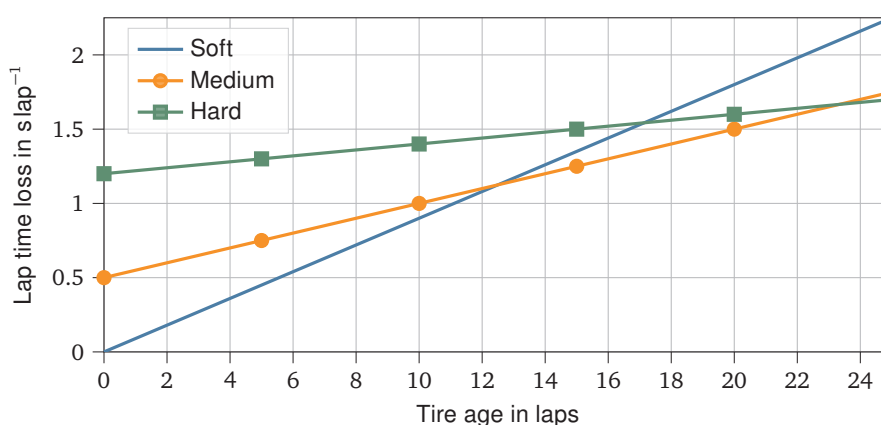


Figure 1.3: Visualization of the lap time loss due to tire degradation for three different compounds using a linear degradation model with example parameters.

In addition to specifying the tire compounds for the pit stops, a race strategist must also determine which compound the car starts the race with. For example, it must be taken into account that a softer tire usually comes up to temperature faster than a harder one. This is important at the start of a race to avoid immediately losing positions due to the initially tight driver field. However, if an early pit stop becomes necessary due to the softer compound, this can throw the driver back into traffic and negate the advantage.

Fuel

For combustion-powered cars, a further aspect of race strategy is determining the amount of fuel to be filled into the car at the start of a race and the amount of fuel to be refilled during pit stops (if permitted by the regulations). A higher mass of a race car causes slower lap times. This is because the acceleration forces increase linearly with increasing vehicle mass according to Newton's law, whereas the transmittable tire forces do not increase to the same extent due to wheel load degressivity, and the available engine power stays the same. Since race cars are built lightweight, fuel mass makes up a considerable part of the total mass of a combustion car at race start. For example, in F1, every participant is allowed to consume up to 110 kg of fuel per race [6, art. 30.5], which equals about 13 % of the total mass at race start (calculated based a minimum vehicle mass without fuel of 743 kg in the 2019 season [7, art. 4.1]). The fuel is burnt during the race and, therefore, the cars get lighter, which results in faster lap times with increasing race progress.

Determining the optimal fuel quantities for the race start and eventually refueling during pit stops depends on various factors. If refueling is not permitted, the starting point is that the fuel must last until the end of the race. If refueling is permitted, it is sufficient if the fuel lasts until the next pit stop. Based on this, it can be weighed up whether it is faster to drive more efficiently and with a reduced amount of fuel or with maximum power and higher consumption. This is, of course, dependent on engine efficiency and track characteristics. For refueling during pit stops, it must furthermore be considered that the fuel mass to be added affects not only the vehicle mass for the subsequent driving but also the time it takes to refuel the car and thus the standstill time.

The time advantage from a slightly lower fuel mass at race start quickly adds up throughout a race. It can be determined by comparing the total time losses in a race due to fuel mass $t_{\text{fuel,tot}}$. Based on the formula for a single lap [8, p. 2989], they can be approximately calculated by [8, p. 2989]

$$B_{\text{fuel}} = \frac{m_{\text{fuel,tot}}}{l_{\text{tot}}} \quad \text{and} \quad (1.2)$$

$$t_{\text{fuel,tot}} = \sum_{l=1}^{l_{\text{tot}}} ((m_{\text{fuel,tot}} - (l-1)B_{\text{fuel}})S_{\text{mass}}). \quad (1.3)$$

B_{fuel} is the fuel consumption per lap, calculated by dividing the total fuel mass for the race $m_{\text{fuel,tot}}$ by the number of laps in the race l_{tot} . S_{mass} is the mass sensitivity of the lap time. It is relatively constant in the range from minimum to maximum fuel mass [9]. Assuming that S_{mass} is 0.03 s kg^{-1} [10, p. 84] and engine power is unaffected, the advantage of starting with a fuel mass of 105 kg instead of 110 kg sums up to almost 4 s in a 50 laps race. When determining the fuel mass at race start, this result must, of course, be compared to the time disadvantage due to possibly lower engine power. It is believed that in recent seasons of F1, Mercedes, for example, has often been able to start with a lower fuel mass without suffering a relevant performance disadvantage to its competitors due to a more efficient engine [11].

Yellow Phases

In addition to tires and fuel mass, another aspect influences race strategy: various variants of yellow flags. Due to the battles for position on the track, accidents frequently occur in motorsport races. After an accident, track marshals must remove the crashed car(s) from the race track. For them to do this safely, the racing speed must be reduced. Therefore, race control has several options:

- Yellow flags waved in the affected sector
- Full-course yellow (in different variants)

In case of a minor danger for marshals and drivers, the favorable option is to wave yellow flags in the affected sector of the race track, indicating to the drivers that they must reduce speed. From a strategy perspective, the effect is negligible. Full-Course Yellow (FCY) phases are called out if a significant danger is present and a reduced speed is necessary for the entire race track. FCY phases thus have a big impact on race strategy. Depending on the regulations of a racing series, there are different realizations of the concept of FCY. In F1, for example, FCY phases are distinguished into Virtual Safety Car (VSC) and Safety Car (SC) phases. A VSC phase prescribes a minimum lap time for every driver which is about 140% of an unaffected lap time [12]. In addition, overtaking is forbidden. Since a VSC phase affects all drivers in roughly the same way, the gaps between them remain more or less constant throughout the phase.

Slight differences occur depending on where on the track a driver is at the start and end of a VSC phase. For example, if he is in a corner when the phase ends, his speed will be much closer to the actual race speed than if he were in the middle of a straight, resulting in a smaller time loss. VSC phases typically last for one to four laps [12]. During an SC phase, a real-world car drives onto the track in addition to the already reduced speed. It is placed ahead of the race leader, who must not overtake it. As with the VSC phase, overtaking is prohibited among the drivers. The SC drives lap times of about 160 % of the normal lap times [12]. Thus, all drivers line up behind it within one to two laps. This status is favorable in terms of safety. On the downside, the race is neutralized, i.e., the gaps between the drivers vanish. An SC phase typically lasts for three to eight laps [12].

In terms of race strategy, it is important to consider that the time lost during a pit stop under VSC and SC conditions is significantly lower compared to normal race speed. This is because the time loss depends on the difference between the time needed to get from pit entry to pit exit when making a pit stop and driving on the race track. Thus, the relative time loss reduces if the speed on the race track is decreased by a FCY phase, while the speed in the pit lane is always limited and, therefore, not affected. [13]

Tactical Opportunities

In the context of battles for position, race strategy offers several tactical opportunities. They are primarily relevant if the attacking driver is fast enough to follow but not to overtake the driver ahead, e.g., because both competitors have a similar performance level or if the track characteristic requires a significant lap time advantage for a successful overtaking maneuver. In such a case, one of the following options can be used to overtake the opponent indirectly [14]:

- Undercut
- Overcut
- Go long

If both cars are on a similar strategy, attacker A can make an early pit stop to fit a fresh set of tires known as an *undercut*. Thus, he can drive faster lap times in the following laps than the defending driver D on the worn-out tires. A will stay ahead after D's pit stop if his cumulative lap time advantage since his pit stop is larger than the gap between the two drivers was before the pit stops. D can counteract the undercut attempt by coming into the pits directly in the next lap. Thus, A has only a single fast lap which is often not enough to overtake. There are some preconditions for the undercut to work out. First, A must not be stuck in traffic after his pit stop such that he can drive fast lap times. This is known as driving in *free air*. Second, the new set of tires must provide enough lap time advantage compared to the old set. Thus, the undercut is usually applied when A changes from a harder to a softer compound. Third, the gap between D and A must have been small already before the attempt. Fourth, the pit stop must work smoothly, as time lost here can hardly be made up on the track. The disadvantage of an undercut attempt is that A is more vulnerable later in the race due to his early pit stop since his tires are older than those of the other drivers.

The *overcut* is similar but works the other way round. It is applied if D changes at first, mostly from softer to harder tires. In this case, A can delay his pit stop by a few laps and try to drive a few faster lap times than those of D and thus overtake him. The precondition for this to work is that A's tires are in reasonably good condition.

Go long means that A delays his pit stop for a significant number of laps when D made his pit stop. Thus, A does not directly gain a position but has much fresher tires at the end of the race, which could be enough to overtake D in that stage. Again, the precondition is that A's tires are in good condition. In addition, this works best if tire degradation stays low for a long time but increases significantly to the end. Executing *go long* also leaves the opportunity to spontaneously reduce the planned number of pit stops by one if tire degradation turns out to be lower than expected. Thus, A could stay in the lead until the end of the race.

Driving Style and Vehicle Setup

Finally, race strategy is one of the multiple influences on driving style and vehicle setup. Driving style comprises whether a driver goes as fast as possible or deliberately slows down to save fuel, reduce tire wear, and preserve the car. In this context, however, factors outside race strategy must also be taken into account, such as engine durability or limitations due to insufficient cooling at high temperatures. Therefore, driving style determination is a compromise between different aspects.

The vehicle setup is mainly tuned from a vehicle dynamics perspective so that the driver has confidence in the car and achieves the best possible lap times. However, the setup of the front and rear spoilers, for example, determines downforce and thus influences tire degradation behavior and top speed, thus overtaking performance on straights. As a result, race strategy aspects cannot be ignored for vehicle setup decisions to prevent, for example, excessive tire degradation or a lack of overtaking ability.

Summary

In summary, race strategy comprises the following aspects:

- It determines the number and timing of pit stops, trying to take advantage of possible benefits from the particular race situation, e.g., yellow phases. Part of this aspect is using tactical opportunities, such as the undercut.
- It determines the tire compounds fitted to the car during the pit stops.
- It determines the fuel mass at the race start and, if refueling is permitted, the amounts refueled during pit stops.
- It is one of the multiple influences on driving style and vehicle setup, as these, in turn, affect the aspects mentioned above, especially the tire degradation behavior.

1.2 Timekeeping in Circuit Motorsport

Timekeeping comprises the activity of measuring all kinds of time information. This is mainly lap times, sector times, and pit stop durations in circuit motorsport. In the context of race strategy, lap times t_{lap} are the most important data because they combine a wide range of relevant information in a single number, for example, on driver performance, car performance, tire condition, and race situation. They are measured by calculating the time delta between a driver's consecutive finish line crossings. Summing up the lap times up to a specified lap l results in the race time at

the end of that lap $t_{\text{race}}(l)$ as stated by [8, p. 2988]

$$t_{\text{race}}(l) = \sum_{i=1}^l t_{\text{lap}}(i). \quad (1.4)$$

Having the race times of all drivers available, the rank positions can be derived. They are assigned in the order of ascending race times. The race is finished when the leader crosses the finish line after completing the specified number of laps (or after a specified elapsed time in some series). The other drivers have to finish their current lap but must not complete eventually missing laps, for example, if they were lapped during the race. The final race times are referred to as *race durations* later on.

A race track is usually divided into three sectors for more fine-granular analyses. The timekeeper, therefore, does measure not only lap times but also sector times. This eases the comparison between different drivers and teams and allows conclusions on a car's setup. In F1, there are even more fine-granular sectors known as marshal-sectors. However, these measurements are not open to the public.

So-called timing boards are used in motorsport to overview the current race state. These boards show the most important data for all drivers in a compact format. An example is given in Table 1.1. *Gap* is the temporal distance to the leader, *interval* that to the driver ahead. The columns *lap*, *sector 1*, *sector 2*, and *sector 3* contain the previous lap- and sector times. As soon as a driver finishes the first sector, this time is entered, and the other two sector times are deleted. Consequently, it is visible in which track sector each driver is located.

Table 1.1: Example timing board showing the race state for the top five drivers at the beginning of lap 16 in the 2019 Hungarian Grand Prix. Driver abbreviations: VER – Verstappen, HAM – Hamilton, LEC – Leclerc, VET – Vettel, SAI – Sainz. Compound abbreviations: M – Medium, S – Soft.

Position	Initials	Gap	Interval	Lap	Sector 1	Sector 2	Sector 3	Compound	Stops
1	VER	–	–	81.844 s	29.343 s	29.206 s	23.083 s	M	0
2	HAM	2.327 s	2.327 s	81.899 s	29.372 s	29.171 s	–	M	0
3	LEC	14.597 s	12.422 s	82.722 s	29.451 s	29.679 s	–	M	0
4	VET	16.800 s	2.333 s	82.748 s	29.380 s	29.708 s	–	M	0
5	SAI	36.751 s	20.118 s	83.486 s	29.643 s	–	–	S	0

The location of the pit lane next to the start-finish straight has several implications for timekeeping. If a car drives through the pit lane, it inevitably also passes the finish line. To guarantee consistent timekeeping, the finish line is either located in front of the first (most race tracks, e.g., Silverstone in Figure 1.4a) or behind the last pit (some race tracks, e.g., Monaco in Figure 1.4b).

Both adjacent lap times are affected by a pit stop. If the finish line is located in front of the pits, the lap time of the *in-lap* (driving into the pits) is affected only slightly, depending on the courses of race track and pit lane as well as the pit speed limit. In F1, for example, the lap time of the in-lap mostly rises by 1 s to 3 s. However, the in-lap is even slightly faster than a normal lap in some cases. This is the case for Silverstone, for example, because the race track has two slow corners in front of the finish line (Figure 1.4a). The lap time of the following lap, known as *out-lap* (driving out of the pits onto the track), mostly rises by 15 s to 25 s in F1. This is mainly due to driving with the pit speed limit but also due to the standstill time while changing tires. If the finish line is located behind the last pit, the effects appear the other way round.

During free practice and qualifying sessions, each stint starts and ends in the pit lane. A stint is defined as the period a car drives on the race track between two pit stops. If the finish line is

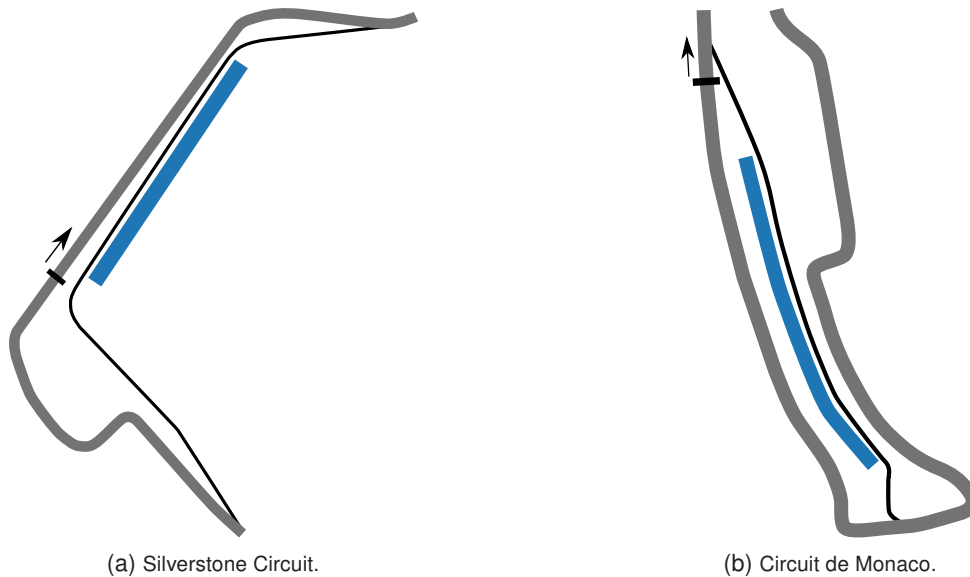


Figure 1.4: Excerpts of the Silverstone and Monaco track maps based on [15]. The pit lanes are shown in black, the tracks themselves in gray. The blue areas mark the actual pit locations of the teams.

located in front of the first pit, the timekeeper can not measure a lap time for the out-lap. The same counts for the in-lap on race tracks with the finish line behind the last pit. In the race, the drivers line up in front of the start line. It is located differently from the finish line on most race tracks. At Silverstone, for example, a start in front of the finish line would mean that some drivers would have to start in a corner, which is disadvantageous compared to the other drivers (Figure 1.4a). Consequently, the start line of the Silverstone circuit is located behind the finish line, approximately in the middle of the start-finish straight. As a result of the different positioning of the start and finish lines, the first lap time of a race can be calculated (from race start until crossing the finish line for the first time). Still, it is not directly comparable to the following lap times due to the different distances covered.

1.3 Short Overview of Popular European Circuit Racing Series

With an estimated 500 million global fans in 2019 [16], the F1 is among the most popular circuit racing series in the world. Besides, there are many other circuit racing series that are primarily aimed at the European market, such as FIA Formula E Championship (FE), Deutsche Tourenwagen Masters (DTM), and FIA World Endurance Championship (WEC). The rules of a racing series are usually divided into technical and sporting regulations. The governing body of a series determines them, often the Fédération Internationale de l'Automobile (FIA). The technical regulations provide a framework for the car's development, while the sporting regulations define how the races are run from a sporting perspective. Therefore, the regulations heavily impact the importance of race strategy for the racing series.

The FE is a racing series based on fully electric race cars that started in 2014 [17]. The series is less attractive in terms of race strategy for two reasons. First, the tires have an all-weather tread and are durable, so they often last the entire race weekend [18] and do not need to be changed during a race. As the batteries of the cars also last the whole race since the introduction of the

second-generation cars in the 2018–2019 season, there are almost no more pit stops in FE. Second, the cars do not burn fuel, and thus their mass does not reduce throughout the race. Consequently, an intelligent energy management strategy is more critical than a conventional race strategy for these races.

The DTM is a touring car series sanctioned by the Deutscher Motorsport Bund. Refueling was banned in the 2012 season [19]. Nevertheless, 2013 and 2014 were seasons in which race strategy played an important role. In those years, drivers had the choice between two tire compounds, *option* (soft) and *prime* (hard). In addition, a Drag Reduction System (DRS) was introduced in 2013 [20]. This system reduces a car's drag by folding down its rear wing flap as soon as the driver activates it. A driver is only allowed to use DRS if he follows another driver within a certain time window, mostly 1 s. Furthermore, the number of activations per race is limited. Folding down the rear wing flap increases the maximum speed on straights and thus eases overtaking maneuvers. At the start of braking for the next corner, the wing is automatically returned to its default position. Since the 2015 season, race strategy has become less important again as the drivers have only a single tire compound at their disposal [21, art. 25.1]. In the 2015 and 2016 seasons [21, art. 39.1], and again in the 2020 season [22], strategic freedom was further restricted by prescribing pit stop windows that allowed pit stops only within certain laps. The DTM regulations were completely changed for the 2021 season. Cars of the FIA GT3 category, which is a common category in many racing series, are now used. It remains to be seen whether race strategy will play a greater role again for the future DTM. At least the 2021 season was run with a single tire compound [23].

The WEC is a long-distance championship that has been running in its current form since 2012 [24]. Four different car categories run simultaneously in the races: LMP1, LMP2, GTE Pro, and GTE Am [25, art. 1.1]. LMP1 and LMP2 include prototypes that are manufactured specifically for this racing series. The GTE category corresponds to the former GT2 category. It is divided into professional drivers (GTE Pro) and amateurs (GTE Am). Most of the races are 4 h, 6 h, and 8 h races, with two exceptions: 1000 Miles of Sebring and 24 Hours of Le Mans. Due to the long race durations, the cars are not only equipped with fresh tires but also refueled during the pit stops [25, art. 12.2.2]. Furthermore, several drivers share a car, so they also change during pit stops [25, art. 12.2.2]. Regarding race strategy, the central aspect of such races is to extend the stints as long as possible to spend as little time as possible in the pit. Stint lengths are usually determined by the need to refuel, as one set of tires often lasts two to three stints. A few hours before the end of a race, the strategy engineers plan the final pit stops such that the cars reach the finish line with only a small amount of fuel left. Given the long durations and different car categories, a simulation of the course of such races is almost impossible.

Due to the importance of F1 for this thesis, the racing series is presented separately and in more detail in the next section.

1.4 The FIA Formula 1 Championship

The F1 is the highest class formula racing series in the world. Ten to twelve teams, each with two drivers, have competed each season in the last decade. A season consists of about 20 races called Grand Prix (GP) on mostly different race tracks. Each driver and team receive championship points based on their position at the end of a race [6, art. 6.4]. At the end of a

season, the driver and team with the most points are awarded the drivers' and constructors' championships [6, art. 6].

The teams in F1 are equipped with enormous budgets, but there are also significant differences between them. While Ferrari had 410 million dollars [26] available at the top end, for example, Force India had to manage with 120 million dollars [27] at the bottom end in the 2018 season. As a result of that amount of money in the sport, each aspect that helps to improve rank positions is exploited. Therefore, in conjunction with a high degree of strategic freedom, race strategy is highly important in this series.

From a strategy point of view, sporting regulations are of primary interest. The number of laps per race is determined such that the race distance exceeds 305 km [6, art. 5.3]. The only exception from this rule is the race in Monaco, which covers a distance of only 260 km due to the low average speed [6, art. 5.3]. Per race weekend, the tire supplier (currently Pirelli) provides two (up to the 2015 season) or three (since the 2016 season) different tire compounds for dry track conditions [6, art. 24.1]. Depending on the specific properties of the race track, they are chosen from a range of four to seven available tire compounds per season, see Table 1.2.

Table 1.2: Overview of available tire compounds in the seasons from 2014 to 2019 as published in [13]. The column names are inspired by the 2019 season compound names to enable comparison of the compounds over the years. A1 is the hardest compound, and A7 is the softest.

Season	A1	A2	A3	A4	A5	A6	A7
2014	Hard	Medium	Soft	Supersoft	–	–	–
2015	Hard	Medium	Soft	Supersoft	–	–	–
2016	Hard	Medium	Soft	Supersoft	Ultrasoft	–	–
2017	Hard	Medium	Soft	Supersoft	Ultrasoft	–	–
2018	Superhard	Hard	Medium	Soft	Supersoft	Ultrasoft	Hypersoft
2019	–	C1	C2	C3	–	C4	C5

For example, if the tires are heavily stressed on a race track in the 2019 season, C1, C2, and C3 are available for selection, whereas C3, C4, and C5 are provided on less stressful city circuits. For simplicity, since the 2019 season, instead of the actual compound names C1 to C5, fans are only shown the relative hardnesses within the selection of the respective race weekend, i.e., soft, medium, and hard. Unless stated otherwise, the compound names in this thesis always refer to the absolute values A1 to A7 as introduced in Table 1.2 to avoid confusion. In addition to the dry compounds, the teams have two wet compounds available: *Intermediate* for light and *wet* for heavy wet conditions. An important rule in this context is that every driver has to use at least two different tire compounds per race [6, art. 24.4].

Refueling during pit stops is banned since the 2010 season. Therefore, the cars have to finish a race with a maximum of 100 kg (2014 to 2016) [28, art. 29.5], 105 kg (2017 and 2018) [29, art. 30.5] or 110 kg (2019) [6, art. 30.5] of fuel.

At the end of the 2000s, it became obvious that the aerodynamic devices of the cars caused a lot of *dirty air* behind them. Due to the generated turbulences, it became harder and harder to pursue a car ahead in close proximity to overtake it. Therefore, F1 introduced a DRS in the 2011 season. A driver is allowed to activate the system on specified straights if he is less than 1 s behind the car ahead of him [6, art. 21.5]. Different sources estimate the speed advantage of the DRS to be between 5 km h^{-1} [3, p. 203] and 15 km h^{-1} [30]. However, overtaking on the track is still hard due to aerodynamics, which is why undercut and overcut are powerful strategy elements in F1 [10, p. 79].

The course of a race weekend is similar to most circuit racing series. In F1, there are three free practice sessions [6, art. 32]. Usually, the first free practice session is used to find a basic setup for the car, the second session to perform some stints under simulated race conditions, and the third session to prepare for the subsequent qualifying. The goal of the qualifying is to determine the starting order for the race. The driver with the fastest lap time starts in the first position, followed by the other drivers in the order of ascending qualifying lap times. In F1, the qualifying is divided into three sessions: Q1, Q2, and Q3 [6, art. 33]. After Q1, the five to six slowest drivers (depending on the size of the starter field of a specific season) are dismissed. Thus their starting order is fixed. The same happens after Q2. The starting order of the ten remaining drivers is determined in Q3. An aspect to mention is that the drivers participating in Q3 must start the race on the tires they have used to set their fastest lap time in Q2 (except for wet conditions on race day). Using the tires from Q2 avoids drivers trying to save their tires in Q3 because the fastest driver should be in pole position. This puts the top ten drivers at a slight disadvantage compared with the rest of the field, which can choose its tires freely. In the race, the drivers finally compete against each other. It finishes as soon as a predefined number of laps is completed [6, art. 5.3].

1.5 Motivation and Goal

In this chapter, the possibilities of race strategy to influence the race result were explained. It was shown why race strategy is an essential aspect of circuit racing when the regulations provide the necessary strategic freedom. This can be further substantiated by some quotes that show that wrong race strategy decisions often lead to sub-optimal race results:

“Mercedes has discovered a ‘bug’ in the tool it uses for its Formula 1 virtual safety car calculations, after concluding its investigation into what went wrong at the Australian Grand Prix” [31]

“Mercedes Formula 1 team boss Toto Wolff says he ‘fully understands’ Ferrari’s unsuccessful decision to switch Sebastian Vettel to a two-stop strategy in the Spanish Grand Prix” [32]

“Lewis Hamilton swept to a virtually unchallenged win in the 2018 Formula 1 Singapore Grand Prix as another Ferrari tactical blunder cost Sebastian Vettel the chance of victory” [33]

This raises the question of how race strategy can be determined reasonably and objectively. Big racing teams, especially those in F1, have entire departments working on this topic. They use the available data from free practice and qualifying sessions as well as from previous races and seasons to parameterize simulation models with which they can simulate a race. Thus, they can prepare a basic race strategy to start with as well as reactions to unforeseen race events. During the race, they continuously reassess their strategic options based on the race situation and adjust the strategy if necessary. However, due to the secrecy resulting from the tough competition, the teams do not publish their methods and models. Consequently, teams with less budget that compete in smaller racing series cannot profit from that knowledge. Furthermore, almost no literature can be found on race strategy, which is why the state of the art in racing teams does not correspond to that in science. Closing this gap and providing a starting point for future research is the motivation of this thesis. The goal is to develop the methods and tools necessary to make objective race strategy decisions. For this purpose, it is essential to be able to evaluate the effects of a decision on the course of a race and, ultimately, the result. Besides this application, some methods and tools can also be used for related tasks, e.g., in autonomous racing, as will be shown in the following chapters.

2 State of the Art

In the following, the state of the art in four research areas related to the simulation of circuit races is highlighted. These areas are racing line generation, lap time simulation (LTS), race simulation (RS), and the automation of race strategy decisions. Afterward, based on the state of the art and the research goal, the research questions are derived.

2.1 Racing Line Generation

Thematic Background

The goal of race strategy and the driver is to complete every lap, and ultimately the race, as fast as possible. The driver can influence the lap times by optimizing the trajectory he drives along the race track. A trajectory is composed of two parts: path and velocity profile. Assuming that the driver exhausts the vehicle dynamics limits, the optimal velocity profile for a given path can be calculated. Thus, determining the fastest trajectory can be substituted by determining the fastest path for a given longitudinal and lateral acceleration potential of the race car, known as *racing line*. According to the literature, a racing line can be sufficiently represented by three basic elements: straights, constant-radius arcs, and clothoids [34, 35]. Nevertheless, a driver needs lots of experience to recognize and drive the optimal racing line. This line can only be determined by considering consecutive corners [36, p. 324], or ultimately the race track as a whole. However, the basic concept behind racing lines can also be explained with a single corner. Trzesniowski [36, p. 322] distinguishes three different types of lines through a corner, which are shown in Figure 2.1:

- Classic racing line (maximum radius and mid-corner speed)
- Early apex line (maximum corner entry speed)
- Late apex line (maximum corner exit speed)

The classic racing line follows the largest possible radius R through the corner. This allows to drive the highest minimum corner speed $v_{c,\max}$ at the maximum lateral acceleration of a car $a_{y,\max}$ as stated by

$$v_{c,\max} = \sqrt{a_{y,\max} \cdot R}. \quad (2.1)$$

An early apex means that the driver starts to turn in early and, therefore, reaches the inner boundary of the track earlier than on the classic racing line. This is the typical line for a driver who tries to overtake another driver on the inner side of the corner [36, p. 322]. In total, the early apex line is slower than the other lines. However, the overtaken driver cannot use the speed

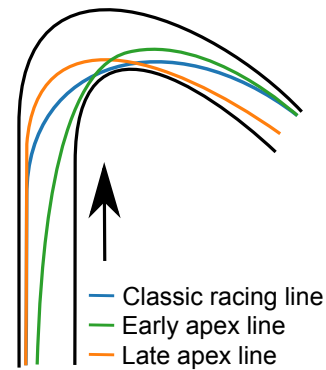


Figure 2.1: Visualization of three different types of lines through a corner: classic racing line, early apex line, and late apex line based on [36, p. 323].

advantage of another line if the overtaking driver blocks it. A late apex is usually chosen if the driver tries to maximize the speed at the corner exit, e.g., if it is followed by a long straight.

The choice between those lines is based on race situation and vehicle dynamics limits. The more longitudinal acceleration potential a car has in relation to its lateral acceleration potential, for example, the more the ideal line will shift towards a late apex [36, p. 322]. Compared to the classic racing line, the late apex line allows later braking and earlier acceleration at the expense of a lower minimum corner speed. This results in a time advantage when the longitudinal force potential is high, as the car spends less long in the area bounded by the lateral acceleration potential and longer in the area bounded by the longitudinal acceleration potential [36, p. 323]. Conversely, if the car is stronger in the lateral than in the longitudinal direction, the driver will prefer the classic racing line to take as much speed as possible through the corner. Apart from these considerations, additional influences such as different friction potentials (tarmac differences, wet spots), corner combinations, and traffic on the race track must be considered in reality [36, p. 322].

Literature

The algorithms used for racing line generation are part of the area of trajectory planning. Some approaches plan entire trajectories while others plan only paths, so a velocity planner must accompany them. Paden et al. [37] distinguish three categories: variational methods, graph search methods, and incremental search methods.

Variational methods are built on a variation of the parameters or control inputs that determine the vehicle's movement. For example, if a series of splines represents the path, the algorithm varies the spline parameters, changing the path on the track to optimize an objective function. This is used by Braghin et al. [38] to minimize the curvature along the path. Optimal control and model predictive control are also part of variational methods. In these approaches, the control inputs of a vehicle model are varied [39–41].

Graph search methods are based on discretizing the possible configuration space, i.e., the track and, if applicable, the velocity profile. These discretization points are called nodes. Edges, often splines, then connect the nodes of adjacent layers. Usually, the algorithms directly remove those edges that violate hard constraints, such as minimum cornering radii or maximum accelerations. In addition, edges that would cause a collision with static or dynamic objects in the vehicle surroundings can be removed. Finally, the cost of every edge is calculated based on a given heuristic. The algorithm then follows the edges with the lowest costs. Popular graph search methods are Dijkstra [42], A* [43, 44], and D* [45–47].

Incremental search methods work similarly to graph search methods. In contrast, however, they do not discretize the configuration space in fixed intervals but sample it randomly. The sampled points are then connected to obtain a path or trajectory. The logic for connecting the points must ensure that the result is drivable in terms of minimum cornering radii and maximum accelerations. Then, the best variant can be chosen, for example, the fastest one. As a consequence of the random sampling, the result improves the longer the algorithm runs, making more and more points available. Thus, early stopping could lead to an unsatisfying result. The principle is used for RRT [48] and its successors RRT* [49] and RRT^x [50], for example.

Many methods published in the field of trajectory planning assume driving at low lateral and longitudinal accelerations and are therefore not well suited for racing line generation. For example, due to the high accelerations occurring in motorsport, the otherwise often neglected effects in the non-linear range of vehicle dynamics should be taken into account in model-based planning approaches, e.g., the influence of longitudinal and lateral wheel load transfers together with non-linear tire models. However, some works related to regular road traffic also address the area of high accelerations since it is relevant, for example, for the planning of evasive maneuvers on normal roads [51, 52].

Methods used for trajectory planning in racing must keep the car at the limits of handling throughout the lap. Their goal is to minimize the lap time directly or indirectly while maintaining the vehicle dynamics limits. Furthermore, the resulting paths and thus the curvature profiles must be smooth to be driven at high speeds. Geometry-based approaches well fulfill this requirement. Braghin et al. [38] minimize the curvature along the race track by solving a quadratic optimization problem that shifts the discretized racing line points between the left and right track boundaries. They state that the optimal racing line is a compromise, weighted depending on the vehicle dynamics, between minimum-curvature line and shortest path. Based on this, Cardamone et al. [53] use a genetic algorithm to optimize the weighting between minimum-curvature line and shortest path individually for each track segment. The idea of Kapania et al. [54] is quite similar to that of Braghin et al. [38]. However, it works the other way round: At first, a velocity profile is generated, followed by a convex optimization of the path.

Apart from geometry-based approaches, many publications in the racing context rely on model-based optimization approaches such as optimal control and model predictive control. One can distinguish planners with a limited optimization horizon [55–57], approaches based on learning model predictive control [58, 59], and globally optimal minimum-lap time planners [60–71]. The publications in the latter category differ primarily in the level of detail of the models. For example, Herrmann et al. [62] also take into account a thermodynamic model of the electric powertrain when determining the optimal trajectory. The optimization over a limited horizon is often chosen to reduce the computational effort if a planner has to run online on the car. Furthermore, static and dynamic obstacles can be considered [55]. Learning model predictive control approaches improve the lap time lap-wise until they reach saturation. Minimum-lap time planners are mostly too slow for real-time applications, so their main application is offline optimization. They come closest to the optimal racing line.

A disadvantage of the model-based approaches is that they require detailed vehicle models and, thus, many well-known parameters. Therefore, other approaches were also investigated. Jain et al. [72] use Bayesian optimization to compute a racing line, which requires only the center line and track widths of the race track and the mass and center of gravity position of the car. However, they find that the approach does not scale well for long tracks with many corners. The publications by Jeon et al. [73] and Arab et al. [74] are based on RRT*. Funke et al. [75], Rizano

et al. [76, 77], and Glaser et al. [78] work with predefined geometries (e.g., straights, clothoids, constant-radius arcs) and maneuvers, respectively, that are composed to obtain the final line.

2.2 Lap Time Simulation

Thematic Background

As the name suggests, an LTS is used to calculate the lap time of a race car as accurately as possible. Therefore, LTS are often used for virtual setup optimization [79, p. 7]. Given the nature of the problem, LTS is closely related to the field of trajectory planning. In contrast to standard trajectory planning methods, however, LTS has its focus on fully exploiting the race car's longitudinal and lateral acceleration capabilities. The influence of a human driver is not modeled in most LTS since it is assumed that race drivers can make full use of their car.

Two groups of LTS can be differentiated. The first one, represented by LTS based on steady-state and quasi-steady-state modeling, acts as a velocity planner for a predefined racing line [79, p. 21], which can be created by an algorithm or from real measurements. The second group is represented by trajectory planning algorithms that simultaneously optimize path and velocity profile, mostly based on optimal control and model predictive control methods with transient simulation models. The three modeling approaches differ in computational speed and accuracy.

As the name suggests, steady-state and quasi-steady-state modeling assumes a steady system state in all discretization points, i.e., the vehicle states are time-independent. Consequently, transient effects such as actuator rate limits, yaw inertia, and the delayed response of tire forces when increasing slip or wheel load are neglected [79, p. 21]. In a steady-state solver, the race car is modeled as a point mass with maximum longitudinal and lateral accelerations. The race track consists of two segment types, straights and corners, whereby each corner has a fixed radius [80, p. 9]. The longitudinal (on straights) and lateral (in corners) acceleration capabilities of the car are considered separately by the solver [81]. Consequently, combined accelerations, e.g., when braking into a corner, are not modeled. This and the assumption that every corner segment is driven with a fixed velocity (due to a single radius) [81] impose limitations on the simulation accuracy. Steady-state solvers no longer make sense due to the computing power available today.

Quasi-steady-state solvers are a significant improvement over steady-state solvers. The race track is represented by small segments, each with a specific curvature [81]. In contrast to a steady-state solver, it is therefore not necessary to decide for each segment whether it is rather a straight or a corner when modeling the track (which in the steady-state solver determines whether it considers purely longitudinal or lateral accelerations in that segment). Consequently, combined accelerations are considered in the quasi-steady-state solver, which increases the validity of the simulation result. In the simplest variant, a g-g diagram is used for modeling the vehicle dynamics [79, p. 21]. It defines the maximum possible longitudinal acceleration for a point mass as a function of the acting lateral acceleration (and vice versa). Depending on the real vehicle behavior, different shapes can be used to model the possible combined accelerations, for example, circle, ellipse, and rhombus. Examples for these shapes are shown in Figure 2.2. They consider that the positive longitudinal acceleration in most cars is limited by the available engine power and not by the tires.

A circle will, in general, overapproximate the possible accelerations since the tires usually cannot transmit equal amounts of longitudinal and lateral force [82, p. 45]. In addition, the

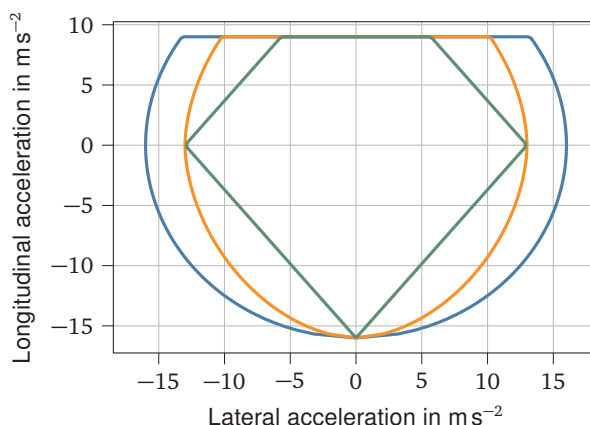


Figure 2.2: An example g-g diagram showing three possible shapes for modeling the acceleration relationship: circle, ellipse, and rhombus. The positive longitudinal acceleration is limited by the engine power and not by the tires.

magnitude of the wheel load transfer differs in longitudinal and lateral directions. A rhombus is on the conservative side for combined accelerations and thus leaves reserves, e.g., for controller interventions if the velocity profile is to be driven on an autonomous race car. An ellipse provides a good compromise for most applications based on these considerations. Instead of using a g-g diagram, a g-g-v diagram can be used to include a velocity dependency of the accelerations [83]. This dependency is important for race cars due to aerodynamic downforce and drag. For increased accuracy, bicycle or two-track models can be employed instead of g-g diagrams to model vehicle dynamics. They allow, for example, the more accurate consideration of wheel load transfers and the effect of wheel load degressivity, i.e., the degressive increase of the tire force potential with increasing wheel load. Furthermore, the powertrain can be modeled in detail.

The task of the quasi-steady-state solver is to calculate the velocity profile. The most common quasi-steady-state solver type is *forward-backward*. Its working principle is visualized in Figure 2.3. First, the racing line curvature is searched for local maxima. These are assumed to represent the apexes of corners where the driver uses the full lateral acceleration potential. Consequently, the maximum possible speed for these points can be derived from the vehicle dynamics model (marked 1 in the figure). Next, the solver iterates along the discretization points in forward and backward directions starting from the apexes. For every discretization point, it determines which share of the lateral acceleration potential is used to keep the car on the track and derives from the vehicle dynamics model how much longitudinal acceleration potential remains. In the forward direction, the positive, and in the backward direction, the negative longitudinal acceleration potential is used to determine the velocity at the next point (marked 2 in the figure). As soon as the forward and backward calculations intersect, the braking point between the adjacent apexes is determined (marked 3 in the figure). [79, 84, 85]

In transient modeling, the vehicle states are time-dependent. Thus, the states of previous points affect those of the following points [79, p. 33]. This kind of modeling is usually utilized in optimal control and model predictive control approaches. The optimization determines the vehicle's control inputs to achieve the fastest possible lap time while staying within the track limits [79, p. 33]. The computational effort is much higher than for the other solver types. For example, computing the minimum-time trajectory with a two-track model based on an optimal control approach takes 151 s [86], whereas computing the velocity profile with a quasi-steady-state approach on a race track more than twice as long on the same hardware takes only 1.2 s [9].

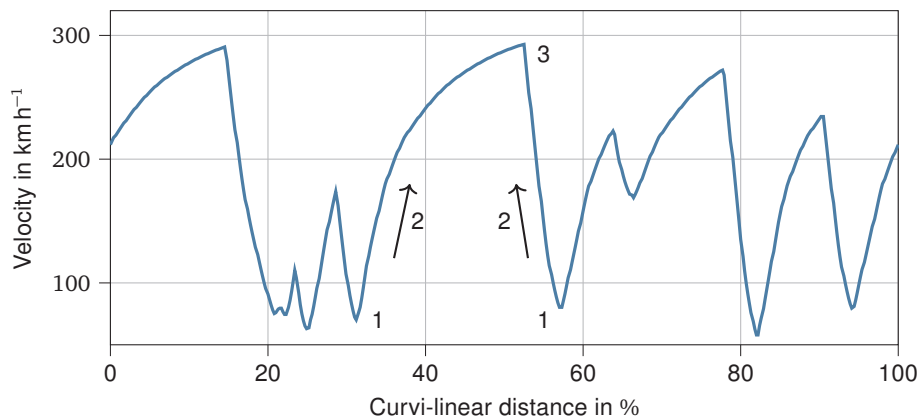


Figure 2.3: Example velocity profile to demonstrate the working principle of a forward-backward solver.

However, the results of transient modeling are more accurate [79, p. 7]. In addition, the racing line must already be available for the quasi-steady-state approach.

Literature

Having presented the basics of the three modeling types, their use in publications is summarized. The quasi-steady-state-based calculation of lap times is a widely used technique in motorsport [38, 83, 87–89]. Brayshaw et al. [85] find that a quasi-steady-state solver and a transient seven-degrees-of-freedom model show similar sensitivities to setup changes. A similar conclusion is drawn by Kelly [90]. Looking for optimal control and model predictive control approaches, Casanova et al. [91], and Tavernini et al. [92] use optimal control to study the influence of different car properties on the time required to perform individual driving maneuvers. The use of optimal control specifically for LTS was first investigated by Casanova [93] and Kelly [90]. Kelly [90] increases the drivability of the result by adding stability criteria to the objective function. Timings et al. [94] pursue a similar goal. In addition to model predictive control, they introduce a compensatory controller to take into account that real drivers cannot perfectly utilize the car's maximum potential. Consequently, it is more likely that the simulated lap time can be achieved in reality. Völkl [80, 95] combines transient modeling and steady-state solver by outsourcing the calculations of the transient states into a separate model. Veneri et al. [96] follow a similar approach by outsourcing vehicle complexity to a quasi-steady-state model that provides a g-g diagram for the optimal control problem. In addition to these publications, most of the model-based optimization approaches listed in Section 2.1 could also be used for LTS [60–66, 70, 71]. Some of them allow further investigations of different influences on lap times, for example, through a thermodynamic tire model [71] or friction coefficient differences along the track [60]. The effects of a thermodynamic tire model on lap times are also analyzed by Kelly et al. [97], and West et al. [98]. While the former publication only considers the temperature influence on the coefficient of friction, the latter also considers its influence on tire wear.

In addition to standard methods, special requirements have also been investigated over the years. Most of the LTS assume a flat track. However, some race tracks have banked corners or steep inclines and declines. For such cases, Lot et al. [70] as well as Perantoni and Limebeer [99, 100] present how to consider information on elevation and banking angle by using a three-dimensional representation of the track in lap time minimization problems. LTS can not only be used to calculate velocity profiles but also to calculate the fuel or energy consumption. Therefore, Limebeer et al. [101] demonstrate a method to obtain the time-optimal energy management strategy. Related to this problem, Ebbesen et al. [102] improve the computational speed at the

price of lower accuracy. Salazar et al. [103] depict a control scheme to follow an offline computed energy management strategy online on the car. Liu et al. [104] investigate how energy and thermal constraints can be considered for energy management in FE. Herrmann et al. [105] describe a velocity planner that minimizes the average lap time of electric race cars, taking into account a limited amount of energy as well as powertrain limitations. It is based on a sequential quadratic programming formulation that is continuously solved while driving and can thus react to deviations in the course of the race, for example, overtaking maneuvers. Optimal control for LTS for motorcycles instead of cars was also investigated [68, 106].

2.3 Race Simulation

The section on RS is divided. First, the fundamentals of RS are presented. Subsequently, the modeling of probabilistic effects is discussed more specifically.

2.3.1 Race Simulation Fundamentals

Thematic Background

In contrast to LTS, RS simulate not only a single lap but an entire race. Consequently, long-term effects, such as tire degradation and mass reduction due to burnt fuel, are no longer negligible, as is normally the case with LTS. Also, the interactions between drivers must be taken into account. The exact physical modeling of most effects would be too complex due to the multitude of influences and would increase computation times so much that a reasonable application of the RS is hardly possible. One example is the degradation behavior of racing tires, which depends not only on the thermodynamics of the tire, the brakes, and the track but also on the forces transmitted and the slip that occurs. Therefore, the concepts available in the literature are mostly based on empirical models that allow fast computing times and simple parameterization.

Literature

Two sources do not deal with a holistic RS, but pick out specific effects. McLaren [107] provides an example of how to calculate the influences of reduced fuel mass and time losses due to pit stops. Farroni et al. [108] investigate tire degradation on a microscopic level by analyzing the influences of temperature and wear on the coefficient of friction.

The little literature available on holistic RS can be grouped into two approaches: a segment-wise discretization [109] and a lap-wise discretization [110–112]. For the segment-wise approach, Bekker et al. [109] divide the race track into small segments, every of which is responsible for a defined fraction of lap time and fuel consumption. The segments permit or forbid overtaking maneuvers depending on their location (i.e., in corners or on straights). Using driver-specific base lap times and considering effects such as burnt fuel mass and aerodynamic losses when following closely after another driver allows calculating the time each driver spends in a segment. Overtaking maneuvers are executed if the attacker is fast enough to overtake the car ahead before reaching the end of the segment. Pit stops are modeled by adding a time part for the driving along the pit lane and one for the standstill. The RS presented by Phillips [110] is based on a lap-wise discretization, i.e., it simulates lap after lap. In every lap, the driver-specific lap times are determined by adding a base lap time (containing driver and car performance), a time part depending on the fuel mass, and one depending on the tire age. Additional time parts are

added to consider grid positions and start from a standstill in the first lap. Summing up all lap times up to the end of a specified lap results in the race time of that lap, compare Equation 1.4. The calculated race times of adjacent drivers are compared to check whether the rear driver is fast enough to overtake. If his race time advantage exceeds a threshold, the maneuver is successful. Otherwise, a minimum distance is established between the drivers. If a driver is much faster than the drivers ahead, he can also overtake multiple drivers within a lap. Salminen's [111] and Sulsters' [112] approaches are similar to that by Phillips [110]. However, Sulsters [112] simplifies the depth of some effects. For example, the pit stop time loss is not track-specific, and only a single car can be overtaken per driver and lap.

2.3.2 Probabilistic Influences in Race Simulations

Thematic Background

Real races are heavily affected by probabilistic influences, e.g., FCY phases. In RS, such influences are typically evaluated using Monte Carlo simulations [10, p. 86]. In Monte Carlo simulations, the random variables included in the models are sampled based on probability distributions. This method allows drawing conclusions about a quantity of interest after many trials have been conducted [113, p. vii]. In the present case, the quantity of interest is the distribution of rank positions at the end of the race. For this distribution to be meaningful, the probabilistic influences on the race must be modeled realistically.

Literature

Table 2.1 contains a qualitative evaluation of the probabilistic effects in RS that were considered in the literature.

Table 2.1: Overview and qualitative evaluation of the modeled probabilistic effects in race simulations, as published in [12]. The more the circle is filled with black, the more detailed the effect was modeled.

Modeled effect	Bekker et al. [109]	Phillips [110]	Salminen [111]	Sulsters [112]
Starting performance				
Variability of lap time				
Variability of pit stop duration				
Accidents and failures				
Damaged car				
Full-course yellow phases				

Starting performance comprises the effect that some drivers are, on average, better starters than others. This depends, for example, on how quickly a driver reacts to the green light and how well he operates the clutch. *Variability of lap time* and *variability of pit stop duration* take into account that laps, as well as pit stops, cannot be perfectly repeated. *Accidents and failures* both result in a retirement of the affected driver(s). However, accidents mostly cause an SC deployment, whereas failures more often result in a VSC phase, as drivers can still drive the car off the track. The modeling of these phases in the simulation is summarized under the item *full-course yellow phases*. Salminen [111] furthermore takes into account that accidents and failures do not necessarily lead to retirement but can also result in a *damaged car* that continues the race with a slow lap time.

Bekker et al. [109] include simple models for starting performance and car failures. For the former, they use a discrete empirical distribution with specific probabilities for each driver-car combination, yielding a finite number of positions that the driver gains or loses. Failures are modeled with a uniformly distributed probability per lap, with no distinction between possible causes. It is also mentioned that the pit stop duration is varied, but this is not further explained. Phillips [110] considers most of the relevant effects. He models lap time and pit stop duration variability with driver-specific normal distributions and log-logistic distributions, respectively, resembling real-world behavior. Salminen [111] is similar to Phillips [110] in general. He neglects the pit stop duration variability but adds a model to consider damaged cars. In comparison to the two papers mentioned before, Sulsters [112] focuses on the modeling of accidents and failures. She applies Bayesian inference to be able to assign a retirement probability to drivers without failures or accidents in the database.

2.4 Automation of Race Strategy Decisions

Thematic Background

In the context of RS, it is of interest how race strategy decisions are made and how they can be automated. The investigation of race strategy decisions is part of the field of sports analytics. Analyzing sports competitions in a retrospective is of great interest to coaches and fans, for example, as it can help to improve the effectiveness of training and provide fans with deeper insights into the sport. However, predicting the results of future events makes up the larger part of the literature. These are mostly two-class (win, lose) or three-class (win, lose, draw) classification problems depending on the sport. Such information is of value for bookmakers and betters, for example. Most approaches in the literature are based on machine learning techniques, e.g., decision trees, artificial neural networks, and support vector machines. They allow identifying relations and patterns in large amounts of data that are difficult to capture with other approaches [114, p. 5]. For a deeper introduction to machine learning techniques, Géron [114] can be recommended. The following presentation of the literature coverage focuses on the thematic background, i.e., sports analytics.

Literature

The literature deals with three application purposes: analyses that are performed before an event (e.g., result prediction), after an event (e.g., statistics), and during an event (e.g., as decision support). Most publications belong to the former two categories. Studies can be found on (American) football [115–118], greyhound racing [119–121], horse racing [122–124], soccer [125–129], swimming [130], basketball [131], hurdle racing [132], javelin throwing [133], rugby [134], and yacht racing [135]. Some research efforts have also been made to predict results for more than one sport using the same methodology [136, 137]. In motorsport, most studies are related to result prediction for the American NASCAR racing series [138–141]. According to Pfitzner et al. [139], the result position of a NASCAR driver correlates with several features such as car speed, qualifying speed, and pole position. Allender [141] finds that driver experience and starting position are the most significant predictors for result position. Another interesting result is that drivers of multi-car teams tend to achieve better results than those of single-car teams in NASCAR [140]. With regard to F1, Stoppels [142] and Stergioudis [143] have their focus on predicting race results with machine learning methods. Especially Stergioudis [143] investigates lots of possible features in the three categories driver features (e.g., qualifying position, races

finished), constructors features (e.g., constructors' championship position, times retired), and other features (e.g., circuit name, average overtakes per race).

The category of in-event analyses is of more interest for automating race strategy decisions. Gartheeban et al. [144, 145] investigate how machine learning methods can be used to decide when to change the pitcher in baseball. Bailey et al. [146], and Sankaranarayanan et al. [147] work on result prediction in cricket. Their methods allow predictions to be made during an event based on the course of the competition. The prediction of the strategies of opponent players in computer games is examined by Weber et al. [148]. Tulabandhula et al. [149] predict the change in position during a stint in NASCAR races based on whether none, two, or four tires were changed in the preceding pit stop. Many features are considered as an input for the prediction, e.g., the current position, the rate of change in position, and the performance of the driver's neighborhood. The thesis of Choo [150] is built on the results of Tulabandhula et al. [149]. Liu et al. [151] published an approach for the automated determination of the electric race strategy in FE races. It is about making optimal use of the available energy throughout the race and reacting to unexpected race events. They use artificial neural networks to predict the car's performance and Monte Carlo tree search to make the decisions. Aversa et al. [152] analyze problems in Ferrari's decision support system, which led to bad decisions in the final race of the 2010 season.

In addition to the publications listed, two sources are in principle relevant to this thesis but whose implementation is unclear. Therefore, they cannot be used scientifically but should be mentioned for completeness. For several seasons, Amazon [153] has been producing graphics that are displayed live during a race and predict, for example, the probability of overtaking in a battle for position. Unfortunately, details about the underlying machine learning models have not been made public. Maiza [154] published an online article about the possible use of a reinforcement learning agent to support race strategy decisions in F1. However, the article lacks details about the implementation and parameterization of the underlying race simulation, the exact network architecture, the pre-processing, and the training process. Since the code has not been published, there is no way to answer these points based on the implementation. In addition, according to the article, the agent was only applied to a single race, leaving the transferability to other races unclear.

2.5 Derivation of the Research Questions

Based on the state of the art presented in the previous sections and the research goal outlined in Section 1.5, the following research questions are posed for this thesis:

1. *How should a race simulation be designed to conduct realistic race strategy studies?*
2. *How can the necessary parameters for a race simulation be robustly determined with little knowledge of the exact vehicle, driver, and track characteristics?*
3. *How can the race strategies of opposing drivers be determined automatically in a race simulation so that own race strategy decisions can be evaluated objectively?*

In the following paragraphs, the scope of these three research questions is explained in more detail.

Research Question 1

To be able to evaluate the effects of race strategy decisions on the race result, a comprehensive RS is needed. Such a simulation allows an objective preparation of different strategies before a race and supports decisions that must be made quickly during a race. Even after a race, it can be used to evaluate alternative strategies and learn for future races. Due to the good compromise between accuracy and computational speed, most available literature sources simulate circuit races based on empirically motivated models and lap-wise discretization. The relevant effects for the deterministic part of a RS are already well modeled in the literature so that real races can be simulated with sufficient accuracy. However, the modeling of probabilistic influences on a race leaves room for improvement. At first, the existing approaches from the literature must be combined into a holistic model. Afterward, new and more accurate implementations of various effects must be developed to be able to evaluate race strategies realistically. These include, in particular, the modeling of the starting performance as well as the modeling of FCY phases. When implementing the RS, the computational effort must always be taken into account. Since the evaluation of probabilistic influences is typically based on Monte Carlo simulations, fast computing times are crucial.

Research Question 2

A robust and largely automated determination of the parameters for the RS is of great importance for the toolchain. From a racing team's internal perspective, most of these parameters can already be well determined before a race using real-world timing data from free practice and qualifying sessions. This is because, at least for the team's cars, many influences on the lap times are known, e.g., the vehicle setup, the engine settings, and the fuel mass. Based on these data, conclusions can, in turn, be drawn about the opposing cars. Furthermore, a team knows all its vehicle parameters so that detailed lap time simulations can be performed to determine parameters for the RS.

In contrast, determining the parameters is much more difficult from an external point of view. Timing data is mostly only available from the race itself so that for simulations before a race, only existing parameter sets from previous races can be used, which must be updated using an LTS. Since the vehicle parameters are also unknown, the LTS must be designed in such a way that it has a low parameterization effort but at the same time represents the current state of the art in motorsport. For example, both hybrid and electric powertrains and DRS should be available. Furthermore, the computational effort should be low so that sensitivity analyses can be performed quickly. The literature research shows that these requirements have not yet been met because the models are either detailed and consequently computationally expensive and difficult to parameterize or somewhat older so that they do not represent today's state of the art. After a race, the timing data can be used to create a suitable parameter set for the RS using automated data processing. Since the parameter determination has received little attention in the literature related to RS so far, suitable procedures must be developed. In this context, particular emphasis must be placed on the robustness of the process since the data are subject to a variety of unknown influences in the race. Thus, falsified data points and outliers must be reliably detected and removed, but also missing data points must be estimated and added if possible, e.g., if a driver did not run each tire compound in a race.

Research Question 3

In general, the user must specify the race strategies of all race participants as an input to a RS. This is not ideal because these strategies are not adjusted when random events occur in the simulated race, for example, a FCY phase. Consequently, the simulation results are not entirely realistic in such cases. Furthermore, it complicates the handling for the user when he has to predict the strategies of all his opponents. To improve on these points and achieve results that allow an objective evaluation of race strategy decisions, a way must be found to determine the race strategies of opponents automatically and live during a simulated race. The reaction to every driver's race situation is particularly important in this context. The literature research shows that this use case has not yet been investigated. However, data-based approaches are primarily used for similar applications. Machine learning methods appear to be particularly suitable for capturing the complex cause-effect relations of such decisions. For simplified relations, optimization-based methods can also be considered.

3 Methodology

In the following, the approach is presented based on which the goal defined in Section 1.5 is to be achieved and the research questions defined in Section 2.5 are to be answered. In addition, the focus of this thesis is defined at the end of the chapter.

3.1 Approach

Figure 3.1 provides an overview of the approach followed in this thesis. As can be seen, in addition to two databases, it consists of four simulation tools, all of which are explained in more detail in the following paragraphs.

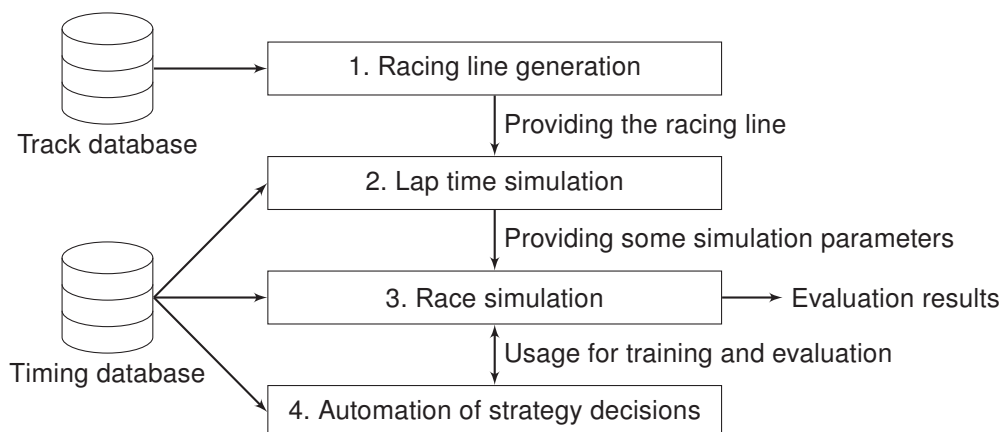


Figure 3.1: Overview of the approach that allows an evaluation of the effects of race strategy decisions on a race.

Tool 1: Racing Line Generation

The goal of the racing line generation is to provide the racing line to the subsequent LTS. However, calculating a minimum-time racing line would require a detailed vehicle model with a correspondingly high parameterization effort and high computing times. Therefore, the minimum-curvature racing line will be used as a substitute. The lap times that can be achieved with this are slightly slower [60, 86] but still sufficiently accurate for most use cases.

Tool 2: Lap Time Simulation

The main task of the LTS developed in this thesis is the determination of various parameters for the subsequent RS. For example, before the race, the LTS allows an estimation of how much lap time advantage the use of the DRS gives or how much lap time disadvantage a kilogram of

fuel causes. Due to the short computing times and to keep the parameterization requirements low, an LTS is to be developed that operates based on the previously generated racing line.

Tool 3: Race Simulation

The RS is the centerpiece of this thesis. It simulates a race based on the inserted parameters and race strategies for the individual participants. The goal of these simulations is to objectively evaluate how different race strategies affect the outcome of a race. Some of the required parameters can be obtained from the previous LTS, while most of them must be fitted based on timing data, which requires a robust automated process.

Tool 4: Automation of Race Strategy Decisions

The goal for the automated making of race strategy decisions is to achieve the best possible race result for the respective driver. Thus, the software must provide a reasonable decision behavior and react to the particular race situation. It is, for example, important to counteract undercut attempts or make use of FCY phases as it is done in reality.

Track and Timing Databases

Two types of databases are required for the presented approach. First, the racing line generation needs track information to determine a racing line. Consequently, a race track database must be created. A common format is to provide the x- and y-coordinates of a track's center line in conjunction with the track width at each discretization point. Second, the validation of the LTS results, the parameterization of the RS models, the training of machine learning models, and the examination of real-world strategy decisions all require timing data. Therefore, a database that provides information on race, lap, position, driver, lap time, and fitted tire compound as they occurred in real races is needed. Furthermore, information on the start and end of FCY phases is required to determine corresponding parameters and probabilities. Finally, at least a lap-wise resolution is necessary for a sufficient informative value of the data.

Development Targets

For the development of the four tools, it must be taken into account that no team-internal data are available. Therefore, the following development targets should be balanced:

- Adequate modeling
- Low parameterization effort
- Low computational effort
- Universal applicability

Adequate modeling is an essential requirement for each simulation tool. It means that all relevant effects must be included and modeled sufficiently accurate for the intended application. At the same time, it must be considered that a high level of detail in the models usually causes disadvantages in terms of parameterization effort and computing times. A *low parameterization effort* is crucial for usability and to be able to parameterize the models at all since detailed parameters are often not open to the public. A *low computational effort* reduces hardware requirements and allows the user to obtain the results quickly (i.e., seconds, not hours). This is particularly important in the context of race strategy evaluation to be able to react quickly to unforeseen events and run millions of simulated races in preparation for a race. The *universal*

applicability aims at the fact that the different tools can be used for different racing teams, racing series, and completely different application purposes. The racing line generation, for example, is also used for the path planning of autonomous race cars.

Given these requirements, empirically motivated and data-based approaches are preferred over highly detailed models in this thesis. For the same reason, this work focuses on tools and methodology rather than specific simulation results, as these are highly dependent on the chosen parameterization.

3.2 Focus Area

This thesis focuses on F1 as an example use case for the developed tools and methods. The main reason for this decision is that for F1, in contrast to most other racing series, comparatively much data is publicly available. This is indispensable for determining the required parameters for the simulation tools. A second reason is that, as outlined in Section 1.4, F1 is not only extremely popular but also provides the necessary strategic freedom. Despite the focus on F1, the tools and methods developed in this thesis are kept general enough to be easily adapted to other circuit racing series. For example, the RS allows for refueling of cars during pit stops, although this is not necessary for F1. Another example is the implementation of various powertrain layouts in the LTS (combustion, hybrid, electric).

The regulations in F1 change throughout the seasons. Consequently, even when focusing on this racing series, it is still hard to compare the data among different seasons. Therefore, it makes sense to search for a period with comparatively stable regulations. This has been the case for the seasons since 2014. In that year, a completely new powertrain was introduced. The old V8 engines were replaced by a hybrid powertrain consisting of a small V6 engine supported by an electric machine. Consequently, this thesis concentrates on the six seasons from 2014 to 2019.

4 Results

This chapter presents the results of the research conducted to answer the research questions stated in Section 2.5. After a brief overview of the two databases, the chapter leads chronologically through the four simulation tools that were introduced in Section 3.1. Finally, the interaction of the individual simulation tools is demonstrated in a case study using an example race.

4.1 Databases

Since neither race track models nor timing data with the required content are publicly available, appropriate databases have to be set up as a first step. The two databases are presented in the following.

4.1.1 Race Track Database

As needed for racing line generation, the description of a race track consists of the x- and y-coordinates of the center line and the track widths along the track as shown in Figure 4.1.

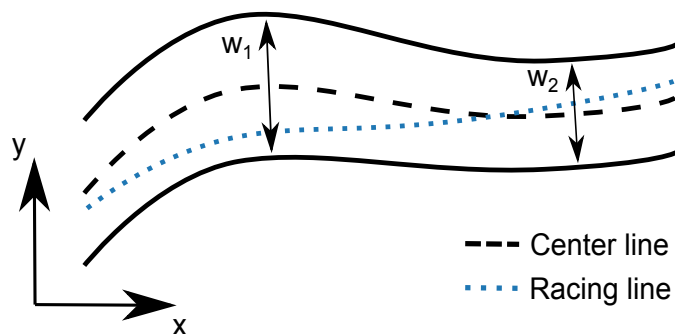


Figure 4.1: Example of a race track section with different track widths w_1 and w_2 along the track.

The center lines of regular race tracks can be obtained from the OpenStreetMap project [155]. However, some races are held on temporary city circuits, such as Monaco and Singapore. The center lines of such courses are not available, as they do not match the regular road layout. Consequently, they are not included in the race track database. The available center lines can already be used to generate racing lines if constant track widths are assumed. However, since the track widths have a considerable influence on the resulting curvature profiles and thus on the LTS results, and since they are not constant along the tracks in reality, more precise modeling is sought. For this purpose, an algorithm is developed and implemented in collaboration with de Paula Suti as part of a semester thesis [156] that extracts the track widths from satellite images from Google Maps using image processing techniques. First, a satellite image is downloaded at

the highest available zoom level for every center line point of a track. Next, the algorithm uses methods such as Hough transform [157, p. 737] and boundary detection [157, p. 814] to detect the race track boundaries in each image. A large part of the software is dedicated to handling cases where no track width can be determined, e.g., when trees obscure the track boundaries on the image or when it is unreasonable, e.g., when two track sections merge. The accuracy of the results depends primarily on the quality of the center line and satellite images. One cause of inaccuracies in the approach described is, for example, that many of the GPS points available on OpenStreetMap were recorded during normal driving on the race track and not precisely on the center line. Another one is that the recording frequency was sometimes very low so that corners are not sampled sufficiently. [156]

The developed race track database is available on GitHub [158]. It contains 25 race tracks, which are mainly driven in F1 and DTM.

4.1.2 Timing Database

Second, a timing database is created that contains the following groups of data:

- Drivers
- FCY phases
- Laps
- Qualifyings
- Races
- Retirements
- Starterfields

The database covers the 121 races that took place in the F1 seasons from 2014 to 2019. Every driver has a unique entry in the *drivers* table. The *FCY phases* table contains one entry for each VSC and SC phase that has been active over the years. The start and end of each phase are available in terms of race time and race progress. The *laps* table contains position, lap time, race time, gap, interval, tire compound, tire age, pit stop information, and driver-specific FCY phase information for all 131 527 laps. An example excerpt of the *laps* table is shown in Table 4.1. It can be seen that for the tire compound data, the designations A1 to A7 are used according to Table 1.2 to be able to compare them across the seasons. The tire ages at the end of the first lap are not equal to one because the tires were already used in qualifying. The last three columns indicate that a VSC phase was deployed towards the end of the first lap, which lasted at least until the end of the lap. The start and end of a FCY phase are calculated in dependence on the estimated lap progress of the respective driver.

The *qualifyings* table gives access to the qualifying lap times and top speeds of all qualifying sessions. The *races* table provides location, date, available tire compounds, the number of planned and driven laps, and comments on any extraordinary events during a race. The *retirements* table stores how many accidents and failures a driver suffered in each season. The *starterfields* table contains all race-specific information. This includes the assignment of each driver to a team, the engine manufacturer of the team, the starting grid position, the finishing status (i.e., finished, did not finish, disqualified), the result position, the number of completed laps, and the top speed during the race.

Table 4.1: Excerpt of the *laps* table showing the race state for positions one to five at the end of the first lap of the 2019 Shanghai Grand Prix. Some columns included in the database [159] are not listed here due to space constraints.

Race ID	Lap number	Position	Driver ID	Lap time	Race time	Gap	Interval	Compound	Tire age	Pit stop duration	VSC start	VSC end	VSC age
103	1	1	1	106.407 s	106.407 s	0.0 s	0.0 s	A4	3 laps	–	0.818	1.0	0.182 laps
103	1	2	15	108.391 s	108.391 s	1.984 s	1.984 s	A4	3 laps	–	0.803	1.0	0.197 laps
103	1	3	40	110.061 s	110.061 s	3.654 s	1.670 s	A4	3 laps	–	0.790	1.0	0.210 laps
103	1	4	12	110.733 s	110.733 s	4.326 s	0.672 s	A4	3 laps	–	0.786	1.0	0.214 laps
103	1	5	27	112.744 s	112.744 s	6.337 s	2.011 s	A4	3 laps	–	0.772	1.0	0.228 laps

The primary source for the timing database is the Ergast API [160]. Multiple checks are performed to detect inconsistencies when importing the relevant data into the timing database. Thus, some minor errors could be discovered and fixed, e.g., in starting grid positions and result rank positions. In addition, access to information such as race times, gaps, and intervals is simplified by writing it directly into the database, allowing it to be retrieved without further calculations during later analyses.

Unfortunately, the Ergast API does not contain all the information necessary to enable the analyses performed in this thesis. In particular, data on FCY phases and tire compounds driven are missing. Since no machine-readable sources can be found for either, the necessary data must be obtained manually. For the FCY phases, the data from the Ergast API is first used to identify laps in which the majority of the driver field suddenly shows increased lap times. Depending on the magnitude and duration of a lap time increase, it can already be estimated whether it is caused by a VSC or SC phase. All suspicious cases are then manually checked using recordings of the races on F1 TV [161]. Thereby, the type of each FCY phase is determined together with the respective start and end race time (i.e., the time that elapsed since the start of the respective race) and entered into the database. To be able to enter the tire compounds driven in each lap into the database, graphics are used that Pirelli makes available on Twitter [162] after each race. These graphics show the stint lengths and the tire compounds driven for all race participants. After manual extraction of the graphics data, various consistency checks occur before they are added to the database, e.g., whether the Pirelli data matches the Ergast API data in pit stop laps. In case of inconsistencies, the data is manually checked and corrected using F1 TV [161].

SQLite was chosen as the database system for the timing database because it combines the efficiency and performance of an SQL database with ease of use. The database is available on GitHub [159].

4.2 Racing Line Generation

This section summarizes the work carried out on racing line generation that has been published in [86]. All related code is available on GitHub [163].

Summary of the Paper

The publication presents the trajectory planning and control parts of an autonomous driving software stack developed for the Roborace competition, in which a team from the Technical University of Munich participated in 2018 and 2019. The software is intended for operation on an autonomous race car and can plan and follow a trajectory at the vehicle dynamics limits. In the presented concept, trajectory planning is divided into two parts. At first, a globally optimal racing trajectory is computed offline before the race. Then, during the race, the precomputed trajectory is used as a reference for the local trajectory planner. The local trajectory planner plans the target trajectories for a time horizon of a few seconds, taking into account the dynamic environment on the race track. Details on the functionality of the final version of the local trajectory planner can be found in [164].

The global trajectory planner is based on the minimum-curvature optimization introduced by Braghin et al. [38] and a forward-backward solver for the velocity profile calculation. The basic idea of Braghin's approach is to formulate a quadratic programming optimization problem to minimize the sum of the discretized curvature values along the race track. The solver, therefore, shifts the racing line points along the center line normal vectors, i.e., to the left and right of the center line. Due to the efficient solving of quadratic programming problems, it is desirable to minimize the sum of the discrete squared curvature κ_i^2 at each center line point i instead of κ_i itself.

The main advantages of the approach are fast computing times and the fact that no vehicle parameters need to be known except for the vehicle width due to the purely geometrical problem description. The result is not equal to the time-optimal solution but comes close to real racing lines, as simulated in [86, p. 1505f.]. Another advantageous aspect is that the minimum-curvature optimization creates smooth paths by nature. This is an important aspect when driving a real car at high speeds.

Compared to the original implementation by Braghin et al. [38], the path optimization is improved by several aspects in this work. First, the quadratic programming formulation is extended by a missing term such that the curvature in the optimization problem resembles the actual curvature. Second, upper and lower curvature boundaries are introduced such that kinematic constraints can be considered within the optimization problem. Third, an iterative invocation of the problem is introduced that significantly reduces the curvature linearization error in corners. These points improve the quality and robustness of the solver significantly.

Relation to the Research Questions

The publication is related to the second research question. It shows how a racing line close to the time-optimal line can be determined despite little knowledge of the vehicle, driver, and track parameters. This is a prerequisite for further answering the research question in the subsequent publication on LTS.

Individual Contribution

Heilmeier as the first author of the publication was responsible for the trajectory planning software in the Roborace project at the Technical University of Munich. He implemented the minimum-curvature optimization problem originally stated by Braghin et al. [38]. Based on essential contributions by his colleague Wischniewski, the two analyzed and extended the formulation of the optimization problem. Besides, Heilmeier implemented the velocity profile calculation and the

behavior state machine and contributed to the overall design of the autonomous driving software stack.

Imprint of the Paper

This is the authors' accepted manuscript of an article published as the version of record in Vehicle System Dynamics ©2020 Informa UK Limited, trading as Taylor & Francis Group, available online: <http://www.tandfonline.com/10.1080/00423114.2019.1631455>.

Minimum Curvature Trajectory Planning and Control for an Autonomous Race Car

Alexander Heilmeier^a, Alexander Wischnewski^b, Leonhard Hermansdorfer^a, Johannes Betz^a, Markus Lienkamp^a and Boris Lohmann^b

^aChair of Automotive Technology, Technical University of Munich, Munich, Germany;

^bChair of Automatic Control, Technical University of Munich, Munich, Germany

ABSTRACT

This paper shows a software stack capable of planning a minimum curvature trajectory for an autonomous race car on the basis of an occupancy grid map and introduces a controller design that allows to follow the trajectory at the handling limits. The minimum curvature path is generated using a quadratic optimization problem (QP) formulation. The key contributions of this paper are the extension of the QP for an improved accuracy of the curvature approximation, the introduction of curvature constraints and the iterative invocation of the QP to significantly reduce linearization errors in corners. On the basis of the resulting raceline, a velocity profile is calculated using a forward-backward-solver that considers the velocity dependent longitudinal and lateral acceleration limits of the car. The advantages and disadvantages of the proposed trajectory planning approach are discussed critically with respect to practical experience from various racetracks. The software stack showed to be robust in a real world environment as it ran successfully on the Roborace DevBot during the Berlin Formula E event in May 2018. The lap time achieved was within a tenth of a second of a human driver and the car reached about 150 km/h and 80% of its acceleration limits.

KEYWORDS

trajectory planning; path planning; control; minimum curvature; autonomous driving; race car

1. Introduction

Regarding the multi-stage autonomy of vehicles based on the SAE categorization, level 5 will enable a completely self-driven vehicle without a driver. This means that all tasks, which have previously been accomplished by a driver, must now be performed using algorithms alone. This includes perceiving the environment, planning trajectories and following them.

To benchmark state-of-the-art software for autonomous cars at the physical limits of a vehicle, a team from the Chair of Automotive Technology and the Chair of Automatic Control of the Technical University of Munich (TUM) takes part in the Roborace competition. Roborace provides an electrically powered, automated level 5 race car called DevBot, which serves as a platform for practical testing of the programmed software. The TUM team developed the complete autonomous driving pipeline for controlling the vehicle [1]. The software was publicly presented in May 2018 at the Formula E Event in Berlin [2]. On this event, the DevBot drove three autonomous

CONTACT Alexander Heilmeier. Email: alexander.heilmeier@tum.de

laps on the racetrack with a maximum velocity of 150 km/h. The lap time achieved was within a tenth of a second of an average human driver [2].

The general workflow to let the vehicle drive autonomously is as follows: Firstly, the environment around the car on the racetrack must be perceived based on walls and free spaces. We focused on the 2D-LiDAR data of the DevBot to create an occupancy grid map of the racetrack. This is done using the SLAM (Simultaneous Localization And Mapping) algorithms gmapping [3] or Google Cartographer [4]. Next, the map is processed to obtain the centerline of the track. This is the input for the trajectory generation presented in more detail below. Finally, the vehicle controller is fed with small trajectory parts with a defined time horizon while the car is driving around the track.

This paper presents the planning and control parts of the software stack allowing an autonomous race car to drive a racetrack at the handling limits. The key contributions are the theoretical formulation of the minimum curvature path optimization problem including its extension for an improved accuracy of the curvature approximation, the introduction of curvature constraints originating from a real car's steering design and the iterative invocation of the QP to significantly reduce linearization errors in corners. Furthermore we introduce several extensions to the planning algorithm to guarantee robust operation in real world application. All these have increased the quality of the solution significantly on the narrow Formula E tracks compared to state-of-the-art approaches.

The rest of the paper is structured as follows: Firstly, related work for the relevant topics is presented. Then creation of the centerline is introduced just before trajectory generation is discussed in more detail. The methodology section ends with a description of the controller design. Afterwards the results are presented and the advantages and disadvantages of the approach are discussed.

2. Related Work

The field of autonomous driving has been the focus of research for several years. Various full vehicle concepts have been proposed, e.g. during the DARPA Grand Challenge [5] and the DARPA Urban Challenge [6–12]. Both focused on rather low speed scenarios but the latter included several navigational aspects and decision-making processes in complex, dynamic environments.

Racing scenarios have not been highly investigated although they are highly relevant for autonomous driving on motorways and in the urban environment, e.g. in emergency evasion situations. An exception is an autonomous Audi TTS that is operated by Stanford University [13–15] reaching competitive performance levels on various tracks. Here, the racing problem is separated into trajectory planning and trajectory control. It was shown that their approach is capable of nearly achieving the vehicle limits. In contrast, [16,17] presents multiple optimal control strategies to solve the trajectory planning and control problem within a single algorithm. The main advantage of this approach is the physically meaningful parametrization via the nonlinear vehicle-model and the fact, that a single algorithm takes care of both problems. This allows to incorporate tire force constraints in a straight-forward way. However, we believe that integrated trajectory planning and control approaches are difficult to scale to complex scenarios due to the non-convex nature of the constraints imposed by multiple vehicles. Furthermore, these algorithms are considered to be more complex during the tuning process due to the mixture of tasks. This point of view is in line with the split of

tasks proposed by many researchers discussing the architecture for autonomous driving systems in road scenarios [18–20].

2.1. From Map to Centerline

To be able to use the information gained by LiDAR scanning of the racetrack, the resulting data must be processed in such a way that it can be used for trajectory planning. In our case, the inputs of the planning module are the racetrack’s centerline and the corresponding track widths. There are several approaches for extricating the road from LiDAR data and generating the corresponding centerline. Common approaches are Support Vector Machine Classification [21], the use of kernel density estimation [22], canny edge detection and subsequent hough transformation [23] or different applications of neural networks, e.g. [24].

The above mentioned methods work for a variety of scenarios and are able to deal with huge street networks, but sometimes fail when it comes to small scales, e.g. single roads. Therefore some other approaches have to be considered with regard to the special environment in which the centerline generation has to be applied in our project. A unique feature of many Formula E racetracks is having walls right next to the tarmac. Therefore, the drivable area can be obtained by considering the walls as track limits. Consequently, the resulting ‘free of obstacles’ area has a tube-like shape. The task of finding the centerline of such a shape is very common in clinical procedures when visualizing human blood vessels. On the one hand there are hand-tuned filters designed to respond to certain structures and on the other hand classification techniques using machine learning. [25] utilizes consecutive image-processing methods (e.g. binarization, skeletonization) and a simple classification for tracking the centerline of human blood vessels. [26,27] reformulate the centerline detection in terms of a regression problem.

2.2. Trajectory Planning for Autonomous Cars

A trajectory defines the path a moving object follows in the space-time domain. Both, path and trajectory planning methods for autonomous cars can be found in literature. As velocity profile planners such as [15,28–30] can generate the missing time domain information based on a given path, both types may be suitable for attaining our goal. Therefore, we do not distinguish between path and trajectory planning.

Survey papers [19,31] serve as an entry point to the topic as they compare different algorithms, e.g. in terms of optimality, completeness and time complexity. The solution approaches can usually be categorized into three classes [19], even though the assignment is often not clearly defined:

- Variational methods
- Graph search methods
- Incremental search methods

In variational methods, the path is usually represented by splines, whose parameters are optimized in terms of a cost function. This includes optimization techniques such as optimal control [32–36] and geometrical optimization [30].

Graph search methods discretize the possible configuration space of the car and use heuristic information to search for a cost optimal path through the graph. The best known example of this type is Dijkstra [37]. It has been improved by the A* family [38,39]. Dynamic environments with moving objects can be considered in D*

algorithms [40–42].

Incremental search methods are similar to the graph search methods but are based on random sampling of the configuration space. The samples are then connected by an algorithm. The cost optimal path to the target point therefore continues to improve the longer the algorithm runs. A widely known approach is the Rapidly-exploring Random Tree (RRT) [43]. It has been succeeded by RRT* [44] and became suitable for dynamic environments as RRT^X [45]. Another relative is EST [46].

Combined approaches are found in the context of the DARPA challenges that were mentioned above. They represent a spectrum of approaches, e.g. optimal control and lattice planners in [6] and Hybrid A* in [7].

Most of the algorithms can only be applied for low to medium lateral accelerations and at low velocities. The amount of research into high performance cars and race cars is much smaller. A major problem here is to plan a valid trajectory that meets all the constraints at high velocity and accelerations while keeping desirable smoothness properties. Several authors [16,17,47] propose to use optimal control for this purpose. [48] extend an optimal control approach by a moving horizon to reduce the computational effort of the optimal control problem. Another approach is the offline optimization of pre-planned maneuvers [49], which can then be composed online according to traffic and driving situation. In the context of minimizing lap time on the basis of optimal control, there are indirect and direct approaches. The former are based on Pontryagin’s minimum principle to find the solution, e.g. [35]. [34] is a representative of the latter category. Here, the optimal control problem is transformed into a nonlinear programming problem. [36] gives an overview of this field and compares the different approaches. Typical disadvantages of optimal control problems are high computation times or high complexity of the parametrization and implementation. [50] and [51] are based on RRT* motion planners. While the former separates the problem into path and velocity profile generation, the latter combines a sparse version of RRT* with model predictive control (MPC). [52,53] present path planning approaches for autonomous race cars based on the concatenation of circular arcs and straights. However, the description of the track imposes some limitations in this approach. The Audi TTS in [54] drives with a similar approach. Here, the pre-computed path is composed of four parts per corner: straight, entry clothoid, constant radius arc and exit clothoid. [30] uses a geometrical optimization of the path along a racetrack to minimize the curvature. [15] divides the trajectory generation into two sequential sub-problems to reduce computational effort. At first a velocity profile is generated before a convex path optimization problem is solved which minimizes the resulting path curvature while taking the vehicle’s handling limits into account.

2.3. Vehicle Control

Nowadays, lateral and longitudinal control systems are widely applied in series-production vehicles. Conceptually they are treated as independent systems using the lateral deviation from a path and the vehicle velocity as reference variables. Due to its more complex nature, the former receives more attention in research activities. In its basic form, it usually relies on an output feedback controller for a look-ahead point placed in front of the vehicle [14,55,56]. Several nonlinear modifications have been proposed in recent years, using Lyapunov [57], Exact-Linearization [58,59] or Sliding-Mode [60,61] design techniques. Another design direction is utilizing optimization based controllers [17,62]. However, their use is sensitive to model quality, numerical effects and

they impose heavy computational demands. For this reason, this design direction usually involves higher development efforts. A detailed comparison of several concepts can be found in [63].

3. Methodology

This section first introduces the architectures of the hardware used in Roborace and the software developed by the TUM team. Then the software parts are described in more detail starting with the centerline generation and ending with the controller implementation. The plots in this section mostly show the same characteristic corner combination of the Berlin Formula E track to be able to follow the progress from one step to the next. The segment is a good benchmark for the algorithms, as it includes the end of a long straight as well as corners with different narrowness.

3.1. Autonomous Driving System Components

During the Roborace competition, we used the so-called DevBot, a race car based on an LMP3 chassis [64]. It was designed as a rapid prototyping platform for autonomous driving algorithms and can therefore be driven by a human driver as well as autonomously. It provides various sensors, of which the following are relevant for this paper: Four LiDAR sensors around the front wheels, an OxtS 4002 inertial measurement unit including GPS, a Kistler SFII P optical velocity sensor and four wheelspeed sensors. Two control units can be utilized to run the algorithms: a Nvidia Drive PX2 for planning and decision tasks and a Speedgoat Mobile Target Machine for real-time control tasks. For more information about the vehicle and the holistic autonomous driving pipeline used in the car, we refer to [1].

3.2. Autonomous Driving Software Architecture

The software is divided into three main modules. The perception module generates a representation of the environment. This includes the detection of track boundaries and objects as well as tarmac recognition. These informations are handed over to the planning module, which is responsible for making decisions at long-term level (race strategy decisions and generation of a global race trajectory) and short-term level (evasion and overtaking maneuvers). It generates local trajectory snippets for the control module. The latter is aimed at tracking the local target trajectories.

The focus of the Berlin competition in 2018 was to set the best possible lap time in an environment free of opponents and obstacles. This allowed the perception and trajectory optimization parts shown in Figure 1 to be carried out offline. Three manually driven laps were used to generate an occupancy grid map of the track by fusing GPS, odometry and LiDAR measurements. This map was then post-processed. Run-off areas and curbs required manual reworking because it is almost impossible to detect them with the used LiDAR. The step also includes the generation of the centerline required for the subsequent optimization algorithm. The actual driving trajectory is obtained in a two-step procedure: first finding a minimum curvature path and then generating a suitable velocity profile that considers the vehicle's handling limits.

Figure 2 depicts the software modules which are active during the driving sessions. The Planning Unit (PU) performs two main tasks. One of them is monitoring the

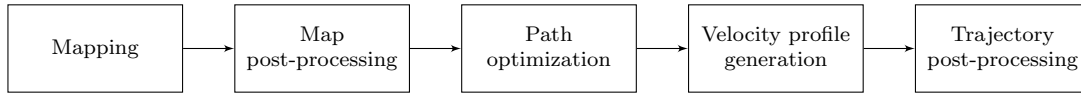


Figure 1. Process flow chart for the generation of a race trajectory (offline).

system mission (Behavior State Machine). This includes launching the car, monitoring the subprocesses, counting the number of laps and adjusting the velocity profile based on given scale factors, e.g. to introduce a tire warm-up lap. The other task is the extraction of high fidelity local trajectory parts from the global trajectory that can be used for control (Local Trajectory Generation). To cover enough distance along the track, we adapt the step size of the trajectory snippet sent to the trajectory tracking controller dependent on the current velocity.

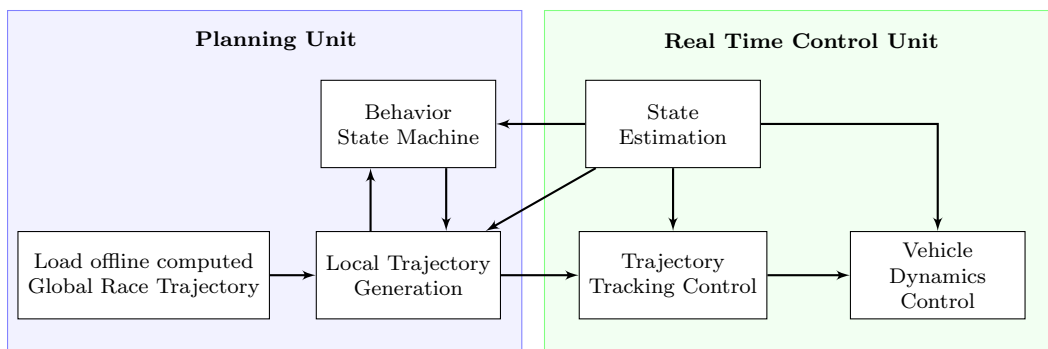


Figure 2. Autonomous driving software architecture of the TUM team (online).

A major advantage of the local trajectory concept is that it allows an extension of the functionality in future work, e.g. to implement the handling of static and dynamic objects that are not considered in the offline computed global race trajectory. To ensure safety in the event of a PU or network failure, much longer foresight emergency trajectories are generated simultaneously. These allow the Real Time Control Unit (RTC) to stop the car safely on its own.

The RTCU implements a two-degree of freedom controller for lateral path and velocity tracking. Both rely heavily on a feed forward control capturing the main vehicle characteristics. The feedback control is based on a state estimate obtained from sensor fusion. This controller interfaces the vehicle via the vehicle dynamics control module using a velocity and curvature set point interface. The latter converts the requested values into steering angles and force requests.

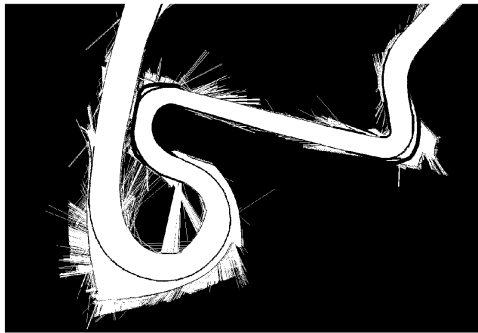
3.3. Generation of a Centerline

Since we use 2D-LiDAR data to generate the circuit map, the post-processed LiDAR data is represented as an occupancy grid map with three possible states: ‘occupied’, ‘unknown’, ‘free of obstacles’. It can be interpreted as an image, which allows the use of certain image-processing methods. In addition to the previously mentioned methods used in clinical procedures for the visualization of human blood vessels, several additional functions have been implemented. These are necessary to obtain viable results for the path optimization. The four main steps for generating the centerline and its

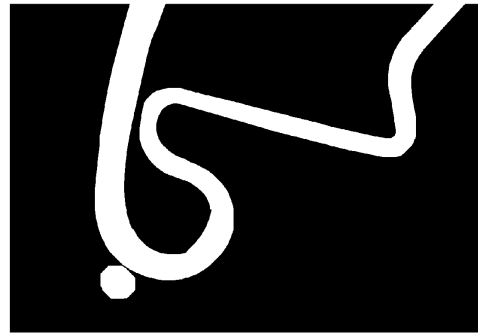
corresponding track widths are visualized in Figure 3:

- Preprocessing of the raw map data
- Smoothing and filtering of the raw map data
- Applying the Euclidean distance transform and extricating the centerline
- Smoothing of the centerline

Firstly, the original grid map is binarized. The resulting grid map consists of a single binary value for each cell (occupied (true), free (false)). The ‘unknown’ space is declared as ‘occupied’ to be on the safe side. The result is shown in Figure 3a.



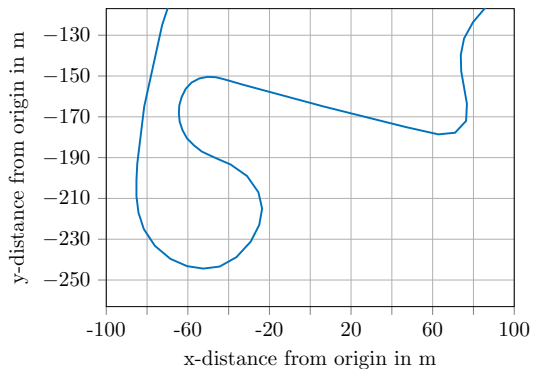
(a) Preprocessing of the raw map data



(b) Smoothing and filtering of the raw map data



(c) Applying euclidean distance transform



(d) Smoothing

Figure 3. Four main steps of centerline generation (top left to bottom right).

Secondly, single pixels or an accumulation of a small, predefined number of pixels with state ‘occupied’ are considered as ‘false positives’ and are discarded by the algorithm. In addition, small ‘free’ areas (e.g. single LiDAR beams as seen in Figure 3a), which are not relevant for the vehicle due to their small spatial extent, are closed, i.e. considered as occupied. Furthermore, the GPS data of the vehicle during mapping is used to clear the grid map of false positives. The result is a smoothed, tube-like shape that represents the drivable area of the racetrack (Figure 3b).

Thirdly, the morphological skeleton method [65] is applied to output the track boundaries. Using the euclidean distance transform [66] allows calculation of the maximum distance from every cell to its nearest track boundary cell. All distance values that are outside the actual racetrack are discarded (Figure 3c). The watershed algorithm [67] (usually used for image segmentation) is applied to detect the maximum

distance to track boundaries while considering the closed-loop nature of the centerline. The resulting centerline has an edgy, discontinuous shape depending on the track layout. These edges cause a sudden change in direction, which is difficult to handle in the subsequent path optimization algorithm.

Lastly, a Savitzky-Golay filter is used [68] to smooth the centerline obtained in the previous step. It is separately applied to the x- and y-coordinates. Finally, the algorithm outputs the xy-coordinates of the centerline and the corresponding track widths (i.e. distance to track boundaries is equal to both sides) of each centerline point (Figure 3d).

3.4. Obtaining a Minimum Curvature Path

For our application, a trajectory consists of the seven variables curvilinear distance s , coordinates x and y , heading ψ , curvature κ , velocity v_x and longitudinal acceleration a_x : $[s, x, y, \psi, \kappa, v_x, a_x]$. We had several requirements concerning the trajectory generation. Firstly, it should achieve the best possible lap time. Secondly, the implementation of the algorithm should be robust and reliable to reduce testing time. Thirdly, the calculation time should not exceed a few seconds. This would enable us to use the same approach for re-planning of the raceline online on the car in the future. Considering the different concepts presented in section 2.2, we decided to split the trajectory planning problem into path optimization and velocity profile generation. Therefore, we calculate the first five dimensions of the trajectory within this section. The velocity and longitudinal acceleration profiles are generated afterwards based on the raceline to obtain a complete trajectory.

The path optimization approach applied significantly extends the work of Braghin [30]. Kapania [15] is quite similar and has also been considered as a basis. The difference is that this approach first generates a velocity profile and then optimizes the path by solving a convex optimization problem on this basis. The process is repeated until the calculated lap time no longer improves. We opted for the approach in [30] because [15] shows a slightly large computing time of approximately 90 s (requiring an average of three iterations with 30 s each) and gives no guarantee of convergence.

The idea of the approach presented in [30] is to vary the path on the track in such a way that the globally summed quadratic curvature is minimized. This is not the minimum time solution, which is generally a compromise between minimum curvature path and shortest path [15]. However, the minimum curvature path is reasonably close to a minimum time path because it allows the highest cornering speeds at a given maximum lateral acceleration as given by

$$v_{\max} = \sqrt{\frac{a_y}{\kappa}}. \quad (1)$$

Figure 4 shows the paths and lap times t_{lap} resulting from shortest path optimization, our final implementation of the iterative minimum curvature optimization and minimum time optimization (based on [34,69]) for the Berlin track segment. All solvers consider a vehicle width of 2.0 m. The discretization step size is 3.0 m. The computation times from centerline import to trajectory output are 4 s for shortest path, 18 s for iterative minimum curvature and 151 s for minimum time optimization on a laptop computer (Intel Core i7 2.7 GHz, 16 GB RAM).

The mentioned minimum time optimization result is based on a two-track model to show the actual time optimal path. However, the velocity profile calculations for

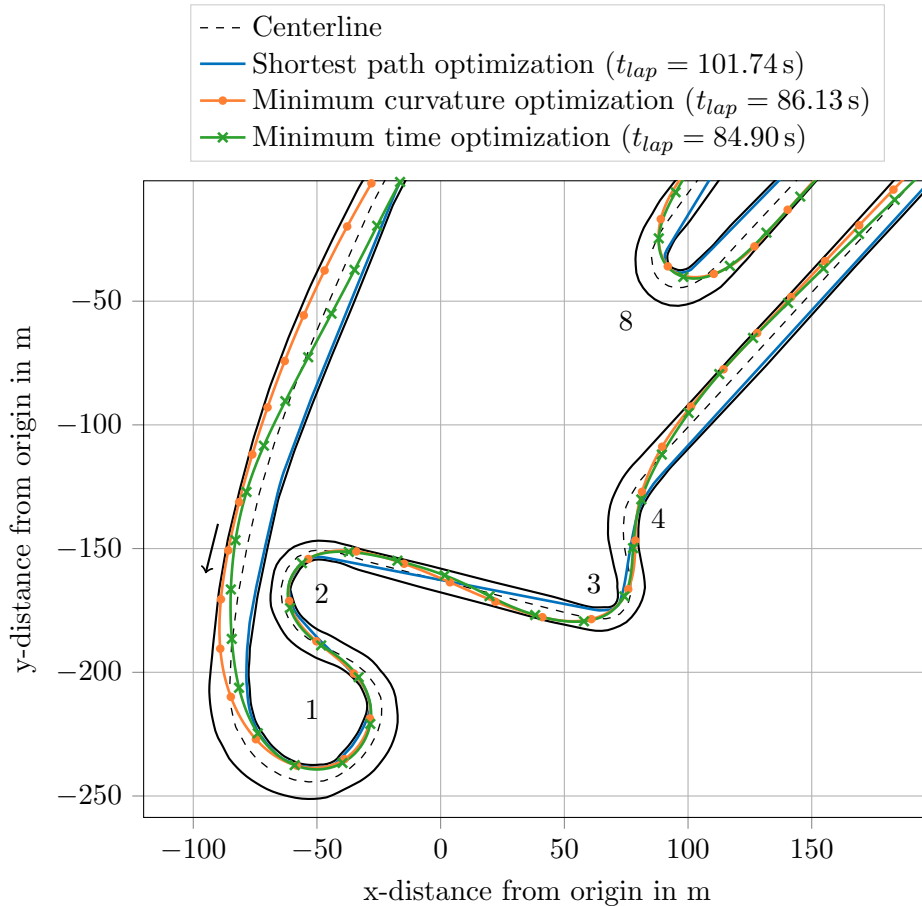


Figure 4. Comparison of the paths resulting from shortest path optimization, (iterative) minimum curvature optimization and minimum time optimization.

the shortest path and minimum curvature results are based on a point mass model. To obtain comparability in terms of the computation time, we also calculated the minimum time result based on a point mass model. In this case, the computation time is 37 s.

As can be seen, the path of the minimum curvature solution is quite close to the solution obtained in the minimum time problem within the corners. On the straights, the minimum time result stays closer to the shortest path as soon as the tires are not fully exploited. The lap time advantage is 1.4% while the computation time of the minimum curvature optimization problem is 50% smaller (comparing the point mass models). For an offline application, the longer computation time can easily be tolerated to obtain the better lap time. For online applications, however, the lower computation time of the minimum curvature problem can be a decisive argument, especially since car computers such as the Nvidia Drive PX2 often have quite slow CPUs in comparison to the Intel Core i7 in the laptop. Further advantages of the minimum curvature path optimization are a very low parametrization effort and the independence from vehicle dynamics parameters as well as the fact that the resulting raceline shows a very smooth curvature profile. All this eases the implementation on a real car significantly. The shortest path solution is primarily interesting for cars that are limited by a low top speed instead of their lateral acceleration capabilities.

3.4.1. Description of the Optimization Problem

The approach is formulated as a QP that can be solved quickly and robustly. The inputs for the optimization problem are the centerline points and the corresponding track widths obtained in section 3.3. Within the optimization problem, we switch from centerline definition to reference line definition. The difference is that the reference line may have different track widths on the left and right sides of the line, i.e. it does not have to be in the middle of the track.

The optimization problem is based on a variation of the raceline points r along the reference line normals, i.e. along the track widths, to minimize the curvature. The notation of the i th raceline point reads

$$\vec{r}_i = \vec{p}_i + \alpha_i \vec{n}_i \quad (2)$$

where $\vec{p}_i = [x_i \ y_i]^T$ is the reference line point and \vec{n}_i the unit length normal vector. The independent parameter α_i is used to move the point \vec{r}_i along the normal vector between the track boundaries as visualized in Figure 5. The track boundaries are transformed into boundaries on α_i given by combination of the left and right track widths $w_{\text{tr,left},i}$ and $w_{\text{tr,right},i}$ and vehicle width w_V as

$$\alpha_i \in \left[-w_{\text{tr,left},i} + \frac{w_{\text{veh}}}{2}, w_{\text{tr,right},i} - \frac{w_{\text{veh}}}{2} \right]. \quad (3)$$

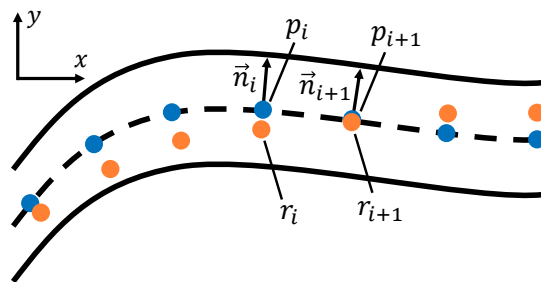


Figure 5. Point notation for the reference line points p and the raceline points r . The optimization parameters $\vec{\alpha}$ move the raceline points along the corresponding normal vectors \vec{n} .

The raceline is defined by third order spline interpolations of the points r in x and y coordinates. This enables us to calculate first and second order derivatives explicitly from the spline representation. In the remainder of the section only the x -part is described for the sake of brevity. The y -part follows by straightforward extension of the proposed concepts. The location of a third order spline and its first and second derivative with respect to t are as follows:

$$x_i(t) = a_i + b_i t + c_i t^2 + d_i t^3, \quad (4)$$

$$x'_i(t) = b_i + 2c_i t + 3d_i t^2, \quad (5)$$

$$x''_i(t) = 2c_i + 6d_i t \quad \text{and} \quad (6)$$

$$t_i(s) = \frac{s - s_{i0}}{\Delta s_i} \quad (7)$$

where t is the normalized curvilinear parameter along one spline segment starting

at the distance s_{i0} . Therefore, $0 \leq t_i \leq 1$ holds. The spline interpolation requires that consecutive splines share their respective beginning and endpoint as well as their first and second derivative at these points. The latter fact ensures smooth curvature along the interpolation. Complying with these constraints, the spline parameters a_i , b_i , c_i and d_i are obtained from the solution of a linear equation system of the form $Az = b$. The matrix A and the vector b formalize the above constraints. Since A is full rank by construction of the problem, it can be inverted and leads to a unique solution for the spline coefficients contained in z .

For the minimum curvature optimization, we want to minimize the discrete squared curvature κ_i^2 of the splines summed along the raceline with N points/splines as follows:

$$\begin{aligned} & \underset{[\alpha_1 \dots \alpha_N]}{\text{minimize}} && \sum_{i=1}^N \kappa_i^2(t) \\ & \text{subject to} && \alpha_i \in [\alpha_{i,\min}, \alpha_{i,\max}] \quad \forall 1 \leq i \leq N. \end{aligned} \tag{8}$$

The L2 norm is chosen because of its desirable properties in terms of numerical optimization and elegant formulation of the resulting optimization problem. From the spline representation it follows that it is most convenient and efficient to evaluate the splines at $t = 0$. These are chosen as discretization points evaluated in the problem above. Fixing $t = 0$ and dropping the spline parameter t for notational convenience, the curvature at the discrete evaluation points can be expressed as [70, p. 373]:

$$\kappa_i = \frac{x'_i y''_i - y'_i x''_i}{(x_i'^2 + y_i'^2)^{\frac{3}{2}}} \quad \text{and} \tag{9}$$

$$\kappa_i^2 = \frac{x_i'^2 y_i''^2 - 2x_i' x_i'' y_i' y_i'' + y_i'^2 x_i''^2}{(x_i'^2 + y_i'^2)^3}. \tag{10}$$

[30] uses a simplified curvature definition omitting the central part $2x'_i x_i'' y_i' y_i''$. This leads to a suboptimal solution in terms of the exact minimum curvature problem. In contrast, our formulation of the optimization problem is based on the exact curvature definition, which results in improved optimization results and allows us to introduce curvature constraints later.

Inserting (10) into the optimization problem stated in (8), we obtain

$$\begin{aligned} & \underset{[\alpha_1 \dots \alpha_N]}{\text{minimize}} && \vec{x}''^T P_{xx} \vec{x}'' + \vec{y}''^T P_{xy} \vec{x}'' + \vec{y}''^T P_{yy} \vec{y}'' \\ & \text{subject to} && \alpha_i \in [\alpha_{i,\min}, \alpha_{i,\max}] \quad \forall 1 \leq i \leq N \end{aligned} \tag{11}$$

by defining $\vec{x}'' = [x''_1 \dots x''_N]^T$ and $\vec{y}'' = [y''_1 \dots y''_N]^T$ as the vector notation for the second derivative of the coordinates at each discretization point.

The matrices P_{xx} , P_{xy} and P_{yy} can be written as:

$$P_{xx} = \begin{bmatrix} \frac{y_1'^2}{(x_1'^2+y_1'^2)^3} & 0 & \cdots & 0 \\ 0 & \frac{y_2'^2}{(x_2'^2+y_2'^2)^3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{y_N'^2}{(x_N'^2+y_N'^2)^3} \end{bmatrix}, \quad (12)$$

$$P_{xy} = \begin{bmatrix} \frac{-2x_1'y_1'}{(x_1'^2+y_1'^2)^3} & 0 & \cdots & 0 \\ 0 & \frac{-2x_2'y_2'}{(x_2'^2+y_2'^2)^3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{-2x_N'y_N'}{(x_N'^2+y_N'^2)^3} \end{bmatrix} \text{ and} \quad (13)$$

$$P_{yy} = \begin{bmatrix} \frac{x_1'^2}{(x_1'^2+y_1'^2)^3} & 0 & \cdots & 0 \\ 0 & \frac{x_2'^2}{(x_2'^2+y_2'^2)^3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{x_N'^2}{(x_N'^2+y_N'^2)^3} \end{bmatrix}. \quad (14)$$

The above structure already suggests to treat the problem as a QP, under the additional assumption that the matrices P_{xx} , P_{xy} and P_{yy} are constant. This can be seen as approximately true because x' and y' are approximately constant since path heading only changes slightly compared to the reference line as long as it is tightly constrained by the inner and outer track boundaries. Therefore, we obtain x' and y' from the case $\alpha = 0$ which corresponds to the reference line. The process can be viewed as linearization of the optimization problem along the reference line.

We now need to express the second derivatives x'' and y'' in terms of the optimization parameters $\vec{\alpha}$. For $t = 0$ x'' evaluates to:

$$x_i''(t = 0) = 2 c_i. \quad (15)$$

The spline coefficients that resemble the second derivative at $t = 0$ can be extracted from the solution vector z , which can be expressed in terms of the inverse of the linear

equation system matrix A and the constant vector b , via the extraction matrix $A_{\text{ex},c}$:

$$\begin{bmatrix} x_1'' \\ x_2'' \\ \vdots \\ x_N'' \end{bmatrix} = 2 A_{\text{ex},c} A^{-1} \left(\underbrace{\begin{bmatrix} p_{1,x} \\ p_{2,x} \\ 0 \\ 0 \\ p_{2,x} \\ p_{3,x} \\ 0 \\ 0 \\ \vdots \\ 0 \\ p_{N,x} \\ p_{1,x} \\ 0 \\ 0 \end{bmatrix}}_{\vec{q}_x} + \underbrace{\begin{bmatrix} n_{1,x} & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & n_{2,x} & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & n_{2,x} & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & n_{3,x} & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & n_{N,x} \\ n_{1,x} & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}}_{M_x} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} \right). \quad (16)$$

The first derivatives x'_i and y'_i are calculated in a similar manner. Reformulating (16) in vector notation gives us

$$\vec{x}'' = T_c \vec{q}_x + T_{n,x} \vec{\alpha}, \quad (17)$$

where $T_c = 2 A_{\text{ex},c} A^{-1}$ and $T_{n,x} = 2 A_{\text{ex},c} A^{-1} M_x$. T_c is equal for the x- and y-coordinate splines and therefore not distinguished. Inserting (17) and the corresponding version for y in (11) we finally obtain

$$\begin{aligned} \underset{[\alpha_1 \dots \alpha_N]}{\text{minimize}} \quad & \vec{\alpha}^T (H_x + H_{xy} + H_y) \vec{\alpha} + (f_x + f_{xy} + f_y)^T \vec{\alpha} + \text{const}, \\ \text{subject to} \quad & \alpha_i \in [\alpha_{i,\min}, \alpha_{i,\max}] \quad \forall 1 \leq i \leq N \end{aligned} \quad (18)$$

where

$$\begin{aligned} H_x &= T_{n,x}^T P_{xx} T_{n,x}, \\ H_{xy} &= T_{n,y}^T P_{xy} T_{n,x}, \\ H_y &= T_{n,y}^T P_{yy} T_{n,y}, \\ f_x &= 2 T_{n,x}^T P_{xx}^T T_c \vec{q}_x, \\ f_{xy} &= T_{n,y}^T P_{xy}^T T_c \vec{q}_x + T_{n,x}^T P_{xy}^T T_c \vec{q}_y, \\ f_y &= 2 T_{n,y}^T P_{yy}^T T_c \vec{q}_y \quad \text{and} \\ \text{const} &= \vec{q}_x^T T_c^T P_{xx} T_c \vec{q}_x + \vec{q}_y^T T_c^T P_{xy} T_c \vec{q}_x + \vec{q}_y^T T_c^T P_{yy} T_c \vec{q}_y. \end{aligned}$$

Neglecting the constant term const , this can be reformulated in terms of a standard

QP problem:

$$\begin{aligned} & \underset{[\alpha_1 \dots \alpha_N]}{\text{minimize}} && \frac{1}{2} \vec{\alpha}^T H \vec{\alpha} + f^T \vec{\alpha} \\ & \text{subject to} && E \vec{\alpha} \leq k. \end{aligned} \quad (19)$$

3.4.2. Adaption of the Optimization Problem

The presented implementation of the optimization problem is quite sensitive to a noisy reference line, for example due to small jumps along a straight originating in map discretization. Therefore, the reference line is preprocessed in four steps before it is handed over to the optimization problem, see Figure 6. Firstly, the original reference line is linearly interpolated to a small step size. Based on this, a spline approximation is calculated to remove the noise. Afterwards, the track widths must be corrected due to the slight displacement of the spline approximation compared to the reference line. Finally, a spline interpolation is performed to obtain the desired step size for the optimization problem, e.g. 3.0 m. Figure 7 compares the curvature profiles for the original and the preprocessed reference lines. It can be seen, that the latter is much smoother.

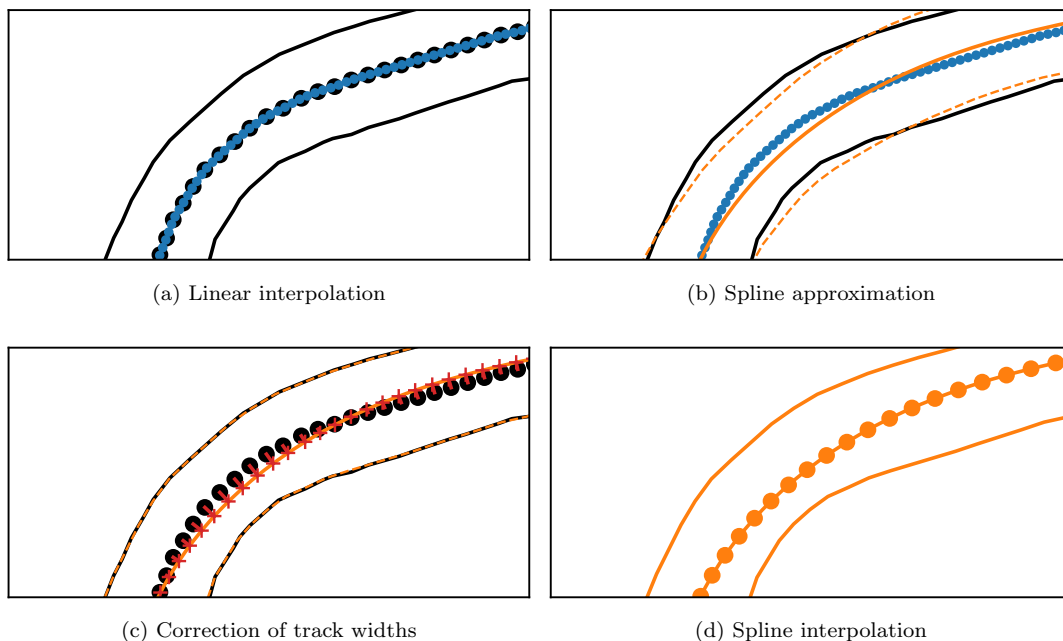


Figure 6. Four steps of reference line preprocessing (top left to bottom right).

To be able to consider the maximum drivable curvature constrained by the car's steering design $\kappa_{\text{bound}} = 0.12 \text{ rad/m}$, we introduced curvature constraints into the optimization problem (19). Therefore, curvature is divided into a static curvature part κ_{ref} originating in the reference line and a variable curvature part κ_{var} originating in the shift along the normal vectors:

$$|\kappa_{\text{ref}} + \kappa_{\text{var}}| \leq \kappa_{\text{bound}}. \quad (20)$$

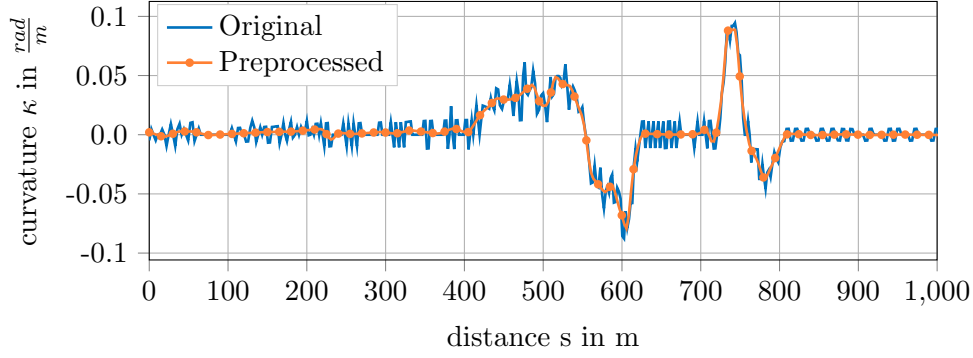


Figure 7. Curvature profiles of the original and the preprocessed reference lines (first 1000 m of the Berlin Formula E track).

For the sake of brevity, only the derivation of the upper boundary is described. The lower boundary is set up accordingly. Defining

$$Q_x = \begin{bmatrix} \frac{y'_1}{(x_1'^2 + y_1'^2)^{\frac{3}{2}}} & 0 & \dots & 0 \\ 0 & \frac{y'_2}{(x_2'^2 + y_2'^2)^{\frac{3}{2}}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{y'_N}{(x_N'^2 + y_N'^2)^{\frac{3}{2}}} \end{bmatrix} \quad \text{and} \quad (21)$$

$$Q_y = \begin{bmatrix} \frac{x'_1}{(x_1'^2 + y_1'^2)^{\frac{3}{2}}} & 0 & \dots & 0 \\ 0 & \frac{x'_2}{(x_2'^2 + y_2'^2)^{\frac{3}{2}}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{x'_N}{(x_N'^2 + y_N'^2)^{\frac{3}{2}}} \end{bmatrix}, \quad (22)$$

we can calculate the curvature of the reference line by

$$\kappa_{\text{ref}} = Q_y T_c \vec{q}_y - Q_x T_c \vec{q}_x. \quad (23)$$

The variable part of the curvature can be stated as

$$\kappa_{\text{var}} = (Q_y T_{n,y} - Q_x T_{n,x}) \vec{\alpha}. \quad (24)$$

Bringing (20) into the standard form as shown in (19), κ_{ref} is considered within the right side of the inequality k_κ and κ_{var} within the left side $E_\kappa \vec{\alpha}$. We finally obtain

$$E_\kappa = Q_y T_{n,y} - Q_x T_{n,x} \quad \text{and} \quad (25)$$

$$k_{\kappa, \text{upper}} = \kappa_{\text{bound}} - \kappa_{\text{ref}}. \quad (26)$$

E_κ is the same for the upper and lower boundary condition and must therefore not be distinguished.

Setting up the original optimization problem before, we assumed the matrices P_{xx} , P_{xy} and P_{yy} being constant. This is approximately true as long as the heading of the optimized path differs only slightly from the reference line. However, in corner entries

and exits the validity of this assumption reduces as can be seen in Figure 4. There, the curvature calculated based on the linearization along the reference line differs significantly from the real curvature. This leads to suboptimal solutions and non-compliance with the given curvature constraints. To overcome this, we implemented a loop around the optimization problem taking advantage of the reference line definition. In the first iteration, the optimization problem is solved as described above. Starting from the second iteration, the reference line is replaced by the previous solution. As the solution is of limited validity during the first iterations, the solution vector $\vec{\alpha}$ is multiplied by a factor of $\frac{1}{3}$ respectively $\frac{2}{3}$ in iterations one and two to limit the displacement from the reference line and therefore the inaccuracies of the curvature calculation. When replacing the reference line by the optimization result the track widths must of course be adapted for the further iterations as it was already done after the spline approximation. In addition, the resulting path must be (spline) interpolated in order to keep equal step sizes that are required by the optimization problem formulation. The termination criterion for the loop is based on the maximum difference between the curvature profiles calculated based on linearizations along the original reference line and along the result itself. We set it to $\Delta\kappa_{\max} = 0.005$ rad/m.

The result for the first and second optimization run (without reduction of $\vec{\alpha}$ for illustration purposes) is displayed in Figure 8. As expected, the path differs strongly at the corner entries and exits since the validity of the linearization decreases in those zones due to the deviating heading. Figure 9 gives an impression of how much the curvature considered within the optimization problem exceeds the real curvature. For the Berlin track, the termination criterion is fulfilled after the fourth iteration. The final optimization result provides a very smooth curvature profile, which is essential for the subsequent velocity profile calculation as well as for the vehicle controller.

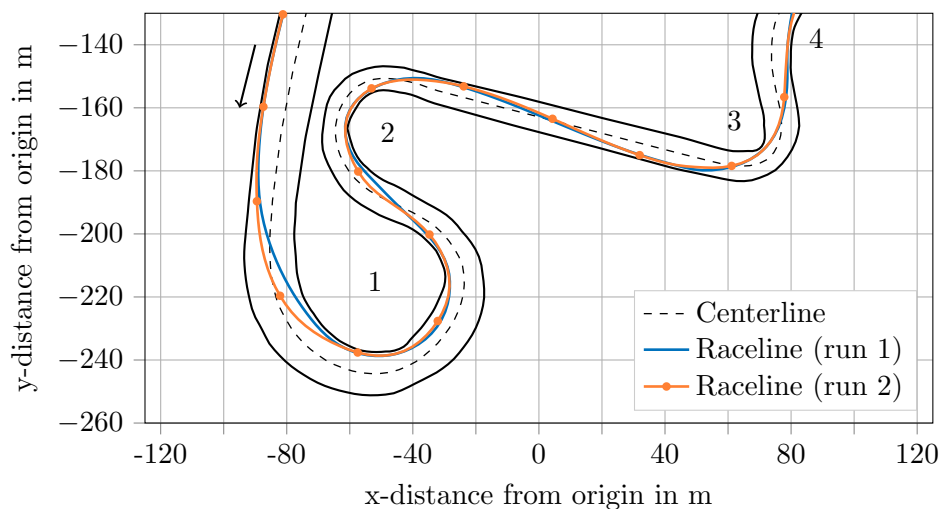


Figure 8. Comparison between the racelines after the first and second optimization runs, where the second problem was solved based on the solution of the first run.

3.5. Generation of a Velocity Profile

Within velocity profile calculation, the car's acceleration limits must be considered as defined by tires, motors and brakes. Again, a fast calculation time is of interest for our application. Another requirement is to exploit the potential of the tires, especially in

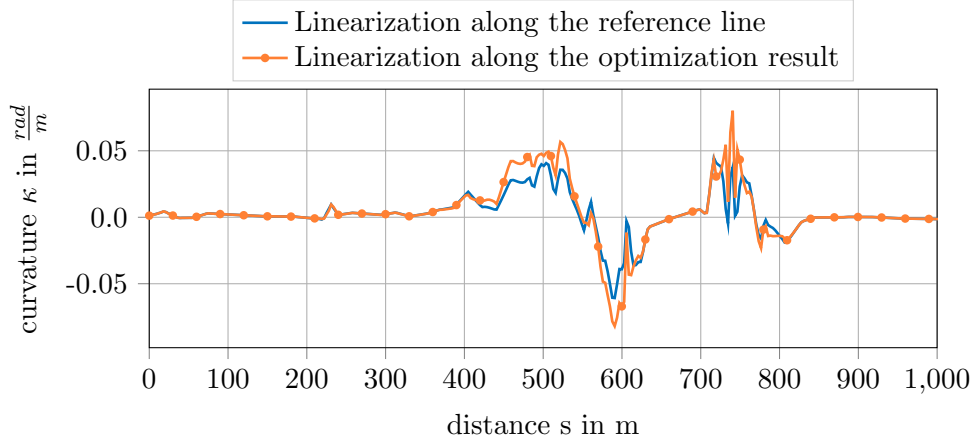


Figure 9. Comparison between the calculated curvature profiles after the first optimization run based on linearizations around the original reference line and around the result itself (first 1000 m of the Berlin Formula E track).

combined slip conditions. A forward-backward-solver was implemented for this purpose. It calculates two velocity profiles, one forward and one backward, that are then intersected. The functional principle of intersecting different partial profiles is widely used in literature, e.g. [15,28–30], and is therefore only briefly summarized.

The basis of the approach is a ggV-diagram containing the longitudinal and lateral acceleration limits of the car at different velocities. The velocity dependency results from aerodynamic effects such as drag and lift. Thus, the diagram is a simplified substitute for the vehicle dynamics and the driving resistances. In a first step, the solver calculates an estimate of the velocity profile based on the smallest lateral acceleration potential of the car $a_{y,\min}$ at any velocity and the curvature profile of the raceline. The estimated velocity profile is then cut the top speed limit of the vehicle. Afterwards the forward calculation procedure is started. It modifies the velocity profile in such a way that it keeps the positive longitudinal as well as the lateral acceleration limits of the car. This is repeated in the backward procedure with the negative longitudinal acceleration limits.

The acceleration profile can then be calculated by

$$a_{x,i} = \frac{v_{x,i+1}^2 - v_{x,i}^2}{2l_i}. \quad (27)$$

Figure 10 shows the velocity and acceleration profiles used for the Berlin track. The maximum velocity was limited to 150 km/h.

3.6. Preparation for Control

While driving, a path-matching algorithm is required to match the car’s position to the global race trajectory because the car cannot follow the planned trajectory exactly due to control errors and external disturbances. For this purpose, the trajectory point with the minimum distance to the car’s center of gravity is used in the first step after starting the car. For all subsequent steps, we work with the minimum distance point that lies within a search window around the expected vehicle position \hat{s}_{i+1} calculated

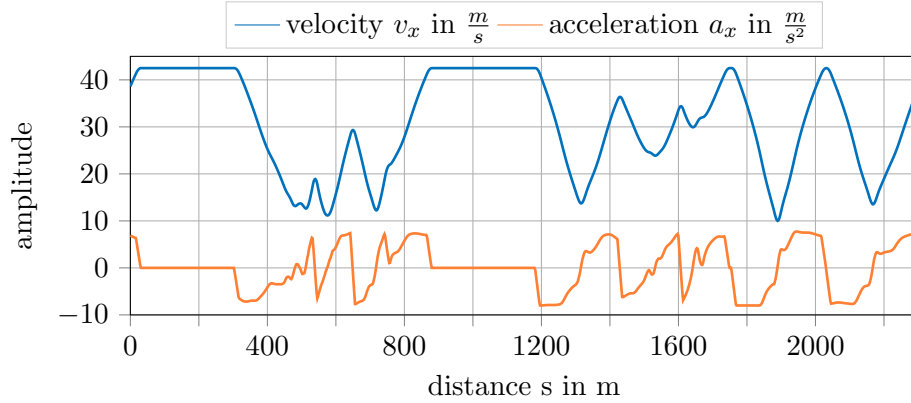


Figure 10. Longitudinal velocity and acceleration profile for a flying lap on the Berlin Formula E track. The maximum velocity was limited to 150 km/h.

by

$$\hat{s}_{i+1} = s_i + v_i (t_{i+1} - t_i). \quad (28)$$

This guarantees that the correct trajectory part is found, even if two parts lie close together, e.g. straight and back straight as in Figure 13.

As already stated, in every cycle we send an emergency trajectory to the controller. The path of this trajectory is equal to the normal one whereas the velocity and acceleration profile are modified such that the car comes to a standstill as quickly as possible, i.e. maximum longitudinal deceleration is used. Since the same velocity profile planner is used, the handling limits of the vehicle are taken into account as for the normal trajectory. However, the look-ahead time of this trajectory is set much higher to have enough space to come to a standstill even in difficult situations. Furthermore, the ggV-diagram for the emergency case allows slightly higher accelerations than the normal one.

3.7. Controller Design

The overall control structure is depicted in Figure 11. The concept encapsulates all dynamics that are closely related to a specific vehicle using a low-level controller. The high-level controller is responsible for providing a suitable path-tracking functionality. The separation increases the system's robustness and eases the transfer of the trajectory-tracking controller to other vehicles. The difficulties of path tracking control for a race car arise from the high nonlinearity of the dynamics and the complex control allocation (split between feed forward and feedback parts and lateral and longitudinal couplings) at the limits of handling.

Resulting from the low-level dynamics abstraction layer, the basic concept of the trajectory controller relies on the description of a point-mass moving along a trajectory. Lateral and longitudinal control are designed separately in the following based on the assumption that the vehicle speed is both constant and known. This enables a gain-scheduling design for the lateral controller based on the vehicle velocity. Defining the lateral path deviation d as the control error, the control design system can be

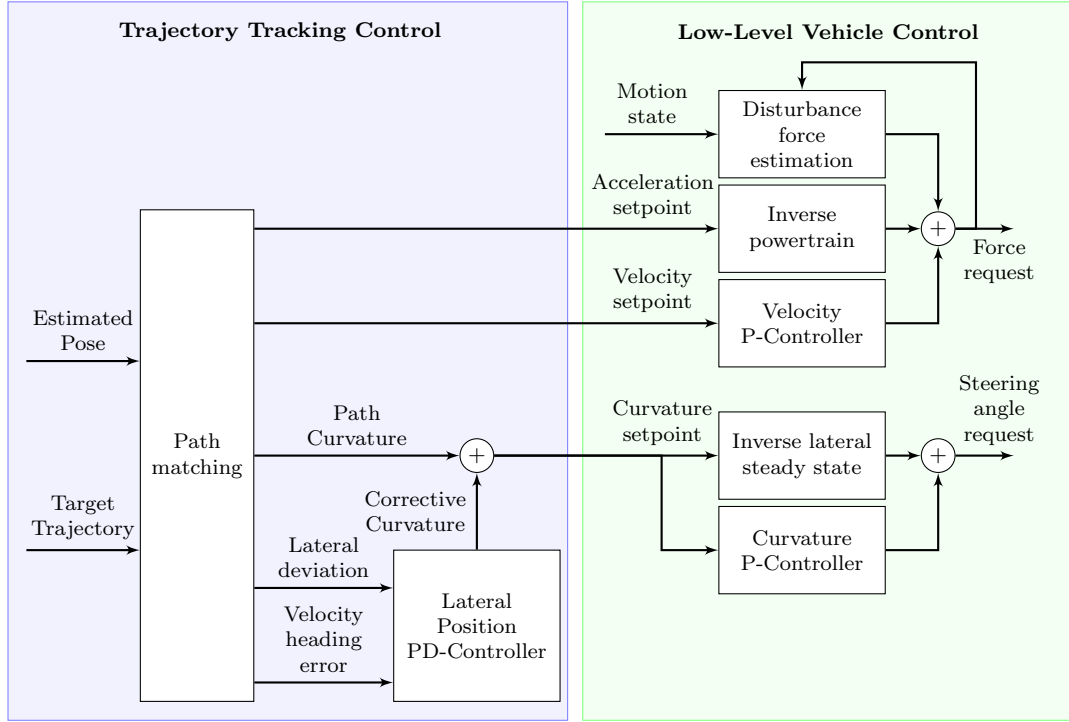


Figure 11. Control concept.

approximated by a double integrator

$$\ddot{d} = a_{y,c} = \kappa_c v^2, \quad (29)$$

where $a_{y,c}$ is the lateral acceleration of the vehicle in path coordinates [55]. Assuming steady-state cornering, we can derive the second equality and use the corrective curvature κ_c as a virtual control input. The feedback law with the undamped eigenfrequency ω_0 and the damping constant D is given by

$$\kappa_c = -\frac{1}{v^2}(\omega_0^2 d + 2D\omega_0 \dot{d}). \quad (30)$$

The resulting closed loop dynamics are independent of the velocity v . Stability of the controller is guaranteed by construction, since quadratic polynomials fulfil the Hurwitz stability criterion if all coefficients are positive. To prevent numerical derivation during implementation of (30), the lateral deviation derivative can be calculated from the difference between the trajectory heading ψ and the vehicle velocity heading, calculated from the vehicle heading ψ_V and the side slip angle β ,

$$\dot{d} = v \sin(\psi - (\psi_V + \beta)). \quad (31)$$

The low-level vehicle control consists of a curvature controller and a velocity controller. Since the sensors cannot measure curvature directly and the side slip angle derivatives are hard to estimate reliably, it is approximated using a steady state as-

sumption

$$\kappa = \frac{\dot{\beta} + \dot{\psi}_V}{v} \approx \frac{\dot{\psi}_V}{v}. \quad (32)$$

The controller itself is designed to be a proportional feedback controller with gain K_κ . Its main purpose is to add counter-steering in case of instabilities. This becomes clear due to the simplified curvature approximation, which actually would allow a reformulation as a yaw rate controller with velocity dependent setpoints. Due to the near neutral setup of the DevBot, the inverse steady-state model could be implemented via a linear relationship between the target curvature κ_T and the steering angle δ using the wheelbase l_V . The overall control law is then given by

$$\delta = K_\kappa (\kappa_T - \kappa) + \kappa_T l_V. \quad (33)$$

The general structure of the velocity controller is similar. It uses proportional feedback to achieve stability and tracking of the velocity set point and combines this with an inverse powertrain model that captures an estimate of the driving resistances and the forces required to overcome inertia. In contrast to the curvature controller, the high level of uncertainty of the driving resistances requires the longitudinal controller to apply additional integral action. It is incorporated by the use of a disturbance observer [71], which outputs an estimate \hat{F}_d for the unmodelled forces acting upon the system. This mechanism can also account for mismatch between the setpoint and the applied force (control variable uncertainty). It was implemented using a steady-state Kalman Filter under the assumption that a constant disturbance acts upon the system. The resulting control law can be written as

$$F = K_v(v_T - v) + F_{PT}(a_{x,T}) + \hat{F}_d, \quad (34)$$

with the powertrain model depending on the feed forward trajectory acceleration $a_{x,T}$

$$F_{PT}(a_{x,T}) = \left(m + \frac{I_F}{r_F^2} + \frac{I_R}{r_R^2} \right) a_{x,T} + 0.5\rho c_w v^2, \quad (35)$$

where I_F and I_R depict the front and rear powertrain inertia and r_F and r_R the corresponding tire radii. ρ is the air density and c_w the effective drag coefficient with the reference area already included.

4. Results and Discussion

In this section we present the results obtained by the suggested methodology and discuss the approach critically.

4.1. Minimum Curvature Trajectory Results

As already stated in Section 3.4, the computation time from centerline import to trajectory output for the Berlin track is 18s with a discretization step size of 3.0m and a raceline length of approximately 2300m. Each of the four QP iterations is

solved in about 0.85 s. The velocity profile calculation requires 65 ms. The rest of the computation time is primarily spent in spline calculations and interpolations. The program is implemented in Python 3 using the numerical math library NumPy. The entire raceline for the Berlin track is shown in Figure 12.

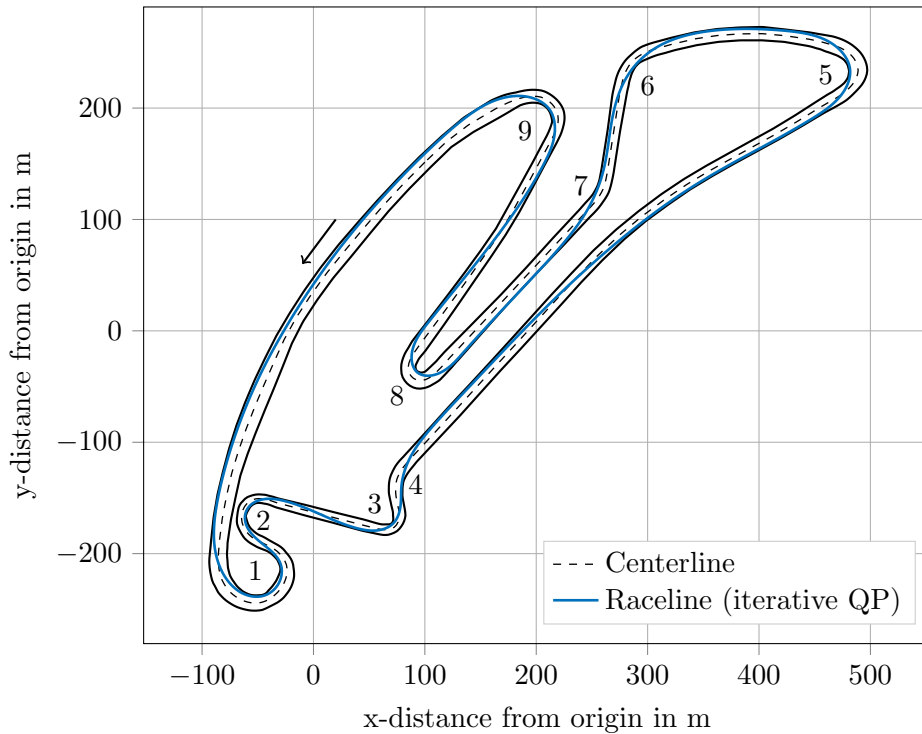


Figure 12. Optimized raceline for the Berlin Formula E track.

During the Berlin event in May 2018 an older version of the QP was used, which had no iteration loops yet. The lap time calculated back then was within half a second of the 91.59 s we achieved in the flying lap in the real event. With the improved optimization problem, we calculate a lap time of 88.41 s taking into account 2.5 m safety distance to the track boundaries as we did in 2018. 86.13 s is the lap time calculated without safety margin. We expect to get close to it in the next race. This should be achievable as we will be able to reduce the safety margin and as all the lap times are calculated on the basis of the same ggV-diagram, which exploits about 80% of the acceleration limits of the car. A further increase of the acceleration limits in the ggV-diagram will probably not be reached in reality due to the simplified vehicle dynamics of the point-mass model in the velocity profile generation.

Figures 13 and 14 visualize the differences of the racelines and curvature profiles between the original problem stated by [30] and the iterative QP formulation for the Upper Heyford test track. It includes a few tight corners and is therefore better suited for the comparison than the Berlin track. Both algorithms are fed with the same smoothed reference line input. The main differences between the racelines can be seen in the corners 2, 5, 6, 7 and 8. Here, the iterative QP makes better use of the track width and therefore avoids unnecessary deflections of the raceline. The differences are even more obvious in the curvature profiles. The iterative QP result shows both a smaller peak curvature and a smoother curve. This results in a lap time of 43.54 s compared to 45.55 s for the original implementation.

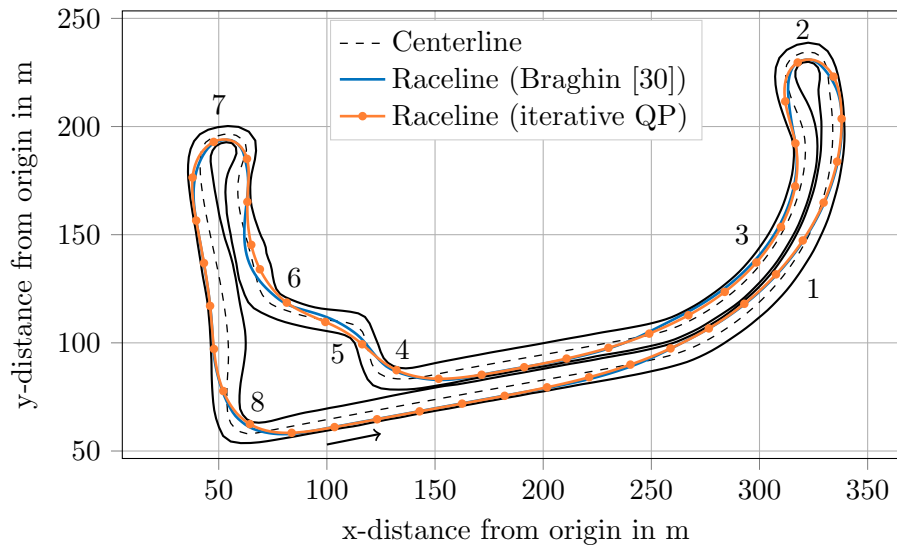


Figure 13. Comparison of the racelines for the Upper Heyford test track between the original formulation in [30] and the iterative QP formulation.

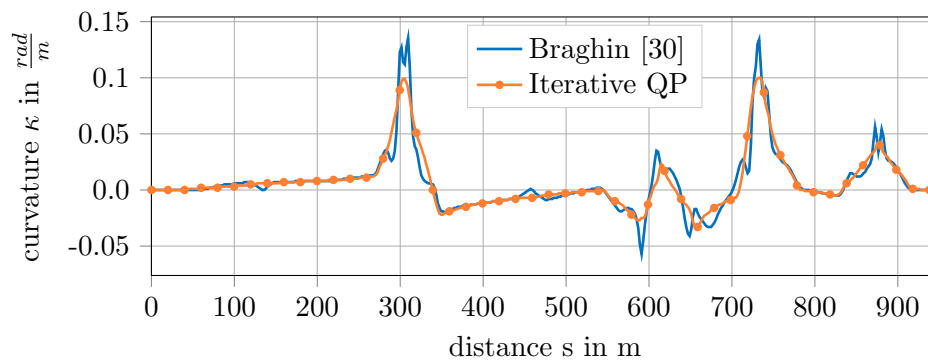


Figure 14. Comparison of the curvature profiles for the Upper Heyford test track between the original formulation in [30] and the iterative QP formulation.

4.2. Control Parameter Effects

In general, the controller must balance good disturbance rejection against oscillatory behavior and noise amplification introduced by high gains. The path tracking parameters have been set to $\omega_0 = 2$ and $D = 0.53$, while the velocity feedback gain was set to $K_v = 1000$ and the curvature feedback gain to $K_\kappa = 0.3$. The tracking results for the Berlin Formula E Track are depicted in Figure 15. The achieved accelerations are depicted in the measured gg-Diagram in Figure 16. It can be seen that the car does not only achieve the claimed accelerations with respect to the longitudinal or lateral direction, but also in the combined setting.

The main reason for the large path tracking errors at some areas of the track are the unmodelled nonlinearities in the vehicle steady-state response to the steering angle input. This could be improved by incorporation of an integral action into the lateral tracking controller. However, this might pose difficulties in case of heavy understeering which might lead to an effect similar to wind-up of the controller. This could not be prevented by standard anti-wind up mechanisms as the maximum steering angle

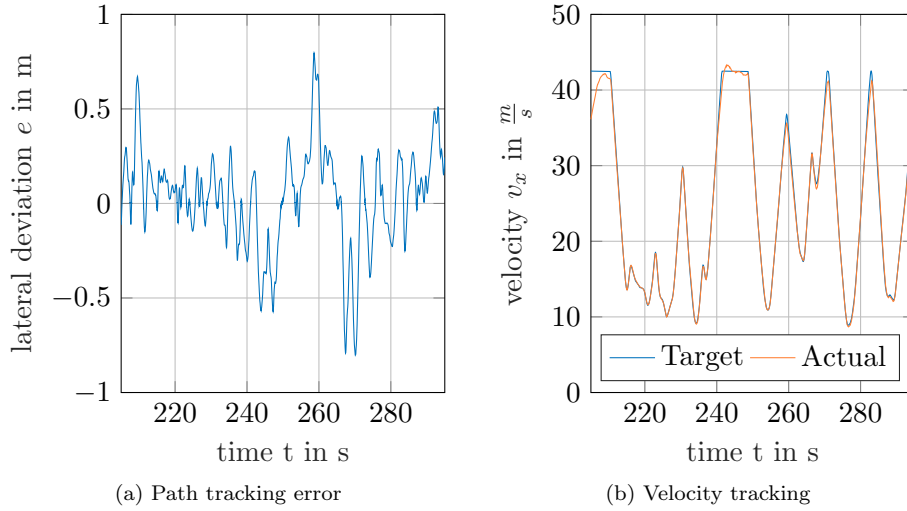


Figure 15. Performance of the overall system at the Formula E track in Berlin 2018.

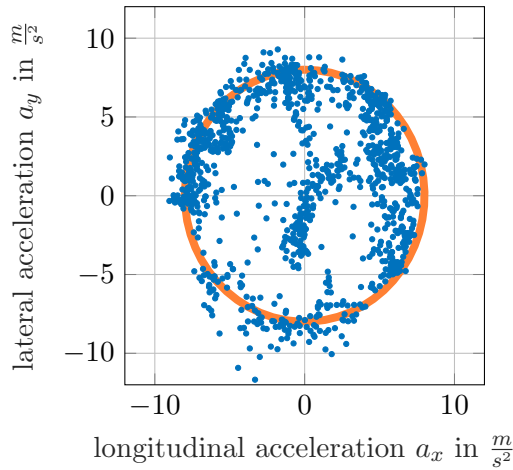


Figure 16. gg-Diagram of the DevBot during its flying lap at the Formula E circuit in Berlin 2018. The raw sensor data was filtered by a moving average filter with a window length of 100 ms before plotting. The red circle specifies the gg-Diagram assumed for planning the trajectory.

which adds an effective lateral acceleration is not known beforehand. Another difficulty during the tuning of the path tracking controllers is that the vehicle response becomes underdamped at high velocities. This poses an upper limit on the maximum achievable speed for a certain closed loop frequency ω_0 . A method to circumvent this would be the application of a fine tuned yaw rate controller, which ensures comparable dynamic behavior over the whole velocity and lateral acceleration range. The velocity tracking error stays below 1 m/s most of the time. The deviations at high speeds result from dynamic torque limitations imposed by the powertrain but not modelled within the controller design.

During the development it was of great interest, how accurate the feed forward law must be to be able to achieve the required control quality. Following [72] one can calculate an upper bound on the allowed disturbance input Δa_y based on the L1-norm

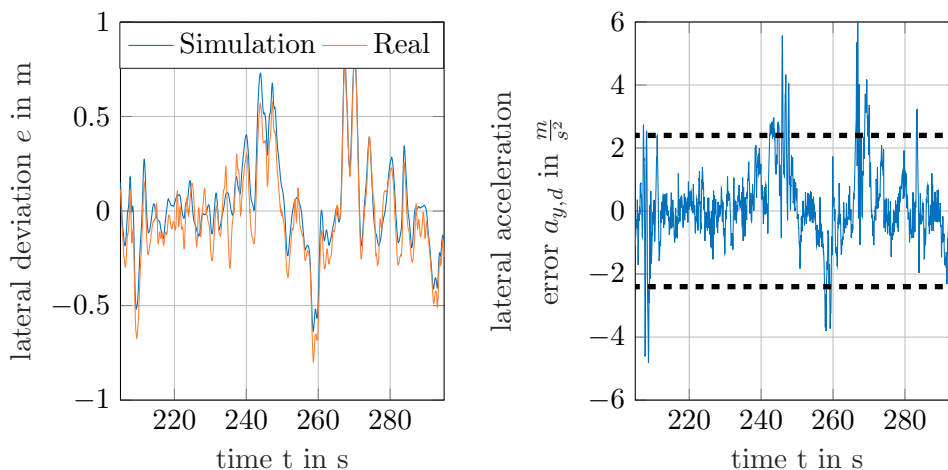
of the linearized error system

$$\|G_c\|_1 = \frac{\|e\|_\infty}{\|\Delta a_y\|_\infty}, \quad (36)$$

where $\|\cdot\|_\infty$ represents the signals peak value. To be meaningful, it is required that the assumptions made earlier about the system dynamics hold at least approximately. This can be checked by calculating the disturbance acting upon the closed-loop system from the sensor data and forward simulation of the linear closed loop system. The disturbance acceleration $a_{y,d}$ is calculated from the target trajectory acceleration $a_{y,T}$, the corrective control acceleration $a_{y,C}$ and the measured acceleration a_y as follows:

$$a_{y,d} = a_y - a_{y,T} - a_{y,C} = a_y - (\kappa_T + \kappa_C) v^2 \quad (37)$$

The resulting disturbance acceleration is used as a system input to the closed loop system resulting from (29) and (30). The predicted lateral control error is compared to the measured control error in Figure 17a. Using the maximum error of 0.8 m and the L1 system norm of the closed loop system of 0.33 (calculated based on the approach of [72]), it follows that the maximum peak on the acceleration disturbance should stay below 2.4 m/s² for all time. This requirement holds nearly everywhere on the track. However, it is violated with a few peaks. This is not harmful, as the derived result is a worst-case bound. It would be reached for example by a step-like signal.



(a) Validation of the linear closed loop model used for controller design using the dataset and control parameters from the Berlin Formula E track.

(b) Lateral acceleration disturbance calculated based on the trajectory, control request and measured lateral acceleration. The dashed lines depict the maximum allowed disturbance level calculated using the L1-norm of the closed-loop system and a maximum control error of 0.8 m.

Figure 17. Results of controller disturbance rejection capabilities.

4.3. Discussion of the Approach

With hindsight, the chosen methodology proved to be a good and reliable way for the planning and control software of an autonomous race car. The workflow from centerline to control is easy to adapt to different race environments and showed robust

performance on various tracks. The computation times remain low for small testing tracks as well as for a whole Formula E track. Furthermore, reasonably fast lap times are achieved.

For the Upper Heyford test track displayed in Figure 13, a lap time improvement of 4.4% was reached compared to the original formulation of the optimization problem, which is a lot in motorsports. This was achieved by the extension of the optimization problem to approximate the real curvature as well as by the introduction of the iterative invocation of the QP. Together with the spline approximation for the reference line input the latter significantly improves robustness when using real world tracks with imperfect reference lines. It also allowed us to introduce curvature constraints into the optimization problem such that the result is drivable with a real car. This is especially important on tighter tracks containing 180° hairpin corners, e.g. Formula E tracks.

A disadvantage arose in the context of the curvilinear coordinate system. In tight corners it can happen that several normals are crossed at the track boundary as shown in Figure 18. This happens if a small corner radius coincides with a large track width and if the discretization step size between two points is small enough. This ambiguity can appear for all approaches based on curvilinear coordinate systems. We solve it by raising the smoothing factor of the spline approximation. This results in the reference line moving towards the inside of the corner, thus avoiding the problem, cp. Figure 6(b).

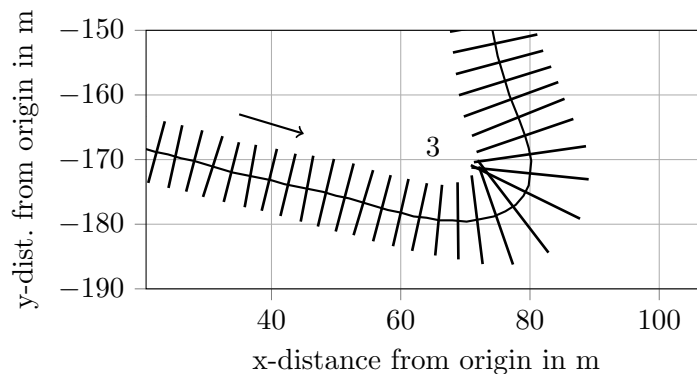


Figure 18. Reference line normals crossing at $x = 72$ m, $y = -171$ m due to a small corner radius in combination with a large track width.

5. Conclusion and Outlook

In this paper, we presented our approach to planning a minimum curvature trajectory for an autonomous race car and introduced a controller design that allows it to follow the trajectory at the handling limits. Improvements to the optimization problem were introduced that resulted in a significant lap time improvement and made it a lot more robust to real world application. The software design proved to work at a velocity of 150 km/h and 80% of the vehicle’s acceleration potential. The whole software pipeline can easily be adapted to various racetracks. The results illustrate the capabilities of the pipeline and its suitability for autonomous motorsports.

In the future, we plan to retain the general approach. The global planner will be complemented by a local trajectory planner to allow overtaking and evasion maneuvers

taking static and dynamic objects on the racetrack into account. This is where we will be able to take advantage of the modular system structure. Furthermore, some effort will be put into the software part that is currently executed offline so that it can be executed online on the car, e.g. to re-plan the raceline due to a new static object appearing during the race. Due to its fast calculation time, the minimum curvature optimization can be performed within reasonable time also on weak CPUs and is therefore well suited for this purpose. In parallel with this, we will further evaluate other optimization approaches that consider a more detailed vehicle dynamics model and directly minimize the lap time.

The entire Python code used in the TUM Roborace team for global trajectory optimization will be published under an open source license on GitHub after publication of this paper.¹

Acknowledgements and Contributions

Alexander Heilmeyer as the first author is responsible for the trajectory planning software. He initiated the idea of the paper and contributed to the analysis and modification of the optimization problem, to the velocity profile generation as well as to the behavior state machine. Furthermore, he contributed essentially to the overall system design including the distributed calculation architecture between offline optimization and online processing. Alexander Wischnewski is responsible for the control software. He is the main contributor to the control system design and contributed essentially to the reformulation of the optimization problem. Leonhard Hermansdorfer is responsible for processing and preparing the map and centerline extrication for the trajectory optimization problem. Johannes Betz contributed to the overall system design and the mapping software. Markus Lienkamp and Boris Lohmann contributed equally to the conception of the research project and revised the paper critically for important intellectual content. They gave final approval of the version to be published and agree to all aspects of the work. As guarantors, they accept responsibility for the overall integrity of the paper. Research was supported by the basic research fund of the Chair of Automotive Technology of the Technical University of Munich.

We would like to thank Roborace for the possibility of demonstrating our software on their car as well as for their support during the testing sessions and the Berlin event. We would also like to thank Fabian Christ, who implemented the minimum time optimization as well as the spline approximation in his master thesis within the project.

References

- [1] Betz J, Wischnewski A, Heilmeyer A, et al. What can we learn from autonomous level-5 motorsport? In: Pfeffer P, editor. 9th International Munich Chassis Symposium 2018; Wiesbaden. Springer Fachmedien Wiesbaden; 2019. p. 123–146.
- [2] The Fast and the Driverless: Munich Team Takes Home Roborace Victory [Internet]. [cited 2018 June 29] ; 2018. Available from: <https://blogs.nvidia.com/blog/2018/06/29/fast-and-driverless-munich-roborace-victory>.
- [3] Grisetti G, Stachniss C, Burgard W. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*. 2007;23(1):34–46.

¹https://github.com/TUMFTM/global_racetrajectory_optimization

-
- [4] Hess W, Kohler D, Rapp H, et al. Real-Time Loop Closure in 2D LIDAR SLAM. In: 2016 IEEE International Conference on Robotics and Automation (ICRA); 2016. p. 1271–1278.
- [5] Thrun S, Montemerlo M, Dahlkamp H, et al. Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*. 2006;23(9):661–692.
- [6] Urmson C, Anhalt J, Bae H, et al. Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*. 2008 June;25(8):425–466.
- [7] Dolgov D, Thrun S, Montemerlo M, et al. Practical Search Techniques in Path Planning for Autonomous Driving. *AAAI Workshop - Technical Report*. 2008 01;.
- [8] Bacha A, Bauman C, Faruque R, et al. Odin: Team VictorTango’s entry in the DARPA Urban Challenge. *Journal of Field Robotics*. 2008 01;25:467–492.
- [9] John L, Jonathan H, Seth T, et al. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*. 2008;25(10):727–774.
- [10] Urmson C, Anhalt J, Bagnell D, et al. Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics*. 2008 aug;25(8):425–466.
- [11] Kammel S, Ziegler J, Pitzer B, et al. Team AnnieWAY’s autonomous system for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*. 2008;25(9):615–639.
- [12] Montemerlo M, Becker J, Bhat S, et al. Junior: The Stanford Entry in the Urban Challenge. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. Chapter 3; p. 91–123.
- [13] Kapania NR, Gerdes JC. Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling. *Vehicle System Dynamics*. 2015; 53(12):1687–1704.
- [14] Laurence V, Goh J, Gerdes J. Path-tracking for autonomous vehicles at the limit of friction. In: 2017 American Control Conference (ACC); May; 2017. p. 5586–5591.
- [15] Kapania N, Subosits J, Gerdes J. A Sequential Two-Step Algorithm for Fast Generation of Vehicle Racing Trajectories. *Journal of Dynamic Systems, Measurement, and Control*. 2016;138(9).
- [16] Liniger A. Path Planning and Control for Autonomous Racing [dissertation]. ETH Zurich; 2018.
- [17] Liniger A, Domahidi A, Morari M. Optimization-based autonomous racing of 1:43 scale RC cars. *Optimal Control Applications and Methods*. 2014;36(5):628–647.
- [18] Serban AC, Poll E, Visser J. A standard driven software architecture for fully autonomous vehicles. In: 2018 IEEE International Conference on Software Architecture Companion (ICSA-C); 2018. p. 120–127.
- [19] Paden B, Čáp M, Yong SZ, et al. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles*. 2016 March; 1(1):33–55.
- [20] Pendleton SD, Andersen H, Du X, et al. Perception, planning, control, and coordination for autonomous vehicles. *Machines*. 2017;5(1).
- [21] Matkan AA, Hajeb M, Sadeghian S. Road Extraction from Lidar Data Using Support Vector Machine Classification. *Photogrammetric Engineering & Remote Sensing*. 2014; 80(5):409–422.
- [22] Miao Z, Wang B, Shi W, et al. A Semi-Automatic Method for Road Centerline Extraction From VHR Images. *IEEE Geoscience and Remote Sensing Letters*. 2014;11(11):1856–1860.
- [23] Guan J, Wang Z, Yao X. A new approach for road centerlines extraction and width estimation. In: IEEE 10th International Conference on Signal Processing Proceedings. IEEE; 24.10.2010 - 28.10.2010. p. 924–927.
- [24] Cheng G, Wang Y, Xu S, et al. Automatic Road Detection and Centerline Extraction via Cascaded End-to-End Convolutional Neural Network. *IEEE Transactions on Geoscience and Remote Sensing*. 2017;55(6):3322–3337.
- [25] Guedri H, Abdallah MB, Nasri F, et al. Computer method for tracking the centerline curve of the human retinal blood vessel. In: 2017 International Conference on Engineering & MIS (ICEMIS). IEEE; 08.05.2017 - 10.05.2017. p. 1–6.
- [26] Sironi A, Lepetit V, Fua P. Multiscale Centerline Detection by Learning a Scale-Space

- Distance Transform. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 23.06.2014 - 28.06.2014. p. 2697–2704.
- [27] Sironi A, Turetken E, Lepetit V, et al. Multiscale Centerline Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2016;38(7):1327–1341.
- [28] Velenis E, Tsiotras P. Optimal velocity profile generation for given acceleration limits: receding horizon implementation. In: *Proceedings of the 2005 American Control Conference*; Vol. 3; June; 2005. p. 2147–2152.
- [29] Brayshaw DL, Harrison MF. A quasi steady state approach to race car lap simulation in order to understand the effects of racing line and centre of gravity location. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*. 2005;219(6):725–739.
- [30] Braghin F, Cheli F, Melzi S, et al. Race Driver Model. *Computers and Structures*. 2008 jul;86(13-14):1503–1516.
- [31] Katrakazas C, Quddus M, Chen WH, et al. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*. 2015;60:416–442.
- [32] Betts JT. Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*. 1998;21(2):193–207.
- [33] Polak E. An Historical Survey of Computational Methods in Optimal Control. *SIAM Review*. 1973;15(2):553—584.
- [34] Perantoni G, Limebeer DJ. Optimal control for a formula one car with variable parameters. *Vehicle System Dynamics*. 2014;52(5):653–678.
- [35] Dal Bianco N, Lot R, Gadola M. Minimum time optimal control simulation of a gp2 race car. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*. 2018;232(9):1180–1195.
- [36] Dal Bianco N, Bertolazzi E, Biral F, et al. Comparison of direct and indirect methods for minimum lap time optimal control problems. *Vehicle System Dynamics*. 2018;0(0):1–32.
- [37] Dijkstra EW. A Note on Two Problems in Connexion with Graphs. *Numer Math*. 1959 dec;1(1):269–271.
- [38] Hart PE, Nilsson NJ, Raphael B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*. 1968 July;4(2):100–107.
- [39] Pohl I. First Results on the Effect of Error in Heuristic Search. Edinburgh University, Department of Machine Intelligence and Perception; 1969. MIP-R.
- [40] Stentz A. Optimal and efficient path planning for partially-known environments. In: *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*; Vol. 4; May; 1994. p. 3310–3317.
- [41] Stentz A. The Focussed D* Algorithm for Real-time Replanning. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. Morgan Kaufmann Publishers Inc.; 1995. p. 1652–1659; IJCAI'95.
- [42] Koenig S, Likhachev M. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*. 2005 June;21(3):354–363.
- [43] Lavalle SM. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*. Iowa State University; 1998.
- [44] Karaman S, Frazzoli E. Optimal kinodynamic motion planning using incremental sampling-based methods. In: *49th IEEE Conference on Decision and Control (CDC)*; Dec; 2010. p. 7681–7687.
- [45] Otte M, Frazzoli E. *RRT-X: Real-Time Motion Planning/Replanning for Environments with Unpredictable Obstacles*. Springer International Publishing; 2015. Chapter 27; p. 461–478.
- [46] Hsu D, Kindel R, Latombe JC, et al. Randomized Kinodynamic Motion Planning with Moving Obstacles. *The International Journal of Robotics Research*. 2002;21(3):233–255.
- [47] Katriniok A, Abel D. LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics. In: *2011 50th IEEE Conference on Decision and Control*

- and European Control Conference; Dec; 2011. p. 6828–6833.
- [48] Gerdtts M, Karrenberg S, Müller-Beßler B, et al. Generating locally optimal trajectories for an automatically driven car. *Optimization and Engineering*. 2008 Apr;10(4):439.
- [49] Glaser S, Vanholme B, Mammar S, et al. Maneuver-Based Trajectory Planning for Highly Autonomous Vehicles on Real Road With Traffic and Driver Interaction. *IEEE Transactions on Intelligent Transportation Systems*. 2010 Sept;11(3):589–606.
- [50] Jeon Jh, Cowlagi RV, Peters SC, et al. Optimal motion planning with the half-car dynamical model for autonomous high-speed driving. In: 2013 American Control Conference; June; 2013. p. 188–193.
- [51] Arab A, Yu K, Yi J, et al. Motion planning for aggressive autonomous vehicle maneuvers. In: 2016 IEEE International Conference on Automation Science and Engineering (CASE); Aug; 2016. p. 221–226.
- [52] Rizano T, Fontanelli D, Palopoli L, et al. Global path planning for competitive robotic cars. In: 52nd IEEE Conference on Decision and Control; Dec; 2013. p. 4510–4516.
- [53] Rizano T, Fontanelli D, Palopoli L, et al. Local Motion Planning for Robotic Race Cars. 52nd IEEE Conference on Decision and Control. 2013;:4510–4516.
- [54] Funke J, Theodosis P, Hindiyeh R, et al. Up to the limits: Autonomous Audi TTS. In: IEEE Intelligent Vehicles Symposium; June; 2012. p. 541–547.
- [55] Guldner J, Tan HS, Patwardhan S. Study of design directions for lateral vehicle control. In: Proceedings of the 36th IEEE Conference on Decision and Control; Vol. 5; Dec; 1997. p. 4732–4737 vol.5.
- [56] Roselli F, Corno M, Savaresi SM, et al. H-Infinity control with look-ahead for lane keeping in autonomous vehicles. 2017 IEEE Conference on Control Technology and Applications (CCTA). 2017;:2220–2225.
- [57] Werling M, Gröll L, Bretthauer G. Invariant trajectory tracking with a full-size autonomous road vehicle. *IEEE Transactions on Robotics*. 2010;26(4):758–765.
- [58] Werling M. Ein neues Konzept für die Trajektoriengenerierung und -stabilisierung in zeitkritischen Verkehrsszenarien. *At-Automatisierungstechnik*. 2012;60(1):53–54.
- [59] Fuchshumer S, Schlacher K, Rittenschober T. Nonlinear Vehicle Dynamics Control - A Flatness Based Approach. In: Proceedings of the 44th IEEE Conference on Decision and Control; Dec; 2005. p. 6492–6497.
- [60] Aguilar LE, Hamel T, Soueres P. Robust path following control for wheeled robots via sliding mode techniques. In: Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97; Vol. 3; Sept; 1997. p. 1389–1395.
- [61] Hamerlain F, Achour K, Floquet T, et al. Trajectory tracking of a car-like robot using second order sliding mode control. In: 2007 European Control Conference (ECC); July; 2007. p. 4932–4936.
- [62] Katriniok A, Maschuw JP, Eckstein L, et al. Optimal Vehicle Dynamics Control for Combined Longitudinal and Lateral Autonomous Vehicle Guidance. In: 2013 European Control Conference (ECC); 2013. p. 974–979.
- [63] Calzolari D, Schürmann B, Althoff M. Comparison of trajectory tracking controllers for autonomous vehicles. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC); Oct; 2017. p. 1–8.
- [64] Roborace [Internet]. [cited 2018 June 01] ; 2018. Available from: <http://roborace.com>.
- [65] Kong TY, Rosenfeld A, editors. *Topological Algorithms for Digital Image Processing*. Elsevier Science Inc.; 1996.
- [66] Maurer CR, Qi R, Raghavan V. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2003 Feb;25(2):265–270.
- [67] Meyer F. Topographic distance and watershed lines. *Signal Processing*. 1994;38(1):113 – 125. *Mathematical Morphology and its Applications to Signal Processing*.
- [68] Orfanidis S. *Introduction to Signal Processing*. Pearson Education, Inc.; 1996.
- [69] Gundlach I, Konigorski U, Hoedt J. Zeitoptimale Trajektorienplanung für automatisiertes

- Fahren im fahrdynamischen Grenzbereich. VDI-Berichte. 2017;(2292):223–234.
- [70] Papula L. *Mathematische Formelsammlung: Für Ingenieure und Naturwissenschaftler*. Springer Fachmedien Wiesbaden; 2017.
- [71] Chen W, Yang J, Guo L, et al. Disturbance-Observer-Based Control and Related Methods—An Overview. *IEEE Transactions on Industrial Electronics*. 2016 Feb;63(2):1083–1095.
- [72] Boyd S, Doyle J. Comparison of peak and RMS gains for discrete-time systems. *Systems and Control Letters*. 1987;9(1):1–6.

4.3 Lap Time Simulation

4.3.1 Quasi-Steady-State Lap Time Simulation

This section summarizes the work carried out on the LTS that has been published in [9]. The LTS is available on GitHub [165].

Summary of the Paper

Many LTS published in the literature no longer match the current state of technology of many racing series because they cannot simulate the hybrid or purely electric powertrains that have been introduced in recent years. The publication presents an LTS that fills this gap and is tailored to the specific requirements in the context of race strategy determination. These include fast computing times in the range of one second to be able to perform parameter studies and a simple parameterization since the exact properties of the race cars are unknown to the public.

Due to these requirements, the developed LTS is based on a quasi-steady-state approach. It can simulate all relevant powertrain types (combustion, hybrid, electric) and topologies (front-wheel drive, rear-wheel drive, all-wheel drive) as well as the DRS. Furthermore, it includes a simplified simulation of various energy management strategies that determine how the available energy is distributed within the powertrain for hybrid powertrains. A simplified two-track model without kinematic relations is used to model the race car. In contrast to a point mass model, the longitudinal and lateral wheel load transfers due to the acting accelerations are considered this way. For them to affect the lap times, a simple self-designed non-linear tire model is used. Apart from the race car model, the solver is important for the result. The newly developed and in the publication presented *forward-backward plus* solver is a hybrid of the two commonly used solver types *forward-backward* and *pure forward* to combine the faster computing times of the former with the higher accuracy of the latter. To expand the application possibilities of the LTS, it can simulate the effects of yellow flags as well as driving through the pit lane.

Compared to other LTS, the developed simulation is particularly suitable for use in the context of race strategy determination. It represents a good compromise between the level of detail, computing speed, and functional range in order to determine various necessary parameters for the RS.

Relation to the Research Questions

The publication is related to the second research question. Using a precomputed racing line as an input, the developed LTS enables the user to robustly determine several parameters for the RS, e.g., qualifying lap time and mass sensitivity of the lap time. The chosen quasi-steady-state approach offers the best compromise between computing time and accuracy for the intended application.

Individual Contribution

The initial version of the LTS was developed in a semester thesis by Geißlinger [166]. After that, Heilmeyer completely reworked the implementation, extended wide parts of the functionality and the solver, and parameterized several race cars based on real-world velocity profiles to obtain the results shown in the paper.

Imprint of the Paper

©2019 IEEE. Reprinted, with permission, from Alexander Heilmeier, Maximilian Geißlinger, and Johannes Betz, A Quasi-Steady-State Lap Time Simulation for Electrified Race Cars, 2019 Fourteenth International Conference on Ecological Vehicles and Renewable Energies (EVER), 2019.

A Quasi-Steady-State Lap Time Simulation for Electrified Race Cars

Alexander Heilmeier*, Maximilian Geisslinger*, Johannes Betz*

*Chair of Automotive Technology
 Technical University of Munich
 Garching, Germany
 Email: alexander.heilmeier@tum.de

Abstract—In motorsports, lap time simulation (LTS) is used by race engineers to evaluate the effects of setup changes on lap time and energy consumption. Many of the LTS published to date are no longer able to meet today’s requirements because more and more racing series are introducing hybrid systems to improve powertrain efficiency. In addition, some racing series have purely electric powertrains. As a result, new types of LTS are needed that can represent the current state of technology. In addition to various powertrain types and topologies, this also includes the drag reduction system as well as the simulation of energy management strategies that control the distribution of energy within the hybrid system. For use as a co-simulation together with a race simulation, yellow flags and pit lanes should also be modeled. This paper presents an LTS that covers these aspects and, thanks to an improved quasi-steady-state solver, delivers accurate results within a short computing time. Particular emphasis was placed on easy parametrization, based on publicly available data. Exemplary results are shown for Formula 1 and Formula E cars on different racetracks. Three different energy management strategies are compared with regard to the most efficient use of the available energy.

Keywords—*quasi-steady-state; lap time simulation; race car; hybrid powertrain; electric powertrain.*

I. INTRODUCTION

Motorsport has always served as a technology demonstration for automobile manufacturers. This continues to be the case during the transition from internal combustion engines (ICE) to electric motors. In 2014, the FIA Formula 1 (F1) introduced a hybrid system supporting a 1.6-litre ICE. With this, the system reached a total thermal efficiency of over 50% [1]. In the same year, the first racing series with a purely electric powertrain, the FIA Formula E (FE), started. Today, several racing series use either hybrid systems or even electric motors alone in their powertrains.

Regardless of the racing series, a racing team needs various simulation tools to successfully participate in a race. One of them is a lap time simulation (LTS). It simulates a single race car with a specific setup for one lap on a given racetrack. LTS outputs are not limited to the calculation of lap time, but also focus on further results, such as energy consumption. It can be utilized for virtual setup optimization, for example. In the case of hybrid or purely electrically powered race cars, further requirements exist. Hereby it is important to be able to simulate the effects of different energy management strategies on lap time and energy consumption in order to find the optimal balance. Long-term effects, such as tire wear or fuel mass loss, are usually omitted in the LTS. This is in contrast to race simulations that simulate an entire race and are used to determine the race strategy [2]. However, both types of simulation work closely together because the LTS provides many of the required parameters for a race simulation, such as the lap time mass sensitivity. Therefore, the use as a co-simulation together with a race simulation is another application case.

II. RELATED WORK

In general, LTS can be divided into two groups. The first group calculates a velocity profile for a given raceline. This can be done using a driver model or under the assumption of a perfect driver. Depending on the model implementation, the group can be further classified into steady-state and quasi-steady-state approaches. The second group simultaneously optimizes the raceline and the velocity profile on a given racetrack with a specific target, e.g. to achieve a minimum lap time or to find the most efficient energy management strategy. These problems are usually solved by applying optimal control techniques to transient simulation models. Steady-state, quasi-steady-state and transient simulations are differentiated by their different approaches to the compromise

between accuracy and computational speed. For a more detailed description and differentiation of the three types, we refer to Siegler et al. [3] and Völkl [4].

Siegler et al. [3] compare the three model types. They find that quasi-steady-state and transient approaches deliver more accurate results than the steady-state modeling. Further research is carried out for a quasi-steady-state approach using a g-g-v diagram calculated by optimal control techniques in Brayshaw et al. [5]. The results show that it has a similar sensitivity to setup changes as an optimized transient solution of a seven-degrees-of-freedom (7DOF) model. Colunga et al. [6] published a method to transform the differential equations of a 7DOF suspension and a transient cornering model into a discrete state-space representation. The approach can be used to evaluate the impact of road roughness, for example. Coming to pure optimal control techniques with transient models, Casanova [7] and Kelly [8] are representatives for the minimum lap time target. The latter improves robustness of the solution in terms of driveability by considering stability criteria in the optimization problem. He also finds that a quasi-steady-state model provides comparable results to a transient model if it is applicable. However, both publications have large computation times of several hours for simulating a single lap. A methodology for the integration of transient modeling into the quasi-steady-state calculation method is presented by Völkl [4]. The transient states are computed in a separate model and solved iteratively by superposition. Timings et al. [9] show an LTS based on model predictive control and extended by a compensatory controller for robustness to driver mistakes and disturbances. Therefore, the calculated lap time is more likely to be achieved in reality. The related publications Perantoni et al. [10] and Limebeer et al. [11] minimize lap time utilizing a 3D-representation of the racetrack. A method to compute the time-optimal energy management strategy for the F1 hybrid powertrain is described by Limebeer et al. [12]. Ebbesen et al. [13] present an approach that leads to a quick solving of the optimal control problem for this case, but results in lower accuracy. However, treating the energy management as an optimization problem does not necessarily deliver driving behavior strategies that are feasible during a race.

III. METHODOLOGY

Our target was to develop a lap time simulation that is suitable for use as a co-simulation together with a race simulation. In the future, the tandem could be utilized to determine and adapt the race strategy of an autonomous race car online during the race, e.g. in

the racing series Roborace [14], [15]. Therefore, a low computation time and robust convergence characteristics were important demands for the solver. Furthermore it should be adjustable to different racing series and reflect the current state of technology. Accordingly, it should provide options for simulating the drag reduction system (DRS) as well as the powertrain topologies rear-wheel drive (RWD), front-wheel drive (FWD) and all-wheel drive (AWD). The same applies to the powertrain types (pure ICE, hybrid system and pure electric motors), including the corresponding energy management strategies. In addition, external influences on the lap time, such as yellow flags or driving through the pit, should also be taken into account. The research on literature shows that there is no LTS fulfilling the stated demands.

Our LTS consists of three parts: racetrack model, vehicle model and solver. Since we do not have access to any team data, the simulation is designed in such a way that it can be parametrized based on publicly available data, such as onboard video streams and lap times. It can be utilized for almost every (circuit) racing series. For this paper, we have focused on F1 because it is the most popular racing series and includes a hybrid powertrain that allows us to analyze and compare different energy management strategies. This section presents the most important aspects of the three parts.

A. Racetrack model

The main purpose of the racetrack model is to provide the curvature profile of the racetrack for the solver. The starting point for this is GPS coordinates of the centerline of the racetrack, which can be obtained from the OpenStreetMap project [16], for example. For various types of LTS, the centerline would be a valid input, since the raceline is found together with the velocity profile during execution of the solver. However, this is computationally expensive. Therefore, we want to enter a pre-computed raceline directly. Then, the solver only needs to calculate the exact velocity profile for it. To obtain the raceline on the basis of the centerline, we implemented a path optimization based on a curvature minimization. The minimum curvature line is reasonable near a real raceline, as it allows the highest cornering speeds for a given lateral acceleration limit by the race car. The approach is based on Braghin et al. [17], but has been significantly extended and will be presented in another paper. The raceline of the Shanghai racetrack is shown in Figure 1.

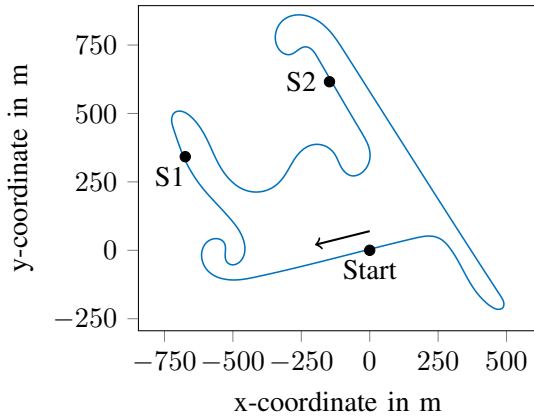


Fig. 1. Raceline of the Shanghai racetrack. S1 and S2 are the boundaries of sector 1 and sector 2. Sector 3 ends at the start/finish line.

Based on the x-y coordinates, the curvature κ_i of the i th raceline point can be calculated by [18, p. 373]

$$\kappa_i = \frac{x'_i y''_i - y'_i x''_i}{(x'^2_i + y'^2_i)^{\frac{3}{2}}}. \quad (1)$$

The curvature profile for our Shanghai raceline is shown in Figure 2. Due to the previous optimization, it is very smooth. Otherwise, it would have to be processed because it is a decisive factor for the calculated velocity profile and lap time. This can be done by applying a moving average filter, for example.

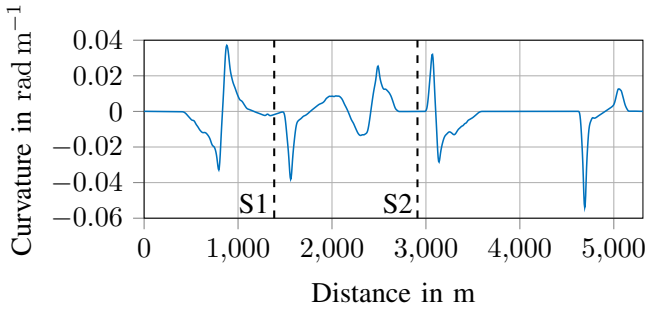


Fig. 2. Curvature profile of the Shanghai raceline. S1 and S2 are the sector boundaries.

For race strategy purposes, it is useful to know how much lap time it costs to drive through the pit with the prescribed speed limit. Furthermore, there is a global speed limit in some racing series, for example in FE. Hence, the racetrack model includes the option to specify a maximum velocity for every discretization point on the raceline.

B. Vehicle model

The vehicle model provides all car-related values to the solver. In order to ensure an easy parametrization, we

kept it simple without neglecting important effects. The basis is a simplified two-track model, which is regarded in steady-state and does not include kinematic relations. All the required parameters are described in Table I.

One central task is to calculate the transmittable tire forces, considering the velocity v and longitudinal and lateral accelerations a_x and a_y . The tire model considers the effect of degressive tire force potential $F_{\text{tire,pot}}$ with rising tire load F_z as well as a friction value μ :

$$F_{\text{tire,pot}} = \mu (p_1 F_z + p_2 F_z^2). \quad (2)$$

The parameters p_1 and p_2 must be adjusted to the specific tire. The friction value can be used to include the effects of weather or tarmac variation on different racetracks. Longitudinal and lateral tire load transfers, as well as the effect of aerodynamic downforce, are considered within tire load calculation. For the front left tire (FL), we obtain (3). The remaining tire loads are calculated accordingly. The simulated tire load profiles for a F1 car in Shanghai are displayed in Figure 3. It can be used to check for plausibility of the aerodynamic downforce as well as to determine the most stressed tire on a particular track.

$$F_{z,\text{FL}} = \frac{1}{2} m g \frac{l_{\text{cog},r}}{l} - \frac{1}{2} m a_x \frac{h_{\text{cog}}}{l} - m a_y \frac{l_{\text{cog},r}}{l} \frac{h_{\text{cog}}}{s} + \frac{1}{2} \frac{1}{2} c_{z,A,f} \rho_{\text{air}} v^2. \quad (3)$$

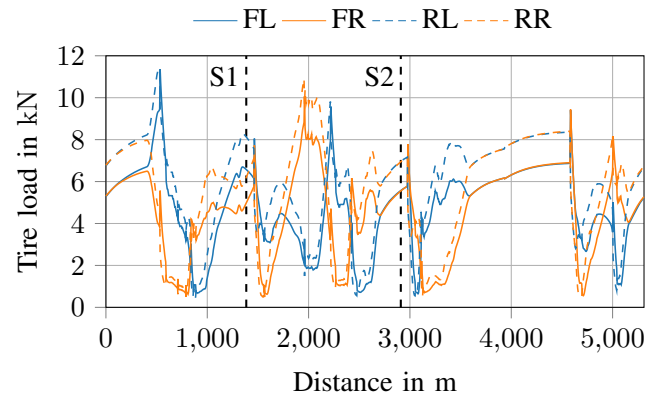


Fig. 3. Simulated tire loads for a F1 car on the Shanghai racetrack (FL = front left, FR = front right, RL = rear left, RR = rear right). S1 and S2 are the sector boundaries.

As depicted in Figure 4, the F1 powertrain consists of an ICE, an additional electric motor (MGU-K), an electrified turbocharger (MGU-H) and an electric energy storage (ES). Kinetic energy can be recuperated in the MGU-K during braking. The MGU-H can recuperate heat energy from the exhaust gases of the ICE. During acceleration, the recuperated energy can be used in the

MGU-K to provide additional torque (“boosting”). In a 2017 F1 car, the energy flows from ES to MGU-K and back are limited to 4 MJ/lap or 2 MJ/lap, respectively [19].

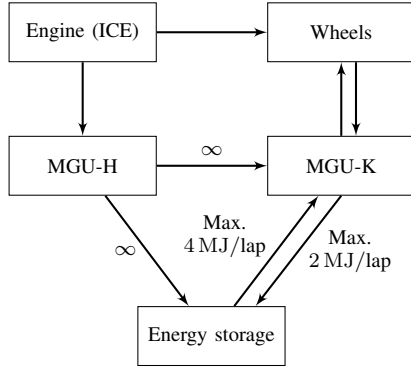


Fig. 4. Powertrain structure and energy flows of a 2017 F1 car.

This powertrain structure is the most general structure in use. It can easily be adapted to the other pure ICE and pure electric motors cases, with the omission of some components. The powertrain topology can be switched between RWD, FWD and AWD and is set to the former for F1 and FE. The power curve of the ICE is modeled by a third-order polynomial. It is fitted based on the maximum power P_{\max} acting at n_{\max} and the power drop P_{diff} appearing on both sides of the maximum power point at the beginning and end of the primarily used engine speed range n_{begin} and n_{end} . Below 75% of n_{begin} , the power level is kept constant. Figure 5 shows the output of the model.

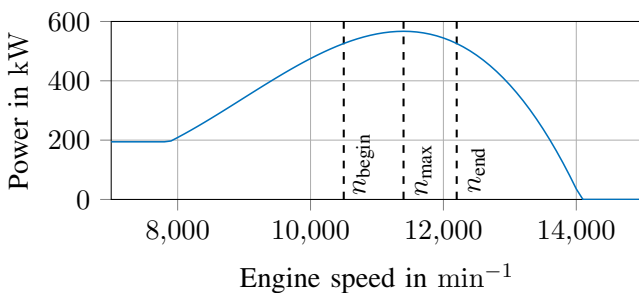


Fig. 5. Modeled power curve of the combustion engine of a 2017 F1 car.

The fuel flow in the i th raceline segment is calculated by

$$b_{e,i} = \sqrt{\frac{P_i}{P_{\max}}} b_{e,\max}. \quad (4)$$

The assumptions for this are that the maximum fuel flow $b_{e,\max}$ is reached at P_{\max} , and that the efficiency at full power is higher than at partial power. In F1, $b_{e,\max}$ is

limited to 100 kg h^{-1} by the regulations. In the electric parts of the powertrain, the efficiencies $\eta_{\text{MGU-K,boost}}$ and $\eta_{\text{MGU-K,re}}$ are considered when boosting (during acceleration) or recuperating (during deceleration) are performed with the MGU-K. Hereby, the MGU-K only acts on the driven axle(s). For the MGU-H, we assume that it recuperates the part $\eta_{\text{MGU-H,re}}$ of the energy that is supplied by the ICE during every acceleration segment. This happens under the assumption that the power curve of the ICE already includes the influence of the MGU-H.

The torque request is distributed between ICE and MGU-K in such a way that the MGU-K is not used for as long as the ICE can fully provide all of the requested torque at the current engine speed. Otherwise, the MGU-K provides boost if the charging state of the ES is sufficient and the energy management (EM) allows it. Therefore, for every discretization point on the raceline, the EM determines whether boosting should be used based on the underlying strategy. We also implemented a virtual accelerator pedal, which can be utilized to simulate the effect of driving under yellow flag conditions. Yellow flags indicate danger on the track, which is why the drivers have to slow down. As no fixed velocity limit exists, this cannot be considered within the track model.

One important aspect is the inclusion of DRS. Since the 2011 season, the driver can activate DRS under certain conditions to reduce aerodynamic drag on long straights, and thus facilitate overtaking due to the increased maximum velocity. Therefore, the normal drag coefficient $c_{w,A}$ is replaced by $c_{w,A,DRS}$ within the DRS zones that are supplied by the track model.

C. Solver

The solver calculates the velocity profile along the raceline with recourse to the vehicle model. The lap time can then be derived from this. Since the LTS should be able to run offline as well as online on a car, the main requirements are accurate results, fast computing time and robust convergence characteristics. For this reason, a quasi-steady-state approach is preferred over steady-state and transient approaches. It considers combined longitudinal and lateral accelerations and different radii for every discretization point, while the vehicle model is still regarded as being in a steady-state [3].

For quasi-steady-state modeling there are two common solver types: “forward/backward” (e.g. [5], [17]) and “pure forward”. The former searches for the minimum radius of every corner and then calculates two velocity profiles starting from the maximum possible velocity in this point, one forward and one backward. This is done considering the acceleration capabilities of

the car. The various parts along the raceline are finally intersected. As the name implies, the “pure forward” solver calculates only one velocity profile in forward direction. In every discretization point it tries to brake down to standstill within the next few raceline segments. If this is possible within the acceleration capabilities, it further accelerates, else it decelerates from the previous step. The “forward/backward” solver is faster but not as realistic because it uses “future” and therefore wrong lateral and longitudinal accelerations as well as velocities during the backward steps to calculate the velocities at preceding points. Furthermore, it is very susceptible to noisy curvature profiles, as they make it difficult to determine the apex exactly. By combining the working principles, our improved “forward/backward plus” solver increases accuracy in comparison to “forward/backward” and decreases computing time in comparison to “pure forward”. The underlying assumption for a reduction of the computing time is that a much larger part of the lap is accelerated rather than decelerated.

A simplified workflow of the solver is depicted in Figure 6. Starting with a given start velocity, the solver loops through the discretized curvature profile. At each point, it first calculates the acting lateral acceleration a_y for the current velocity. Together with a separately determined estimation of the longitudinal acceleration a_x in the current step, it is then used to calculate the tire loads, cp. (3), and thus the tire force potentials, cp. (2). For the car to be able to stay on track, the lateral acceleration forces $F_{y,f}$ and $F_{y,r}$ calculated by

$$\begin{aligned} F_{y,f} &= m a_y \frac{l_{\text{cog},r}}{l} \quad \text{and} \\ F_{y,r} &= m a_y \frac{l_{\text{cog},f}}{l} \end{aligned} \quad (5)$$

must stay below the according tire force potentials $F_{\text{tire,pot},f}$ and $F_{\text{tire,pot},r}$ of the front and rear axle. If this is the case, we stay within the forward loop. Working on the assumption that a race driver always uses the full potential of either the tire or the powertrain, the solver calculates the longitudinal acceleration force F_x as given by

$$F_{\text{tire,remain},f} = \sqrt{F_{\text{tire,pot},f}^2 - F_{y,f}^2}, \quad (6)$$

$$F_{\text{tire,remain},r} = \sqrt{F_{\text{tire,pot},r}^2 - F_{y,r}^2} \quad \text{and}$$

$$F_x = \min(F_{\text{tire,remain},f/r}, F_{\text{powertrain}}). \quad (7)$$

The powertrain topology determines which axles should be considered in (7). (6) comprises that the tire force potential is divided among longitudinal and lateral

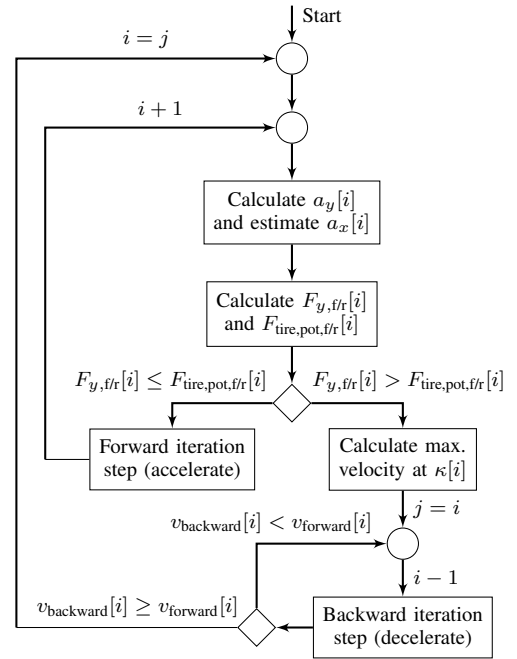


Fig. 6. Workflow of the “forward/backward plus” solver.

forces, according to the friction circle. F_x is used to obtain the longitudinal acceleration a_x for the upcoming track segment. Together with its step size s_{step} , the new velocity at the next point $i + 1$ can be derived by

$$v_{i+1} = \sqrt{v_i^2 + 2 a_x s_{\text{step}}}. \quad (8)$$

If the potential of one axle is exceeded by the respective lateral force, the car is too fast. In this case, we first calculate the maximum possible velocity at the current curvature κ_i . This is achieved within a separate loop, in which the car accelerates from a low velocity until the full tire potentials are used. The maximum possible velocity cannot be determined directly due to the mutual influence between aerodynamic forces and tire potentials. In longitudinal direction, the driven tires only transmit the force required to overcome drag and rolling resistance as we can assume $a_x = 0 \text{ m s}^{-2}$ at the apex. Afterwards, a temporary backward loop is started, in which the longitudinal deceleration potential of the preceding steps is used. This fully utilizes the tire potentials of both axles, under the assumption of an ideal brake force distribution. In each of these intermediate steps, the velocity as well as the acting longitudinal and lateral acceleration at the preceding point are again approximated in a separate loop. As before, this is necessary because velocity (and therefore aerodynamic downforce), longitudinal and lateral acceleration and tire loads and potentials influence each other. The loop stops as soon as the velocity calculated backward v_{backward} intersects

the originally calculated velocity v_{forward} . Subsequently, the forward loop is resumed at point j that originally caused the backward loop.

The proper start velocity of the lap is initially unknown. A solution for this is to temporarily add a small part of the lap in front of the actual lap. The size of this part must be long enough to ensure that the car decelerates at least once before entering the actual lap to be simulated.

The lap time can be calculated by summing up the time intervals $t_{\text{int},i}$ of every track segment obtained by

$$t_{\text{int},i} = 2 \frac{s_{\text{step}}}{v_i + v_{i+1}}. \quad (9)$$

Of course, the solver calculates many other variables as well, such as gears, engine speeds or energy consumption in the hybrid powertrain. These calculations are performed in a straightforward manner and are not, therefore, explained in detail here.

D. Parametrization

Some of the vehicle model parameters required can be derived from the technical regulations [19]. For the rest, techniques such as sound analysis and video analysis based on onboard video streams must be used. Velocity and engine speed profiles, as well as the gear choices can be automatically extracted from an onboard video stream by optical character recognition (OCR), for example. This data can also be used to draw further conclusions, such as regarding engine power and gear ratios, for example. The parameters applied for the F1 car in Shanghai can be found in Tables I and II.

IV. RESULTS

The entire simulation is implemented in Python 3 using the numerical math library NumPy. One simulation run for a lap in Shanghai (including the determination of the correct start velocity) takes about 1.3s on a laptop computer (Intel i7 2.7 GHz, 16 GB RAM). The raceline with a length of 5,310 m is discretized in steps of 5 m for this example. In literature, step sizes of 10 m are often used [8], [13]. This step size, however, leads to a significant deviation in lap times in our simulation. The following paragraphs present the validation of the LTS, as well as some interesting simulation results.

A. Validation

The simulation was validated in terms of velocity profiles obtained from onboard video streams of various qualifying laps of the 2017 season. Figure 7 compares

TABLE I. OVERVIEW OF THE REQUIRED SIMULATION PARAMETERS WITH EXEMPLARY VALUES OF A 2017 F1 CAR.

Parameter	Description	Example value
General		
m	Mass incl. driver and fuel	733 kg
l	Wheelbase	3.6 m
s	Trackwidth	1.6 m
$l_{\text{cog},r}$	Distance center of gravity to rear axle	1.632 m
h_{cog}	Height of center of gravity	0.335 m
$c_{w,A}$	Drag coefficient (incl. ref. area)	1.56 m ²
$c_{w,A,DRS}$	Drag coefficient (incl. ref. area) (DRS)	1.295 m ²
$c_{z,A,f}$	Lift coefficient front (incl. ref. area)	2.20 m ²
$c_{z,A,r}$	Lift coefficient rear (incl. ref. area)	2.68 m ²
ρ_{air}	Air density	1.18 kg m ⁻³
Engine		
n_{begin}	Engine speed (start of range used)	10,500 min ⁻¹
n_{max}	Engine speed (maximum power)	11,400 min ⁻¹
n_{end}	Engine speed (end of range used)	12,200 min ⁻¹
P_{max}	Maximum power at n_{max}	567 kW
P_{diff}	Power drop at n_{begin} and n_{end}	41 kW
$b_{e,\text{max}}$	Maximum fuel flow	100 kg h ⁻¹
$P_{\text{max,MGU-K}}$	Maximum power (MGU-K)	120 kW
$M_{\text{max,MGU-K}}$	Maximum torque (MGU-K)	200 N m
$\eta_{\text{MGU-K,boost}}$	Efficiency (MGU-K, boost)	0.9
$\eta_{\text{MGU-K,re}}$	Efficiency (MGU-K, recuperation)	0.15
$\eta_{\text{MGU-H,re}}$	Efficiency (MGU-H)	0.1
$v_{\text{min,MGU-K}}$	Minimum velocity to use MGU-K	100 km h ⁻¹
Gearbox		
i_{trans}	Transmission ratios	0.040
n_{shift}	Shift speeds	10,000 min ⁻¹
e_i	Torsional mass factors	1.16
η_g	Efficiency of gearbox and transmission	0.96
Tire		
f_{roll}	Rolling resistance	0.03
C_{tire}	Circumference of tire	2.073 m
$p_{1,f}$	Tire model parameter	1.66 N ⁻¹
$p_{2,f}$	Tire model parameter	-2.5e-5N ⁻²
$p_{1,r}$	Tire model parameter	2.03 N ⁻¹
$p_{2,r}$	Tire model parameter	-2.0e-5N ⁻²

TABLE II. GEARBOX SIMULATION PARAMETERS WITH EXEMPLARY VALUES OF A 2017 F1 CAR.

gear	1	2	3	4	5	6	7	8
i_{trans}	0.040	0.070	0.095	0.117	0.143	0.172	0.190	0.206
n_{shift} (min ⁻¹)	10,000	11,800	11,800	11,800	11,800	11,800	11,800	-
e_i	1.16	1.11	1.09	1.08	1.08	1.08	1.07	1.07

Hamilton's lap in Shanghai to the simulation result. In general, the profiles fit well together, e.g. in term of maximum and minimum velocities. At a distance of 3,950 m, reality and simulation show a small kink originating from DRS activation. However, it can be seen that in reality some passages can be passed much faster, e.g. at 2,050 m, 2,400 m and 3,300 m. This can be explained by different racelines. In reality, drivers often use the curbs in the corners to drive a raceline with a larger radius. Since we do not know the exact track widths including the curbs, such effects are not included in our raceline calculation. Another inaccuracy is that the braking points in the real data are somewhat earlier than in the simulation. The reason for this is the tire model, which currently assumes the same potentials in longitudinal and lateral direction. Further validation including velocity profile, engine

TABLE III. COMPARISON OF THE SECTOR TIMES OF A F1 CAR ON THE SHANGHAI RACETRACK.

	Simulation	Reality	Difference
Sector 1	24.437 s	24.036 s	0.401 s
Sector 2	27.222 s	27.079 s	0.143 s
Sector 3	39.847 s	40.563 s	-0.716 s
Lap time	91.506 s	91.678 s	-0.172 s

speed profile, gear choices and energy consumption was carried out against the professional LTS RaceSim [20] showing good correlation.

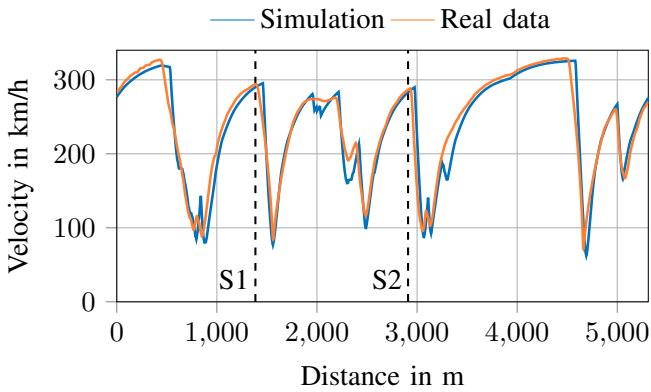


Fig. 7. Simulated velocity profile for a F1 car in comparison to the real profile of Hamilton's qualifying lap on the Shanghai racetrack. S1 and S2 are the sector boundaries.

For Shanghai, a lap time of 91.506 s is calculated under qualifying conditions, i.e., DRS is allowed and the car starts with a fully charged ES and applies the MGU-K wherever possible. In reality, Hamilton reached 91.678 s. The small difference shows accurate results can be achieved despite the fact that many parameters are derived from publicly available data. Table III shows that sectors 1 and 3 differ most. In sector 1 it can be traced back to a higher top speed on the straight and a faster passage through the chicane in reality. In sector 3 it is explainable by later braking points at the end of the straights in simulation.

We also simulated the Budapest, Monza and Spielberg racetracks. The aerodynamic coefficients as well as the transmission ratios were therefore slightly adapted to the track characteristics (Monza is a low downforce track while Budapest requires much more downforce) based on information provided by Pirelli [21]. For Monza we furthermore changed the friction value in (2) to $\mu = 0.65$ because the 2017 qualifying was held on a wet track. We found that the qualifying lap times calculated fell within a range of ± 1.5 s of the real results without tuning of other parameters. In conclusion, we can state that the most important aspects have been modeled and a valid parametrization example found. It should be noted,

however, that despite the simple models, the absolute value results depend significantly on the parameters being properly adjusted. Without insight into internal team data, they can only be regarded as reference values. Nevertheless, these values are usually less important than the relative changes between multiple simulation runs with varying inputs to determine whether a setup change was advantageous or disadvantageous, or to obtain a sensitivity, such as the lap time mass sensitivity.

B. Sensitivity analysis for a Formula 1 car

In addition to the velocity profile, a great deal of additional data is output by the LTS, such as the tire loads, engine speeds and charging state of the ES. The simulation is therefore well suited for sensitivity analyses, for instance in order to quantify the influence of vehicle mass on lap time and energy consumption. An example is shown in Figure 8 proving that the relationship between mass and lap time is almost linear when the relevant range from 730 kg to 830 kg is considered. The lap time mass sensitivity is 0.062 s kg^{-1} in this case.

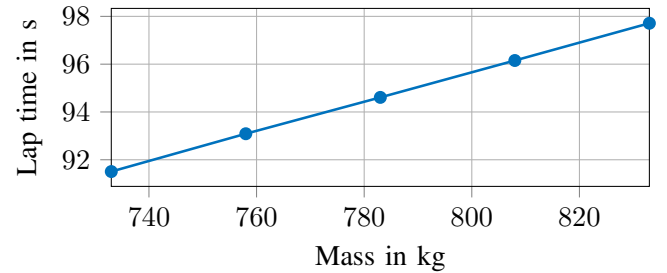


Fig. 8. Simulated influence of mass on lap time for a F1 car on the Shanghai racetrack.

C. Energy management strategies for a Formula 1 car

Another important aspect is the evaluation of different energy management strategies. The key question is where on the track the MGU-K boost must be used to obtain the biggest lap time advantage from the limited energy available. The energy management strategy can be entered into the simulation by specifying whether or not boosting is allowed for each discretization point. We evaluate three exemplary strategies: “first come, first boost”, “longest time to braking point” and “lowest speed”. The first one uses the boost as soon as the tires allow it and until the energy is exhausted. The second strategy is based on the idea that the velocity advantage gained by the additional energy is used for as long as possible before braking. “Lowest speed” follows the approach that a given energy input allows a greater speed advantage at low speeds

TABLE IV. COMPARISON OF DIFFERENT ENERGY MANAGEMENT STRATEGIES FOR THE F1 HYBRID SYSTEM IN SHANGHAI WITH A LIMITED AMOUNT OF ELECTRIC ENERGY AVAILABLE.

Energy management strategy	t_{lap}	Δt_{lap}	m_{fuel}
No boost at all	94.445 s	1.320 s	1.93 kg
First come, first boost	93.602 s	0.477 s	1.90 kg
Longest time to braking point	93.134 s	0.009 s	1.90 kg
Lowest speed	93.125 s	-	1.89 kg

than at high speeds. To be able to apply the second and third energy management strategy, we first run one solver iteration without boosting to obtain an initial velocity profile. This profile is then used to calculate where the boost is applied for the respective strategy. Due to the mutual influence between velocity profile and boosting, a few additional iterations must be performed. The final result is found as soon as no further changes occur in the ES state at the end of the lap. This usually requires one to three iterations.

The F1 car simulated starts with 2 MJ of electrical energy in the ES and deactivated recuperation to find out how the available energy can be used most efficiently. As Table IV shows, “lowest speed” gives the best lap time and lowest fuel consumption on the Shanghai racetrack. Further simulations with other racetracks indicate that either “lowest speed” or “longest time to braking point” always results in the fastest lap time. It follows that the energy used shortly before a braking point brings very little advantage. This insight is often used in motorsports, where the drivers disengage the throttle shortly before the end of a straight and “sail” for a short time. This technique is known as “lift & coast”.

D. Simulation of a Formula E car

In contrast to Formula 1 cars, Formula E cars are made up of many standard components. In the 2017/2018 season, the teams were only allowed to develop electric motors, power electronics, gearbox and rear suspension. Many of the parameters required can, therefore, be obtained from the FIA regulations [19]. Table V contains the parameters used for the simulation.

The car is simulated on the Berlin FE racetrack with a raceline length of 2,302 m. Compared to F1, the FE racetracks are shorter and narrower, as they are usually built within cities especially for this event. The calculated qualifying lap time of 69.656 s is, once again, close to the 69.620 s achieved by Lucas Di Grassi in the 2017/2018 event.

For FE teams, it is important to know how much energy a car consumes during a lap and how this changes with the application of the lift & coast technique or under

TABLE V. EXEMPLARY SIMULATION PARAMETERS OF A 2017/2018 FE CAR.

Parameter	Description	Example value
General		
m	Mass incl. driver	880 kg
l	Wheelbase	3.1 m
s	Trackwidth	1.3 m
$l_{\text{cog,r}}$	Distance center of gravity to rear axle	1.194 m
h_{cog}	Height of center of gravity	0.345 m
$c_{w,A}$	Drag coefficient (incl. ref. area)	1.15 m ²
$c_{z,A,f}$	Lift coefficient front (incl. ref. area)	1.24 m ²
$c_{z,A,r}$	Lift coefficient rear (incl. ref. area)	1.52 m ²
Engine		
$P_{\text{max,MGU-K}}$	Maximum power (MGU-K)	200 kW
$M_{\text{max,MGU-K}}$	Maximum torque (MGU-K)	150 N m
$\eta_{\text{MGU-K,boost}}$	Efficiency (MGU-K, boost)	0.9
$\eta_{\text{MGU-K,re}}$	Efficiency (MGU-K, recuperation)	0.9
Gearbox		
i_{trans}	Transmission ratios	0.056, 0.091
n_{shift}	Shift speed	19,000 min ⁻¹
e_i	Torsional mass factors	1.04, 1.04
η_g	Efficiency of gearbox and transmission	0.96
Tire		
f_{roll}	Rolling resistance	0.02
c_{tire}	Circumference of tire	2.168 m
$p_{1,f}$	Tire model parameter	1.22 N ⁻¹
$p_{2,f}$	Tire model parameter	-2.5e-5N ⁻²
$p_{1,r}$	Tire model parameter	1.42 N ⁻¹
$p_{2,r}$	Tire model parameter	-2.0e-5N ⁻²

yellow flag conditions. As with the yellow flags, lift & coast is integrated into the simulation via the virtual accelerator pedal. Therefore, it is set to 0% 20 m before each braking point. Under yellow flag conditions, it is set to 30%. Figures 9 and 10 show the effects of lift & coast application and yellow flag in sector 2 on velocity and ES state profiles for one lap in Berlin. With lift & coast, the lap time calculated is increased by 0.488 s. However, as can be seen in the ES state graph, applying the energy saving technique is the only way the driver can stay within the average energy available per lap. This means he has to drive many economical laps in order to be able to drive one lap at full power, for example to overtake another driver. With a yellow flag, there is much more energy left at the end of the lap.

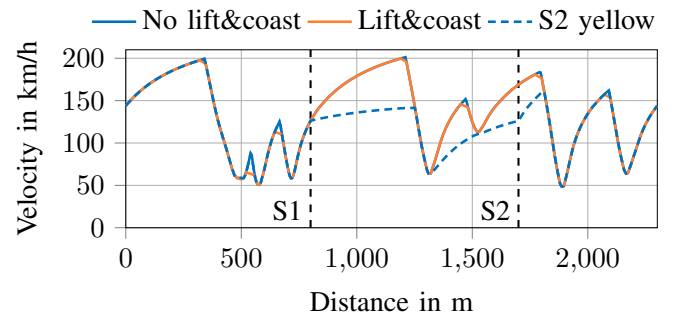


Fig. 9. Simulated velocity profiles for a FE car on the Berlin racetrack with and without using lift & coast as well as under yellow flag conditions in sector 2. S1 and S2 are the sector boundaries.

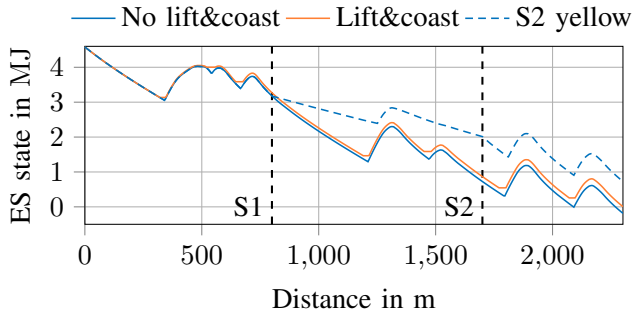


Fig. 10. Simulated ES state profiles of a FE car on the Berlin racetrack with and without using lift & coast as well as under yellow flag conditions in sector 2. The initial ES charge state was set to 4.58 MJ, which equals the energy a car has available for one lap in Berlin, on average. S1 and S2 are the sector boundaries.

V. DISCUSSION

The results show not only the functionality of our LTS, but also exemplary application cases. The LTS can be used by a race engineer to virtually optimize the vehicle setup and energy management strategy of a race car, or co-simulate a race to optimize race strategy. This option is especially suitable due to the fast computing time it enables, as well as the fact that it also allows speed limits and yellow flags to be considered. Compared to other published LTS, our proposal is easy to parametrize and has a solver that provides a good compromise between computing time and accuracy. In addition, it can easily be adapted to various racing series with different powertrain structures and topologies and includes the current state of technology. To increase model accuracy further, we would require a much more precise knowledge of vehicle parameters. Therefore, this is not suitable for our application.

There are, however, some limitations in the present state. Firstly, the tire force potential is currently assumed to be equal in longitudinal and lateral direction. This is not true for real tires. Secondly, gear changes occur infinitely fast. As a result, the model sometimes changes gears where a real driver would not. Thirdly, the modeling of MGU-K and MGU-H is kept very simple in the present state. For example, the MGU-K is only used if the torque supplied by the ICE is exhausted. In practice, it makes sense to keep the load point of the ICE in the highest efficiency range for as long as possible. As a result, the energy consumption calculation cannot be assumed to be exact. Fourthly, the maximum change in longitudinal acceleration within two consecutive segments is currently not constrained. This leads to inaccuracies in the transition points between acceleration and deceleration phases. Lastly, the energy management strategy calculations currently need a few

additional solver iterations to find a valid result, due to the mutual influence between velocity profile and boosting strategy. This slightly reduces the suitability for applications with high computing time requirements.

In our future work, we will, therefore, concentrate on lifting the mentioned limitations, as well as on extending the vehicle and track library of the LTS. We also want to automate the parameter fitting process to be able to better validate the simulation models and become familiar with the error range of the simulation results. Finally, the LTS shall be used together with the mentioned race simulation to optimize race strategy.

ACKNOWLEDGMENTS AND CONTRIBUTIONS

Research was supported by the basic research fund of the Chair of Automotive Technology of the Technical University of Munich. Alexander Heilmeier leads the research project and contributed to the functional development of the lap time simulation. Maximilian Geisslinger wrote his term thesis within the project, contributed to the methodology of the lap time simulation and performed parts of the data analysis to obtain the exemplary simulation parameters. Johannes Betz contributed to the conception of the research project and revised the paper critically.

We would like to thank Steffen Kosuch for providing a free trial license for the professional LTS RaceSim [20].

REFERENCES

- [1] M. Westerhoff, *Formula 1 Engine from Mercedes with over 50 Percent Efficiency*, <https://www.springerprofessional.de/engine-technology/race-cars/formula-1-engine-from-mercedes-with-over-50-percent-efficiency/15061334>, 2017, accessed: 2019-02-10.
- [2] A. Heilmeier, M. Graf and M. Lienkamp, *A Race Simulation for Strategy Decisions in Circuit Motorsports*, 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 2986-2993, Maui, Hawaii, USA, 2018. DOI: 10.1109/ITSC.2018.8570012.
- [3] B. Siegler, A. Deakin and D. Crolla, *Lap Time Simulation: Comparison of Steady State, Quasi-Static and Transient Racing Car Cornering Strategies*, SAE Technical Paper Series, no. 2000-01-3563, 2000.
- [4] T. Völkl, *Erweiterte quasistatische Simulation zur Bestimmung des Einflusses transienten Fahrzeugverhaltens auf die Rundenzeit von Rennfahrzeugen*, PhD Thesis, Technical University of Darmstadt, 2013.
- [5] D. Brayshaw and M. Harrison, *A quasi steady state approach to race car lap simulation in order to understand the effects of racing line and centre of gravity location*, Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, vol. 219, no. 6, pp. 725-739, 2005. DOI: 10.1243/095440705X11211.

-
- [6] I. F. Colunga and A. Bradley, *Modelling of transient cornering and suspension dynamics, and investigation into the control strategies for an ideal driver in a lap time simulator*, Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, vol. 228, no. 10, pp. 1185-1199, 2014. DOI: 10.1177/0954407014525362.
- [7] D. Casanova, *On Minimum Time Vehicle Manoeuvring: The Theoretical Optimal Lap*, PhD Thesis, Cranfield University, School of Mechanical Engineering, 2000.
- [8] D. P. Kelly, *Lap Time Simulation with Transient Vehicle and Tyre Dynamics*, PhD Thesis, Cranfield University, School of Engineering, Automotive Studies Group, 2008.
- [9] J. Timings and D. Cole, *Robust lap-time simulation*, Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, vol. 228, no. 10, pp. 1200-1216, 2014. DOI: 10.1177/0954407013516102.
- [10] G. Perantoni and D. J. N. Limebeer, *Optimal Control of a Formula One Car on a Three-Dimensional Track Part 1: Track Modeling and Identification*, Journal of Dynamic Systems, Measurement, and Control, vol. 137, no. 2, p. 21010, 2014. DOI: 10.1115/1.4028253.
- [11] D. J. N. Limebeer and G. Perantoni, *Optimal Control of a Formula One Car on a Three-Dimensional Track Part 2: Optimal Control*, Journal of Dynamic Systems, Measurement, and Control, vol. 137, no. 5, p. 51019, 2015. DOI: 10.1115/1.4029466.
- [12] D. J. N. Limebeer, G. Perantoni and A.V. Rao, *Optimal control of Formula One car energy recovery systems*, International Journal of Control, vol. 87, no. 10, pp. 2065-2080, 2014. DOI: 10.1080/00207179.2014.900705.
- [13] S. Ebbesen, M. Salazar, P. Elbert, et al., *Time-optimal Control Strategies for a Hybrid Electric Race Car*, IEEE Transactions on Control Systems Technology, vol. 26, no. 1, pp. 233-247, 2018. DOI: 10.1109/TCST.2017.2661824.
- [14] Roborace Ltd., *Roborace*, <https://roborace.com>, accessed: 2019-02-10.
- [15] J. Betz, A. Wischnewski, A. Heilmeyer, et al., *What can we learn from autonomous level-5 Motorsport?*, 9th International Munich Chassis Symposium 2018, Springer Vieweg, Wiesbaden, 2019. DOI: 10.1007/978-3-658-22050-1_12.
- [16] OpenStreetMap Foundation, *OpenStreetMap*, <https://www.openstreetmap.org>, accessed: 2019-02-10.
- [17] F. Braghin, F. Cheli, S. Melzi, et al., *Race Driver Model*, Computers and Structures, vol. 86, no. 13-14, pp. 1503-1516, 2008. DOI: 10.1016/j.compstruc.2007.04.028.
- [18] L. Papula, *Mathematische Formelsammlung*, Springer Vieweg, Wiesbaden, 2017. DOI: 10.1007/978-3-658-16195-8.
- [19] Fédération Internationale de l'Automobile, *Regulations*, <https://www.fia.com/regulations>, accessed: 2019-02-10.
- [20] D.A.T.A.S. Ltd., *Data Analysis Tools & Simulation*, <http://www.datas-ltd.com>, accessed: 2019-02-10.
- [21] Pirelli & C. S.p.A., *F1 and Motorsport infographics*, <https://racingspot.pirelli.com/global/en-ww/infographics>, accessed: 2019-02-10.

4.3.2 Revision of the Tire Force Calculation

Compared to its state in the publication, the calculation of the transmissible tire forces in the LTS has been revised later on. First, a nominal wheel load $F_{z,0}$ was added to the tire model to make its parameterization less dependent on the wheel load range, which is relevant in motorsport due to downforce. The tire force potential in longitudinal direction $F_{x,\text{pot}}$ is now calculated by:

$$F_{x,\text{pot}} = \mu_w \cdot F_z \cdot \mu_x(F_z) = \mu_w \cdot F_z \left(\mu_{x,0} + \frac{\partial \mu_x}{\partial F_z} (F_z - F_{z,0}) \right). \quad (4.1)$$

μ_w serves to modify the coefficient of friction due to weather conditions. The coefficient of friction of the tire μ_x is composed of the value at nominal load $\mu_{x,0}$ and a term that modifies it depending on how much the wheel load deviates from the nominal load. Since $\frac{\partial \mu_x}{\partial F_z}$ is negative, the transmissible tire force increases less than the wheel load in accordance with the effect of wheel load degressivity. In the lateral direction, the same relations apply (with their own parameters).

Second, the calculation of the maximum possible braking acceleration in the LTS now takes into account that the weaker wheel limits the transmissible braking force for both wheels of an axle. This is done under the assumption that the driver avoids a braking force demand exceeding the transmissible force so that none of the wheels locks and thus the tire is damaged.

The latest release of the LTS is available on GitHub [165].

4.4 Race Simulation

4.4.1 Deterministic Race Simulation

This section summarizes the work carried out on the deterministic part of the RS that has been published in [8]. The RS is available on GitHub [167].

Summary of the Paper

The publication presents a RS that is based on a lap-wise discretization, i.e., the race is simulated lap after lap. This principle combines the advantages of fast computing times, acceptable parameterization effort, easy extensibility, and sufficiently accurate results. In each lap, the lap times of all drivers are calculated by summing up different time parts. Starting from a basic lap time, which represents the lap time a car can drive under perfect conditions, i.e., with little fuel mass and fresh tires, the following influences are added: car- and driver-specific performance drawbacks, race start from a standstill and from different grid positions, pit stops (which affect both in-lap and out-lap), tire degradation (dependent on tire age and compound), and the fuel mass aboard the car. Comparing the expected race times of consecutive drivers at the end of each lap allows modeling driver interactions. If the theoretical time advantage of an attacking driver is large enough, an overtaking maneuver is simulated. The required time advantage represents the difficulty of overtaking on a particular race track.

The implementation combines various parts already described in the literature and extends them by additional aspects. One of these aspects is that the simulation ensures that the minimum distances between cars are maintained even after a pit stop when drivers return from the pit lane to the race track, which improves the realism and has not been considered so far. Other

implemented improvements are introducing a time loss for the overtaking driver during an overtaking maneuver and the possibility of considering team orders.

Relation to the Research Questions

The publication is related to the first research question. The developed RS enables the user to simulate circuit races in order to evaluate the effects of different race strategies on the race result. The RS was designed to meet the specific requirements in the context of race strategy determination, e.g., fast computing times. Probabilistic effects on the race were not modeled at the time of this publication. This was left to the subsequent publication.

Individual Contribution

Heilmeier developed and implemented the RS based on the literature and an introduction to the methodology of RS by Graf. Furthermore, he conducted the literature research and the validation of the simulation. Graf performed the data analysis to obtain the example simulation parameters.

Imprint of the Paper

©2018 IEEE. Reprinted, with permission, from Alexander Heilmeier, Michael Graf, and Markus Lienkamp, A Race Simulation for Strategy Decisions in Circuit Motorsports, 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018.

A Race Simulation for Strategy Decisions in Circuit Motorsports

Alexander Heilmeier¹ and Michael Graf² and Markus Lienkamp¹

Abstract— In motorsports, every racing participant pursues the goal to finish the race in the shortest time possible. There are several ways to influence the race with race strategy decisions, e.g. the timing of pit stops and the choice of tires. In order to be able to evaluate the race strategy before and to quickly adjust it during a race, a tool is required that simulates an entire race in a short time. For this purpose, the paper presents the methodology of a race simulation for circuit motorsports. It simulates races in dependency of various race strategy inputs and is based on a lap-wise discretization. It includes the effects of tire degradation, fuel mass loss, pit stops and overtaking maneuvers. The simulation parameters are chosen in such a way that they can be determined based on publicly accessible lap time data. The Formula 1 2017 Abu Dhabi Grand Prix is analyzed as an exemplary result. An application area is the support of race engineers in present motorsports. Future work aims towards the automation of strategy decisions in motorsports with autonomous racecars.

I. INTRODUCTION

In circuit motorsports, the race participants drive a certain number of laps against each other on a closed racetrack. The cars are built according to technical regulations that are dependent on the racing series, e.g. Formula 1 or Le Mans Prototype (LMP) in the World Endurance Championship (WEC). The better the rank position at the end of the race, the more points the driver and his team receive. Often two or more cars are operated per team. The points are accumulated over a season to determine the winners of the driver and team championships at the end. Usually a race weekend starts with several practice sessions. Afterwards, in a qualifying session, the starting order of the race is determined based on the best lap time. During a race, drivers can come to their pit to refuel, change tires or repair broken parts of the car. This results in a time loss compared to the other drivers that continue the race.

The timing of a pit stop and the determination of the actions to be carried out in it is part of the race strategy. It is a key factor for a satisfying race outcome. Porsche [1] give an overview of the topics covered by race strategy in the context of endurance races. According to this, it includes the consideration of tire degradation, fuel mass loss, pit stops and the reaction to safety cars and yellow flags. Furthermore, it takes the driver and the pit crew into account. As Fig. 1 visualizes, we divide these aspects into three categories: pit stops, driving strategy and response to race events. Driving strategy includes how aggressively the driver drives the car.

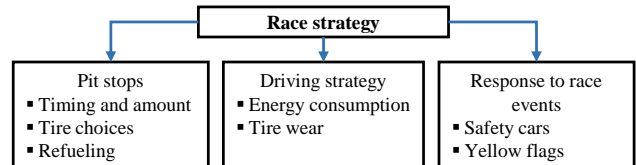


Fig. 1. Aspects of race strategy.

This affects not only lap times but also tire wear and energy consumption.

Race strategies of different racing series differ greatly because they depend heavily on technical and sporting regulations [2]. However, the common characteristic of circuit races is that every race participant tries to finish the race in the shortest time possible. The well-founded definition of a race strategy therefore requires a tool that simulates the race based on various strategy inputs. These tools are known as race simulations.

II. RELATED WORK

At first, we differentiate race simulations from the more popular lap time simulations. The most important aspects are shown in Fig. 2. The goal of a lap time simulation is to calculate the exact lap time with the current car setup. Therefore, it simulates one lap with one car neglecting long-term effects. The underlying models are mostly physically motivated. To summarize, a lap time simulation provides a microscopic view on one race lap. Siegler [3] give an overview of three different types of lap time simulations and compare them. Further research is carried out for a quasi-steady-state approach in Brayshaw [4], for a transient approach in Colunga [5] and for a robust approach based on model predictive control in Timings [6].

A race simulation, in contrast, has a macroscopic view on the race, including race events such as pit stops and long-term effects such as tire degradation. All participating cars and their interactions are simulated together for all the laps. This is done with empirical models to keep the calculation times and amount of required simulation parameters in usable limits. The globally relevant vehicle characteristics are implicitly contained in the simulation parameters, since a fast car is represented by a fast lap time. The goal is to calculate the final race durations of all participants, e.g. for the evaluation of a race strategy. However, both types of simulation work closely together because the lap time simulation can provide many of the required parameters for the race simulation.

¹Alexander Heilmeier and Markus Lienkamp are with Chair of Automotive Technology, Faculty of Mechanical Engineering, Technical University of Munich, Garching, Germany alexander.heilmeier@tum.de

²Michael Graf is with BMW Motorsport, Munich, Germany michael.gm.graf@bmw-motorsport.com

Lap time simulation	Race simulation
<ul style="list-style-type: none"> ▪ Microscopic view on one lap neglecting long-term effects ▪ Physically motivated models (e.g. two-track car model, engine model) ▪ One car simulated for one lap ▪ Goal: Calculation of the exact lap time with the current configuration, e.g. to evaluate the setup 	<ul style="list-style-type: none"> ▪ Macroscopic view on the whole race including long-term effects ▪ Empirical models (e.g. tire degradation model, overtaking model) ▪ All participating cars simulated for the whole race with interactions ▪ Goal: Calculation of the final race times, e.g. to evaluate the race strategy

Fig. 2. Comparison between lap time simulation and race simulation.

Little literature can be found when it comes to race strategy. Tulabandhula [7] investigate how to support tire-changing decisions in the NASCAR series based on machine learning. Hirst [8] writes about the analysis and visualization of data for race strategy decisions in Formula 1. McLaren [9] address the fuel mass loss calculation and the pit time loss in a small example. Farroni [10] look at the modelling of tire wear and its effect to the car's performance. Bekker [11] published a holistic race simulation suitable for race strategy evaluation. They use a time-based approach combined with stochastic elements and include passing maneuvers, refueling during pit stops and car failures. In the internet, Porsche [1] give an overview of the facets of race strategy. Bi [12] highlights the significance of big data for race predictions in Formula 1. An important source is Phillips [13], who presents a race simulation for the Formula 1 2014 season on his blog. It uses some of Bekker's ideas, e.g. the time-based approach, and adds missing parts such as a tire degradation model.

III. METHODOLOGY

Our main demands for a race simulation are robust results and a fast calculation time. The former can be achieved by modeling all relevant effects. The latter allows running many simulations for the race preparation on the one hand and the live usage of the simulation during a race on the other hand. However, it does not need to be extremely accurate because the race situation usually changes a few times during a race anyway.

When analyzing the implementation of Bekker [11], there are several reasons why it is unsuitable for our use case. Firstly, the very relevant effect of tire degradation is not included. Secondly, the racetrack is divided into approximately 40 sectors with a length 150 m each. Each of them must be parameterized individually, e.g. with its lap time proportion, fuel consumption proportion and if passing maneuvers are allowed. This is difficult or impossible to do based on publicly accessible data. Thirdly, passing maneuvers must be completed within one sector in order to take place. This means that overtaking on long straights is not simulated realistically. Moreover, it is difficult to set the right minimum

time gap required for overtaking for every sector. In addition, only one car can overtake at a time, whereas in reality a slow driver is often overtaken by two cars at once. Fourthly, the Drag Reduction System (DRS) is missing, as the paper was published before its introduction to Formula 1. Since the 2011 season, the driver can activate it under certain conditions to reduce drag on long straights and thus facilitate overtaking.

Due to our requirements and experience, the same applies to the implementation of Phillips [13]. Firstly, he primarily uses average lap times from the second free practice session as basic lap time for the simulation. The disadvantage of this method is the unknown fuel state of the cars. Secondly, he considers only two rubber compounds for his tire degradation model. He assumes that the softer compound is 0.7 s per lap faster for a fresh tire and wears twice as fast as the harder compound. Thirdly, the minimum time gap between two cars is only checked at the end of a lap. However, it is essential to check this again after the completion of the pit stops at the beginning of a lap because drivers may drive back onto the track very close in front or behind another driver. Fourthly, Phillips only considers a time loss for the overtaken driver. Based on our experience, it is important to also include a time loss for the overtaking driver as usually both fight for their positions. Fifthly, he uses a constant time penalty per lap in terms of fuel in his fuel mass model. However, it makes more sense to split the calculation into separate parts. Neither Bekker nor Phillips implement any possibility to consider team orders in the simulation.

Subsequently, we present the principle of our race simulation, which addresses these aspects. After an overview has been given, the submodels are discussed in more detail. The paper concentrates on Formula 1 because it is the most popular racing series. Nevertheless, the race simulation can be adapted to other racing series as well. A basic principle of our design is to keep it as simple as possible and as detailed as necessary. The more detailed the models are, the more input data is required to feed them. This is particularly problematic for us because we are limited to the small amount of data that the FIA [14] (International Automobile Federation) makes publicly accessible.

A. Overview of the race simulation

The race simulation is performed lap by lap, as Fig. 3 shows. There are several reasons for discretizing the race in laps instead of time. The most important are that a time discretization would need a lot more input data and the computational effort would be much higher. This is because the various competitors would have to be simulated with their individual racing lines on the track. In contrast, for lap discretization, lap times are sufficient as a basis. The downside of lap discretization is that it prevents the more detailed modeling of some aspects. Overtaking, for example, could be implemented more precisely by simulating the individual racing lines. However, as mentioned previously, this is not necessary.

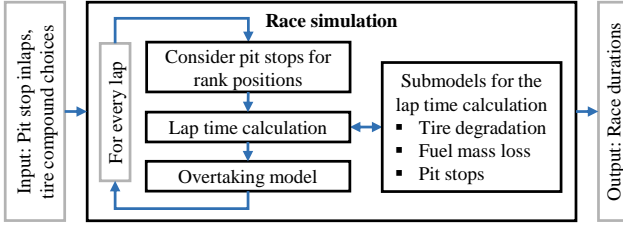


Fig. 3. Race simulation workflow.

As with Bekker [11] and Phillips [13], the user is required to provide the pit stop information for every driver as an input. This includes the expected pit stop inlaps as well as the tire compound choices. It should be emphasized that the race simulation simulates the race depending on the input, but does not optimize it. The output of the simulation are the race durations and thus also the rank positions of all the drivers at the end of the race. At the beginning of each lap, the simulation must check for position changes due to pit stops that began in the previous lap and end in the current lap. Afterward, it calculates the current lap time for every competitor starting from a base lap time. Time losses and gains due to tire wear, fuel mass loss, pit stops and other factors are then added. In the next step, the algorithm checks whether overtaking occurs in the current lap based on the computed lap times and the time gaps between the cars after the previous lap. Overtaking influences driver rank positions as well as their lap times. Afterward, the final lap times and rank positions are known and the next lap can be simulated. This is repeated until the race is completed.

B. Race simulation without competitor interaction

In the first approach, interactions between the competitors are omitted. Therefore, every car can be simulated by itself. The race time after a lap

$$t_{\text{race,currentlap}} = \sum_{\text{lap}=1}^{\text{currentlap}} t_{\text{lap}}(\text{lap}) \quad (1)$$

follows from the summation of the separate lap times t_{lap} up to this lap. As Eq. 2 shows, the lap times are summed up based on several time elements that are explained in the following paragraphs.

$$\begin{aligned} t_{\text{lap}}(\text{lap}) = & t_{\text{base}} + t_{\text{tire}}(a_{\text{tire}}, c_{\text{tire}}) + t_{\text{fuel}}(\text{lap}) \\ & + t_{\text{car}} + t_{\text{driver}} + t_{\text{grid}}(\text{lap}) \\ & + t_{\text{pit,inlap/outlap}}(\text{lap}) \end{aligned} \quad (2)$$

1) *Base lap time t_{base}* : The base lap time is the basis for all lap time calculations. McLaren [9], Bekker [11] and Phillips [13] use a similar approach. However, we define it as the lap time the best racecar of a given racing series needs on a given track for one lap under perfect conditions during the race. To obtain it, the fastest qualifying lap time t_Q is utilized. During the qualifying, optimal and comparable conditions can be assumed because the car setups are finalized, they carry a minimum amount of fuel, the tires are in perfect

condition and the drivers try to run an optimal lap. Another (but usually less accurate) source is a lap time simulation. As displayed in Eq. 3, the expected delta time between qualifying and race $t_{\text{gap,racepace}}$ of the fastest qualifying driver is then added in order to take into account the fact that the race pace is slower than the qualifying pace. This mainly occurs due to engine durability and fuel consumption limitations. The primary source for the delta time are the free practice sessions. With this approach, we obtain a robust basis for the lap time calculations. In addition, it allows us to quickly modify the lap times for all the drivers at once during a race, e.g. because of bad weather conditions.

$$t_{\text{base}} = t_Q + t_{\text{gap,racepace}} \quad (3)$$

2) *Lap time loss due to tire degradation t_{tire}* : A racing tire runs through different phases during a race. Usually, the warm-up period is followed by a short peak performance phase. Afterward, one can observe an almost constant performance decrease before the tire completely degrades at the end of its life [10]. This effect is called tire degradation [1], [10]. In Formula 1, Pirelli supplies three out of seven different compounds for dry conditions per race ranging from Hypersoft to Superhard [15]. Typically, the harder the compound, the slower it will be at the beginning [16], but the longer its stable performance phase lasts. Tire degradation can be expressed as lap time loss t_{tire} over tire age a_{tire} in laps. Therefore, the simulation provides empirically based logarithmic and linear formulas for the different rubber compounds c_{tire} :

$$\begin{aligned} t_{\text{tire,log}}(a_{\text{tire}}, c_{\text{tire}}) = & \log(a_{\text{tire}} \cdot k_{1,\text{log}}(c_{\text{tire}}) + 1) \\ & \cdot k_{2,\text{log}}(c_{\text{tire}}) + k_3(c_{\text{tire}}) \end{aligned} \quad (4)$$

$$t_{\text{tire,lin}}(a_{\text{tire}}, c_{\text{tire}}) = a_{\text{tire}} \cdot k_{2,\text{lin}}(c_{\text{tire}}) + k_3(c_{\text{tire}}) \quad (5)$$

Fig. 4 shows an example for two compounds modelled with Eq. 4. Due to our experience, a logarithmic function represents the tire behavior in the usable range better than the quadratic function that Phillips [13] uses. The simpler linear model is used, if there is only a little data available. The equations show that the degradation factors $k_{1,\text{log}}$ and $k_{2,\text{log}}$ respectively $k_{2,\text{lin}}$ as well as the time offset k_3 depend on the tire compound. k_3 expresses the basic time loss that appears for the medium and harder compounds in comparison to the softer tire [16]. The models are parameterized based on (fuel mass corrected) timing data from the free practices, qualifying and previous races. They are adjusted independently for every driver or at least every team because the degradation is heavily influenced by driving style and vehicle balance.

3) *Lap time loss due to fuel mass t_{fuel}* : For combustion-powered cars, the additional fuel mass leads to significantly worse lap times at the beginning of a race. As it progresses, time loss decreases due to consumed fuel. In Formula 1, there is a fuel mass loss of about 100 kg during one race. The lap time loss that results from this effect can be calculated by subtracting the consumed fuel mass $m_{\text{fuel,consumed}}$ from the total fuel mass $m_{\text{fuel,tot}}$ and then multiplying it with the mass sensitivity of the lap time $s_{\text{lap,mass}}$ [9], Eq. 6. $m_{\text{fuel,consumed}}$

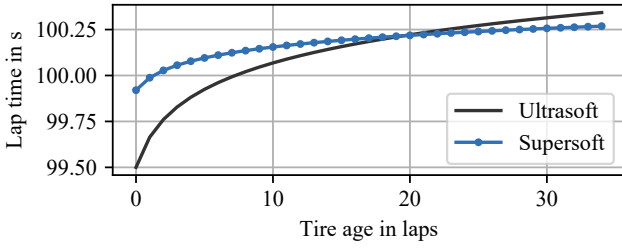


Fig. 4. Lap time effect of the tire degradation modelled with a logarithmic function for the Ultrasoft and Supersoft compounds. As of lap 20, the harder Supersoft leads to a better lap time. This is usually the point where one would pit at the latest to replace the Ultrasoft tires in order to avoid losing time against the Supersoft.

is derived from the fuel consumption per lap $B_{\text{fuel,perLap}}$ as shown in Eq. 7.

$$t_{\text{fuel}}(\text{lap}) = (m_{\text{fuel,tot}} - m_{\text{fuel,consumed}}(\text{lap})) \cdot s_{\text{lap,mass}} \quad (6)$$

$$m_{\text{fuel,consumed}}(\text{lap}) = B_{\text{fuel,perLap}} \cdot \text{lap} \quad (7)$$

The mass sensitivity of the lap time and the fuel consumption per lap are usually obtained from a lap time simulation using an engine consumption map. The subdivision of this effect into a mass analysis and the sensitivity allows us to easily include refueling during pit stops in those racing series where it is allowed. Side effects of the fuel mass occur in reality but are currently disregarded in the simulation, e.g. its effect on the tire degradation.

4) *Lap time loss due to car abilities t_{car} and driver abilities t_{driver}* : In racing series without unified cars, there are obviously differences in the lap times of the different manufacturers, even if all other factors were identical. The same applies to different drivers. Therefore, the simulation considers a constant time offset for both. They can be estimated from the qualifying session or previous races, for example.

5) *First lap time loss due to starting grid position t_{grid}* : Using the same model as Phillips [13], a handicap is added for every driver based on his grid position p_{grid} at the race start, because cars at the end of the grid are further away from the starting line. t_{firstlap} is added because the first lap takes longer than the following laps due to starting from a standstill:

$$t_{\text{grid}} = p_{\text{grid}} \cdot t_{\text{perGridPos}} + t_{\text{firstlap}} \quad (8)$$

6) *Lap time loss due to pit stops $t_{\text{pit,inlap/outlap}}$* : Pit stops influence different aspects of the race situation. To begin with, they lengthen the lap times of inlap and outlap by $t_{\text{pit,inlap}}$ respectively $t_{\text{pit,outlap}}$. Both parameters depend on the track layout. The inlap is affected by a few seconds because the lap time is taken at the finish line, which is usually located a slightly behind the pit lane entrance. The pitting cars are already slower at this point due to the pit speed limit of 80 km/h. Similar to Bekker [11] and in contrast to Phillips [13], the outlap lap time increase is split into a pit lane drive-through time under speed limit $t_{\text{pitdrive,outlap}}$ and a standstill time $t_{\text{standstill}}$. $t_{\text{standstill}}$ depends on the actions a team takes

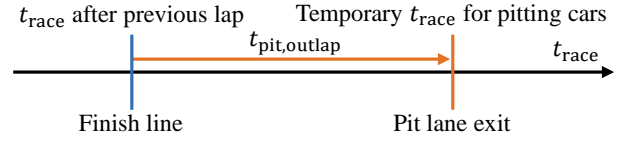


Fig. 5. Pit stop time losses $t_{\text{pit,outlap}}$ are added temporarily to the previous race times t_{race} of pitting cars to obtain comparable race times after pit stops.

during the pit stop. Since refueling was banned in the 2010 season for Formula 1, pit stops mainly involve changing tires or replacing broken spoiler parts. Pit stop penalties t_{penalty} are also important. These usually add five or ten seconds of additional standstill time due to race offenses. As Eq. 9 demonstrates, the simulation provides all the possibilities.

$$t_{\text{pit,outlap}} = t_{\text{pitdrive,outlap}} + t_{\text{standstill}} + t_{\text{penalty}} \quad (9)$$

After having completed the pit stops, the drivers drive back onto the track. The pit lane exit is usually located at the end of the start finish straightaway. This does not match the lap-wise discretization because a normal lap starts and ends at the finish line. Comparable race times are required, however, to be able to consider rank position changes due to the pit stops. As Fig. 5 clarifies, this is solved by adding $t_{\text{pit,outlap}}$ not only to the lap time t_{lap} of the outlap, but temporarily to the race times of the inlap as well. By doing so, the pit cars are virtually released on the finish line. These temporary race times are then used to reorder the rank positions after the pit stops at the beginning of a lap. For the next section, we need to keep in mind that overtaking of drivers in the pit lane occurs without any additional time loss.

C. Race simulation with competitor interaction

After having completed the basic race simulation, overtaking maneuvers are now added:

$$t_{\text{race,currentlap}} = \sum_{\text{lap}=1}^{\text{current lap}} t_{\text{lap}}(\text{lap}) + t_{\text{overtaking}}(\text{lap}) \quad (10)$$

Modeling overtaking maneuvers and their time loss $t_{\text{overtaking}}$: If the race times of two or more competitors are close together after a lap, it must be checked whether overtaking occurs accordingly Fig. 6. Again, the basic approach is similar to Phillips [13], but has been extended by various aspects. Bekker's implementation [11] differs somewhat more due to their discretization of the racetrack into sectors.

At first, the algorithm adds the calculated lap times for the current lap (without including overtaking) to the latest race times to get the current race times. Afterward, it checks whether the DRS is applicable for any driver. As mentioned, it reduces the drag on some straightaways and therefore the lap time. In Formula 1, drivers may use it when the first two laps are completed and the gap to the driver immediately in front is less than one second, i.e. $t_{\text{DRSwindow}} = 1 \text{ s}$. In that case, the lap time and race time of the follower are reduced

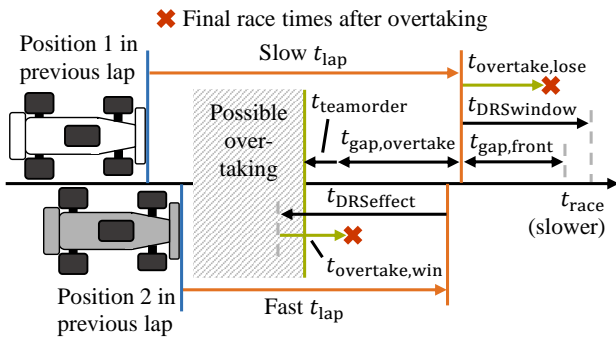


Fig. 6. The overtaking model in detail. The driver in position 2 in the previous lap is fast enough to overtake in the current lap with the help of DRS. As a result, both receive a time malus $t_{\text{overtake,win/lose}}$.

TABLE I
EXEMPLARY TEAM ORDER MATRIX.

	Leader	Driver 1	Driver 2
Follower			
Driver 1		-	-0.1 s
Driver 2		0.15 s	-

by a track-specific time $t_{\text{DRSeffect}}$. The final race time gap between the drivers is then compared to a minimum time gap $t_{\text{gap,overtake}}$ that usually allows overtaking on the particular track. This value primarily depends on the track layout. It is a measure of how easy it is to overtake on the track. If, for example, the follower's race time is 1.2 s faster than that of the driver immediately in front of him, then he can easily overtake on a track with a minimum time gap of 0.9 s. As team orders are common in various racing series, the tool also considers a team order matrix as shown in Table I. It contains modifiers $t_{\text{teamorder}}$ for the minimum time gap, depending on the driver constellation. If the number one driver of a team drives behind the number two driver of the same team, for example, then he does not need to fulfill the entire minimum time gap to overtake. According to the example, it would be reduced by 0.1 s. Slipstream effects are not considered for overtaking. Depending on the track and the racing series, it can be either an advantage or a disadvantage for the follower because driving in turbulent air is not beneficial to the downforce and can therefore cancel out the effect of reduced drag.

If the calculated race time advantage of the rear driver is large enough, then overtaking occurs. In that case, both competitors receive a time malus $t_{\text{overtake,win}}$ respectively $t_{\text{overtake,lose}}$ on their lap time as Eq. 11 states. This is done because the racing lines of the involved drivers differ from the time optimal racing line when they are in a position fight. A side effect of this relation is multiple overtaking. If three or more cars drive close by, it can happen, that the second and third car overtake the first car at once due to the overtaking time loss of the first overtaking maneuver.

$$t_{\text{overtaking}} = t_{\text{DRSeffect}} + t_{\text{overtake,win/lose}} \quad (11)$$

On the other hand, if the rear driver is not fast enough to

overtake but the calculated race time gap to the car in front is small or negative, then the gap has to be artificially enlarged to a minimum value in the simulation. This is because in reality, the follower cannot drive arbitrarily close to the car in front of him. The minimum time gap $t_{\text{gap,front}}$ must be kept between all the cars after overtaking is completed as well as after pit stops are completed.

D. Required parameters

In Table II, an overview of the parameters for the race simulation is provided. Sample values are given for the 2017 Abu Dhabi Grand Prix. They are adjusted based on the lap times published by the FIA [14] and will be used for the simulation in the results section afterward. Table III shows additionally required simulation parameters for the different drivers and manufacturers. For a better overview, we consider only the six drivers of the three top teams, i.e. Mercedes with Bottas and Hamilton, Ferrari with Räikkönen and Vettel and Red Bull with Ricciardo and Verstappen. Because there is only limited data available, the linear tire degradation model is used. The team order matrices are omitted for this example.

IV. RESULTS

The race simulation is implemented in Python. The calculation time for a race is one to two seconds on a common computer (Intel i7 2.7 GHz, 16GB RAM). In Fig. 7, the simulation result for the 2017 Abu Dhabi Grand Prix with the previously listed parameter settings is shown in comparison to the actual timing data. We prefer to plot the race time gaps by comparing every driver to a virtual driver with a constant lap time. It is chosen so that the race duration of the virtual driver corresponds to the race duration of the leader at the end of the race. This method clearly visualizes where the drivers have gained and lost time against the winners average lap time. Additionally, pit stops appear as a sudden rise, e.g. for Verstappen in lap 14.

The comparison between actual and simulated timing data shows a good match. The curved courses of the gap times indicate the effects of fuel mass loss and tire degradation. Furthermore, one can see that similar lap time increases occur for pit stop inlaps and outlaps in simulation and actuality, e.g. for Bottas in laps 21 and 22. As of lap 25, Hamilton closes in on Bottas after his pit stop, but is obviously not fast enough to overtake. Consequently, a minimum time gap is kept. In the simulation, Ricciardo is able to catch Vettel after his pit stop in lap 22 with the help of DRS. This can be seen from the fact that his time gap decreases a bit instead of increasing due to $t_{\text{overtake,win}}$. For him, the effects of fuel mass loss and tire degradation almost cancel each other out after his pit stop, resulting in an almost constant time gap. Therefore, according to our simulation, Vettel would have caught him again later in the race around lap 40. In reality, Ricciardo retired in lap 20 due to a mechanical failure. Looking at the actual data, one can also see that Hamilton stops his hunt for Bottas in lap 52. Of course, this aspect is not included in the simulation data.

TABLE II

OVERVIEW OF THE REQUIRED SIMULATION PARAMETERS. POSSIBLE SOURCES: FP – FREE PRACTICE, LTS – LAP TIME SIMULATION, PR – PREVIOUS RACES, Q – QUALIFYING, R – RULES.

Parameter	Description	Source	Example
Track			
t_Q	Fastest qualifying lap time	Q, (LTS)	96.23 s
$t_{\text{gap,pace}}$	Delta time between qualifying and race pace	FP	3.67 s
$s_{t_{\text{lap,mass}}}$	Mass sensitivity of lap time	LTS	0.033 s/kg
$t_{\text{pit,inlap}}$	Pit stop time loss (inlap)	FP	2.0 s
$t_{\text{pitdrive,outlap}}$	Pit lane drive-through time (outlap)	FP	18.5 s
$t_{\text{perGridPos}}$	First lap time increase per grid position	PR, (LTS)	1.0 s/pos
t_{firstlap}	First lap time increase due to standstill	PR, (LTS)	2.5 s
$t_{\text{gap,overtake}}$	Time gap required for overtaking	PR	1.0 s
$t_{\text{DRSeffect}}$	Time gain due to DRS effect	PR, (LTS)	-0.8 s
Driver			
t_{driver}	Additional time due to driver abilities	Q	0.1 s
p_{grid}	Starting grid position of driver	Q	1
t_{penalty}	Penalty time in pit stop	R	5.0 s
$t_{\text{teamorder}}$	Team order modification of overtaking gap	-	1.0 s
Tire			
$k_{1,\text{log}}$	Parameter of log. tire degradation model	FP, Q, PR	1.0 s/lap
$k_{2,\text{log}}$	Parameter of log. tire degradation model	FP, Q, PR	1.0
$k_{2,\text{lin}}$	Parameter of linear tire degradation model	FP, Q, PR	0.02 s/lap
k_3	Time offset depending on tire compound	FP, Q, PR	0.3 s
Car			
t_{car}	Additional time due to car abilities	Q, (PR)	0.1 s
$m_{\text{fuel,tot}}$	Total fuel mass at race start	R	100.0 kg
$B_{\text{fuel,perLap}}$	Fuel consumption per lap	LTS, FP, R	1.79 kg/lap
$t_{\text{standstill}}$	Average standstill time in pit stop	PR	3.0 s
Race			
$t_{\text{gap,front}}$	Minimum time gap between two cars	PR	0.95 s
$t_{\text{overtake,win}}$	Overtaking time loss (gaining one position)	PR	0.1 s
$t_{\text{overtake,lose}}$	Overtaking time loss (losing one position)	PR	0.6 s
$t_{\text{DRSwindow}}$	DRS window	R	1.0 s

Table IV allows comparing the actual and simulated race durations and rank positions at the end of the Abu Dhabi Grand Prix. Both aspects show that the simulation results are a good match for the actual race. The slightly higher difference in Hamilton's race time can be explained by his stopped hunt for Bottas. It should be noted, however, that these absolute value results depend primarily on a proper adjustment of the parameters as soon as the most important effects have been modeled. The mean signed deviations of the simulated lap times against the actual lap times range from -1.06 s to 0.73 s with standard deviations ranging

TABLE III

ADDITIONAL SIMULATION PARAMETERS FOR DRIVERS AND CARS. TIRE COMPOUNDS: SUS – SUPERSOFT, US – ULTRASOFT.

Parameter	Ricciardo	Räikkönen	Hamilton
Driver			
p_{grid}	4	5	2
Tire at start	US	US	US
Pit stops: Inlap / Tire	19 / SUS	15 / SUS	24 / SUS
t_{driver}	0.0 s	0.307 s	0.0 s
Tire			
$k_{2,\text{lin,US}}$	0.050 s/lap	0.029 s/lap	0.020 s/lap
$k_{3,\text{US}}$	0.0 s	0.0 s	0.0 s
$k_{2,\text{lin,SUS}}$	0.050 s/lap	0.039 s/lap	0.030 s/lap
$k_{3,\text{SUS}}$	0.556 s	0.290 s	0.900 s
Car			
t_{car}	0.244 s	0.370 s	0.0 s
	Vettel	Verstappen	Bottas
Driver			
p_{grid}	3	6	1
Tire at start	US	US	US
Pit stops: Inlap / Tire	20 / SUS	14 / SUS	21 / SUS
t_{driver}	0.0 s	0.637 s	0.1 s
Tire			
$k_{2,\text{lin,US}}$	0.020 s/lap	0.018 s/lap	0.020 s/lap
$k_{3,\text{US}}$	0.0 s	0.0 s	0.0 s
$k_{2,\text{lin,SUS}}$	0.013 s/lap	0.036 s/lap	0.020 s/lap
$k_{3,\text{SUS}}$	0.868 s	0.127 s	0.600 s
Car			
t_{car}	0.370 s	0.244 s	0.0 s

TABLE IV

COMPARISON OF THE ACTUAL AND SIMULATED RACE DURATIONS AND RANK POSITIONS AT THE END OF THE ABU DHABI GRAND PRIX. IF RICCIARDO IS TAKEN OUT OF THE SCORE DUE TO HIS RETIREMENT, THE RANK POSITIONS MATCH AND ARE THEREFORE ONLY LISTED ONCE.

Driver	Actual dur.	Simulated dur.	Δt_{race}	Position
Ricciardo	retired	5687.008 s	retired	retired (4)
Vettel	5674.498 s	5676.086 s	1.588 s	3
Räikkönen	5700.108 s	5698.952 s	-1.156 s	4 (5)
Verstappen	5701.293 s	5701.962 s	0.669 s	5 (6)
Hamilton	5658.582 s	5654.477 s	-4.105 s	2
Bottas	5652.924 s	5653.077 s	0.153 s	1

between 0.63 s to 1.09 s for the different drivers. The absolute average for all drivers is 0.47 s for the mean deviations and 0.80 s for the standard deviations.

V. DISCUSSION

The example race shows how the different time dependencies fit together and form a complete race simulation. With the developed tool the results, i.e. race durations and rank positions, can be forecasted based on the supplied pit stop information. A race engineer can use it by running several simulations with different race strategies before a race and selecting the one with the best result for his own driver or team. During a race, he can quickly adapt the strategy to unexpected circumstances by defining the current race situation as a starting point and, building on this, testing various action alternatives. Both aspects help to achieve the

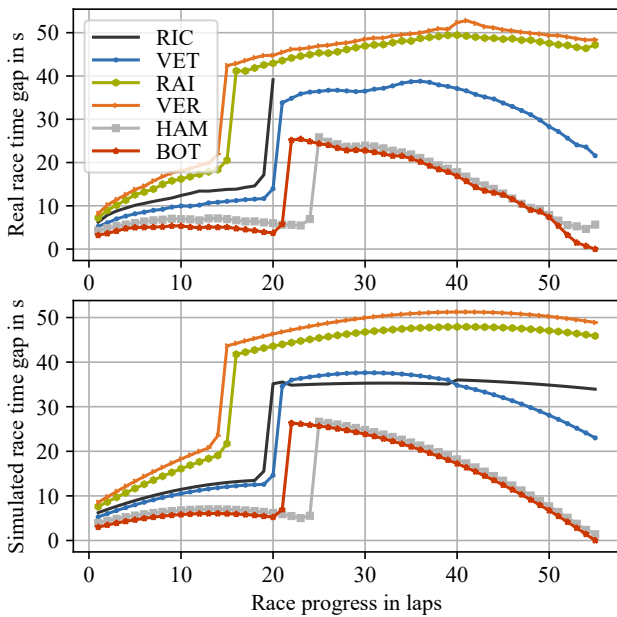


Fig. 7. Actual (upper chart) and simulated (lower chart) race time gaps to a virtual driver with a constant lap time of about 104 s for the 2017 Abu Dhabi Grand Prix. Ricciardo retired in lap 20. Driver abbreviations: RIC – Ricciardo, VET – Vettel, RAI – Räikkönen, VER – Verstappen, HAM – Hamilton, BOT – Bottas.

best possible result. The usage is not limited to racing teams, as the simulation parameters can be determined based on publicly accessible lap time data. The achieved accuracy is sufficient for the application in professional motorsports environments. The points criticized in Bekker’s [11] and Phillips’ [13] models have been improved. One of the most important enhancements in this context is the tire degradation model, which is individually adjusted to the various compounds, drivers and tracks. Another point to mention is the check of the minimum time gaps between all drivers, which is repeated after pit stops have been completed. Further enhancements are the consideration of the overtaking time loss for both drivers involved in an overtaking maneuver and the implementation of a team order matrix.

There are, however, some limitations in the present state. Firstly, energy consumption (fuel consumption in the context of this paper, which refers to Formula 1) is currently implemented with a constant value per lap. Especially for energy limited racing series such as LMP/WEC and Formula E, it would be beneficial to model this in more detail, e.g. by considering overtaking maneuvers. Secondly, driving strategy is not represented adequately because the driver greatly influences lap time, energy consumption and tire degradation. Thirdly, the simulation does not include a safety car, even though it is often decisive for the race. For this reason, the simulation cannot currently be used for all possible race events. Finally, it should be mentioned that lapping cannot be considered due to the lap-wise discretization. This is because laps have already been calculated in which the car to be overtaken is still on the track when looking at the time

axis. However, this does not have much influence because lapping normally takes place without resistance and therefore its influence on the race is negligible.

Future work aims towards the modelling of new aspects, e.g. driving strategy and safety car, as well as detailing existing submodels, e.g. energy consumption and tire degradation. Apart from this, we want to improve the automatic adjustment of parameters based on timing data to relieve the race engineer. We also consider the integration of stochastic elements, as proposed by Bekker [11] and Phillips [13], e.g. for position gains and losses at race start. The final goal is to automatically optimize the race strategy based on the developed race simulation by using the simulation inputs as design variables. The energies available in each lap could also be included as design variables, e.g. to save energy during the race, which can then be used to attack or repel the competitors at the end of the race. The optimization would allow robots to define and adjust their race strategy before and during a race on their own and to react to unforeseen events. Therefore, it could be used for autonomous racing cars in the future, e.g. Roborace [17].

VI. CONCLUSION

In this paper, we presented the methodology of a race simulation that is able to quickly simulate an entire circuit race. It is based on a lap-wise discretization and includes various aspects, such as tire degradation, fuel mass loss, pit stops and overtaking maneuvers. Only publicly accessible lap time data is required as a basis for the simulation parameters. The results shown illustrate the capabilities of the simulation and its suitability to support strategy decisions in motorsports.

ACKNOWLEDGMENTS AND CONTRIBUTIONS

Research was supported by the basic research fund of the Chair of Automotive Technology of the Technical University of Munich. Alexander Heilmeyer leads the research project and contributed to the functional development of the race simulation. Michael Graf contributed to the methodology of the race simulation and performed the data analysis to obtain the exemplary simulation parameters. Markus Lienkamp contributed to the conception of the research project and revised the paper critically for important intellectual content. He gave final approval of the version to be published and agrees to all aspects of the work. As a guarantor, he accepts responsibility for the overall integrity of the paper.

REFERENCES

- [1] Dr. Ing. h.c. F. Porsche AG. (2017). Race strategy: a high-paced game with many unknowns, [Online]. Available: <https://presskit.porsche.de/motorsport/en/mediaguide-2017/topic/in-focus/race-strategy.html> (visited on 04/15/2018).
- [2] Fédération Internationale de l’Automobile. (2018). Regulations, [Online]. Available: <https://www.fia.com/regulations> (visited on 04/15/2018).

- [3] B. Siegler, A. Deakin, and D. Crolla, "Lap Time Simulation: Comparison of Steady State, Quasi-Static and Transient Racing Car Cornering Strategies," *SAE Technical Paper Series*, no. 2000-01-3563, 2000.
- [4] D. Brayshaw and M. Harrison, "A quasi steady state approach to race car lap simulation in order to understand the effects of racing line and centre of gravity location," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 219, no. 6, pp. 725–739, 2005. DOI: 10.1243/095440705X11211.
- [5] I. F. Colunga and A. Bradley, "Modelling of transient cornering and suspension dynamics, and investigation into the control strategies for an ideal driver in a lap time simulator," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 228, no. 10, pp. 1185–1199, 2014. DOI: 10.1177/0954407014525362.
- [6] J. Timings and D. Cole, "Robust lap-time simulation," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 228, no. 10, pp. 1200–1216, 2014. DOI: 10.1177/0954407013516102.
- [7] T. Tulabandhula and C. Rudin, "Tire Changes, Fresh Air, and Yellow Flags: Challenges in Predictive Analytics for Professional Racing," *Big Data*, vol. 2, no. 2, pp. 97–112, 2014. DOI: 10.1089/big.2014.0018.
- [8] T. Hirst, *Wrangling F1 Data with R*, Pre-published manuscript, 2016. [Online]. Available: <https://leanpub.com/wranglingf1datawithr/>.
- [9] McLaren Racing Limited, "Formula One Race Strategy," Royal Academy of Engineering, Tech. Rep. [Online]. Available: <https://www.raeng.org.uk/publications/other/14-car-racing> (visited on 04/15/2018).
- [10] F. Farroni, A. Sakhnevych, and F. Timpone, "Physical modelling of tire wear for the analysis of the influence of thermal and frictional effects on vehicle performance," *Proceedings of the Institution of Mechanical Engineers, Part L: Journal of Materials: Design and Applications*, vol. 231, no. 1-2, pp. 151–161, 2016. DOI: 10.1177/1464420716666107.
- [11] J. Bekker and W. Lotz, "Planning Formula One race strategies using discrete-event simulation," *Journal of the Operational Research Society*, vol. 60, no. 7, pp. 952–961, 2009. DOI: 10.1057/palgrave.jors.2602626.
- [12] F. Bi. (2014). How Formula One Teams Are Using Big Data To Get The Inside Edge, [Online]. Available: <https://www.forbes.com/sites/frankbi/2014/11/13/how-formula-one-teams-are-using-big-data-to-get-the-inside-edge/#5affeeac5588> (visited on 04/15/2018).
- [13] A. Phillips. (2014). Building a Race Simulator, [Online]. Available: <https://flmetrics.wordpress.com/> (visited on 04/15/2018).
- [14] Fédération Internationale de l'Automobile. (2017). Formula 1 Event & Timing Information, [Online]. Available: <https://www.fia.com/events/fia-formula-one-world-championship/season-2017/formula-one> (visited on 04/15/2018).
- [15] Pirelli & C. S.p.A. (2018). Formula 1 tyre range, [Online]. Available: <https://www.pirelli.com/tyres/en-ww/motorsport/homepage-f1> (visited on 04/15/2018).
- [16] S. Mitchell. (2018). Performance deltas revealed for F1 2018 tyre compounds, [Online]. Available: <https://www.motorsport.com/f1/news/pirelli-2018-tyre-compound-deltas-1013128/> (visited on 04/15/2018).
- [17] Roborace Ltd. (2018). Roborace Homepage, [Online]. Available: <https://roborace.com/> (visited on 04/15/2018).

4.4.2 Parameter Determination for the Deterministic Race Simulation

Since only some of the required parameters for the RS can be determined with the LTS, the parameter determination based on timing data is of particular importance for this work, i.e., from an external point of view. Based on preliminary work in semester theses by Bayer [168] and Stahn [169] as well as a master thesis by Müller [170], an automated process is developed and implemented in a master thesis by Faist [171] to convert the timing data of the 121 races in the timing database into parameter sets that can be used in the RS. The main challenge in implementing automated data processing is to reliably detect and remove values from the timing data that are either falsified by the race situation or outliers. An example of falsified timing data is laps that have been affected by FCY phases. Due to the significantly slower lap times during such phases, the inclusion of such laps in the parameter fitting process would falsify most of the parameters to be determined. Outliers, which must also be removed during automated data processing, are values that deviate greatly from the standard behavior and would distort the parameter values if they were considered. For example, a lap should not be considered for determining the tire degradation model parameters if a driver missed the correct braking point and thereby deteriorated his lap time by a second. If a parameter cannot be determined from the timing data or when only too little raw data is available for a robust fitting, a procedure must be available that still enables a reasonable determination, e.g., by using information from other drivers, teams, races, or seasons. Under certain circumstances, it may also be the case that some parameters cannot be determined at all from the timing data. For example, the time loss due to the start from a standstill cannot be determined if a FCY phase was deployed in the first lap.

In the following paragraphs, a summary of the procedures implemented by Faist [171] is presented for the most critical parameters since they have not yet been published. The RS parameter files for all 121 races in the database are available on GitHub [167].

Determination of the Basic Lap Time t_{base}

The basic lap time t_{base} is the sum of two parts [8]: the fastest qualifying lap time t_Q and a race pace time delta $t_{\text{racepace}} \cdot t_{\text{racepace}}$ represents a variety of hard-to-capture differences between qualifying and race configuration. These include, for example, slightly reduced engine power in the race, which is necessary for durability and fuel consumption. The advantage of the hybrid boost is also reduced in the race since, on average, only as much electrical energy can be taken from the storage per lap as is recuperated. Finally, it must be compensated for the fact that t_Q includes the time advantage of using DRS since drivers can use DRS in qualifying without restrictions in all available DRS zones. For t_{base} , however, the DRS time advantage must be removed, since it is taken into account via its own parameter t_{drs} in the RS.

For the determination of t_Q , the fastest lap time of all three qualifying sessions is used instead of using that of the last session, as in some races the conditions deteriorated significantly during qualifying. The determination of t_{racepace} is based on the driver with the fastest race lap. His fastest qualifying lap time is subtracted from the fastest race lap time to obtain the parameter. It is required that the driver has participated in all three qualifying sessions to avoid distortion of the value due to possibly significantly improved conditions during qualifying. If this is not the case, the fastest overall qualifying lap time is subtracted instead. t_{racepace} is usually in the range of two to three seconds. However, it can be significantly negative if the qualifying took place in wet and the race in dry conditions or significantly positive if vice versa [171, p. 76].

Time Losses due to Car and Driver Performance Drawbacks t_{car} and t_{driver}

Both parameters are determined based on qualifying lap times, as it can be assumed that these are largely free of other influences, e.g., tire degradation and fuel mass, and represent the maximum performance of the participants. t_{car} is determined by calculating the difference between the fastest lap time of the team $\min(t_{\text{lap,driver1},i}, t_{\text{lap,driver2},i})$ and the fastest lap time of the corresponding qualifying session $t_{\text{lap,min},i}$ and then using the mean over all j available (i.e., one to three) qualifying sessions as stated by [171, p. 59]

$$t_{\text{car}} = \frac{\sum_{i=1}^j (\min(t_{\text{lap,driver1},i}, t_{\text{lap,driver2},i}) - t_{\text{lap,min},i})}{j}. \quad (4.2)$$

If t_{car} cannot be determined for a race, e.g., because a team did not set a lap time in qualifying, the mean of all available t_{car} values of the team in the corresponding season is taken. t_{driver} is determined similarly. First, the differences between the team's two drivers' lap times and the faster of the two lap times are calculated for each qualifying session. Then, the mean of the j available (i.e., one to three) values is taken to obtain t_{driver} as stated by [171, p. 56]

$$t_{\text{driver}} = \frac{\sum_{i=1}^j (t_{\text{lap,driver},i} - \min(t_{\text{lap,driver},i}, t_{\text{lap,teammate},i}))}{j}. \quad (4.3)$$

Within each team, the values are then shifted such that the faster of the two drivers is assigned $t_{\text{driver}} = 0$ s. His original offset is subtracted from the teammate's t_{driver} value to keep the relative distance the same. If no value can be determined, the mean of all available t_{driver} values of the driver in the corresponding season is used. By averaging the qualifying sessions, robust parameters are obtained since, especially for drivers and cars with similar speeds, chance often determines whether they were faster or slower in the decisive session. Furthermore, the influence of changing environmental conditions during the qualifying is mostly eliminated. [171, p. 56]

Time Loss due to Fuel Mass t_{fuel}

Three parameters are required for the calculation of t_{fuel} [8]: The fuel mass at race start $m_{\text{fuel,tot}}$, the fuel consumption per lap B_{fuel} , and the mass sensitivity of the lap time S_{mass} . $m_{\text{fuel,tot}}$ is set to the maximum fuel mass that may be consumed per race in the corresponding season according to the regulations, as no further data are available. The allowed consumable fuel mass per race was 100 kg from 2014 to 2016 [172, art. 30.5], 105 kg in 2017 and 2018 [173, art. 30.5], and 110 kg in 2019 [6, art. 30.5]. The fuel consumption per lap B_{fuel} is consequently calculated by dividing $m_{\text{fuel,tot}}$ by the planned number of laps of a race since it can be assumed that the teams will make full use of the fuel mass in the car. The mass sensitivity of the lap time S_{mass} is an important parameter because it also influences the fitting of the tire degradation model parameters in the subsequent step. The basic principle for the fitting of this parameter is to compare early and late lap times $t_{\text{lap,early}}$ and $t_{\text{lap,late}}$ of a race that have been driven with the same tire compound at the same tire age. This removes the lap time influence caused by tire degradation. The difference in lap times, together with the difference in mass due to burnt fuel Δm_{fuel} , therefore allows the calculation of a temporary sensitivity for that race, driver, tire compound, and tire age, as stated by

$$S_{\text{mass,tmp}} = \frac{\Delta t_{\text{lap}}}{\Delta m_{\text{fuel}}} = \frac{t_{\text{lap,early}} - t_{\text{lap,late}}}{\Delta l \cdot B_{\text{fuel}}}. \quad (4.4)$$

However, calculating a sensitivity based on only two lap times is highly susceptible to minor unrecognized influences on the underlying lap times. For improved robustness, the chosen implementation only evaluates situations with at least three data points. A linear regression model allows the determination of a temporary sensitivity ($S_{\text{mass,tmp}}$ corresponds to the slope of the model) [171, p. 45f.]. An example of the described procedure is given in Figure 4.2. The data is from the 2015 Hungarian Grand Prix, where Hamilton used three different sets of A3 compound tires. The first set of tires was already run in qualifying so that a tire age of five laps is reached in the third race lap. As can be seen, a linear regression model is fitted for every tire age, yielding four temporary sensitivities $S_{\text{mass,tmp}}$ in this case.

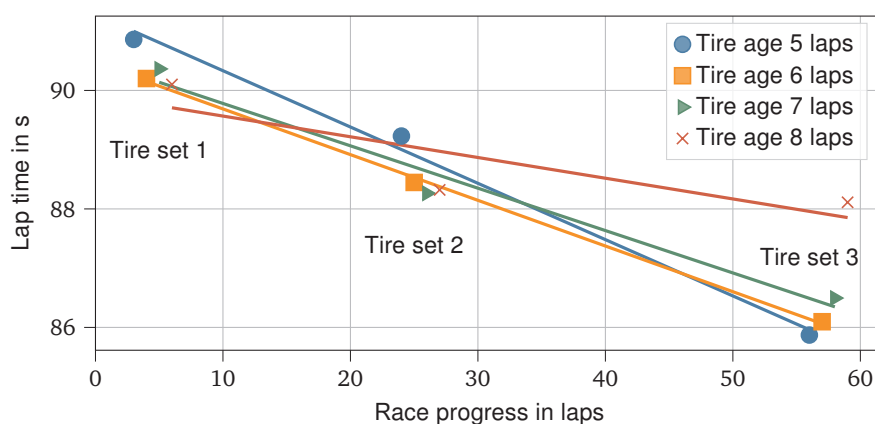


Figure 4.2: Example visualization of the process for determining temporary mass sensitivity values based on some of Hamilton's lap times in the 2015 Hungarian Grand Prix. The three point clusters belong to three different tire sets of the A3 compound. The four lines depict the resulting linear regression models fitted for each tire age, with the slope corresponding to a temporary mass sensitivity value in each case.

When looking at a single driver, three data points would only be available if he drove at least three sets of tires of the same compound in a race, which is rarely the case. Therefore, lap time data with the same tire compound and age of both team drivers are merged before the linear regression models are fitted [171, p. 47]. Since it was found that the mass sensitivity of the lap time is only slightly dependent on the respective team, the available temporary sensitivities of all teams in a race are finally combined into a single value S_{mass} by calculating their median [171, p. 47]. Thereby, even comparatively large deviations of one of the temporary sensitivities (as in the example for a tire age of eight laps) hardly distort the final value of S_{mass} .

The robustness of the determination of S_{mass} is further improved by removing laps affected by FCY phases or pit stops as well as short stints with less than five laps from the raw data. The same applies to laps affected by driver interactions. They are characterized by a time gap to the surrounding drivers of less than 0.8 s. [171, p. 47f.]

Time Loss due to Tire Degradation t_{tire}

The parameterization of the tire degradation models (one per compound) is possible by analyzing the course of the lap times within a stint. At first, falsified raw data are excluded in the same way as for t_{fuel} [171, p. 48]. Afterward, the time loss caused by fuel mass t_{fuel} is removed from the data. Figure 4.3 shows the difference between the original and corrected lap times using Hamilton's first and third stint in Shanghai 2019 as an example. Both stints were driven on the A4 compound.

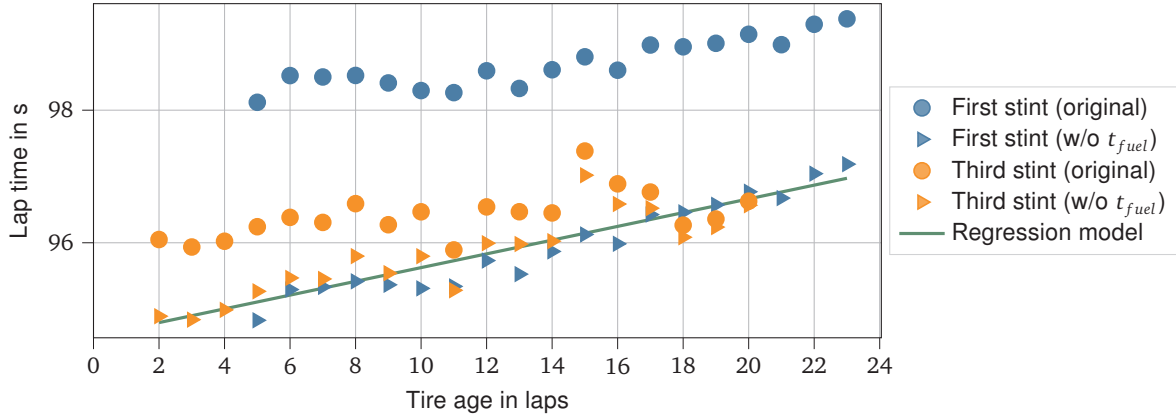


Figure 4.3: Hamilton's lap times of the first and third stint in the 2019 Chinese Grand Prix, both driven on the A4 compound. Besides the original values, the values after removing t_{fuel} are shown. In addition, the fitted linear regression model is plotted.

The comparison of the original lap times of the first and third stints shows an offset of around 2 s. In addition, particularly in the third stint, the lap times at the beginning and end are almost equally fast, despite a tire age difference of nearly 20 laps. Consequently, the raw data are useless for determining tire degradation. After removing t_{fuel} a common trend can be seen, which allows fitting a tire degradation model.

In general, a variety of regression models can be used to represent t_{tire} , e.g., linear, quadratic, or cubic polynomials. An analysis of the different models turns out that the linear regression model is the only one that is robust enough to be fitted automatically [171, p. 88]. Due to the limited number of data points, higher-order models often tend to overfit, leading to large deviations, especially at the beginning and end of a stint. The linear model is defined as [8, p. 2988]

$$t_{tire,lin}(a_{tire}, c_{tire}) = k_0(c_{tire}) + a_{tire} \cdot k_{1,lin}(c_{tire}), \quad (4.5)$$

where a_{tire} is the tire age and c_{tire} is the tire compound. After having fitted the model to the data, the proper value for k_0 can be determined from the intercept after subtracting the base lap time of the race t_{base} . $k_{1,lin}$ corresponds to the slope of the model. The fitted linear regression model for the example is included in Figure 4.3.

For increased robustness, the degradation model parameters are determined team-wise by combining both drivers' available lap time data. If one of the available tire compounds was not driven by a team during a race, but by other teams, the missing parameters are fitted on all available lap time data of that compound in the race, i.e., across teams. This makes it possible to simulate race strategies even with tire compounds that were not driven in reality. [171, p. 49]

4.4.3 Probabilistic Race Simulation

This section summarizes the work carried out on the probabilistic part of the RS that has been published in [12]. The RS is available on GitHub [167].

Summary of the Paper

Real-world races are subject to probabilistic effects. The effects with the strongest impact on the course of the race are failures and accidents, as they often lead to VSC and SC phases. In

addition, there are probabilistic effects with smaller impacts, e.g., lap time variations. The RS presented in Section 4.4.1 lacks these probabilistic aspects so far.

The publication [12] presents an extension of the RS to include probabilistic effects on a race. Based on a literature review, the following effects are modeled:

- Accidents and failures
- VSC and SC phases
- Starting performance of the drivers
- Variability in lap times and pit stop durations

The modeling and parameterization are done based on real-world data from the timing database. Gauss, Beta, and log-logistic distributions are used for the modeling depending on the real distribution of an influence. Wherever possible and useful, the models are parameterized specifically for drivers, teams, and seasons. The modeling of SC in the lap-wise discretized RS is solved by driver-specific *ghost cars*. This approach allows a realistic simulation of the run-up phase and the race's restart and works for lapped drivers. Monte Carlo simulations are used to obtain the resulting rank distribution of a race, which allows an evaluation of different race strategies despite the probabilistic effects. In addition to the expected ranking, the stability of the race strategies against unexpected events can be assessed.

The implementation of probabilistic effects is based on the extension of approaches from the literature, e.g., the determination of accident and failure probabilities, as well as on newly developed approaches, e.g., the modeling of SC phases. The parameterization is carried out on a broad data basis for the first time. Despite the extensions, the computing time per run remains in the range of 100 ms. Thus, for Monte Carlo simulation, 10 000 simulation runs can be completed within 250 s to 300 s using four CPU cores.

Relation to the Research Questions

The publication is related to the first research question. It extends the deterministic part of the RS by probabilistic effects so that the simulation results reflect the possible range of real-world results. This makes it possible to compare different race strategies in preparation for a race and to make informed decisions during a race.

Individual Contribution




Heilmeier developed most of the approaches and extensions presented in the paper. He implemented them, performed the parameterization, and created and analyzed the results. Preliminary work took place in the semester thesis by Rohbogner [174] and the master thesis by Faist [171]. Faist also initiated the idea for modeling the starting performance as presented in the paper.

Imprint of the Paper

The paper was published under an open access Creative Commons CC BY 4.0 license and is available online: <https://www.mdpi.com/2076-3417/10/12/4229>.

Article

Application of Monte Carlo Methods to Consider Probabilistic Effects in a Race Simulation for Circuit Motorsport

Alexander Heilmeier ^{1,*} , Michael Graf ², Johannes Betz ¹  and Markus Lienkamp ¹ 

¹ Institute of Automotive Technology, Technical University of Munich, 85748 Garching, Germany; betz@ftm.mw.tum.de (J.B.); lienkamp@ftm.mw.tum.de (M.L.)

² BMW Motorsport, 80939 Munich, Germany; michael.gm.graf@bmw-motorsport.com

* Correspondence: alexander.heilmeier@tum.de

Received: 11 May 2020; Accepted: 15 June 2020; Published: 19 June 2020



Abstract: Applying an optimal race strategy is a decisive factor in achieving the best possible result in a motorsport race. This mainly implies timing the pit stops perfectly and choosing the optimal tire compounds. Strategy engineers use race simulations to assess the effects of different strategic decisions (e.g., early vs. late pit stop) on the race result before and during a race. However, in reality, races rarely run as planned and are often decided by random events, for example, accidents that cause safety car phases. Besides, the course of a race is affected by many smaller probabilistic influences, for example, variability in the lap times. Consequently, these events and influences should be modeled within the race simulation if real races are to be simulated, and a robust race strategy is to be determined. Therefore, this paper presents how state of the art and new approaches can be combined to modeling the most important probabilistic influences on motorsport races—accidents and failures, full course yellow and safety car phases, the drivers' starting performance, and variability in lap times and pit stop durations. The modeling is done using customized probability distributions as well as a novel “ghost” car approach, which allows the realistic consideration of the effect of safety cars within the race simulation. The interaction of all influences is evaluated based on the Monte Carlo method. The results demonstrate the validity of the models and show how Monte Carlo simulation enables assessing the robustness of race strategies. Knowing the robustness improves the basis for a reasonable determination of race strategies by strategy engineers.

Keywords: racing; simulation; strategy; circuit; motorsport; monte carlo

1. Introduction

Motorsport races are competitions held to determine a ranking among the participants. In circuit races, the result depends not only on the driver and car performance but also on race strategy. Since the participants of such races drive a certain number of laps on a closed circuit, they can drive into their pits at the end of every lap. Pit stops are mostly taken in order to obtain a fresh set of tires, allowing the driver to drive a faster lap time than with an old set. However, since pit stops take some time, one must find an effective compromise between benefit and expense. These aspects are determined by race strategy.

Race simulations are used to simulate and compare the effects of different race strategies. The common approach to building such simulations is to discretize the race lap-wise, as we presented in an earlier paper [1]. Thus, the first simulation step in every lap l is to calculate the expected lap times t_{lap} of the drivers. To obtain them, the simulation adds up a number of different time parts in a lap time model, as shown in Equation (1) [1]. t_{base} is the lap time that the fastest car-driver combination

can theoretically achieve in the race, that is, when the tires are fresh and the car has almost no fuel on board. It therefore takes into account the characteristics of the race track. To this basis are added: t_{tire} for the effect of tire degradation (dependent on tire age a_{tire} and compound c_{tire}), t_{fuel} for the time lost due to the fuel mass that is carried in the car, t_{car} and t_{driver} for the car and driver abilities, t_{grid} for the time that is lost at the race start (dependent on grid position p_g) and $t_{\text{pit,in-lap/out-lap}}$ for the time that is lost in pit stops [1].

$$t_{\text{lap}}(l) = t_{\text{base}} + t_{\text{tire}}(a_{\text{tire}}, c_{\text{tire}}) + t_{\text{fuel}}(l) + t_{\text{car}} + t_{\text{driver}} + t_{\text{grid}}(l, p_g) + t_{\text{pit,in-lap/out-lap}}(l) \quad (1)$$

Consecutive lap times of a driver are summed up to obtain his race times t_{race} at the end of every lap as given by [1]

$$t_{\text{race}}(l) = \sum_{i=1}^l t_{\text{lap}}(i). \quad (2)$$

The race times are the central element of the simulation. For example, they can be compared in order to determine whether an overtaking maneuver would have taken place in reality. This will be the case if the calculated race time of the pursuer is sufficiently smaller (i.e., faster) than that of the car in front. Figure 1 visualizes the simulation flow chart. The strategy engineer specifies the participants' strategies and obtains their race durations (and therefore also their final rank positions).

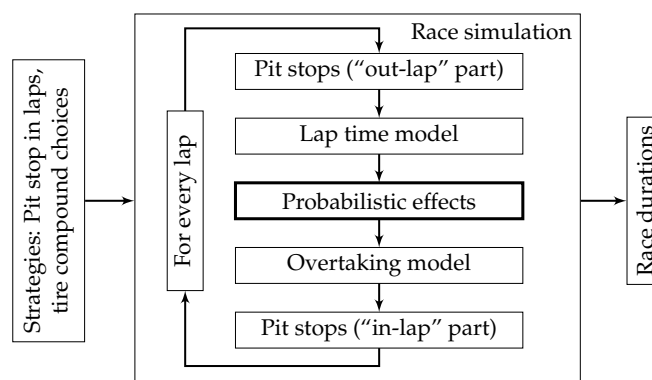


Figure 1. Flow chart of a lap-wise discretized race simulation (based on the flow chart presented in [1]). The block “probabilistic effects” contains most of the extensions proposed in this paper.

The lap-wise discretization provides a good compromise between computational effort, parameterization effort, and accurate results. Although it causes some modeling difficulties, as will be discussed later, it is preferred over other approaches because the fast computing times are essential for the intended application. They enable the strategy engineer to quickly compare the results of many different strategies before and during a race. Moreover, they make the application of the Monte Carlo method, and thus the ideas presented in this paper, possible in the first place. For comparison: A time-discrete simulation or a simulation of the individual driving lines of every car would be orders of magnitude slower and would be hard to parameterize.

In the presented state, however, the simulation misses one crucial aspect: real races are often decided by random events. If, for example, a driver causes an accident, race control usually sends a safety car on the track. Among other things, it significantly reduces the race speed while the crashed car is removed from the track, which leads to increased lap times. Consequently, probabilistic events such as this should be modeled within the race simulation to allow the strategy engineer to determine a stable race strategy. “Stable” in this context means that the strategy should be robust against unforeseen events, that is, it should have a high probability on the targeted rank position.

2. Related Work

The literature and also this paper concentrate on the FIA Formula 1 World Championship because it is the most popular circuit racing series, and accordingly, most data is available. Nevertheless, the presented ideas could be adapted to other racing series as well.

For the determination of the simulation parameters, as well as for the analysis of the probabilistic influences in this paper, a comprehensive database is required. In our case, it is based on the Ergast API [2], which is a web service that hosts Formula 1 timing data. It provides us, for example, with lap times, positions, and pit stop durations. We restructured and extended the data for our purpose, for example, by adding information on accidents and failures, safety car and virtual safety car phases, and the tire compounds used. Furthermore, several cross-checks are carried out during the creation of the database, for example, that the tire compound only changes during a pit stop. Our database covers the Formula 1 hybrid era, that is, the seasons from 2014–2019. It is available under an open-source license on GitHub (<https://github.com/TUMFTM/f1-timing-database>).

Little literature is available that deals with race simulations, and especially the modeling of probabilistic effects in this context. The published works on this topic are based on the Monte Carlo simulation (MCS) concept. MCS “uses random sampling to study properties of systems with components that behave in a random fashion” [3] p. 1. The result of interest for a strategy engineer is the final rank position of his driver. Therefore, MCS is applied by implementing realistic models for the probabilistic effects and then simulating a huge amount of races to determine the estimated distribution of the rank positions. Table 1 gives an overview of the literature coverage on this topic as well as an evaluation of the presented models. The evaluation is based on an assessment of the completeness and accuracy of the models. The models and evaluations are explained in more detail in the next paragraphs.

Table 1. Overview and evaluation of the modeled probabilistic effects in motorsport published in the literature.

Modeled Effect	Bekker et al. [4]	Phillips [5]	Salminen [6]	Sulsters [7]
Starting performance				
Variability of lap time				
Variability of pit stop duration				
Accidents and failures				
Damaged car				
Full course yellow phases				

2.1. Starting Performance

“Starting performance” in this context means the driver’s performance at the start of the race. A good starter will, on average, gain positions during the race start. This can be modeled by sampling the number of lost or gained positions at the race start from an empirical distribution based on historical data [4,7]. The sampled changes are then applied to the drivers’ positions. The problem with this approach is that the positions are changed without changing the respective lap times accordingly, which is not realistic. Furthermore, the treatment of edge cases is unclear. For example, how should it be handled if the driver on the third grid position should win two positions, and the driver on the second grid position should win one position?

Another possibility is used by Phillips [5] and Salminen [6]. They convert the driver-specific average number of positions lost or gained in the first lap $\overline{p}_{\text{change,start}}$ into a positive or negative delta time. The result is used as the mean $\mu_{\text{startperf}}$ of a Gauss distribution, cp. Equation (3). The corresponding standard deviation $\sigma_{\text{startperf}}$ is set 0.25 s [6] or 1 s [5]. In the first lap, the distribution

is sampled to obtain $t_{\text{startperf}}$ for every driver, see Equation (4). It can be considered in t_{grid} , which was introduced in Equation (1).

$$\mu_{\text{startperf}} = \overline{p_{\text{change,start}}} \cdot 0.25 \text{ s} \quad (3)$$

$$t_{\text{startperf}} \sim \mathcal{N}(\mu_{\text{startperf}}, \sigma_{\text{startperf}}^2) \quad (4)$$

Our criticism of this variant is that using the average number of gained and lost positions per driver distorts the probabilities. For example, Lewis Hamilton (world champion in 2018) lost, on average, 0.8 positions in the first laps of the 2018 races, whereas Lance Stroll (ranked 18 of 20 in 2018) gained 1.9 positions [8]. There are several such examples. This is because drivers starting mostly in front positions have only a small potential to improve their position compared to drivers starting at the back of the starting grid. Additionally, the used values for $\sigma_{\text{startperf}}$ are not based on data.

2.2. Variability of Lap Time and Pit Stop Duration

When analyzing real lap times, it can be observed that they are scattered around a mean value, since no driver can perfectly repeat a lap. For the analysis of this effect, other influences on the lap times have to be removed as far as possible, for example, the effects of tire degradation and burned fuel mass. Therefore, quadratic polynomials of the form $t_{\text{lap,poly}}(l) = k_2 l^2 + k_1 l + k_0$ are fitted to the real lap times t_{lap} for every stint. This is visualized in the upper part of Figure 2. Since we only want to include “clean” laps, the first two laps (heavily affected by the start of the race) and all laps that are affected by pit stops or full course yellow (FCY) phases have to be removed from the process. FCY phases are used by race control to reduce speed when there is danger on the race track.

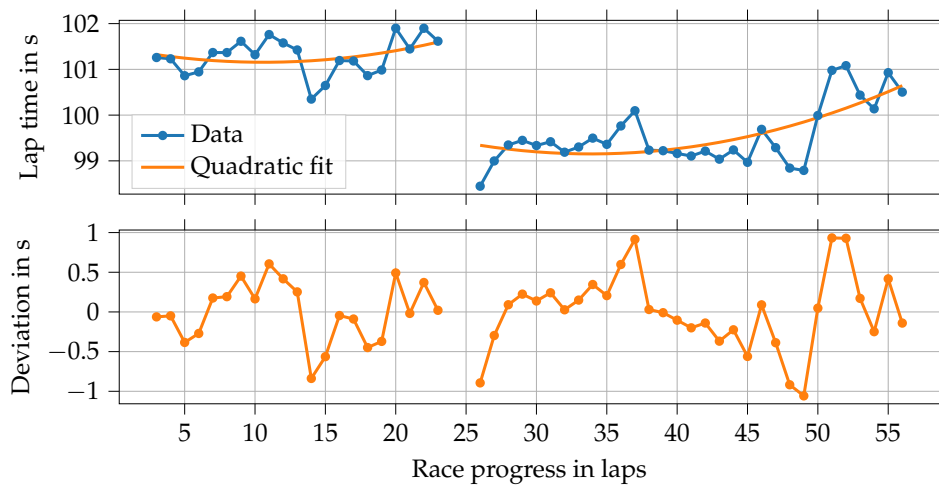


Figure 2. Visualization of the lap time variability of Hamilton in Austin 2019. The deviations are the difference between real lap times and a quadratic polynomial fitted separately for every stint. The first two laps, as well as laps affected by pit stops or full course yellow phases, are not included in this process.

The driver-specific lap time deviations $t_{\text{lap,dev}}$ shown in the lower part of Figure 2 can then be calculated by

$$t_{\text{lap,dev}}(l) = t_{\text{lap}}(l) - t_{\text{lap,poly}}(l). \quad (5)$$

The deviations are approximately normally distributed. Accordingly, Sulsters [7], Phillips [5] and Salminen [6] model the lap time variability by adding a sample $t_{\text{lap,var}}$ from a Gauss distribution

with zero mean and driver-specific standard deviation (cp. Equation (6)) to every lap time t_{lap} in Equation (1).

$$t_{\text{lap,var}} \sim \mathcal{N}(0, \sigma_{\text{lap,var}}^2) \quad (6)$$

As can be seen in Figure 3, pit stop durations vary as well. The plot indicates that the pit stop durations of Mercedes are mostly very close to the minimum duration of the races, whereas Force India's pit stops often take significantly longer.

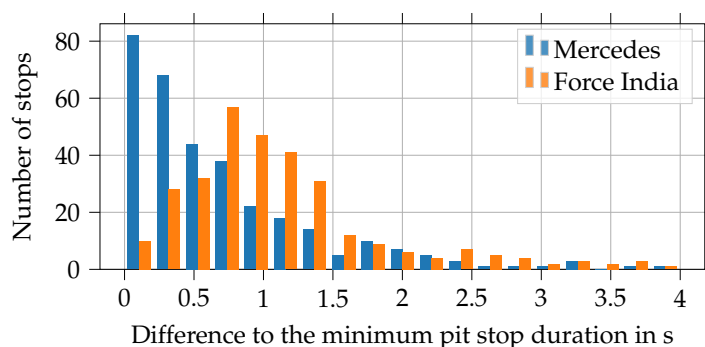


Figure 3. Histogram plot showing the differences between the pit stop durations and the minimum pit stop duration of the races for Mercedes and Force India up to a maximum deviation of 4 s (seasons 2014–2019).

The data demonstrate that a symmetric Gauss distribution would not fit for modeling this variability. Therefore, Phillips [5] uses a log-logistic distribution, also known as Fisk distribution \mathcal{F} , to model this effect, cp. Equation (7). The distribution is fitted individually for every team using the three parameters *shape*, *loc*, and *scale* [9]. A sample from the distribution is then added to $t_{\text{pit,in-lap/out-lap}}$ of Equation (1) when a driver performs a pit stop. Bekker et al. [4] state that they included the effect, but do not describe it any further.

$$t_{\text{pit,var}} \sim \mathcal{F}(\text{shape}, \text{loc}, \text{scale}) \quad (7)$$

2.3. Accidents and Failures

Accidents and (technical) failures both result in the driver being unable to continue the race, known as “did not finish” (DNF) or “retirement”. In literature, no distinction is drawn between the two causes. Bekker et al. [4] use driver-specific probabilities P_{dnf} to determine in every lap if a driver retires. Phillips [5] and Salminen [6] extend the driver-specific probabilities by a lap dependency to consider that there are significantly more retirements during the first lap than in other laps. This results from the small distances between the drivers shortly after the start of the race.

All the probabilities P_{dnf} are based on the fraction of real DNFs of a driver. Consequently, they would be derived to an unrealistic zero if a driver did not have a DNF within the scope of the database. Table 2 demonstrates that this is the case with Lewis Hamilton in the 2019 season, for example. Sulsters [7], therefore, uses Bayesian inference to transfer the knowledge from all available DNFs in the database to the particular driver to determine if he retires within a race. As a result, even drivers without a DNF get a (low) probability for retirement. The approach seems promising. However, Sulsters [7] also does not differentiate between accidents and failures, although both causes affect the races differently, as will be shown later.

Table 2. Number of accidents and (technical) failures for the drivers of Mercedes and Ferrari in the 2019 season (data from [10]).

Driver	Team	Accidents	Failures
Lewis Hamilton	Mercedes	0	0
Valtteri Bottas	Mercedes	1	1
Sebastian Vettel	Ferrari	1	2
Charles Leclerc	Ferrari	2	1

2.4. Damaged Car

In contrast to cars that retire due to an accident or failure, a damaged car can continue the race. Salminen [6] is the only author to modeling this case. He introduces a 1% damage chance per overtaking maneuver for each of the involved cars, and another 1% that both are damaged. A uniform distribution models the effect of damage according to Equation (8). Besides, damaged cars perform an immediate pit stop at the end of the lap.

$$t_{\text{damage}} \sim \mathcal{U}(2\text{ s}, 60\text{ s}) \quad (8)$$

The problem with modeling damaged cars is that a lot of detailed data is required to consider them correctly, especially the relationship between damage, accidents, and FCY phases. To our knowledge, this data is not available to the public. Since Salminen [6] does not describe where the values come from, the modeling of the effect seems unclear.

2.5. Full Course Yellow Phases

A regular yellow flag indicates minor danger in a sector of the race track, for example, a slow car. The effect on the race is negligible. Full course yellow phases indicate a significant danger and therefore have a bigger impact. They limit the speed of the drivers, which increases the lap times. In Formula 1, FCY phases are differentiated into virtual safety car (VSC) and safety car (SC) phases. The race control decides, depending on the danger for the race participants and marshals, which variant is deployed. A VSC prescribes a minimum lap time $t_{\text{lap,vsc}}$ for every driver. Figure 4 indicates, that $t_{\text{lap,vsc}}$ is about 140% of the fastest unaffected lap time of a race $t_{\text{lap,min}}$. Since every driver has to keep to it immediately, the time intervals between the drivers remain more or less constant. During an SC phase, drivers must also reduce their speed, similar to a VSC phase. Additionally, a physical car drives out of the pit lane onto the race track in front of the race leader. It drives much slower than the race cars and must not be overtaken. Therefore, it increases the lap times further to $t_{\text{lap,sc}} \approx 1.6 \cdot t_{\text{lap,min}}$ as soon as the drivers reach it on the track, cp. Figure 4. As a result, the gaps between the drivers vanish. A crucial element for race strategy is that the relative time loss for driving through the pit lane is significantly reduced during VSC and SC phases since it depends on how long it takes to pass the pit lane on the race track in comparison to driving through it. Under SC or VSC conditions, the cars drive slower on the track while driving through the pit lane is always speed limited and, therefore, not affected.

It can be seen in Figure 4 that the lap times vary considerably during SC and VSC phases. Therefore, when calculating the average lap time increases mentioned in the previous paragraph, we apply the following criteria in order to consider only representative lap times. For SC phases, we use those laps in which the SC has already been on track for more than 1.5 laps before the lap and which are not the last lap of the phase. For VSC phases, the lap must be fully covered by the phase. Generally, only the lap times of the drivers on positions 1–3 are used. The values are averaged over all seasons 2014–2019.

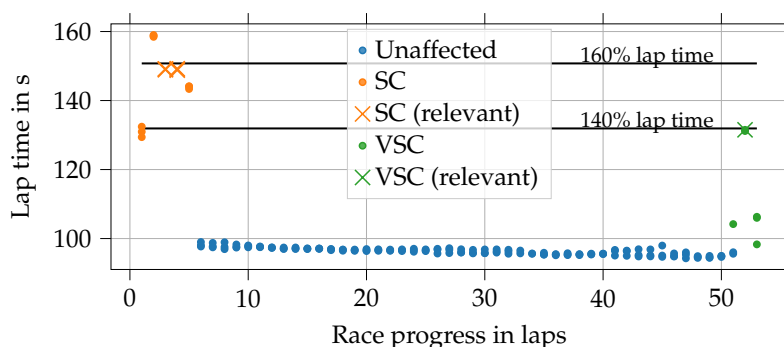


Figure 4. Lap times of the drivers on positions 1–3 in Le Castellet 2018. The first laps are affected by a safety car (SC) phase, the last laps by a virtual safety car (VSC) phase. Crosses mark those lap times that are chosen to calculate the average lap time increase during SC and VSC phases. The horizontal lines indicate 140% and 160% of the fastest lap time of the race $t_{lap,min}$, which is used as a basis.

Sulsters [7] models an SC by increasing the lap time of every driver for five laps by 20%. Phillips [5] simulates SC phases for six laps. He distinguishes between run-up phase (20% lap time increase) and following phase (40% lap time increase). Additionally, he reduces tire wear during an SC phase. Salminen [6] is the only author distinguishing between VSC and SC phases. The VSC is modeled by a lap time increase of 20%. The SC implementation is similar to Phillips [5] except for the reduced tire wear, which is neglected. However, Salminen [6] takes into account the reduced time loss of pit stops during FCY phases $t_{pit,in-lap/out-lap,fcy}$:

$$t_{pit,in-lap/out-lap,fcy} = 0.5 \cdot t_{pit,in-lap/out-lap}. \quad (9)$$

When comparing the implementations in the literature with reality, it becomes clear that several aspects are not modeled accurately. Firstly, FCY phases in the presented approaches always start and end exactly on a completed lap, which is not the case in reality. Secondly, in reality, VSC and SC always appear at the same point in race time t_{race} for every driver, which is different from appearing at certain race progress, especially if some slower drivers are lapped during the race. For example, if an SC is deployed in lap 30, it starts when the race leader is in lap 30. A lapped driver, in contrast, would already be affected in lap 29 in reality. Therefore, it makes more sense to simulate the SC start at a specified race time, for example, $t_{race} = 3000$ s, instead of a specified race progress as in literature. Thus, it affects the race of every driver as it would in reality. Thirdly, according to our assessment of the data, $t_{lap,vsc}$ and $t_{lap,sc}$ are much bigger than what is assumed in literature. Fourthly, the time loss of pit stops under FCY conditions $t_{pit,in-lap/out-lap,fcy}$ is highly dependent on the race track layout and cannot simply be halved. Since this is an important element of race strategy, it should be parameterized more accurately.

3. Methodology

With the lap-wise discretized race simulation (described in Section 1) as a basis, there are few alternatives to MCS for evaluating the effects of probabilistic influences. One option would be to use what-if scenarios, for example, “What happens if the SC gets deployed on lap 30?”. However, this can only be dealt with if we refrain from considering combinations of many probabilistic influences due to the rapidly increasing complexity. Another idea would be to discretize the possible range of the random variables and then simulate races for all combinations (full factorial design). In contrast to MCS, this approach would lead to better sampling of the parameter space in low-probability regions. MCS, on the other hand, provides more meaningful results because it utilizes probability distributions that represent real behavior. Besides, a full factorial design suffers from the curse of dimensionality, which quickly increases computation time. As in literature, MCS is therefore preferred. MCS requires

that the generated random numbers are independent and identically distributed [3] p. 4. Provided that the computer's random number generator (RNG) fulfills this requirement, it also holds for most of the commonly used random distributions, since they are sampled based on the RNG. This also applies to the distributions used in this paper: Gauss distribution [11], Beta distribution [12] and log-logistic (Fisk) distribution [12].

The following sections describe how we have modeled the influences presented in the previous chapter in order to overcome the mentioned limitations. Damaged cars are not considered as we do not have the necessary data available. Besides, it is a rare case that a car that has been involved in an accident is only damaged so slightly that it can continue the race.

3.1. Modeling of Starting Performance

To be able to distinguish between good and bad starters, we need a reference, that is, an average starter. Therefore, we measured the times between race start and crossing the start line (in front of the starting grid) t_s as a function of starting grid position p_g for the 2019 races. This was done using videos from the cockpit perspective, which are available on F1 TV [13]. As Figure 5 reveals, a square root function is a good approximation of the average starter.

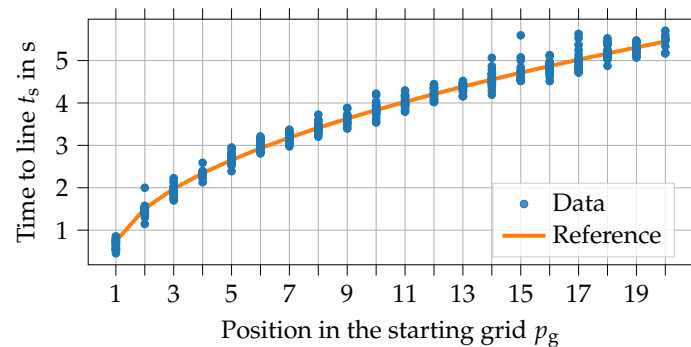


Figure 5. Measured data and reference curve (square root function) for the time between race start and crossing the start line t_s as a function of starting grid position p_g in the 2019 season.

The square root function is physically rational if we assume a constant acceleration for the race start phase. This hypothesis can be made because Formula 1 cars are grip limited and not power limited in the lower speed range. The function is established as follows:

$$t_s = \sqrt{\frac{2(p_g - p_s)}{a_{\text{avg}}}} + t_r. \quad (10)$$

The parameters p_s and t_r stand for the (virtual) position of the start line and the reaction time of a human driver. They shift the origin so that a driver who would start directly on the start line would only have to overcome his reaction time. p_s is set 0.8 because the start line is located only slightly in front of the pole position. Therefore, the distance to the pole starter is significantly smaller than the usual distance between two grid positions (which is 8 m). As a consequence, p_s cannot be set 0. For the reaction time t_r we use 0.2 s. The average acceleration during the race start a_{avg} is then determined using a least-square fit. This results in $a_{\text{avg}} = 11.2 \text{ m s}^{-2}$ when evaluating the data of the 2019 season as shown in Figure 5 and using the distance of 8 m between two starting grid positions.

With the parameterized reference curve, as depicted in Figure 5, we can calculate the differences to the measured data points for every driver. These deviations are then used to calculate mean and standard deviation of a driver-specific Gauss distribution, which is used to modeling the starting performance $t_{\text{startperf}}$ as stated by Equation (4). Samples from these distributions are added to the

first lap time in the race simulation. The parameters of the drivers of the 2019 season can be found in Table A1 in Appendix A.

3.2. Modeling of Variability of Lap Time and Pit Stop Duration

The modeling of the variability of lap time and pit stop duration is adapted from literature as presented in Equations (6) and (7). However, the parameterization was carried out on our significantly larger database. The parameters of the drivers and teams of the 2019 season are given in the Tables A1 and A2 in Appendix A.

3.3. Determination of Accident and Failure Probabilities

As mentioned, we want to differentiate between accidents and (technical) failures. Therefore, we assume that an accident depends on the driver, while a failure depends on the car, that is, the team. If a team changed its name from one season to the next, for example, when Sauber became Alfa Romeo Racing, we treat it under its original name to ensure that the failure probabilities are determined correctly. The accident and failure probabilities are determined by applying Bayesian inference, as suggested by Sulsters [7]. For Bayesian inference, a prior distribution and a likelihood function are required. As with Sulsters [7], the Beta distribution is used as prior distribution, and the Bernoulli distribution as likelihood function (the possible race outcomes are: “finished” or “did not finish”). The prior distribution parameters $\hat{\alpha}$ and $\hat{\beta}$ are determined to [7] p. 11

$$\hat{\alpha} = \left(\frac{1 - \hat{\mu}}{\hat{\sigma}^2} - \frac{1}{\hat{\mu}} \right) \hat{\mu}^2 \quad \text{and} \quad (11)$$

$$\hat{\beta} = \hat{\alpha} \left(\frac{1}{\hat{\mu}} - 1 \right). \quad (12)$$

$\hat{\mu}$ and $\hat{\sigma}$ stand for mean and standard deviation of the prior distribution. They are determined using the total accident fraction per driver, and the total failure fraction per team. Hereby, only drivers and teams with at least 30 races in the database are considered. The two prior distributions for accidents and failures then represent our knowledge about the respective probabilities on the entire database.

Thereafter, driver-, team- and season-specific posterior distributions are calculated taking into account the corresponding accident and failure fractions within the particular season. This proceeding combines the overall knowledge with the specific influence factors of driver, team, and season. For the chosen combination of prior distribution and likelihood function, the posterior distributions are also a Beta($\alpha + z$, $\beta + N - z$) distribution [7] p. 12. z is the number of accidents or failures in the respective season, and N stands for the number of attended races in that season. Figure 6 shows the resulting probability density functions of the accident prior distribution and three driver-specific accident posterior distributions.

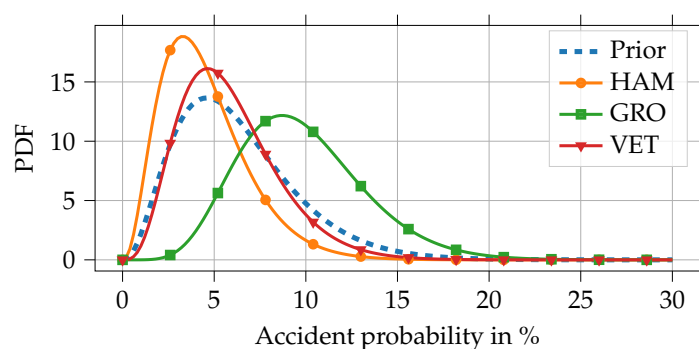


Figure 6. Probability density functions (PDF) of the prior accident probability distribution in comparison to three driver-specific distributions of the 2018 season. Driver abbreviations: HAM–Hamilton, GRO–Grosjean, VET–Vettel.

Finally, the mean values of the posterior distributions are used as accident P_{accident} and failure probabilities P_{failure} for the simulation. The parameters of the drivers and teams of the 2019 season are given in the Tables A1 and A2 in Appendix A.

3.4. Determination of Full Course Yellow Phases in Combination with Accidents and Failures

The determination of FCY phases and retirements must be performed before starting the actual race simulation in order to have the required information available even if backward drivers reach the specified start of a phase in an earlier lap than the race leader. The alternative would be to determine the retirements and their corresponding FCY phases “live” during the simulated race, as used in some of the literature. A small example shows why this does not work correctly with the lap-wise discretization principle. Looking at exemplary race times in Table 3, we find that driver 1 is ahead of driver 2 and driver 3 in laps 20–22 because he reaches the end of each lap earlier (actually driver 3 was even lapped because t_{race} (driver 1, lap 21) $<$ t_{race} (driver 3, lap 20)). Assuming that the simulation would decide in lap 22 that a VSC phase should be activated at $t_{\text{race}} = 2110$ s, we can conclude that it would affect driver 1 shortly after starting into lap 22, while driver 2 and driver 3 would have already been affected in lap 21. Therefore, the problem is that once the simulation decides to activate the VSC phase in lap 22, the previous lap has already been fully simulated due to lap-wise discretization. As a consequence, the SC could not be considered for driver 2 and driver 3 in lap 21 anymore.

Table 3. Exemplary race times t_{race} for three drivers at the end of laps 20–22 to explain a modeling difficulty arising from the lap-wise discretization.

Lap	t_{race} (driver 1)	t_{race} (driver 2)	t_{race} (driver 3)
20	2000 s	2010 s	2102 s
21	2100 s	2120 s	2204 s
22	2200 s	2230 s	2306 s

The solution is to determine all FCY phases and retirements before starting the actual race simulation, as explained in the following. For the definition of FCY phases, a process to fix start race times, durations, and type (VSC or SC) is required. The definition must happen in conjunction with the determination of accidents and failures since they are the causes of FCY phases. For our process, we assume that accidents lead to SC phases. In contrast, if a driver retires due to a failure, he tries to drive to a safe spot. Therefore, we assume that this either causes a VSC phase or no FCY phase at all. We use the following procedure to keep the overall chances of SC, VSC, accidents, and retirements as realistic as possible, although it violates the real cause-effect principle in the case of SC phases:

1. Determine SC phases (quantity, start, duration) and derive accidents
2. Determine failures (quantity, start) and derive VSC phases (duration)
3. Convert race progress to race time

Determine SC phases and derive accidents

The SC phases are fixed at first because they have a significant impact on race strategy, and therefore their probability of occurrence should be no conditional probability. The quantity of SC phases for a race is chosen between zero and three, whereby empirical probabilities $P_{\text{sc,quant}}$ according to the real fractions of the seasons 2014–2019 are used for each of the options, see Figure 7. The exact values are given in Table A3 in Appendix A.

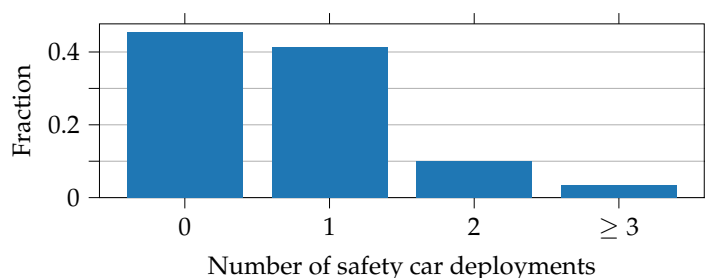


Figure 7. Fraction of races in the seasons 2014–2019 with the specified number of safety car deployments.

Then, the start of every SC phase is defined. Therefore, the race is divided into six groupings (first lap, $\leq 20\%$, $\leq 40\%$, $\leq 60\%$, $\leq 80\%$, $\leq 100\%$) with individual probabilities $P_{sc,start}$. The laps in each group are then assigned the same proportion of the corresponding probability. This classification can be compared with the actual data in Figure 8. The exact values for $P_{sc,start}$ are given in Table A4 in Appendix A. The first lap has to be considered separately since over 36 % of the SC phases start here, which can be explained by the small distances between the drivers shortly after the start that cause a high probability of accidents.

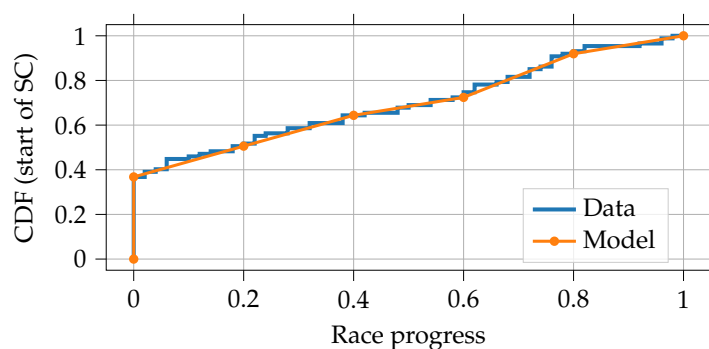


Figure 8. Real and modeled cumulative distribution functions (CDF) of the start of the safety car phases in the seasons 2014–2019.

The duration of an SC phase is chosen to be between two and eight laps with empirical probabilities $P_{sc,duration}$ derived from data of the seasons 2014–2019. The exact values are given in Table 4. The start of an SC phase is further modified by a uniform distribution $\mathcal{U}(0, 1)$ to include the fact that it does not start precisely at the point laps are completed.

Table 4. Probabilities for the duration of a safety car phase (seasons 2014–2019).

Probability	2 Laps	3 Laps	4 Laps	5 Laps	6 Laps	7 Laps	8 Laps
$P_{sc,duration}$	0.182	0.25	0.227	0.193	0.057	0.068	0.023

Before finally saving the created phase, it is assured that it does not overlap an already existing phase. This would be the case if

$$r_{fcy,s,n} \leq r_{fcy,e,e} + r_{fcy,d} \quad \text{and} \quad (13)$$

$$r_{fcy,s,e} - r_{fcy,d} \leq r_{fcy,e,n} \quad (14)$$

both hold. $r_{fcy,s/e,n}$ is the race progress at the start and end of the new and $r_{fcy,s/e,e}$ the race progress at the start and end of the existing FCY phase currently in comparison. $r_{fcy,d}$ is a minimum distance which should be kept between two phases.

As mentioned before, we assume that every SC phase is caused by an accident. Therefore, the simulation chooses one driver who retires at the start of every SC phase. The selection happens based on the drivers' accident probabilities P_{accident} that were determined earlier. Selecting only a single driver for an accident is a simplification, since sometimes two or even more drivers are involved in reality. However, our available data is not detailed enough to be able to modeling and parameterize these cases. Furthermore, retired drivers are not crucial for race strategy determination.

Determine failures and derive VSC phases

Thereafter, the simulation determines, for those drivers not involved in an accident, whether they suffer a failure. The team-specific failure probability P_{failure} determined earlier is used in this respect. Subsequently, the simulation checks for every failure appearing if it causes a VSC phase using the conditional probability $P(\text{vsc}|\text{failure})$. Assuming that every VSC is caused by a failure, it can be calculated using the number of VSC phases n_{vsc} and the number of failures n_{failures} (2015–2019, as the VSC was introduced in 2015) in the database:

$$P(\text{vsc}|\text{failure}) = \frac{n_{\text{vsc}}}{n_{\text{failures}}} = 0.227. \quad (15)$$

This is a simplification because there are some cases where, after an accident, VSC phases were first activated and shortly afterward replaced by an SC phase, for example. However, as before, the available data is not detailed enough to analyze these cases. The start of the failure (and probably of the phase) is sampled from a uniform distribution $\mathcal{U}(0, n_{\text{laps}})$ since no outstanding race section could be identified in the data. n_{laps} stands for the number of laps in the race. The duration of a possible VSC phase is chosen in the range between one and four laps, with empirical probabilities $P_{\text{vsc,duration}}$, and modified by a uniform distribution $\mathcal{U}(0, 1)$ as with the start of SC phases. The exact probabilities for the duration determination are given in Table 5.

Table 5. Probabilities for the duration of a virtual safety car phase (seasons 2015–2019).

Probability	1 Lap	2 Laps	3 Laps	4 Laps
$P_{\text{vsc,duration}}$	0.479	0.396	0.021	0.104

Convert race progress to race time

Due to the lap-based nature of the information in the database, the definition of FCY phases and retirements is also based on laps (i.e., race progress). However, as mentioned in Section 2.5, race times are required instead of race progress so that every driver can be affected at the same point in time. This is achieved by converting the race progress information into race times using a pre-simulation of the actual race with a single driver. It gives a reasonable estimate at which race time a particular stage of progress is reached during the race. Thus, the progress information of the FCY phases can be converted to race times. Deviations between the race times of the pre-simulation and the real race simulation cause no problems, as they change the start and duration of the phases equally for every driver.

3.5. Modeling of Accidents, Failures, and Full Course Yellow Phases

The modeling of accidents and failures is implemented by simply taking the concerned driver out of the race as soon as his race time exceeds the defined time of retirement.

Modeling of the virtual safety car

The VSC is modeled by increasing the lap times of the drivers to $t_{\text{lap,vsc}} = 1.4 \cdot t_{\text{base}}$, cp. Section 2.5. However, since FCY phases can start and end at any point during a lap, we have to calculate the lap

fractions that are driven normally f_n and affected by the VSC phase f_{vsc} to obtain the correct lap time. If, for example, the phase starts within the current lap and ends in a later lap, the resulting lap time t_{lap} can be calculated by

$$f_n = \frac{t_{race,vsc,s} - t_{race,l-1}}{t_{lap,n}}, \quad (16)$$

$$f_{vsc} = 1.0 - f_n, \text{ and} \quad (17)$$

$$t_{lap} = f_n \cdot t_{lap,n} + f_{vsc} \cdot t_{lap,vsc}, \quad (18)$$

where $t_{race,vsc,s}$ is the start race time of the VSC phase, $t_{race,l-1}$ the race time of the driver at the end of the previous lap and $t_{lap,n}$ the unaffected lap time of the driver in the current lap. Similar calculations are performed when the phase ends. Overtaking is forbidden in the simulation if a VSC affects at least 50% of a lap. Due to the limited speed, tire degradation and fuel consumption are reduced to 50% during the phase. This is an estimation since the exact values cannot be derived from the publicly available data. In reality, the saved fuel is, of course, consumed after the phase, for example, by increasing the engine power. In the simulation, the average consumption per lap after an FCY phase is therefore automatically adjusted so that the saved fuel is used up by the race finish.

Modeling of the safety car

For the realistic modeling of SCs, we use driver-individual “safety car ghosts” (SCGs). The concept is illustrated in Figure 9. An SCG can be imagined as a virtual car that is only visible to its corresponding driver and does not affect any other driver. Since it is a safety car, it cannot be overtaken. The driver-individual handling is necessary, since the drivers may be affected by the same SC in different laps due to lap-wise discretization. An SC deployment is modeled in two stages, a run-up stage and a following stage. If a driver reaches the start race time of an SC phase, we assume that his SCG starts driving on the finish line exactly at this time. Equally to the VSC, the lap time of the respective driver is then increased up to $t_{lap,vsc}$ for the remaining part of the lap and following laps to simulate the run-up stage under full course yellow condition. Every driver catches up with his SCG within several laps, since it drives at 160% of the base lap time, cp. Section 2.5. The first SCG lap time is even slower to modeling the real behavior where the SC waits for the leading driver at the pit exit. If a driver’s calculated race time at the end of a lap is below that of his SCG, he would have overtaken it. Thus his lap time is artificially increased to stay behind. Keeping a minimum temporal spacing $t_{gap,sc}$ between the drivers is hereby assured by adding $p \cdot t_{gap,sc}$ to the individual SCG race times, where p stands for the drivers’ rank positions. Tire degradation and fuel consumption are reduced to 25% (again an estimation) while driving behind an SC. The value is smaller than that of the VSC phase because the speed is even lower. The SCGs remain active until the end of the lap in which the SC phase ends, even if the originally determined end time is reached before the end of the lap. This models the fact that the SC can only leave the race track by entering the pit lane at the end of a lap. This proceeding allows a realistic simulation of the re-start of the race with small gaps between the drivers. After each SC phase, the drag reduction system (DRS) is deactivated for two laps, as in reality. The DRS allows drivers to reduce drag resistance on straights when following another driver at a close range. It was introduced in the 2011 season to ease overtaking.

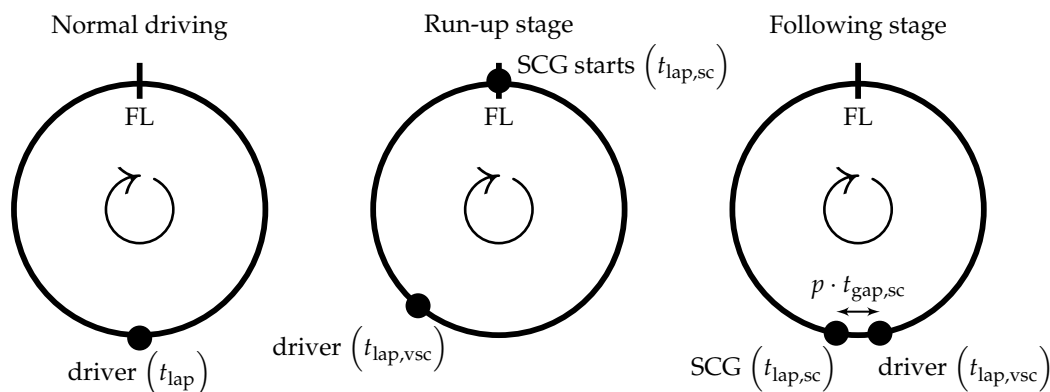


Figure 9. Illustration of the safety car ghost (SCG) concept. For reasons of illustration, the lap-wise discretization is disregarded here. The circle symbolizes a lap on the race track starting and ending at the finish line (FL). During normal driving (left), a race driver drives with his calculated lap time t_{lap} . As soon as his race time t_{race} exceeds the start of a safety car phase, he is slowed down to $t_{lap,vsc}$ and his SCG starts driving on the finish line with a lap time of $t_{lap,sc}$. This is the run-up stage (center). After some time, the driver will catch up with his SCG due to its slower lap time. As he is not allowed to overtake the SCG, he will follow it and keep the minimum temporal spacing $p \cdot t_{gap,sc}$, even though his “free” lap time would still be $t_{lap,vsc}$. This is the following stage (right). At the end of a safety car phase, the SCG always disappears at the finish line.

Adjustment of pit time losses under FCY condition

As mentioned, it is crucial to consider that the relative pit time loss $t_{pit,in-lap/out-lap}$ reduces if a driver drives through the pit lane during an FCY phase. Therefore, smaller pit time losses are added if entering or leaving the pit is fully covered by an FCY phase. During an SC phase, the time losses are often even smaller than during a VSC phase. Since the data are not publicly available, we measured them using videos from the cockpit perspective in 2018 and 2019, which are available on F1 TV [13]. As Table 6 indicates, the differences between normal conditions and FCY conditions vary largely depending on the track layout. Substitute values (for $t_{pit,in-lap/out-lap,vsc}$ and $t_{pit,in-lap/out-lap,sc}$) from similar tracks can be used for race tracks for which no VSC or SC phases have been declared in 2018 and 2019.

Table 6. Time loss when driving through the pit lane under normal conditions, during a virtual safety car (VSC) phase, and during a safety car (SC) phase. The values were measured using videos from the cockpit perspective in the 2018 and 2019 seasons, which are available on F1 TV [13].

Race Track	$t_{pit,in-lap/out-lap}$	$t_{pit,in-lap/out-lap,vsc}$	$t_{pit,in-lap/out-lap,sc}$
Catalunya	19.04 s	10.03 s	7.88 s
Melbourne	17.85 s	12.95 s	12.61 s
Monza	20.60 s	15.42 s	10.16 s
Suzuka	19.48 s	15.05 s	13.60 s

4. Results

The race simulation is implemented in Python. The computation time for a race with 20 participants (as in the 2019 season) is 90 ms to 110 ms on a common computer (Intel i7-6820HQ) including the pre-simulation. For MCS, the races can be simulated independently from each other. Therefore, the calculation time benefits from multiple CPU cores almost linearly. This allows us to perform 10,000 simulation runs in about 250 s to 300 s using all four cores of the CPU.

For reasons of clarity, only the six drivers of the three currently dominating teams (Mercedes, Ferrari, Red Bull) are simulated and shown in this section.

4.1. Effect of Full Course Yellow Phases

Figure 10 shows a comparison between real and simulated race time gaps for laps 29 to 37 of the 2018 Chinese Grand Prix. The gaps $t_{\text{race,gap}}$ are calculated by subtracting the lap-wise race times of a virtual driver from those of the real drivers, cp. Equation (19). The lap time of the virtual driver $t_{\text{lap,virt}}$ is constant and chosen so that his total race duration corresponds to that of the race winner, as given by Equation (20). The plot then clearly visualizes where the drivers gain (negative gradient) and lose (positive gradient) time during the race in comparison to the average lap time. The yellow boxes in the figure mark the laps affected by an SC phase. For this example, the phase was set to fit the real race in the simulation.

$$t_{\text{race,gap}}(l) = t_{\text{race}}(l) - l \cdot t_{\text{lap,virt}} \quad (19)$$

$$t_{\text{lap,virt}} = \frac{t_{\text{race,winner}}}{n_{\text{laps}}} \quad (20)$$

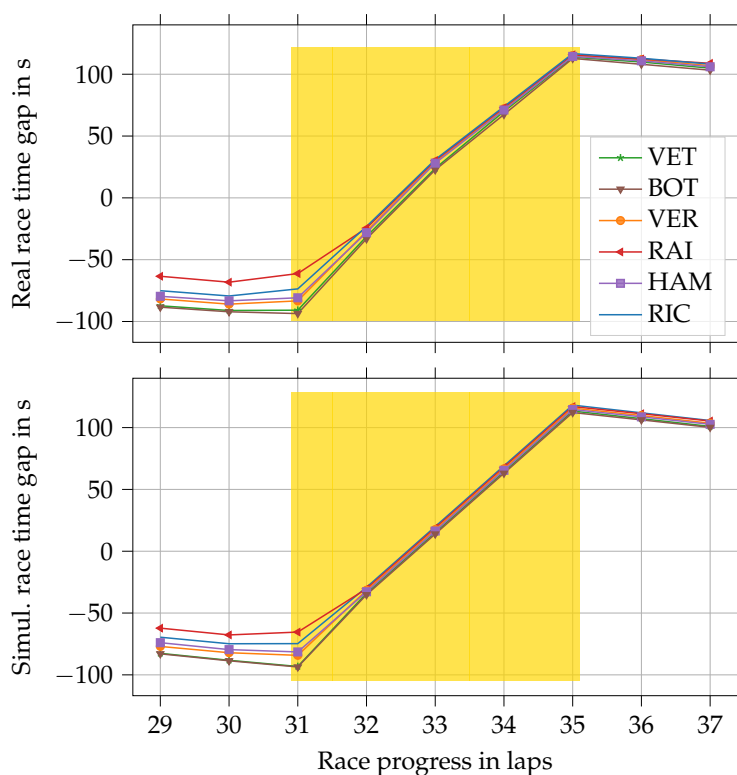


Figure 10. Actual (upper graph) and simulated (lower graph) race time gaps to a virtual driver with a constant lap time in the 2018 Chinese Grand Prix. The yellow boxes mark the laps affected by a safety car phase. Driver abbreviations: VET–Vettel, BOT–Bottas, VER–Verstappen, RAI–Räikkönen, HAM–Hamilton, RIC–Ricciardo.

The figure demonstrates a good correspondence between real and simulated data. As in reality, drivers approach the SC in the simulation and follow it with the corresponding lap time. As can be seen from the small gradient, lap 31 is only slightly affected by the SC in simulation and reality because the SC gets deployed very late in that lap. In this example, all six drivers catch up with the SC within lap 32 because they were not far apart before the SC deployment. Accordingly, they simply follow the SC in laps 32–35. The figure also proves that the re-start of the race happens similarly in simulation and reality in lap 36.

4.2. Analysis of Monte Carlo Simulations

Figure 11 presents an exemplary MCS output for three of the six drivers. The plots show the fraction of races that the drivers have completed on the respective positions. They tell us about the expected race outcome with a given race strategy, which can be used to evaluate different variants. The first row in the figure (w/o MCS) displays the deterministic simulation output, that is, without using MCS. The second and third rows (w/MCS) show the resulting position distributions for two different race strategies for Verstappen when using MCS. In this example, Verstappen's pit stop was postponed from lap 14 (strategy 1) to lap 19 (strategy 2).

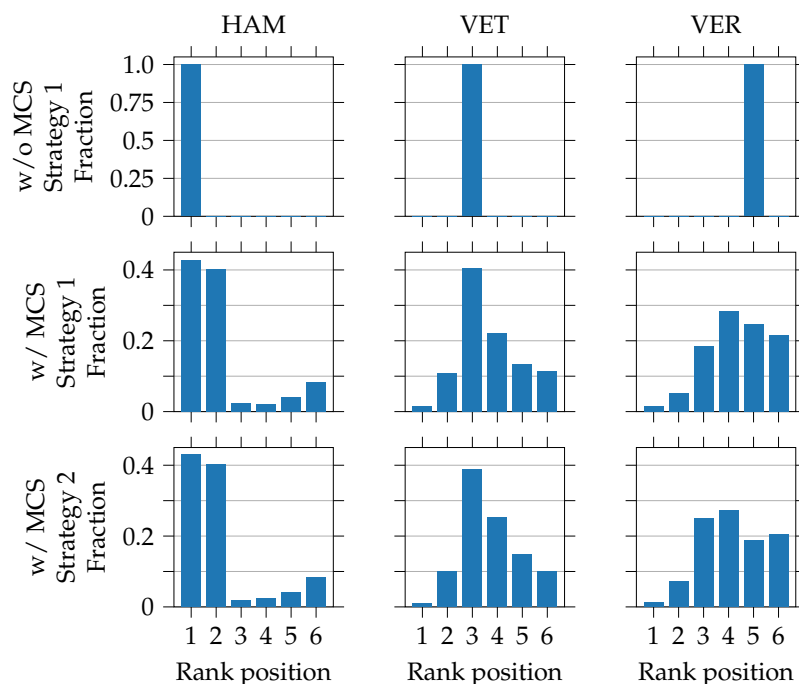


Figure 11. Resulting position distributions for three of the six simulated drivers in an exemplary race without using Monte Carlo simulation (MCS) (w/o MCS) and with using MCS (w/MCS, 10,000 simulation runs). The pit stop of Verstappen was postponed from lap 14 (strategy 1) to lap 19 (strategy 2) in this example. Driver abbreviations: HAM–Hamilton, VET–Vettel, VER–Verstappen.

We want to point out three aspects of the figure. Firstly, the second row provides much more information than the first row because MCS was applied. For Verstappen, for example, it turns out that fifth place in the race is only the second most likely outcome, although the deterministic simulation shows this as a result. For Hamilton, it is almost as likely to finish second as it is to finish first. Secondly, we can observe a shift in the position distribution of Verstappen when switching from strategy 1 to strategy 2. It results in an improvement in third positions. This would again not have been visible in the deterministic simulation. Similar investigations can also be carried out with different tire compounds, for example. Thirdly, the fraction of rank position six is slightly above the previous ones for Hamilton. Retirements due to accidents and failures explain this.

Hence, MCS allows us to determine a basic race strategy before a race that already considers probabilistic influences. At the same time, we gain an idea of the robustness of the strategy against unforeseen events because it is fixed, that is, not adapted to the race situation during the simulation. During a real race, of course, this basic strategy must be adapted to the current situation, for example, SC phases. Without MCS, we would only obtain a single result, which is much less helpful for strategy determination.

It is not only final positions that can be evaluated with the combination of race simulation and MCS. Figure 12 depicts the distribution of race durations after 10,000 simulation runs for this exemplary race. The durations of races without an SC phase differ only slightly. The widely distributed hill between 95 min to 103 min indicates races with a single SC phase. Races with two and three SC phases also have a high spread but appear relatively seldom.

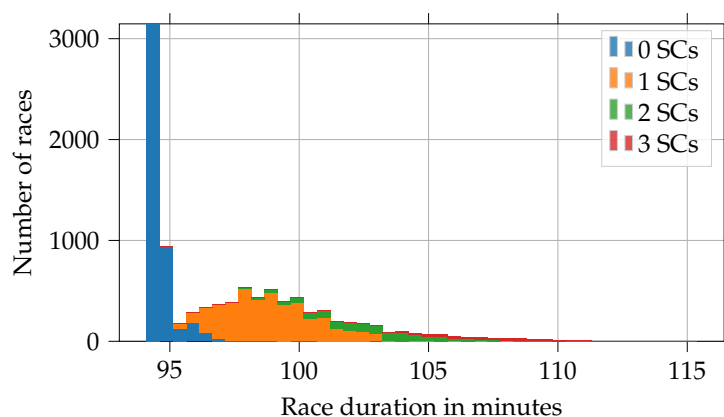


Figure 12. Race durations of the race winner in an exemplary race after 10,000 simulation runs.

However, the fraction of SC phases in the simulation fits well with the actual data. This is outlined in Table 7. The VSC fractions differ slightly due to the conditional probability depending on previous failures, cp. Equation (15).

Table 7. Comparison between actual (seasons 2014–2019 for safety car (SC), 2015–2019 for virtual safety car (VSC)) and simulated (10,000 simulation runs of an exemplary race with 20 drivers) full course yellow phase fractions.

	0 Phases	1 Phase	2 Phases	3 Phases
SC (actual)	0.455	0.413	0.099	0.033
SC (simulated)	0.462	0.416	0.091	0.031
Δ SC	0.007	0.003	−0.008	−0.002
VSC (actual)	0.637	0.294	0.040	0.029
VSC (simulated)	0.618	0.304	0.067	0.011
Δ VSC	−0.019	0.010	0.027	−0.018

But how reliable is the MCS result after, for example, 10,000 simulation runs? According to the law of large numbers, it approaches the expectation ever more closely, the more often the random experiment is repeated. Table 8 shows an evaluation of Hamilton's mean rank position in an exemplary race after 20 batches of simulation runs. It can be seen that the deviation between the batches decreases with a rising number of simulation runs per batch. This behavior is similar for all drivers. In most cases, 10,000 simulation runs offer a good compromise between computation time and certainty. However, it must be emphasized that the result of the race itself depends strongly on a correct parameterization of the race simulation.

Table 8. Mean rank positions and deviations (95 % confidence) for Hamilton in an exemplary race after 20 simulation batches.

Simulation Runs Per Batch	Mean Position	Deviation (95%)
100	2.115	± 0.309
1000	2.126	± 0.082
10,000	2.107	± 0.029
100,000	2.107	± 0.011

5. Discussion

As shown in Section 4.2, the results of the deterministic race simulation do not indicate that they often do not represent the most likely outcome of a race. Additionally, the theoretically fastest strategy for a race often turns out to be fragile when probabilistic effects are considered. Consequently, race simulations should include these effects, which can be evaluated using MCS. Thus, the strategy engineer can benefit from information on the position distribution and robustness of different race strategies against unforeseen events.

The significance of the MCS results depends on the accurate modeling of the probabilistic influences. Therefore, we extended existing ideas (e.g., using Bayesian inference for accidents and failures) and developed new approaches (determination of FCY phases, modeling of safety cars, starting performance) to improve on the points criticized in the literature. Of note is the FCY phase implementation, which affects the drivers equally regardless of their respective race progress. The SCG concept allows the realistic modeling of safety cars despite the lap-wise discretization. The example in the results section outlines that the approach represents reality well. The separate consideration of accidents and failures increases model accuracy and strengthens the cause-effect relationship with the FCY phases. Finally, the presented model for the starting performance does not distort probabilities for drivers starting predominantly at the front or back of the starting grid. The database with the seasons 2014–2019 allows a significantly improved and extended parameterization compared to the literature.

There are, however, some inaccuracies. In reality, at the end of an SC phase, lapped drivers may often catch up one lap to restore the correct ranking of the drivers for the re-start. This behavior cannot be simulated due to lap-wise discretization. For the same cause, an SC in the simulation does not start directly in front of the leader, but at a particular race time. However, this disadvantage was largely eliminated by increasing the first SCG lap time, so that the drivers quickly catch up with their SCG. Another inaccuracy is that we can currently only consider one driver per accident that causes an SC phase. This could be eliminated if more detailed data were available, which would allow us to analyze accidents involving several cars. A thorough analysis of the conditional probability for VSC phases after failures would also be desirable in that case.

The computing time of the race simulation was kept at a similar level despite the extensions. For obtaining the results of Monte Carlo simulations even faster, the introduction of Latin Hypercube sampling could be investigated in the future. Presumably, this reduces the number of simulation runs required to achieve a defined maximal deviation. Regarding our future research, we aim to focus on the optimization of race strategy using the developed race simulation. The challenge is that the basic strategy before a race cannot be determined independently for a single team since every team aims for an optimum. Consequently, the mutual effects must be taken into account. Furthermore, during a race, the basic strategy needs to be quickly adapted to the current race situation, for example, in the case of an SC. This requires a solution that can provide the results very quickly.

6. Summary

In this paper, we presented several new approaches and extensions to modeling important probabilistic effects on a motorsport race within a lap-wise discretized race simulation. This includes driver-specific starting performance, accident and failure probabilities, as well as the determination of full course yellow phases and the modeling of safety cars. The displayed results illustrate the validity of the SC model and show how a strategy engineer can benefit from evaluating probabilistic effects using Monte Carlo simulation when determining race strategies.

The entire Python code of the race simulation is available under an open-source license on GitHub (<https://github.com/TUMFTM/race-simulation>).

Author Contributions: A.H., as the first author, developed most of the presented approaches and extensions, implemented them in the race simulation, and performed large parts of the data analysis to obtain the required parameters. M.G., as the mentor of the research project, critically scrutinized the approaches. M.G., J.B., and M.L. contributed to the conception of the research project and revised the paper critically. Conceptualization,

A.H., M.G., J.B.; data curation, A.H.; formal analysis, A.H.; methodology, A.H.; software, A.H.; validation, A.H.; writing—original draft preparation, A.H.; writing—review and editing, A.H.; visualization, A.H.; supervision, M.G., J.B. and M.L. All authors have read and agreed to the published version of the manuscript.

Funding: Research was supported by the basic research fund of the Institute of Automotive Technology of the Technical University of Munich.

Acknowledgments: We want to acknowledge Marcel Faist, who worked on the automated parameterization of the race simulation during his master’s thesis within the research project. He initiated the idea for the modeling of the starting performance based on a reference curve (the establishment of the square root function in the presented form was carried out by A.H.). Besides, he carried out all measurements using the videos from the cockpit perspective on F1 TV [13].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CDF	Cumulative Distribution Function
DNF	Did Not Finish
DRS	Drag Reduction System
FCY	Full Course Yellow
MCS	Monte Carlo Simulation
PDF	Probability Density Function
RNG	Random Number Generator
SC	Safety Car
SCG	Safety Car Ghost
VSC	Virtual Safety Car

Appendix A. Parameterization of the Probabilistic Effects in the Race Simulation

Table A1. Driver-specific parameterization for accident probability P_{accident} , lap time variability $t_{\text{lap,var}}$ and starting performance $t_{\text{startperf}}$ for all drivers of the 2019 season. P_{accident} is season-specific, the other values are valid for all seasons 2014–2019.

Driver	P_{accident} (2019)	$t_{\text{lap,var}}$ in s ($\mathcal{N}(\mu, \sigma^2)$)	$t_{\text{startperf}}$ in s ($\mathcal{N}(\mu, \sigma^2)$)
Alexander Albon	0.045	$\mathcal{N}(0, (0.628)^2)$	$\mathcal{N}(0.050, (0.168)^2)$
Antonio Giovinazzi	0.058	$\mathcal{N}(0, (0.737)^2)$	$\mathcal{N}(0.006, (0.148)^2)$
Carlos Sainz Jnr	0.045	$\mathcal{N}(0, (0.606)^2)$	$\mathcal{N}(-0.097, (0.126)^2)$
Charles Leclerc	0.058	$\mathcal{N}(0, (0.533)^2)$	$\mathcal{N}(-0.042, (0.125)^2)$
Daniel Ricciardo	0.058	$\mathcal{N}(0, (0.517)^2)$	$\mathcal{N}(-0.047, (0.112)^2)$
Daniil Kvyat	0.058	$\mathcal{N}(0, (0.586)^2)$	$\mathcal{N}(0.027, (0.146)^2)$
George Russell	0.072	$\mathcal{N}(0, (0.759)^2)$	$\mathcal{N}(0.044, (0.174)^2)$
Kevin Magnussen	0.058	$\mathcal{N}(0, (0.608)^2)$	$\mathcal{N}(-0.014, (0.171)^2)$
Kimi Raikkonen	0.058	$\mathcal{N}(0, (0.499)^2)$	$\mathcal{N}(0.088, (0.245)^2)$
Lance Stroll	0.058	$\mathcal{N}(0, (0.642)^2)$	$\mathcal{N}(-0.095, (0.135)^2)$
Lando Norris	0.058	$\mathcal{N}(0, (0.603)^2)$	$\mathcal{N}(0.003, (0.142)^2)$
Lewis Hamilton	0.045	$\mathcal{N}(0, (0.459)^2)$	$\mathcal{N}(-0.052, (0.098)^2)$
Max Verstappen	0.058	$\mathcal{N}(0, (0.473)^2)$	$\mathcal{N}(-0.001, (0.171)^2)$
Nico Hulkenberg	0.058	$\mathcal{N}(0, (0.548)^2)$	$\mathcal{N}(-0.027, (0.115)^2)$
Pierre Gasly	0.045	$\mathcal{N}(0, (0.591)^2)$	$\mathcal{N}(0.044, (0.154)^2)$
Robert Kubica	0.045	$\mathcal{N}(0, (0.841)^2)$	$\mathcal{N}(0.029, (0.128)^2)$
Romain Grosjean	0.072	$\mathcal{N}(0, (0.612)^2)$	$\mathcal{N}(0.102, (0.156)^2)$
Sebastian Vettel	0.045	$\mathcal{N}(0, (0.458)^2)$	$\mathcal{N}(-0.050, (0.115)^2)$
Sergio Perez	0.058	$\mathcal{N}(0, (0.550)^2)$	$\mathcal{N}(-0.009, (0.152)^2)$
Valtteri Bottas	0.058	$\mathcal{N}(0, (0.482)^2)$	$\mathcal{N}(-0.028, (0.135)^2)$

Table A2. Team-specific parameterization for failure probability P_{failure} and pit stop duration variability $t_{\text{pit,var}}$ for all teams of the 2019 season. P_{failure} is season-specific, $t_{\text{pit,var}}$ is valid for all seasons 2014–2019. The parameterization of the pit stop duration variability is based on pit stop durations that are at most 4 s longer than the minimum pit stop duration of a race.

Team	P_{failure} (2019)	$t_{\text{pit,var}}$ in s (\mathcal{F} (shape, loc, scale))
AlfaRomeo	0.056	$\mathcal{F}(5.827, -0.953, 2.327)$
Ferrari	0.086	$\mathcal{F}(2.414, -0.153, 0.737)$
HaasF1Team	0.101	$\mathcal{F}(5.324, -0.667, 1.866)$
McLaren	0.117	$\mathcal{F}(2.639, -0.235, 0.980)$
Mercedes	0.041	$\mathcal{F}(1.563, -0.046, 0.480)$
RacingPoint	0.056	$\mathcal{F}(3.498, -0.249, 1.188)$
RedBull	0.071	$\mathcal{F}(2.045, -0.094, 0.598)$
Renault	0.086	$\mathcal{F}(3.876, -0.419, 1.433)$
ToroRosso	0.071	$\mathcal{F}(4.290, -0.568, 1.606)$
Williams	0.071	$\mathcal{F}(2.562, -0.240, 0.966)$

Table A3. Probabilities for a specified quantity of safety car (SC) deployments in a race (seasons 2014–2019).

Probability	0 SC	1 SC	2 SC	≥ 3 SC
$P_{\text{sc,quant}}$	0.455	0.413	0.099	0.033

Table A4. Probabilities for the start of a safety car phase in one of the six groupings of the race (seasons 2014–2019). The probability of each grouping is divided equally among the laps in it, depending on the total number of laps in the respective race.

Probability	1st Lap	2nd Lap to 20%	20 to 40%	40 to 60%	60 to 80%	80 to 100%
$P_{\text{sc,start}}$	0.364	0.136	0.136	0.08	0.193	0.091

References

- Heilmeier, A.; Graf, M.; Lienkamp, M. A Race Simulation for Strategy Decisions in Circuit Motorsports. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2986–2993.10.1109/ITSC.2018.8570012. [CrossRef]
- Newell, C. Ergast Motor Racing Developer API. Available online: <http://ergast.com/mrd> (accessed on 8 May 2020).
- Lemieux, C. *Monte Carlo and Quasi-Monte Carlo Sampling*; Springer Series in Statistics; Springer: New York, NY, USA, 2009.10.1007/978-0-387-78165-5. [CrossRef]
- Bekker, J.; Lotz, W. Planning Formula One race strategies using discrete-event simulation. *J. Oper. Res. Soc.* **2009**, *60*, 952–961.10.1057/palgrave.jors.2602626. [CrossRef]
- Phillips, A. Building a Race Simulator. Available online: <https://f1metrics.wordpress.com/2014/10/03/building-a-race-simulator> (accessed on 8 May 2020).
- Salminen, T. Race Simulator: Downloadable R Program Code. Available online: <https://f1strategyblog.wordpress.com/2019/05/10/race-simulator-downloadable-r-program-code> (accessed on 8 May 2020).
- Sulsters, C. Simulating Formula One Race Strategies. Master’s Thesis, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands, 2018.
- Collantine Media Ltd. RaceFans Race Data 2018. Available online: <https://www.racefans.net/2018-f1-season/2018-f1-statistics/2018-f1-race-data> (accessed on 8 May 2020).
- The SciPy Community. Documentation on scipy.stats.fisk. Available online: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.fisk.html>(accessed on 29 May 2020).
- Collantine Media Ltd. RaceFans Retirements Data 2019. Available online: <https://www.racefans.net/2019-f1-season/2019-f1-statistics/2019-f1-retirements-penalties> (accessed on 29 May 2020).
- Box, G.E.P.; Muller, M.E. A Note on the Generation of Random Normal Deviates. *Ann. Math. Stat.* **1958**, *29*, 610–611.10.1214/aoms/1177706645. [CrossRef]

12. Leemis, L. Document Page of the Math Department of the College of William & Mary. Available online: <http://www.math.wm.edu/~leemis/chart/UDR/PDFs> (accessed on 8 May 2020).
13. Formula One Digital Media Limited. F1 TV. Available online: <https://f1tv.formula1.com> (accessed on 29 May 2020).



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

4.5 Automation of Race Strategy Decisions

Three different approaches were developed to automate race strategy decisions in the RS:

- Basic Strategy Optimization
- Supervised Learning Virtual Strategy Engineer
- Reinforcement Learning Virtual Strategy Engineer

The first one is based on conventional optimization, while the other two use machine learning methods. They are presented in the following sections.

4.5.1 Basic Strategy Optimization and Supervised Learning Virtual Strategy Engineer

This section summarizes the work carried out on the automation of race strategy decisions that has been published in [13]. The RS including all implemented approaches for the automation of race strategy decisions is available on GitHub [167].

Summary of the Paper

So far, humans have determined a driver's race strategy. Thus, the user of the RS has to determine not only the strategy of his team's driver(s) but also that of all other race participants. For easier use and more realistic results of the simulation as well as a support for own strategy decisions, especially in smaller teams without dedicated strategy engineers, it is therefore aimed to automate race strategy decisions via software.

One part of the publication shows how the race strategy of a driver can be optimized in simulation, assuming some simplifications, mainly a race without opponents. The method is called Basic Strategy Optimization (BSO). However, this approach is insufficient for real races, especially in high-class motorsport. Therefore, the supervised learning Virtual Strategy Engineer (VSE) (hereafter referred to as *supervised VSE*) is presented in addition. It is based on two artificial neural networks to automate race strategy decisions. The first one is invoked once per driver and lap to determine whether the driver should make a pit stop to change his tires. Nine different features are provided to the artificial neural network, based on which the decision is made, e.g., race progress, tire age, rank position, and whether a FCY phase is active. If it decides for a pit stop, the second artificial neural network is invoked to determine the tire compound that should be fitted to the car. It is provided with six different features, e.g., the currently fitted compound and the race track name. As the name suggests, the supervised VSE is trained on real-world data from the timing database. The detailed analysis of the decision behavior in different situations shows that the supervised VSE makes comprehensible decisions. It can therefore be used well in combination with the RS. Finally, the paper gives an outlook on an alternative reinforcement learning VSE (hereafter referred to as *reinforcement VSE*), which is trained directly in the RS using reinforcement learning. This approach is possible due to the previously implemented probabilistic effects.

Artificial neural networks make it possible to learn the complex relations behind race strategy decisions and apply this knowledge when making decisions in new situations. Due to fast inference times, the influence on the computing time of the simulation is small, even when used for many drivers.

Relation to the Research Questions

The publication answers the third research question. It presents three approaches for the automated making of race strategy decisions, one based on solving an optimization problem and two based on machine learning techniques. In combination with the RS, it is thus possible to objectively evaluate and compare the effects of different race strategies.

Individual Contribution

The preliminary work for the supervised VSE was done in a semester thesis by Thomaser [175]. Heilmeier revised and extended the methodology, implemented it into the RS, performed the data analysis, and created the results. After his semester thesis, Thomaser started to work on the reinforcement VSE in his master thesis [176]. He wrote the outlook section on the reinforcement VSE in the paper.

Imprint of the Paper

The paper was published under an open access Creative Commons CC BY 4.0 license and is available online: <https://www.mdpi.com/2076-3417/10/21/7805>.

Article

Virtual Strategy Engineer: Using Artificial Neural Networks for Making Race Strategy Decisions in Circuit Motorsport

Alexander Heilmeier ^{1,*} , André Thomaser ¹, Michael Graf ² and Johannes Betz ¹ 

¹ Institute of Automotive Technology, Technical University of Munich, 85748 Garching, Germany; andre.thomaser@tum.de (A.T.); johannes.betz@tum.de (J.B.)

² BMW Motorsport, 80939 Munich, Germany; michael.gm.graf@bmw-motorsport.com

* Correspondence: alexander.heilmeier@tum.de

Received: 25 September 2020; Accepted: 31 October 2020; Published: 4 November 2020



Abstract: In circuit motorsport, race strategy helps to finish the race in the best possible position by optimally determining the pit stops. Depending on the racing series, pit stops are needed to replace worn-out tires, refuel the car, change drivers, or repair the car. Assuming a race without opponents and considering only tire degradation, the optimal race strategy can be determined by solving a quadratic optimization problem, as shown in the paper. In high-class motorsport, however, this simplified approach is not sufficient. There, comprehensive race simulations are used to evaluate the outcome of different strategic options. The published race simulations require the user to specify the expected strategies of all race participants manually. In such simulations, it is therefore desirable to automate the strategy decisions, for better handling and greater realism. It is against this background that we present a virtual strategy engineer (VSE) based on two artificial neural networks. Since our research is focused on the Formula 1 racing series, the VSE decides whether a driver should make a pit stop and which tire compound to fit. Its training is based on timing data of the six seasons from 2014 to 2019. The results show that the VSE makes reasonable decisions and reacts to the particular race situation. The integration of the VSE into a race simulation is presented, and the effects are analyzed in an example race.

Keywords: race; simulation; strategy; motorsport; machine learning; neural network; decision making

1. Introduction

The goal of every participant in a motorsport race is to finish in the best possible position. An optimum result does not only depend on the speed of the driver and the car, but also requires a lot of other aspects to be worked out. One aspect of circuit motorsport whose importance for a good result is not immediately evident is the pit stop.

Depending on the nature and regulations of the specific racing series, pit stops may be undertaken to replace the tires, refuel the car, change drivers, or repair broken parts. Naturally, while this is going on, the car is stationary, and the driver is losing time compared to the drivers in the race. Furthermore, for safety reasons, driving through the pit lane is restricted to a speed limit and is therefore much slower than simply driving past the pit lane on the race track. This results in a further time loss. However, pit stops also have benefits. With a new set of tires, for example, the driver will be able to achieve significantly faster lap times than with a worn-out set of tires (an effect known as tire degradation). The central aspect of race strategy determination is to balance the cost and benefit of pit stops, such that the total race duration is as short as possible.

If we imagine a race without opponents or probabilistic influences, we can easily calculate the estimated race duration. Figure 1 illustrates how a variation of the pit stop lap and differences in the

choice of tire compounds (either from soft to medium or from soft to hard) influence the race duration of a 1-stop race. In general, a softer compound initially results in faster lap times, but it also degrades faster than a harder compound. Consequently, in this example, if the pit stop took place before lap 13, we would choose to switch to the hard compound.

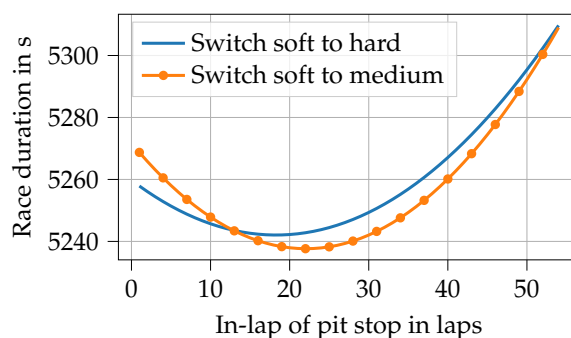


Figure 1. Race durations in an example 1-stop race in relation to the pit stop lap. Two example compound choices are shown.

The simple model of a race without opponents also allows us to explain how full-course yellow (FCY) phases affect strategic decisions. An FCY phase is deployed by race control to reduce the race speed if there is a hazard on the track, e.g., due to an accident. From a strategic perspective, it is essential to know that time lost during a pit stop is significantly less under FCY conditions. This is because it depends on the difference between the time needed to drive from pit entry to pit exit on the track and the time needed to drive through the pit lane. Under FCY conditions, cars drive slower on the track, while going through the pit lane is always speed limited and, therefore, not affected. Figure 2 illustrates how the optimum pit stop lap changes when an FCY phase commences in the middle of lap 14 and lasts until the end of lap 17.

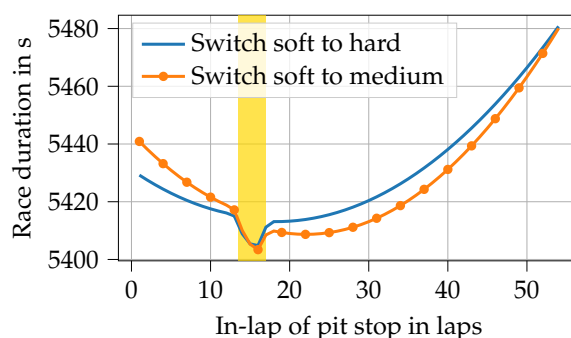


Figure 2. Race durations in an example 1-stop race with a full-course yellow phase lasting from the middle of lap 14 until the end of lap 17 in relation to the pit stop lap. Two example compound choices are shown.

It is evident that the minimum (i.e., shortest) race duration is achieved by entering the pit in lap 16. This is because the pit stop in-lap and out-lap (the finish line is crossed within the pit lane) are both covered by the FCY phase if the driver enters the pit in lap 16. If the pit stop occurred in lap 17, only the in-lap component would benefit from the reduced time loss during an FCY phase. If, however, the pit stop was made earlier than lap 16, the length of the second stint and thus the tire degradation in that stint would be increased, resulting in a longer race duration. However, we should not forget that the simple model does not include other drivers into consideration, which is why no positions are lost during a pit stop. In reality, most drivers enter the pit directly when the FCY phase commences, for this reason.

In contrast to the simplified model, battles for position are an essential aspect of real-world racing. Race strategy offers several tactical opportunities in this context, such as undercut, overcut, and go long [1], of which the undercut is most prominent. It is used if a driver is stuck behind another driver on the track, i.e., if he is fast enough to follow him, but not fast enough to be able to overtake him directly. By performing an earlier pit stop than the driver in front, the pursuer can attempt to take advantage of having a new set of tires. If he gains enough time before his opponent makes his pit stop, he can pass him indirectly while the opponent is still in the pit lane. The greater the advantage of a new set of tires and the less traffic the pursuer faces after his pit stop, the better this maneuver works. Apart from driver interactions, real-world races are also affected by various probabilistic influences, such as lap time variation and FCY phases. Consequently, comprehensive race simulations (RS) are used to evaluate the outcome of different strategic options.

1.1. Functionality of Race Simulations

In [2], we presented a RS that simulates a race taking into account the most important effects on lap times: tire degradation, burned fuel mass, and interactions between the drivers, i.e., overtaking maneuvers. It simulates the race lap-wise, i.e., lap for lap. In each lap l , it calculates the lap time t_{lap} for each driver. This is done by adding several time components, see Equation (1) [2]. The base lap time t_{base} represents the track characteristics. This can be regarded as the minimum lap time that can be achieved by the fastest driver-car combination under optimum conditions. It is increased by t_{tire} to include the effect of tire degradation (depending on the tire age a and compound c), t_{fuel} for the effect of fuel mass aboard the car, and t_{car} and t_{driver} that represent car and driver abilities. At the start of the race, t_{grid} adds a time loss that varies in relation to the grid position p_g . Pit stops also increase the lap time. This is incorporated by $t_{pit,in-lap/out-lap}$. The definition of the pit stops, i.e., in-laps, tire compound choices, and the amount of fuel to be added, currently has to be prescribed by the user as an input to the simulation.

$$t_{lap}(l) = t_{base} + t_{tire}(a, c) + t_{fuel}(l) + t_{car} + t_{driver} + t_{grid}(l, p_g) + t_{pit,in-lap/out-lap}(l) \quad (1)$$

Consecutive lap times can be summed up to obtain the race time $t_{race}(l)$ at the end of a lap as given by

$$t_{race}(l) = \sum_{i=1}^l t_{lap}(i). \quad (2)$$

A comparison of different drivers' race times at the end of a lap allows the algorithm to check whether overtaking maneuvers have occurred, which would result in a modification of the lap times of the drivers concerned. When a race is fully simulated, the RS returns each driver's final race durations as the main output. These inherently include the final positions. In [3], the RS was significantly extended by probabilistic effects, such as lap time variation, variation in pit stop duration, and FCY phases. They are evaluated by performing thousands of simulation runs according to the Monte Carlo principle, resulting in a distribution of the result positions.

There are two applications in which the RS can be used. Before the race, the user is interested in determining a fast basic strategy to start with and assessing several what-if scenarios to prepare him for the race. During the race, the RS can be used to continually re-evaluate the strategy by predicting the race outcomes of different strategic options, starting from the current state.

1.2. Research Goal and Scope

As stated above, the user currently has to insert the pit stop information for all drivers into the RS. This applies to all published race simulations that are known to the authors. This is not only time-consuming, but also requires the user to make a valid estimate of the opponent driver's race strategies. Besides, user-defined strategies are fixed for the simulated race and can therefore not take advantage of tactical opportunities arising from random events during a race. For example,

as described above, it could be beneficial to make a pit stop under FCY conditions. To overcome these limitations, we aim to automate strategy determination by means of software.

Classic optimization methods are hardly applicable in this use case. Firstly, the RS would serve as an objective function for the optimization algorithm. However, the non-linear and discontinuous RS already eliminates many optimization algorithms. Secondly, this variant would again suffer from the problem that the strategy could not be adapted to suit the race situation during the simulated race, as was already the case with the user input. Thirdly, it would not be possible to optimize each driver's strategy simultaneously, from a selfish perspective. However, this is what happens in reality. Fourthly, the computation time would be high, due to the many RS evaluations required to find the optimum. This is all the more so when one considers that each strategy variant would again require many runs due to the probabilistic effects that are evaluated by Monte Carlo simulation.

The use of stochastic models, e.g., a Markov chain, would be another possibility for the automation of strategy decisions. However, the fact that they are based on transition probabilities has several disadvantages compared to other methods. Firstly, even in situations where a pit stop does not seem reasonable based on the available timing data, stochastic models have a chance to decide in favor of a pit stop because there are few such cases in the training data. An example would be if the tires were replaced due to a defect and not due to wear, e.g., after a collision with another car. Other methods learn the underlying mechanisms, which is why rare cases have much less influence on the decision behavior. Secondly, there is no repeatability in the decisions of stochastic models, which makes no sense if we assume that the race situation reasons the decisions. This also makes it difficult to understand and verify the decision behavior of the model. Thirdly, the available real-world training data is very limited. Therefore, the transition probabilities for rare situations cannot be realistically determined given the enormous number of possible race situations.

It is these aspects that gave us the idea of developing a virtual strategy engineer (VSE) based on machine-learning (ML) methods. These allow us to model relationships between inputs (e.g., tire age) and output (e.g., pit stop decision) that are otherwise hard to determine. The idea is to call the VSE of a driver once per lap to take the relevant strategy decisions (whether or not the driver should make a pit stop, which tires are to be fitted, and whether the car is to be refueled) based on the current race situation, tire age, etc. This concept has several more advantages. Firstly, the VSE could be used for any number of drivers while still behaving selfishly for the corresponding driver. Secondly, most ML methods are computationally cheap during inferencing. Thirdly, the VSE could be used not only in simulation, but also to support time-critical strategy decisions in real-world races without a simulation, e.g., when an FCY phase suddenly occurs.

Our simulation environment and the idea for the VSE can be applied to various types of circuit-racing series. For several reasons, however, our research focuses on the FIA Formula 1 World Championship (F1). This is because it provides vast strategic freedom, since the teams can choose from three different dry tire compounds per race and because it is one of the most popular circuit-racing series. Thus, timing data are publicly available that we can use to create a database (which will be presented in more detail later on) for training ML algorithms. This aspect is crucial, since we have no access to internal team data. Consequently, in this paper, we assume that a pit stop is always performed to fit a new set of tires, since there has not been any refueling in F1 since the 2010 season. For the application in other circuit racing series, the race simulation parameters would have to be adjusted for the respective races. Furthermore, the VSE would have to be re-trained on corresponding real-world timing data. For similar regulations, the presented VSE concept should still be usable. For racing series with clearly different regulations, however, the structure of the VSE would, of course, have to be revised. This would be the case, for example, if refueling was allowed during pit stops.

2. Related Work

This section focuses on the topical background. The ML algorithms are interpreted as tools and are therefore not considered any further. The literature reveals that decision making in

sports and sports analytics has become the subject of extensive research during the last two decades. This is due to both the increasing availability of data and the success of ML algorithms that enable meaningful evaluation of data. Possible outcomes of sports analytics include result prediction, performance assessment, talent identification, and strategy evaluation [4]. Consequently, many different stakeholders are interested in such analyses, e.g., bookmakers and betters, fans, commentators, and the teams themselves.

2.1. Analysis Before or After an Event

2.1.1. (American) Football

Predicting the winners and losers of football matches mostly involves the application of ML methods (decision trees, neural networks (NNs), support vector machines) [5–7]. Delen et al. [7] found that classification-based predictions work better than regression-based ones in this context. Leung et al. [8] use a statistical approach rather than ML.

2.1.2. Greyhound Racing

Predicting the results of greyhound races was analyzed in several publications [9–11]. Various ML methods (decision trees, NNs, support vector machines) outperform human experts in several betting formats.

2.1.3. Horse Racing

Harville [12] uses a purely mathematical approach to determine the probabilities of different outcomes of multi-entry competitions, here horse races. In addition, NNs were applied in predicting the results of horse races [13,14].

2.1.4. Soccer

ML methods also dominate when it comes to result prediction (win, lose, draw) in soccer [15–19]. Joseph et al. [15] found that a Bayesian network constructed by a domain expert outperforms models that are constructed based on data analysis. Tax et al. [17] mention that cross-validation is not appropriate for time-ordered data, which is commonplace in sports. Prasetyo et al. [18] combined real-world and video game data to enable improved training of their logistic regression model.

2.1.5. Motorsport

In terms of motorsport, it is mainly the American NASCAR racing series that is investigated in the literature. Graves et al. [20] established a probability model to predict result positions in a NASCAR race. Pfitzner et al. [21] predicted the outcome of NASCAR races using correlation analysis. They found that several variables correlate with result positions, e.g., car speed, qualifying speed, and pole position. Depken et al. [22] found that drivers in NASCAR are more successful in a multi-car team than in a single-car one. Allender [23] states that driver experience and starting position are the most important predictors of NASCAR race results. Stoppels [24] studied predictions of race results in the 2017 F1 season based on data known before the race, e.g., weather and starting grid position. Stergioudis [25] used ML methods to predict result positions in F1.

2.1.6. Various Sports

Result prediction was also investigated for several other sports, generally using NNs or regression analysis. These included swimming [26], basketball [27], hurdle racing [28], javelin throwing [29], and rugby [30].

2.1.7. Broad Result Prediction

In contrast to the previous studies, several publications consider result prediction for more than one sport. McCabe et al. [31] use an NN with input features that capture the quality of sports teams in football, rugby, and soccer. Dubbs [32] uses a regression model to predict the outcomes of baseball, basketball, football, and hockey games. Reviews of available literature and ML techniques for result prediction are available in [4,33]. Haghghat et al. [33] emphasize that there is a lack of publicly available data, making it hard to compare results among different publications.

2.2. In-Event Analysis

2.2.1. Motorsport

The work done by Tulabandhula et al. [34] comes closest to what is pursued in this paper. Using ML methods, they predict the change in position during the next outing (equal to a stint) in NASCAR races on the basis of the number of tires changed in the preceding pit stop. Over 100 possible features were tested, including current position, average position in previous outings, and current tire age. Support vector regression and LASSO performed best, reaching R^2 values of 0.4 to 0.5. It was found that “further reduction in RMSE, increase in R^2 , and increase in sign accuracy may not be possible because of the highly strategic and dynamic nature of racing” ([34], p. 108). The work of Choo [35] is similar to that of Tulabandhula et al. In recent seasons, Amazon [36] applied ML techniques to provide new insights into F1 races for fans. Since they are in a partnership with the F1 organizers, they have access to a vast amount of live data from the cars and the track. Unfortunately, they do not publish how their ML algorithms are set up, trained, and evaluated. Aversa et al. [37] discuss the 2010 F1 season’s final race, in which Ferrari’s decision support system resulted in incorrect conclusions in terms of race strategy. The paper concentrates on the analysis and possible improvements to the decision-making process in the team. Liu et al. [38] combine an NN, which predicts the performance of a Formula E race car, with Monte Carlo tree search to decide on the energy management strategy in Formula E races.

2.2.2. Various Sports

For baseball, Gartheeban et al. use linear regression in [39] and a support vector machine in [40] to decide when a pitcher should be replaced and to determine the next pitch type, respectively. Bailey et al. [41] and Sankaranarayanan et al. [42] apply regression models to predict the match outcome in cricket while the game is in progress. Weber et al. [43] present an approach to model opponent behavior in computer games. They outline that ML algorithms can predict the strategies of human players in a game before they are executed.

2.3. Conclusions

From an engineering perspective, we can conclude that it is mostly two-class (win, lose) and three-class (win, lose, draw) classification problems that are considered in the literature. Many papers emphasize the importance of feature generation, often performed on the basis of domain knowledge. ML algorithms are almost always used, with NNs being increasingly applied in recent years.

From a topical perspective, little literature is available that considers in-event decisions, particularly motorsport. This is surprising, considering the amount of money and data in the sport. The work done by Tulabandhula et al. [34] comes closest to our target. However, it is not applicable in our case, since their algorithm does not predict whether the driver should make a pit stop. The horizon is also limited to a single stint, whereas we have to track the entire race to obtain accurate decisions. Additionally, in NASCAR, only a single tire compound is available. As a consequence, it is necessary to develop a new approach.

3. Methodology

The field of ML covers a vast number of methods. We looked into several classic methods, such as support vector machines and random forests, as well as into NNs. We found that NNs performed much better with our problem than traditional methods, especially in terms of prediction quality, robustness, and comprehensibility. Therefore, we chose to focus on them in this paper. The main emphasis is on NNs trained on real data, a method known as supervised learning. We conclude with a brief outlook towards a reinforcement learning approach based on training in the race simulation.

As indicated in Figure 3, we chose to split the strategy decision in the VSE into two parts: the pit stop decision and choice of tire compound. The idea is that one NN first determines whether the driver should make a pit stop. If it decides to make a pit stop, a second NN determines which tire compound should be fitted. The process implies that the choice of the next compound does not influence the pit stop decision. On the downside, this can lead to a slight decrease in the accuracy of the pit stop prediction. On the other hand, the separation facilitates training, allows us to use NN architectures tailored to the specific application, and shortens inference times (as the tire compound choice NN only needs to be invoked if a pit stop is made), which is indeed why we opted for it. Accordingly, after the description of the database, each NN will be given its own separate treatment in the following.

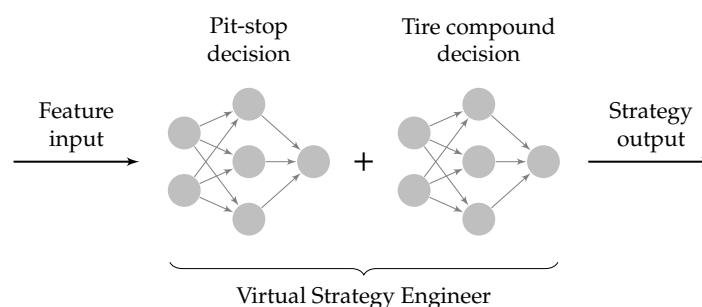


Figure 3. Schematic overview of the virtual strategy engineer (VSE). The neural network for the tire compound decision is called only if the first neural network has decided to make a pit stop.

3.1. Formula 1 Database

The training of NNs requires a huge amount of data. The database we created covers the F1 hybrid era from 2014 to 2019. This period is appropriate as the regulations were relatively stable at that time. The data in the database originate from different sources. The basics, such as lap times, positions, and pit stop information, are obtained from the Ergast API [44]. We then added a large amount of information, e.g., the tire compound used in each lap, and FCY phases (start, end, type). It should be noted that the FCY phases in F1 are divided into two types: virtual safety car (VSC) and safety car (SC) phases. During VSC phases, drivers have to reduce their speed, which increases lap times to about 140% of an unaffected time [3]. In SC phases, a real-world car drives onto the race track, which must not be overtaken. This increases the lap times further to about 160% [3]. Table 1 gives an overview of the entries in the laps table of the database, which contains the most relevant data. The entire database is available under an open-source license on GitHub (<https://github.com/TUMFTM/f1-timing-database>).

The database contains 131,527 laps from 121 races. In 4430 of these laps, the drivers entered the pit. The tires were changed in 4087 of these pit stops, which corresponds to 3.1% of all laps. Accordingly, there is a strong imbalance in the data regarding the pit stop decision. This must be taken into consideration when performing training. Besides this, the large number of changes taking place between seasons makes training difficult. For example, Table 2 shows that new softer tire compounds were introduced in 2016 and 2018 [45], while the 2018 superhard and supersoft compounds were no longer used in 2019 [46]. Additionally, tire dimensions were increased for the 2017 season [47]. Another aspect is that the cars are subject to constant development, so their performance cannot

be compared, not even between the beginning and end of the same season. Similarly, driver skills evolve as driving experience grows. Furthermore, air and asphalt temperatures differ between races in different years. Lastly, seasons are not always held on the same race tracks or with the same number of races per season. For example, the race in Malaysia was discontinued after 2017. This means that when training the NNs on the entire database, perfect results cannot be expected. However, we cannot train them only with driver, team, race, or season data, since the available amount of data will quickly be insufficient.

Table 1. Relevant entries in the laps table of the database. All content refers to the end of the respective lap.

Field	Unit	Description
race_id	–	Unique race ID
lapno	–	Lap number
position	–	Position
driver_id	–	Unique driver ID
laptime	s	Lap time
racetime	s	Race time (cumulated lap times)
gap	s	Distance to race leader
interval	s	Distance to driver in front
compound	–	Current tire compound
tireage	laps	Tire age
pitstopduration	s	Pit stop duration
startlapprog_vsc	–	Lap progress at the beginning of a VSC phase (if VSC starts in current lap)
endlapprog_vsc	–	Lap progress at the end of a VSC phase (if VSC ends in current lap)
age_vsc	laps	Duration of a VSC phase
startlapprog_sc	–	Lap progress at the beginning of an SC phase (if SC starts in current lap)
endlapprog_sc	–	Lap progress at the end of an SC phase (if SC ends in current lap)
age_sc	laps	Duration of an SC phase

Table 2. Overview of available tire compounds in the seasons 2014 to 2019. The column names are inspired by the 2019 season compound names, so as to enable comparison of the compounds over the years. A1 is the hardest compound, and A7 is the softest.

Season	A1	A2	A3	A4	A5	A6	A7
2014	Hard	Medium	Soft	Supersoft	–	–	–
2015	Hard	Medium	Soft	Supersoft	–	–	–
2016	Hard	Medium	Soft	Supersoft	Ultrasoft	–	–
2017	Hard	Medium	Soft	Supersoft	Ultrasoft	–	–
2018	Superhard	Hard	Medium	Soft	Supersoft	Ultrasoft	Hypersoft
2019	–	C1	C2	C3	–	C4	C5

It should also be mentioned that the teams cannot choose freely from the tire compounds available in a particular season. For each race, they are given two (2014 and 2015) or three (from 2016) compounds, which are selected by the tire manufacturer depending on the characteristics of the race track from the available compounds stated in Table 2. Consequently, within a particular race, one focuses on the relative differences between the tire compound options soft and medium (2014 and 2015) or soft, medium, and hard (from 2016 onward), regardless of their absolute hardness. To get a feeling for the different durability of the compounds, Table 3 shows the average normalized tire age at the time of a tire change.

Table 3. Comparison of the average tire age (normalized by race length) when tires are changed, depending on the relative compound and season. The values apply to data filtered according to Section 3.2.2.

Seasons	Average Age Hard	Average Age Medium	Average Age Soft
≤ 2015	–	33.3%	26.8%
≥ 2016	38.8%	34.6%	31.6%
Overall	38.8%	34.1%	29.5%

3.2. Automation of Pit Stop Decisions Using a Neural Network

There are many factors influencing the decision of what is the right time to make a pit stop. We want the NN to learn the underlying relationships between inputs (features) and output (prediction). Thus, we have to provide it with the relevant features.

3.2.1. Feature Selection

The output feature is selected first, since the choice of input features is easier to follow with a known target.

Output Feature

The output feature determines the type of prediction that we are pursuing: classification or regression. In a classification problem, the NN is trained to decide on one of several possible classes. In our case, this would be pit stop or no pit stop, which is a binary output. In a regression problem, the NN tries to predict a quantity, such as race progress remaining until the next pit stop. Since this feature is not available in the database, we added it manually for a comparison of the two prediction types. In our test, we found that classification works much better than regression. The main reason for this is that our manually added output feature is independent of unexpected race events, since we do not know when the teams would have made their pit stop without the event. Consequently, with a regression output, the NN cannot recognize that a change in the example input feature FCY condition is related to a change in the output feature, as outlined in Table 4. We therefore opted for classification.

Table 4. Comparison of example output features in a classification and a regression approach.

Lap	FCY Condition	Output Feature (Classification)	Output Feature (Regression)
10	Not active	No pit stop	0.1
11	Not active	No pit stop	0.05
12	Active	Pit stop	0.0

Input Features

We tested many different input features that were created on the basis of domain knowledge. These were assessed in an iterative process, since the prediction quality depends on both the feature set and the network architecture. Table 5 contains the input feature set that performs best on our data (the metric will be explained later) in combination with the final NN architecture.

Race progress and tire age progress are numerical features, and the rest are categorical features. Since the races are similar in distance but differ in the number of laps (depending on the track length), both progress features are normalized by the total number of laps of a race. Tire age progress is furthermore processed such that it increases 50% more slowly during VSC and 75% more slowly during SC phases, as the tires are significantly less stressed during FCY phases. The position is split into leader and pursuers. It turned out that the leader often decides differently to his pursuers, e.g., when a safety car appears. Relative compound names are preferred over absolute ones (A1, A2, etc.), as they enable better comparison between different seasons and race tracks. Furthermore, the central compounds A2, A3, A4, and A5 are much more strongly represented in the database,

which would cause training issues. The race track category feature was added as an indicator of the absolute level of tire stress on the respective race track. The categorization is given in Table A1 in the Appendix A. It is based on the hardest absolute tire compound available for the respective race track in the 2019 season. Thus, Category 1 contains tracks with high tire stress, whereas the stress is lowest in Category 3. The five possible FCY status values are defined in Table 6. It is important to distinguish between whether an FCY phase starts in the current lap or has already been active for some time. In most cases, drivers enter the pits in the first lap of an FCY phase. In later laps, there is either a high risk that the phase will end while the driver is still in the pit (VSC phase) or that many positions are lost because the drivers have already queued up behind the safety car (SC phase). The number of remaining pit stops contributes significantly to the prediction quality. With this feature, the NN does not tend to predict any further pit stops after the last stop. We trained the NN for races with one, two, and three pit stops. Based on experience, more than three pit stops only occur in wet conditions or due to technical problems (in the seasons under consideration). In a later section, we will explain how this feature can be determined in a race simulation. The tire change of pursuer feature indicates whether the pursuer of a driver changed his tires in the previous lap, which helps to counteract undercut attempts. This is improved if combined with the close ahead feature, which is true if a driver is a maximum 1.5 s ahead of his pursuer.

Table 5. Input feature set for a neural network to take the pit stop decision.

Feature	Type	Value Range	Evaluation Point
Race progress	Numerical	[0.0, 1.0]	End of previous lap
Tire age progress	Numerical	[0.0, 1.0]	End of previous lap
Position	Categorical	{leader, pursuer}	End of previous lap
Relative compound	Categorical	{soft, medium, hard}	Current lap
Race track category	Categorical	{1, 2, 3}	– (constant)
FCY status	Categorical	{0, 1, 2, 3, 4}	Current lap
Remaining pit stops	Categorical	{0, 1, 2, 3}	Current lap
Tire change of pursuer	Categorical	{true, false}	End of previous lap
Close ahead	Categorical	{true, false}	End of lap before previous lap

Table 6. Explanation of the full-course yellow (FCY) status feature.

FCY Status	Description
0	No FCY phase active
1	First lap of a VSC phase active at the end of the lap
2	Further laps of a VSC phase active at the end of the lap
3	First lap of an SC phase active at the end of the lap
4	Further laps of an SC phase active at the end of the lap

The features are evaluated at different points. In reality, the decision to make a pit stop during a lap must be taken before the driver passes the pit lane entry. Accordingly, the decision is mostly based on information available at the end of the previous lap. An exception is the close ahead feature. If the pursuer drives into the pit, he loses some time in the in-lap, i.e., before crossing the finish line in the pit lane. Consequently, he will in most cases fall out of the 1.5 s window. Since we have no better alternative with the available data, the feature is evaluated based on the lap before the previous lap. There are three features whose values are taken from the current lap: relative compound, remaining pit stops, and FCY status. This is done for the former two to obtain the correct information, even if the tire was changed in the previous lap. The evaluation of the FCY status is always based on the latest information, as also happens in reality, where teams react very quickly to VSC and SC phases.

3.2.2. Pre-Processing

Several filters are applied when fetching the data from the database:

- Wet races are removed, since tire change decisions under such conditions are mainly based on driver feedback and track dampness (which is not included in the data available to us). This is also the reason why wet compounds were not included in the input feature set.
- Data relating to drivers making more than three pit stops in a race are removed, as already explained.
- Data relating to drivers making their final pit stop after a race progress of 90% are removed. Such pit stops are either made in the event of a technical problem or due to the new rule in the 2019 season that states that the driver with the fastest lap time is awarded an extra championship point (if he also finishes within the top 10). Since the car is lightest shortly before the end of a race (due to the burned fuel), there is the best chance for a fast lap time. Accordingly, drivers who are so far ahead towards the end of a race that they would not drop a position by making a pit stop can afford to stop and get a new set of tires. However, we did not want the NN to learn this behavior, since it is specific to the current regulations.
- Data relating to drivers with a lap time above 200 s or a pit stop duration above 50 s are removed. Such cases occur in the event of technical problems, damage to the car, or interruption of the race.
- Data relating to drivers with a result position greater than ten are removed. This is done in order to train the NN on successful strategies only. Of course, a finishing position within the top ten is no guarantee of a good strategy. However, as top teams have more resources, we assume that they tend to make better decisions. This does not introduce bias into the training data, since there are no features in the set that would subsequently be underrepresented, as would be the case with team names, for example.

Applying all the filters results in 62,276 entries (i.e., laps) remaining. The earliest race data presented to the NN are from the end of the first lap. This is because some features, such as the ahead value, are not available before the start of the race. Consequently, the first pit stop decision can be taken for lap two. However, as pit stops in the first lap are not generally made on the basis of worn tires but due to technical problems, this does not cause any issues.

Further on in the process, specified test races are separated from the data to exclude them from the training. Then, the pre-processor removes the mean and normalizes the range of numerical features. Categorical features are one-hot encoded. Next, the data are split into subsets for training, validation, and testing (in the evaluation pipeline) or training and validation (in the final training pipeline), as shown in Figure 4. The training section refers to the training of the NN. The validation section is used to recognize overfitting during training, which ends the training process ('early stopping'). The test part is used to evaluate the prediction quality of the trained NN. Final training takes place without a test part to maximize the amount of training data available. When evaluating different NN architectures and feature sets, we apply 10-fold cross-validation to obtain a reliable statement regarding prediction quality. During cross-validation and splitting, subsets are created in such a way that they preserve the percentage of samples for each class ('stratification'), which is essential with imbalanced data.

3.2.3. Metrics

The most common metric used to assess ML algorithms is accuracy. However, it is not appropriate in the case of a binary classification problem with imbalanced data because it places too little focus on the positive class (in our case, pit stop). Precision and recall are better choices. Precision represents the fraction of correct positive predictions. Recall represents the fraction of the positive class that was correctly predicted. For example, if the NN predicts a total of four pit stops, of which three are correct, then the precision is 0.75. If there are ten pit stops in the training data, then the recall is 0.3. Usually, there is a trade-off between precision and recall: an increase in one results in a decrease in the other. Since both measures are of similar relevance to us, we decided to take the F_1 score introduced in

Equation (3) [48]. This is the harmonic mean of precision p and recall r and, therefore, does not reach a high value if one of the two is low.

$$F_1 = 2 \frac{p \cdot r}{p + r} \quad (3)$$

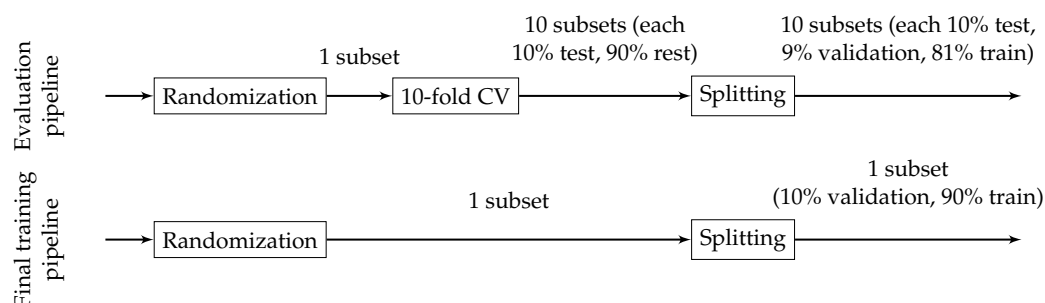


Figure 4. Data splitting pipeline during evaluation (top) and final training (bottom). Cross validation (CV) and splitting is always stratified.

3.3. Automation of Tire Compound Decisions Using a Neural Network

As with the pit stop decision, we need to choose features that allow the NN to learn the relevant relationships for the decision on which tire compound should be selected during a pit stop.

3.3.1. Feature Selection

As before, the output feature is explained first.

Output Feature

The NN shell predicts the relative compound that is chosen during a pit stop, i.e., soft, medium, or hard. Due to the limited amount of training data, we decided to consider the 2014 and 2015 seasons, in which only two compounds were available per race, by using a special input feature, instead of training a separate NN. In contrast to the pit stop decision, this is a multi-class classification problem. Consequently, there are three neurons in the NN's output layer, of which the compound with the highest probability is chosen.

Input Features

The feature set for the tire compound decision is much smaller than for the pit stop decision, as can be seen from Table 7. The features race progress, remaining pit stops, and relative compound are already familiar from the pit stop decision. For the compound decision, the race track feature is preferable to the race track category due to the improved prediction quality. This does not lead to overfitting in this case, as training is still carried out across the different seasons and teams. For better generalizability, however, the race track category could also be used (at the expense of a worse prediction quality). The fulfilled second compound feature indicates whether a driver has already used two different compounds in a race. The regulations in F1 require this. The last feature, number of available compounds, is two for the seasons prior to 2016 or three from 2016 onwards. In the former case, the NN learns to neutralize the hard output neuron very well.

Table 7. Input feature set for a neural network to take the compound decision.

Feature	Type	Value Range	Evaluation Point
Race progress	Numerical	[0.0, 1.0]	End of previous lap
Remaining pit stops	Categorical	{0, 1, 2, 3}	Current lap
Relative compound	Categorical	{soft, medium, hard}	Current lap
Race track	Categorical	{Austin, Baku, ..., Yas Marina}	– (constant)
Fulfilled second compound	Categorical	{true, false}	Current lap
Number of avail. compounds	Categorical	{2, 3}	– (constant)

3.3.2. Pre-Processing

Here, data filtering is less strict than with the pit stop decision. Since the available amount of training data is much smaller, we cannot remove too much of it. Only the 4087 laps with a tire change are relevant. As with the pit stop decision, the entries are filtered for races in dry conditions with one to three pit stops. Furthermore, data relating to a result position above 15 are removed. This results in 2757 entries remaining.

Pre-processing, including splitting the data into subsets, is similar to that of the pit stop decision. The data are, however, not as imbalanced. Figure 5 shows an overweight on the medium compound decisions. This is because the regulations force the top ten drivers to start the race on the tires on which they set the fastest lap time in the second part of the qualifying. In most cases, this means that they have to start on the soft compound, which is why they do not often change back to the soft compound during the race. The imbalance in the data is small enough that no further action is needed.

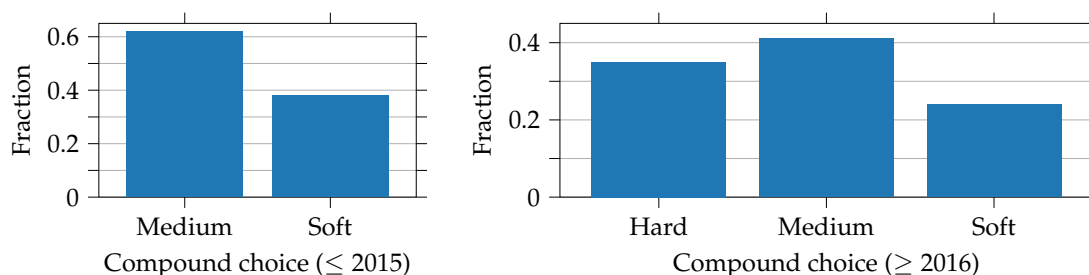


Figure 5. Fractions of compounds chosen during pit stops in the 2014 and 2015 seasons (**left**) and from the 2016 season onwards (**right**). The distributions shown represent the situation after filtering the data according to Section 3.3.2.

3.3.3. Metrics

The goal is to make as many correct compound choice predictions as possible. Accordingly, accuracy is the metric used in evaluating the NN. It represents the fraction of correct predictions.

3.4. Optimization of Basic Race Strategies

We mentioned above that the remaining pit stops feature significantly improves the prediction quality of the VSE, which is why it was included in the input feature sets. Consequently, if the VSE is integrated into the race simulation, the intended number of pit stops for a race must be determined. As mentioned in the introduction, if we consider the interactions between drivers and probabilistic influences, there is no optimum strategy. However, if these effects are omitted and if we assume some simplifications (which will be explained in the following), it can be done using a two-step approach:

1. Determination of all possible tire compound combinations for each reasonable number of pit stops
2. Determination of the optimum stint lengths for each of those combinations

Once the minimum race duration is known for each compound combination, we can compare them and take the fastest.

3.4.1. Determination of all Possible Tire Compound Combinations

As mentioned above, the optimum number of pit stops in a dry F1 race without any unforeseen events is between one and three (in the seasons considered). Together with the three relative compounds per race, soft (S), medium (M), and hard (H), we can now determine all the possible combinations. The regulations require every driver to use two different compounds per race. This reduces the number of possible combinations. Furthermore, in a race without opponents, the order of the tire compounds, for example, if the driver first uses the hard compound and then the soft one or vice versa, is irrelevant. This is valid if we assume that tire degradation is independent of changing track conditions and the change in vehicle mass (due to burned fuel). The former statement can be explained by the fact that the track has already been driven in three training sessions and the qualifying. The latter statement is based on the fact that the fuel mass is at most about 13% of the vehicle mass (in the 2019 season). We therefore assume that it does not influence the result too much. This is supported by the assumption that, with reduced mass, the drivers simply drive at higher accelerations, thus keeping tire degradation at a similar level. We thus obtain the following possible combinations (referred to from this point as ‘sets’):

- 1-stop strategies: SM, SH, MH
- 2-stop strategies: SSM, SSH, SMM, SMH, SHH, MMH, MHH
- 3-stop strategies: SSSM, SSSH, SSMM, SSMH, SSHH, SMMM, SMMH, SMHH, SHHH, MMMH, MMHH, MHHH

As mentioned, the start compound is fixed for each driver who qualifies in the top ten. In this case, the number of sets decreases even further. In general, however, 22 possible sets remain per driver.

3.4.2. Determination of Optimum Stint Lengths for Each Set

Optimum stint lengths are those with the minimum race duration for the respective set. When omitting driver interactions, a driver’s race duration is simply the sum of his lap times. Looking at Equation (1), we find that there are four lap-dependent elements: t_{grid} , t_{fuel} , $t_{\text{pit,in-lap/out-lap}}$, and t_{tire} . The grid time loss and time loss due to fuel mass occur independently of the race strategy and can therefore be omitted when determining optimum stint lengths. For a known amount of pit stops per set, pit time losses are also fixed. Thus, under the afore mentioned assumptions, the race duration of a given set depends only on the time loss due to tire degradation and, therefore, on the stint lengths.

A set’s optimum stint lengths can either be determined by calculating the race durations for all possible combinations of stint lengths (1 lap/69 laps, 2 laps/68 laps, etc. in a race with 70 laps), or, in the case of a linear tire degradation model, by solving a mixed-integer quadratic optimization problem (MIQP). Solving the MIQP is much faster, notably with the 2- and 3-stop strategies. Especially with no very detailed timing data, the linear degradation model is mostly used anyway, as it is the most robust model. The formulation of the MIQP is introduced in the following.

With l_{tot} being the total number of laps in a race, the objective of the MIQP reads as follows:

$$\min t_{\text{race}}(l_{\text{tot}}) \hat{=} \min \sum_{l=1}^{l_{\text{tot}}} t_{\text{tire}}(l). \quad (4)$$

Splitting the race into a fixed number of stints N (pit stops = $N - 1$) with stint index i , compounds c_i and stint lengths α_i as optimization variables, the problem can be formulated as follows:

$$\begin{aligned} & \min_{[\alpha_1 \dots \alpha_N]} \sum_{i=1}^N \sum_{a=1}^{\alpha_i} t_{\text{tire}}(a, c_i) \\ & \text{subject to } \sum_{i=1}^N \alpha_i = l_{\text{tot}} \\ & \alpha_i \in \mathbb{N}^+ \quad \forall 1 \leq i \leq N. \end{aligned} \quad (5)$$

Next, we introduce the linear tire degradation model. Thus, the time loss t_{tire} is determined by tire age a and two coefficients k_0 and k_1 , which are dependent on the tire compound c :

$$t_{\text{tire}}(a, c) = k_0(c) + k_1(c) \cdot a. \quad (6)$$

In general, a softer tire is faster in the beginning (k_0 is smaller than for a harder compound), but also displays a faster degradation (k_1 is higher than for a harder compound). Inserting the tire degradation model into Equation (5), we obtain:

$$\sum_{i=1}^N \sum_{a=1}^{\alpha_i} t_{\text{tire}}(a, c_i) = \sum_{i=1}^N \left(k_0(c_i) \cdot \alpha_i + k_1(c_i) \sum_{a=1}^{\alpha_i} a \right). \quad (7)$$

Using the Gaussian sum for the last part and rewriting $k_0(c_i)$ and $k_1(c_i)$ as $k_{0,i}$ and $k_{1,i}$, respectively, gives:

$$\sum_{i=1}^N \sum_{a=1}^{\alpha_i} t_{\text{tire}}(a, c_i) = \sum_{i=1}^N \left(k_{0,i} \cdot \alpha_i + k_{1,i} \left(\frac{1}{2} \alpha_i^2 + \frac{1}{2} \alpha_i \right) \right). \quad (8)$$

For the case that the tires are not new at the start of a stint, for instance for the top ten starters, a starting age $a_{s,i}$ is introduced into the formulation. Accordingly, α_i is replaced by $\alpha_i + a_{s,i}$ and the offset loss of $a_{s,i}$ is subtracted:

$$\begin{aligned} \sum_{i=1}^N \sum_{a=1}^{\alpha_i} t_{\text{tire}}(a, c_i, a_{s,i}) &= \sum_{i=1}^N \left(k_{0,i} (\alpha_i + a_{s,i}) + k_{1,i} \left(\frac{1}{2} (\alpha_i + a_{s,i})^2 + \frac{1}{2} (\alpha_i + a_{s,i}) \right) \right. \\ &\quad \left. - \left(k_{0,i} \cdot a_{s,i} + k_{1,i} \left(\frac{1}{2} a_{s,i}^2 + \frac{1}{2} a_{s,i} \right) \right) \right). \end{aligned} \quad (9)$$

Switching to vector notation, we obtain the final formulation:

$$\begin{aligned} \min_{[\alpha_1 \dots \alpha_N]} \quad & \vec{\alpha}^T H \vec{\alpha} + f^T \vec{\alpha} \\ \text{subject to} \quad & \sum_{i=1}^N \alpha_i = l_{\text{tot}} \\ & \alpha_i \in \mathbb{N}^+ \quad \forall 1 \leq i \leq N, \end{aligned} \quad (10)$$

where

$$H = \begin{bmatrix} 0.5 \cdot k_{1,1} & 0 & \dots & 0 \\ 0 & 0.5 \cdot k_{1,2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0.5 \cdot k_{1,N} \end{bmatrix}, \quad (11)$$

$$f = \begin{bmatrix} k_{1,1}(0.5 + a_{s,1}) + k_{0,1} \\ \vdots \\ k_{1,N}(0.5 + a_{s,N}) + k_{0,N} \end{bmatrix}. \quad (12)$$

Since the total number of laps in a race is known, the length of the last stint can be calculated by subtracting the sum of the lengths of the previous stints from it. This alternative formulation reduces the number of optimization variables by one. However, it causes coupling terms in H , making it more difficult to automatically create the matrix in relation to a given number of pit stops. We therefore opted for the decoupled version.

4. Results

First, the optimization of a basic race strategy is presented. Then, the results of the neural networks are discussed.

4.1. Optimization of Basic Race Strategies

We used ECOS_BB, which is part of the CVXPY Python package [49], to solve the MIQP. The computation time for the 22 sets in an exemplary 55 laps race is about 150 ms on a standard computer (Intel i7-6820HQ). This includes the time needed for creating the sets, setting up the optimization problem, and performing the post-processing. In comparison, the computation time needed for a brute-force approach is about 20 s. However, should we wish to use a different tire degradation model or take into account further lap-dependent effects, it is also a valid option.

The results of the exemplary 1-stop race shown in Figure 1 were calculated using the brute-force method by simulating all possible combinations of stint lengths. Solving the MIQP for the parameters in Table 8 results in the following optimum stint lengths for the two compound combinations in the figure:

- S for 22 laps + M for 33 laps resulting in a race duration of 5237.65 s
- S for 18 laps + H for 37 laps resulting in a race duration of 5242.07 s

Checking the results against Figure 1 proves that these are the fastest stint length combinations. The fastest 2-stop and 3-stop strategies for this race lead to race durations of 5240.95 s and 5252.23 s, respectively. Consequently, the 1-stop strategy is, in this case, also the fastest overall.

Table 8. Parameters of an example race used to optimize the basic race strategy.

Option	Value
Number of laps in the race	$l_{\text{tot}} = 55$ laps
Minimum number of pit stops	1
Maximum number of pit stops	3
Start compound	Soft
Tire age at race start	$a_s = 2$ laps
Degradation model parameters (hard)	$k_0 = 1.2$ s, $k_1 = 0.016$ s/lap
Degradation model parameters (medium)	$k_0 = 0.5$ s, $k_1 = 0.05$ s/lap
Degradation model parameters (soft)	$k_0 = 0.0$ s, $k_1 = 0.09$ s/lap

The intention behind determining the fastest basic strategy is that it can be used as a basis for the remaining pit stops feature. We therefore stipulated that each additional pit stop must provide an advantage of at least 2 s for us to consider the respective strategy to be the fastest basic strategy. This is because, in reality, with every pit stop there is the risk of losing significantly more time than planned, for instance if the tires cannot be fastened immediately. Consequently, an additional pit stop must provide a considerable advantage to outweigh the extra risk.

4.2. Automation of Pit Stop Decisions Using a Neural Network

4.2.1. Choice of Neural Network Hyperparameters

We use TensorFlow [50] and the Keras API to program the NNs in Python. Training is performed until there is no decrease in loss on the validation data for five consecutive iterations ('early stopping with patience'). For selecting the hyperparameters, we started with recommendations from Geron [48], evaluated many different combinations, and compared them in terms of their prediction quality, i.e., F_1 score. The parameters finally chosen are given in Table 9.

Table 9. Hyperparameters of the neural network for the pit stop decision.

Hyperparameter	Value
Number of hidden layers	3
Number of neurons per layer	64
Activation functions	ReLU (hidden layers), sigmoid (output layer)
L2 regularization	0.0005
Optimizer	Nadam
Loss function	Binary cross entropy
Training class weights	no pit stop: 1, pit stop: 5
Training batch size	256

The resulting feed-forward NN (FFNN) consists of three fully connected hidden layers of 64 neurons each. In addition to early stopping, L2 regularization is applied to decrease the tendency of overfitting. Class weights are used to increase the loss due to incorrect predictions of the pit stop class, which forces the NN to improve on these predictions during training. Otherwise, it would mostly predict no pit stop, simply because this already results in a minimum loss due to the imbalanced data. The batch size is chosen so large to have a decent chance that each batch contains some pit stops. With these hyperparameters, an average of $F_1 \approx 0.35$ is reached on the test data in the evaluation pipeline.

Figure 6 shows the progressions of the NN output for three drivers when inserting real data from the 2019 Brazilian Grand Prix that was excluded from the training data. The output can be interpreted as the predicted pit stop probability. One can generally observe slightly s-shaped curves that drop back down towards zero once the real pit stops have taken place. Several jumps are visible, which are caused by changes in position, FCY status, tire changes of pursuers, and the close ahead feature. The massive jump at Verstappen's last pit stop (at a race progress of about 75%), for example, is caused by a safety car phase. It can also be seen that the NN output remains consistently around zero after the last pit stop of a race.

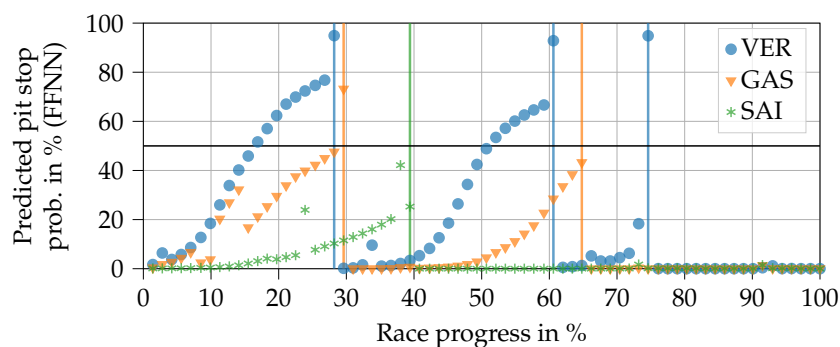


Figure 6. Progressions of pit stop probabilities for three drivers predicted by a feed-forward NN (FFNN) using real input data from the 2019 Brazilian Grand Prix. The vertical lines indicate that a real pit stop was made by the respective driver. The horizontal line at 50% probability separates the no pit stop from the pit stop region. The race was excluded from the training data. Driver abbreviations: VER—Verstappen, GAS—Gasly, SAI—Sainz.

The plot helps to explain why the FFNN does not achieve a better F_1 score. For example, the NN already predicts Verstappen's first and second pit stops for eight laps and six laps respectively, before they have taken place. In contrast, two stops by Sainz and Gasly are not predicted (in fact, they would be predicted a few laps too late). Similar problems occur with other races. We conclude that, in principle, the FFNN captures the relationships between inputs and outputs well. However, it does not succeed in sufficiently delaying the increase in output probability during an ordinary race

progress, while still allowing it to increase quickly in the case of sudden events, e.g., FCY phases. This would require input data from more than a single lap.

We therefore switched to recurrent neural networks (RNNs), which operate with time-series input data. Their neurons store an internal state, which allows them to learn relationships between consecutive steps. In our case, we insert series of consecutive laps. Pre-processing of these series is as described in Section 3.2.2. Instead of a single lap, units of several consecutive laps are processed with the same pipeline. The only difference is that time series that include laps before the first lap must be artificially filled with ‘neutral values’, cp. Table 10.

Table 10. Example time series input data for a length of 4 laps showing how values before lap 1 are artificially filled with ‘neutral values’.

Feature	Lap -1	Lap 0	Lap 1	Lap 2
Race progress	0.0	0.0	0.02	0.04
Tire age progress	0.04	0.04	0.06	0.08
Position	3	3	2	2
Relative compound	Soft	Soft	Soft	Soft
Race track category	2	2	2	2
FCY status	0	0	0	0
Remaining pit stops	2	2	2	2
Tire change of pursuer	False	False	False	False
Close ahead	False	False	True	True

As with the FFNN, many different architectures were tested. We found that pure RNNs can reach high scores on the test data ($F_1 \approx 0.9$), but tend to output many incomprehensible predictions. An example of this is given in Figure 7. In particular, Sainz’ predictions in the last part of the race are clearly wrong, since the remaining pit stops feature is zero there. This demonstrates that the F_1 score alone has little significance. The predictions of the RNN jump strongly from lap to lap, which is why it predicts significantly less false positives and false negatives than the FFNN, where the predicted probability increases slowly but steadily. However, as the example shows, the pit stop predictions of the RNN are often placed completely incorrectly. This is much worse for making automated pit stop decisions than predicting the pit stop just a few laps too early or too late, as was the case with the FFNN.

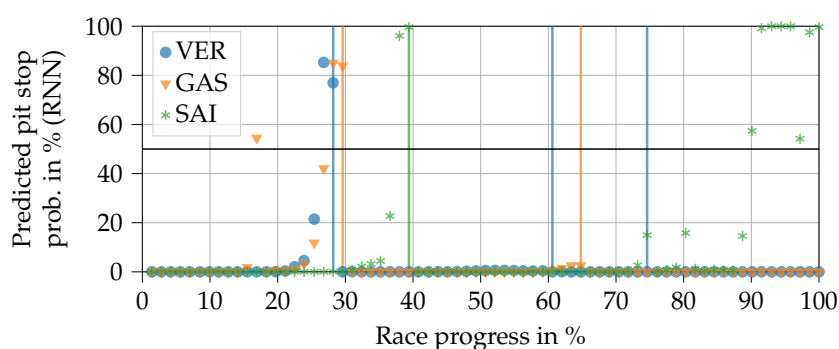


Figure 7. Progressions of pit stop probabilities for three drivers predicted by a recurrent NN (RNN) using real input data from the 2019 Brazilian Grand Prix. The vertical lines indicate that a real pit stop was made by the respective driver. The horizontal line at 50% probability separates the no pit stop from the pit stop region. The race was excluded from the training data. Driver abbreviations: VER—Verstappen, GAS—Gasly, SAI—Sainz.

We therefore combined the advantages of both types by creating a hybrid NN. The following procedure gave the best results in our tests. The FFNN is first created and trained as described.

Then a new model with an equivalent structure to that of the FFNN is created, and a single LSTM (long short-term memory) neuron is added after the original output. Subsequently, the weights for all layers that existed in the FFNN are copied to the hybrid NN and marked as untrainable. Finally, the hybrid NN is trained with data sequences. Only the weights of the LSTM neuron are then adjusted. It thus learns to modify the output of the FFNN such that its predictions become increasingly accurate.

Table 11 gives an overview of the additional hyperparameter choices. More than one RNN neuron as well as other neuron types did not result in an improvement in prediction quality. We found that a sequence length of four laps provides the best compromise between prediction quality and inference time. Increasing this to five laps would, for example, increase the inference time by about 10% for a barely measurable improvement in prediction quality. The final configuration achieved an average score of $F_1 \approx 0.59$ on the test data in the evaluation pipeline.

Table 11. Hyperparameters of the neural network for the pit stop decision (RNN section).

Hyperparameter	Value
Number of RNN neurons	1
RNN neuron type	LSTM
Sequence length	4 laps

Figure 8 reveals a resulting plot shape of the hybrid NN output resembling a hockey stick. Compared to the FFNN, for example, the curves rise later and more steeply for Verstappen. Accordingly, there is a smaller number of false positives. As can be seen from Verstappen's last pit stop, which was triggered by a safety car phase, the hybrid NN can pass through sudden jumps of the FFNN section despite the LSTM cell. Compared to pure RNN, the predictions of the hybrid NN can be understood far better. In this example, the hybrid NN does not improve on the two pit stops by Sainz and Gasly compared to the FFNN. However, at least for Gasly's second stop, the predicted probability would have exceeded 50% only one lap later. The plot also shows that the output does not drop back down to zero directly after a pit stop, unlike with the FFNN. This is due to the LSTM cell. However, we have not observed any case in which the value remains above the 50% line (even in 'worst case scenarios').

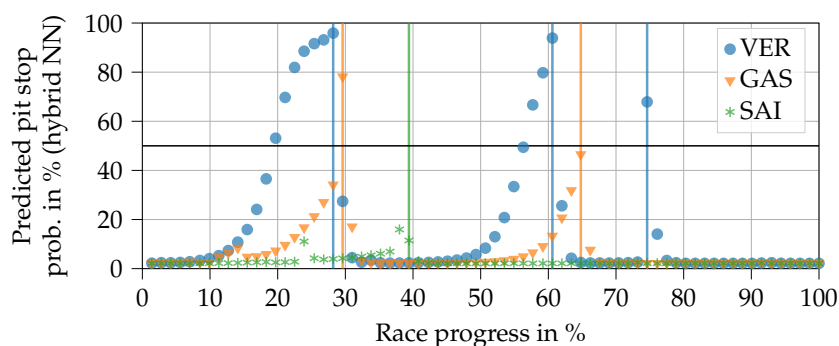


Figure 8. Progressions of pit stop probabilities for three drivers predicted by a hybrid NN using real input data from the 2019 Brazilian Grand Prix. The vertical lines indicate that a real pit stop was made by the respective driver. The horizontal line at 50% probability separates the no pit stop from the pit stop region. The race was excluded from the training data. Driver abbreviations: VER—Verstappen, GAS—Gasly, SAI—Sainz.

Table 12 contains the confusion matrix when the final trained hybrid NN predicts the pit stops on the entire database (filtered according to Section 3.2.2). It can be seen that there are 2048 false positives that are mostly caused by the fact that pit stops are often already predicted for several laps before they

actually take place. Overall, 537 pit stops happen without the NN predicting them. This is caused by cases that are either atypical (e.g., unusual early pit stops) or cannot be fully explained by the existing features. Despite the worse F_1 score, we consider the hybrid NN superior to the RNN due to its more comprehensible decision behavior.

Table 12. Confusion matrix of the predictions of the final trained pit stop decision NN on the entire database (filtered according to Section 3.2.2).

		Prediction	
		No pit stop	Pit stop
Truth	No pit stop	57,836	2048
	Pit stop	537	1855

4.2.2. Analysis of Feature Impact

To examine the plausibility of the final trained NN, we checked the predictions for real-world races. We also investigated its reactions to variations in single input features and determined whether they matched expectations. An example 1-stop race with standard values according to Table 13 was created for this purpose. Three example analyses are presented in the following.

Table 13. Standard values used to create the example race for the feature impact analysis of the pit stop decision NN.

Feature	Standard Value
Race progress	[0.0, 1.0]
Tire age progress	[0.0, 1.0]
Position	2 (pursuer)
Relative compound	Soft
Race track category	2
FCY status	0
Remaining pit stops	1
Tire change of pursuer	False
Close ahead	False

Figure 9 shows the predictions for all three relative compounds when the driver does not make a pit stop. As expected, pit stop probabilities increase steadily until the end of the race. It is also in line with expectations that the soft compound reaches the 50% line first, followed by medium and then hard. It seems surprising that the pit stops are not generally predicted earlier, so that the values match those in Table 3. This can be explained by the fact that many pit stops, in reality, are triggered not only by tire age but also by other factors, e.g., FCY phases. There are no such triggers in our exemplary race.

Figure 10 contains the prediction curves for a 1-, 2-, and 3-stop race respectively (varying the remaining pit stops feature). In this example, the pit stops are performed as soon as the predicted probability exceeds 50%. The soft compound is replaced by the medium compound in the first pit stop to comply with regulations. In subsequent pit stops, the soft compound is fitted. The pit stop in the 1-stop race takes place rather late on, considering that the soft compound is used in the first stint. This was already reasoned for the previous plot. In the 2-stop race, the first stop is placed at a race progress of 30%, which meets our expectations. With a length of 35% race progress each, the second and third stints are equivalently long. Knowing that the middle stint is run on medium and the last on soft, we would expect the middle stint to be longer. However, the NN does not know which compound will be fitted next, which is a disadvantage of the separated decision process. The 3-stop race decisions fit in well with expectations. The second stint on medium tires is the longest, while all others are relatively short.

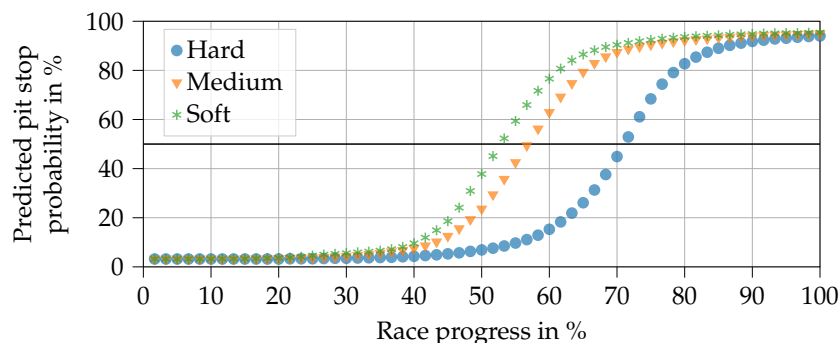


Figure 9. Progressions of pit stop probability predictions in an exemplary race when varying the relative tire compound, as outputted by the final trained hybrid NN. The horizontal line at a 50% probability level separates the no pit stop from the pit stop region.

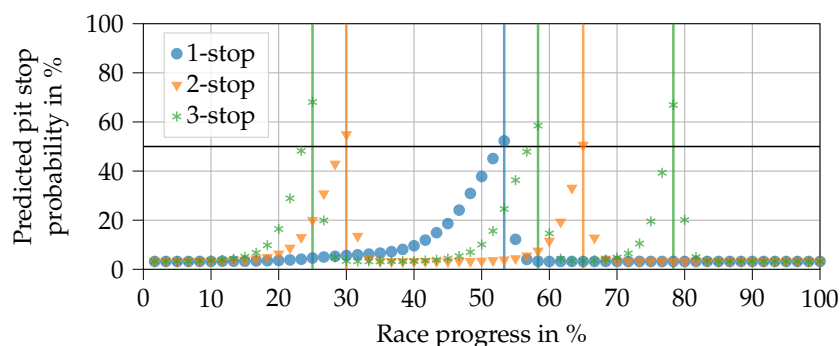


Figure 10. Progressions of pit stop probability predictions in an exemplary race when varying the intended number of pit stops as outputted by the final trained hybrid NN. The vertical lines indicate a pit stop that was actually performed. The horizontal line at a 50% probability level separates the no pit stop from the pit stop region.

Figure 11 shows the predictions when an SC is deployed in laps 10 and 15, each with a duration of three laps. A possible undercut situation can also be seen in laps 15 and 25. This is done by setting the tire change of pursuer feature to true in these laps along with the close ahead feature being set to true until lap 25. It can be seen that the early SC phase at a race progress around 17% does not trigger a pit stop, whereas the second one does. This is in line with expectations since, in reality, drivers often refrain from entering the pits if an SC phase is deployed at the beginning of a race, when the tires are still fresh. From the predictions with undercut attempts, we can see that the NN reacts well and decides for a pit stop if the attempt is close to a pit stop that is planned anyway.

4.3. Automation of Tire Compound Decisions Using a Neural Network

4.3.1. Choice of Neural Network Hyperparameters

As with the pit stop decision, many hyperparameter sets were trained and compared. Table 14 gives an overview of the chosen values. The NN is established with a single layer of 32 neurons. The loss and activation functions of the output layer are adapted to the case of three possible compound choices. In the final configuration, we achieved an average accuracy of $acc \approx 0.77$ on the test data in the evaluation pipeline. Table 15 shows the confusion matrix for the predictions of the final trained NN on the entire database (filtered according to Section 3.3.2). As the accuracy already indicates, the majority of decisions are correctly predicted.

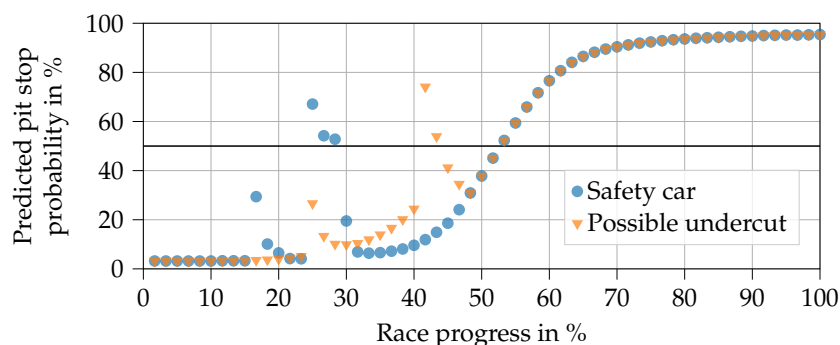


Figure 11. Progressions of pit stop probability predictions in an exemplary race as outputted by the final trained hybrid NN. The plot shows the reactions to safety car phases and possible undercut attempts. The horizontal line at a 50% probability level separates the no pit stop from the pit stop region.

Table 14. Hyperparameters of the neural network for the tire compound decision.

Hyperparameter	Value
Number of hidden layers	1
Number of neurons per layer	32
Activation functions	ReLU (hidden layer), softmax (output layer)
L2 regularization	0.001
Optimizer	Nadam
Loss function	Sparse categorical cross entropy
Training batch size	32

Table 15. Confusion matrix of the predictions of the final trained compound choice NN on the entire database (filtered according to Section 3.3.2).

		Prediction		
		Hard	Medium	Soft
Truth	Hard	444	95	62
	Medium	75	1123	147
	Soft	39	130	642

Figure 12 visualizes the NN output using the real input data for Gasly in the 2019 Brazilian Grand Prix. As can be seen, Gasly started with the soft compound, then changed to a set of medium tires, switching back to soft tires for the final stint. In this example, the NN predicts both compound choices correctly.

The progression of the curves are in line with expectations. In the first stint, the medium compound is more likely to be picked than the hard compound, since the NN knows that there are two stops remaining. It can also be seen that the attached soft compound has a probability of almost zero to be chosen. This happens to fulfill the rule that requires every driver to use two different compounds per race. Further analyses of various races and drivers showed that the NN meets this requirement consistently, presumably due to the remaining pit stops and fulfilled second compound features. At the beginning of the second stint, the hard compound has the highest probability (at a race progress of 30% to 40%). This would allow the driver to finish the race on the tires if he returned to the pits directly after the first stop. As the race progresses, the probability of choosing medium rises and falls until soft is the most likely choice. This is precisely where the second pit stop takes place. In the third stint, the remaining race progress is so small that it would only make sense to choose the soft compound again if another pit stop occurred.

If we have a race with only two available compounds (as in seasons 2014 and 2015), the NN's hard compound output remains zero. Such a case is shown in Figure 13.

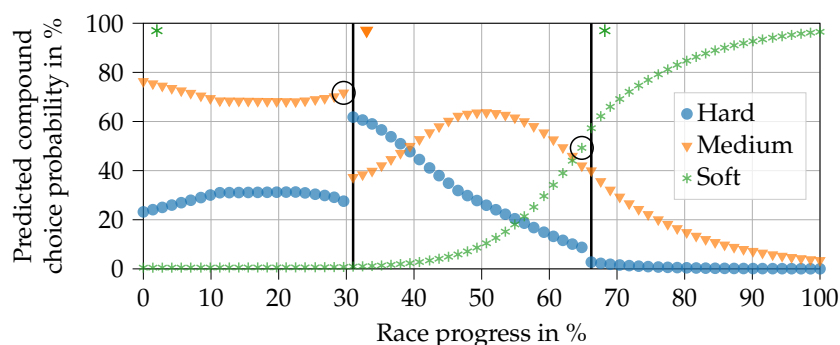


Figure 12. Progressions of the predicted compound choice probabilities for Gasly using real input data from the 2019 Brazilian Grand Prix. The vertical lines indicate a real pit stop. The symbols to the right of the lines at the top of the figure represent the actual compound fitted in the corresponding stint. The race was excluded from the training data.

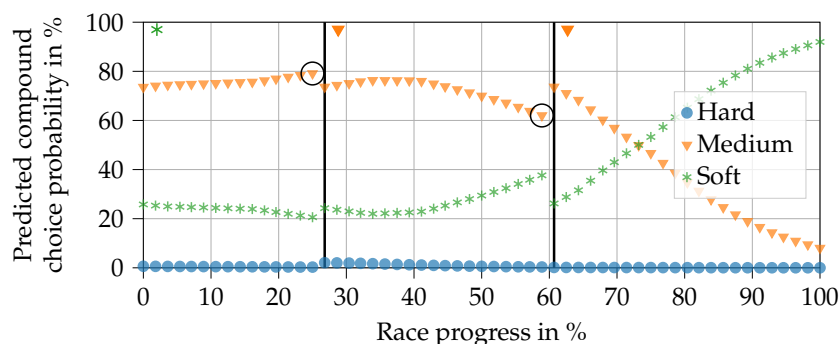


Figure 13. Progressions of the predicted compound choice probabilities for Rosberg using real input data from the 2014 United States Grand Prix. The vertical lines indicate a real pit stop. The symbols to the right of the lines at the top of the figure represent the actual compound fitted in the corresponding stint. The race was excluded from the training data.

4.3.2. Analysis of Feature Impact

As before, we can analyze the behavior of the compound decision NN by varying single input features. Again, a 1-stop race was created with standard values, as shown in Table 16. The example analysis is shown in Figure 14.

Table 16. Standard values used to create the race for the feature impact analysis of the compound decision NN.

Feature	Standard Value
Race progress	[0.0, 1.0]
Remaining pit stops	1
Relative compound	Soft
Race track	Austin
Fulfilled second compound	False
Number of avail. compounds	3

The top plot shows that during the first 40% of the race, the NN selects the hard compound, even though the hard compound is already fitted to the car and only a single pit stop is planned. This may appear unexpected at first but it can be explained. Firstly, most drivers begin the race with a soft or medium compound, which is why virtually no training data are available for a hard compound in the first stint, cp. Table 17. Secondly, if a driver starts with the hard compound, it is unlikely that he

will change his tires before a race progress of 40%, especially if he is on a 1-stop strategy. Consequently, the first section of the prediction is not relevant, since the pit stop will usually take place in the middle of the race at the earliest.

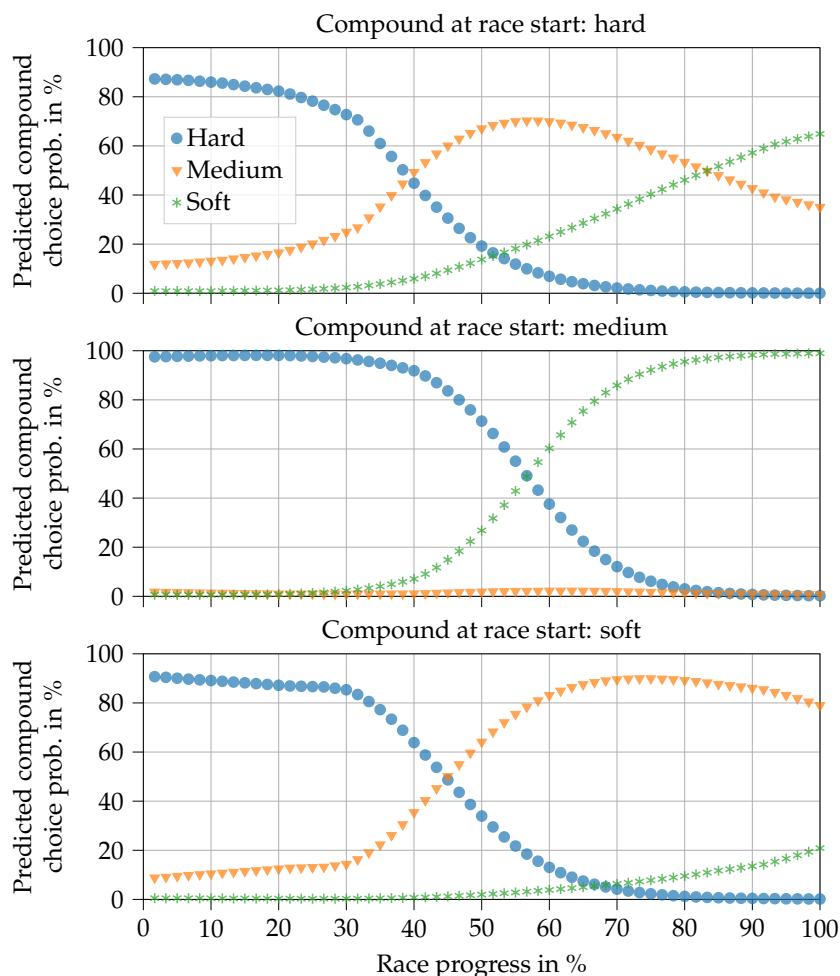


Figure 14. Progressions of the compound choice probability predictions in an example race, as outputted by the final trained NN. The compound fitted to the car at the start of the race is hard in the top plot, medium in the middle plot, and soft in the bottom plot.

Table 17. Overview of tire compounds fitted at the race start. The values apply to data filtered according to Section 3.3.2.

Seasons	Start on Hard	Start on Medium	Start on Soft
≤2015	–	11.5%	88.5%
≥2016	4.4%	32.5%	63.1%
Overall	3.1%	26.0%	70.9%

The middle and bottom plots are in line with expectations. In the first part of the race, the NN chooses the hard compound. In the second part, it chooses either soft or medium, depending on the compound that is currently fitted to the car. Thus, the NN satisfies the regulations. It can also be seen that the soft compound is chosen above a race progress of 55% (middle plot), whereas the medium compound is selected above 45% (bottom plot). This is in line with the differences in durability between the two compounds.

4.4. Integrating the Virtual Strategy Engineer into the Race Simulation

Figure 15 gives an overview of the integration of the VSE into the race simulation. As can be seen, the VSE is called once per lap and driver to make the strategy decision.

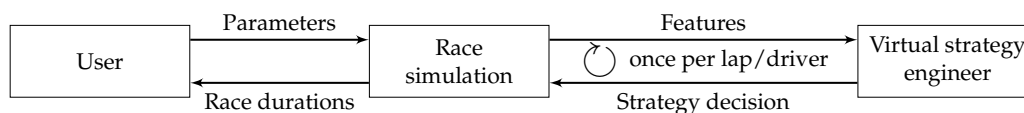


Figure 15. Overview of the integration of the virtual strategy engineer into the race simulation.

The race simulation is implemented in the Python programming language. One simulation run for an example race with 20 drivers and 55 laps takes about 100 ms on a standard computer (Intel i7-6820HQ) if the strategy decisions are fixed. Since thousands of runs must be performed for Monte Carlo simulation, the time per run must remain in this magnitude, even if the VSE determines the participants' strategies (although the runs are independent of each other and can be executed in parallel on CPUs with multiple cores). We could only achieve this by using TensorFlow Lite [51], a reduced framework whose original purpose is to bring ML functionality to devices such as mobile phones. Consequently, the trained models were converted into TensorFlow Lite models. Afterward, they only contain the functionality required for inferencing. This enables us to achieve calculation times of about 200 ms per simulation run.

The race simulation allows us to compare the results when using different sources on which to base strategy decisions. For the analysis, we use simulation parameters that were fitted for the 2019 Austrian Grand Prix. The race is a good example because, in reality, there were no accidents or breakdowns, and therefore no FCY phases. Five variants of race strategy determination will be analyzed:

- Variant 1: All drivers use the strategy they have used in the real race
- Variant 2: Hamilton uses his optimized basic strategy, all other drivers use their real-race strategy
- Variant 3: The VSE determines Hamilton's strategy, all other drivers use their real-race strategy
- Variant 4: The VSE determines Bottas' strategy, all other drivers use their real-race strategy
- Variant 5: The VSE determines all drivers' strategies

Hamilton and Bottas were chosen as test drivers because they achieve average result positions of 3.4 and 3.8 in the simulation with their real-race strategies, which allows the VSE to improve with different strategies. The real-race strategies of all drivers are given in Table A2, in the Appendix A. Hamilton's basic strategy was determined as medium—medium (from lap 29)—soft (from lap 60) using the optimization algorithm presented. The VSE is based on the final trained NNs that were analyzed in Sections 4.2.2 and 4.3.2.

We performed 10,000 simulation runs for each variant, which enabled us to make a reasonable estimate of the average result positions [3]. Probabilistic influences (lap time variability, race start performance, pit stop duration variability) were activated. FCY phases were deactivated for the first stage, to be added later. We furthermore deactivated retirements for all variants such that the result positions were based solely on racing and strategy decisions. The results of this first stage are shown in Table 18. For simplicity, we display only the top five drivers of the simulated races.

First, it is noticeable that Hamilton's result position in reality (five) differs strongly from that of the simulation with the real-race strategies (3.4 in Variant 1), which also causes large deviations for Bottas and Vettel. In the present case, this can be explained by the fact that Hamilton received a new front wing during his pit stop in reality, which increased the pit stop time loss by about 8 s. This is not included in the simulation. In general, it must also be considered that a race simulation cannot perfectly reproduce a real race, for instance, because of model inaccuracies or parameterization errors.

Furthermore, the progress of the race changes with different strategies. Consequently, the following results can only be compared with each other and not with a real-world race.

Table 18. Real and average result positions of the top five drivers after 10,000 simulation runs of the 2019 Austrian Grand Prix. Five different variants of race strategy determination are compared. Probabilistic influences were activated, FCY phases were deactivated.

Driver	Real Race	Variante 1	Variante 2	Variante 3	Variante 4	Variante 5
Verstappen	1	1.2	1.2	1.2	1.2	1.0
Leclerc	2	2.0	2.1	2.2	2.0	2.5
Hamilton	5	3.4	3.1	2.9	3.4	3.6
Bottas	3	3.8	4.0	4.1	3.7	3.6
Vettel	4	5.7	5.7	5.7	5.9	7.4

Hamilton achieves an average position of 3.4 using his real strategy (Variant 1). By switching to the optimized basic strategy (Variant 2), this can be improved to 3.1. This happens mainly at the expense of Bottas, who drops from 3.8 to 4.0. The VSE (Variant 3) results in further improvement to 2.9. Using the VSE for Bottas has almost no impact on the average result position (Variant 4). If all drivers in the simulated race are led by the VSE (Variant 5), we can see that the result positions are quite different. Some drivers profit from the VSE decisions, e.g., Verstappen and Bottas, while others perform worse, e.g., Hamilton and Vettel.

In the second stage, we repeated the entire procedure with activated FCY phases. These are randomly generated according to [3]. The results are given in Table 19.

Table 19. Real and average result positions of the top five drivers after 10,000 simulation runs of the 2019 Austrian Grand Prix. Five different variants of race strategy determination are compared. Both probabilistic influences and randomly generated FCY phases were activated.

Driver	Real Race	Variante 1	Variante 2	Variante 3	Variante 4	Variante 5
Verstappen	1	1.3	1.4	1.4	1.3	1.2
Leclerc	2	2.1	2.1	2.3	2.1	2.6
Hamilton	5	3.5	3.4	2.7	3.6	3.6
Bottas	3	3.9	4.0	4.1	3.5	3.6
Vettel	4	5.9	5.9	6.0	6.2	7.2

With activated FCY phases, we can see that the results are slightly worse than Variant 1 in the previous table. This is because FCY phases often allow drivers at the back to catch up with those at the front, which can involve overtaking maneuvers as soon as the phase is over. It is also evident that the basic strategy (Variant 2) improves Hamilton's average result only slightly because it can hardly play out its optimality here (it was determined for a free track and without FCY phases). Variant 3 indicates that Hamilton benefits massively from the VSE that reacts to FCY phases in the simulation. His average result position improves to 2.7. For Bottas, the VSE does not change that much, but it is still remarkable (Variant 4). Interestingly, in Variant 5, the results are quite similar to those of the first stage without FCY phases. This is in line with expectations, since the FCY phases do not change whether the VSE's decision characteristics fit the corresponding driver parameterization well.

In conclusion, we can say that an optimized basic strategy can improve the results. However, this, of course, depends on the strategy used in comparison, as well as on the driver parameterization. Furthermore, the advantage disappears as soon as FCY phases are considered. The improvement that can be achieved by the VSE depends on whether the combination of decision characteristics and driver parameterization are compatible or not. In some cases, we also saw that the VSE worsens the result of the driver. However, particularly with FCY phases, the VSE often improves the result, due to its ability to react to the race situation.

5. Discussion

Starting with the simplest race simulation variant—a single driver on a free track with no probabilistic influences—an optimization problem can determine the fastest race strategy. The MIQP cannot only be used to set the remaining pit stops feature for the VSE, but also to obtain an initial strategic impression of a race. This is supported by fast computation times. On the downside, however, the optimization problem formulation is based on several assumptions from which the race can differ significantly. This includes the assumption of a free race track, as well as the assumption that the tire degradation can be described by a linear model and is independent of the changing vehicle mass. Furthermore, the optimization result is hugely dependent on the parameters of the tire degradation model. Consequently, the less these assumptions apply in a race, for instance, due to an FCY phase, the worse the basic strategy performs.

The VSE was developed to make automated race strategy decisions in a race simulation in all situations. We can conclude that, in general, it makes reasonable decisions and reacts consistently with the race situation, for instance, in the case of FCY phases or undercut attempts. Fast inference times make it suitable for use with any number of drivers, and its training on the entire database makes it universally applicable across all drivers, teams, and seasons. Using real-world data also prevents the VSE from learning modeling errors or parameterization errors, as might occur with reinforcement learning.

However, training on the entire database is also a disadvantage. Drivers, cars, and teams are subject to constant development, and the regulations change from season to season. This means that the VSE is not able to learn a ‘perfect’ reaction to specific situations. Training on real data is not ideal either, because even in real races, the strategy engineers sometimes make wrong decisions. Another aspect is the limited amount of data that is available to us, which also restricts the possible features. For example, an F1 team has a lot of sensor data at its disposal, which they use to determine the tire condition. However, from an external perspective, we cannot even distinguish reliably if the lap times of a driver deteriorate due to tire degradation or due to another engine mapping. Unfortunately, with the available data, we were also unable to create features that lead the VSE to make active use of advanced tactical opportunities such as undercuts. A possible feature that we did not consider is that pit stops are in reality, if possible, placed in such a way that the driver can drive freely afterward and is stuck behind other cars. Especially if sector times were available, this could help to improve the prediction quality. Another point is that in reality, strategic decisions are not only dependent on timing data, but also on game-theoretical factors. This is difficult to train, because we do not know which opponent(s) a driver was battling against in his race and what the expected results of all strategy combinations were that led to the subsequent decisions. When using the VSE in a race simulation, the method for specifying the intended number of pit stops could be improved. The current implementation based on the optimized basic strategy has the disadvantage that the number of pit stops is not adjusted, for example to the number of FCY phases.

Several points remain for future work. Firstly, merging the two stages of the current decision-making process into a single network could improve the prediction quality, since the NN would then consider the next tire compound for the pit stop decision. Secondly, with access to more detailed and extensive data, new features could be developed for improved predictions. A better representation of the tire state seems to be a good starting point. Thirdly, we believe that at the moment, a meaningful metric is missing to compare the characteristics of different race tracks objectively. Such a metric could be beneficial for both NNs to replace the race track and race track category features. A possible starting point could be the energy that is transferred via the tires per lap. Fourthly, the application and testing of the VSE in real-world races would enable a further evaluation of its decisions. However, since this will probably not be possible for us, we will focus on further analysis in simulation and comparison with the reinforcement learning approach currently under development.

6. Outlook: Reinforcement Learning Approach

Since we have no access to any team and are dependent on publicly available data, our future work will focus on a reinforcement learning approach. The race simulation will therefore be used as an environment for training an agent to make race strategy decisions. Here, as in real-world races, the agent can choose between four actions at the end of each lap: no pit stop, pit stop (hard), pit stop (medium), pit stop (soft). Thus, the decision is not divided into two stages, as in the methodology presented above. After rendering a decision, a lap is simulated under consideration of the chosen action. Before the agent chooses its next action, it is rewarded for its previous action, in two parts. The first part is determined by the difference between the current lap time and an average lap time calculated for the driver in a pre-simulation without opponents. Thus, the agent learns that it can increase its reward by fitting a fresh set of tires. The second part considers the number of positions that the driver has won or lost since the previous decision. Consequently, the agent will learn that a pit stop costs positions. By optimizing the strategy, the agent autonomously maximizes its reward.

In its current state, the agent is trained specifically for each race. During training, the other drivers' strategies in the simulated race can be controlled by the VSE or be predefined by the user. To ensure that the agent can be used universally for all drivers and situations, the driver controlled during training is randomly selected before each training episode.

In initial tests, the agent achieved better results in the race simulation than the VSE. For the 2019 Chinese Grand Prix, in which there were 20 drivers, it achieved an average position of 8.46 over 1000 simulated races, each with a randomly selected driver. The VSE only manages an average position of 9.51. Various factors can explain the better performance of the agent. Firstly, the VSE is trained on many different real races and learns to make reasonable decisions 'on average', which may, however, not be a perfect fit for the driver, car, track, and race situation. The agent, in contrast, is trained specifically for the race in question. Secondly, the agent makes a pit stop decision in combination with the next compound, which enables better timing of the pit stop compared to the two-stage approach of the VSE. Thirdly, in contrast to the VSE, the agent does not need the intended number of pit stops for the race. This simplifies handling and allows him to amend the number of pit stops during the race if the situation changes.

The results of the initial tests seem promising, but must be evaluated in further races. Furthermore, we have to analyze whether the agent is able to learn and make active use of advanced tactical opportunities, such as the undercut. It is also necessary to check whether the agent's strategies seem reasonable, since in training based purely on simulation, it will also learn the simulation and parameterization errors.

7. Summary

This paper presents a methodology for automating race strategy decisions in circuit motorsport. The focus is on the Formula 1 racing series and, thus, on the optimal determination of pit stops to replace worn-out tires. To be able to determine an initial estimate of the optimal race strategy, a quadratic optimization problem was set up that minimizes a driver's race duration, assuming a race without opponents. However, this estimate is not sufficient to be used in a race simulation or a real-world race, as it takes neither the opponent drivers nor the particular race situation into account. Therefore, we developed a virtual strategy engineer (VSE) based on two artificial neural networks. The first one decides in each respective lap whether the driver should make a pit stop. If it chooses to call for a pit stop, a second neural network determines which of the available tire compounds should be fitted to the car. Both neural networks were trained on Formula 1 timing data of the seasons from 2014 to 2019. The presented results of a race simulation, in which the VSE is used to make the race strategy decisions, indicate that the VSE makes reasonable decisions and is able to adapt its strategy to the particular race situation. Thus, the VSE improves a race simulation's realism and can support a real strategy engineer in his decisions.

The entire Python code for the race simulation, including the trained VSE models, is available under an open-source license on GitHub (<https://github.com/TUMFTM/race-simulation>).

Author Contributions: The initial concept of automated decision-making based on neural networks was developed within the research project by A.T. under A.H.'s supervision in his semester thesis. As the first author, A.H. revised and extended the methodology. He reworked the feature selection and pre-processing, introduced the hybrid neural network, and integrated the approach into the race simulation. He also provided the following parts of the paper: database creation, formulation and implementation of the optimization problem, data analysis, and result evaluation. Moreover, he wrote most parts of the paper. After completing his semester thesis, A.T. started working on the reinforcement learning approach for his master's thesis within the research project. He wrote the outlook on this topic. As the mentor of the research project, M.G. critically scrutinized the approaches. M.G. and J.B. contributed to the conception of the research project and critically revised the paper. Conceptualization, A.H., A.T., M.G., J.B.; data curation, A.H.; formal analysis, A.H.; methodology, A.H., A.T.; software, A.H., A.T.; validation, A.H.; writing—original draft preparation, A.H., A.T.; writing—review and editing, A.H.; visualization, A.H.; supervision, M.G., J.B. All authors have read and agreed to the published version of the manuscript.

Funding: Research was supported by the basic research fund of the Institute of Automotive Technology of the Technical University of Munich.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

F1	FIA Formula 1 World Championship
FCY	Full-Course Yellow
FFNN	Feed-Forward Neural Network
LSTM	Long Short-Term Memory
MIQP	Mixed-Integer Quadratic Programming
ML	Machine-Learning
NN	Neural Network
RNN	Recurrent Neural Network
RS	Race Simulation
SC	Safety Car
VSC	Virtual Safety Car
VSE	Virtual Strategy Engineer (combination of both NNs)

Appendix A. Data Tables

Table A1. The race track categorization is based on the hardest absolute tire compound available for the respective race track in the 2019 season. Thus, Category 1 contains tracks with high tire stress, while the stress is lowest in Category 3.

Location	Category
Austin	2
Baku	2
Budapest	2
Catalunya	1
Hockenheim	2
Kuala Lumpur	2
Le Castellet	2
Melbourne	2
Mexico City	2
Monte Carlo	3
Montreal	3
Monza	2
Sakhir	1

Table A1. Cont.

Location	Category
Sao Paulo	1
Shanghai	2
Silverstone	1
Singapore	3
Sochi	2
Spa	1
Spielberg	2
Suzuka	1
YasMarina	3

Table A2. Race strategies applied in the 2019 Austrian Grand Prix. The first lap with a new compound is stated in brackets. The real compound names are C2 (hard), C3 (medium), and C4 (soft). The drivers are listed in order of result position.

Driver	Strategy
Verstappen	Medium – hard (32)
Leclerc	Soft – hard (23)
Bottas	Medium – hard (22)
Vettel	Soft – hard (22) – soft (51)
Hamilton	Medium – hard (31)
Norris	Soft – Medium (26)
Gasly	Soft – hard (26)
Sainz	Medium – hard (42)
Räikkönen	Soft – hard (24)
Giovinazzi	Soft – hard (25)
Pérez	Medium – hard (29)
Ricciardo	Medium – soft (47)
Hülkenberg	Medium – hard (27)
Stroll	Medium – hard (26)
Albon	Medium – hard (36)
Grosjean	Medium – hard (35)
Kvyat	Medium – hard (33)
Russell	Medium – hard (28)
Magnussen	Soft – hard (12) – soft (63)
Kubica	Medium – hard (20)

References

- Palmer, J. Jolyon Palmer's Analysis: Singapore and the Art of Undercutting. Available online: <https://www.formula1.com/en/latest/article.jolyon-palmers-analysis-singapore-and-the-art-of-undercutting.1NgVyVsZnHTDEA9wi0s5IW.html> (accessed on 26 August 2020).
- Heilmeier, A.; Graf, M.; Lienkamp, M. A Race Simulation for Strategy Decisions in Circuit Motorsports. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2986–2993. [CrossRef]
- Heilmeier, A.; Graf, M.; Betz, J.; Lienkamp, M. Application of Monte Carlo Methods to Consider Probabilistic Effects in a Race Simulation for Circuit Motorsport. *Appl. Sci.* **2020**, *10*, 4229. [CrossRef]
- Bunker, R.P.; Thabtah, F. A machine learning framework for sport result prediction. *Appl. Comput. Inform.* **2019**, *15*, 27–33. [CrossRef]
- Purucker, M. Neural network quarterbacking. *IEEE Potentials* **1996**, *15*, 9–15. [CrossRef]
- Kahn, J. *Neural Network Prediction of NFL Football Games*; University of Wisconsin-Madison: Madison, WI, USA, 2003.
- Delen, D.; Cogdell, D.; Kasap, N. A comparative analysis of data mining methods in predicting NCAA bowl outcomes. *Int. J. Forecast.* **2012**, *28*, 543–552. [CrossRef]

8. Leung, C.K.; Joseph, K.W. Sports Data Mining: Predicting Results for the College Football Games. *Procedia Comput. Sci.* **2014**, *35*, 710–719. [CrossRef]
9. Chen, H.; Rinde, P.B.; She, L.; Sutjahjo, S.; Sommer, C.; Neely, D. Expert prediction, symbolic learning, and neural networks. An experiment on greyhound racing. *IEEE Expert* **1994**, *9*, 21–27. [CrossRef]
10. Johansson, U.; Sonstrod, C. Neural networks mine for gold at the greyhound racetrack. In Proceedings of the International Joint Conference on Neural Networks, Portland, OR, USA, 20–24 July 2003. [CrossRef]
11. Schumaker, R.P.; Johnson, J.W. An Investigation of SVM Regression to Predict Longshot Greyhound Races. *Commun. IIMA* **2008**, *8*, 67–82.
12. Harville, D.A. Assigning Probabilities to the Outcomes of Multi-Entry Competitions. *J. Am. Stat. Assoc.* **1973**, *68*, 312–316. [CrossRef]
13. Williams, J.; Li, Y. A Case Study Using Neural Networks Algorithms: Horse Racing Predictions in Jamaica. In Proceedings of the 2008 International Conference on Artificial Intelligence, Las Vegas, NV, USA, 14–17 July 2008; pp. 16–22.
14. Davoodi, E.; Khanteymooori, A.R. Horse racing prediction using artificial neural networks. In Proceedings of the 11th WSEAS International Conference on Neural Networks, Evolutionary Computing and Fuzzy Systems, Iasi, Romania, 13–15 June 2010; pp. 155–160.
15. Joseph, A.; Fenton, N.; Neil, M. Predicting football results using Bayesian nets and other machine learning techniques. *Knowl.-Based Syst.* **2006**, *19*, 544–553. [CrossRef]
16. Arabzad, S.M.; Araghi, M.; Soheil, S.N.; Ghofrani, N. Football Match Results Prediction Using Artificial Neural Networks; The Case of Iran Pro League. *Int. J. Appl. Res. Ind. Eng.* **2014**, *1*, 159–179.
17. Tax, N.; Joustra, Y. Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach. *Trans. Knowl. Data Eng.* **2015**, *10*, 1–13. [CrossRef]
18. Prasetio, D.; Harlili, D. Predicting football match results with logistic regression. In Proceedings of the 2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA), George Town, Malaysia, 16–19 August 2016. [CrossRef]
19. Rein, R.; Memmert, D. Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science. *SpringerPlus* **2016**, *5*. [CrossRef] [PubMed]
20. Graves, T.; Reese, C.S.; Fitzgerald, M. Hierarchical Models for Permutations. *J. Am. Stat. Assoc.* **2003**, *98*, 282–291. [CrossRef]
21. Pfitzner, C.B.; Rishel, T.D. Do Reliable Predictors Exist for the Outcomes of NASCAR Races? *Sport J.* **2008**, *8*, 2.
22. Depken, C.A.; Mackey, L. Driver Success in the Nascar Sprint Cup Series: The Impact of Multi-Car Teams. *SSRN Electron. J.* **2009**. [CrossRef]
23. Allender, M. Predicting The Outcome Of NASCAR Races: The Role Of Driver Experience. *J. Bus. Econ. Res.* **2011**, *6*. [CrossRef]
24. Stoppels, E. Predicting Race Results Using Artificial Neural Networks. Master's Thesis, University of Twente, Enschede, The Netherlands, 2017.
25. Stergioudis, A. Machine-learning Based F1 Race Prediction Engine. Available online: <https://www.f1-predictor.com> (accessed on 1 July 2020).
26. Edelmann-Nusser, J.; Hohmann, A.; Henneberg, B. Modeling and prediction of competitive performance in swimming upon neural networks. *Eur. J. Sport Sci.* **2002**, *2*, 1–10. [CrossRef]
27. Miljkovic, D.; Gajic, L.; Kovacevic, A.; Konjovic, Z. The use of data mining for basketball matches outcomes prediction. In Proceedings of the IEEE 8th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, 10–11 September 2010. [CrossRef]
28. Przednowek, K.; Iskra, J.; Przednowek, K.H. Predictive Modeling in 400-Metres Hurdles Races. In Proceedings of the 2nd International Congress on Sports Sciences Research and Technology Support, Rome, Italy, 24–26 October 2014. [CrossRef]
29. Maszczyk, A.; Gołaś, A.; Pietraszewski, P.; Rocznio, R.; Zając, A.; Stanula, A. Application of Neural and Regression Models in Sports Results Prediction. *Procedia Soc. Behav. Sci.* **2014**, *117*, 482–487. [CrossRef]
30. Pretorius, A.; Parry, D.A. Human Decision Making and Artificial Intelligence. In Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists on SAICSIT, Johannesburg, South Africa, 26–28 September 2016; ACM Press: New York, NY, USA, 2016. [CrossRef]

31. McCabe, A.; Trevathan, J. Artificial Intelligence in Sports Prediction. In Proceedings of the Fifth International Conference on Information Technology: New Generations (itng 2008), Las Vegas, NV, USA, 7–9 April 2008. [CrossRef]
32. Dubbs, A. Statistics-free sports prediction. *Model Assist. Stat. Appl.* **2018**, *13*, 173–181. [CrossRef]
33. Haghighat, M.; Rastegari, H.; Nourafza, N. A Review of Data Mining Techniques for Result Prediction in Sports. *ACSII Adv. Comput. Sci. Int. J.* **2013**, *2*, 7–12.
34. Tulabandhula, T.; Rudin, C. Tire Changes, Fresh Air, and Yellow Flags: Challenges in Predictive Analytics for Professional Racing. *Big Data* **2014**, *2*, 97–112. [CrossRef]
35. Choo, C.L.W. Real-Time Decision Making in Motorsports: Analytics for Improving Professional Car Race Strategy. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2015.
36. Amazon Web Services, Inc. F1 Insights Powered by AWS. Available online: <https://aws.amazon.com/de/f1> (accessed on 1 July 2020).
37. Aversa, P.; Cabantous, L.; Haefliger, S. When decision support systems fail: Insights for strategic information systems from Formula 1. *J. Strateg. Inf. Syst.* **2018**, *27*, 221–236. [CrossRef]
38. Liu, X.; Fotouhi, A. Formula-E race strategy development using artificial neural networks and Monte Carlo tree search. *Neural Comput. Appl.* **2020**, *32*, 15191–15207. [CrossRef]
39. Gartheeban, G.; Guttag, J. A Data-Driven Method for in-Game Decision Making in MLB: When to Pull a Starting Pitcher. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining—KDD'13, Chicago, IL, USA, 11–14 August 2013; ACM Press: New York, NY, USA, 2013; pp. 973–979. [CrossRef]
40. Gartheeban, G.; Guttag, J. Predicting the Next Pitch. MIT Sloan Sports Analytics Conference 2012, Boston, MA, USA, 2–3 March 2012.
41. Bailey, M.; Clarke, S. Predicting the Match Outcome in One Day International Cricket Matches, while the Game is in Progress. *J. Sport. Sci. Med.* **2006**, *5*, 480–487.
42. Sankaranarayanan, V.V.; Sattar, J.; Lakshmanan, L.V.S. Auto-play: A Data Mining Approach to ODI Cricket Simulation and Prediction. In Proceedings of the 2014 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 24–26 April 2014; pp. 1064–1072. [CrossRef]
43. Weber, B.G.; Mateas, M. A data mining approach to strategy prediction. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Games, Milano, Italy, 7–10 September 2009. [CrossRef]
44. Newell, C. Ergast Motor Racing Developer API. Available online: <http://ergast.com/mrd> (accessed on 26 August 2020).
45. Pirelli & C. S.p.A. 2018 Slick Tires. Available online: <https://twitter.com/pirellisport/status/933679440214810624> (accessed on 26 August 2020).
46. Pirelli & C. S.p.A. Comparison of Slick Tire Compounds 2018/2019. Available online: <https://twitter.com/pirellisport/status/1067288810256355328> (accessed on 26 August 2020).
47. Pirelli & C. S.p.A. Pirelli presents new wider 2017 Formula 1 tyres. Available online: <https://www.pirelli.com/tyre/ww/en/news/2016/07/19/pirelli-presents-new-wider-2017-formula-1-tyres> (accessed on 26 August 2020).
48. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed.; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2019.
49. Diamond, S.; Boyd, S. CVXPY: A Python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.* **2016**, *17*, 2909–2913.
50. Google LLC. TensorFlow. Available online: <https://www.tensorflow.org> (accessed on 26 August 2020).
51. Google LLC. TensorFlow Lite. Available online: <https://www.tensorflow.org/lite> (accessed on 26 August 2020).

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

4.5.2 Reinforcement Learning Virtual Strategy Engineer

In the outlook of the paper [13], the concept for the reinforcement VSE is presented. Instead of training the VSE on real-world data, as it was done for the supervised VSE, it is trained exclusively within the RS. This approach is investigated and implemented in a master thesis by Thomaser [176]. Starting from the paper's outlook, this section provides some supplementary information on the basics of the reinforcement learning approach and its implementation as developed in his thesis [176].

Thematic Background and Implementation

Reinforcement learning is based on an agent autonomously learning, based on rewards, which actions are beneficial and detrimental in a given environment. In the present case, the agent has four possible actions A to choose from: no pit stop, pit stop (soft compound), pit stop (medium compound), and pit stop (hard compound) [176, p. 31]. Let the return G_l in lap l be equal to the sum of expected future rewards r as stated by [177, p. 54]

$$G_l = \sum_{i=l+1}^{l_{\text{tot}}} r_i. \quad (4.6)$$

Then, the action-value function $q_\pi(Z, A)$ expresses the expected future return G_l for a given race state Z , an action under study A , and a pursued policy π [177, p. 58]. The larger the value of $q_\pi(Z, A)$, the more beneficial the action A . The race state Z is defined by the features based on which the agent makes the decisions. This feature set is similar to that of the supervised VSE, compare [13]. New, for example, is the information about the expected number of lost positions during a pit stop. It is calculated from the average time lost during a pit stop and based on the assumption that none of the following drivers will pit. In addition, the reinforcement VSE receives the driver's name since it adapts its strategy for each driver individually. $q_\pi(Z, A)$ implicitly contains an expectation for the further course of the race and future actions chosen according to the policy π . The greedy policy is the most obvious, where the action with the highest expected q -value is chosen. During training, however, the effects of the different actions are to be explored so that a policy must be used that also selects actions that do not promise the highest return in the current training state. This can be compared to the breakout from a local optimum in evolutionary algorithms.

The expected returns must be approximated since the set of possible race states and actions is too large to test each possible combination. One of the most famous variants is deep q-learning [178], in which an artificial neural network called deep q-network is used to approximate the q -values. In the present case, a deep q-network with two layers of 64 neurons each is used [176, p. 38].

It turned out that the agent learns best when it receives a reward immediately after each decision, rather than only at the end of a race [176, p. 32]. This is due to the fact that in the latter case, it is unclear which of the 60 actions (in a race with 60 laps, as an example) has which contribution to the reward. For a balanced decision behavior, the immediate reward is composed of two parts [176, p. 32f.]:

$$r_{\text{laptime},l} = t_{\text{lap,ref}} - t_{\text{lap}}(l) \quad \text{and} \quad (4.7)$$

$$r_{\text{p},l} = 5(p_{l-1} - p_l). \quad (4.8)$$

The first part of the reward $r_{\text{lapttime},l}$ is larger the smaller (i.e., faster) the lap time of the lap after the decision $t_{\text{lap}}(l)$ is compared to a reference lap time $t_{\text{lap,ref}}$. $t_{\text{lap,ref}}$ corresponds to the average lap time of the driver in a presimulation of the race without opponents. Thus, a new set of tires that enables faster lap times leads to a higher reward. If only this first part of the reward were taken into account, the reinforcement VSE would too often opt for a pit stop to keep the lap times low with fresh tires, resulting in position losses. Therefore, the second part of the reward $r_{p,l}$ is based on the change in position within the lap after the decision. Thus, it decreases if the driver loses position due to a pit stop or old tires. The difference in positions is multiplied by five to give a higher weighting to the position loss compared to the lap time loss. The factor of five is empirically determined such that the reinforcement VSE shows a balanced behavior in which it does not constantly pit and get new tires, nor does it pit at all [176, p. 33].

At the end of the race, two additional terms are added to the total reward r_{tot} . First, the final position achieved is taken into account by $r_{p,\text{final}}$. It is based on the deviation from the reference final position p_{ref} , as stated by [176, p. 33]

$$r_{p,\text{final}} = 10(p_{\text{ref}} - p_{\text{final}}). \quad (4.9)$$

To determine p_{ref} , the previously mentioned average lap times of all drivers from the presimulation are sorted in ascending order. p_{ref} then corresponds to the rank position of the controlled driver in this list. Second, r_{diffcomp} is introduced to reduce the reward by 100 if the agent did not use two different tire compounds during the race. This factor was also determined empirically, so the agent adheres to the rule. The total reward of a race r_{tot} can then be calculated by [176, p. 34]

$$r_{\text{tot}} = \sum_{l=2}^{l_{\text{tot}}} (r_{\text{lapttime},l} + r_{p,l}) + r_{p,\text{final}} + r_{\text{diffcomp}}. \quad (4.10)$$

As can be seen, the first lap ($l = 1$) is excluded since the agent cannot influence it (the earliest opportunity for a pit stop is at the end of the first lap).

Training the reinforcement VSE for a particular race is done by simulating that race thousands of times with probabilistic effects enabled. The probabilistic effects in the simulation are crucial for the agent to see as many different variants of a race as possible during training and thus store the q -values in the deep q-network for as many race states as possible. For the reinforcement VSE to learn meaningful strategy decisions, all drivers in a race must react to the individual race situation with their strategy, e.g., in the case of FCY phases. Consequently, two training variants seem reasonable. First, the supervised VSE can be used to make the strategy decisions for all drivers except the driver controlled by the reinforcement VSE in the particular race. In this variant, the driver for whom the reinforcement VSE makes the decisions is randomly selected before each simulated race. Second, a multi-agent simulation is conceivable in which the reinforcement VSE makes the strategy decisions simultaneously for all drivers in the race. The decisions for each driver are made based on a central deep q-network that also incorporates the experience of all other drivers. The studies conducted by Thomaser [176] show comparable results for both variants.

Results

As indicated in the outlook of the paper [13], further investigations showed that the reinforcement VSE achieves an equivalent or better average result position in the simulated races considered than the BSO or the supervised VSE. Table 4.2 shows a comparison between BSO, supervised

VSE, and reinforcement VSE for three exemplary selected races of the 2019 season. For each of the three races, 10 000 simulation runs are performed for BSO, supervised VSE, and reinforcement VSE, resulting in a total of 90 000 simulation runs. At the beginning of each simulation run, one of the 20 participating drivers is randomly selected to be controlled by BSO, supervised VSE, or reinforcement VSE, depending on which variant is currently being evaluated. The strategies of the other 19 drivers in a race are always determined by the supervised VSE. After the 10 000 simulation runs per strategy source and race, the average result position of the controlled drivers is calculated. The lower this value, the better the controlled drivers, and thus the respective strategy source performed on average. [176, p. 45]

Table 4.2: Comparison of BSO, supervised VSE, and reinforcement VSE based on the average result position achieved with randomly selected drivers after 10 000 simulated races. Probabilistic effects, including accidents and failures, are enabled. The results are taken from Thomaser [176, p. 45].

Strategy source	2019 Belgian GP	2019 Singapore GP	2019 Chinese GP
BSO	9.3	8.9	8.2
Supervised VSE	9.5	9.5	9.5
Reinforcement VSE	8.9	8.2	8.2

With a random selection from 20 participating drivers, given the same strategy source and no retirements, an average result position \bar{p} of 10.5 is to be expected, as determined by the following calculation:

$$\bar{p} = \frac{\sum_{i=1}^{20} i}{20} = 10.5. \quad (4.11)$$

Table 4.2 shows that the randomly selected drivers controlled by the supervised VSE reach an average result position of 9.5 in all three races. This deviation is explained by drivers retiring from the races due to accidents and failures. Comparing BSO and supervised VSE with the reinforcement VSE shows that the latter performs on average equal or better in all three races. Compared to the BSO, the main advantage of the reinforcement VSE is that it can react to the individual race situation and thus exploit strategic opportunities. Compared to the supervised VSE, its advantage lies primarily in the specific training for the respective race directly in the simulation, whereby, for example, the tire degradation model parameterization is learned implicitly [176, p. 45]. It was also observed that the reinforcement VSE, unlike the supervised VSE, can learn to make use of advanced strategic opportunities. It can make active use of the undercut, for example, if it helps to gain a positional advantage in the race [176, p. 57ff.]. The reinforcement VSE thus offers the possibility to come close to an optimal race strategy taking into account the individual race situation. However, the transferability to reality strongly depends on how well the parameterization of the simulation fits it.

4.6 Case Study

In the following, the interaction of the individual simulation tools is presented using a case study of the 2019 Chinese Grand Prix in Shanghai. This race is suitable for the case study for several reasons. First, it is held on a race track and not on a city circuit, so it is available in the race track database. Second, a parameter set is available for the LTS, as the required onboard video recording of a qualifying lap with overlaid telemetry data is available for this race. Third, the race took place under normal weather conditions and was not subject to any disturbances except for

a short VSC phase in the beginning. The computing times given in this chapter were determined on an Apple MacBook Pro 2019 (Intel Core i7 9750H, 16 GB RAM) with Python 3.8.

4.6.1 Racing Line Generation

As one of the parameters, the vehicle width must be inserted into the optimization problem. It is important to know that race tracks mostly have curbs installed on the inside and exit sides of corners that allow drivers to exceed the track limits. If this run-off zone is wide enough, the drivers can exceed the track boundary by up to one vehicle width within the scope of the regulations [6, art. 27.3]. This is not included in the race track models. As an approximation, the generation of the racing line is therefore done with a virtual vehicle width of 1.5 m, although F1 cars are 2 m wide. Thus, the racing line is planned closer to the track boundaries in the corners, with the car exceeding them up to 25 cm.

Including all pre-processing and post-processing, the racing line for the Shanghai International Circuit with a center line length of 5451 m is generated within 30 s. The discretization step size along the race track was set to 5 m. The resulting racing line is displayed in Figure 4.4. In addition, Figure 4.5 shows the first corners of Figure 4.4 in more detail. As expected, the racing line changes from the right to the left side of the track between corners two and three. The interesting point about this corner combination is that the calculated racing line does not go back to the inner track boundary in corner number four when accelerating out of corner three. However, this is in line with Hamilton's real driving line in the onboard video recording [179].

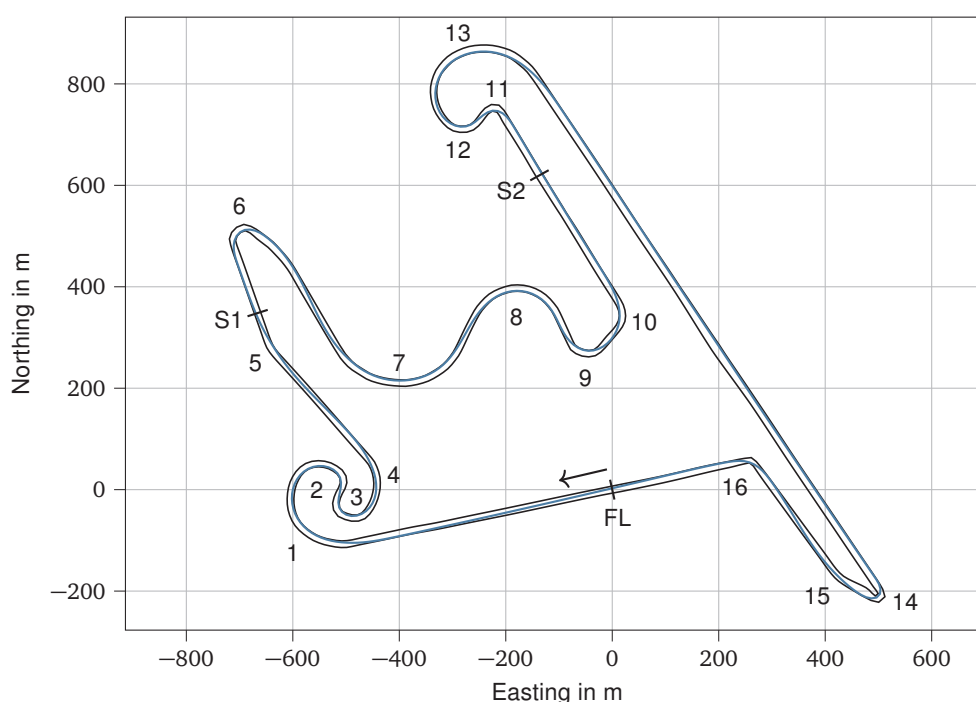


Figure 4.4: The racing line on the Shanghai International Circuit as generated by the (iterative) minimum-curvature approach. The numbers from one to 16 indicate the corners. S1 – Sector 1, S2 – Sector 2, FL – Finish line.

A comparison between the generated minimum-curvature line and a minimum-time line was carried out in [86] for an electric, LMP3-based race car on the FE Berlin race track. It showed that the minimum-curvature line is reasonably close to the minimum-time line. This is especially true for curved sections of the track. However, in sections where tire potential is not fully exploited

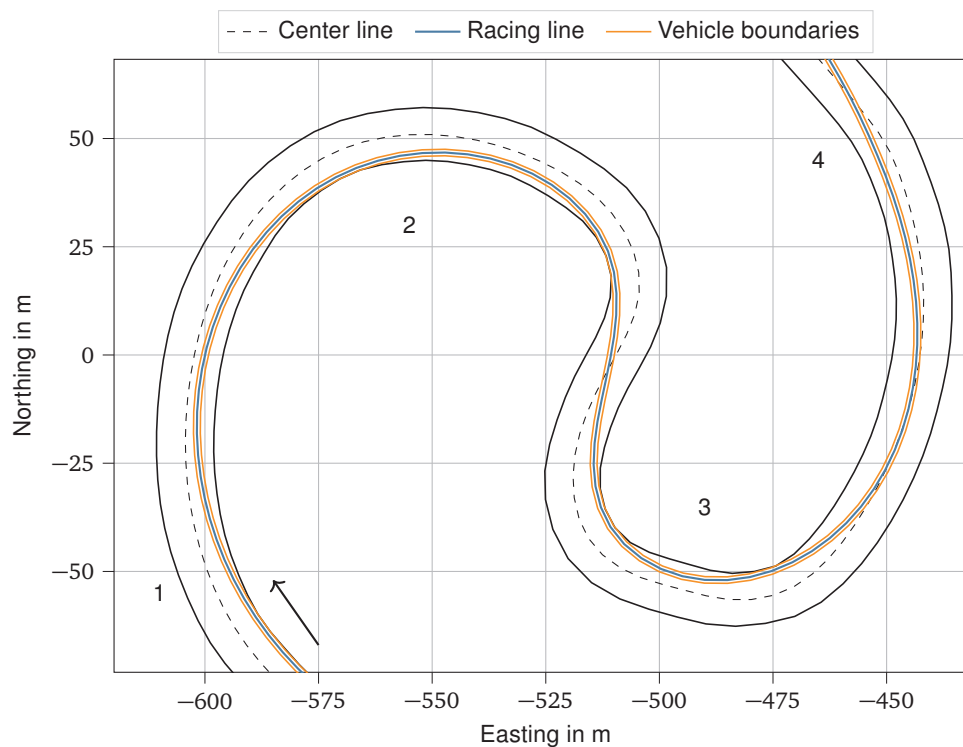


Figure 4.5: Detailed view of the racing line in the corners one to four on the Shanghai International Circuit. The traces of the vehicle width are displayed, which are considered in the optimization problem.

and acceleration is limited by the powertrain, such as accelerating on relatively straight sections, driving on a shorter line is faster than driving on the minimum-curvature line. A similar comparison is not possible for a F1 car, as the available minimum-time optimization can only deal with cars that do not require gear changes. Furthermore, some of the required parameters are unknown for F1 cars. However, it is assumed that the minimum-curvature line is also a reasonable approximation for F1 since the large driving power means that the racing line is determined in large areas by the tire potential. The comparison of the calculated line with Hamilton's driving line in the onboard video recording [179] supports this assumption.

Figure 4.6 contains a comparison of the curvature profiles of the center line and the final racing line. The curvilinear distance was normalized for this plot because the length of the center line differs slightly from that of the racing line. Compared to the input, the curvature profile of the racing line is much smoother, making it suitable for real-world driving. This is one of the biggest advantages of the approach, which inherently creates a smooth curvature profile due to the optimization objective. The calculated curvature profile serves as input for the subsequent LTS.

4.6.2 Lap Time Simulation

Following the racing line generation, the LTS is executed. In the paper on the LTS [9], a vehicle parameter set is used that was created based on the onboard video recording from Hamilton's qualifying lap in the 2017 Chinese Grand Prix [179]. Even though this case study addresses the 2019 Chinese Grand Prix, this parameter set is used as a basis as no more recent video with overlaid telemetry data is available. This decision is supported by the fact that the fastest qualifying lap time in 2019 (91.547 s) is almost the same as in 2017 (91.678 s). Therefore, using a similar vehicle parameter set is sufficient for this case study, even though some parameters,

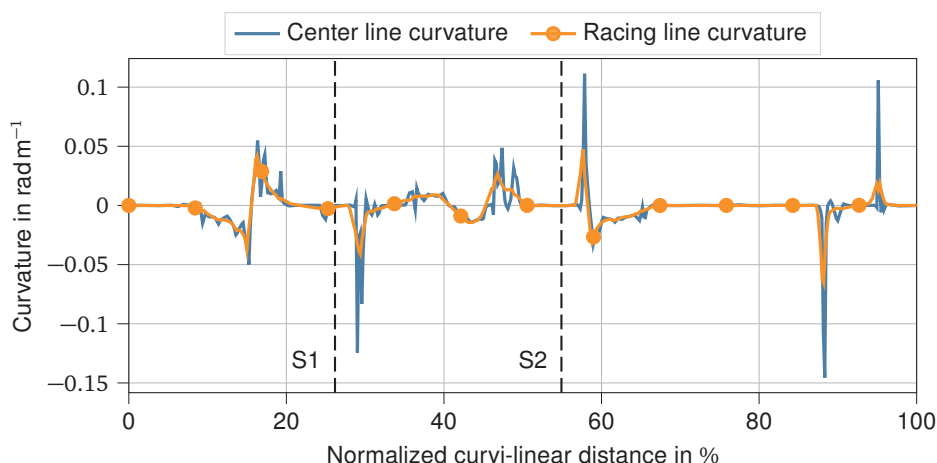


Figure 4.6: Comparison of the curvature profiles of the (already smoothed) center line and the generated racing line. S1 – Sector 1, S2 – Sector 2.

such as engine power, have certainly changed over the seasons due to development progress. One full simulation run of the LTS takes less than 1 s.

Simulation Results

A comparison between the sector and lap times Hamilton achieved in reality and the simulation is given in Table 4.3. The simulation results differ slightly compared to the paper [9]. This is mainly due to the new racing line, which is now based on the race track database containing varying track widths. In addition, as stated in Section 4.3.2, the calculation of the transmissible tire forces in the LTS has been revised compared to its state in the publication. As a result, some parameters also had to be slightly adjusted.

Table 4.3: Comparison between Hamilton's real sector and lap times (qualifying of the 2017 Chinese Grand Prix) and the simulation results on the Shanghai International Circuit.

	Timing data	Simulation	Deviation
Sector 1	24.036 s	24.418 s	0.382 s
Sector 2	27.079 s	27.084 s	0.005 s
Sector 3	40.563 s	40.640 s	0.077 s
Lap time	91.678 s	92.142 s	0.464 s

Especially the first sector time differs comparatively strongly between the real-world data and the simulation result. This can be due to inaccurately positioned sector boundaries, for example, which have to be estimated for the simulation based on the track map. However, since the other two sector times fit better, the cause seems to be rather in the calculated velocity profile. Figure 4.7 shows the simulated velocity profile of Hamilton in comparison to the real profile extracted from an onboard video recording [179] by optical character recognition. Since the curvilinear distance is not included in the video, it is calculated from the displayed speed and the video's frame rate. This inevitably results in a deviation of the curvilinear distance, which is removed from the plot by normalizing to the respective total length.

The general shape of the two profiles agrees well, and the minimum and maximum speeds in the various corners are close together. However, there are also some spots with deviations. In reality, Hamilton starts the lap with a higher speed of 287 km h^{-1} compared to 270 km h^{-1} in the LTS, which builds up a time advantage over the first 10% of the track and thus explains the comparatively large deviation in the first sector time. The reason for the speed difference

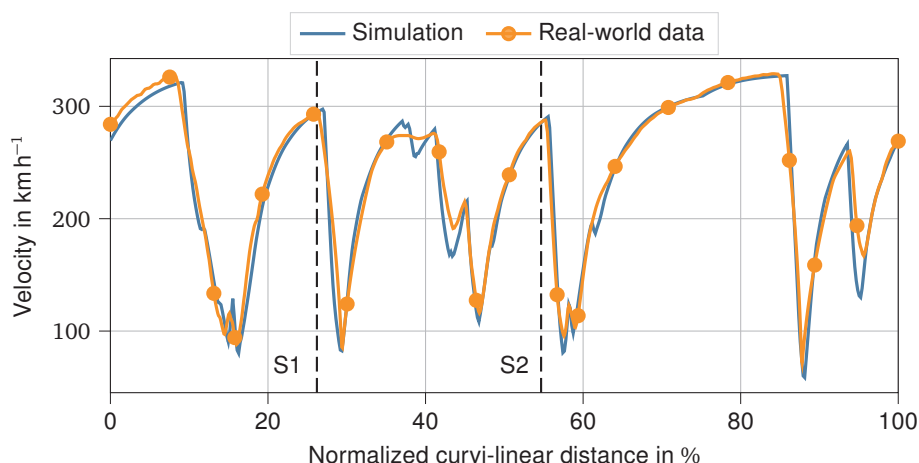


Figure 4.7: Comparison of Hamilton's real velocity profile (from the qualifying for the 2017 Chinese Grand Prix) and the velocity profile simulated with the lap time simulation on the Shanghai International Circuit. S1 – Sector 1, S2 – Sector 2.

cannot be determined from the available data, as the speeds match well at the end of the lap. Hamilton likely got through the last corner of the track a little better at the beginning of the lap than at the end. As the lap progresses, there is a significant difference in the corner combination between 35 % and 40 %, through which Hamilton drives at an almost constant speed in the real world. Differences in the racing line and the tire model (due to the high speeds, especially at high wheel loads) can be considered the cause. The significantly lower simulated speeds in the corners at 45 % and 95 % are primarily due to deviations in the race track and racing line, respectively. Especially in the final corner, the onboard video recording [179] shows that Hamilton makes heavy use of the curbs to increase the radius. When braking at high speeds, it is also noticeable that the simulation has a slightly later brake point. This speaks for deviations in the drag coefficient or the achievable braking forces. All in all, the deviations are within the expected range, taking into account the uncertainties in the parameterization and the racing line that are unavoidable due to the data situation.

Parameter Determination for the Race Simulation

Two different configurations of the car, one for the qualifying and one for the race, are required in the LTS to determine parameters for the RS. In the qualifying configuration, full engine power is available, the electric motor in the hybrid powertrain is used wherever possible, and DRS is activated in all zones. Slightly less power is available in the race configuration due to limited fuel mass and engine durability. The power disadvantage is assumed to be 5 % (only for the combustion engine) for this case study, as the exact value is unknown. In addition, the energy for the electric motor is limited such that the battery's state of charge at the end of the lap is the same as at the beginning. The hybrid strategy is set to *longest time to braking point*, in which the energy is distributed as far away as possible from the braking points until it is used up. This strategy is chosen because the use of the available energy is most efficient in terms of lap time at low speeds and far away from the next braking point [9]. DRS is activated or deactivated depending on the simulation objective. As a side note on the different engine settings in qualifying and race, it should be mentioned that the FIA has issued a rule in the 2020 season that the same engine settings must be used in both qualifying and race, so the difference will no longer apply in the future [180]. Table 4.4 contains the parameters that can currently be estimated using the LTS.

Table 4.4: RS parameters that can be estimated using the LTS. The example values are determined for the 2019 Chinese Grand Prix. The values in the *timing data* column were determined according to the procedures described in Section 4.4.2.

Description	LTS configuration	RS parameter	Timing data	Simulation
Qualifying lap time	Qualifying (DRS active)	t_Q	91.678 s	92.142 s
Race pace time delta	Race (DRS inactive)	t_{racepace}	1.812 s	1.360 s
Mass sensitivity of the lap time	Race (DRS inactive)	S_{mass}	0.031 s kg^{-1}	0.053 s kg^{-1}
Fuel consumption per lap	Race (DRS inactive)	B_{fuel}	$1.964 \text{ kg lap}^{-1}$	$1.851 \text{ kg lap}^{-1}$
Time gain due to DRS	Race (DRS active)	t_{drs}	0.553 s	0.416 s

The difference between the simulated and real-world qualifying lap times t_Q was already discussed before. The time difference between qualifying and race configuration t_{racepace} is smaller in the simulation than it was determined based on timing data. On the one hand, this may be due to effects that cannot be taken into account in the LTS, e.g., a lower track temperature on the race day. On the other hand, the value determined from timing data is also subject to uncertainties. However, the exact value is not decisive because the parameter is identical for all drivers and thus only influences the race duration and not the course of the simulated race. S_{mass} is significantly larger in the simulation than when it is determined based on timing data. The discrepancy may result from the fact that the value determined from timing data is based on the median of all available sensitivities of the race (according to Section 4.4.2), which is not a perfect fit for each driver. Another possible explanation is that the driver brakes and accelerates less in reality than in the simulation, so the vehicle mass has less effect on the lap time. The velocity profile in Figure 4.7 supports this hypothesis. Especially in the corners at 35 %, 45 %, and 92 %, the minimum speeds of the real-world profile are higher. In addition, short braking phases can be seen in the simulation in some acceleration phases, for example, at 62 %. The fuel consumption B_{fuel} as well as the time advantage due to DRS t_{drs} fit well to the values determined based on timing data. Overall, it is important to note that both the determination of parameters via LTS and from timing data are subject to uncertainties due to the data situation. If used within a team with knowledge of the exact parameters, significantly more accurate results are to be expected for both methods.

With a model for the clutch engagement process and appropriate parameterization of the powertrain at low engine speeds, the parameters representing the increase of the first lap time due to grid position $t_{\text{p,grid}}$ and due to starting from a standstill $t_{\text{standstill}}$ could also be estimated. The same applies to the time losses when driving through the pit lane, $t_{\text{pitdrive,inlap}}$ and $t_{\text{pitdrive,outlap}}$, if the correct curvature profile of the pit lane was available.

4.6.3 Deterministic Race Simulation

Even though the post-simulation of an actual race is not the primary use case of the RS, its functionality and the effects of different parameterizations can be well illustrated in a comparison between simulation results and real-world data. Therefore, for this case study, two variants are analyzed to explain the intended usage of the RS:

- Automatically generated parameter set based on timing data (according to Section 4.4.2)
- Manually adjusted parameter set (starting from the first variant)

For clarity, only the six drivers of the three teams dominating in 2019 – Ferrari, Mercedes, and Red Bull – who also finished the real race in the top six positions are simulated for the case study.

Random influences in the simulation are deactivated in this first step to obtaining reproducible results. However, the VSC phase of the real race is simulated, as otherwise, a fair comparison is hardly possible. The computing time for each simulation run is about 30 ms (which scales almost linearly to about 115 ms with 20 instead of six drivers).

Visualizing race time gaps is particularly suitable for graphical analysis of the course of a race. Knowing the race duration $t_{\text{race,tot}}$ (i.e., the final race time of the race winner) and the total number of laps l_{tot} , the race time gap Δt_{race} in lap l is calculated as stated by

$$\Delta t_{\text{race}}(l) = t_{\text{race}}(l) - \frac{t_{\text{race,tot}}}{l_{\text{tot}}} \cdot l. \quad (4.12)$$

Δt_{race} compares the race time of a driver in lap l to that of a virtual driver with a constant lap time, chosen as the average lap time of the race winner. The resulting plot allows to quickly determine phases where a driver gains (negative slope) or loses (positive slope) time against the virtual driver. Furthermore, the temporal distances between the drivers and, thus, their positions are visible, and pit stops are identifiable by a sudden rise. Figure 4.8 shows how the actual and simulated race time gaps for the six drivers evolve over the 2019 Chinese Grand Prix.

Course of the Real Race

The top plot in Figure 4.8 shows the race times gaps of the real-world race. The yellow bar indicates a VSC phase, which was active from the end of the first lap until the middle of the second lap. This is why the gaps of all drivers rise strongly in the first two laps. Apart from the change of position on the first lap (Table 4.5, barely visible in the plot), Hamilton's and Bottas's races are largely uneventful. Hamilton wins the race after 5526.35 s. Vettel loses position three to Leclerc on lap one (Table 4.5) but can regain it on lap eleven. It should be noted that this change of position happened due to a team order. The plot shows that Vettel can hardly gain ground in the laps after that, which is why he would have been too slow for a successful overtaking maneuver on his own. He pits on lap 18, one lap after Verstappen, narrowly defending his position against Verstappen's undercut attempt. As the race progresses, the gap between the two drivers widens again so that they finish in positions three and four. Leclerc delays his first pit stop until lap 22 and thus loses much time on the old tires compared to Vettel and Verstappen. Consequently, he falls behind both of them after the stop. The advantage of the fresher tires at the end of the race is visible in the plot but is not enough to win back a position. Gasly is significantly slower than the other five drivers, so there is hardly any interaction. An interesting aspect is brought up by his last stop shortly before the end of the race, where Gasly changes back to fresh tires with the softest compound A6. This enables him to set the fastest race lap with little fuel mass and score an additional championship point. This lap can be recognized by the sharp drop in his race time gap on the penultimate lap. The maneuver is only worthwhile if, as in this case, there is no loss of position to fear due to the pit stop.

Table 4.5: Actual and simulated rank positions in different states of the 2019 Chinese Grand Prix. Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.

	HAM	BOT	VET	VER	LEC	GAS
Grid position (real-world)	2	1	3	5	4	6
Position after first lap (real-world)	1	2	4	5	3	6
Final position (real-world)	1	2	3	4	5	6
Final position (simulation V1)	2	1	4	3	5	6
Final position (simulation V2)	1	2	3	4	5	6

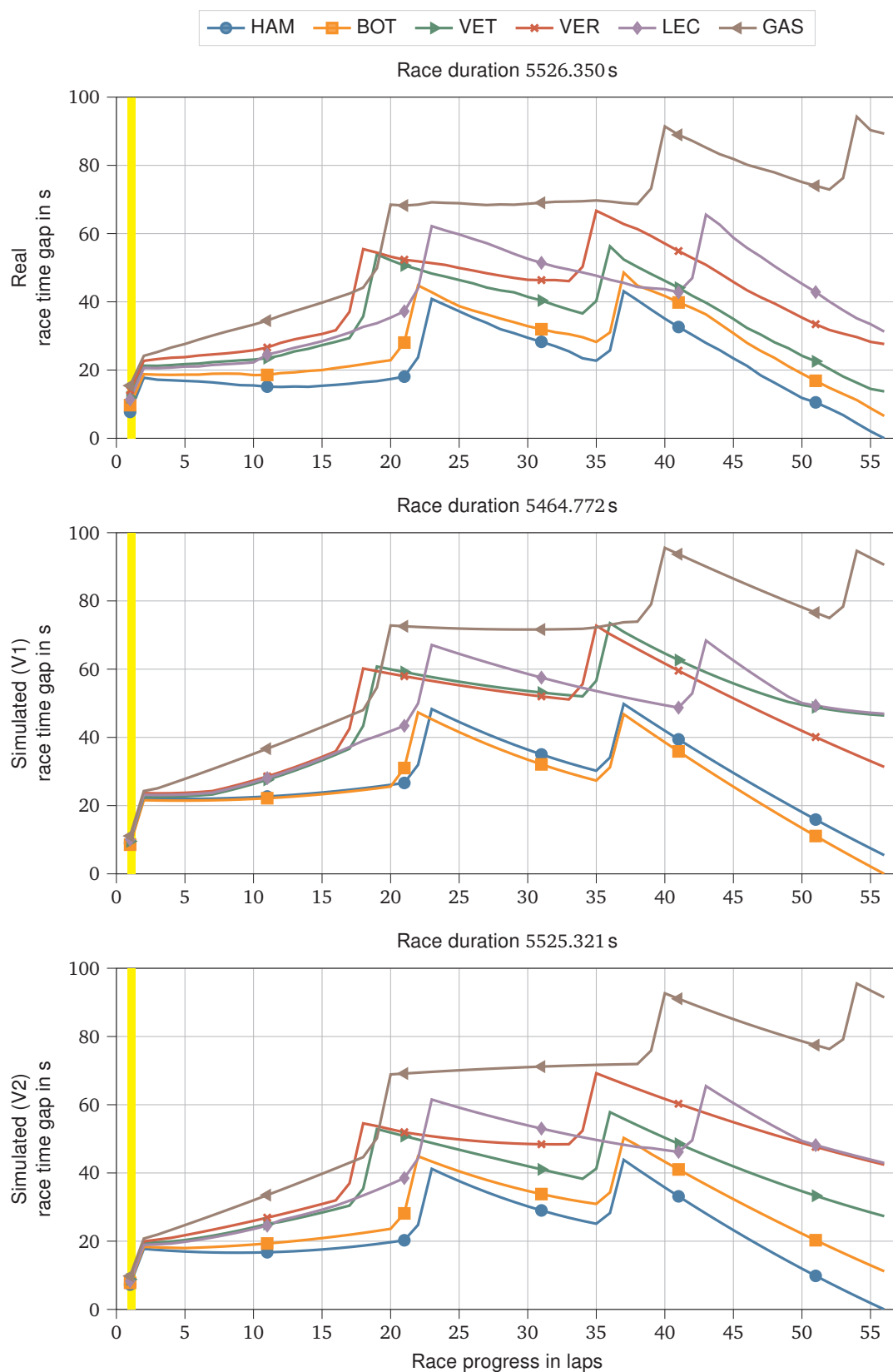


Figure 4.8: Real (first chart) and simulated (other charts) race time gaps to a virtual driver with a constant lap time for the 2019 Chinese Grand Prix. A virtual safety car phase was active from the end of the first lap until the middle of the second lap (yellow bar). Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.

Course of the Simulated Race (Variant 1)

Before proceeding with the evaluation, it should be noted that the reference lap time is always adjusted to the race duration of the winner of the corresponding simulation run to create the plots. This facilitates comparing the race time gaps between the different variants even if the race durations do not match. Differences in this value are acceptable as they can be easily adjusted via the race pace time delta t_{racepace} . As mentioned in Section 4.4.2, the value represents many different influences that can hardly be determined, which means that it cannot be expected to be determined too accurately in the automated fitting process. For variant 1, this results in the fact that the simulated race duration is about 62 s too fast, which corresponds to a required increase of t_{racepace} of slightly more than 1 s.

For the parameterization according to variant 1, the simulated race course generally agrees well with the real race. Starting with the two Mercedes drivers, Hamilton and Bottas, three deviations stand out. The first deviation is that Bottas finishes ahead of Hamilton. This is because they do not swap positions directly on the first lap, as was the case in the actual race. There are several reasons for this. First, due to the lap-wise discretization of the simulation, the first lap (like any other) can only be considered as a whole. Since the last 20 % of the lap is already driven under FCY conditions with a prescribed minimum lap time, Hamilton's possible time advantage over Bottas is reduced, making overtaking more difficult. Second, the start phase of a race is strongly influenced by chance, as all the cars are close together. The drivers lining up in the first few corners after the race start and the resulting distances between them can therefore not be simulated realistically in a deterministic simulation. The second deviation between reality and simulation can be seen in Hamilton's and Bottas's first stint (the same happens in the first stint of Vettel, Leclerc, and Verstappen). If both drivers are similarly fast, the simulated DRS effect causes them to stick closely together without the attacker being able to overtake the driver ahead. In reality, after a few unsuccessful overtaking attempts, the attacker would often drop back a bit to get out of turbulent air and save tires. He can then adjust his race strategy by, for example, attempting an undercut, or he can hope that the tires of the car ahead will degrade faster so that an overtaking maneuver becomes possible after a few laps. Two drivers sticking together usually does not affect the overall race in the simulation too much. The main effect is that the affected drivers each receive a time penalty t_{duel} of 0.3 s in the corresponding laps, which represents that they are in a duel and no longer follow the time-optimal racing line. As with the previous deviation, this one occurs less frequently and for shorter periods if probabilistic effects are taken into account in the simulation. This is because the lap times also vary slightly in that case, which means that the attacker can either overtake sooner or later (if he is faster) or falls out of the DRS window and then falls behind (if he is slower). The third deviation is that both drivers show identical degradation behavior in the second stint, although Bottas' lap times increase more in reality (evident from the stronger positive curvature of the curve). This happens because the tire models are parameterized together for a team's drivers to have enough data points, as was described in Section 4.4.2. Thus, there are inevitably deviations for the individual drivers in direct comparison to reality.

Comparing the race progressions of the two Ferrari drivers, Vettel and Leclerc, it turns out that their first stint is too slow in the simulation. In addition to the tire parameters, this is also due to t_{duel} , which has been described already. Another noticeable aspect of the first stint is that Vettel and Leclerc, like the two Mercedes drivers, do not swap positions on the first lap. Consequently, Vettel's overtaking maneuver back to the third position on lap eleven does also not occur in the simulation. As soon as Leclerc drives freely (no interaction with other drivers), his race course matches reality well. The picture is different for Vettel. Since Red Bull driver Verstappen

sticks directly to the two Ferrari drivers in the first stint, his undercut attempt is successful in the simulation, unlike in reality, as a result of which he narrowly overtakes Vettel at his first pit stop. Vettel then gets stuck behind Verstappen in his second stint, which changes his subsequent race course the most compared to reality. In addition, Vettel's third stint indicates that his tire degradation is parameterized too high (visible in the strongly positive curvature). There are two reasons for this. First, the parameterization for both team drivers is done together, as explained before. Second, a single degradation model is parameterized for each tire compound and driver. If the same compound, as in this example A4, is driven once in the first and the third stint and shows different degradation behavior, this cannot be considered. The causes for such behavior can be manifold and are, therefore, difficult to approximate. For example, there is certainly a correlation between vehicle mass and degradation rate so that the tires degrade more slowly towards the end of a race due to lower fuel mass. However, it was decided that modeling at such a level of detail is not reasonable as the data situation would not allow for a meaningful parameterization.

The race course of the two Red Bull drivers, Verstappen and Gasly, fits well with reality. Verstappen's first stint is a bit too slow because he runs into Vettel and Leclerc, Gasly's because the first tire compound is parameterized slightly too slowly.

Course of the Simulated Race (Variant 2)

Manual adjustment involves changing the parameters based on graphical analyses of the race time gap diagrams so that the result of the RS matches the real-world race as closely as possible. The procedure starts with parameters that affect the race course of all drivers (e.g., the mass sensitivity S_{mass}) and then gets more detailed step by step (car performances t_{car} , driver performances t_{driver} , tire model parameters). Finally, the race pace time delta t_{racepace} is used to adjust the race duration.

The mass sensitivity S_{mass} from variant 1 fits well in the present case and does not need to be adjusted. Since the vehicle mass continuously decreases in the course of the race, a wrong sensitivity is recognizable by wrong slopes at the beginning of the race, whereas they fit towards the end of the race. Since the two position changes between Hamilton and Bottas as well as between Vettel and Leclerc in lap one do not occur in the simulation for the reasons already mentioned, their starting positions for variant 2 are adjusted so that they correspond to the positions at the end of the first lap (Table 4.5). For this case study, the further course of the race (and thus the validity of the deterministic RS) can therefore be evaluated without this influence. The time loss on entering the pit lane is reduced by 1 s to fit reality. Subsequently, the vehicle and driver skills, as well as the tire parameters, are adjusted. Next, a team order is created for the two Ferrari drivers at the level of the overtaking threshold $t_{\text{gap,overtake}}$ so that Vettel can overtake Leclerc as soon as he is slightly faster, as in reality. Finally, t_{racepace} is increased so that the race duration matches the real one better.

A comparison of the plots shows that the simulation result of variant 2 largely agrees with the real race. The only visible deviation is a slight positive curvature in the final stints of Vettel, Verstappen, and Bottas. This is due to the tire degradation being parameterized too high. However, since all three drivers drove the first and third stints on the A4 compound, adjusting the degradation to fit the third stints inevitably worsens the agreement for the first stints, as mentioned before.

Variant 2 shows that the RS can reproduce reality well with appropriate parameterization. For suitable parameterization before a race, upstream simulation tools, e.g., the LTS, meaningful data from free practice and qualifying sessions, as well as experience values from past races

and seasons, are required. After a race, the developed automated parameter determination procedure quickly provides a valuable parameter set based on the race's available timing data (as shown as variant 1). If higher simulation accuracy is required and additional time for parameter tuning is available, the user can further adapt the obtained parameter set (as shown as variant 2).

4.6.4 Probabilistic Race Simulation

As indicated in the previous section, a deterministic RS does not take into account that probabilistic effects strongly influence the actual race, e.g., through position gains and losses in the starting phase. Thus, the deterministic RS represents only one manifestation from the possible spectrum of results. Probabilistic effects on the race must be considered to obtain information about this spectrum and, thus, the robustness of a strategy. The joint evaluation of all influences on the result is carried out by Monte Carlo simulations.

For this case study, 10 000 races are simulated with probabilistic effects activated. These include the variation of lap times and pit stop durations and the consideration of the starting performances of the drivers. Accidents and retirements, as well as the resulting FCY phases, are not artificially generated in this section because the automated adaption of race strategies to the dynamic race situation will follow in Section 4.6.5. Consequently, the VSC phase from the real-world race remains as before. The deterministic part of the RS is parameterized using the manually adjusted parameter set (variant 2). The resulting final position distributions are shown in Figure 4.9. The computing time for the 10 000 simulation runs is about 75 s. With 20 instead of six drivers, this increases to about 285 s.

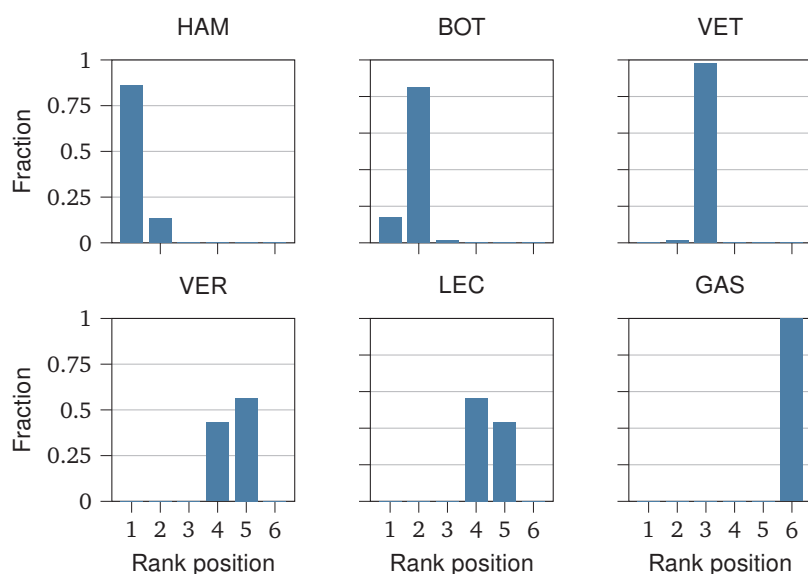


Figure 4.9: Distributions of final rank positions after 10 000 simulation runs of the 2019 Chinese Grand Prix with activated probabilistic effects and without adjustments of the race strategies. Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.

The results of Hamilton, Bottas, Vettel, and Gasly agree well with the deterministic result of variant 2 in Figure 4.8. However, this is different for Verstappen and Leclerc. In the deterministic simulation, Verstappen finishes the race ahead of Leclerc, which is the case in only 43 % of the races when probabilistic effects are taken into account. This can be explained by Figure 4.8, which shows that Leclerc catches up with Verstappen shortly before the end of the race, who

loses time due to degrading tires. Thus, a slight, random advantage in one of the last laps is enough for Leclerc to overtake him.

4.6.5 Race Strategy Evaluation

In this section, it becomes clear that the random generation of failures and accidents and the simulation of the FCY phases triggered by them are also important parts of the RS. This is because they require each driver to react to the particular race situation to achieve a good result. This allows to train the automated making of race strategy decisions within the RS (i.e., the reinforcement VSE) and to evaluate the expected average result position of different decision sources against each other under realistic conditions.

Phases of a Race Weekend

The paper [13] presents three options for determining a race participant's race strategy by software: Basic Strategy Optimization (BSO), supervised VSE, and reinforcement VSE. BSO optimizes the stint lengths and the tire compounds selected, assuming a race with no opponents. Supervised VSE and reinforcement VSE are based on machine learning methods, with the supervised VSE being trained on real-world timing data and the reinforcement VSE being trained in simulation. Depending on the particular phase of a race weekend, not all options are always suitable for making race strategy decisions. In this context, it should be kept in mind that the supervised VSE is the only one of the options that makes its race strategy decisions independently of the parameterization of the RS. This can be an advantage or disadvantage, depending on the parameter accuracy. Table 4.6 gives an overview on the suitability of the three options in dependence on the phase of a race weekend. The ratings are justified in the following paragraphs.

Table 4.6: Overview of the suitability of the three available options for determining the race strategy by software in dependence on the particular phase of the race weekend. The more the circle is filled with black, the more suitable an option is.

	BSO	Supervised VSE	Reinforcement VSE
Before the race	●	◐	◐
During the race	◐	●	●
After the race	◐	○	●

Before the race, the main interest of a strategy engineer is to prepare a few basic strategies, one of which will be followed if the race proceeds normally. The main tool for these considerations is the BSO. Neglecting driver interactions and race events, the BSO can be used to determine the strategy with the fastest race duration. It can be assumed that at least two variants (e.g., a 1-stop and a 2-stop strategy) are prepared in reality to be able to react to the direct competitors in the race with a change of strategy if necessary. For example, one could switch to a 1-stop strategy if the competitor pursues a 2-stop strategy and one expects that one cannot compete with him when pursuing the same strategy. The expected result positions for the prepared strategy variants can be determined along with an estimate of their robustness using the RS, as described in sections 4.6.3 and 4.6.4. Since the parameterization of the simulation models will never be perfect before the race, it is furthermore useful to determine pit stop windows for each strategy variant. By slightly varying the tire degradation model parameters, the optimization results change, allowing the width of the pit stop windows to be estimated. Since there is a high probability of FCY phases, especially in the early course of the race [12], it can be assumed that

variants must be prepared for this case as well. The relevant question in this context is at what stage of the race an early pit stop is worthwhile or whether an additional stop makes sense. This can be investigated, for example, by manually generating FCY phases in the RS. It can then be used to check how supervised VSE and reinforcement VSE would react.

During the race, the RS supports the user if the race develops differently than expected. For example, if the tire degradation deviates strongly from the original parameterization, the BSO can be used to recalculate the optimal pit stop windows. In case of sudden FCY phases, supervised VSE and reinforcement VSE can be used to support the decision for or against a pit stop. In addition, the reinforcement VSE supports the user in making use of advanced strategic opportunities, such as the undercut. In all these cases, the RS is used to predict the further course of the race and the expected result based on the current race state and the strategy decisions under consideration.

After a race, the race events that occurred are known, and the parameterization of the RS can be refined. On this basis, the strategy followed throughout the race can be questioned, i.e., whether a different strategy would have led to a better result. However, such a post-simulation is still subject to uncertainties, so the results should be treated cautiously. Differences in the output strategies compared to output before and during the race are only to be expected for BSO and reinforcement VSE, since their results depend on the parameterization, in contrast to the supervised VSE. Its output during the race is identical to that after the race. Since the parameters of the RS can be well adjusted after a race, the focus of this phase is on the reinforcement VSE, as it also takes into account the interactions between the drivers.

For this case study, some of the above considerations are elaborated with examples in the following paragraphs. Leclerc is chosen as the driver for these examples because his race course in Figure 4.8 and his distribution of final rank positions in Figure 4.9 indicate that there is a good chance to improve his result position by applying another race strategy.

Before continuing, it should be mentioned that two variants of the RS are distinguished in the following:

- Regular RS
- Simplified RS

For the simulation of entire races, the RS is used in its regular form. For some applications, however, it makes sense to use a condensed form, which will be referred to as *simplified RS* in the following. Only a single driver is simulated in this form, i.e., interactions with other drivers are neglected. This variant is also the basis for the BSO, as presented in [13].

Phase 1: Before the Race

Preparations To better replicate the situation before the race, the slightly less accurate, automatically determined parameterization from Section 4.6.3 (variant 1) is used, even though it was created based on data from the actual race. As already mentioned, the data situation from an external point of view is not sufficient to allow a parameterization based on data from before the race. However, this does not affect the following steps.

In preparation for the later evaluations, the reinforcement VSE must first be trained on these parameters in the RS, as presented in Section 4.5.2. For this purpose, 250 000 laps (corresponding to almost 4500 races) of the Chinese Grand Prix are simulated. The training is done with all 20 drivers in order not to falsify the probabilities of failures and accidents in combination with the

FCY phases. For each race, the driver for whom the reinforcement VSE selects the strategy is randomly selected. The strategies of the other drivers in the simulated races are determined by the supervised VSE so that they react to the respective race situations. The training takes about 50 min, as only a single core of the CPU is currently used for most parts of the training. Therefore, training time could be significantly reduced in the future by parallelizing the program or using a GPU.

Determination of a Basic Strategy As introduced before, determining a basic strategy is the first step for further analyses. The start tire compound for the optimization is fixed to A4 because Leclerc qualified within the top ten. Executing the BSO with Leclerc’s parameters returns the optimal stint lengths (for a race without opponents) for all possible 1-stop, 2-stop, and 3-stop strategies. The computing time for the optimization is about 15 ms. Table 4.7 shows the strategies that lead to the fastest race duration for the given number of pit stops. The simulated race durations $t_{\text{race,tot}}$ were determined in the simplified RS since this is the underlying assumption for the BSO.

Table 4.7: Comparison between Leclerc’s best 1-stop, 2-stop, and 3-stop strategies as determined by the BSO for the 2019 Chinese Grand Prix.

Strategy	Start compound	First stop	Second stop	Third stop	$t_{\text{race,tot}}$	$\Delta t_{\text{race,tot}}$
BSO 1-stop	A4	in-lap 17, A6	–	–	5474.382 s	2.674 s
BSO 2-stop	A4	in-lap 14, A3	in-lap 35, A3	–	5471.708 s	–
BSO 3-stop	A4	in-lap 12, A3	in-lap 28, A4	in-lap 42, A4	5476.892 s	5.184 s

Before evaluating the results in the table, it should be noted that the order of the stints (except for the first stint from race start until first pit stop, for which the compound is predefined in this example) can be swapped in the simplified RS without changing the race duration. The sequences shown are, therefore, only one possible variant. In general, however, multi-stop strategies tend to switch to a harder compound in the first stop to retain the option of making one stop less if, for example, tire degradation turns out to be lower than expected. Furthermore, it should be noted that the race durations of the simplified RS are not comparable to the race durations presented later on when all cars are taken into account.

The simulated race durations show that the optimized 2-stop strategy is the fastest with an advantage of 2.674 s. The second-fastest strategy is the 1-stop strategy. It can be recognized that for this strategy, the starting tires should be swapped for an A6 tire set relatively early in the race at the end of lap 17. This does not seem reasonable, as A6 is the softest compound available, but it would have to last a long 39 laps to finish the 56 laps race. The reason for this is an inaccurate parameterization of the A6 compound. The parameter set used for Leclerc is comparatively slow at the start but hardly degrades at all, which is exactly the opposite of what is expected for the softest compound. The wrong parameterization can be explained by the fact that neither of the two Ferrari drivers ran the compound in reality (in fact, among the top six drivers, it was only run by Gasly), which means that it had to be determined based on other drivers during the automated parameter determination, as described in Section 4.4.2. According to the results, the 3-stop strategy is not worthwhile on the track in Shanghai.

Evaluation of the Basic Strategy The regular RS is used to get an idea of Leclerc’s expected result position. For this example, the top six drivers are simulated. Leclerc’s strategy is always set to the fastest previously determined basic strategy, i.e., the 2-stop strategy shown in Table 4.7. For the other five drivers, the strategy can be determined by either the BSO, the supervised

VSE, or the reinforcement VSE. The resulting rank positions and race durations when simulating those three variants for the 2019 Chinese Grand Prix are included in Table 4.8.

Table 4.8: Simulated resulting rank positions p and race durations $t_{\text{race,tot}}$ for Leclerc in the 2019 Chinese Grand Prix. For the probabilistic simulation results, the average values after 10000 simulation runs are given.

Strategy Leclerc	Strategies of other drivers	Deterministic simulation	Probabilistic simulation
BSO	BSO	$p = 4, t_{\text{race,tot}} = 5502.815 \text{ s}$	$\bar{p} = 3.9, \bar{t}_{\text{race,tot}} = 5486.675 \text{ s}$
BSO	Supervised VSE	$p = 3, t_{\text{race,tot}} = 5479.926 \text{ s}$	$\bar{p} = 3.4, \bar{t}_{\text{race,tot}} = 5480.356 \text{ s}$
BSO	Reinforcement VSE	$p = 5, t_{\text{race,tot}} = 5516.228 \text{ s}$	$\bar{p} = 3.7, \bar{t}_{\text{race,tot}} = 5484.117 \text{ s}$

Both the deterministic and the probabilistic simulation results suggest that Leclerc has a good chance of finishing the race in at least the fourth position. Only in the deterministic simulation the use of the reinforcement VSE for the other drivers causes Leclerc to perform comparatively poorly, resulting in the fifth position. However, this is not evident in the corresponding probabilistic simulation, where Leclerc achieves an average position of 3.7. This indicates that the fifth position is an edge case in the deterministic simulation.

In addition to the previous conclusions, Leclerc's results can be used to infer how the various strategy sources used for the other drivers compare to BSO. The worse Leclerc performs, the better the strategies of the other drivers. As expected, Leclerc finishes best when the supervised VSE is used for the other drivers. This is because it is not adapted to the specific race, in contrast to BSO and reinforcement VSE. When BSO is used for all drivers, Leclerc ends up close to his fourth starting position in both simulation variants. Since this is his worst result in the comparison, BSO seems to provide the best strategies for the other drivers in this example. The reinforcement VSE is in between the other two options. Without FCY phases, as in this example, it usually falls slightly behind or is about equal to the BSO, depending on the race and the driver. The real advantages of the reinforcement VSE become clear only when FCY phases are considered.

Determination of the Pit Stop Windows Next, the pit stop windows of the prepared basic strategies can be determined. Various aspects can be included in the window width. One of them is a possibly lower or higher tire degradation, which can, for example, be caused by deviating temperatures on race day. This is examined using the 1-stop strategy from Table 4.7 as an example, since the associated race durations can be plotted in dependence on the possible in-laps in two dimensions, in contrast to multi-stop strategies. For this purpose, two scenarios are simulated, in which the tire degradation of the A4 compound is 15% lower and higher than initially parameterized. Figure 4.10 shows the resulting race durations as a function of the in-lap of the pit stop. The optima could be found with the BSO, as before, but for the plot shown, full-factorial simulation of all possible in-laps is necessary.

It can be seen that the optimal in-lap for the high degradation scenario (+15%) is two laps earlier, and for the low degradation scenario (−15%) two laps later than initially, resulting in a pit stop window from lap 15 to lap 19 in this example. If the degradation of the other compound (A6) were also varied, this would also affect the pit stop window. Such parameter studies can be used to prepare various scenarios for the race.

Another aspect for determining the pit stop window width could be to take into account how many laps before or after the optimum the driver could come into the pits to still be within one or two seconds of the minimum race duration. This would be interesting if, due to the race situation, one wants to come into the pits earlier or later than initially planned, for example, for an undercut. Based on a full-factorial simulation, one could also examine the window width for which it makes

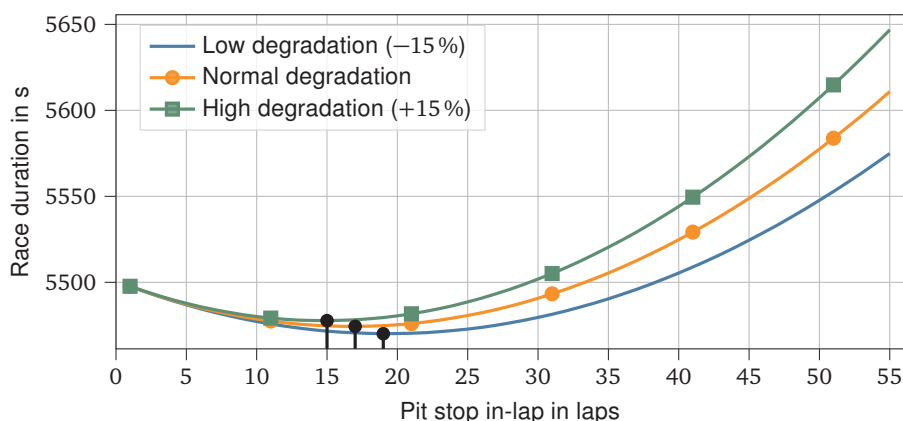


Figure 4.10: Simulated race durations for Leclerc at the 2019 Chinese Grand Prix as a function of the pit stop in-lap for three different 1-stop scenarios. Black dots mark the optimal in-laps of the three scenarios. The race durations were determined in the simplified RS.

sense to come into the pits if a VSC phase gets deployed. Due to the reduced time loss when driving through the pit lane, an early or postponed pit stop under VSC conditions still pays off, although it is no longer optimal if one considers tire degradation alone. For SC phases, similar considerations apply in principle, but many additional aspects must be considered. For example, if the drivers have already queued up behind the SC, a driver will lose many positions during a pit stop if the opponents behind do not pit. Another aspect is related to the tire compound. Since the tires are subjected to less stress than usual during an SC phase, they cool down considerably. Consequently, a driver can be overtaken more easily at the restart after an SC phase if his tire compound is too hard and therefore does not come up to temperature quickly enough to be competitive.

Preparation for Early FCY Phases As an example preparation for FCY phases early in the race, four different phases are simulated, and the responses of supervised VSE and reinforcement VSE are investigated. These responses can be used as a starting point for further considerations. Table 4.9 shows the reactions of supervised VSE and reinforcement VSE for Leclerc in dependence on the race progress range that is covered by the various phases. In this example, the same strategy source was set for the other five drivers as for Leclerc, i.e., supervised VSE for the one and reinforcement VSE for the other column. This is mentioned because the VSE considers the surroundings of each driver, so the results are not independent of which strategies the opponents follow.

Table 4.9: Comparison of the strategies output by supervised VSE and reinforcement VSE for Leclerc in the 2019 Chinese Grand Prix in the simulation without FCY phases and with some early FCY phases.

FCY phases	Supervised VSE	Reinforcement VSE
Without VSC/SC phase	in-lap 18, A3 / in-lap 39, A4	in-lap 23, A3
VSC phase from 2.5 to 4.5 laps	in-lap 19, A3 / in-lap 39, A4	in-lap 26, A3
SC phase from 2.5 to 7.9 laps	in-lap 22, A3 / in-lap 41, A4	in-lap 31, A3 / in-lap 41, A4
VSC phase from 7.5 to 9.5 laps	in-lap 21, A3 / in-lap 40, A4	in-lap 26, A3
SC phase from 7.5 to 12.9 laps	in-lap 24, A3 / in-lap 43, A4	in-lap 9, A4 / in-lap 30, A3

It can be seen from Table 4.9 that the supervised VSE for Shanghai still relies on a 2-stop strategy in all simulated variants. Due to the lower degradation during the FCY phases, the pit stops are only delayed slightly. This fits with the results in [13], where it was shown that

only at the start of phases from about 25 % race progress the strategy is directly adjusted. The reinforcement VSE uses 1-stop strategies for the race without FCY phases as well as for those with VSC phases. It does not react to the early SC phase with an immediate pit stop but switches to a 2-stop strategy. The reason for this switch is not visible, since in this specific case, a position is neither gained nor lost due to the additional stop. In the race with the late SC phase (starting in the eighth lap), the reinforcement VSE takes the opportunity and changes to a fresh set of tires during the phase. The fact that it does not pit directly in the eighth lap but the ninth lap suggests that this SC phase is just at the lower limit of race progress for which the reinforcement VSE dares to make a pit stop with Leclerc. This is also evident from the other drivers. For example, Hamilton pits directly on lap eight, whereas Vettel and Bottas stop later in the race. In conclusion, it can be said that a pit stop is more likely to be avoided with early FCY phases, even in reality. This is because, on the one hand, the tires are still relatively fresh, and, on the other hand, the driver field is still comparatively close together, so that many position losses are to be feared with a pit stop.

Phase 2: During the Race

The situation during a race cannot be realistically represented in this case study. Since there is no access to a stream providing timing data live during a race, a tool would have to be developed that fakes this stream based on the data stored in a lap-wise manner in the timing database. Furthermore, appropriate interfaces to the simulation tools would have to be developed that can process this stream to subsequently enable simulations based on the respective state of the race. Accordingly, the usage of the developed simulation tools during a race is not further elaborated for the case study. A few aspects of the strategy decisions of supervised VSE and reinforcement VSE during races are addressed in [13] and in Section 4.5.2.

Phase 3: After the Race

After the race, the parameters of the RS can be fit more precisely, and the occurred FCY phases are known. This allows the BSO to provide a more accurate estimate of the fastest strategy (again under the assumption of having no opponents). Furthermore, the reinforcement VSE can be trained specifically for the actual race course to compare its reactions with those in reality. To replicate the situation for this case study, it is assumed that the manually fitted parameters from variant 2 in Section 4.6.3 correspond to the parameter set that was determined more precisely after the race. Thus, differences from the *before race* results become apparent in the following, for which the parameters from variant 1 were used.

At first, the basic strategy for the new parameter set is calculated. When FCY phases must be taken into account in the BSO, quadratic programming cannot be used and is replaced by full-factorial simulation. This is the case for the 2019 Chinese Grand Prix due to the VSC phase at the beginning of the race. Subsequently, the reinforcement VSE is trained with the new parameters and specifically for the FCY phases that occurred in reality. Since Leclerc is studied here, only he is controlled by the reinforcement VSE during training. Due to the significantly reduced variation of the race (fixed FCY phases and driver), the number of training steps can be reduced to 100 000 laps. The results of using BSO and reinforcement VSE as a strategy source in the deterministic RS are compared to the strategy Leclerc pursued in the real race in Table 4.10. The five race participants besides Leclerc were simulated with the strategies they pursued in the real race.

Table 4.10: Comparison of Leclerc's simulated result positions and race durations in the 2019 Chinese Grand Prix with three different race strategies. The races were simulated in the deterministic RS with the top six drivers.

Strategy source	Start	First stop	Second stop	Sim. result position	Sim. race duration
Real-world strategy	A4	in-lap 22, A3	in-lap 42, A4	5	5568.251 s
BSO	A4	in-lap 14, A3	in-lap 35, A3	4	5556.056 s
Reinforcement VSE	A4	in-lap 15, A3	in-lap 34, A3	4	5556.303 s

It can be seen that BSO and reinforcement VSE follow virtually the same strategy, leading to a better result in the post-simulation than with the real-world strategy. In both cases, Leclerc remains in the fourth position ahead of Verstappen. Comparing the BSO strategy with that in Table 4.7 shows no difference. On the one hand, this is because the short VSC phase at the beginning of the race has hardly any influence on the course of the race. On the other hand, the tire degradation model parameters for Leclerc in the parameterization of variant 1 and variant 2 match well. The fact that the BSO and reinforcement VSE strategies are nearly identical indicates that Leclerc's race is hardly influenced by driver interactions when following the optimal strategy in the simulation. This can be seen in Figure 4.11, for which Leclerc is simulated following the BSO strategy listed in Table 4.10, while the other drivers follow their strategies from the real race.

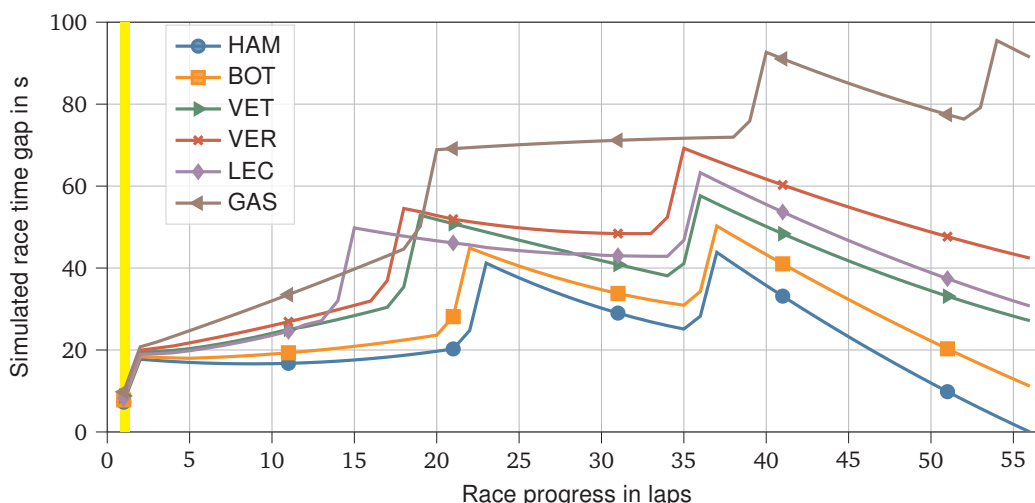


Figure 4.11: Simulated race time gaps to a virtual driver with a constant lap time for the 2019 Chinese Grand Prix. A virtual safety car phase was active from the end of the first lap to the middle of the second lap (yellow bar). Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.

The results suggest that Leclerc could have performed better with an earlier pit stop. A possible explanation would be that Ferrari initially pursued a 1-stop strategy for Leclerc and then decided comparatively late to switch to a 2-stop strategy because they saw higher degradation than expected. However, as already mentioned, the evaluations taken in this paragraph can only serve to reveal possible alternatives for the future, as they are still subject to parameterization errors. In addition, the effects of one's actions on the other drivers' strategies are not taken into account.

Summary

The section has shown which questions are relevant for a strategy engineer during a race weekend and how they can be addressed with the developed tools. It became apparent that most

of the work lies in preparing for a race so that it is possible to react quickly to different situations during the race. The post-race evaluations can reveal potential for future improvements. The given examples show how BSO, supervised VSE, and reinforcement VSE can be used according to Table 4.6 to best support the strategy engineer in his work. The strategy considerations presented are nicely summarized in the following quote: “While a change in the weather tends to cause the maximum amount of disruption, and tire degradation is the most common reason to change plans, the sweet spot between the two is reached when a Safety Car is deployed. It can change a race in an instant and lead teams to make big decisions – though, in reality, those decisions have most likely already been made.” [181].

5 Discussion

This chapter shows the contribution of the individual publications to answering the research questions. Furthermore, critical aspects are highlighted, and an outlook on future research is given.

5.1 Answers to the Research Questions

The following paragraphs answer the three research questions posed in Section 2.5 based on the research conducted.

Research Question 1

How should a race simulation be designed to conduct realistic race strategy studies?

To be useful for race strategy studies, a RS must be able to simulate the course of a race with sufficient accuracy if the parameters are set properly. This capability has been demonstrated for the developed RS in Section 4.6.3. Consequently, it can be said that the lap-wise discretization of a race is a valid design. The same applies to the models. Although comparatively simple relations are used to model tire degradation, burnt fuel mass, and overtaking, these models represent real-world racing mechanics with sufficient accuracy. In addition to the fundamental deterministic relations, the consideration of probabilistic effects is important to put the simulation results in perspective. As addressed in Section 4.6.4, the result of a deterministic simulation is not always the most likely result when probabilistic effects are taken into account. Furthermore, the consideration of probabilistic effects is a prerequisite for using the RS for reinforcement learning, as it was done for the reinforcement VSE. To evaluate the probabilistic effects in a short time, nevertheless, in thousands of runs, the RS as a whole must be designed in such a way that the models require low computational effort. As a final design aspect, the interface between RS and strategy source should be designed flexible enough to allow for the integration of a wide variety of strategy sources as needed.

Research Question 2

How can the necessary parameters for a race simulation be robustly determined with little knowledge of the exact vehicle, driver, and track characteristics?

To answer the question, it must be distinguished whether the parameters for the RS are to be determined before or after a race and whether or not there is access to team-internal data and all timing data. For this thesis, there was no access to team-internal data, and timing data was only available for the races themselves, not for preceding free practice and qualifying sessions. Starting from this point, two methods have to be combined to enable parameter determination before and after a race. The LTS together with the upstream racing line generation

allow determining some crucial parameters of the RS before a race. These parameters can be integrated with available parameter sets from previous races to simulate the race. Since the LTS itself was designed to be used with little parameterization effort, a qualifying lap recording with overlaid telemetry data is enough to adjust its parameters sufficiently accurately to the respective race, as explained in Section 4.6.2. After a race, a large amount of timing data is available that can be processed automatically to create a parameter set for the RS. However, this processing poses several challenges, as various unknown influences have acted on the data in the real-world race, which often cannot be separated in retrospect, e.g., the influence of a damaged front wing on the lap time. In addition, required data is often unavailable, e.g., if a driver did not run one of the available tire compounds. Consequently, to robustly create reasonable parameter sets, much of the data must be sorted out or combined across multiple drivers, which reduces accuracy. If necessary, the LTS can be used for plausibility checks of some crucial parameters after the automated parameter determination, as it allows to determine them in isolation. Furthermore, in case it is important to resimulate the real race as accurately as possible, the generated parameter set can be improved manually by the user, as shown in Section 4.6.3. The combination of both simulation and real-world data processing allows a robust and largely automated determination of the RS parameters for the required use cases from an external point of view.

Research Question 3

How can the race strategies of opposing drivers be determined automatically in a race simulation so that own race strategy decisions can be evaluated objectively?

Making reasonable race strategy decisions requires taking into account the specifics of the particular driver-car combination and incorporating and exploiting the ever-changing race situation. Fulfilling both aspects should therefore be the goal in automated race strategy determination. Due to the many influences on the course of a race and the nature of the problem, conventional optimization is only applicable for this task with reduced complexity. Thus, BSO allows a fast strategy determination specifically adapted to the tire degradation model parameters of the respective driver-car combination but neglects all driver interactions. Although this theoretically limits the transferability of the determined strategies to the actual race, Section 4.6.5 has shown that the strategies work well as long as no unexpected race events occur. Besides optimization, machine learning methods are another possibility to tackle the task. The implementation in the form of the supervised VSE has shown that reasonable decisions can be made based on data from past seasons, also taking into account sudden race events. The combination of specifically adapted strategies and inclusion of the dynamic race situation could only be achieved with a reinforcement learning approach, in which the agent is trained in the RS specifically for the respective race. Section 4.6.5 has shown that all three variants of automated strategy determination have different strengths. Consequently, they should be used depending on the particular phase of a race weekend to best support the evaluation of race strategy decisions.

5.2 Critical Points

The detailed discussions of the respective tools are included in the corresponding publications. Therefore, the criticism in this section is written from a more general perspective on the entire toolchain.

General Situation

The most critical point for the modeling, parameterization, and validation performed in this thesis is the extensive secrecy in motorsport, which leads to poor availability of know-how and data of all kinds, especially from an external point of view. As a result, the teams' actual state of the art is unclear in most aspects, making a final assessment of the performance of the tools impossible. Primarily due to limited data, the development and evaluation of the tools focused on F1. Even for F1, however, the data situation can only be rated as sufficient, so that the achievable accuracies of LTS and RS shown in the corresponding papers and the case study in Section 4.6 could probably be improved. In the LTS, for example, the parameters of the tires, aerodynamics, and engine performance have a significant influence on the results, which are available within a racing team with a high degree of accuracy but can only be estimated from an external point of view. In the RS, for example, the unknown vehicle condition (fuel mass and engine mode) and the unknown targets for the driver (tire-friendly and energy-saving driving vs. aggressive driving) during the practice sessions and the race make parameterization difficult.

Approaches

In retrospect, it can be stated that the approaches chosen for the individual simulation tools have proven to be a good compromise between the development targets stated in section 3.1, especially taking into account the poor data situation. For racing line generation and LTS, alternative approaches with higher model accuracy are conceivable. For example, both tools could be combined when using an optimal control approach, such as some of the literature shown in Section 2. However, in addition to the significantly higher modeling effort, this would have resulted in a greater parameterization effort, which would not have allowed reasonable use in the context of this thesis.

For the RS, the use of mainly empirical models seems to be the only reasonable option. However, improving the simulation accuracy could be achieved by changing the lap-wise discretization to a time-based discretization, as this would significantly increase the freedom in modeling. For example, overtaking maneuvers could be modeled depending on the course of the race track and the DRS zones. Also, the inaccuracies in the modeling of VSC and SC phases addressed in [12] could be solved. The common disadvantages, especially the further increasing parameterization effort and the higher computing times, could probably be compensated by further work on the automated parameter determination and changing the programming language. A prototype of such a time-discrete race simulation is available on GitHub [182].

Based on the research, machine learning methods seem to be a reasonable way to automate race strategy decisions. The advantages outweigh the disadvantages of the approach, e.g., that sometimes decisions are made that the user cannot comprehend. The reinforcement learning approach offers the greatest potential for future research. It requires less data and is less susceptible to regulations and race track changes than supervised learning. However, the approach needs further development as the existing implementation shows unstable decision behavior in certain situations. This is the case, for example, when it was trained with all 20 drivers of a race and is then used for simulations that include only the top six drivers, as was the case in the case study in Section 4.6. As a result, the last of the six drivers has no opponent behind him, which has never occurred for him in training. As he does not lose any position due to a pit stop, the reinforcement VSE decides to put on fresh tires every few laps.

Plausibilization and Validation

For each of the tools, the results were checked for plausibility and, where possible, validated against real data:

- The racing line calculated by the racing line generation was compared with that of an optimal control approach for the FE race track in Berlin in [86], showing the expected similarities and differences. Furthermore, the racing line generation results were plausibilized on many race tracks in the course of the research work. An excerpt from this procedure is shown in the case study in Section 4.6. In the context of the Roborace racing series, the racing lines calculated by the racing line generation were driven with a real autonomous race car on various European race tracks.
- For the LTS, the calculated velocity profile of the Shanghai race track was validated against real data and the result of the professional LTS RaceSim [183] in [9]. The same was done for the Monza race track in [166]. In the case study in Section 4.6, the comparison with real data for the Shanghai race track was renewed for the slightly updated LTS. In addition, many different scenarios were plausibilized during the development of the LTS, for example, the effects of mass variations, power variations, and friction value variations.
- For the validation of the deterministic RS, it was shown that a race can be accurately resimulated if an appropriate parameter set is used. This was done in [8] for the 2017 Abu Dhabi Grand Prix and in the case study in Section 4.6 for the 2019 Chinese Grand Prix. In [12], the newly established submodels for the probabilistic part of the RS were developed and validated using real data. The interaction of all submodels was then tested in various ways [12]. First, the effects of an SC phase on the race course were validated based on real data. Then, the effects of various strategy and SC scenarios were plausibilized. Finally, the correct fraction of FCY phases in the simulated races and the reproducibility of the results of the probabilistic Monte Carlo simulations were tested. In addition, a plausibilization of Monte Carlo simulation results was performed within the case study in Section 4.6 for the 2019 Chinese Grand Prix.
- The optimization-based BSO was validated in [13] (within the constraints imposed by the underlying principle). For the supervised VSE, the characteristics of its pit stop decisions and compound choices were analyzed and plausibilized in detail in [13]. However, due to the machine learning approach, it must be assumed that there are edge cases for which the decision behavior might not be plausible. The same is true for the reinforcement VSE introduced in Section 4.5.2. Its decision behavior was analyzed and plausibilized in [176]. However, as described before, it became apparent that it is not always stable in situations that were not or insufficiently represented during training.

It should be noted that the LTS, the RS, as well as the different variants of the VSE were only tested and validated using data from F1. Using the tools for other racing series should be possible without problems but requires a renewed validity check.

Parameterizability

A point to mention is the comparatively high effort for the parameterization of the RS. The parameters themselves are kept simple through the extensive use of empirical models. However, depending on the number of drivers and teams, the total number of parameters to be determined per race quickly adds up. For example, for a F1 race in the 2019 season, with 20 drivers and ten teams per race, about 200 parameters need to be determined. The number can be reduced, for example, by parameterizing and simulating only the relevant opposing drivers. Nevertheless, the effort for manual parameterization is high. By using the procedure presented in Section 4.4.2, it is possible to automatically create a parameter set for a race, at least retrospectively. It can then be used as a starting point for subsequent races on the respective race track. For higher accuracy or parameterization based solely on data from practice sessions and qualifying, experience and manual work are nevertheless required.

Missing Aspects

Wet races were not considered within the scope of this thesis. In the RS, worsening conditions can, in principle, be easily implemented via a further time loss. In reality, however, setting such a parameter would hardly be possible, even within a team. On the one hand, a measure would be needed to represent the humidity of the track from *dry* to *standing water*. On the other hand, models would have to be set up to map the effects of such a measure on lap times in combination with the available tire compounds. In addition, further effects would also have to be taken into account, such as increasing accident probabilities. All of this would have to be developed and parameterized based on a comparatively small number of wet races (eleven out of 121 races in the timing database [159]). For these reasons, it was decided to focus on dry conditions in this thesis. In practice, it can be assumed that feedback from the drivers is used in most cases to decide when it makes sense to switch to wet or dry tires. In addition, of course, the lap time data of competitors who have already changed before making the own decision will also be taken into account.

Game theory is another topic that has not been considered in this thesis but plays an important role in real-world races. The goal in this context is to adapt the strategy to the actions of the closest competitor(s). The developed simulation tools are needed as a prerequisite for the associated simulations. Often, two-player games are used for game theory studies because they are well understood. Consequently, it must be known against which opponent the own driver mainly races. With this information, the average expected outcomes of the different options (e.g., comparing three possible strategies for each of the two drivers) could be simulated using the probabilistic RS. Depending on the results, the own strategy decision can be made. Another aspect in this context is team strategy. Since there are two drivers per team, one driver can try to protect the other driver from an opponent, for example, or the strategies could be split to maximize the expected points of the whole team rather than just for one driver. Sulsters [112] and Salminen [184] can be used as a starting point on these aspects.

Chapter 4.6.3 showed that the same tire compound degrades at different rates at the beginning and end of a race, which currently cannot be represented in the RS. One reason for this observation is the missing dependence of the tire degradation model on the vehicle mass. When more data are available, this aspect should be investigated since the degradation model has a large impact on the RS results.

5.3 Outlook

The toolchain developed in this thesis provides a solid starting point for further research and practical application. The scientific relevance is evident, for example, from the fact that the RS has already been used in another publication by Piccinotti et al. [185] to evaluate automated race strategy decisions similar to [13]. Great interest is also directed towards the racing line generation, which is already widely used in trajectory planning on race tracks. Finally, industrial relevance can be assumed due to a question from a F1 team member concerning the reinforcement VSE, indicating that the racing teams are working on similar topics.

Approaches for future research have been addressed in the various publications and the previous criticism. Apart from improving the tools and analyzing game-theoretical aspects, the focus should be on further evaluating the toolchain, both in terms of achievable accuracy and transferability to other racing series. For example, it would be interesting to apply the tools and methods developed in this thesis for sprint races to endurance races and investigate what mainly influences race strategy in long-distance motorsport. However, such investigations depend on the availability of new data. In addition, to obtain meaningful results, the automated parameter determination should be further developed and improved before evaluation.

Another possible field of future research is investigating the effects of changes to the regulations or technical specifications on race strategies and the course of a race. Examples of the former are a change of the minimum mass of the race cars, the repeatedly discussed reintroduction of fuel stops, or the inversion of the starting order. Regarding the technical specifications, it would be interesting to see the impact of changes in tire development specifications made to the respective manufacturer in terms of desired degradation behavior to enable exciting races. These aspects are particularly interesting with regard to the new F1 regulations for the 2022 season.

Concerning real-world application, the use within a racing team should, of course, be the goal to be able to incorporate aspects from practical use. Beyond that, however, use in the context of sports betting could also be a possible use case. For example, the tools could help bookmakers determine the odds of the bets offered in a well-founded way. On the user side, it could be investigated whether profit could be increased by using the toolchain.

6 Summary

This thesis deals with developing methods and simulation tools needed for the realistic simulation of circuit races. This is an essential prerequisite for the objective evaluation of race strategy decisions in motorsport.

In the beginning, an introduction to motorsport and especially the relevant aspects of race strategy was given. The general goal of race strategy is to achieve the best possible result(s) for one's own driver(s) under the given conditions. Race strategy determines when and how often a car comes to the pit, for example, to put on fresh tires, refuel, or change drivers. The possible options for action depend on the regulations of the respective racing series. The advantages of a pit stop, e.g., faster lap times due to a fresh set of tires, have to be weighed against the disadvantage that the driver loses time and often positions compared to the rest of the field during the stop. In addition, many other aspects play into the decision. For example, the strategy must be adapted to sudden changes in the race situation, including retirements and accidents, to take advantage of the opportunities that arise.

Caused by great secrecy in motorsport, only a few publications and relevant data are available in the context of race simulation and race strategy determination. This thesis aims to fill this gap and thus prepare the ground for further research. As a starting point, a literature review was conducted in four research areas related to the simulation of circuit races: racing line generation, Lap Time Simulation (LTS), Race Simulation (RS), and automated decision making in sports. The review revealed various possibilities for improving established approaches as well as open research areas. Based on these findings, three overarching research questions were established.

To answer the research questions, the approach was first defined. It consists of four simulation tools that form a contiguous toolchain: racing line generation, LTS, RS, and the automated making of race strategy decisions. Due to the data situation, the importance of race strategy for the races, and the popularity, it was decided to focus and evaluate the conducted research on the FIA Formula 1 Championship (F1). Nevertheless, the tools were developed in such a way that they are transferable to other racing series. Fast computing times and a simple parameterization were defined as further requirements for the tools.

The goal of the racing line generation is to calculate the racing line for a given race track. The chosen implementation minimizes the sum of the discretized quadratic curvature of the racing line. Advantages of this approach are fast computing times and an inherently smooth curvature profile. It was shown that the resulting racing line is close to the minimum-time solution. In the context of this tool, a database was also created that provides the center lines, track widths, and calculated racing lines of 25 race tracks from around the world.

The racing line serves as an input for the LTS. Its goal is to calculate the lap time for a given car as accurately as possible. Therefore, for each point along the discretized racing line, the possible and used lateral accelerations are calculated employing a simplified two-track model to determine the tire potential remaining for longitudinal acceleration. Considering the limits introduced by

the powertrain, this can be used to determine the velocity profile and thus the expected lap time. Within the toolchain, the LTS can be used to determine some critical parameters for the subsequent RS in case real-world timing data are not yet available or to verify parameters determined from timing data by automated data processing.

The RS enables the simulation of the expected course of a race and, therefore, an evaluation of the effects of different race strategies on the race result. The implementation is based on the mainly empirically-based calculation of the lap times of the individual drivers. It takes into account, for example, the degradation of tires and the reduction in vehicle mass due to burnt fuel. Furthermore, the interactions between the drivers, i.e., overtaking maneuvers, are modeled. Another important aspect is the integration of probabilistic effects, such as lap time variations or failure and accident probabilities. Failures and accidents can cause yellow phases, which reduce the time lost during a pit stop and are thus particularly relevant for race strategy determination.

The race strategies of all drivers in a race must be given to the RS. Consequently, a realistic simulation of a race requires a component that performs this task, at least for the opposing drivers. For this purpose, three different concepts of the Virtual Strategy Engineer (VSE) were developed. One is based on a quadratic optimization problem, two use machine learning techniques. Based on the driver-specific parameterization of the tire degradation model, the optimization problem determines an optimal basic strategy for the race for which opposing cars are neglected. The machine learning methods, in contrast, are invoked once per lap and driver during the (simulated or real) race to decide whether or not to make a pit stop depending on the current race situation.

For the determination of the parameters required for the RS, a database of real-world timing data with 121 F1 races of the 2014 to 2019 seasons was created. In addition, it allows the simulation results of LTS and RS to be compared with reality and the race strategy decisions made by the VSE to be compared with those made by human strategy engineers.

After the presentation and discussion of the developed methods and tools, the interaction of the toolchain was shown through a case study. For this purpose, the toolchain was worked through from front to back using the 2019 Chinese Grand Prix in Shanghai as an example, and the results were presented and evaluated for each of the simulation tools. In the last part of the case study, it was shown how the RS can be used together with the VSE in the different phases of a race weekend to evaluate race strategies objectively. It became apparent that a large part of the strategy work consists of preparing for different scenarios for the race.

Subsequently, the research questions were answered. It was concluded that the lap-wise discretization of a race, as well as the empirical basis of many of the models, are valid design choices for the RS. In addition, the importance of including probabilistic effects to put simulation results into perspective was emphasized. Regarding the parameterization of the RS, it was stated that the combination of racing line generation and LTS supports parameter determination well, especially when timing data are not yet available before a race. In addition, it was found that from the external point of view of this thesis, robust automated parameter determination based on the timing data available after a race is of great importance. Finally, it was answered how race strategy decisions can be made in an automated way by using the VSE. Especially the reinforcement learning approach promises great potential when training is improved.

The thesis was concluded by pointing out critical points and future research needs. This part also addressed aspects that were not considered, such as wet races and game-theoretical influences on strategy decisions. All code and the databases used within the thesis are available under an open-source license on GitHub.

List of Figures

Figure 1.1:	Layout of the Hockenheimring located in southern Germany based on [4].	2
Figure 1.2:	Comparison of the cumulative time losses due to tire degradation with and without making a pit stop in an example scenario.	3
Figure 1.3:	Visualization of the lap time loss due to tire degradation for three different compounds using a linear degradation model with example parameters. ...	4
Figure 1.4:	Excerpts of the Silverstone and Monaco track maps based on [15]. The pit lanes are shown in black, the tracks themselves in gray. The blue areas mark the actual pit locations of the teams.	9
Figure 2.1:	Visualization of three different types of lines through a corner: classic racing line, early apex line, and late apex line based on [36, p. 323].	14
Figure 2.2:	An example g-g diagram showing three possible shapes for modeling the acceleration relationship: circle, ellipse, and rhombus. The positive longitudinal acceleration is limited by the engine power and not by the tires.	17
Figure 2.3:	Example velocity profile to demonstrate the working principle of a forward-backward solver.	18
Figure 3.1:	Overview of the approach that allows an evaluation of the effects of race strategy decisions on a race.	25
Figure 4.1:	Example of a race track section with different track widths w_1 and w_2 along the track.	29
Figure 4.2:	Example visualization of the process for determining temporary mass sensitivity values based on some of Hamilton's lap times in the 2015 Hungarian Grand Prix. The three point clusters belong to three different tire sets of the A3 compound. The four lines depict the resulting linear regression models fitted for each tire age, with the slope corresponding to a temporary mass sensitivity value in each case.	88
Figure 4.3:	Hamilton's lap times of the first and third stint in the 2019 Chinese Grand Prix, both driven on the A4 compound. Besides the original values, the values after removing t_{fuel} are shown. In addition, the fitted linear regression model is plotted.	89
Figure 4.4:	The racing line on the Shanghai International Circuit as generated by the (iterative) minimum-curvature approach. The numbers from one to 16 indicate the corners. S1 – Sector 1, S2 – Sector 2, FL – Finish line.	149
Figure 4.5:	Detailed view of the racing line in the corners one to four on the Shanghai International Circuit. The traces of the vehicle width are displayed, which are considered in the optimization problem.	150
Figure 4.6:	Comparison of the curvature profiles of the (already smoothed) center line and the generated racing line. S1 – Sector 1, S2 – Sector 2.	151

Figure 4.7: Comparison of Hamilton’s real velocity profile (from the qualifying for the 2017 Chinese Grand Prix) and the velocity profile simulated with the lap time simulation on the Shanghai International Circuit. S1 – Sector 1, S2 – Sector 2.152

Figure 4.8: Real (first chart) and simulated (other charts) race time gaps to a virtual driver with a constant lap time for the 2019 Chinese Grand Prix. A virtual safety car phase was active from the end of the first lap until the middle of the second lap (yellow bar). Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.155

Figure 4.9: Distributions of final rank positions after 10 000 simulation runs of the 2019 Chinese Grand Prix with activated probabilistic effects and without adjustments of the race strategies. Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.158

Figure 4.10: Simulated race durations for Leclerc at the 2019 Chinese Grand Prix as a function of the pit stop in-lap for three different 1-stop scenarios. Black dots mark the optimal in-laps of the three scenarios. The race durations were determined in the simplified RS.163

Figure 4.11: Simulated race time gaps to a virtual driver with a constant lap time for the 2019 Chinese Grand Prix. A virtual safety car phase was active from the end of the first lap to the middle of the second lap (yellow bar). Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.165

List of Tables

Table 1.1:	Example timing board showing the race state for the top five drivers at the beginning of lap 16 in the 2019 Hungarian Grand Prix. Driver abbreviations: VER – Verstappen, HAM – Hamilton, LEC – Leclerc, VET – Vettel, SAI – Sainz. Compound abbreviations: M – Medium, S – Soft.	8
Table 1.2:	Overview of available tire compounds in the seasons from 2014 to 2019 as published in [13]. The column names are inspired by the 2019 season compound names to enable comparison of the compounds over the years. A1 is the hardest compound, and A7 is the softest.	11
Table 2.1:	Overview and qualitative evaluation of the modeled probabilistic effects in race simulations, as published in [12]. The more the circle is filled with black, the more detailed the effect was modeled.	20
Table 4.1:	Excerpt of the <i>laps</i> table showing the race state for positions one to five at the end of the first lap of the 2019 Shanghai Grand Prix. Some columns included in the database [159] are not listed here due to space constraints.	31
Table 4.2:	Comparison of BSO, supervised VSE, and reinforcement VSE based on the average result position achieved with randomly selected drivers after 10 000 simulated races. Probabilistic effects, including accidents and failures, are enabled. The results are taken from Thomaser [176, p. 45].	148
Table 4.3:	Comparison between Hamilton’s real sector and lap times (qualifying of the 2017 Chinese Grand Prix) and the simulation results on the Shanghai International Circuit.	151
Table 4.4:	RS parameters that can be estimated using the LTS. The example values are determined for the 2019 Chinese Grand Prix. The values in the <i>timing data</i> column were determined according to the procedures described in Section 4.4.2.	153
Table 4.5:	Actual and simulated rank positions in different states of the 2019 Chinese Grand Prix. Driver abbreviations: HAM – Hamilton, BOT – Bottas, VET – Vettel, VER – Verstappen, LEC – Leclerc, GAS – Gasly.	154
Table 4.6:	Overview of the suitability of the three available options for determining the race strategy by software in dependence on the particular phase of the race weekend. The more the circle is filled with black, the more suitable an option is.	159
Table 4.7:	Comparison between Leclerc’s best 1-stop, 2-stop, and 3-stop strategies as determined by the BSO for the 2019 Chinese Grand Prix.	161
Table 4.8:	Simulated resulting rank positions p and race durations $t_{\text{race,tot}}$ for Leclerc in the 2019 Chinese Grand Prix. For the probabilistic simulation results, the average values after 10 000 simulation runs are given.	162
Table 4.9:	Comparison of the strategies output by supervised VSE and reinforcement VSE for Leclerc in the 2019 Chinese Grand Prix in the simulation without FCY phases and with some early FCY phases.	163

Table 4.10: Comparison of Leclerc’s simulated result positions and race durations in the 2019 Chinese Grand Prix with three different race strategies. The races were simulated in the deterministic RS with the top six drivers.165

Bibliography

- [1] R. Brawn and A. Parr, *Total Competition: Lessons in Strategy from Formula One*, Simon & Schuster UK Limited, 2017, ISBN: 9781471162381.
- [2] J. Walz, *Geschichte des Motorsports*, Delius Klasing Verlag Bielefeld, 2017, ISBN: 9783667112996.
- [3] M. Trzesniowski and P. Eder, *Handbuch Rennwagenteknik: Gesamtfahrzeug*, Springer Vieweg Wiesbaden, 2017, DOI: 10.1007/978-3-658-15537-7.
- [4] Wikipedia. „Hockenheimring“, Available: <https://en.wikipedia.org/wiki/Hockenheimring> [visited on 01/23/2022].
- [5] Dr. Ing. h.c. F. Porsche AG. „Race strategy: a high-paced game with many unknowns“, 2017. Available: <https://presskit.porsche.de/motorsport/en/mediaguide-2017/topic/in-focus/race-strategy.html> [visited on 01/23/2022].
- [6] Fédération Internationale de l'Automobile. „2019 F1 Sporting Regulations“, 2019. Available: https://www.fia.com/sites/default/files/2019_sporting_regulations_-_2019-03-12.pdf [visited on 01/23/2022].
- [7] Fédération Internationale de l'Automobile. „2019 F1 Technical Regulations“, 2019. Available: https://www.fia.com/sites/default/files/2019_technical_regulations_-_2019-03-12.pdf [visited on 01/23/2022].
- [8] A. Heilmeier, M. Graf and M. Lienkamp, „A Race Simulation for Strategy Decisions in Circuit Motorsports“, in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2986–2993, DOI: 10.1109/ITSC.2018.8570012.
- [9] A. Heilmeier, M. Geisslinger and J. Betz, „A quasi-steady-state lap time simulation for electrified race cars“, in *2019 Fourteenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*, 2019, DOI: 10.1109/EVER.2019.8813646.
- [10] The Chelsea Magazine Company, „Strategic thinking – How RaceWatch helps to predict the unpredictable“, *Racecar Engineering*, no. 7, pp. 79–86, 2019.
- [11] C. Nimmervoll. „Benzinverbrauch: Warum Mercedes' Vorteil doppelt zählt“, 2018. Available: <https://www.motorsport-total.com/formel-1/news/benzinverbrauch-warum-mercedes-vorteil-doppelt-zaehlt-18031606> [visited on 01/23/2022].
- [12] A. Heilmeier, M. Graf, J. Betz and M. Lienkamp, „Application of Monte Carlo Methods to Consider Probabilistic Effects in a Race Simulation for Circuit Motorsport“, *Applied Sciences*, vol. 10, no. 12, p. 4229, 2020, DOI: 10.3390/app10124229.
- [13] A. Heilmeier, A. Thomaser, M. Graf and J. Betz, „Virtual Strategy Engineer: Using Artificial Neural Networks for Making Race Strategy Decisions in Circuit Motorsport“, *Applied Sciences*, vol. 10, no. 21, p. 7805, 2020, DOI: 10.3390/app10217805.

- [14] J. Palmer. „*Jolyon Palmer's Analysis: Singapore and the art of undercutting*“, 2019. Available: <https://www.formula1.com/en/latest/article.jolyon-palmers-analysis-singapore-and-the-art-of-undercutting.1NgVyVsZnHTDEA9wi0s5IW.html> [visited on 01/23/2022].
- [15] Fédération Internationale de l'Automobile. „*F1 Archives*“, Available: <https://www.fia.com/f1-archives> [visited on 01/23/2022].
- [16] Liberty Media Corporation. „*F1 Fact Sheet 2019*“, 2019. Available: <http://libertymedia.com/pdfs/Formula-1-Fact-Sheet-July-2019.pdf> [visited on 08/26/2020].
- [17] Fédération Internationale de l'Automobile. „*History of Formula E*“, Available: <https://www.fiaformulae.com/en/discover/history> [visited on 01/23/2022].
- [18] Fédération Internationale de l'Automobile. „*Formula E's Michelin Pilot Sport EV tyres explained*“, 2021. Available: <https://www.fiaformulae.com/en/news/2021/april/formula-e-michelin-tyres-explained> [visited on 01/23/2022].
- [19] T. Grüner. „*Mercedes vs. Audi vs. BMW*“, 2011. Available: <https://www.auto-motor-und-sport.de/motorsport/dtm-2012-auto-technik-regeln-mercedes-vs-bmw-vs-audi> [visited on 01/23/2022].
- [20] A. Reiners. „*DTM: Die Regeländerungen und Neuheiten 2013*“, 2013. Available: <https://www.speedweek.com/dtm/news/37228/DTM-Die-Regelaenderungen-und-Neuheiten-2013.html> [visited on 01/23/2022].
- [21] Deutscher Motorsport Bund. „*2016 DTM Sporting Regulations*“, 2016.
- [22] S. Haidinger. „*DTM set to bring back pitstop window in 2020*“, 2020. Available: <https://www.motorsport.com/dtm/news/pitstop-window-return-drs-p2p/4845434> [visited on 01/23/2022].
- [23] S. Haidinger. „*DTM set to use single Michelin tyre in first GT3 season*“, 2021. Available: <https://www.motorsport.com/dtm/news/single-michelin-medium-tyre-gt3-cars/6506631> [visited on 01/23/2022].
- [24] Fédération Internationale de l'Automobile. „*WEC Archives*“, Available: <https://www.fiawec.com/en/past-seasons/36> [visited on 01/23/2022].
- [25] Fédération Internationale de l'Automobile. „*2021 WEC Sporting Regulations*“, 2021.
- [26] D. Rencken. „*The cost of F1 revealed: How much teams spent in 2018 – part two*“, 2018. Available: <https://www.racefans.net/2018/12/26/the-cost-of-f1-revealed-how-much-teams-spent-in-2018-part-two> [visited on 01/23/2022].
- [27] D. Rencken. „*The cost of F1 revealed: How much teams spent in 2018 – part one*“, 2018. Available: <https://www.racefans.net/2018/12/19/how-much-f1-teams-spent-race-2018-part-one> [visited on 01/23/2022].
- [28] Fédération Internationale de l'Automobile. „*2014 F1 Sporting Regulations*“, 2014.
- [29] Fédération Internationale de l'Automobile. „*2017 F1 Sporting Regulations*“, 2017.
- [30] Motorsport Technology. „*Overtaking in Formula 1 – DRS*“, 2018. Available: <https://motorsport.tech/formula-1/overtaking-in-formula-1-drs> [visited on 01/23/2022].
- [31] J. Noble. „*Mercedes discovers true cause of Hamilton's lost Australian GP win*“, 2018. Available: <https://www.autosport.com/f1/news/135092/mercedes-discovers-true-cause-of-lost-win> [visited on 01/23/2022].
- [32] S. Mitchell and V. Khorounzhiy. „*Mercedes F1 boss Wolff 'understands' Ferrari's Spanish GP strategy*“, 2018. Available: <https://www.autosport.com/f1/news/136004/wolff-understands-ferrari-costly-vettel-stop> [visited on 01/23/2022].

- [33] S. Codling. „*Singapore Grand Prix: Lewis Hamilton wins, F1 title rival Vettel third*“, 2018. Available: <https://www.autosport.com/f1/news/138756/hamilton-wins-in-singapore-vettel-only-third> [visited on 01/23/2022].
- [34] J. P. Samper Mejia, P. A. Theodosis and J. C. Gerdes, „Using a Path-Fitting Algorithm to Analyze the Racing Techniques of a Skilled Driver“, in *Proceedings of the ASME 2013 Dynamic Systems and Control Conference*, 2013, DOI: 10.1115/dscc2013-4106.
- [35] P. A. Theodosis and J. C. Gerdes, „Generating a Racing Line for an Autonomous Racecar Using Professional Driving Techniques“, in *Proceedings of the ASME 2011 Dynamic Systems and Control Conference*, 2011, DOI: 10.1115/dscc2011-6097.
- [36] M. Trzesniowski and P. Eder, *Handbuch Rennwagentchnik: Datenanalyse, Abstimmung und Entwicklung*, Springer Vieweg Wiesbaden, 2017, DOI: 10.1007/978-3-658-15547-6.
- [37] B. Paden, M. Cap, S. Z. Yong, D. Yershov and E. Frazzoli, „A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles“, *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016, DOI: 10.1109/TIV.2016.2578706.
- [38] F. Braghin, F. Cheli, S. Melzi and E. Sabbioni, „Race driver model“, *Computers & Structures*, vol. 86, no. 13-14, pp. 1503–1516, 2008, DOI: 10.1016/j.compstruc.2007.04.028.
- [39] J. T. Betts, „Survey of Numerical Methods for Trajectory Optimization“, *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998, DOI: 10.2514/2.4231.
- [40] E. Polak, „An Historical Survey of Computational Methods in Optimal Control“, *SIAM Review*, vol. 15, no. 2, pp. 553–584, 1973, DOI: 10.1137/1015071.
- [41] B. Gutjahr, L. Groll and M. Werling, „Lateral Vehicle Trajectory Optimization Using Constrained Linear Time-Varying MPC“, *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2016, DOI: 10.1109/tits.2016.2614705.
- [42] E. W. Dijkstra, „A Note on Two Problems in Connexion with Graphs“, *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959, DOI: 10.1007/BF01386390.
- [43] P. E. Hart, N. J. Nilsson and B. Raphael, „A Formal Basis for the Heuristic Determination of Minimum Cost Paths“, *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968, DOI: 10.1109/TSSC.1968.300136.
- [44] I. Pohl, *First Results on the Effect of Error in Heuristic Search*, (MIP-R), Edinburgh University, Department of Machine Intelligence and Perception, 1969.
- [45] A. Stentz, „Optimal and efficient path planning for partially-known environments“, in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 1994, pp. 3310–3317, DOI: 10.1109/ROBOT.1994.351061.
- [46] A. Stentz, „The Focussed D* Algorithm for Real-time Replanning“, in *Proceedings of the 14th International Joint Conference on Artificial Intelligence – Volume 2*, 1995, pp. 1652–1659, ISBN: 1558603638.
- [47] S. Koenig and M. Likhachev, „Fast replanning for navigation in unknown terrain“, *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, 2005, DOI: 10.1109/TRO.2004.838026.
- [48] S. M. Lavalle, „Rapidly-Exploring Random Trees: A New Tool for Path Planning“, Iowa State University, 1998.
- [49] S. Karaman and E. Frazzoli, „Optimal kinodynamic motion planning using incremental sampling-based methods“, in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 7681–7687, DOI: 10.1109/CDC.2010.5717430.

- [50] M. Otte and E. Frazzoli, „RRT-X: Real-Time Motion Planning/Replanning for Environments with Unpredictable Obstacles“, in *Springer Tracts in Advanced Robotics*, 2015, pp. 461–478, DOI: 10.1007/978-3-319-16595-0_27.
- [51] A. Katriniok and D. Abel, „LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics“, in *IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 6828–6833, DOI: 10.1109/CDC.2011.6161257.
- [52] J. K. Subosits and J. C. Gerdes, „From the Racetrack to the Road: Real-Time Trajectory Replanning for Autonomous Driving“, *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 309–320, 2019, DOI: 10.1109/tiv.2019.2904390.
- [53] L. Cardamone, D. Loiaco, P. L. Lanzi and A. P. Bardelli, „Searching for the optimal racing line using genetic algorithms“, in *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, 2010, DOI: 10.1109/itw.2010.5593330.
- [54] N. Kapania, J. Subosits and J. Gerdes, „A Sequential Two-Step Algorithm for Fast Generation of Vehicle Racing Trajectories“, *Journal of Dynamic Systems, Measurement, and Control*, vol. 138, no. 9, 2016, DOI: 10.1115/1.4033311.
- [55] A. Liniger, A. Domahidi and M. Morari, „Optimization-based autonomous racing of 1:43 scale RC cars“, *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2014, DOI: 10.1002/oca.2123.
- [56] M. Gerds, S. Karrenberg, B. Müller-Beßler and G. Stock, „Generating locally optimal trajectories for an automatically driven car“, *Optimization and Engineering*, vol. 10, no. 4, pp. 439–463, 2008, DOI: 10.1007/s11081-008-9047-1.
- [57] R. Verschueren, S. De Bruyne, M. Zanon, J. V. Frasch and M. Diehl, „Towards time-optimal race car driving using nonlinear MPC in real-time“, in *53rd IEEE Conference on Decision and Control*, 2014, DOI: 10.1109/cdc.2014.7039771.
- [58] M. Brunner, U. Rosolia, J. Gonzales and F. Borrelli, „Repetitive learning model predictive control: An autonomous racing example“, in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, DOI: 10.1109/cdc.2017.8264027.
- [59] U. Rosolia, A. Carvalho and F. Borrelli, „Autonomous racing using learning Model Predictive Control“, in *2017 American Control Conference (ACC)*, 2017, DOI: 10.23919/acc.2017.7963748.
- [60] F. Christ, A. Wischnewski, A. Heilmeier and B. Lohmann, „Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients“, *Vehicle System Dynamics*, vol. 59, no. 4, pp. 588–612, 2021, DOI: 10.1080/00423114.2019.1704804.
- [61] A. Rucco, G. Notarstefano and J. Hauser, „An Efficient Minimum-Time Trajectory Generation Strategy for Two-Track Car Vehicles“, *IEEE Transactions on Control Systems Technology*, vol. 23, no. 4, pp. 1505–1519, 2015, DOI: 10.1109/tcst.2014.2377777.
- [62] T. Herrmann, F. Passigato, J. Betz and M. Lienkamp, „Minimum Race-Time Planning-Strategy for an Autonomous Electric Racecar“, in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, DOI: 10.1109/itsc45102.2020.9294681.
- [63] D. P. Kelly and R. S. Sharp, „Time-optimal control of the race car: a numerical method to emulate the ideal driver“, *Vehicle System Dynamics*, vol. 48, no. 12, pp. 1461–1474, 2010, DOI: 10.1080/00423110903514236.

- [64] G. Perantoni and D. J. Limebeer, „Optimal control for a Formula One car with variable parameters“, *Vehicle System Dynamics*, vol. 52, no. 5, pp. 653–678, 2014, DOI: 10.1080/00423114.2014.889315.
- [65] N. Dal Bianco, R. Lot and M. Gadola, „Minimum time optimal control simulation of a GP2 race car“, *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 232, no. 9, pp. 1180–1195, 2017, DOI: 10.1177/0954407017728158.
- [66] N. Dal Bianco, E. Bertolazzi, F. Biral and M. Massaro, „Comparison of direct and indirect methods for minimum lap time optimal control problems“, *Vehicle System Dynamics*, vol. 57, no. 5, pp. 665–696, 2019, DOI: 10.1080/00423114.2018.1480048.
- [67] R. Verschueren, M. Zanon, R. Quirynen and M. Diehl, „Time-optimal race car driving using an online exact hessian based nonlinear MPC algorithm“, in *2016 European Control Conference (ECC)*, 2016, DOI: 10.1109/ecc.2016.7810277.
- [68] L. Leonelli and D. J. N. Limebeer, „Optimal control of a road racing motorcycle on a three-dimensional closed track“, *Vehicle System Dynamics*, vol. 58, no. 8, pp. 1285–1309, 2020, DOI: 10.1080/00423114.2019.1617886.
- [69] E. Alcalá, V. Puig, J. Quevedo and U. Rosolia, „Autonomous racing using Linear Parameter Varying-Model Predictive Control (LPV-MPC)“, *Control Engineering Practice*, vol. 95, p. 104270, 2020, DOI: 10.1016/j.conengprac.2019.104270.
- [70] R. Lot and F. Biral, „A Curvilinear Abscissa Approach for the Lap Time Optimization of Racing Vehicles“, *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 7559–7565, 2014, DOI: 10.3182/20140824-6-za-1003.00868.
- [71] A. J. Tremlett and D. J. N. Limebeer, „Optimal tyre usage for a Formula One car“, *Vehicle System Dynamics*, vol. 54, no. 10, pp. 1448–1473, 2016, DOI: 10.1080/00423114.2016.1213861.
- [72] A. Jain and M. Morari, „Computing the racing line using Bayesian optimization“, in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, DOI: 10.1109/cdc42340.2020.9304147.
- [73] J. H. Jeon, R. V. Cowlagi, S. C. Peters, S. Karaman, E. Frazzoli, P. Tsiotras and K. Iagnemma, „Optimal motion planning with the half-car dynamical model for autonomous high-speed driving“, in *2013 American Control Conference*, 2013, pp. 188–193, DOI: 10.1109/ACC.2013.6579835.
- [74] A. Arab, K. Yu, J. Yi and D. Song, „Motion planning for aggressive autonomous vehicle maneuvers“, in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, 2016, pp. 221–226, DOI: 10.1109/COASE.2016.7743384.
- [75] J. Funke, P. Theodosis, R. Hindiyeh, G. Stanek, K. Kritatakirana, C. Gerdes, D. Langer, M. Hernandez, B. Müller-Bessler and B. Huhnke, „Up to the limits: Autonomous Audi TTS“, in *2012 IEEE Intelligent Vehicles Symposium*, 2012, pp. 541–547, DOI: 10.1109/IVS.2012.6232212.
- [76] T. Rizano, D. Fontanelli, L. Palopoli, L. Pallottino and P. Salaris, „Global path planning for competitive robotic cars“, in *52nd IEEE Conference on Decision and Control*, 2013, pp. 4510–4516, DOI: 10.1109/CDC.2013.6760584.
- [77] T. Rizano, D. Fontanelli, L. Palopoli, L. Pallottino and P. Salaris, „Local Motion Planning for Robotic Race Cars“, *52nd IEEE Conference on Decision and Control*, pp. 4510–4516, 2013.

- [78] S. Glaser, B. Vanholme, S. Mammar, D. Gruyer and L. Nouveliere, „Maneuver-Based Trajectory Planning for Highly Autonomous Vehicles on Real Road With Traffic and Driver Interaction“, *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 589–606, 2010, DOI: 10.1109/TITS.2010.2046037.
- [79] B. Siegler, „Lap Time Simulation for Racing Car Design“, PhD thesis, University of Leeds, 2002.
- [80] T. Völkl, „Erweiterte quasistatische Simulation zur Bestimmung des Einflusses transienten Fahrzeugverhaltens auf die Rundenzeit von Rennfahrzeugen“, PhD thesis, Technical University of Darmstadt, 2013.
- [81] B. Siegler, A. Deakin and D. Crolla, „Lap Time Simulation: Comparison of Steady State, Quasi-Static and Transient Racing Car Cornering Strategies“, in *SAE Technical Paper Series*, 2000, DOI: 10.4271/2000-01-3563.
- [82] M. Trzesniowski, *Handbuch Rennwagentechnik: Fahrwerk*, Springer Vieweg Wiesbaden, 2017, DOI: 10.1007/978-3-658-15545-2.
- [83] D. A. Doyle, G. Cunningham, G. White and J. Early, „Lap Time Simulation Tool for the Development of an Electric Formula Student Car“, in *SAE Technical Paper Series*, 2019, DOI: 10.4271/2019-01-0163.
- [84] D. Metz and D. Williams, „Near time-optimal control of racing vehicles“, *Automatica*, vol. 25, no. 6, pp. 841–857, 1989, DOI: 10.1016/0005-1098(89)90052-6.
- [85] D. Brayshaw and M. Harrison, „A quasi steady state approach to race car lap simulation in order to understand the effects of racing line and centre of gravity location“, *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 219, no. 6, pp. 725–739, 2005, DOI: 10.1243/095440705X11211.
- [86] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp and B. Lohmann, „Minimum curvature trajectory planning and control for an autonomous race car“, *Vehicle System Dynamics*, vol. 58, no. 10, pp. 1497–1527, 2020, DOI: 10.1080/00423114.2019.1631455.
- [87] B. Lenzo and V. Rossi, „A Simple Mono-Dimensional Approach for Lap Time Optimisation“, *Applied Sciences*, vol. 10, no. 4, p. 1498, 2020, DOI: 10.3390/app10041498.
- [88] S. Malcher, M. Bargende, M. Grill, U. Baretzky, H. Diel and S. Wohlgemuth, „Virtual Optimization of Race Engines Through an Extended Quasi Steady State Lap Time Simulation Approach“, in *SAE Technical Paper Series*, 2018, DOI: 10.4271/2018-01-0587.
- [89] R. P. Costa and R. Bortolussi, „Lap Time Simulation of Formula SAE Vehicle With Quasi-steady State Model“, in *SAE Technical Paper Series*, 2016, DOI: 10.4271/2016-36-0164.
- [90] D. P. Kelly, „Lap Time Simulation with Transient Vehicle and Tyre Dynamics“, PhD thesis, Cranfield University, 2008.
- [91] D. Casanova, R. Sharp and P. Symonds, „Minimum Time Manoeuvring: The Significance of Yaw Inertia“, *Vehicle System Dynamics*, vol. 34, no. 2, pp. 77–115, 2000, DOI: 10.1076/0042-3114(200008)34:2;1-g;ft077.
- [92] D. Tavernini, M. Massaro, E. Velenis, D. I. Katzourakis and R. Lot, „Minimum time cornering: the effect of road surface and car transmission layout“, *Vehicle System Dynamics*, vol. 51, no. 10, pp. 1533–1547, 2013, DOI: 10.1080/00423114.2013.813557.

- [93] D. Casanova, „On Minimum Time Vehicle Manoeuvring: The Theoretical Optimal Lap“, PhD thesis, Cranfield University, 2000.
- [94] J. Timings and D. Cole, „Robust lap-time simulation“, *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 228, no. 10, pp. 1200–1216, 2014, DOI: 10.1177/0954407013516102.
- [95] T. Völkl, M. Muehlmeier and H. Winner, „Extended Steady State Lap Time Simulation for Analyzing Transient Vehicle Behavior“, *SAE International Journal of Passenger Cars – Mechanical Systems*, vol. 6, no. 1, pp. 283–292, 2013, DOI: 10.4271/2013-01-0806.
- [96] M. Veneri and M. Massaro, „A free-trajectory quasi-steady-state optimal-control method for minimum lap-time of race vehicles“, *Vehicle System Dynamics*, vol. 58, no. 6, pp. 933–954, 2020, DOI: 10.1080/00423114.2019.1608364.
- [97] D. P. Kelly and R. S. Sharp, „Time-optimal control of the race car: influence of a thermodynamic tyre model“, *Vehicle System Dynamics*, vol. 50, no. 4, pp. 641–662, 2012, DOI: 10.1080/00423114.2011.622406.
- [98] W. J. West and D. J. N. Limebeer, „Optimal tyre management for a high-performance race car“, *Vehicle System Dynamics*, 2020, DOI: 10.1080/00423114.2020.1802047.
- [99] G. Perantoni and D. J. N. Limebeer, „Optimal Control of a Formula One Car on a Three-Dimensional Track – Part 1: Track Modeling and Identification“, *Journal of Dynamic Systems, Measurement, and Control*, vol. 137, no. 5, 2015, DOI: 10.1115/1.4028253.
- [100] D. J. N. Limebeer and G. Perantoni, „Optimal Control of a Formula One Car on a Three-Dimensional Track – Part 2: Optimal Control“, *Journal of Dynamic Systems, Measurement, and Control*, vol. 137, no. 5, 2015, DOI: 10.1115/1.4029466.
- [101] D. J. Limebeer, G. Perantoni and A. Rao, „Optimal control of Formula One car energy recovery systems“, *International Journal of Control*, vol. 87, no. 10, pp. 2065–2080, 2014, DOI: 10.1080/00207179.2014.900705.
- [102] S. Ebbesen, M. Salazar, P. Elbert, C. Bussi and C. H. Onder, „Time-optimal Control Strategies for a Hybrid Electric Race Car“, *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 233–247, 2018, DOI: 10.1109/TCST.2017.2661824.
- [103] M. Salazar, C. Balerna, E. Chisari, C. Bussi and C. H. Onder, „Equivalent Lap Time Minimization Strategies for a Hybrid Electric Race Car“, in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, DOI: 10.1109/cdc.2018.8618724.
- [104] X. Liu, A. Fotouhi and D. J. Auger, „Optimal energy management for formula-E cars with regulatory limits and thermal constraints“, *Applied Energy*, vol. 279, p. 115805, 2020, DOI: 10.1016/j.apenergy.2020.115805.
- [105] T. Herrmann, A. Wischnewski, L. Hermansdorfer, J. Betz and M. Lienkamp, „Real-Time Adaptive Velocity Optimization for Autonomous Electric Cars at the Limits of Handling“, *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 4, pp. 665–677, 2021, DOI: 10.1109/tiv.2020.3047858.
- [106] N. Dal Bianco, R. Lot and K. Matthys, „Lap time simulation and design optimisation of a brushed DC electric motorcycle for the Isle of Man TT Zero Challenge“, *Vehicle System Dynamics*, vol. 56, no. 1, pp. 27–54, 2018, DOI: 10.1080/00423114.2017.1342847.
- [107] McLaren Racing Limited, „Formula One Race Strategy“, Royal Academy of Engineering. Available: <https://www.raeng.org.uk/publications/other/14-car-racing> [visited on 01/23/2022].

- [108] F. Farroni, A. Sakhnevych and F. Timpone, „Physical modelling of tire wear for the analysis of the influence of thermal and frictional effects on vehicle performance“, *Proceedings of the Institution of Mechanical Engineers, Part L: Journal of Materials: Design and Applications*, vol. 231, no. 1-2, pp. 151–161, 2016, DOI: 10.1177/1464420716666107.
- [109] J. Bekker and W. Lotz, „Planning Formula One race strategies using discrete-event simulation“, *Journal of the Operational Research Society*, vol. 60, no. 7, pp. 952–961, 2009, DOI: 10.1057/palgrave.jors.2602626.
- [110] A. Phillips. „*Building a Race Simulator*“, 2014. Available: <https://f1metrics.wordpress.com/2014/10/03/building-a-race-simulator> [visited on 01/23/2022].
- [111] T. Salminen. „*Race Simulator*“, 2019. Available: <https://f1strategyblog.wordpress.com/2019/05/10/race-simulator-downloadable-r-program-code> [visited on 01/23/2022].
- [112] C. Sulsters, „Simulating Formula One Race Strategies“, Research Paper MSc Business Analytics, Vrije Universiteit Amsterdam, 2018.
- [113] C. Lemieux, *Monte Carlo and Quasi-Monte Carlo Sampling*, (Springer Series in Statistics), Springer New York, 2009, DOI: 10.1007/978-0-387-78165-5.
- [114] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, Second, O’Reilly Media, Inc., 2019, ISBN: 9781492032649.
- [115] C. K. Leung and K. W. Joseph, „Sports Data Mining: Predicting Results for the College Football Games“, *Procedia Computer Science*, vol. 35, pp. 710–719, 2014, DOI: 10.1016/j.procs.2014.08.153.
- [116] M. Purucker, „Neural network quarterbacking“, *IEEE Potentials*, vol. 15, no. 3, pp. 9–15, 1996, DOI: 10.1109/45.535226.
- [117] D. Delen, D. Cogdell and N. Kasap, „A comparative analysis of data mining methods in predicting NCAA bowl outcomes“, *International Journal of Forecasting*, vol. 28, no. 2, pp. 543–552, 2012, DOI: 10.1016/j.ijforecast.2011.05.002.
- [118] J. Kahn, „Neural Network Prediction of NFL Football Games“, *World Wide Web Electronic Publication*, 2003.
- [119] H. Chen, P. B. Rinde, L. She, S. Sutjahjo, C. Sommer and D. Neely, „Expert prediction, symbolic learning, and neural networks. An experiment on greyhound racing“, *IEEE Expert*, vol. 9, no. 6, pp. 21–27, 1994, DOI: 10.1109/64.363260.
- [120] U. Johansson and C. Sonstrod, „Neural networks mine for gold at the greyhound race-track“, in *Proceedings of the International Joint Conference on Neural Networks, 2003*, 2003, DOI: 10.1109/ijcnn.2003.1223680.
- [121] R. P. Schumaker and J. W. Johnson, „An Investigation of SVM Regression to Predict Longshot Greyhound Races“, *Communications of the IIMA*, vol. 8, no. 2, pp. 67–82, 2008.
- [122] J. Williams and Y. Li, „A Case Study Using Neural Networks Algorithms: Horse Racing Predictions In Jamaica“, in *Proceedings of the 2008 International Conference on Artificial Intelligence*, 2008, pp. 16–22.
- [123] E. Davoodi and A. R. Khanteymooori, „Horse racing prediction using artificial neural networks“, in *Proceedings of the 11th WSEAS International Conference on Neural Networks and 11th WSEAS International Conference on Evolutionary Computing and 11th WSEAS International Conference on Fuzzy Systems*, 2010, pp. 155–160.

- [124] D. A. Harville, „Assigning Probabilities to the Outcomes of Multi-Entry Competitions“, *Journal of the American Statistical Association*, vol. 68, no. 342, pp. 312–316, 1973, DOI: 10.1080/01621459.1973.10482425.
- [125] N. Tax and Y. Joustra. „Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach“, 2015. DOI: 10.13140/RG.2.1.1383.4729.
- [126] D. Prasetio and D. Harlili, „Predicting football match results with logistic regression“, in *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, 2016, DOI: 10.1109/icaicta.2016.7803111.
- [127] S. M. Arabzad, M. Araghi, S.-N. Soheil and N. Ghofrani, „Football Match Results Prediction Using Artificial Neural Networks; The Case of Iran Pro League“, *International Journal of Applied Research on Industrial Engineering*, vol. 1, pp. 159–179, 2014.
- [128] A. Joseph, N. Fenton and M. Neil, „Predicting football results using Bayesian nets and other machine learning techniques“, *Knowledge-Based Systems*, vol. 19, no. 7, pp. 544–553, 2006, DOI: 10.1016/j.knosys.2006.04.011.
- [129] R. Rein and D. Memmert, „Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science“, *SpringerPlus*, vol. 5, no. 1, 2016, DOI: 10.1186/s40064-016-3108-2.
- [130] J. Edelmann-Nusser, A. Hohmann and B. Henneberg, „Modeling and prediction of competitive performance in swimming upon neural networks“, *European Journal of Sport Science*, vol. 2, no. 2, pp. 1–10, 2002, DOI: 10.1080/17461390200072201.
- [131] D. Miljkovic, L. Gajic, A. Kovacevic and Z. Konjovic, „The use of data mining for basketball matches outcomes prediction“, in *IEEE 8th International Symposium on Intelligent Systems and Informatics*, 2010, DOI: 10.1109/sisy.2010.5647440.
- [132] K. Przednowek, J. Iskra and K. H. Przednowek, „Predictive Modeling in 400-Metres Hurdles Races“, in *Proceedings of the 2nd International Congress on Sports Sciences Research and Technology Support*, 2014, DOI: 10.5220/0005082201370144.
- [133] A. Maszczyk, A. Gołaś, P. Pietraszewski, R. Roczniok, A. Zając and A. Stanula, „Application of Neural and Regression Models in Sports Results Prediction“, *Procedia – Social and Behavioral Sciences*, vol. 117, pp. 482–487, 2014, DOI: 10.1016/j.sbspro.2014.02.249.
- [134] A. Pretorius and D. A. Parry, „Human Decision Making and Artificial Intelligence“, in *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists – SAICSIT 2016*, 2016, DOI: 10.1145/2987491.2987493.
- [135] F. Tagliaferri, A. Philpott, I. Viola and R. Flay, „On risk attitude and optimal yacht racing tactics“, *Ocean Engineering*, vol. 90, pp. 149–154, 2014, DOI: 10.1016/j.oceaneng.2014.07.020.
- [136] A. McCabe and J. Trevathan, „Artificial Intelligence in Sports Prediction“, in *Fifth International Conference on Information Technology: New Generations (itng 2008)*, 2008, DOI: 10.1109/itng.2008.203.
- [137] A. Dubbs, „Statistics-free sports prediction“, *Model Assisted Statistics and Applications*, vol. 13, no. 2, pp. 173–181, 2018, DOI: 10.3233/MAS-180428.
- [138] T. Graves, C. S. Reese and M. Fitzgerald, „Hierarchical Models for Permutations“, *Journal of the American Statistical Association*, vol. 98, no. 462, pp. 282–291, 2003, DOI: 10.1198/016214503000053.

- [139] C. B. Pfitzner and T. D. Rishel, „Do Reliable Predictors Exist for the Outcomes of NASCAR Races?“, *The Sport Journal*, 2008.
- [140] C. A. Depken and L. Mackey, „Driver Success in the Nascar Sprint Cup Series: The Impact of Multi-Car Teams“, *SSRN Electronic Journal*, 2009, DOI: 10.2139/ssrn.1442015.
- [141] M. Allender, „Predicting The Outcome Of NASCAR Races: The Role Of Driver Experience“, *Journal of Business & Economics Research (JBER)*, vol. 6, no. 3, 2011, DOI: 10.19030/jber.v6i3.2403.
- [142] E. Stoppels, „Predicting Race Results using Artificial Neural Networks“, Master thesis, University of Twente, 2017.
- [143] A. Stergioudis. „*Machine-learning Based F1 Race Prediction Engine*“, 2017. Available: <https://www.f1-predictor.com> [visited on 01/23/2022].
- [144] G. Gartheeban and J. Gutttag, „A Data-Driven Method for in-Game Decision Making in MLB: When to Pull a Starting Pitcher“, in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining – KDD '13*, 2013, pp. 973–979, DOI: 10.1145/2487575.2487660.
- [145] G. Gartheeban and J. Gutttag, „Predicting the Next Pitch“, in *MIT Sloan Sports Analytics Conference 2012*, 2012.
- [146] M. Bailey and S. Clarke, „Predicting the Match Outcome in One Day International Cricket Matches, while the Game is in Progress“, *Journal of sports science & medicine*, vol. 5, pp. 480–487, 2006.
- [147] V. V. Sankaranarayanan, J. Sattar and L. V. S. Lakshmanan, „Auto-play: A Data Mining Approach to ODI Cricket Simulation and Prediction“, in *Proceedings of the 2014 SIAM International Conference on Data Mining*, 2014, pp. 1064–1072, DOI: 10.1137/1.9781611973440.121.
- [148] B. G. Weber and M. Mateas, „A data mining approach to strategy prediction“, in *2009 IEEE Symposium on Computational Intelligence and Games*, 2009, DOI: 10.1109/cig.2009.5286483.
- [149] T. Tulabandhula and C. Rudin, „Tire Changes, Fresh Air, and Yellow Flags: Challenges in Predictive Analytics for Professional Racing“, *Big Data*, vol. 2, no. 2, pp. 97–112, 2014, DOI: 10.1089/big.2014.0018.
- [150] C. L. W. Choo, „Real-Time Decision Making in Motorsports: Analytics for Improving Professional Car Race Strategy“, Master thesis, Massachusetts Institute of Technology, 2015.
- [151] X. Liu and A. Fotouhi, „Formula-E race strategy development using artificial neural networks and Monte Carlo tree search“, *Neural Computing and Applications*, vol. 32, no. 18, pp. 15191–15207, 2020, DOI: 10.1007/s00521-020-04871-1.
- [152] P. Aversa, L. Cabantous and S. Haefliger, „When decision support systems fail: Insights for strategic information systems from Formula 1“, *The Journal of Strategic Information Systems*, vol. 27, no. 3, pp. 221–236, 2018, DOI: 10.1016/j.jsis.2018.03.002.
- [153] Amazon Web Services, Inc. „*F1 Insights powered by AWS*“, Available: <https://aws.amazon.com/de/f1> [visited on 01/23/2022].
- [154] A. Maiza. „*Reinforcement Learning for Formula 1 Race Strategy*“, June 2020. Available: <https://towardsdatascience.com/reinforcement-learning-for-formula-1-race-strategy-7f29c966472a> [visited on 01/23/2022].

-
- [155] OpenStreetMap Foundation. „*OpenStreetMap*“, Available: <https://www.openstreetmap.org> [visited on 01/23/2022].
- [156] A. de Paula Suiti, „Creation of Race Track Models Based on Online Map Services“, Semester thesis, Technical University of Munich, 2020.
- [157] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Fourth, Pearson, 2018, ISBN: 9781292223049.
- [158] A. Heilmeier. „*Race Track Database*“, Available: <https://github.com/TUMFTM/racetrack-database>.
- [159] A. Heilmeier. „*Formula 1 Timing Database 2014 – 2019*“, Available: <https://github.com/TUMFTM/f1-timing-database>.
- [160] C. Newell. „*Ergast Motor Racing Developer API*“, Available: <http://ergast.com/mrd> [visited on 01/23/2022].
- [161] Formula One World Championship Limited. „*F1 TV*“, Available: <https://f1tv.formula1.com>.
- [162] Pirelli Motorsport. „*Twitter channel of Pirelli Motorsport*“, Available: <https://twitter.com/pirellisport>.
- [163] A. Heilmeier, T. Herrmann, T. Stahl, L. Hermansdorfer and F. Christ. „*Global Race Trajectory Optimization*“, Available: https://github.com/TUMFTM/global_racetrajectory_optimization.
- [164] T. Stahl, A. Wischnewski, J. Betz and M. Lienkamp, „Multilayer Graph-Based Trajectory Planning for Race Vehicles in Dynamic Scenarios“, in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3149–3154.
- [165] A. Heilmeier. „*Lap Time Simulation*“, Available: <https://github.com/TUMFTM/laptime-simulation>.
- [166] M. Geißlinger, „Development of a Lap Time Simulation“, Semester thesis, Technical University of Munich, 2018.
- [167] A. Heilmeier. „*Race Simulation*“, Available: <https://github.com/TUMFTM/race-simulation>.
- [168] F. Bayer, „Development of a Method for the Automated Determination of the Parameters of a Race Simulation“, Semester thesis, Technical University of Munich, 2019.
- [169] S. Stahn, „Development of a Model to Simulate Overtaking Maneuvers on Racetracks“, Semester thesis, Technical University of Munich, 2019.
- [170] L. Müller, „Development of a Model for the Behavior of Tires over the Course of a Race“, Master thesis, Technical University of Munich, 2018.
- [171] M. Faist, „Data Analysis and Robust Parameter Determination for a Race Simulation“, Master thesis, Technical University of Munich, 2020.
- [172] Fédération Internationale de l'Automobile. „*2016 F1 Sporting Regulations*“, 2016.
- [173] Fédération Internationale de l'Automobile. „*2018 F1 Sporting Regulations*“, 2018.
- [174] L. Rohbogner, „Extension of a Race Simulation using Monte Carlo Methods“, Semester thesis, Technical University of Munich, 2019.
- [175] A. Thomaser, „Automation of Racing Strategy Decisions with Neural Networks“, Semester thesis, Technical University of Munich, 2020.
- [176] A. Thomaser, „Automation of Racing Strategy Decisions with Reinforcement Learning“, Master thesis, Technical University of Munich, 2020.

- [177] R. S. Sutton and A. G. Barto, *Reinforcement Learning – An Introduction*, Second, The MIT Press, 2018, ISBN: 9780262039246.
- [178] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller. „*Playing Atari with Deep Reinforcement Learning*“, 2013. Available: <http://arxiv.org/abs/1312.5602>.
- [179] Formula One World Championship Limited. „*2017 Chinese Grand Prix | Lewis Hamilton Onboard Pole Lap*“, 2017. Available: <https://www.youtube.com/watch?v=yx6BbFTNyy0> [visited on 01/23/2022].
- [180] Formula One World Championship Limited. „*F1's 'Party mode' ban – What are the changes to engine modes and why do they matter?*“, 2020. Available: <https://www.formula1.com/en/latest/article.qualifying-engine-modes-what-are-the-proposed-changes-and-why-do-they-matter.OI3cSGqWS6mONcBHUFncL.html> [visited on 01/23/2022].
- [181] M. Youson. „*The insider's guide to... F1 strategists*“, Jan. 2020. Available: <https://www.formula1.com/en/latest/article.watch-the-insiders-guide-to-f1-strategists.6VT3445ugVd0bh85Pw5sIO.html> [visited on 01/23/2022].
- [182] A. Heilmeier. „*Time-Discrete Race Simulator*“, Available: <https://github.com/heilmeiera/time-discrete-race-simulator>.
- [183] D.A.T.A.S. Ltd. „*RaceSim*“, Available: <http://www.datas-ltd.com> [visited on 01/23/2022].
- [184] T. Salminen. „*Game Theory Simulation Model on Race Strategy Decisions*“, 2016. Available: <https://f1strategyblog.wordpress.com/2016/10/07/game-theory-simulation-model-on-race-strategy-planning> [visited on 01/23/2022].
- [185] D. Piccinotti, A. Likmeta, N. Brunello and M. Restelli, „Online Planning for F1 Race Strategy Identification“, in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2021. Available: https://prl-theworkshop.github.io/prl2021/papers/PRL2021_paper_1.pdf.

Prior Publications

During the development of this dissertation, publications and student theses were written in which partial aspects of this work were presented.

Journals; Scopus/Web of Science listed (peer-reviewed)

- [12] A. Heilmeier, M. Graf, J. Betz and M. Lienkamp, „Application of Monte Carlo Methods to Consider Probabilistic Effects in a Race Simulation for Circuit Motorsport“, *Applied Sciences*, vol. 10, no. 12, p. 4229, 2020, DOI: 10.3390/app10124229.
- [13] A. Heilmeier, A. Thomaser, M. Graf and J. Betz, „Virtual Strategy Engineer: Using Artificial Neural Networks for Making Race Strategy Decisions in Circuit Motorsport“, *Applied Sciences*, vol. 10, no. 21, p. 7805, 2020, DOI: 10.3390/app10217805.
- [60] F. Christ, A. Wischnewski, A. Heilmeier and B. Lohmann, „Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients“, *Vehicle System Dynamics*, vol. 59, no. 4, pp. 588–612, 2021, DOI: 10.1080/00423114.2019.1704804.
- [86] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp and B. Lohmann, „Minimum curvature trajectory planning and control for an autonomous race car“, *Vehicle System Dynamics*, vol. 58, no. 10, pp. 1497–1527, 2020, DOI: 10.1080/00423114.2019.1631455.

Conferences, Periodicals; Scopus/Web of Science listed (peer-reviewed)

- [8] A. Heilmeier, M. Graf and M. Lienkamp, „A Race Simulation for Strategy Decisions in Circuit Motorsports“, in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2986–2993, DOI: 10.1109/ITSC.2018.8570012.
- [9] A. Heilmeier, M. Geisslinger and J. Betz, „A quasi-steady-state lap time simulation for electrified race cars“, in *2019 Fourteenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*, 2019, DOI: 10.1109/EVER.2019.8813646.

Non-thesis-relevant publications; Scopus/Web of Science listed (peer-reviewed)

J. Betz, A. Wischnewski, A. Heilmeier, F. Nobis, T. Stahl, L. Hermansdorfer and M. Lienkamp, „A software architecture for an autonomous racecar“, in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, 2019, DOI: 10.1109/VTCSpring.2019.8746367.

J. Betz, A. Wischnewski, A. Heilmeier, F. Nobis, L. Hermansdorfer, T. Stahl, T. Herrmann and M. Lienkamp, „A software architecture for the dynamic path planning of an autonomous racecar at the limits of handling“, in *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, 2019, DOI: 10.1109/ICCVE45908.2019.8965238.

J. Betz, A. Heilmeier, A. Wischnewski, T. Stahl and M. Lienkamp, „Autonomous Driving – A Crash Explained in Detail“, *Applied Sciences*, vol. 9, no. 23, p. 5126, 2019, DOI: 10.3390/app9235126.

Thesis-relevant open-source software

- [158] A. Heilmeier. „*Race Track Database*“, Available: <https://github.com/TUMFTM/racetrack-database>.
- [159] A. Heilmeier. „*Formula 1 Timing Database 2014 – 2019*“, Available: <https://github.com/TUMFTM/f1-timing-database>.
- [163] A. Heilmeier, T. Herrmann, T. Stahl, L. Hermansdorfer and F. Christ. „*Global Race Trajectory Optimization*“, Available: https://github.com/TUMFTM/global_racetrajectory_optimization.
- [165] A. Heilmeier. „*Lap Time Simulation*“, Available: <https://github.com/TUMFTM/laptime-simulation>.
- [167] A. Heilmeier. „*Race Simulation*“, Available: <https://github.com/TUMFTM/race-simulation>.
- [182] A. Heilmeier. „*Time-Discrete Race Simulator*“, Available: <https://github.com/heilmeiera/time-discrete-race-simulator>.

Supervised Student Theses

The following student theses were written within the framework of the dissertation under the supervision of the author in terms of content, technical and scientific support as well as under relevant guidance of the author. In the following, the bachelor, semester and master theses relevant and related to this dissertation are listed. Many thanks to the authors of these theses for their extensive support within the framework of this research project.

- [156] A. de Paula Suti, „Creation of Race Track Models Based on Online Map Services“, Semester thesis, Technical University of Munich, 2020.
 - [166] M. Geißlinger, „Development of a Lap Time Simulation“, Semester thesis, Technical University of Munich, 2018.
 - [168] F. Bayer, „Development of a Method for the Automated Determination of the Parameters of a Race Simulation“, Semester thesis, Technical University of Munich, 2019.
 - [169] S. Stahn, „Development of a Model to Simulate Overtaking Maneuvers on Racetracks“, Semester thesis, Technical University of Munich, 2019.
 - [170] L. Müller, „Development of a Model for the Behavior of Tires over the Course of a Race“, Master thesis, Technical University of Munich, 2018.
 - [171] M. Faist, „Data Analysis and Robust Parameter Determination for a Race Simulation“, Master thesis, Technical University of Munich, 2020.
 - [174] L. Rohbogner, „Extension of a Race Simulation using Monte Carlo Methods“, Semester thesis, Technical University of Munich, 2019.
 - [175] A. Thomaser, „Automation of Racing Strategy Decisions with Neural Networks“, Semester thesis, Technical University of Munich, 2020.
 - [176] A. Thomaser, „Automation of Racing Strategy Decisions with Reinforcement Learning“, Master thesis, Technical University of Munich, 2020.
- M. Wu, „Analysis of Race Strategies in Motorsport Racing Series“, Master thesis, Technical University of Munich, 2017.
- P. Enzinger, „Simulative Optimization of Race Strategies“, Semester thesis, Technical University of Munich, 2019.