Optimizing Sets of Solutions for Controlling Constrained Nonlinear Systems

Bastian Schürmann and Matthias Althoff

***Abstract*— We present a novel control approach for formally solving reach-avoid problems for constrained and disturbed nonlinear systems by optimizing over reachable sets. Reach-avoid problems arise in many modern control applications of safety-critical systems, such as autonomous driving and human-robot collaboration. They require us to control all states from a set of initial states in finite time into a final set while guaranteeing the satisfaction of state and input constraints despite the presence of disturbances, both external and in the form of uncertain measurements. We optimize over reachable sets, thereby simultaneously improving the control performance and guaranteeing constrain satisfaction for all solutions. Moreover, our new approach involves a novel combination of state-dependent feedforward and feedback control, which leads to better control performance compared to existing approaches as demonstrated in several numerical examples. Our algorithm is particularly suited for computing motion primitives used in maneuver automata, which realizes fast and efficient online planning. The online applicability is supported by the simple structure of the resulting controller which is a time-varying linear tracking controller.**

***Index Terms*— Set-based control, reach-avoid problems, nonlinear systems, disturbed systems, constrained systems, reachability analysis, optimization**

I. Introduction

In many modern control applications, such as autonomous driving or robots collaborating with humans, guaranteeing safety becomes increasingly important. This is a hard task since these systems have complex, nonlinear dynamics, are limited by state and input constraints, and are affected by uncontrollable inputs such as sensor noise or external disturbances.

These kinds of problems can be seen as reach-avoid problems, where the system has to be controlled in a fixed time from some initial set towards a desired goal set, while avoiding unsafe regions despite disturbances and constraints. For example, a car should be steered from its initial position to a goal position while avoiding unsafe regions, such as other lanes, other traffic participants, or velocities that are too high. At the same time, the input set of the controller is limited by maximum acceleration, maximum braking, and maximum steering capabilities. The same is true if we consider a robotic manipulator that should move from one configuration to another, while satisfying torque constraints and not colliding with human co-workers, surrounding objects, or itself.

The authors are with the Department of Informatics, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany (email: bastian.schuermann@tum.de, althoff@tum.de).

For a single initial state and undisturbed systems, these reach-avoid problems can be solved efficiently using numerical optimization tools such as multiple shooting [1]. However, if we have to consider a set of initial states, e.g., if the exact initial state is not known beforehand or cannot exactly be measured due to noisy sensors, these methods fail, as they would have to solve an optimization problem for each of the uncountably many states of a continuous set.

In this paper, we present a new way of computing controllers which formally solve reach-avoid problems for a set of solutions despite disturbances that can be efficiently executed. In contrast to classical optimal control algorithms, which only consider a single trajectory and therefore are able to numerically integrate this trajectory in the cost and constraint functions, we optimize directly over the whole reachable set [2]–[4] and include the reachable set computation directly in the cost and constraint functions.

As the computations of reachable sets require some time, we significantly reduce the online computation time by pre-computing the controllers and corresponding reachable sets a priori and storing them in a safe maneuver automaton [5]; see Fig. 1. There, many short trajectory pieces with different initial and final sets are pre-computed, which are often referred to as motion primitives (Fig. 1, ①). These motion primitives can be concatenated if the initial set of a motion primitive is completely contained in the final set of the previous one (Fig. 1, ②). The maneuver automaton contains the motion primitives as states and connectable motion primitives as transitions, as presented in Fig. 1, ②. After pre-computing all of the motion primitives and constructing the maneuver automaton offline in advance, the online planning and control problem simplifies to a discrete planning problem (Fig. 1, ③–⑥). In order to have a maneuver automaton with as many transitions as possible – to be more flexible during the online planning – we need controllers which are able to steer all states from a large initial set into a small final set.

Related Work

There exist many methods in the literature which aim to control constrained nonlinear systems despite disturbances. However, it is still an open challenge to provide guarantees without becoming too conservative or resulting in too large computation times. A direct way to obtain optimal control inputs, which satisfy state and input constraints, is by solving the

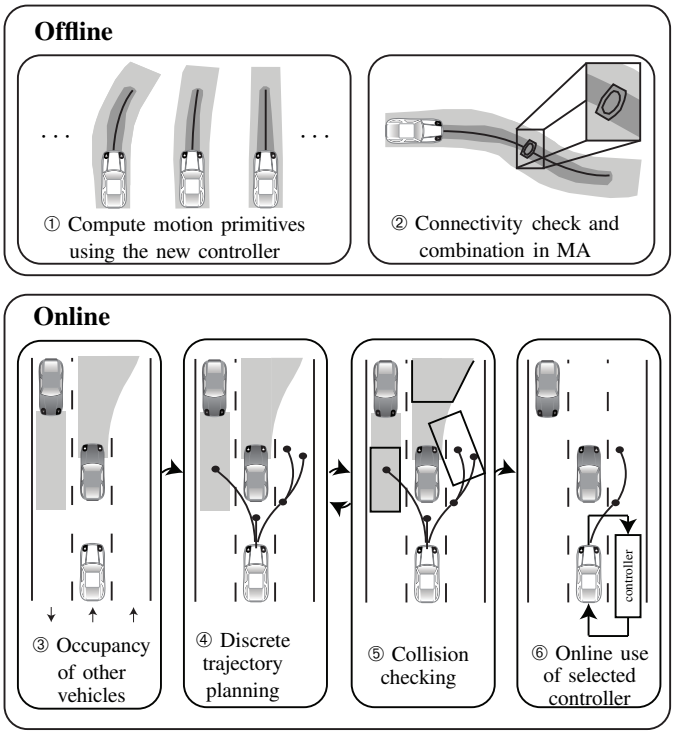


Fig. 1. Overview of robust maneuver automata (MA) design for an example in automated driving using our new control approach.

Hamilton-Jacobi-Bellman (HJB) equation or using dynamic programming [6]–[9]. Since solving the HJB equation is only possible for low-dimensional systems, it becomes difficult to apply for high-dimensional systems.

Another well-known method is to use model predictive control (MPC), which allows us to take state and input constraints into account. There are two different approaches: In the explicit case, the state space is divided into different regions, and for each region, an optimal control law is computed which satisfies the constraints [10]–[17]. Due to this division of the state space, this approach becomes easily computationally infeasible for higher-dimensional systems. In implicit MPC, on the other hand, the optimization problem is solved online, requiring to find a solution in real time, which is often infeasible if formal guarantees are required.

An established form of robust MPC is tube-based MPC, where an auxiliary controller is used to keep the system in a tube around the optimized trajectory [18]–[20]; for linear systems, the superposition principle is exploited, but it is also possible for nonlinear systems [21]–[24] under certain assumptions and with increased conservativeness, as explained in Sec. III and illustrated in Fig. 4. An approach that combines MPC with reachability analysis is shown in [25]; however, they also consider fixed feedback controllers and use conservative approximations of the reachable sets.

Another method which controls all states around a single trajectory is presented in [26]. This method exploits so-called trajectory robustness, which is, however, restricted to feedback-linearizable and differentially flat systems, and it does not take disturbances into account. An extended version using feedback control is shown in [27].

Controlling all states in so-called funnels, similar to the tubes in tube-based MPC, to a final set for disturbed nonlinear systems can be done using LQR trees [28]–[30]. There, the authors use sums-of-squares programming to find LQR tracking controllers, which satisfy certain constraints. This method is also successfully applied to safe maneuver automata [31]; however, the complexity of sums-of-squares techniques can grow very fast with the system dimension. Similar approaches which can be used for motion planning are shown in [32], where the positive invariant sets of different controllers are concatenated in trees to obtain maneuver automata. More approaches for safe motion planning using maneuver automata can also be found in [33], [34] and using reachability analysis for verifying the safety of maneuvers in [35].

A large class of controllers which provide formal guarantees are abstraction-based controllers [36]–[52]. They have the advantage that they can even take complex specifications into account; however, as they often do this by discretizing the state and input spaces, they suffer from the curse of dimensionality. Some methods try to avoid computing abstractions of the whole state space [53]–[57], but mainly focus on the satisfaction of more complex requirements and only consider undisturbed systems. Other approaches for controller synthesis with formal specification (again, only for undisturbed systems) combine genetic algorithms with Lyapunov functions [58].

In earlier works, we addressed the problem at hand by combining optimal control with reachability analysis. In [59], [60], we obtained a piecewise-constant control law by interpolating optimal open-loop trajectories of extreme states and generators, respectively. In [61], we optimized continuous feedback controllers, but only for disturbed linear systems.

Contributions

In this paper, we combine formal verification with controller synthesis. Instead of treating both steps – controller synthesis and verification – independently, we combine them by optimizing the controller to obtain a tighter reachable set, which in turn is used to verify the controller. This allows us to obtain controllers with optimized performance and safety guarantees despite constraints, external disturbances, and measurement noise. The resulting approach is easy to apply and scales well with the system dimension, and the resulting controller has a simple structure, therefore allowing fast sampling times during its application. We develop a novel unification and extension of the feedforward control from [60] and the closed-loop feedback optimization from [61], which leads to improved performance and faster computation times compared to the previous works. The resulting controller includes a state-dependent feedforward controller, which provides a unique reference trajectory for any state from the initial set. We track them with a continuous feedback controller, which is obtained by optimizing over reachable sets.

Organization

The remainder of this paper is organized as follows. After a formal problem statement in Sec. II, we give an overview in Sec. III. The main part of this paper is Sec. IV, where

we present the new approach in detail. This is followed by a discussion of the algorithm in Sec. V. The applicability of the algorithm is shown in four numerical examples in Sec. VI, where we compare it to existing approaches, before we conclude with a summary in Sec. VII.

II. Formal Problem Statement

We consider a disturbed, nonlinear, time-continuous system of the form

$$\dot{x}(t) = f(x(t), u(t), w(t)), \tag{1}$$

with states $x(t) \in \mathbb{R}^n$, inputs $u(t) \in \mathbb{R}^m$, and disturbances $w(t) \in \mathcal{W} \subset \mathbb{R}^d$ ($\mathcal{W}$ is compact, i.e., closed and bounded). We do not require any stochastic properties for $w(\cdot)$; we only assume that any possible disturbance trajectory is bounded at any point in time in the compact set $\mathcal{W}$. We denote this by $w(\cdot) \in \mathcal{W}$, which is a shorthand for $w(t) \in \mathcal{W}, \forall t \in [0, t_f]$, where $t_f \in \mathbb{R}_0^+$ is the final time. The same shorthand is also used for state and input trajectories throughout the paper. We denote the solution of (1) with initial state $x(0)$, input $u(\cdot)$, and disturbance $w(\cdot)$ at time t as $\xi(x(0), u(\cdot), w(\cdot), t)$. The solution satisfies the following two properties:

$$\xi(x(0), u(\cdot), w(\cdot), 0) = x(0),$$
$$\dot{\xi}(x(0), u(\cdot), w(\cdot), t) = f\Big(\xi\big(x(0), u(\cdot), w(\cdot), t\big), u(t), w(t)\Big),$$
$$\forall t \in \mathbb{R}_0^+.$$

If we consider an undisturbed system, we use $\xi(x(0), u(\cdot), 0, t)$ to denote the solution without disturbances, i.e., $\mathcal{W} = 0$. The measurement of the system is modeled by a function h, returning the measurement vector $\hat{x} \in \mathbb{R}^n$ subject to a compact set of measurement errors $\mathcal{V} \subset \mathbb{R}^o$:

$$\hat{x} \in \{h(x, \nu) | \nu \in \mathcal{V}\}.$$

The task is to find a control law $u_{ctrl}(\hat{x}, t)$ for system (1) which guarantees that all states in an initial set $\mathcal{X}_0 \subset \mathbb{R}^n$ are steered into a final set $\mathcal{S}_f \subset \mathbb{R}^n$ around an end state $x^{(f)}$ after time t_f despite disturbances and measurement errors. We minimize the size of the final set by solving

$$\min_{u_{ctrl}} \max_{x \in \mathcal{S}_f} \|x - x^{(f)}\|_1. \tag{2}$$

In addition to its numerical advantages, the reason for using the ℓ_1 norm for our cost function is that many target sets are axis-aligned boxes. Therefore, we prefer a set whose bounding axis-aligned box is as small as possible. When using the ℓ_1 norm, we minimize the sum of the side lengths of this bounding box. If we used, for example, the ℓ_2 norm instead, we would minimize the sum of the squared sides, therefore not appropriately considering smaller dimensions. The extreme case would be using the infinity norm, as illustrated with the following example: Let us consider a two-dimensional set whose maximum expanse in the first dimension is denoted by s_1 and in the second dimension by s_2, with $s_1 \geq s_2$. In the case that there exist constraints which prevent us from reducing the size of s_1, there is no change in the ℓ_∞ norm if we reduce the size of s_2, while any cost function using the ℓ_1 norm would actually benefit from reducing s_2.

Furthermore, we consider convex constraints on the states and inputs, i.e.,

$$\xi(x(0), u(\cdot), w(\cdot), t) \in \mathcal{X}, \quad \forall t \in [0, t_f],$$
$$u(t) \in \mathcal{U}, \quad \forall t \in [0, t_f],$$

where $\mathcal{X}$ and $\mathcal{U}$ are both convex sets in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively.

Note that during offline computation, the locations of most non-convex constraints, such as other traffic participants in automated driving, are not known. Therefore, we use this approach to compute the maneuvers offline in advance, while taking convex input constraints, e.g., maximum acceleration or steering, and convex state constraints, e.g., maximum velocity, into account. The non-convex dynamic constraints are handled during the online planning using motion primitives (see Fig. 2), which is a standard approach and can be done using existing techniques (see e.g., [62]).

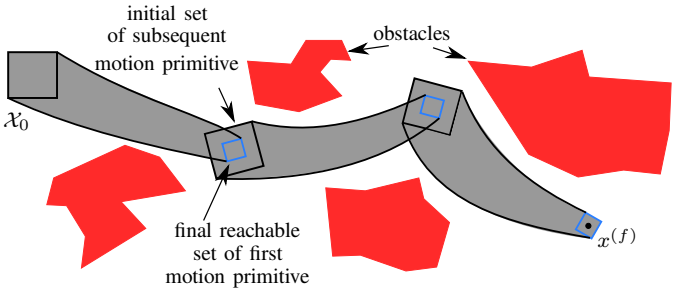


Fig. 2. Using motion primitives (gray) to steer all states from $\boldsymbol{\mathcal{X}_0}$ to $\boldsymbol{x^{(f)}}$ while avoiding nonconvex obstacles (red). The final set (blue box) of one motion primitive is always contained inside the initial set (black box) of the following motion primitive.

For most of the paper, we want to find a final set $\mathcal{S}_f$ which is as small as possible. If instead the task is to steer all states into a given final set, then we would have to adapt our algorithms by adding this as an additional constraint. In this case, however, it might be possible that no solution exists, depending on the choice of constraints, final time, and final set.

III. Overview

Many classical control schemes tackle the reach-avoid problem in Sec. II with a feedforward reference trajectory and a feedback controller which tracks this reference trajectory [63]:

$$u_{classic}(\hat{x}, t) = u_{ff}(t) + u_{fb}(\hat{x}(t), t).$$

While this works well if the actual trajectory starts close to the reference trajectory or if there are no input constraints, this approach struggles with satisfying the input constraints and providing a good performance for states further away from the reference trajectory. To balance performance and constraint satisfaction, we propose the idea illustrated in Fig. 3: Instead of a single feedforward trajectory, we introduce a state-dependent feedforward controller $u_{ff}(\hat{x}(0), t)$ together with a feedback controller $u_{fb}(\hat{x}(t), t)$:

$$u_{ctrl}(\hat{x}, t) = u_{ff}(\hat{x}(0), t) + u_{fb}(\hat{x}(t), t).$$

We thereby obtain a unique reference trajectory for each of the infinitely many initial states. The feedback controller therefore

only has to counteract the disturbances, which allows it to be much more aggressive without violating input constraints.

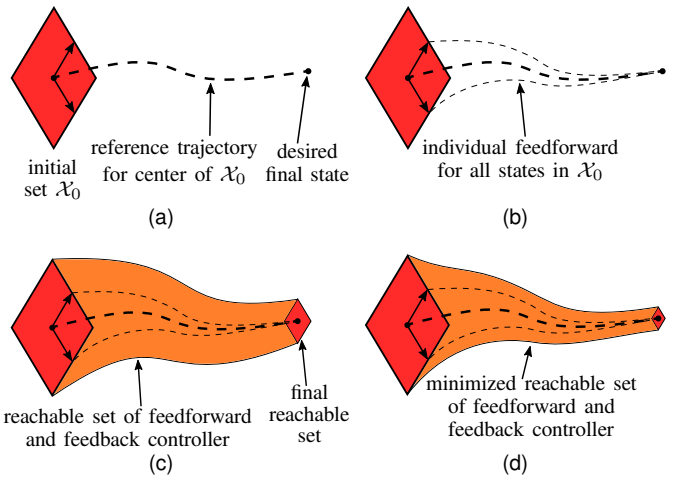


Fig. 3. Our combined control approach: Computing a reference trajectory from the center of the initial set to the desired final state (a). Computing the feedforward controller for each state of the initial set, which steers them to the origin (b). Counteracting disturbances with feedback controller; minimizing over the reachable set by optimizing the controller parameters (c). Obtaining state-dependent feedforward and feedback controllers with reachable sets of reduced size (d).

Since we compute the different parts – reference trajectory, state-dependent feedforward for all states, and feedback controller – after each other, we have to ensure enough remaining input capacities and a sufficient distance to constraints. Therefore, we have to tighten the constraints for the reference trajectory and for the state-dependent feedforward controller, respectively. Let us introduce the new constraint sets $\mathcal{X}_{ref}, \mathcal{X}_{ff}$ and $\mathcal{U}_{ref}, \mathcal{U}_{ff}$ such that

$$\mathcal{X}_{ref} \subseteq \mathcal{X}_{ff} \subseteq \mathcal{X}, \tag{3}$$
$$\mathcal{U}_{ref} \subseteq \mathcal{U}_{ff} \subseteq \mathcal{U}. \tag{4}$$

In previous works, such as [18], the above sets are exclusively reserved, resulting in the distribution of input utilization illustrated in Fig. 4. In contrast, we introduce a flexible and therefore less conservative distribution, shown in Fig. 5. We minimize the applied inputs in an optimization problem and are able to use the unused capacities of previous steps in contrast to the previous example.

The choice of $\mathcal{U}_{ref}$ and $\mathcal{U}_{ff}$ are design choices. As the computations for the reference trajectory and the feedforward controller can be done very fast, especially compared to the optimization of the feedback controller, we could also minimize over the allowed inputs to find the smallest input set such that all states are controlled close enough to the desired final state for the undisturbed system.

Remark 1: To accelerate the proposed optimization procedure, we initially simplify the applied reachability analysis so that it is not over-approximative and we sometimes use linearized and time-discretized dynamics during our controller synthesis. Since we use over-approximative reachability analysis in the end, we ensure that our results are formally correct for the constrained nonlinear dynamics despite disturbances.

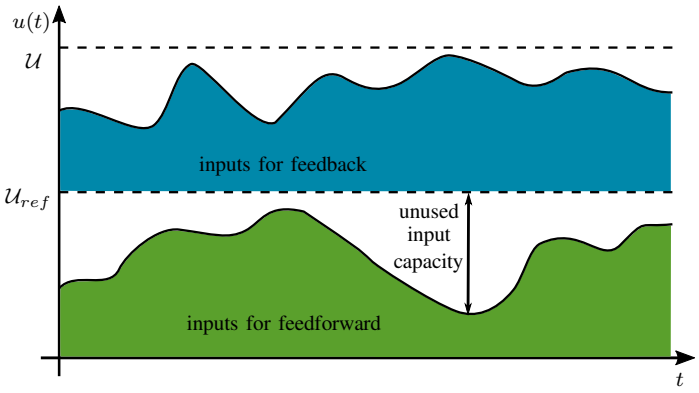


Fig. 4. Illustration of a classical trajectory tracking approach where the reference trajectory and the feedback controller are designed independently. Therefore, the controller cannot benefit from the fact that the reference trajectory does not use the full input capacities.

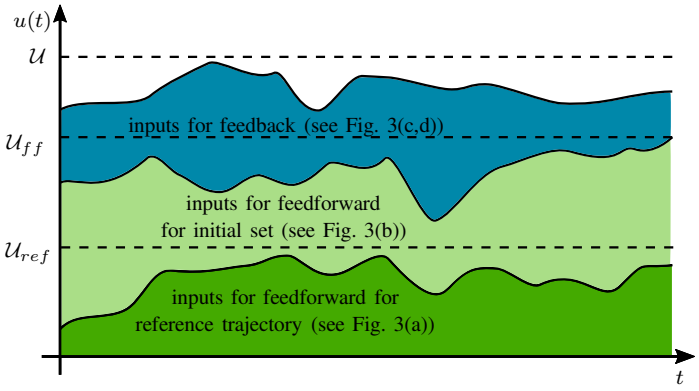


Fig. 5. Illustration of the three different input constraints which are used in our optimization problem. While we restrict the inputs for initial optimizations, we allow the later optimizations to apply the unused parts of the input constraints of previous optimizations.

IV. Main Part

Before we explain the two parts of our controller in detail, let us first define the reachable set of a system.

Definition 1 (Reachable Set): For system (1), the reachable set $\mathcal{R}_{t,\mathcal{U},\mathcal{W}}(\mathcal{S}) \subset \mathbb{R}^n$ for a time t, inputs $u(\cdot) \in \mathcal{U} \subset \mathbb{R}^m$, disturbances $w(\cdot) \in \mathcal{W} \subset \mathbb{R}^d$, and a set of initial states $\mathcal{S} \subset \mathbb{R}^n$ is the set of end states of trajectories starting in $\mathcal{S}$ after time t, i.e.,

$$\mathcal{R}_{t,\mathcal{U},\mathcal{W}}(\mathcal{S}) = \{x(t) \in \mathbb{R}^n | \exists x(0) \in \mathcal{S}, u(\cdot) \in \mathcal{U}, w(\cdot) \in \mathcal{W} : \xi(x(0), u(\cdot), w(\cdot), t) = x(t)\}.$$

The reachable set over a time interval $[t_1, t_2]$ is the union

$$\mathcal{R}_{[t_1,t_2],\mathcal{U},\mathcal{W}}(\mathcal{S}) = \bigcup_{t \in [t_1,t_2]} \mathcal{R}_{t,\mathcal{U},\mathcal{W}}(\mathcal{S}).$$

If we consider the reachable set for a system with feedback $u_{fb}(\hat{x}(t))$, we denote by $\mathcal{R}_{t,u_{fb},\mathcal{W}}(\mathcal{S})$ the reachable set obtained for the closed-loop dynamics $\dot{x}(t) = f(x(t), u_{fb}(\hat{x}(t)), w(t))$, where the measurement errors are modeled as disturbances which only affect the control laws. If we consider systems without disturbances, we use $\mathcal{R}_{t,\mathcal{U}}(\mathcal{S})$ as a shorthand for $\mathcal{R}_{t,\mathcal{U},0}(\mathcal{S})$, i.e., $\mathcal{W} = 0$. Since it is not possible to compute exact reachable sets for most systems [64], we compute over-approximations instead. As shown in

[65], it is possible to compute the over-approximation to any desired accuracy.

The efficiency of the computation of reachable sets depends on the chosen set representation. A type of set which has very favorable properties for reachability analysis are zonotopes [66]:

Definition 2 (Zonotope): A set is called a zonotope if it can be written as

$$\mathcal{Z} = \left\{ x \in \mathbb{R}^n \middle| x = c + \sum_{i=1}^{p} \alpha_i g^{(i)}, \alpha_i \in [-1, 1] \right\}.$$

Here, $c \in \mathbb{R}^n$ defines the center of the zonotope, and $g^{(i)} \in \mathbb{R}^n, i \in \{1, \dots, p\}$, are $p = o\,n$ generators, with o denoting the order of the zonotope. We use $\langle c, g^{(1)}, \dots, g^{(p)} \rangle$ as a more concise notation of $\mathcal{Z}$. A zonotope with n linearly independent generators is called a parallelotope.

For simplicity, we assume that the initial set $\mathcal{X}_0$, the set of measured initial states $\hat{\mathcal{X}}_0 := \{h(x^{(0)}, \nu) | x^{(0)} \in \mathcal{X}_0, \nu \in \mathcal{V}\}$, the disturbance set $\mathcal{W}$, and the set of measurement noise $\mathcal{V}$ are given as zonotopes or can be over-approximated by zonotopes. This is actually not very conservative in practice, as the most common class of sets used to specify such sets are boxes, which are a special class of zonotopes. Zonotopes offer the advantage that they can represent a set with only a few parameters. While one needs 2^n extreme points to describe a box, a zonotope needs only n generators. This linear scaling of generators with dimension allows an efficient set representation even in high dimensions. Not only the number of vertices but also the number of half-spaces needed to represent a zonotope can grow very high: as described in [67], an n dimensional zonotope with p generators can have up to $2\binom{p}{n-1}$ half-spaces.

In the presented control law, we therefore take advantage of the efficient set representation of zonotopes by designing algorithms which are based only on the generators and do not require us to compute any vertices or half-spaces. Let us now explain the parts of the control law in more detail. The resulting synthesis procedure is summarized in Alg. 1 and is explained subsequently.

Algorithm 1 Set-Based Optimal Controller Synthesis Algorithm

1: $(x_{ref}, u_{ref}) \leftarrow$ solution of optimization problem (9) for the center trajectory
2: $(A_k, B_k) \leftarrow$ linearization and time-discretization of $f(x, u, 0)$ along x_{ref}, u_{ref} using (10) and (12)
3: $(u_{ff}(g_{\hat{x}}^{(1)}), \dots, u_{ff}(g_{\hat{x}}^{(p)})) \leftarrow$ solution of optimization problem (15) for feedforward control of initial set
4: $(Q, R) \leftarrow$ solution of optimization problem (18) for the optimal feedback matrix
5: **if** all optimization problems feasible **then** store controller and reachable set
6: **end if**

A. State-Dependent Feedforward Control

We want to find an individual feedforward control law for each state $\hat{x}^{(0)} \in \hat{\mathcal{X}}_0$, such that it is steered as close as possible to $x^{(f)}$, i.e.,

$$\begin{aligned} u_{ff}(\cdot) = \arg\min_{u_{ff}(\cdot)} \max_{\hat{x}^{(0)} \in \hat{\mathcal{X}}_0} & \|\xi(\hat{x}^{(0)}, u_{ff}(\hat{x}^{(0)}, \cdot), 0, t_f) - x^{(f)}\|_1 \\ & + \gamma_{ff} \int_0^{t_f} \|u_{ff}(\hat{x}^{(0)}, t)\|_1 dt \quad (5) \\ \text{s.t. } \forall t \in [0, t_f], \forall \hat{x}^{(0)} \in \hat{\mathcal{X}}_0 : & \xi(\hat{x}^{(0)}, u_{ff}(\hat{x}^{(0)}, \cdot), 0, t) \in \mathcal{X}_{ff}, \\ & u_{ff}(\hat{x}^{(0)}, t) \in \mathcal{U}_{ff}, \end{aligned}$$

where we minimize the difference between each state to the desired final state as well as the applied inputs weighted by $\gamma_{ff} \in \mathbb{R}^+$. We consider the set of all possibly measured initial states $\hat{\mathcal{X}}$, such that we obtain a valid feedforward control law for each of them.

For a general, nonlinear system with a nonlinear control law, it is not feasible to solve (5) for every possible state of the initial set as there are infinitely many. By restricting the feedforward control to linearized dynamics with linear control laws, however, we can overcome this problem by interpolating finitely many solutions using the superposition principle of linear systems. To find a controller which is easy to compute and apply online on the real system, we choose a control law which takes advantage of our zonotopic representation of the states:

$$u_{ff}(\hat{x}, t) = u_{ff}(c_{\hat{x}}, t) + \sum_{i=1}^{p} \alpha_i(\hat{x}) u_{ff}(g_{\hat{x}}^{(i)}, t), \quad (6)$$

where $\alpha_i(\hat{x})$ refers to the i-th entry of the parameter vector $\alpha(\hat{x})$ encoding $\hat{x}$ inside the zonotope of initial measured states

$$\hat{\mathcal{X}}_0 = \langle c_{\hat{x}}, g_{\hat{x}}^{(1)}, \dots, g_{\hat{x}}^{(p)} \rangle, \quad (7)$$

i.e.,

$$\hat{x} = c_{\hat{x}} + \sum_{i=1}^{p} \alpha_i(x) g_{\hat{x}}^{(i)}, \quad (8)$$

and therefore $\alpha_i \in [-1, 1]$. So, the control laws $u_{ff}(\hat{x}, t)$ for any state $\hat{x}$ depend only on the $p + 1$ input trajectories $u_{ff}(c_{\hat{x}}, \cdot), u_{ff}(g_{\hat{x}}^{(1)}, \cdot), \dots, u_{ff}(g_{\hat{x}}^{(p)}, \cdot)$. The basic idea to solve (5) is the following: Since a zonotope is defined as a linear superposition of a center and several generators, the desired final set has a center which is as close as possible to the desired final state $x^{(f)}$ (see Fig. 3(a)) and generators which are as small as possible (see Fig. 3(b)).

We find a reference trajectory which steers the center $c_{\hat{x}}$ of $\hat{\mathcal{X}}_0$ as close as possible to the final state by solving an optimization problem of the form (see line 1 of Alg. 1 and Fig. 3(a))

$$\begin{aligned} u_{ref}(\cdot) = \arg\min_{u_{ref}(\cdot)} & \|(\xi(c_{\hat{x}}, u_{ref}(\cdot), 0, t_f) - x^{(f)})\|_1 \\ & + \gamma_{ref} \int_0^{t_f} \|u_{ref}(t)\|_1 dt \quad (9) \\ \text{s.t. } \forall t \in [0, t_f] : & \xi(c_{\hat{x}}, u_{ref}(\cdot), 0, t) \in \mathcal{X}_{ref}, \\ & u_{ref}(t) \in \mathcal{U}_{ref}, \end{aligned}$$

which is similar to (5) but with the center of the initial set as the only state. We denote the resulting reference trajectory by $x_{ref}(\cdot) = \xi(c_{\hat{x}}, u_{ref}(\cdot), 0, \cdot)$. By restricting the inputs to be piecewise constant, this control problem can be efficiently solved using numerical optimization algorithms such as multiple-shooting [1]. This restriction to piecewise-constant inputs is a simplification often used in practice, and their sampling time can be chosen freely by the user without changing any general properties, as shown later in Thm. 1.

To be able to apply the superposition principle, as an intermediate step, we linearize the system along the reference trajectory (see line 2). We use feedforward and feedback controllers to keep the real trajectories close enough to the reference trajectory so that the linearization errors do not get too large. To obtain the linearization points t'_k, we first divide the reference trajectory into N parts of duration $\Delta t = \frac{t_f}{N},\ N \in \mathbb{N}$ and then choose the middle of the time intervals $[t_k, t_{k+1}]$ with $t_k = k\Delta t,\ k \in \{0, \dots, N\}$, i.e., $t'_k = \frac{1}{2}(t_{k+1} - t_k),\ k \in \{0, \dots, N-1\}$. The system matrices of the linear dynamics are then given by

$$A_{c,k} = \left.\frac{\partial f(x,u,0)}{\partial x}\right|_{\substack{x=x_{ref}(t'_k)\\u=u_{ref}(t'_k)}}, \tag{10}$$
$$B_{c,k} = \left.\frac{\partial f(x,u,0)}{\partial u}\right|_{\substack{x=x_{ref}(t'_k)\\u=u_{ref}(t'_k)}}.$$

By restricting the inputs to be constant in each time interval, we can treat the system as a discrete-time, linear system of the form

$$x(t_{k+1}) = A_k x(t_k) + B_k u(t_k), \tag{11}$$

with

$$A_k = e^{A_{c,k}\Delta t},\ B_k = \int_0^{\Delta t} e^{A_{c,k}\tau} d\tau B_{c,k}. \tag{12}$$

Lemma 1: Let us define the shorthands $\hat{\gamma}_{ff} = \frac{t_f}{N}\gamma_{ff}$, $\bar{A} = A_{N-1}\dots A_0$ and $\bar{B}_k = A_{N-1}\dots A_{k+1}B_k, k \in \{0, \cdots, N-1\}$ with $\bar{B}_{N-1} = B_{N-1}$. Then, for the linearized and time-discretized dynamics, the optimization problem

$$\begin{aligned}\min_{u_{ff}(\cdot)} &\left\|\bar{A}c_{\hat{x}} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(c_{\hat{x}}, t_k) - x^{(f)}\right\|_1 \\ &+ \sum_{i=1}^{p}\left\|\bar{A}g_{\hat{x}}^{(i)} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(g_{\hat{x}}^{(i)}, t_k)\right\|_1 \\ &+ \hat{\gamma}_{ff}\left(\sum_{k=0}^{N-1}\left\|u_{ff}(c_{\hat{x}}, t_k)\right\|_1 + \sum_{i=1}^{p}\left\|u_{ff}(g_{\hat{x}}^{(i)}, t_k)\right\|_1\right),\end{aligned}$$

minimizes an upper bound of the cost function of (5).

Proof: Using the discretized dynamics in (11), it is well known that the state at time t_N can be obtained as

$$\begin{aligned}\xi(\hat{x}^{(0)}, u_{ff}(\cdot), 0, t_N) =& \bar{A}\hat{x}^{(0)} + \bar{B}_0 u_{ff}(\hat{x}^{(0)}, t_0) + \dots \\ &+ \bar{B}_{N-1}u_{ff}(\hat{x}^{(0)}, t_{N-1}).\end{aligned}$$

Since the set of measured initial states is a zonotope $\hat{\mathcal{X}}_0 = \langle c_{\hat{x}}, g_{\hat{x}}^{(1)}, \dots, g_{\hat{x}}^{(q)}\rangle$, we can use the fact that any initial state $\hat{x}^{(0)}$ can be expressed as

$$\hat{x}^{(0)} = c_{\hat{x}} + \sum_{i=1}^{p}\alpha_i(\hat{x}^{(0)})g_{\hat{x}}^{(i)}, \forall \hat{x}^{(0)} \in \hat{\mathcal{X}}_0.$$

This expression of $\hat{x}^{(0)}$ makes it possible to rewrite optimization problem (5) without its input cost term:

$$\begin{aligned}&\min_{u_{ff}(\cdot)}\max_{\hat{x}^{(0)}\in\hat{\mathcal{X}}_0}\left\|\xi(\hat{x}^{(0)}, u_{ff}(\hat{x}^{(0)}, \cdot), 0, t_N) - x^{(f)}\right\|_1 \\ &= \min_{u_{ff}(\cdot)}\max_{\hat{x}^{(0)}\in\hat{\mathcal{X}}_0}\left\|\bar{A}\hat{x}^{(0)} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(\hat{x}^{(0)}, t_k) - x^{(f)}\right\|_1 \\ &= \min_{u_{ff}(\cdot)}\max_{\alpha\in[-1,1]^p}\left\|\bar{A}c_{\hat{x}} + \sum_{i=1}^{p}\bar{A}g_{\hat{x}}^{(i)}\alpha_i + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(c_{\hat{x}}, t_k)\right. \\ &\qquad\left. + \sum_{i=1}^{p}\sum_{k=0}^{N-1}\bar{B}_k u_{ff}(g_{\hat{x}}^{(i)}, t_k)\alpha_i - x^{(f)}\right\|_1 \\ &\le \min_{u_{ff}(\cdot)}\max_{\alpha\in[-1,1]^p}\left\|\bar{A}c_{\hat{x}} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(c_{\hat{x}}, t_k) - x^{(f)}\right\|_1 \\ &\qquad + \sum_{i=1}^{p}\left\|\left(\bar{A}g_{\hat{x}}^{(i)} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(g_{\hat{x}}^{(i)}, t_k)\right)\alpha_i\right\|_1 \\ &= \min_{u_{ff}(\cdot)}\left\|\bar{A}c_{\hat{x}} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(c_{\hat{x}}, t_k) - x^{(f)}\right\|_1 \\ &\qquad + \sum_{i=1}^{p}\left\|\bar{A}g_{\hat{x}}^{(i)} + \sum_{k=0}^{N-1}\bar{B}_k u_{ff}(g_{\hat{x}}^{(i)}, t_k)\right\|_1. \end{aligned} \tag{13}$$

The last equality holds due to the fact that we choose the α_i such that they maximize the 1-norm and since $|\alpha_i| \le 1$, the values of α_i can be neglected. For the input cost term, for piecewise constant inputs as obtained from the discretized dynamics, it holds that

$$\gamma_{ff}\int_0^{t_f}\|u_{ff}(\hat{x}, t)\|_1 dt = \hat{\gamma}_{ff}\sum_{k=0}^{N-1}\|u_{ff}(\hat{x}, t_k)\|_1.$$

This can be bounded analogously to the state costs in (13) as:

$$\begin{aligned}&\max_{\hat{x}^{(0)}\in\hat{\mathcal{X}}_0}\hat{\gamma}\left(\sum_{k=0}^{N-1}\|u_{ff}(\hat{x}, t_k)\|_1\right) \\ &= \max_{\alpha\in[-1,1]^p}\hat{\gamma}\left(\sum_{k=0}^{N-1}\left\|u_{ff}(c_{\hat{x}}, t_k) + \sum_{i=1}^{p}u_{ff}(g_{\hat{x}}^{(i)}, t_k)\alpha_i\right\|_1\right) \\ &\le \max_{\alpha\in[-1,1]^p}\hat{\gamma}\left(\sum_{k=0}^{N-1}\|u_{ff}(c_{\hat{x}}, t_k)\|_1 + \sum_{i=1}^{p}\left\|u_{ff}(g_{\hat{x}}^{(i)}, t_k)\alpha_i\right\|_1\right) \\ &= \hat{\gamma}\left(\sum_{k=0}^{N-1}\|u_{ff}(c_{\hat{x}}, t_k)\|_1 + \sum_{i=1}^{p}\left\|u_{ff}(g_{\hat{x}}^{(i)}, t_k)\right\|_1\right).\end{aligned}$$

■

Lemma 1 allows us to obtain a new optimization problem, which minimizes an upper bound for the size of the reachable set. While the original optimization problem, which contains any possible combination of α_i, becomes hard to compute, the new efficient formulation decouples the influences of the individual generators.

The optimization problem is still coupled through the constraints, as the state and overall input must satisfy the tightened state and input constraints, $\mathcal{X}_{ff}$ and $\mathcal{U}_{ff}$, respectively. For reasons of computational efficiency, we simplify checking constraints by only checking them for the reachable sets at the time points t_k, following the principle in Remark 1. To check if reachable zonotopes $\mathcal{R}_{t_k,u_{ff}}(\hat{\mathcal{X}}_0)$ satisfy convex state constraints of the form $\mathcal{X}_{ff} = \{x \in \mathbb{R}^n | Cx \leq d\}$, we use the following lemma.

Lemma 2: A zonotope $\mathcal{Z} = \langle c, g^{(1)}, \dots, g^{(p)} \rangle$ satisfies convex constraints of the form $\mathcal{X}_{ff} = \{x \in \mathbb{R}^n | Cx \leq d\}$ if

$$Cc + \sum_{i=1}^{p} |Cg^{(i)}| \leq d, \tag{14}$$

where the absolute value and less or equal operators are both performed element-wise.

Proof: Using the zonotope representation results in

$$\begin{aligned} &Cx \leq d, \forall x \in \mathcal{X}_{ff} \\ \Leftrightarrow\ &Cc + \sum_{i=1}^{p} \alpha_i Cg^{(i)} \leq d, \forall \alpha_i \in [-1, 1]. \end{aligned}$$

The left side of the above inequality can be further bounded by

$$\begin{aligned} &Cc + \sum_{i=1}^{p} \alpha_i Cg^{(i)} \leq Cc + \sum_{i=1}^{p} |\alpha_i Cg^{(i)}| \\ \leq &Cc + \sum_{i=1}^{p} \underbrace{|\alpha_i|}_{\leq 1} |Cg^{(i)}| \leq Cc + \sum_{i=1}^{p} |Cg^{(i)}|. \end{aligned}$$

In fact, there exists an α with $\alpha_i = \pm 1$, such that the computed bound is touched. ∎

We must also ensure that the sum of all possible inputs for the center and generators does not exceed the input bounds:

Corollary 1: The feedforward control law (6) satisfies the input constraint $\mathcal{U}_{ff} = \{u \in \mathbb{R}^m | C_u u \leq d_u\}$ if and only if

$$\begin{aligned} &C_u u_{ff}(c_{\hat{x}}, t_k) + \sum_{i=1}^{p} |C_u u_{ff}(g_{\hat{x}}^{(i)}, t_k)| \leq d_u, \\ &\forall k \in \{0, \dots, N-1\}. \end{aligned}$$

Proof: Using the same proof concept as in Lemma 2 results in

$$\begin{aligned} &u_{ff}(c_{\hat{x}}, t_k) + \sum_{i=1}^{p} \alpha_i u_{ff}(g_{\hat{x}}^{(i)}, t_k) \in \mathcal{U}_{ff}, \\ &\forall \alpha \in [-1, 1]^p, \forall k \in \{0, \dots N-1\} \\ \Leftrightarrow\ &C_u u_{ff}(c_{\hat{x}}, t_k) + \sum_{i=1}^{p} |C_u u_{ff}(g_{\hat{x}}^{(i)}, t_k)| \leq d_u, \\ &\forall k \in \{0, \dots, N-1\}. \end{aligned}$$

∎

To have a more accurate solution, we use the reference inputs for the center of the measured initial set, i.e., $u_{ff}(c_{\hat{x}}, \cdot) = u_{ref}(\cdot)$, which we compute for the actual nonlinear dynamics and therefore without any linearization errors. Let us combine the results from Lemma 1, Lemma 2, and Corollary 1 into a single optimization problem, which is presented in the following theorem and solved in line 3 in Alg. 1.

Theorem 1: For the time-discretized and linearized system (11), the following optimization problem minimizes an upper-bound for the cost function in (5) while taking the constraints $\mathcal{X}_{ff}$ and $\mathcal{U}_{ff}$ into account:

$$\min_{\substack{u_{ff}(g_{\hat{x}}^{(1)}), \\ \dots, u_{ff}(g_{\hat{x}}^{(p)})}} \left\| \begin{bmatrix} \bar{A} g_{\hat{x}}^{(1)} + \bar{B} u_{ff}(g_{\hat{x}}^{(1)}) \\ \vdots \\ \bar{A} g_{\hat{x}}^{(q)} + \bar{B} u_{ff}(g_{\hat{x}}^{(q)}) \end{bmatrix} \right\|_1 + \hat{\gamma}_{ff} \left\| \begin{bmatrix} u_{ff}(g_{\hat{x}}^{(1)}) \\ \vdots \\ u_{ff}(g_{\hat{x}}^{(q)}) \end{bmatrix} \right\|_1 \tag{15}$$

s.t. $\forall k \in \{0, \dots, N-1\}$:

$$C_u u_{ff}(c_{\hat{x}}, t_k) + \sum_{i=1}^{q} |C_u u_{ff}(g_{\hat{x}}^{(i)}, t_k)| \leq d_u, \tag{16}$$

$$Cx_{ref}(t_k) + \sum_{i=1}^{p} |Cg^{(i)}(t_k)| \leq d, \tag{17}$$

with the shorthands $\hat{\gamma}_{ff} = \frac{t_f}{N} \gamma_{ff}$, $\bar{B} = \left[\bar{B}_0, \cdots, \bar{B}_{N-1}\right]$, $u_{ff}(g_{\hat{x}}^{(i)}) = [u_{ff}(g_{\hat{x}}^{(i)}, t_0)^T, \dots, u_{ff}(g_{\hat{x}}^{(i)}, t_{N-1})^T]^T$, and equivalently for $u_{ff}(c_{\hat{x}})$.

Proof: It follows from Lemma 1, together with the fact that

$$\|a\|_1 + \|b\|_1 = \left\| \begin{bmatrix} a \\ b \end{bmatrix} \right\|_1,$$

that we can obtain the cost function in the form of (15). The input constraint (16) follows directly from Corollary 1 and the state constraint (17) from Lemma 2. ∎

It is shown in [68, Ch. 6] and [69] that the norm and absolute value can be transformed to linear constraints and cost functions. Therefore, this problem can be solved efficiently in a single linear program. As we consider a time-discrete system for faster computation, we check the state and input constraints only at sampling times, too. Since we have piecewise-continuous inputs, this does not make a difference for the input constraints. To account for the behavior of the states between sampling times, we tightened the state constraints $\mathcal{X}_{ff}$. We formally check the inter-sampling times in the next part by computing reachable sets which over-approximate all possible solutions during time-intervals.

B. Feedback Control

To synthesize a feedback controller, we extend the idea from [61], where it was restricted to linear systems without measurement noise: We use a time-varying linear feedback controller whose parameters are obtained by optimizing over the reachable set of the closed-loop system. This allows us to find a controller which minimizes the effects from linearization errors, disturbances and measurement noise, while satisfying the input constraints at all times (see Fig. 3(c)–(d)).

To simplify the reachability analysis as well as to obtain a fast-to-evaluate control law, we choose a linear control structure with time-varying feedback matrix $K(t)$:

$$u_{fb}(\hat{x}(t), t) = K(t)(\hat{x}(t) - \hat{x}_{ff}(t)),$$

with $\hat{x}_{ff}(t) = \xi\big(\hat{x}(0), u_{ff}(\hat{x}(0), \cdot), 0, t\big)$. Since the computational effort depends on the number of optimization variables, we do not optimize all entries of K. Instead, we optimize the weighting matrices of LQR controllers [70], where we restrict the matrices Q and R to be diagonal matrices for computational efficiency. Since we can further normalize the matrices by choosing the first entry of Q to be equal to 1, we only have to consider $n+m-1$ optimization variables. This is a large improvement compared to the approach in [61], where the number of optimization variables increased by $M(n+m-1)$, where M denotes the number of different sets of optimization matrices which were needed to compensate the size of the initial set without violating input constraints. This much larger number of optimization variables restricted the application of [61] to linear systems. Note that if desired, one can also use non-diagonal Q and R matrices. We recompute the feedback matrix $K(t)$ at each sampling time based on the changing linearized dynamics. This results in a controller which adapts to the actual nonlinear dynamics while preserving the simple control structure of a piecewise linear control law.

The nonlinear optimization problem (line 4) for the Q and R matrices is given by:

$$\min_{Q,R} \max_{x \in \mathcal{R}_{t_f,u_{ctrl},\mathcal{W}}(\mathcal{X}_0)} \|x - x^{(f)}\|_1 \tag{18}$$
$$\text{s.t. } \mathcal{R}_{t_f,u_{ctrl},\mathcal{W}}(\mathcal{X}_0) \subseteq \mathcal{X},$$
$$\forall \hat{x} \in h(x,\nu), \forall x \in \mathcal{R}_{t,u_{ctrl},\mathcal{W}}(\mathcal{X}_0), \forall \nu \in \mathcal{V}, \forall t \in [0, t_f]:$$
$$u_{ctrl}(\hat{x}, t) \in \mathcal{U}.$$

Analogous to Lemma 1, an upper bound for the cost function can be computed much easier as

$$\min_{Q,R} \|c - x^{(f)}\|_1 + \sum_{i=1}^{p} \|g^{(i)}\|_1,$$

with c and $g^{(1)}, \ldots, g^{(p)}$ denoting the center and the generators of $\mathcal{R}_{t,u_{ctrl},\mathcal{W}}(\mathcal{X}_0)$.

By using the real cost function or the over-approximation and by checking the constraints for all times as shown in Lemma 2, we can express the optimization problem in the standard form, i.e., a cost function which returns a scalar value subject to minimization and a constraint function which returns a vector of values which are all negative if the constraints are satisfied. This allows us to use existing nonlinear programming solvers. Since we are able to obtain the applied inputs from the reachability analysis in the form of zonotopes as well, we can use the same method for state constraints to check the input constraints for all times. By incorporating reachability analysis inside the optimization problem, we are able to directly optimize over all possible trajectories in a single nonlinear optimization problem (18) (see Fig. 3(d)).

V. Discussion of the Algorithm

In this section, we discuss different aspects of the presented control algorithm.

A. Formal Guarantees

One of the advantages of our approach is that it does not require proofs for stability or for constraint satisfaction. Verifying the stability or safety of complex, nonlinear dynamics that are affected by disturbances and measurement noise, restricted by state and input constraints, and controlled by switching control laws is a very hard task when using classical approaches. Finding a Lyapunov function for such a system easily becomes infeasible in practice. Instead, safety and constraint satisfaction is guaranteed by the included reachability analysis for all initial states and all possible disturbances. In addition, if the optimization results in a feasible controller, we know that none of the possible trajectories can intersect with any forbidden states. By computing all optimization problems and reachable set computations offline in advance, we are able to only consider motion primitives in our maneuver automaton for which a feasible solution has been found. Thus, even if we cannot find a feasible controller for a certain motion primitive, we know this offline in advance and can recompute this motion primitive (see Alg. 1 line 5). During online planning, the resulting maneuver automaton can be applied together with fail-safe techniques [71] to ensure that a safe control strategy exists for any situation.

B. Optimality

As we rely on nonlinear programming algorithms that do not guarantee convergence to a global optimal solution, we can only expect to obtain a local optimum. In fact, there is no efficient method able to obtain globally optimal controllers for disturbed, nonlinear systems [6], [72]. Most optimal control approaches only consider open-loop dynamics or only undisturbed feedback controllers [73], [74]. Our synthesis approach optimizes over the whole reachable set, i.e., all possible trajectories resulting from any possible disturbance and measurement noise realization, and chooses the controller which minimizes all of them. This is not done in any comparable method, as most other formal methods consider fixed controllers, as e.g., in tube-based MPC, or fixed control inputs for the whole set, such as in abstraction-based methods.

C. Complexity

When we discuss the complexity of our algorithm, we have to distinguish between online and offline complexity. The critical part is the online complexity, as this restricts the sampling times of the controlled system and therefore its performance.

Since we use a piecewise linear controller, the online complexity is very low. We can store the different controller matrices in a look-up table, and the computation itself only requires two matrix vector multiplication and few additions with a combined complexity of $\mathcal{O}(n\,m)$ (with n denoting the number of states and m the number of inputs). Simulation examples, like in [29], show that this look-up table does not need to get too large: for an autonomous vehicle example, 20 different motion primitives are already enough for planning in certain environments. This result matches the observation of

[75], which states that "a small number of patterns is sufficient to model the vast majority of traffic scenes."

We cannot provide fixed complexity bounds for the offline complexity, as it relies on nonlinear programming for which no complexity bounds exist. Since we cannot bound the number of iterations of the nonlinear programming algorithm, we cannot bound the overall complexity. Let us still give an idea of the complexity of the different steps.

Computing the reference trajectory requires solving a nonlinear program, however only for a single state with piecewise constant inputs, which can be solved fast in practice. For the state-dependent feedback controller, we have to solve a single linear program. The complexity of solving linear programs depends on the exact implementation and the number and type of constraints. There exist algorithms with polynomial time complexity in the number of optimization variables and constraints [76]. If we consider zonotopes of a fixed order, then the number of optimization variables and number of constraints grows polynomially as well (with $\mathcal{O}(n^2)$, if $n > m$). In practice, the computation time for solving linear programs, even those with many constraints, is low and does not add much to the overall computation time.

The most computationally expensive part is the optimization of the feedback controller. As mentioned in the beginning, the number of optimization variables grows linearly with $n+m-1$. Evaluating the cost and constraint function for a single step requires the computation of the reachability analysis and some algebraic operations on the resulting zonotopes, which together have a complexity of $\mathcal{O}(n^3)$.

VI. Numerical Example

To illustrate the applicability of our new controller and its improved performance compared to earlier work, we revisit the examples found in papers [61] and [60], as well as a nonlinear tube-based MPC example from [77] and a planning example from [78]. We implement our approach in MATLAB and use the following toolboxes: the CORA toolbox [79] for reachability analysis, the ACADO toolbox [80] for optimizing the reference trajectory, the CVX toolbox [81] for linear programs for optimizing the generators, and fmincon with the active-set algorithm from MATLAB for solving the nonlinear program. The computations are performed on a laptop with a 3.1 GHz Intel dual-core i7 processor and 16 GB memory.

A. Linear Example: Vehicle Platoon

Let us first revisit the numerical example from [61], where we compute a controller for a continuous-time, linear system subject to constraints and disturbances. We consider a platoon with four vehicles, where the dynamics of each vehicle $i \in \{1,2,3,4\}$ is given by

$$\dot{p}^{(i)} = v^{(i)}, \qquad \dot{v}^{(i)} = a^{(i)} + w^{(i)},$$

where $p^{(i)}$ denotes the position of the $i-$th vehicle, $v^{(i)}$ its velocity, and $a^{(i)}$ its acceleration, i.e., the controllable input. The disturbances, i.e., the uncontrollable inputs, are denoted by $w^{(i)}$. To model the whole platoon, we use the absolute states of the first vehicle and the relative states of the second, third, and fourth vehicles, i.e., we consider the eight-dimensional state vector $x = [p^{(1)}, v^{(1)}, p^{(1)} - p^{(2)} - c_s, v^{(1)} - v^{(2)}, p^{(2)} - p^{(3)} - c_s, v^{(2)} - v^{(3)}, p^{(3)} - p^{(4)} - c_s, v^{(3)} - v^{(4)}]^T$, the input vector $u = [a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)}]$, and disturbance vector $w = [w^{(1)}, w^{(2)}, w^{(3)}, w^{(4)}]$. Here, $c_s \in \mathbb{R}^+$ denotes a safety constant, defining a minimal safe distance. The resulting dynamics are given by

$$\begin{aligned}
\dot{x}_1 &= x_2, & \dot{x}_2 &= u_1 + w_1,\\
\dot{x}_3 &= x_4, & \dot{x}_4 &= u_1 - u_2 + w_1 - w_2,\\
\dot{x}_5 &= x_6, & \dot{x}_6 &= u_2 - u_3 + w_2 - w_3,\\
\dot{x}_7 &= x_8, & \dot{x}_8 &= u_3 - u_4 + w_3 - w_4.
\end{aligned}$$

We assume that all inputs are constrained between $u_i(\cdot) \in [-10, 10]\frac{m}{s}, i \in \{1,2,3,4\}$ and all disturbances vary freely in the interval $w_i(\cdot) \in [-1,1]\frac{m}{s}$. Moreover, we have the state constraint that the vehicles must keep the minimal safety distance, i.e., $x_3, x_5, x_7 > 0$. We consider the following scenario, which can be used as a motion primitive in a platooning maneuver automaton: The vehicles start with initial states ranging freely in the box $[-0.2, 0.2]m \times [19.8, 20.2]\frac{m}{s} \times [0.8, 1.2]m \times [-0.2, 0.2]\frac{m}{s} \times [0.8, 1.2]m \times [-0.2, 0.2]\frac{m}{s} \times [0.8, 1.2]m \times [-0.2, 0.2]\frac{m}{s}$, i.e., the vehicles drive with different velocities around $20\frac{m}{s}$ behind each other. We consider a final state $[21m, 22\frac{m}{s}, 1m, 0\frac{m}{s}, 1m, 0\frac{m}{s}, 1m, 0\frac{m}{s}]^T$ that should be reached after $1s$, i.e., the whole platoon should speed up to $22\frac{m}{s}$ and align in a safe distance of $c_s + 1m$ between each vehicle.

In Fig. 6, we show the resulting final reachable sets for the controller from [61] in blue and with the new approach in green. We see that the new approach is able to obtain much smaller reachable sets and its computation time of around one minute is around five times faster than the old approach. This is quite impressive as even the old approach performs much better than constant LQR tracking controllers (red); see [61].

B. Nonlinear Example: Kinematic Vehicle

Let us now consider the autonomous vehicle example from [60]. The kinematic car model, which covers the most important dynamics of a car, is given by

$$\dot{v} = a + w_1,\ \dot{\Psi} = b + w_2,\ \dot{x} = v\,cos(\Psi),\ \dot{y} = v\,sin(\Psi),$$

where the states v, Ψ, x, and y are the velocity, the orientation, and the positions in x and in y directions, respectively. The acceleration a and the normalized steering angle b are the inputs, and w_1 and w_2 are additive disturbances. They are constrained to lie in the intervals $a \in [-9.81, 9.81]\frac{m}{s^2}$, $b \in [-0.4, 0.4]\frac{rad}{s}$, $w_1 \in [-2,2]\frac{m}{s^2}$, and $w_2 \in [-0.08, 0.08]\frac{rad}{s}$. These are the same values as in [60] with the exception of w_1 and w_2 sets which are enlarged by a factor of four compared to our earlier work.

We show the reachable set for a "turn left" maneuver in Fig. 7. There, we start from the initial set $\mathcal{X}_0 = [19.8, 20.2]\frac{m}{s} \times [-0.02, 0.02]rad \times [-0.2, 0.2]m \times [-0.2, 0.2]m$ and want to get as close as possible to the final state $x^{(f)} = \left[20\,\frac{m}{s}, 0.2\,rad, 19.87\,m, 1.99\,m\right]^T$ after $1s$. The computation of the new controller takes around two minutes. For a better

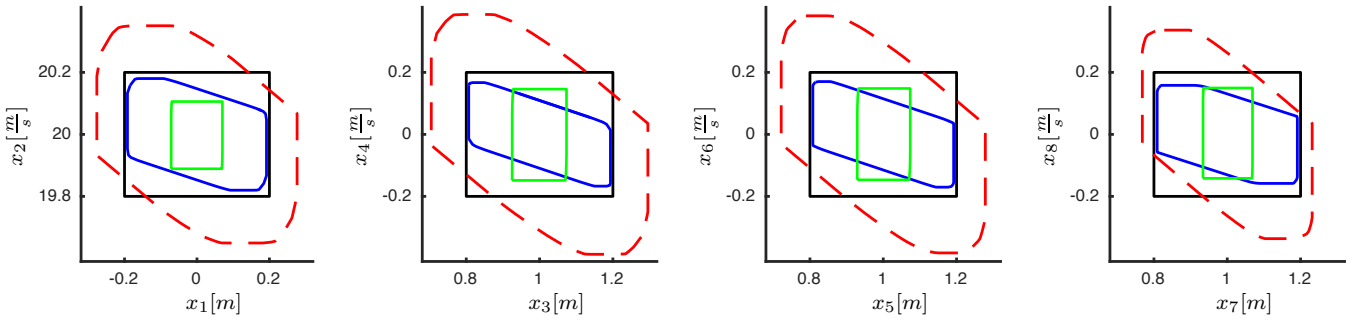


Fig. 6. Linear platooning example: Initial (black) and shifted final sets (green) for our new controller, projected onto the (x_1, x_2), the (x_3, x_4), the (x_5, x_6), and the (x_7, x_8) planes. For comparison, the final sets of our controller from [61] (blue) and an LQR controller (red) are shown.

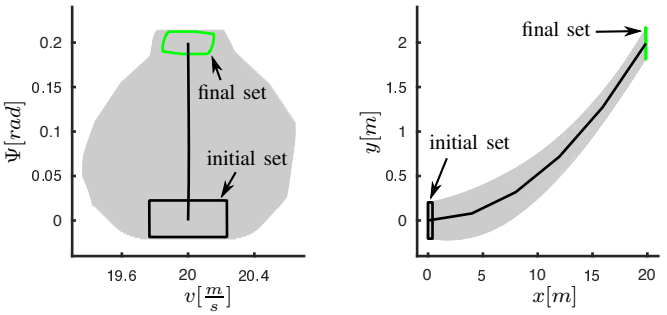


Fig. 7. Nonlinear car example: Reachable sets for the "turn left" maneuver with our new controller, projected to the (v, Ψ) and the (x, y) planes. The initial set is plotted in black, the final set in green, and the reachable set for all times between in gray. The black line shows the center trajectory.

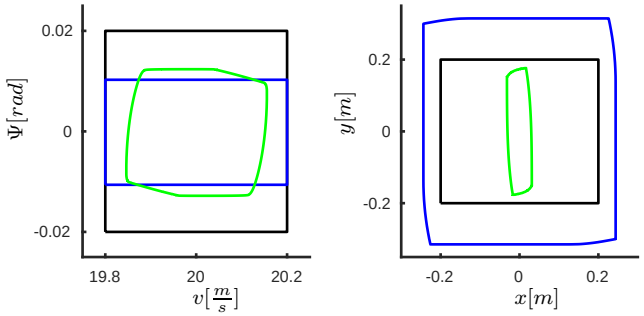


Fig. 8. Nonlinear car example: Initial (black) and shifted final set (green) for our new controller, projected to the (v, Ψ) and the (x, y) planes, for the "turn left" maneuver. For comparison, the final set of the controller from [60] (blue).

illustration, we show again the initial set with the shifted final set in Fig. 8. The reachable set of the old controller with the new disturbance set is again shown in blue and the reachable set of the new controller in green. We see that the additional feedback controller allows us to reach the shifted initial set, while the old controller, which applies only the feedforward controller in an iterative fashion, similar to MPC, results in a very large reachable set. In [60], we compare our old controller with LQR tracking controllers. As their reachable sets with the increased disturbances would become much larger than the sets in Fig. 8, we omit them for clarity.

We compute 200 simulations with random disturbances for this scenario with our new controller and show them in Fig. 9. We see that the simulations cover almost the whole reachable

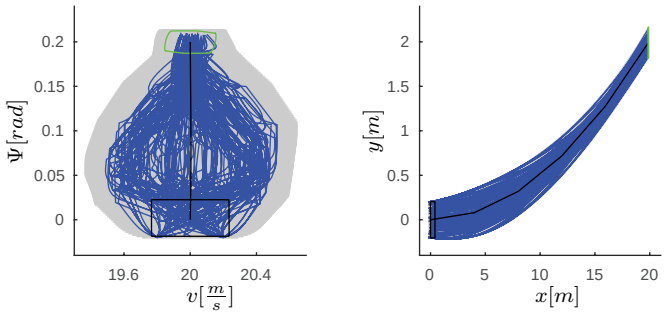


Fig. 9. Nonlinear car example: 200 simulations of the controller for the "turn left" maneuver are shown in blue.

set for the x, y plane and a large part of the v, Ψ plane. This demonstrates that our approach is not overly conservative, but that the reachable sets are justified by actual disturbance effects. In the same fashion, we can compute other maneuvers such as "drive straight" or "turn right." Since the shifted final sets always fit in the initial set, we can concatenate all of them with each other and obtain a fully connected maneuver automaton. Due to the dynamics' invariance with respect to the initial position and orientation, we would only need to discretize the velocity dimension if we want to compute a maneuver automaton for many possible initial states.

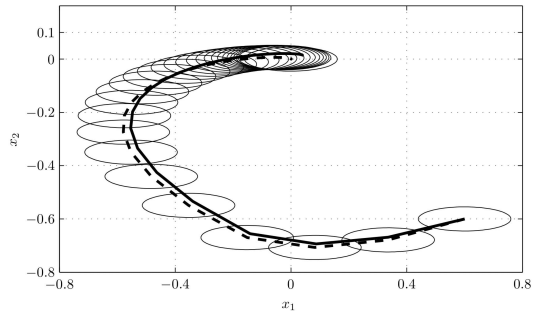
C. Nonlinear Example: Comparison with Tube-Based MPC

In our third example, we apply our approach to the example from [77]. The system dynamics are given by

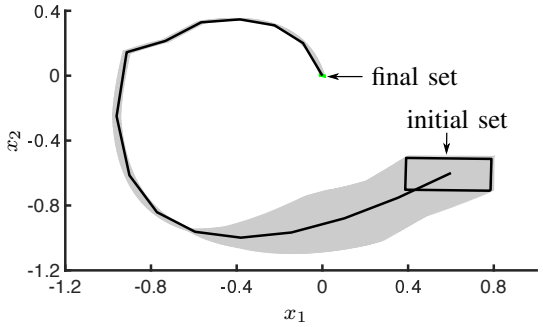
$$\dot{x}_1 = -x_1 + 2x_2 + 0.5u,$$
$$\dot{x}_2 = -3x_1 + 4x_2 - 0.25x_2^3 - 2u + w,$$

with the constrained control input $u(\cdot) \in [-2, 2]$ and the bounded disturbance $w(\cdot) \in [-0.1, 0.1]$. The authors from [77] solve the control problem of stabilizing the origin for this unstable system using a nonlinear tube-based MPC controller, which is applicable for continuous-time nonlinear systems. The resulting tube for a single example is shown in Fig. 10(a). We solve the same problem with our approach, where we start from an initial set which contains their control invariant set. The results are shown in Fig. 10(b) and take less than $30s$ to compute. We see that the system with our controller converges to a small set around the origin. Our reachable sets are thus

much smaller than the control invariant set from [77], which determines the size of the tube. This shows the advantage of our controller based on the actual reachable sets rather than a fixed-size tube based on control invariant regions.



(a) Tube-based MPC solution; taken from [77].



(b) Solution from our algorithm.

Fig. 10. Comparison of tube-based MPC with our approach: Reference trajectory and control invariant sets of the tube-based MPC solution (a). Reachable set (gray) of our controller for the same example with initial set in black and final set in blue (b). The initial set of (a) is contained in the initial set of (b).

D. Planning Example: Car Model Based on Real Data

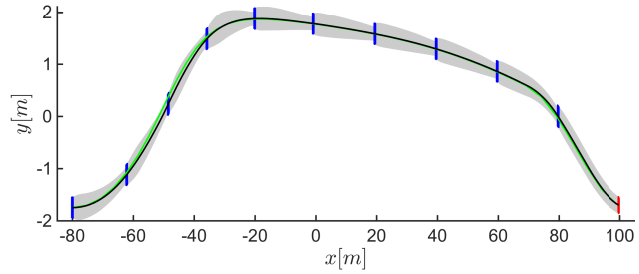


Fig. 11. Double lane-change maneuver from [78] driven using motion primitives with our new controller along planned trajectory (green). Initial sets of motion primitives are shown in blue, final sets in red, reachable sets in gray, and their reference trajectories in black.

As a last example, we revisit the planning problem from [78] to demonstrate how our approach can be applied to trajectory planning and control of real vehicles. The main question when modeling a real system is how to find a model which captures all real world behavior. Since this is not possible with a deterministic model, we use a model with uncertainties and aim on finding one which includes the real-world behavior in an over-approximative fashion. In [78], we use conformance testing

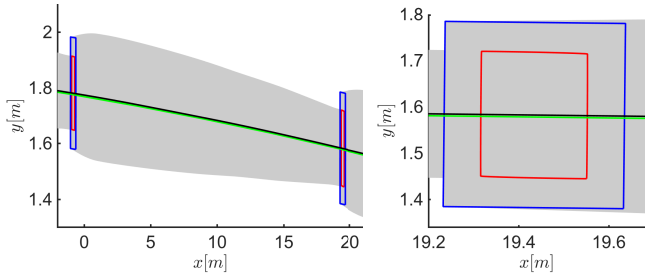


Fig. 12. Zoom into a single motion primitive: Full motion primitive on the left, final set (red) with initial set from following maneuver (blue) on the right. Initial set is a box rotated by the orientation of the reference trajectory.

to find such a model for an autonomous car. This is done by taking a number of test drives with a real vehicle in addition to simulations of a high-fidelity model and recording the applied inputs and the measured outputs. In a second step, we increase the sets for external disturbances and measurement noise until all measured behavior can be reproduced by our uncertain model. Doing this allows us to increase the confidence in our model, as it is able to represent the reality for all of these test cases.

The resulting model is given by

$$\begin{aligned}\dot{v} &= a + w_a,\\ \dot{\Psi} &= \frac{v}{l\left(1+\left(\frac{v}{v_{ch}}\right)^2\right)}(\delta + w_\delta),\\ \dot{x} &= v\,\cos(\Psi),\\ \dot{y} &= v\,\sin(\Psi),\end{aligned}$$

with the same states as before in Sec. VI.B, the control inputs acceleration a and steering angle δ, and disturbances $w_a \in [-0.1, 0.1]\frac{m}{s^2}$ and $w_\delta \in [-0.5, 0.5]°$. In addition, we assume that there is an additive measurement uncertainty for each state, with $\nu_v \in [-0.075, 0.075]\frac{m}{s}$, $\nu_\Psi \in [-0.3, 0.3]°$, and $\nu_x \in [-0.025, 0.025]m$, $\nu_y \in [-0.025, 0.025]m$. The characteristic velocity v_{ch} is a parameter which computes as $v_{ch} = \sqrt{\frac{l^2 c_f c_r}{m(c_r l_r - c_f l_f)}}$, with c_f, c_r denoting the cornering stiffness of the front and rear wheels, l_f, l_r the distances between the front and rear axis to the center of gravity, $l = l_f + l_r$ their sum, and m the vehicle mass [82]. We also consider an input constraint which results from the friction circle

$$\sqrt{a_{long}^2 + a_{lat}^2} \leq a_{max},$$

with the longitudinal acceleration $a_{long} = a + w_a$ and lateral acceleration $a_{lat} = v\dot{\theta} = \frac{v^2}{l\left(1+\left(\frac{v}{v_{ch}}\right)^2\right)}(\delta + w_\delta)$, both with respect to the orientation of the vehicle.

For the online planning in [78], we first use a simplified point-mass model to obtain a planned trajectory which considers other traffic participants and match this planned trajectory using motion primitives which have been computed using the conformant model, thereby ensuring the drivability of the resulting maneuver.

Here, we use our new approach to compute the required motion primitives for the planning problem and show in Fig. 11 the result for a double-lane change maneuver. Each motion primitive has a duration of $2s$, and they are computed such that for each motion primitive, the shifted final set lies inside the initial set. In Fig. 12, we show a zoom into a single motion primitive and see that the final set of this motion primitive is actually contained inside the initial set of the following. As in the previous examples, the reachable sets from our new approach are again smaller than the ones in [78].

VII. Conclusion

We present a novel set-based control algorithm which guarantees the satisfaction of constraints for disturbed, nonlinear systems by optimizing over reachable sets. In contrast to existing approaches, reachability analysis is not used as a pure verification tool, but for optimization in the controller synthesis. This allows us to obtain controllers with smaller reachable sets, despite the presence of external disturbances and measurement noise, and the restriction through constraints. Our approach is particularly suited for the generation of maneuver automata, which can be used for efficient online planning. In contrast to existing approaches, we introduce a state-dependent feedforward controller based on zonotopes, which steers all states of an initial set to the desired final set. The feedback controller only has to counteract disturbances and linearization errors, not initial deviation, and therefore can be much more aggressive than for classical approaches. The resulting controller is a time-varying linear controller, which is easy to implement and allows for fast execution times. We show in four numerical examples the advantages of our new approach, where we compare it to three of our older approaches and a tube-based MPC approach. Our new approach leads to much smaller reachable sets and can handle much larger constraints compared to the other approaches.

Acknowledgment

The author gratefully acknowledges financial support from the European Commission project UnCoVerCPS under grant number 643921.

References

[1] H. G. Bock and K.-J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 1603–1608, 1984.

[2] M. Althoff, S. Bak, D. Cattaruzza, X. Chen, G. Frehse, R. Ray, and S. Schupp, "ARCH-COMP17 category report: Continuous and hybrid systems with linear continuous dynamics," in *Proc. of the 4th International Workshop on Applied Verification of Continuous and Hybrid Systems*, 2017, pp. 143–159.

[3] X. Chen, M. Althoff, and F. Immler, "ARCH-COMP17 category report: Continuous systems with nonlinear dynamics," in *Proc. of the 4th International Workshop on Applied Verification of Continuous and Hybrid Systems*, 2017, pp. 160–169.

[4] E. Asarin, T. Dang, G. Frehse, A. Girard, C. Le Guernic, and O. Maler, "Recent progress in continuous and hybrid reachability analysis," in *Proc. of the IEEE Conference on Computer Aided Control Systems Design*, 2006, pp. 1582–1587.

[5] D. Heß, M. Althoff, and T. Sattel, "Formal verification of maneuver automata for parameterized motion primitives," in *Proc. of the International Conference on Intelligent Robots and Systems*, 2014, pp. 1474–1481.

[6] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific Belmont, MA, 2005.

[7] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Springer, 2008.

[8] J. Lygeros, C. Tomlin, and S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, no. 3, pp. 349–370, 1999.

[9] A. B. Kurzhanski, I. M. Mitchell, and P. Varaiya, "Optimization techniques for state-constrained control and obstacle problems," *Journal of Optimization Theory and Applications*, vol. 128, no. 3, pp. 499–521, 2006.

[10] F. Borrelli, *Constrained Optimal Control of Linear and Hybrid Systems*. Springer, 2003.

[11] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for linear and hybrid systems*. Cambridge University Press, 2015.

[12] M. de la Pena, A. Bemporad, and C. Filippi, "Robust explicit MPC based on approximate multi-parametric convex programming," in *Proc. of the 43rd Conference on Decision and Control*, 2004, pp. 2491–2496.

[13] E. C. Kerrigan and J. M. Maciejowski, "Feedback min-max model predictive control using a single linear program: robust stability and the explicit solution," *International Journal of Robust and Nonlinear Control*, vol. 14, pp. 395–413, 2004.

[14] A. Alessio and A. Bemporad, *A Survey on Explicit Model Predictive Control*. Springer, 2009, pp. 345–369.

[15] A. Grancharova, T. A. Johansen, and P. Tøndel, *Computational Aspects of Approximate Explicit Nonlinear Model Predictive Control*. Springer, 2007, pp. 181–192.

[16] E. N. Pistikopoulos, "Perspectives in multiparametric programming and explicit model predictive control," *AIChE Journal*, vol. 55, no. 8, pp. 1918–1925, 2009.

[17] D. Raimondo, S. Riverso, C. Jones, and M. Morari, "A robust explicit nonlinear MPC controller with input-to-state stability guarantees," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 9284 – 9289, 2011.

[18] D. Q. Mayne, M. M. Seron, and S. V. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219 – 224, 2005.

[19] S. V. Raković, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2746–2761, 2012.

[20] S. V. Raković, B. Kouvaritakis, R. Findeisen, and M. Cannon, "Homothetic tube model predictive control," *Automatica*, vol. 48, no. 8, pp. 1631–1638, 2012.

[21] M. Rubagotti, D. M. Raimondo, A. Ferrara, and L. Magni, "Robust model predictive control with integral sliding mode in continuous-time sampled-data nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 556–570, 2011.

[22] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgöwer, "Robust model predictive control for nonlinear discrete-time systems," *International Journal of Robust and Nonlinear Control*, vol. 13, no. 3-4, pp. 229–246, 2003.

[23] D. Q. Mayne, E. C. Kerrigan, E. J. van Wyk, and P. Falugi, "Tube-based robust nonlinear model predictive control," *International Journal of Robust and Nonlinear Control*, vol. 21, no. 11, pp. 1341–1353, 2011.

[24] S. Singh, A. Majumdar, J.-J. Slotine, and M. Pavone, "Robust online motion planning via contraction theory and convex optimization," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2017, pp. 5883–5890.

[25] J. Bravo, T. Alamo, and E. Camacho, "Robust MPC of constrained discrete-time nonlinear systems based on approximated reachable sets," *Automatica*, vol. 42, no. 10, pp. 1745 – 1751, 2006.

[26] A. A. Julius and A. K. Winn, "Safety controller synthesis using human generated trajectories: Nonlinear dynamics with feedback linearization and differential flatness," in *Proc. of the American Control Conference*, 2012, pp. 709–714.

[27] A. K. Winn and A. A. Julius, "Feedback control law generation for safety controller synthesis," in *Proc. of the 52nd Conference on Decision and Control*, 2013, pp. 3912–3917.

[28] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[29] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[30] P. Reist and R. Tedrake, "Simulation-based lqr-trees with input and state constraints," in *Proc. of the International Conference on Robotics and Automation*, 2010, pp. 5504–5510.

[31] A. Majumdar and R. Tedrake, "Robust online motion planning with regions of finite time invariance," in *Algorithmic Foundations of Robotics X*. Springer, 2013, pp. 543–558.

[32] A. Weiss, C. Danielson, K. Berntorp, I. Kolmanovsky, and S. Di Cairano, "Motion planning with invariant set trees," in *Proc. of the Conference on Control Technology and Applications*, 2017, pp. 1625–1630.

[33] I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia, "Automated composition of motion primitives for multi-robot systems from safe LTL specifications," in *Proc. of the International Conference on Intelligent Robots and Systems*, 2014, pp. 1525–1532.

[34] R. G. Sanfelice and E. Frazzoli, "A hybrid control framework for robust maneuver-based motion planning," in *Proc. of the American Control Conference*, 2008, pp. 2254–2259.

[35] J. H. Gillula, G. M. Hoffmann, H. Huang, M. P. Vitus, and C. J. Tomlin, "Applications of hybrid reachability analysis to robotic aerial vehicles," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 335–354, 2011.

[36] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[37] M. Zamani, G. Pola, M. Mazo Jr., and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1804–1809, 2012.

[38] J. A. DeCastro and H. Kress-Gazit, "Synthesis of nonlinear continuous controllers for verifiably correct high-level, reactive behaviors," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 378–394, 2015.

[39] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.

[40] A. Girard and S. Martin, "Motion planning for nonlinear systems using hybridizations and robust controllers on simplices," in *Proc. of the 47th Conference on Decision and Control*, 2008, pp. 239–244.

[41] A. Girard, "Controller synthesis for safety and reachability via approximate bisimulation," *Automatica*, vol. 48, no. 5, pp. 947–953, 2012.

[42] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[43] Y. Li, J. Liu, and N. Ozay, "Computing finite abstractions with robustness margins via local reachable set over-approximation," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 1–6, 2015.

[44] J. Liu, N. Ozay, U. Topcu, and R. M. Murray, "Synthesis of reactive switching protocols from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 58, no. 7, pp. 1771–1785, 2013.

[45] J. Liu and N. Ozay, "Finite abstractions with robustness margins for temporal logic-based control synthesis," *Nonlinear Analysis: Hybrid Systems*, vol. 22, pp. 1 – 15, 2016.

[46] P. Nilsson and N. Ozay, "Incremental synthesis of switching protocols via abstraction refinement," in *Proc. of the 53rd Conference on Decision and Control*, 2014, pp. 6246–6253.

[47] N. Ozay, J. Liu, P. Prabhakar, and R. M. Murray, "Computing augmented finite transition systems to synthesize switching protocols for polynomial switched systems," in *Proc. of the American Control Conference*, 2013, pp. 6237–6244.

[48] G. Pola, A. Girard, and P. Tabuada, "Symbolic models for nonlinear control systems using approximate bisimulation," in *Proc. of the 46th Conference on Decision and Control*, 2007, pp. 4656–4661.

[49] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia, "Reactive synthesis from signal temporal logic specifications," in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, 2015, pp. 239–248.

[50] M. Rungger, M. Mazo Jr., and P. Tabuada, "Specification-guided controller synthesis for linear systems and safe linear-time temporal logic," in *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control*. ACM, 2013, pp. 333–342.

[51] S. Sadraddini and C. Belta, "Safety control of monotone systems with bounded uncertainties," in *Proc. of the 55th Conference on Decision and Control*, 2016, pp. 4874–4879.

[52] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning," *Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, 2012.

[53] M. Zamani, A. Abate, and A. Girard, "Symbolic models for stochastic switched systems: A discretization and a discretization-free approach," *Automatica*, vol. 55, pp. 183–196, 2015.

[54] E. M. Wolff, U. Topcu, and R. M. Murray, "Automaton-guided controller synthesis for nonlinear systems with temporal logic," in *Proc. of the International Conference on Intelligent Robots and Systems*, 2013, pp. 4332–4339.

[55] E. M. Wolff and R. M. Murray, "Optimal control of nonlinear systems with temporal logic specifications," in *Robotics Research*. Springer, 2016, pp. 21–37.

[56] J. A. DeCastro and H. Kress-Gazit, "Nonlinear controller synthesis and automatic workspace partitioning for reactive high-level behaviors," in *Proc. of Hybrid Systems: Computation and Control*, 2016.

[57] I. Papusha, J. Fu, U. Topcu, and R. M. Murray, "Automata theory meets approximate dynamic programming: Optimal control with temporal logic constraints," in *Proc. of the 55th Conference on Decision and Control*, 2016, pp. 434–440.

[58] C. Verdier and M. Mazo Jr., "Formal controller synthesis via genetic programming," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7205 – 7210, 2017, 20th IFAC World Congress.

[59] B. Schürmann and M. Althoff, "Convex interpolation control with formal guarantees for disturbed and constrained nonlinear systems," in *Proc. of the International Conference on Hybrid Systems: Computation and Control*, 2017, pp. 121–130.

[60] ——, "Guaranteeing constraints of disturbed nonlinear systems using set-based optimal control in generator space," in *Proc. of the 20th IFAC World Congress*, 2017, pp. 12 020–12 027.

[61] ——, "Optimal control of sets of solutions to formally guarantee constraints of disturbed linear systems," in *Proc. of the American Control Conference*, 2017, pp. 2522–2529.

[62] E. Frazzoli, "Robust hybrid control for autonomous vehicle motion planning," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[63] K. J. Aström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2010.

[64] A. Platzer and E. M. Clarke, "The image computation problem in hybrid systems model checking," in *Proc. of Hybrid Systems: Computation and Control*, 2007, pp. 473–486.

[65] M. Rungger and M. Zamani, "Accurate reachability analysis of uncertain nonlinear systems," in *Proc. Hybrid Systems: Computation and Control*. ACM, 2018, pp. 61–70.

[66] M. Althoff and G. Frehse, "Combining zonotopes and support functions for efficient reachability analysis of linear systems," in *Proc. of the 55th IEEE Conference on Decision and Control*, 2016.

[67] P. Gritzmann and B. Sturmfels, "Minkowski addition of polytopes: computational complexity and applications to Gröbner bases," *SIAM Journal on Discrete Mathematics*, vol. 6, pp. 246–269, 1993.

[68] G. B. Dantzig and M. N. Thapa, *Linear Programming 1: Introduction*. Springer, 2006.

[69] M. Grant, S. Boyd, and Y. Ye, "Disciplined convex programming," in *Global Optimization: from Theory to Implementation*. Springer, 2006, pp. 155–210.

[70] H. Kwakernaak and R. Sivan, *Linear optimal control systems*. Wiley-interscience New York, 1972.

[71] S. Magdici and M. Althoff, "Fail-safe motion planning of autonomous vehicles," in *Proc. of the 19th International IEEE Conference on Intelligent Transportation Systems*, 2016, pp. 452–458.

[72] O. Schütze, "Set oriented methods for global optimization," Ph.D. dissertation, Univ. Paderborn, 2004.

[73] F. L. Lewis, *Optimal Control*. Wiley, 1986.

[74] E. B. Lee and L. Markus, *Foundations of Optimal Control Theory*. Wiley, 1967.

[75] H. Zhang, A. Geiger, and R. Urtasun, "Understanding high-level semantics by modeling traffic patterns," in *Proc. of the IEEE International Conference on Computer Vision*, 2013, pp. 3056–3063.

[76] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[77] S. Yu, H. Chen, and F. Allgöwer, "Tube MPC scheme based on robust control invariant set with application to lipschitz nonlinear systems," in *Proc. of the 50th Conference on Decision and Control and European Control Conference*, 2011, pp. 2650–2655.

[78] B. Schürmann, D. Heß, J. Eilbrecht, O. Stursberg, F. Köster, and M. Althoff, "Ensuring drivability of planned motions using formal methods," in *Proc. of the Intelligent Transportation Systems Conference*, 2017, pp. 1661–1668.

[79] M. Althoff, "An introduction to CORA 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015, pp. 120–151.
[80] B. Houska, H. Ferreau, and M. Diehl, "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.
[81] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.
[82] M. Werling and D. Liccardo, "Automatic collision avoidance using model-predictive online optimization," in *Proc. of the 51st IEEE Conference on Decision and Control*, 2012, pp. 6309–6314.

Bastian Schürmann is a Ph.D. candidate in Computer Science at Technische Universität München, Germany. He received a Bachelor of Science in Electrical and Computer Engineering from Technische Universität Kaiserslautern, Germany in 2012; a Master of Science in Electrical Engineering from the University of California, Los Angeles, USA in 2014; and a Master of Science in Engineering Cybernetics from Universität Stuttgart, Germany in 2015. In 2018, he was a visiting student researcher at the California Institute of Technology. His research focuses on combining control theory, reachability analysis, and optimization. Application scenarios include the controlling of safety-critical systems, such as self-driving vehicles and robots collaborating with humans.

Matthias Althoff is an assistant professor in computer science at Technische Universität München, Germany. He received his diploma engineering degree in Mechanical Engineering in 2005, and his Ph.D. degree in Electrical Engineering in 2010, both from Technische Universität München, Germany. From 2010 to 2012, he was a postdoctoral researcher at Carnegie Mellon University, Pittsburgh, USA, and from 2012 to 2013 an assistant professor at Technische Universität Ilmenau, Germany. His research interests include formal verification of continuous and hybrid systems, reachability analysis, planning algorithms, nonlinear control, automated vehicles, and power systems.