2015 IEEE Intelligent Vehicles Symposium (IV)
June 28 - July 1, 2015. COEX, Seoul, Korea

Stochastic Model Predictive Controller with Chance Constraints for Comfortable and Safe Driving Behavior of Autonomous Vehicles

David Lenz[1] and Tobias Kessler[1] and Alois Knoll[2]

***Abstract*— In this paper, we address the application of stochastic model predictive control with chance constraints to autonomous driving. We use a condensed formulation of a linearized vehicle model to setup a quadratic program with nonlinear chance constraints, which can be solved with off-the-shelf optimization algorithms. We further show how obstacle information in the path planning stage can be converted into a set of linear state constraints that can be directly used in the control algorithm. The resulting controller is potentially real-time capable and achieves a tradeoff between safety and comfort in its control behavior.**

I. INTRODUCTION

We consider a vehicle model with stochastic disturbances, actuator limits and chance constraint states. The actuators, e.g. the steering wheel angle are physically limited and these limits occur in everyday driving situations. Furthermore actuator and especially sensor values are obtained with stochastic disturbances, which yields uncertainties in the states.

In general, the controller of an autonomous vehicle receives a reference trajectory from the planning layer and does not have direct access to obstacle information from the sensor fusion. The planner is assumed to generate a collision free trajectory with a certain safety margin. A Linear Quadratic Regulator (LQR) addresses this control problem. As obstacles are not taken into account directly, one has to tune this controller to either obtain a good tracking behavior or a comfortable parametrization. A safe parameter set accepts high control inputs to stay as close to the reference as possible, while a strategy aiming at driving comfort tends to minimize the control energy. Therefore, the tracking behavior influences the choice of safety margins in the planning layer. A combination of both control strategies is necessary, usually depending on the current situation. This trade-off between tracking quality and driving comfort only serves as an example for conflicting goals. It is desirable to include even more (conflicting) requirements in the controller design and to realize different behaviors for varying situations.

In this work we calculate a simplified obstacle representation in the form of state constraints that can be used in the control algorithm. Our proposed approach, which we call Chance Constraint Stochastic Model Predictive Controller (CC-SMPC), includes these constraints in the problem definition and therefore is safe by design. In the parametrization step, only comfort issues have to be taken into account. Fig. 1 sketches the tracking behaviors and covariances of the different controllers. Summing up, this paper addresses the following challenges:

State constraints construction: A new method for obtaining convex linear state constraints from obstacles

Simplified controller parametrization: Only comfort influences choice of parameters

Separation of design steps: Controller performance does not influence planning layer directly (e.g. safety margins)

Stochastic system capabilities: A chance constraint system in the controller design

This paper is organized as follows. In section II we give a literature overview. Section III formally defines the stochastic system and introduces the optimization problem. A method for constructing state constraints is shown in section IV. The controller design and the optimization formulation are given in section V. Section VI states simulation results with a single track car model and section VII concludes this work.

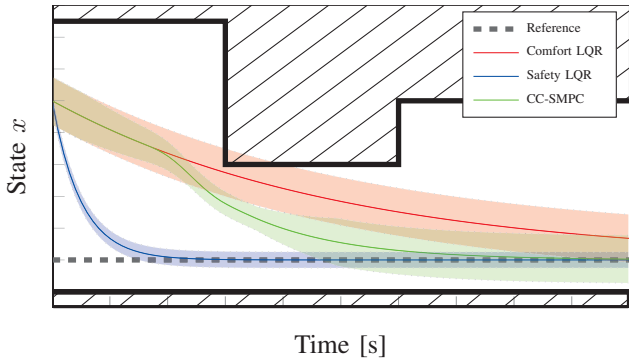


Fig. 1. Schematic comparison of the CC-SMPC control approach to LQR controllers designed to be as safe and as comfortable as possible. The hatched patches depict obstacles and the semi-transparent areas the current error covariances.

II. RELATED WORK

In the literature, uncertainties are often only treated in the planning layer, as for example in [1]. The controller is assumed to follow the generated trajectory with known disturbances. We instead aim at dealing with the disturbances and uncertainties in the controller itself. Similar approaches can be found in the literature. All aim to minimize a cost function with respect to the system dynamics equations. From a control theory point, several approaches deal with inequality constraints in an optimal control scenario. In [2] the authors provide a suboptimal solution for the finite and infinite horizon LQR problem with inequality constraints on the system inputs and outputs.

[1]David Lenz and Tobias Kessler are with fortiss GmbH, An-Institut Technische Universität München, Munich, Germany

[2]Alois Knoll is with Robotics and Embedded Systems, Technische Universität München, Munich, Germany

In the literature we find several ways to approximate chance constraints [3]. A comparison of an approximation approach using Boole's inequality and ellipsoid relaxation is given in [4]. Both yield computationally solvable results, while the Boole's inequality approach tends to complicate the optimization problem and the ellipsoid approximation approach yields too conservative approximations. [5] introduces the multi-dimensional Chebyshev inequality to approximate the chance constraints. The subset defined by the inequality constraints is approximated by an maximum volume inscribed ellipsoid. The constraints tightening method [6] is referenced by [7] to transform the chance constraint into deterministic constraints if each constraint can be treated individually. This is achieved by a so-called risk allocation stage, which is another optimization algorithm. Another approach to transform the probabilistic problem into a deterministic one is given by [8]. They encode the covariance matrices in the state and include them in the optimization problem. Note that there exist several more approaches for the constraint approximation like the Bernstein approximation or the scenario approximation [3]. In contrast to other work, we use the nonlinear chance constraints directly, see section V-B. Also, we limit the chance constraints to the system states. A more general problem formulation can be found in [9]. Similar to the mentioned approaches above, the chance constraints formulation and optimization is also applied to higher level planning as in [10] or for multi-agent control as in [11].

Many publications discuss how to minimize the computational complexity and construct or adopt a special optimization algorithm. [12] introduces a sparse condensed formulation which has nice structural properties of the optimization problem. [13] transforms the constraint model predictive control problem to a quadratic control problem making use of the Moore-Penrose pseudo-inverse of the input matrix B. The resulting problem is solved with an interior point method. [14] constructs an efficient algorithm to solve constraint LQR problems based on the proof that the constructed controller is piece-wise affine. Computationally heavy parts of the algorithm are computed offline. We aim at using a standard optimization algorithm with an appropriate problem structure, see section V.

III. PROBLEM DEFINITION

In this section, we first introduce the stochastic system definition and the single-track model used in this paper. Subsequently we formally state the optimization problem.

A. System Definition

The stochastic dynamic time-discrete model of a (mobile robotic) system can be expressed as

$$\hat{\boldsymbol{x}}_{k+1} = f(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{u}}_k, \boldsymbol{w}_k), \quad \boldsymbol{w}_k \sim N(0, \boldsymbol{\Sigma}_{\boldsymbol{w},k}) \tag{1}$$

where $\hat{\boldsymbol{x}}_k \in \widehat{\boldsymbol{\mathcal{X}}}$ denotes the state, $\hat{\boldsymbol{u}}_k \in \widehat{\boldsymbol{\mathcal{U}}}$ the control input, and $\boldsymbol{w}_k$ is the error characteristics of the motion model. The (possibly nonlinear) model described in (1) can be linearized around a nominal trajectory $(\boldsymbol{x}_k^*, \boldsymbol{u}_k^*, \boldsymbol{w}_k^* = 0)$ given from a motion planning algorithm. Only considering the deviations from this reference $\boldsymbol{x}_k := \hat{\boldsymbol{x}}_k - \boldsymbol{x}_k^*$, $\boldsymbol{u}_k := \hat{\boldsymbol{u}}_k - \boldsymbol{u}_k^*$, the system can be approximatly written as

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{W}_k \boldsymbol{w}_k \tag{2}$$

$$\text{with } \boldsymbol{A}_k = \frac{\partial f}{\partial \hat{\boldsymbol{x}}_k}, \ \boldsymbol{B}_k = \frac{\partial f}{\partial \hat{\boldsymbol{u}}_k}, \ \boldsymbol{W}_k = \frac{\partial f}{\partial \boldsymbol{w}_k}. \tag{3}$$

The covariance of the deviation propagates with:

$$\boldsymbol{\Sigma}_{x,k+1} = \boldsymbol{A}_k \boldsymbol{\Sigma}_{\boldsymbol{x},k} \boldsymbol{A}_k^\intercal + \boldsymbol{W}_k \boldsymbol{\Sigma}_{\boldsymbol{w},k} \boldsymbol{W}_k^\intercal . \tag{4}$$

Note, that we assume the control input $\boldsymbol{u}$ to be deterministic.

B. Single-Track Car Model

As a car dynamics model, we use a form of the single-track model with steering curvature σ and the acceleration a as control inputs. x and y denote the position of the car's rear axle, θ the heading. Following the notation of section III-A, we define $\hat{\boldsymbol{x}}_k := (x_k, y_k, \theta_k, v_k)$, $\hat{\boldsymbol{u}}_k := (\sigma_k, a_k)$, and $\hat{\boldsymbol{w}}_k := (w_{1,k}, w_{2,k})$. The discretized system dynamics with sample time Δt can be formulated as

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ y_k \\ \theta_k \\ v_k \end{pmatrix} + \Delta t \begin{pmatrix} v_k \cos\theta_k \\ v_k \sin\theta_k \\ v_k(\sigma_k + w_{1,k}) \\ a_k + w_{2,k} \end{pmatrix} \tag{5}$$

$$\hat{\boldsymbol{x}}_{k+1} = f(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{u}}_k, \hat{\boldsymbol{w}}_k, \Delta t) \tag{6}$$

and rewritten in terms of the deviations as

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k, \Delta t). \tag{7}$$

The control algorithm itself is not limited to this vehicle model but is generally applicable for stochastic systems.

C. Optimization Problem Definition

For the system (2) we define the constrained minimization problem

$$\min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_N} J(\boldsymbol{u}_1, ..., \boldsymbol{u}_N, \boldsymbol{x}_1, ...\boldsymbol{x}_N) \tag{8}$$

$$\text{subject to } \boldsymbol{u}_k \in F_{\boldsymbol{u},k} \tag{9}$$

$$P\Big(\bigcap_{k=1}^{N} \{\boldsymbol{x}_k \in F_{\boldsymbol{x},k}\}\Big) \geq \alpha \tag{10}$$

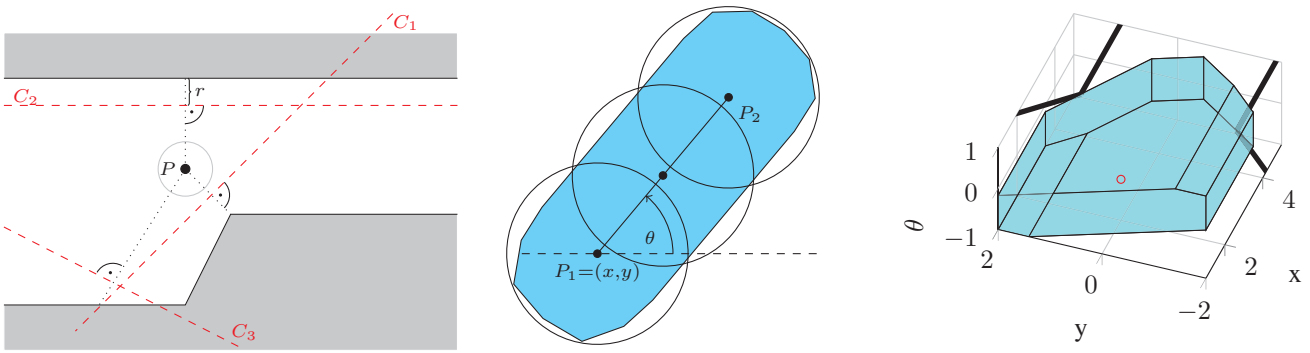
with the sets $F_{\boldsymbol{x},k}$ and $F_{\boldsymbol{u},k}$

$$F_{\boldsymbol{x},k} := \{\boldsymbol{x}_k \in \boldsymbol{\mathcal{X}} | \boldsymbol{t}_{i,k,\boldsymbol{x}} \boldsymbol{x}_k \leq s_{i,k,\boldsymbol{x}} \forall i \in [1..M_{\boldsymbol{x}}]\} \tag{11}$$

$$F_{\boldsymbol{u},k} := \{\boldsymbol{u}_k \in \boldsymbol{\mathcal{U}} | \boldsymbol{t}_{i,k,\boldsymbol{u}} \boldsymbol{u}_k \leq s_{i,k,\boldsymbol{u}} \forall i \in [1..M_{\boldsymbol{u}}]\}. \tag{12}$$

$\alpha \in [0..1[$ expresses the probability all linear state constraints are satisfied. $F_{\boldsymbol{x},k}$ denotes the set of all valid states $\boldsymbol{x}$ at step k. $\boldsymbol{t}$ and s are the parameters defining one linear constraint and are assumed to be deterministic. (10) is probabilistic and will be referred to as chance constraint. The cost function J is chosen as the quadratic function

$$J(\boldsymbol{u}_1, ..., \boldsymbol{u}_N, \boldsymbol{x}_1, ...\boldsymbol{x}_N) = E\Big\{ \sum_{k=0}^{N-1} (\boldsymbol{x}_k^\intercal \boldsymbol{Q}_k \boldsymbol{x}_k + \boldsymbol{u}_k^\intercal \boldsymbol{R}_k \boldsymbol{u}_k) + \boldsymbol{x}_N^\intercal \boldsymbol{Q}_N \boldsymbol{x}_N \Big\}. \tag{13}$$



(a) Local convexification method

(b) Approximation of the vehicle shape with three circles

(c) Constraint polyhedron

Fig. 2. Methods for creating linear state constraints for a vehicle shape. First a local convexification of the obstacle information (a) is performed for every circular disc of the vehicle shape approximation (b). The position constraints are transformed into state constraints considering the orientation θ. The result is a convex region (polyhedron) in the state space (c).

IV. CONSTRAINT DETERMINATION

In contrast to most control theory publications, we also address the question where constraints originate from.

First of all, there are actuator saturations which come from the mechanical design and are physical limits. In case of a vehicle, this is the limited steering angle and the limited braking and acceleration ability. These constraints are generally constant or only changing slowly.

Secondly, state constraints exist that depend on the current position in the environment and the obstacle geometries around that point. These can be *virtual* constraints, i.e., constraints that a desired tracking error should not be exceeded or *physical* constraints whose violation leads to a crash with an obstacle.

Here we focus on *physical* state constraints, as these are most complex but can be precalculated from a trajectory planning layer along a reference. The transformation of obstacles into a set of linear, convex, but time-dependent constraints beforehand allows the optimization problem for model-predictive control to be solved efficiently. We propose two steps to obtain the state constraints for a given reference point $\boldsymbol{x}_k^*$. First, a local convexification method for position and afterwards a transformation of multiple position constraints into a set of state constraints. This process is summarized in Fig. 2 and explained in the subsequent sections.

A. Local Convexification

For a local convex description of the environment for a disc at point P and with radius r, we apply an adapted algorithm from [15] extended for discs. The process is depicted in Fig. 2(a) and can be described briefly as follows. First, we find the closest obstacle to point P and draw a line C_1 perpendicular to the line connecting P and the closest obstacle point (dotted) with distance r from the obstacle. This yields a linear constraint. We proceed by finding the closest point on an obstacle that is not already covered by the existing linear constraints and repeat this process. We proceed iteratively until all obstacles are accounted for in the set of constraints. In [15] the authors note, that this method is not the least conservative approximation of the environment. As we are not limited by the method of construction for the local convex hull, any other method can be used here as well.

B. From Position to State Constraints

We approximate the polygonal shape of the vehicle with a series of circles with equal radius of r as shown in Fig. 2(b). For each circle, we transform the positional constraints into state constraints. The process is presented for the circle around P_2. l denotes the distance of P_1 and P_2. First, we apply the local convexification around the point P_2 and get a set of linear constraints on the position of P_2. For brevity, we omit all subscripts for the linear constraints and only consider one constraint. We call this positional constraint $\hat{\boldsymbol{t}} P_2 \leq \hat{s}$ which is equivalent to

$$\hat{\boldsymbol{t}} P_1 + \hat{\boldsymbol{t}} \begin{pmatrix} l\cos\theta \\ l\sin\theta \end{pmatrix} \leq \hat{s}\,. \tag{14}$$

Now, (14) includes the three kinematic states (x, y, θ) we are interested in for collision checking, but the relationship is nonlinear. As we are interested in the constraints along the reference trajectory, we linearize around θ^*, apply small-angle approximations and get

$$\hat{\boldsymbol{t}} P_1 + \underbrace{l\hat{\boldsymbol{t}} \begin{pmatrix} \cos\theta^* + \sin\theta^*\theta^* \\ \sin\theta^* - \cos\theta^*\theta^* \end{pmatrix}}_{constant} + \underbrace{l\hat{\boldsymbol{t}} \begin{pmatrix} -\sin\theta^* \\ \cos\theta^* \end{pmatrix}}_{:=t_\theta} \theta \leq \hat{s}\,. \tag{15}$$

With an appropriate choice for s, this can be written as

$$\underbrace{(\hat{\boldsymbol{t}}, t_\theta, 0)}_{\boldsymbol{t}} \boldsymbol{x} \leq s\,, \tag{16}$$

which is exactly the constraint form we defined in section III-C. Note, that the last zero in (16) indicates that we are not imposing constraints on the velocity. In a dynamically changing environment also constraints on the velocity will appear. Repeating this procedure for all used discs, the resulting set of constraints form a polyhedron in the state space. An example is shown in Fig. 2(c).

V. MODEL PREDICTIVE CONTROLLER

In this section, we reformulate the optimization problem described in section III in order to solve it efficiently with an off-the-shelf optimization algorithm. We use a condensed formulation to obtain a quadratic program without constraints. We then show how we incorporate the nonlinear chance constraint within the optimization. For efficiency, all gradients and hessians are calculated analytically.

A. Block Matrix Formulation

We rearrange the equations from (4) and group them into the condensed formulation

$$\begin{aligned} \boldsymbol{\xi} &= \mathcal{A}x_0 + \mathcal{B}\boldsymbol{\nu} + \mathcal{W}\boldsymbol{\omega} \\ \mathcal{S}_{\boldsymbol{x}} &= \mathcal{A}\boldsymbol{\Sigma}_{\boldsymbol{x},0}\mathcal{A}^\intercal + \mathcal{W}\mathcal{S}_{\boldsymbol{w}}\mathcal{W}^\intercal \end{aligned} \tag{17}$$

with the vectors

$$\boldsymbol{\xi} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_N \end{pmatrix}, \boldsymbol{\nu} = \begin{pmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_{N-1} \end{pmatrix}, \boldsymbol{\omega} = \begin{pmatrix} \boldsymbol{w}_0 \\ \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_{N-1} \end{pmatrix} \tag{18}$$

and block matrices

$$\mathcal{A} = \begin{bmatrix} \boldsymbol{A}_0 \\ \boldsymbol{A}_1\boldsymbol{A}_0 \\ \vdots \\ \prod_{k=0}^{N-1} \boldsymbol{A}_k \end{bmatrix}$$

$$\mathcal{B} = \begin{bmatrix} \boldsymbol{B}_0 & 0 & \dots & 0 \\ \boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{B}_1 & 0 & \dots & 0 \\ \vdots & & & & \vdots \\ \prod_{k=1}^{N-1} \boldsymbol{A}_k\boldsymbol{B}_0 & & \dots & & \boldsymbol{B}_{N-1} \end{bmatrix}$$

$$\begin{aligned} &\mathcal{W} \text{ analog to } \mathcal{B} \\ &\mathcal{S}_{\boldsymbol{x}} = \operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{x},1}, \dots, \boldsymbol{\Sigma}_{\boldsymbol{x},N}) \\ &\mathcal{S}_{\boldsymbol{w}} = \operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{w},0}, \dots, \boldsymbol{\Sigma}_{\boldsymbol{w},N-1}) \\ &\mathcal{Q} = \operatorname{diag}(\boldsymbol{Q}, \dots, \boldsymbol{Q}, \boldsymbol{Q}_N),\ \mathcal{R} = \operatorname{diag}(\boldsymbol{R}, \dots, \boldsymbol{R}) . \end{aligned} \tag{19}$$

The cost function (13) is rewritten as

$$J(\boldsymbol{\nu}, \boldsymbol{\xi}) = E\{\boldsymbol{\xi}^\intercal \mathcal{Q}\boldsymbol{\xi} + \boldsymbol{\nu}^\intercal \mathcal{R}\boldsymbol{\nu}\} . \tag{20}$$

As we aim at optimizing $\boldsymbol{\nu}$, we can eliminate $\boldsymbol{\xi}$ from the cost function by plugging (17) into (20) and get

$$J = \boldsymbol{\nu}^\intercal(\mathcal{R} + \mathcal{B}^\intercal\mathcal{Q}\mathcal{B})\boldsymbol{\nu} + 2\boldsymbol{x}_0^\intercal\mathcal{A}^\intercal\mathcal{Q}\mathcal{B}\boldsymbol{\nu} + \underbrace{\boldsymbol{x}_0^\intercal\mathcal{A}^\intercal\mathcal{Q}\mathcal{A}\boldsymbol{x}_0}_{\text{constant}} . \tag{21}$$

B. Chance Constraints Transformation and Approximation

In this section, we show how to derive inequality constraints from the chance constraints (10). Note, that the expressed probabilities are valid for each state $\boldsymbol{x}_i$ and per time step k. Using the condensed formulation of (17) the constraints can be rewritten as

$$P\big(\boldsymbol{\xi} \in \mathcal{F}_{\boldsymbol{\xi}}\big) \geq \alpha, \quad \mathcal{F}_{\boldsymbol{\xi}} : \bigcap_{i=1}^{M_{\boldsymbol{x}}} \{\boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\boldsymbol{\xi} \leq s_{i,\boldsymbol{x}}\} \tag{22}$$

$$\boldsymbol{\nu} \in \mathcal{F}_{\boldsymbol{\nu}}, \quad \mathcal{F}_{\boldsymbol{\nu}} : \bigcap_{i=1}^{M_{\boldsymbol{u}}} \{\boldsymbol{t}_{i,\boldsymbol{\nu}}^\intercal\boldsymbol{\nu} \leq s_{i,\boldsymbol{u}}\} \tag{23}$$

with vectors $\boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal$, $\boldsymbol{t}_{i,\boldsymbol{\nu}}^\intercal$ being $\boldsymbol{t}_{i,k,\boldsymbol{x}}^\intercal$, $\boldsymbol{t}_{i,k,\boldsymbol{u}}^\intercal$ respectively but with appropriately padded zeros. As $\boldsymbol{\xi}$ is normally distributed with a covariance of $\mathcal{S}_{\boldsymbol{x}}$, each constraint in (22) is a one dimensional Gaussian with covariance of

$$\sigma_i := \boldsymbol{t}_{i,\boldsymbol{x}}^\intercal\mathcal{S}_{\boldsymbol{x}}\boldsymbol{t}_{i,\boldsymbol{x}} . \tag{24}$$

Again, (22) denotes the constraints on the state, but in order to use it in the defined optimization problem, it has to be reformulated in terms of the input vector $\boldsymbol{\nu}$. Thus plugging into (17) leads to

$$\boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\boldsymbol{\xi} \leq s_{i,\boldsymbol{x}} \Leftrightarrow \boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\mathcal{B}\boldsymbol{\nu} \leq s_{i,\boldsymbol{x}} - \boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\mathcal{A}\boldsymbol{x}_0 . \tag{25}$$

We define

$$s_{i,\boldsymbol{x},\boldsymbol{\nu}} := s_{i,\boldsymbol{x}} - \boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\mathcal{A}\boldsymbol{x}_0, \quad \boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal := \boldsymbol{t}_{i,\boldsymbol{\xi}}^\intercal\mathcal{B} , \tag{26}$$

which yields transformed chance constraints of the states in $\boldsymbol{\nu}$. Note that the transformation in (25) does not change the distribution of the constraint and thus the standard deviation of each one-dimensional Gaussian stays untouched. With this transformation (22) can be stated as

$$P\left(\boldsymbol{\xi} \in \mathcal{F}_{\boldsymbol{\xi}}\right) = P\left(\bigcap_{i=1}^{M_{\boldsymbol{x}}} \{\boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal\boldsymbol{\nu} \leq s_{i,\boldsymbol{x},\boldsymbol{\nu}}\}\right) \geq \alpha . \tag{27}$$

Using Boole's inequality, this probability can be conservatively approximated as

$$b(\boldsymbol{\nu}) := \sum_{i=1}^{M_{\boldsymbol{x}}} P\big(\boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal\boldsymbol{\nu} \leq s_{i,\boldsymbol{x},\boldsymbol{\nu}}\big) \geq \alpha . \tag{28}$$

Following e.g. [4] we can reformulate (28) using the standard Gaussian cumulative distribution function Φ

$$b(\boldsymbol{\nu}) = \sum_{i=1}^{M_{\boldsymbol{x}}} \left(\Phi\left(\frac{s_{i,\boldsymbol{x},\boldsymbol{\nu}} - \boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal\boldsymbol{\nu}}{\sqrt{\sigma_i}}\right)\right) \geq \alpha . \tag{29}$$

This yields one highly nonlinear, scalar constraint. Most prior arts continue to linearize (29) by splitting the sum into $M_{\boldsymbol{x}}$ equations and introduce one coupling inequality constraint. We use the transformed form of equation (29) directly in the optimization program. To improve the solver's performance we determine the derivate of equation (29) as

$$\nabla b(\boldsymbol{\nu}) = \sum_{i=1}^{M_{\boldsymbol{x}}} \left(\varphi\left(\frac{s_{i,\boldsymbol{x},\boldsymbol{\nu}} - \boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal\boldsymbol{\nu}}{\sqrt{\sigma_i}}\right)\frac{-\boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal}{\sqrt{\sigma_i}}\right) , \tag{30}$$

where φ denotes the probability density function of the standard normal distribution.

As a further optimization, also the hessian of the constraint function can be calculated to

$$\begin{aligned} &\nabla^2 b(\boldsymbol{\nu}) = \nabla(\nabla b(\boldsymbol{\nu}))^\intercal = \\ &\sum_{i=1}^{M_{\boldsymbol{x}}} \left(\varphi\left(\frac{s_{i,\boldsymbol{x},\boldsymbol{\nu}} - \boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal\boldsymbol{\nu}}{\sqrt{\sigma_i}}\right)\frac{-\boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}\boldsymbol{t}_{i,\boldsymbol{\xi},\boldsymbol{\nu}}^\intercal}{\sigma_i}\right) , \end{aligned} \tag{31}$$

C. Optimization Algorithm

In order to implement the optimization problem (8) we need the following components from the previous sections:

- the cost function from (21). The according gradient and hessian are trivial to compute as this is a quadratic cost function.
- the nonlinear chance constraint $b(\boldsymbol{\nu})$ from (29) with its gradient (30) and hessian (31)
- the linear inequality constraints on the input from (23)
- currently we have no equality constraints

This is the standard form of nonlinear optimization problems and can be solved by algorithms like *interior-point* or *Sequential Quadratic Programming (SQP)* methods.

VI. EVALUATION

To evaluate the proposed algorithm, we implement the optimization problem for the single-track model in MATLAB[1].

A. Example 1: Comparison with LQR

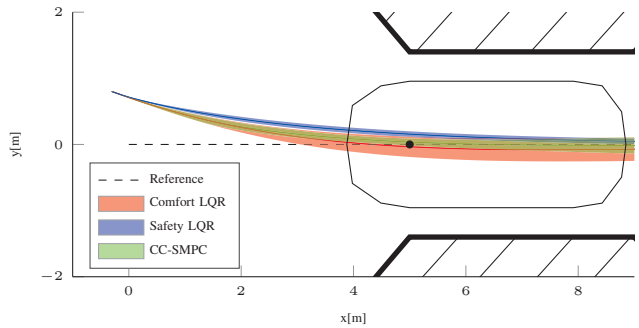


Fig. 3. Comparison of the different algorithms with indicated error covariance when entering a tunnel. The car shape shows the width of the tunnel.

We consider a tunnel environment with a straight reference trajectory at $y = 0$ as depicted in Fig. 3. As parameters, we chose a horizon of $N = 25$, a chance constraint bound of $\alpha = 0.95$, and weighting matrices of $\boldsymbol{Q} = \mathbf{1}, \boldsymbol{R} = \mathbf{1}$. As control constraints for this system, we define $|\sigma| \leq 0.3$, $|a| \leq 2$ and a process noise with constant covariance of $\boldsymbol{\Sigma_w} = \mathrm{diag}(0.5, 0.02)$. We consider a fixed starting point $\boldsymbol{x}_0 = (-0.3, 0.8, -0.3, 0)$. As sampling interval Δt we chose $\Delta t = 0.05$. For comparison, we take two different parametrizations for a standard nonlinear finite-horizon LQR and one deterministic constraint model predictive controller (C-MPC):

- *comfort LQR* has the same cost function as CC-SMPC, i.e., $\boldsymbol{Q}_{\mathrm{comfort}} = \boldsymbol{Q}$ and $\boldsymbol{R}_{\mathrm{comfort}} = \boldsymbol{R}$.
- *safety LQR* yields a better tracking behavior with $\boldsymbol{Q}_{\mathrm{safety}} = 5 \cdot \boldsymbol{Q}$
- *C-MPC* uses the same formulation as CC-SMPC but ignores the stochasticity of the system. Thus, the chance constraint reduces to linear inequality constraints

The simulation result for one run is depicted in Fig. 3 for the trajectories in the cartesian plane with the associated error covariance. The C-MPC variant has been omitted in the picture as it is very similar to the comfort LQR. As sketched in Fig. 1, the covariance of the safety LQR is very narrow in contrast to the comfort LQR. The CC-SMPC follows the behavior of the comfort LQR at the beginning, but when entering the tunnel, the covariance region contracts. As the controller stabilizes in the middle of the tunnel, the variance can safely grow again.

Repeating the previous simulation for 1000 times, Table I shows the average fail rate (i.e., percentage of runs violating the constraints), acceleration control effort $\sum |a|$, and steering control effort $\sum |\sigma|$. First, it can be seen that the comfort-LQR is violating the constraints repeatedly compared to the stronger parametrization of the safety LQR. This is also reflected in the difference in control effort. The CC-SMPC algorithm on the other hand has costs comparable to the comfort version, but a fail rate of 0%. This shows that the aspired tradeoff between safety and comfort is achieved. It is notable, that the C-MPC controller — although considering the constraints in a deterministic fashion — has a very high fail rate for this scenario. This shows, that considering the random errors is indeed necessary to achieve the desired controller behavior.

TABLE I
SUMMARY OF COMPARISON OF DIFFERENT USED ALGORITHMS

Method	Fail Rate	$\sum \lvert a \rvert$	$\sum \lvert \sigma \rvert$
CC-SMPC	0%	114.9	18.7
LQR(comfort)	79.3%	115.4	19.1
LQR(safety)	0.2%	143.3	24.6
C-MPC	72.6%	114.0	18.8

B. Example 2: A More Complex Scenario

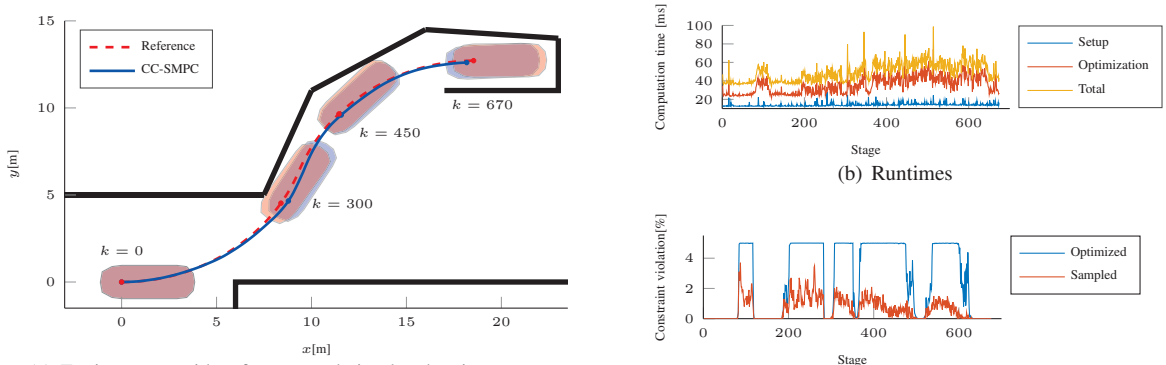
In this setup we consider the single track vehicle with the same parameters as in the previous section in a parking lot like environment. Fig. 4(a) shows the simulated vehicle compared to the reference trajectory. It is observable that the CC-SMPC algorithm takes a safer trajectory if the reference is very close to the obstacles (at $k = 300$) but tends to navigate back to the reference if more free space is available. The total runtime[2], depicted in Fig. 4(b), varies for each stage and tends to be longer in constraint environments. Nevertheless the mean runtime is below 50ms which suggests that the approach can be optimized to operate in real time. The CC-SMPC algorithm always shows a higher constraint violation than the Monte Carlo sampled constraints as shown in Fig. 4(c). This is expected as Boole's Inequality is a conservative approximation.

C. Evaluation of the Optimization Algorithm

Table II compares the solutions of different optimization algorithms for the vehicle model in the tunnel environment from section VI-A. The shown numbers are the mean values of the computations at each state. As a baseline we consider the results of the quadratic problem without chance

[1]with the Optimization Toolbox by The MathWorks

[2]measured on a Intel Xeon 2.8Ghz CPU



(a) Environment with reference and simulated trajectory

(b) Runtimes

(c) Chance constraint

Fig. 4. Analysis of one simulation run with the CC-SMPC controller in a complex environment. (a) Environment, reference, and a single simulation. At $k = 300$ the controller deviates from the reference to fulfill the chance constraint. (b) Runtimes. Time to generate block matrix formulation for optimization (Setup), optimization, and total time spent. (c) Active chance constraint calculated through optimization and computed with 5000 Monte Carlo simulations.

constraints, referred to as Quadprog. As well the SQP as the Active Set method only yield a merely slower runtime for the chance constraint problem. The Interior Point method approximately needs the double amount of iterations and cost function calls which results in a doubled runtime. Thus, the Active Set optimization is best suited for our problem.

TABLE II

STATISTICS FOR DIFFERENT ALGORITHMS FOR ONE COMPLETE RUN

Method	Runtime [ms]	J	# Iter	# Calls J
Interior Point	21.9	5.704	4.2	6.2
SQP	12.2	5.704	2.3	3.4
Active Set	11.7	5.701	2.1	3.4
Quadprog	9.3	5.779	4.8	4.8

VII. CONCLUSION AND FUTURE WORK

We have shown how stochastic model predictive control with chance constraints can be applied to control an autonomous vehicle. We achieved a tradeoff between comfort and tracking quality depending on the current requirements that arise in the presence of obstacles. This simplifies controller parametrization and the choice of safety margins in the trajectory planning, as no implicit knowledge of the tracking quality has to be encoded in the safety margins. We can detect when safe tracking is not possible anymore and trigger an emergency stop. We further introduced a method to obtain locally convex linear state constraints by combining sets of convex position constraints. The presented runtimes suggest, that the method is applicable to a real car with further optimization.

As future work, a feedback matrix can be added into the optimization process. This allows CC-SMPC to run with a lower frequency than an underlying feedback controller. Second, the state constraint construction can be improved, as approximating the car shape with many circles leads to stochastically dependent constraints and Boole's inequality is too conservative. Additionally real-world scenarios including other vehicles will be evaluated. The proposed approach is capable of such situations as constraints may be added or removed in each step. The vehicle model can be further extended to cover e.g. control delays.

REFERENCES

[1] J. Van Den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[2] M. Sznaier and M. J. Damborg, "Suboptimal Control of Linear Systems with State and Control Inequality Constraints," in *IEEE Conference on Decision and Control*, 1987, pp. 761–762.

[3] A. Nemirovski and A. Shapiro, "Convex Approximations of Chance Constrained Programs," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 969–996, 2006.

[4] M. Vitus and C. Tomlin, "On Feedback Design and Risk Allocation in Chance Constrained Control," in *IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 734–739.

[5] Z. Zhou and R. Cogill, "An Algorithm for State Constrained Stochastic Linear-Quadratic Control," in *American Control Conference*, 2011, pp. 1476–1481.

[6] J. Yan and R. R. Bitmead, "Incorporating state estimation into model predictive control and its application to network traffic control," *Automatica*, vol. 41, no. 4, pp. 595–604, Apr. 2005.

[7] M. Ono and B. Williams, "Iterative Risk Allocation: A New Approach to Robust Model Predictive Control with a Joint Chance Constraint," in *IEEE Conference on Decision and Control*, 2008, pp. 3427–3432.

[8] M. Shin and J. Primbs, "A Fast Algorithm for Stochastic Model Predictive Control with Probabilistic Constraints," in *American Control Conference*, 2010, pp. 5489–5494.

[9] Y. Ma, S. Vichik, and F. Borrelli, "Fast Stochastic MPC with Optimal Risk Allocation Applied to Building Control Systems," in *IEEE Conference on Decision and Control and European Control Conference*, 2012, pp. 7559–7564.

[10] M. P. Vitus and C. J. Tomlin, "A probabilistic approach to planning and control in autonomous urban driving," in *IEEE Conference on Decision and Control*, 2013, pp. 2459–2464.

[11] D. Lyons, J. Calliess, and U. D. Hanebeck, "Chance Constrained Model Predictive Control for Multi-Agent Systems with Coupling Constraints," in *American Control Conference*, 2012, pp. 1223–1230.

[12] J. Jerez, E. Kerrigan, and G. Constantinides, "A sparse and condensed QP formulation for predictive control of LTI systems," *Automatica*, vol. 48, no. 5, pp. 999–1002, 2012.

[13] G. M. Mancuso and E. C. Kerrigan, "Solving Constrained LQR Problems by Eliminating the Inputs from the QP," in *IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 507–512.

[14] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[15] S. Patil, J. van den Berg, and R. Alterovitz, "Estimating Probability of Collision for Safe Motion Planning under Gaussian Motion and Sensing Uncertainty," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 3238–3244.