Set-Based Predictive Control for Collision Detection and Evasion

Jeremy Crowley, Yegeta Zeleke, Berk Altın and Ricardo G. Sanfelice

Abstract— **We propose a set-based predictive control framework to predict inbound dynamic obstacles and optimize trajectories in the interest of safely guiding a vehicle towards a target. To account for uncertainties, the set-based controller generalizes conventional model predictive control and predicts the set that the state of a dynamical system might belong to. This generalization is used to formulate collision avoidance as a hard constraint in the set-based predictive control algorithm. As a proof-of-concept, the proposed framework is applied to a ground vehicle attempting to reach a target while anticipating and evading collisions with obstacles in the operating environment. Other applications of the controller and the associated optimal control problem are discussed.**

I. Introduction

Recent research interest in autonomous vehicles for use in civilian and military applications has been increasingly ubiquitous. Modern autonomous vehicles are involved in numerous applications ranging from rescue missions to package deliveries. For many autonomous systems, model predictive control (MPC) is preferred as a control strategy since collision avoidance can be embedded into the constraints of the corresponding optimal control problem, along with other hard constraints such as actuator limitations. A supervisory controller based on nonlinear MPC is used in [1] for a pursuit-evasion game involving two fixed-wing autonomous aircrafts. An MPC scheme that integrates tracking and stabilization with a higher priority on collision avoidance was discussed in [2]. In [3], nonstationary obstacles on the path of an autonomous are avoided using MPC, where collision avoidance is embedded as a soft constraint with high cost. The control strategy in [3] is implemented in a hierarchical fashion, where a high-level central MPC schemes supervises low-level local MPC schemes. A numerical method for aerial pursuit-evasion games is solved via MPC in [4]. Soft constraints are also used in the MPC formulation of [5] to ensure that a network of three autonomous helicopters operate without collisions in a shared environment. A target seeking scenario in an a priori unknown environment with obstacles is realized via MPC in [6]. A predictive controller is used in [7] for lane keeping and collision avoidance of an autonomous ground vehicle. Recently, MPC was extended to hybrid dynamical systems in [8], [9].

This paper proposes a novel collision detection and avoidance strategy based on set-based predictive control. Building from the so-called set dynamical systems framework in [10], [11], [12], where the state of a system is identified by a set rather than a point, the set-based predictive controller generalizes conventional MPC along the lines of the tube-based MPC approach [13]. Although collision detection and evasion have been presented in [3],[7],[14] modeling uncertainties that arise due to disturbances and measurement noise has not been studied to the best of our knowledge. In order to account for these unseen dynamical behaviors, we propose a method by which a controller will predict sequences of sets that the state of a discrete-time dynamical system might belong to, and imposes constraints on the predicted sets for robustness purposes.

Unlike other set-based approaches in the conventional MPC literature (for example, [15], [16], [17]—see also the survey [18] for the tube-based MPC technique), where the optimal control problem is to minimize solutions of a "classical" dynamical system (in the sense that the state is identified by a point), the optimal control problem associated with our strategy is to minimize solutions of a set dynamical system. In other words, *the minimization associated with our strategy occurs over sequences of sets, rather than points*. In addition to collision detection and avoidance, the proposed strategy can find use in various applications such as uncertainty propagation, reachable set computation, and safety analysis for conventional discrete-time systems. A detailed comparison of the proposed strategy with existing collision and avoidance schemes is outside the scope of this exploratory work, and will be tackled in the future.

The rest of the paper is organized as follows. Section II outlines the collision detection and evasion problem and details our proposed solution based on set-based predictive control. As a proof-of-concept, in Section III, simulation results for the proposed control algorithm are validated via simulations, followed by experimental results for an autonomous ground vehicle seeking a target in the presence of obstacles. Concluding remarks are given in Section IV.

Notation: We use $\mathbb{R}$ to represent real numbers and $\mathbb{R}_{\geq 0}$ its nonnegative subset. The set of nonnegative integers is denoted $\mathbb{N}$. The 2-norm is denoted $|.|$. The distance of a vector $x \in \mathbb{R}^n$ to a nonempty set $\mathcal{A} \subset \mathbb{R}^n$ is $|x|_{\mathcal{A}} := \inf_{a \in \mathcal{A}} |x - a|$. Let $X, Y \subset \mathbb{R}^n$. The notation $X \subset Y$ indicates that X is a subset of Y, not necessarily proper. The Minkowski sum of X and Y is denoted $X + Y$. The notation $\mathcal{P}(X)$ denotes the set of nonempty subsets of X. $\mathbb{P}_p^n$ denotes the set of compact convex polytopes in $\mathbb{R}^n$ with p vertices. For any $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, $(x, y) := \begin{bmatrix} x^\top & y^\top \end{bmatrix}^\top$. Given

*Research partially supported by NSF Grants no. ECS-1710621 and CNS-1544396, by AFOSR Grants no. FA9550-16-1-0015, FA9550-19-1-0053, and Grant no. FA9550-19-1-0169, and by CITRIS and the Banatao Institute at the University of California.

Jeremy Crowley is with the Department of Aeronautics and Astronautics, Stanford University. Yegeta Zeleke, Berk Altın, and Ricardo G. Sanfelice are with the Department of Electrical and Computer Engineering, University of California, Santa Cruz, CA 95064, USA; crowleyj@stanford.edu, yzeleke@ucsc.edu, berkaltin@ucsc.edu, ricardo@ucsc.edu.

a (set-valued) mapping F and a set X in its domain, we use $F(X)$ to denote the image of X under F.

II. Set-Based Predictive Control

Consider the discrete-time dynamical system

$$x^+ = f(x, u, w) \tag{1}$$

with state $x \in \mathbb{R}^n$, input $u \in \mathbb{R}^m$, and disturbance $w \in \mathbb{R}^l$, where $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l \to \mathbb{R}^n$. In (1), x^+ denotes the value of the state x after a discrete transition under the input u and disturbance w. In the presence of measurement noise, at any discrete time $j \in \mathbb{N}$, the value of the state x_j is not known with certainty, but can be estimated to belong to a set X_j. For example, if the additive noise $v_j \in \mathbb{R}^n$ satisfies $|v_j| \leq \delta$, then given the state measurement $\hat{x}_j = x_j + v_j$, we have $x_j \in X_j$, where $X_j = \{x \in \mathbb{R}^n : |x - \hat{x}_j| \leq \delta\}$. More generally, suppose that $w_j \in W$ and $(-v_j) \in V$ for some sets W and V, at every $j \in \mathbb{N}$. By (1),

$$\begin{aligned} x_{j+1} = f(\hat{x}_j - v_j, u_j, w_j) &\in f(\hat{x}_j + V, u_j, W) \\ &= f(X_j, u_j, W) =: F(X_j, u_j) \end{aligned} \tag{2}$$

where $F : \mathbb{R}^n \times \mathbb{R}^m \rightrightarrows \mathbb{R}^n$ is a *set-valued* mapping.

Set-based approaches along the lines of the *difference inclusion* $x^+ \in F(x, u)$ are common in the robust MPC literature [13], where the optimal control problem is designed to constrain trajectories (or solutions) $x_0, x_1, \ldots, x_N$ to sequences of sets $X_0, X_1, \ldots, X_N$, or "tubes", so that $x_j \in X_j$ for all $j \in \{0, 1, \ldots, N\}$. As a key difference with those approaches, in this paper, we deal *directly* with trajectories given by sequences of sets $X_0, X_1, \ldots,$ that are not necessarily tubes [10], [11], [12].

A. The Predictive Control Problem

We propose a *set-based predictive control* scheme for discrete-time systems with solutions given by sequences of sets. Given the difference inclusion $x^+ \in F(x, u)$ modeling the system to control, which can arise from factoring in uncertainties in (1), instead of predicting sequences of points $x_0, x_1, \ldots, x_N \in \mathbb{R}^n$ and selecting the sequence with the least cost, we propose to predict sequences of sets $X_0, X_1, \ldots, X_N \subset \mathbb{R}^n$ and select the sequence with the least cost for the *set dynamical system*

$$X^+ = F(X, U). \tag{3}$$

As in standard MPC, the ingredients of this controller include a prediction horizon $N \geq 1$, a control horizon M with $1 \leq M \leq N$, a stage cost $L : \mathcal{P}(\mathbb{R}^n \times \mathbb{R}^m) \to \mathbb{R}_{\geq 0}$, a terminal cost $V : \mathcal{P}(\mathbb{R}^n) \to \mathbb{R}_{\geq 0}$, a mixed-constraint set $\mathcal{C} \subset \mathcal{P}(\mathbb{R}^n \times \mathbb{R}^m)$, and a terminal constraint set $X_V \subset \mathcal{P}(\mathbb{R}^n)$. Note that L (respectively, V) assigns a cost to every nonempty subset of $\mathbb{R}^n \times \mathbb{R}^m$ (respectively, $\mathbb{R}^n$) as opposed to the formulation in [13]. Similarly, the constraint set $\mathcal{C}$ (respectively, X_V) is a collection of nonempty subsets of $\mathbb{R}^n \times \mathbb{R}^m$ (respectively, $\mathbb{R}^n$).

The proposed controller operates by solving the following problem to predict input sets $U_0^\star, U_1^\star, \ldots, U_{N-1}^\star$ such that the solution $X_0^\star, X_1^\star, \ldots, X_N^\star$ of the set dynamical system (3) minimizes the cost in (4), subject to constraints.

Problem 2.1: Given the prediction horizon N, stage cost L, terminal cost V, constraint sets $\mathcal{C}$ and X_V, and initial condition set X_0, find sequences $X^\star := \{X_j^\star\}_{j=0}^N$ and $U^\star := \{U_j^\star\}_{j=0}^{N-1}$ minimizing the cost

$$\mathcal{J}(X^\star, U^\star) := \left(\sum_{j=0}^{N-1} L(X_j^\star, U_j^\star) \right) + V(X_N^\star), \tag{4}$$

subject to the constraints $X_0^\star = X_0$ and

$$\begin{cases} X_{j+1}^\star = F(X_j^\star, U_j^\star) & \forall j \in \{0, 1, \ldots, N-1\} \\ (X_j^\star, U_j^\star) \in \mathcal{C} & \forall j \in \{0, 1, \ldots, N-1\} \\ X_N^\star \in X_V. & \end{cases}$$

The optimal input sequence is applied until time step M, at which point the process is repeated for the new initial condition. That is, the sequence $U_0^\star, U_1^\star, \ldots, U_{M-1}^\star$ is applied to obtain the solution $X_0^\star, X_1^\star, \ldots, X_M^\star$, and then Problem 2.1 is re-solved by setting the initial condition set $X_0 = X_M^\star$. This process is summarized in Algorithm 1, where i is the time step of the closed loop and $\widehat{X} := \{\widehat{X}_i\}_{i=0}^\infty$ is the resulting closed-loop state trajectory.

Algorithm 1: Set-based predictive control.

```
Obtain initial system state $\widehat{X}_0$, set $i = 0$ and $X_0 = \widehat{X}_0$
while True do
    Solve Problem 2.1
    $j = 0$
    for $j \leq M - 1$ do
        $\widehat{X}_{i+1} = X_{j+1}^\star = F(X_j^\star, U_j^\star)$
        $i = i + 1$, $j = j + 1$
    end
    Set $X_0 = X_M^\star$
end
```

Remark 2.2: In tube-based MPC, the minimization is performed over nominal point-based trajectories, and the constraints are defined using subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$. To ensure that constraints are satisfied under uncertainties, an auxiliary controller is employed to steer trajectories towards that of the nominal system. In contrast, the minimization in Problem 2.1 is performed directly over *set-based trajectories*, and the constraints are defined using *power sets* of $\mathbb{R}^n$ and $\mathbb{R}^m$. As such, the proposed framework differs considerably from tube-based MPC and similar set-based methodologies.

B. Applications

The formulation in (2) shows that the set dynamical system (3) can be used to compute uncertainty propagation in the discrete-time system (1). In addition, as noted in [10], the set dynamical system in (3) arises in a wide range of problems. Below, we discuss three example applications.

1) Reachable Set Computation: The system in (3) can be used to compute the reachable set of (1) from an initial set X_0 up to timestep N if $F(X,U) = f(X,U,0)$. To compute reachable sets using Problem 2.1, the stage cost and terminal cost can be selected such that

$$L(X,U) = V(X) = \frac{a}{1 + b \int_{\mathbb{R}^n} \imath_X(x)\, dx} \tag{5}$$

for some $a, b > 0$, where $\imath_X$ is the indicator function of X. Roughly speaking, $L(X,U) = V(X) \geq a > 0$ is inversely proportional to the volume of X, so minimizing the cost in (4) leads to larger sets. The constraint sets can be selected as $\mathcal{C} = \mathcal{P}(\mathcal{X} \times \mathcal{U})$ and $X_N^\star = \mathcal{P}(\mathcal{X})$ for some $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{U} \subset \mathbb{R}^m$, representing the state and the input constraints, respectively. To make this problem tractable, it would be necessary that $\mathcal{U} \neq \mathbb{R}^m$. When the initial set X_0 has no volume, the integral in (5) can be replaced with other mappings measuring the size of X.

2) Safety Analysis: When the mapping F in (3) is derived via (2), a robust safety analysis for (1) can be conducted by checking whether given a closed *safe set* K and $K' \subset K$, there exists a sequence of inputs $U_0, U_1, \ldots$ such that the corresponding solution $X_0, X_1, \ldots$ of (3) satisfies $X_j \subset K$ for all $j \in \mathbb{N}$ when $X_0 = K'$. If the state and the input constraints can again be represented by some sets $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{U} \subset \mathbb{R}^m$, safety analysis can be conducted by repeatedly solving Problem 2.1 as in Algorithm 1, with $\mathcal{C} = \mathcal{P}(\mathcal{X} \times \mathcal{U})$, $\quad X_V = \mathcal{P}(\mathbb{R}^n)$ and $L(X,U) = V(X) = \sup_{x \in X} |x|_K$. If $K' = K$ and $\mathcal{U} = \{0\}$, this problem reduces to the forward invariance problem for the inclusion $x^+ \in F(x, 0)$.

3) Collision Avoidance: An autonomous vehicle with the dynamics in (1) that is trying to reach a target set X^* while avoding obstacles can be controlled by the predictive control strategy of Algorithm 1. To ensure convergence to the target, the cost functions L and V can be designed such that $L(X,U) = V(X) = 0$ if $X \subset X^*$. Obstacle avoidance can be ensured by choosing $\mathcal{C}$ to be a subset of $\mathcal{P}(\mathcal{X} \times \mathcal{U})$ for a state constraint set $\mathcal{X}$ and input constraint set $\mathcal{U}$. Next, we detail the application of Algorithm 1 to collision avoidance.

C. Collision Detection and Evasion for Autonomous Vehicles

We consider a scenario in which a vehicle with the dynamics in (1) has to avoid a stationary obstacle represented by a set Y and reach a target set X^*. We assume that the dynamics in (1) are *overapproximated* by the set-based dynamics in (3), where the set state X and input U are compact convex polytopes, and F maps compact convex polytopes to compact convex polytopes. The choice of polytopic sets are meant to facilitate computations, especially in the case where the mapping F is affine and single valued: since affine transformations map vertices to vertices, the set $F(X,U)$ is precisely the convex hull of $F(X^V, U^V)$, where X^V and U^V are the set of vertices of X and U, respectively. Such a scenario can arise when the mapping f is affine and the set W in (2) is also taken to be a compact convex polytope. In the case where the mapping f in (1) is not affine, F can be nevertheless be chosen such that $F(X,U) \supset f(X,U)$ is a compact convex polytope.

Remark 2.3: A similar set-based approach to motion planning is proposed in [19], where the focus is on the reachable set computation of nonlinear systems under disturbances, with initial conditions belonging to zonotopic sets.

1) Selecting the Cost Functions and the Terminal Constraint: To ensure that the vehicle can reach the target X^* in the absence of collisions, the cost functions L and V can be designed to have distance-like properties. The terminal constraint set X_V should be selected so that $X_V \cap X^*$ is nonempty, as the converse would prevent the vehicle from reaching X^* when $M = N$—this is similar to conventional MPC, where X^* would be the origin, and X_V would be a neighborhood of the origin. For simplicity, we let $L(X,U) = \sum_{i=1}^{p} |x_i|_{X^*}$ and $V(X) = c \sum_{i=1}^{p} |x_i|_{X^*}$ for some $c > 0$, where $x_0, x_1, \ldots, x_p$ are the vertices of X.

2) Encoding Safety Constraints into $\mathcal{C}$: The mixed constraint set $\mathcal{C}$ can be utilized to prevent collisions between the vehicle and the obstacle. The state could be subject to $X \subset \mathcal{X}$, where $\mathcal{X}$ is the state constraint set, and the input could be subject to $U \subset \mathcal{U}$, where $\mathcal{U}$ is the input constraint set. The set $\mathcal{X}$ typically models the geometry of the operating environment of the vehicle-obstacle system, while the input constraint set $\mathcal{U}$ could arise due to actuator limitations like saturation. Denote by H an output mapping so that collisions correspond to $H(X)$ intersecting Y. Then, the mixed constraint set can be selected as

$$\begin{aligned} \mathcal{C} = \{X \times U \subset \mathcal{X} \times \mathcal{U} : \sigma(H(X), Y) \geq \varsigma\} \\ \cap \left(\mathbb{P}^n_{p_x} \times \mathbb{P}^m_{p_u}\right) \cap \widehat{\mathcal{C}}, \end{aligned} \tag{6}$$

for some $\varsigma > 0$ and $p_x, p_u \in \mathbb{N}$. Above, the mapping $\sigma(H(X), Y) := \inf_{(x,y) \in X \times Y} |H(x) - y|$ is a measure of the shortest path between the sets $H(X)$ and Y, and the generic set $\widehat{\mathcal{C}} \subset \mathcal{P}(\mathbb{R}^n \times \mathbb{R}^m)$ can be used to encode other constraints. In most cases, H would be a linear function extracting the position coordinates.

Other safety constraints can also be encoded in $\mathcal{C}$. In particular, the trajectory of the vehicle in the intersample period can be approximated by $\operatorname{con}(X \cup F(X,U))$, the convex hull of X and $F(X,U)$, and the term $H(X)$ in (6) can be replaced with $H\left(\operatorname{con}\left(X \cup F(X,U)\right)\right)$.

D. Challenges of Problem 2.1 and Algorithm 1

From a computational standpoint, there are several challenges associated with Problem 2.1 and Algorithm 1.

- **(C1)** The presence of state and input sets, along with the constraints, prevents the use of standard techniques to solve Problem 2.1, making its solution difficult in general.
- **(C2)** The computational burden associated with a numerical solution of Problem 2.1 can be high enough to prevent online implementation.
- **(C3)** Perturbations on the set dynamical system (e.g. delays, unmodeled dynamics) can adversely affect the performance of Algorithm 1 and lead to constraint violations.

In essence, the above challenges are similar to those arising in the context of conventional optimal control and

MPC. Although Problem 2.1 is somewhat conceptual, it can be solved in certain cases. For example, in the case of the scenario described in Section II-C, Problem 2.1 can be solved using nonlinear programming methods. In general, Problem 2.1 can be solved suboptimally with acceptable computational burden. The amount of computational burden deemed acceptable would naturally depend on the application. For online implementation of Algorithm 1, if (3) is derived from continuous-time dynamics, the time to compute should to be reasonably smaller than the sampling period.

These challenges are discussed further in Section III. In particular, as a proof-of-concept, we show how Algorithm 1 can be used in an applied setting, demonstrating the value of the proposed framework. The simulations in Section III show how **(C1)**-**(C2)** can be addressed by careful selection of the constraints and costs. To address **(C2)**-**(C3)** and show that Algorithm 1 can tolerate perturbations while running fast enough for online implementation, experiments are conducted in Section III-D, with the results in Section III-C forming a baseline for comparison.

III. Application to Collision Detection and Evasion for Vehicles

We now show how the set-based predictive control scheme outlined in Section II can be effectively applied to a ground vehicle in the presence of static and dynamic obstacles.

A. *Vehicle and Obstacle Dynamics*

We model our ground vehicle using the Dubins model

$$\begin{aligned}\dot{q}_1 &= v\cos(\theta),\\ \dot{q}_2 &= v\sin(\theta),\\ \dot{\theta} &= (v/L)\tan(\phi) =: \omega,\end{aligned} \tag{7}$$

where q_1 and q_2 are the Cartesian coordinates, θ is the heading angle, v is the speed, L is the length of the vehicle, and ϕ is the steering angle. The exact discretization of (7) with the step size T yields the model

$$x^+ = f(x,u,w) = \begin{bmatrix} q_1 + u_1\frac{2\cos(\theta+u_2)\sin(u_2)}{\omega} \\ q_2 + u_1\frac{2\sin(\theta+u_2)\sin(u_2)}{\omega} \\ \theta + 2u_2 \end{bmatrix},$$

where $x := (q_1, q_2, \theta)$ and $u := (u_1, u_2) = (v, T\omega/2)$. Although the system in (3) allows for the *explicit* inclusion of disturbances, for simplicity, the set dynamical system which we will use for prediction purposes will be given by the mapping $F(X,U) = f(X,U,0)$, and the effects of disturbances on (7) will be embedded into the set state X.

B. *Constraint Selection*

For simplicity, we do not impose any state constraints on the vehicle, i.e., $\mathcal{X} = \mathbb{R}^3$. Similarly, $X_V = \mathbb{R}^3$. The target set is taken as $X^* = \{-0.50\} \times \{0.06\} \times \mathbb{R}$, and the input constraint set $\mathcal{U}$ in (6) is taken as

$$\mathcal{U} = \{(u_1,u_2) : 0 \le u_1 \le \beta, -(T\alpha)/2 \le u_2 \le (T\alpha)/2\}.$$

Here, $\beta \ge 0$ and $\alpha \ge 0$ limit the magnitude of the vehicle's speed in m/s and steering angle in radians, respectively.

For Problem 2.1 to be computationally viable for real-time implementation, a concern raised in **(C2)**, inputs for set dynamics, as in (3), are taken to be singletons. That is, for each $j \in \{0, 1, \dots, N-1\}$, the decision variable $U_j^\star$ of Problem 2.1 is chosen from subsets of $\mathbb{R}^p$ consisting of a single element, i.e., we only consider the case where $p_u = 1$ in (6). Furthermore, we assume a scenario where the state components (q_1, q_2) are subject to uncertainty, while θ can be measured exactly. More specifically, we assume that

$$x \in [z_1, z_2] \times [z_3, z_4] \times \{z_5\} \subset \mathbb{R} \times \mathbb{R} \times \mathbb{R} \tag{8}$$

for some z_1, z_2, z_3, z_4, z_5. The rectangular set in (8) corresponds to the translation of the set V in (2). To formalize this in (6), we take $p_x = 4$ and $\widehat{\mathcal{C}} = \mathcal{P}(\mathbb{R}^2) \times \mathbb{R} \times \mathcal{P}(\mathbb{R})$, as $n = 3$ and $m = 1$; i.e. any $(X, U) \in \mathcal{C}$ should be such that the projection of X onto the θ space should be a singleton. The output mapping H is chosen as a linear mapping such that $H([z_1, z_2] \times [z_3, z_4] \times \{z_5\}) = [z_1, z_2] \times [z_3, z_4]$.

The selection of the constraints with this structure ensures that the set $F(X, U)$ is of the same size as X (in terms of its area in the (q_1, q_2) space), and prevents a scenario where the resulting system is "uncontrollable". The choice of rectangular sets further reduces the computational burden of computing the propagation of the polytope in (8), which can be represented by a matrix in $\mathbb{R}^{3\times 4}$ (or equivalently a vector in $\mathbb{R}^{12}$): since the dynamics of q_1 and q_2 are decoupled, letting $z := (z_1, z_2, z_3, z_4, z_5)$, where z_i's come from (8), for prediction purposes, we rely on the discrete-time model

$$z^+ = z + \begin{bmatrix} Tu_1\frac{2\cos(z_5+u_2)\sin(u_2)}{u_2} \\ Tu_1\frac{2\cos(z_5+u_2)\sin(u_2)}{u_2} \\ Tu_1\frac{2\sin(z_5+u_2)\sin(u_2)}{u_2} \\ Tu_1\frac{2\sin(z_5+u_2)\sin(u_2)}{u_2} \\ 2u_2 \end{bmatrix}. \tag{9}$$

Section II-C outlines the safety constraint, which requires the nontrivial computation of $\sigma(H(X), Y)$ in (6). This can be accomplished by minimizing $|P_1\lambda_1 - P_2\lambda_2|^2$ via a quadratic program, where, for each $i \in \{1, 2\}$, $P_i \in \mathbb{R}^{n\times p_i}$ represents a polytope with each column corresponding to a vertex, and $\lambda_i \in \mathbb{R}^{p_i}$ is subject to constraints such that $P_i\lambda_i$ represents convex combinations.

C. *Nominal Results*

The simulations[1], conducted on a MacBook Pro computer with 2.5 GHz Intel core i5 processor and 8GB RAM, were run with the step size $T = 0.05\,\mathrm{s}$, $N = 8$, $M = 2$, $\varsigma = 0.05\,\mathrm{m}$. The parameters for the input constraints were chosen as $\beta = 0.78$ and $\alpha = \pi/6$. The simulations assume a 2.5 m×3.0 m operating environment, with a rectangular obstacle with dimensions 0.05 m×0.05 m centered at $(0.025, 0.083)$. The size of the set used for the vehicle is the same size as the vehicle used in the upcoming experiments, namely 0.451 m×0.331 m. As opposed to solving the quadratic program discussed in Section (III-B),

[1] github.com/HybridSystemsLab/SetBasedPredictionCollisionAndEvasion

to address **(C2)**, we underapproximate the minimum distance between the two rectangular polytopes with the formula $\max\{|c_1 - c_2| - (r_1 + r_2), 0\}$, where c_i is the center coordinate of polytope i and r_i is the distance from c_i to a vertex of polytope i, corresponding to the distance between circular overapproximations centered at x_i with radius r_i. For 180 optimization tasks, the resulting runtime of the modified algorithm has an average of 0.1008 seconds with standard deviation of 0.0558 seconds. The minimum and maximum runtimes are 0.0361 and 0.3607 seconds, respectively.

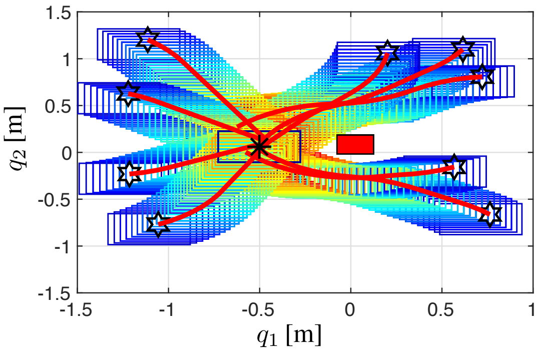

Fig. 1: Simulation results from nine initial conditions.

Figure 1 shows that the algorithm is successful in avoiding the obstacle and converges to a region near the target $(*)$. For each trajectory, the set of states in the same prediction horizon use the same color. The prediction horizons start colored in blue and gradually transitions to red. It can be seen in Figure 2 that the safety constraints are satisfied at all times. The dashed line at the top subfigure depicts ς and the trajectories converge to a small neighborhood. The dashed line at the bottom subfigure depicts $D_{\min} = 2(\sqrt{L^2 + W^2}) = 1.1184$, where L is the length and W is the width of the rectangle polytope.

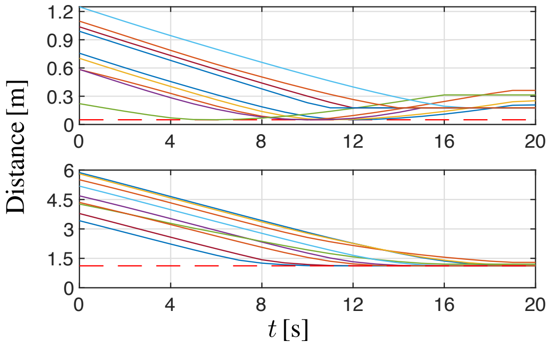

Fig. 2: Top: Minimum distance from polytope to obtacle. Bottom: Sum of distances between vertices and obstacle.

To fully use the predictive qualities of this algorithm, simulations were run with a dynamic obstacle from four different initial conditions. The obstacle state was subject to the same polytopic constraints as the vehicle, evolving under the effect of a known constant input with $0 \leq u_1 \leq \beta$. As such, the model in (9) was used to predict the motion of the obstacle. The parameters used for these simulations were the same as the static obstacle case except for the target, which was taken as $X^* = \{0\} \times \{0\} \times \mathbb{R}$. Figure 3 shows that the vehicle can successfully converge to a neighborhood of the target while avoiding collisions.

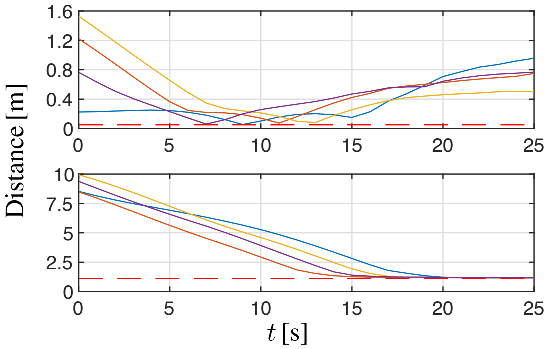

Fig. 3: Dynamic obstacle simulations data, as in Figure 2.

D. Experimental Results

To demonstrate the applicability of Algorithm 1 in a real-world setting, experiments reproducing the simulation scenarios with a stationary obstacle were carried out. In particular, we show that despite the challenges outlined in **(C1)**-**(C3)**, Algorithm 1 guides the vehicle toward the target while avoiding collision with the obstacle. In addition, we discuss the effects of uncertainties arising from computational delays, the simplicity of the employed model in (7), and quantify the time-to-compute the control input for Problem 2.1.

1) Experimental setup: The considered experimental system is a multi-node robotic system comprising a Windows computer with Intel i5 dual core (3.20 and 3.19 GHz) processor and 8 GB RAM, a motion capture system, a radio-frequency (RF) communication system, and a radio-controlled (RC) vehicle (communication up to 2.4 GHz). Data from eight Flex-13 cameras are transferred to MATLAB with the OptiTrack motion-capture software. A near real-time communication between MATLAB and the motion capture hardware is managed using the OptiTrack NatNet library, yielding a minimal communication latency of 8.3 ms.

2) Analysis of Experimental Results: A series of initial conditions are considered to confirm the simulation results in Section III-C (Figure 4). As in Figure 1, each initial condition is marked by a 6 edged star, the obstacle is marked by a red rectangle, and the target is marked by black star. Continuous red lines show the phase portraits of the vehicle trajectories. As in Figure 1, the set of states in the same prediction horizon are colored similarly for each run. From the latter color coding, one can observe the existence of a gap between consecutive set states at certain times due to computational delays, a phenomenon not observed in the simulations. Nevertheless, despite delays and model uncertainties, the vehicle successfully avoids the obstacle and reaches a neighborhood of the target.

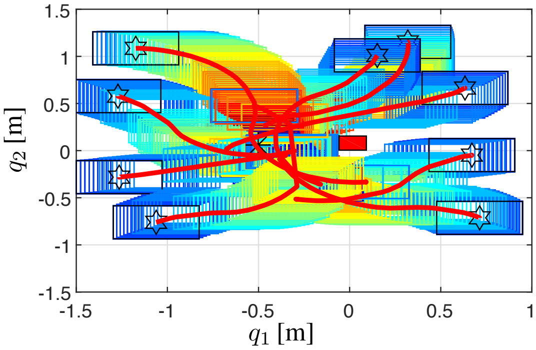

Fig. 4: Experimental results corresponding to Figure 1.

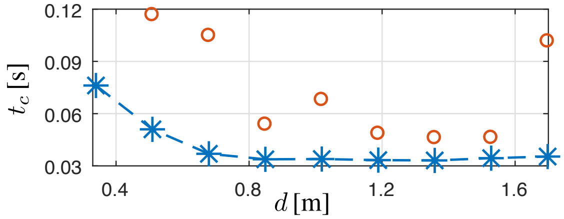

Fig. 5: Mean (circle) and median (star) of computational time.

The dependency of the computational delay on the distance between the vehicle and obstacle is illustrated in Figure 5. That is, the predicted trajectories illustrated in Figure 4 are categorized according to the initial distance from the vehicle to the obstacle. For example, all predicted trajectories where the vehicle-distance is less than 0.4 m belong to the same category. For each category, we gather all the computational latency and report the mean and median on Figure 5. As it can be seen from there, the computation time increases as the vehicle gets closer to the obstacle. The median computation time is approximated by the polynomial (the blue dashed line) $t_c(d) = -0.07(d)^3 + 0.26(d)^2 - 0.32(d) + 0.15$, where d denotes the distance from the vehicle to the obstacle.

Despite some performance degradation, the results illustrate the effectiveness of the proposed algorithm on an experimental platform in the presence of the challenges listed in **(C1)**-**(C3)**. This motivates the use of set-based predictive control for motion planning and control in mobile robotics, as well as the development of formal stability guarantees for Algorithm 1 and numerical tools to solve Problem 2.1.

IV. Conclusion

This paper presented a set-based predictive control approach to collision avoidance and path planning, derived from the extension of optimal control problems in MPC to set dynamical systems, to account for uncertainties. Simulation and experimental results show that the optimal control problem can be solved suboptimally to reduce computational burden, and in doing so, guide an autonomous vehicle safely towards a target. Future work will focus on the development of numerical tools and formal feasibility/stability guarantees.

References

[1] J. M. Eklund, J. Sprinkle, and S. S. Sastry, "Switched and symmetric pursuit/evasion games using online model predictive control with application to autonomous aircraft," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 3, pp. 604–620, 2012.

[2] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1204–1216, 2017.

[3] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in *ASME 2010 dynamic systems and control conference*, vol. 1. American Society of Mechanical Engineers, 2010, pp. 265–272.

[4] S. Kang, H. J. Kim, and M.-J. Tahk, "Aerial pursuit-evasion game using nonlinear model predictive guidance," in *AIAA Guidance, Navigation, and Control Conference*, 2010, pp. 1–12.

[5] H. J. Kim, D. H. Shim, and S. Sastry, "Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 5. IEEE, 2002, pp. 3576–3581.

[6] J. Liu, P. Jayakumar, J. Stein, and T. Ersal, "Combined speed and steering control in high speed autonomous ground vehicles for obstacle avoidance using model predictive control," *IEEE Transactions on Vehicular Technology*, 2017.

[7] A. Carvalho, Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, "Predictive control of an autonomous ground vehicle using an iterative linearization approach," in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 2013, pp. 2335–2340.

[8] B. Altın, P. Ojaghi, and R. G. Sanfelice, "A model predictive control framework for hybrid dynamical system," in *6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018*, 8 2018, pp. 128 – 133.

[9] B. Altın and R. G. Sanfelice, "Asymptotically Stabilizing Model Predictive Control for Hybrid Dynamical Systems," in *American Control Conference (ACC), 2019*, 7 2019, p. to appear.

[10] R. G. Sanfelice, "Asymptotic properties of solutions to set dynamical systems," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 2287–2292.

[11] N. Risso and R. G. Sanfelice, "Detectability and invariance properties for set dynamical systems," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 1030 – 1035, 2016, 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

[12] ——, "Sufficient conditions for asymptotic stability and feedback control of set dynamical systems," in *2017 American Control Conference (ACC)*, May 2017, pp. 1923–1928.

[13] S. V. Raković and D. Q. Mayne, "Robust model predictive control for obstacle avoidance: discrete time case," in *Assessment and Future Directions of Nonlinear Model Predictive Control*. Springer, 2007, pp. 617–627.

[14] Z. Chao, L. Ming, Z. Shaolei, and Z. Wenguang, "Collision-free UAV formation flight control based on nonlinear MPC," in *Electronics, Communications and Control (ICECC), 2011 International Conference on*. IEEE, 2011, pp. 1951–1956.

[15] A. Gautam, Y. C. Soh, and Y. Chu, "Set-based model predictive consensus under bounded additive disturbances," in *2013 American Control Conference*, June 2013, pp. 6157–6162.

[16] G. A. Gonalves and M. Guay, "Robust discrete-time set-based adaptive predictive control for nonlinear systems," *Journal of Process Control*, vol. 39, pp. 111 – 122, 2016.

[17] H. H. Nguyen, A. Savchenko, S. Yu, and R. Findeisen, "Improved robust predictive control for Lure systems using set-based learning," *IFAC-PapersOnLine*, p. to appear, 2018, 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018.

[18] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967 – 2986, 2014.

[19] B. Schrmann and M. Althoff, "Guaranteeing constraints of disturbed nonlinear systems using set-based optimal control in generator space," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11 515 – 11 522, 2017, 20th IFAC World Congress.