Obstacle Avoidance in Dynamic Environments via Tunnel-following MPC with Adaptive Guiding Vector Fields

Albin Dahlin, Yiannis Karayiannidis

Abstract— This paper proposes a motion control scheme for robots operating in a dynamic environment with concave obstacles. A Model Predictive Controller (MPC) is constructed to drive the robot towards a goal position while ensuring collision avoidance without direct use of obstacle information in the optimization problem. This is achieved by guaranteeing tracking performance of an appropriately designed receding horizon path. The path is computed using a guiding vector field defined in a subspace of the free workspace where each point in the subspace satisfies a criteria for minimum distance to all obstacles. The effectiveness of the control scheme is illustrated by means of simulation.

I. INTRODUCTION

Navigating autonomous agents to a goal position in a dynamic environment with both moving obstacles, such as humans and other autonomous systems, and static obstacles is a common problem in robotics. The resulting trajectory must be collision-free and obey possible robot constraints. Traditionally, motion planning problems have been solved considering static maps, but dynamic environments with moving obstacles require online adjustments of the planned robot path to avoid crashes. A common method to tackle such problems is to construct closed form control laws that result in closed loop reactive dynamical systems (DS) that possess desirable stability and convergence properties. Specifically, artificial potential fields [1], repelling the robot from the obstacles, have become popular [2], [3]. However, a drawback of the additive potential field methods is that they may yield local minimum other than the goal point, i.e. the robot could get stuck at a position away from the goal. To address this issue, navigation functions [4]–[6] and harmonic potential fields have emerged [7]–[11]. A repeated assumption in the aforementioned methods enabling the proof of (almost) global convergence is the premise of disjoint obstacles. However, intersecting obstacles are frequently occurring, e.g. when modelling complex obstacles as a combination of several simpler shapes, or when the obstacle regions are padded to take robot radius or safety margins into account. In [12] the authors presented a environment modulation algorithm, here referred to as ModEnv⋆, to obtain

a workspace of disjoint obstacles such that the convergence properties of the aforementioned DS methods are preserved.

With the increase of computational power and development of robust numerical solvers for optimization problems, optimization-based techniques, such as Model Predictive Control (MPC), have become popular. Compared to the closed form control laws, MPC allows for an easy encoding of the system constraints. MPC is typically used as a local planner given a global reference path or waypoints which are computed based on the static environment. All dynamic obstacles are accommodated in the MPC formulation. Commonly, the obstacle regions (or approximation of the regions) are explicitly expressed in the optimization problem, either as hard constraints [13]–[15] or soft constraints by including a penalizing term in the cost function [16], [17]. Due to the receding horizon nature of the MPC, these works do not provide convergence guarantees. Specifically, in environments with large obstacles, or where intersecting obstacles creates concave regions, the MPC solution may lead to local attractors at obstacle boundaries. While many MPC formulations focus on trajectory tracking, given a reference trajectory, path-following MPC [18] gives highest priority to the minimization of the robot deviation from some geometric reference path, with less focus on the velocity profile. Tunnel-following MPC [19] extends the path-following MPC scheme by imposing a constraint on tracking error, restricting the robot to stay within some specified distance to the reference path.

In this work, we present a motion control scheme which combines a DS method for receding horizon path generation with a tunnel-following MPC. This yields an attracting behavior towards the goal with ensured collision avoidance although no explicit obstacle constraints are used in the optimization problem formulation. In particular, the number of constraints for the inner control loop is independent of the number and shape of obstacles. In contrast to a pure closed form approach, embedding the closed form DS scheme in an MPC scheme allows for simple adaptation of the robot constraints to find an admissible and smooth control input. Compared to other MPC approaches, collision avoidance is here achieved by relying on a reference path generator, simplifying the formulation of the optimal control problem to be independent of workspace complexity. Overall, the contribution of the work is summarized below:

- An MPC framework that allows realization of DS-generated trajectories when they do not obey system constraints.
- The MPC solver is guaranteed to provide existence of

collision-free solutions at all times.
- The formulation of the optimal control problem in the proposed MPC is independent of workspace complexity.

II. PRELIMINARIES

A. Starshaped sets and star worlds

A set $A \subset \mathbb{R}^n$ is *starshaped with respect to* (w.r.t.) x if for every point $y \in A$ the line segment $l[x, y]$ is contained by A. The set A is said to be *starshaped* if it is starshaped w.r.t. some point, i.e. $\exists x$ s.t. $l[x, y] \subset A, \forall y \in A$. The set A is *strictly starshaped w.r.t.* x if it is starshaped w.r.t. x and any ray emanating from x crosses the boundary only once. We say that A is strictly starshaped if it is strictly starshaped w.r.t. some point.

Given a collection of obstacles $\mathcal{O} = \{\mathcal{O}^1, \mathcal{O}^2, ...\}$ in $\mathbb{R}^n$, the free space $\mathcal{F} = \mathbb{R}^n \backslash \bigcup_{\mathcal{O}^j \in \mathcal{O}} \mathcal{O}^j$ is said to be a *star world* if all obstacles are strictly starshaped. A star world where all obstacles are mutually disjoint is defined as a *disjoint star world*. For more information on starshaped sets and star worlds, see [20] and [12].

B. Obstacle avoidance for dynamical systems in star worlds

Given a star world, obstacle avoidance can be achieved using a DS approach [10] with dynamics:

$$\dot{r} = \eta(r, r^g, \mathcal{O}) = M(r, \mathcal{O})(r^g - r), \quad (1)$$

where $\mathcal{O}$ is the collection of strictly starshaped obstacles forming the star world, $\mathcal{F}^\star$, r is the current robot position and $r^g \in \mathcal{F}^\star$ is the goal position. $M(\cdot, \cdot)$ is a modulation matrix used to adjust the attracting dynamics to r^g based on the obstacles tangent spaces. Convergence to r^g is guaranteed for a trajectory following (1) from any initial position $r^0 \in \mathcal{F}^\star$ if $\mathcal{F}^\star$ is a disjoint star world and no obstacle center point is contained by the line segment $l[r^0, r^g]$. For more information, see [10], [11].

ModEnv* (Algorithm 2 in [12]) provides a way to establish a disjoint star world $\mathcal{F}^\star \subset \mathcal{F}$ from a free space, $\mathcal{F}$, formed by possibly intersecting convex and/or polygon obstacles. In essence, the algorithm combines all clusters of intersecting obstacles and extend the resulting obstacle regions such that they are strictly starshaped and mutually disjoint. The center points of the reshaped obstacles are placed outside the line segment $l[r^0, r^g]$ to satisfy the condition for convergence of the dynamics (1). The algorithm is not complete in the sense that there may be cases where it does not find a solution when such in fact exists. In case no disjoint star world is found, the algorithm returns a (intersecting) star world satisfying $\mathcal{F}^\star = \mathcal{F}$, and obstacle avoidance guarantees for (1) are remained while the convergence property is not obtained.

III. PROBLEM FORMULATION

Consider a robot of radius a operating in the Cartesian plane with discrete-time kinematic model

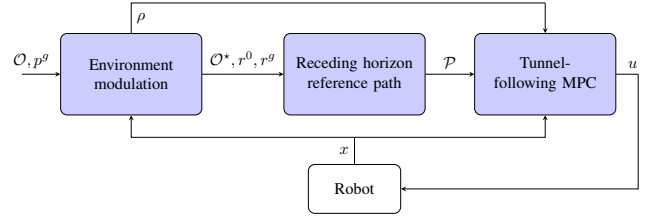$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ p_k &= h(x_k). \end{aligned} \quad (2)$$

Fig. 1: Proposed motion control scheme.

Here, $x_k \in \mathcal{X} \subset \mathbb{R}^n$ is the robot state, $p_k \in \mathbb{R}^2$ is the robot position and $u_k \in \mathcal{U} \subset \mathbb{R}^m$ is the control signal at time instance k. It is assumed that there exists a control input such that the robot does not move, i.e. $\forall x \in \mathcal{X}, \exists u' \in \mathcal{U}$ s.t. $f(x, u') = x$. The robot is operating in a world containing a collection of dynamic, possibly intersecting, obstacles, $\tilde{\mathcal{O}}_k = \{\tilde{\mathcal{O}}_k^1, \tilde{\mathcal{O}}_k^2, ...\}$, which are either convex shapes or polygons.

Remark 1: Although $\tilde{\mathcal{O}}_k$ formally contains only convex shapes and polygons, the formulation allows for more general complex obstacles as intersections are allowed. In particular, any shape can be described as a combination of several convex and polygon regions.

To take into account the robot radius we define the dilated obstacles as $\mathcal{O}_k = \{\tilde{\mathcal{O}}_k^j \oplus \mathbb{B}[\mathbf{0}, a]\}_{\tilde{\mathcal{O}}_k^j \in \tilde{\mathcal{O}}_k}$, where $\oplus$ is the Minkowski sum and $\mathbb{B}[\mathbf{0}, a]$ is the closed ball of radius a centered at $\mathbf{0} = [0, 0]^T$. The free space that includes all collision-free robot positions is then given as $\mathcal{F}_k = \mathbb{R}^2 \backslash \bigcup_{\mathcal{O}_k^j \in \mathcal{O}_k} \mathcal{O}_k^j$. The objective is to find a control policy that enforces the robot to stay in the free set at all times, $p_k \in \mathcal{F}_k, \forall k$, and drives it to a goal position $p^g \in \mathbb{R}^2$.

Assumption 1: The obstacles move slow compared to the sampling frequency, i.e. over a control sampling period, Δt, the obstacle positions are constant.

Assumption 2: The obstacles do not actively move into a region occupied by the robot, such that the implication $p_{k+1} \in \mathcal{F}_k \Rightarrow p_{k+1} \in \mathcal{F}_{k+1}$ holds.

Assumption 1 suggests that the movements of obstacles can be accurately represented using discrete motion models, which is a common approach in development of sample-based control laws. Assumption 2 implies that the obstacles are aware of the robot's position and do not appear aggressive, thereby preventing any collisions while the robot remains stationary. Note that this does not guarantee collision avoidance when the robot is in motion.

In the following sections, we will omit the time notation for convenience unless some ambiguity exists.

IV. CONTROL DESIGN

We propose a motion control scheme depicted in Fig. 1 which consists of three main components. First, the obstacles are modified to form a disjoint star world using our previous work [12]. The star world is designed as a strict subspace of the free space with any interior point having an appropriately selected minimum clearance to the obstacles. Next, a DS approach which ensures obstacle avoidance and convergence to the goal within disjoint star worlds is utilized to generate a receding horizon reference path. Finally, a tunnel-following

MPC is used to compute a control sequence which ensures close path tracking such that collision avoidance guarantees are obtained. Details are given in the following subsections.

A. Environment modulation

The proposed method relies on generating a reference path with a (time-varying) minimum clearance, ρ, to all obstacles using the DS approach (1). To this end, the set of inflated obstacles $\mathcal{O}^\rho = \{\mathcal{O}^j \oplus \mathbb{B}(\mathbf{0}, \rho)\}_{\mathcal{O}^j \in \mathcal{O}}$ is defined, see Fig. 2a, with corresponding clearance set $\mathcal{F}^\rho = \mathbb{R}^2 \setminus \bigcup_{\mathcal{O}^j \in \mathcal{O}^\rho} \mathcal{O}^j \subset \mathcal{F}$. Here, $\mathbb{B}(\mathbf{0}, \rho)$ is the open ball of radius ρ centered at $\mathbf{0}$. That is, $\mathcal{F}^\rho$ is the set of all robot positions where the closest distance to an obstacle is at least ρ. As stated in Section II-B, any star world is positively invariant for the dynamics (1) and convergence to a goal position is guaranteed for a disjoint star world. Since $\mathcal{O}^\rho$ may include both intersecting and non-starshaped obstacles, $\mathcal{F}^\rho$ provides none of the aforementioned guarantees. Thus, the objective of the environment modulation is to find a disjoint star world $\mathcal{F}^\star \subset \mathcal{F}^\rho$ containing an initial position, r^0, and a goal position, r^g, for the reference path. A procedure to specify ρ and to compute $\mathcal{O}^\star$, r^0 and r^g is given in Algorithm 1 and the steps are elaborated below.

Algorithm 1: Environment modulation

Parameters: γ, $\bar{\rho}$
Input   : $\mathcal{O}$, p^g, p
Output   : $\mathcal{O}^\star, r^0, r^g, \rho$

1   $\rho \leftarrow \bar{\rho}/\gamma$;
2   **do**
3 | $\rho \leftarrow \gamma\rho$;
4 | $\mathcal{P}^0 \leftarrow \mathbb{B}(p, \rho) \setminus \{\mathcal{O}^j \oplus \mathbb{B}(\mathbf{0}, \rho)\}_{\mathcal{O}^j \in \mathcal{O}}$;
5   **while** $\mathcal{P}^0 = \emptyset$;
6   $\mathcal{O}^\rho \leftarrow \{\mathcal{O}^j \oplus \mathbb{B}[\mathbf{0}, \rho]\}_{\mathcal{O}^j \in \mathcal{O}}$;
7   $r^0 = \arg\min_{r^0 \in \mathcal{P}^0} \|r^0 - p\|_2$;
8   $r^g = \arg\min_{r^g \notin \mathcal{O}^\rho} \|r^g - p^g\|_2$;
9   Compute $\mathcal{O}^\star$ using ModEnv* with $\mathcal{O}^\rho$, r^0 and r^g;
10   **foreach** $\mathcal{O}^j \in \mathcal{O}^\star$ **do**
11 | **if** $CH(\mathcal{O}^j) \cap \left\{ r^0 \cup r^g \cup \bigcup_{\mathcal{O}^l \in \mathcal{O}^\star \setminus \mathcal{O}^j} \mathcal{O}^l \right\} = \emptyset$
        | **then**
12 | | $\mathcal{O}^j \leftarrow CH(\mathcal{O}^j)$
13 | **end**
14   **end**

---

1) Clearance selection (line 1-5): The initial reference position is chosen within the initial reference set $\mathcal{P}^0 = \mathcal{F}^\rho \cap \mathbb{B}(p, \rho)$, depicted as green area in Fig. 2a. To have a valid initial reference position, ρ is set to a strict positive value such that $\mathcal{P}^0$ is nonempty. Such a ρ is guaranteed to exist for any collision-free robot position[1], $p \in \mathcal{F}$. In Algorithm 1, ρ is initially set to a base value $\bar{\rho}$ and reduced by a factor γ until $\mathcal{P}^0 \neq \emptyset$. Here, $\bar{\rho} \in \mathbb{R}^+$ and $\gamma \in (0, 1)$ are algorithm parameters.

---

[1] $\mathcal{F}$ is an open set, so $\exists \rho > 0$ s.t. $\mathbb{B}(p, \rho) \in \mathcal{F}$. Thus, the closest distance to an obstacle is at least ρ, i.e. $p \in \mathcal{F}^\rho$, and it follows that $p \in \mathcal{P}^0 \Rightarrow \mathcal{P}^0 \neq \emptyset$.

2) Initial and goal reference position selection (line 7-8): The reference path should ideally be a curve from the current robot position, p, to the goal, p^g. However, since $\mathcal{F}^\rho$ is a strict subset of $\mathcal{F}$ it is possible that $p \notin \mathcal{F}^\rho$ or $p^g \notin \mathcal{F}^\rho$. In particular, this occurs when the robot or goal position is located closer than a distance ρ to an obstacle. To account for these situations, we define the initial reference position $r^0 = \arg\min_{r^0 \in \mathcal{P}^0} \|r^0 - p\|_2$ and reference goal $r^g = \arg\min_{r^g \in \mathcal{F}^\rho} \|r^g - p^g\|_2$.

3) Establishment of a disjoint star world (line 9): ModEnv* is used to construct a disjoint star world $\mathcal{F}^\star \subset \mathcal{F}^\rho$ based on the inflated obstacles, $\mathcal{O}^\rho$, such that $r^0 \in \mathcal{F}^\star$ and $r^g \in \mathcal{F}^\star$.

4) Convexification (line 10-14): While convergence to the goal position is guaranteed following the dynamics (1) for any disjoint star world, the behaviour is not always the most intuitive in concave regions. To obtain a more direct path, any concave obstacle is made convex using the convex hull provided this does not violate the two conditions: 1) r^0 and r^g remain exterior points of the obstacle, and 2) the resulting obstacle region does not intersect with any other obstacle. When the obstacles are made convex, unnecessary "detours" in concave regions are avoided, compare Fig. 2b and 2d.

B. Receding horizon reference path

The reference path is given as a parametrized regular curve

$$\mathcal{P} = \left\{ r \in \mathbb{R}^2 : s \in [0, N] \rightarrow r(s) \right\} \tag{3}$$

where $N \in \mathbb{N}^+$ is a parameter determining the path horizon and r is given by the solution to the ODE

$$\frac{dr(s)}{ds} = \Delta p_{\max} \bar{\eta}(r(s), r^g, \mathcal{O}^\star), \quad r(0) = r^0. \tag{4}$$

Here, $\bar{\eta}(\cdot, \cdot, \cdot) = \frac{\eta(\cdot, \cdot, \cdot)}{\|\eta(\cdot, \cdot, \cdot)\|_2}$ are the normalized dynamics in (1) and $\Delta p_{\max} = \max_{u \in \mathcal{U}, x \in \mathcal{X}} \|\frac{\partial h}{\partial x}(x) f(x, u)\|_2$ is the maximum linear displacement which can be achieved by the robot in one sampling instance. The use of the normalized dynamics is instrumental for the MPC problem formulation. Since $\bar{\eta}$ is a vector of unit length, the arc length of the receding horizon path, $\mathcal{P}$, is $\Delta p_{\max} N$ unless the dynamics (4) converge to r^g before this distance is reached. As the path is initialized in the star world $\mathcal{F}^\star$ and the dynamics are positively invariant in any star world, we have $\mathcal{P} \subset \mathcal{F}^\star$. Moreover, since $\mathcal{F}^\star \subset \mathcal{F}^\rho$, any point in $\mathcal{P}$ is at least at a distance ρ from any obstacle in $\mathcal{O}$. Assuming $\mathcal{F}^\star$ has successfully been constructed as a disjoint star world, r is guaranteed to converge towards r^g, i.e. $\lim_{N \to \infty} r(N) = r^g$.

C. Tunnel-following MPC

To find a control input which drives a robot with dynamics (2) along the reference path (3), a nonlinear MPC is formulated. The objective is to find a solution which result in a fast movement while staying close enough to the reference path such that collision avoidance is obtained. To derive a trajectory from the reference path, the state is extended with the path coordinate s and a virtual control signal Δs is introduced which determines the path coordinate increment,

i.e. $s_{i+1} = s_i + \Delta s_i$. The MPC is formulated based on the extended state $\xi_i = [\bar{x}_i^T \ s_i]^T$ with dynamics

$$\xi_{i+1} = \begin{bmatrix} f(\bar{x}_i, \bar{u}_i) \\ s_i + \Delta s_i \end{bmatrix} = f_\xi(\xi_i, z_i), \tag{5}$$

where $z_i = [\bar{u}_i^T \ \Delta s_i]^T$ is the extended control signal. Here we have used the notation $\bar{x}_i$ and $\bar{u}_i$ to distinguish the internal variables of the controller from the real system variables. According to (3), the path coordinate is restricted to $[0, N]$ and we have the state constraint $\xi_i \in \Xi = \mathcal{X} \times [0, N]$. The control input is constrained by $z_i \in \mathcal{Z} = \mathcal{U} \times [0, 1]$. The lower bound on Δs is chosen to ensure a forward motion of the reference trajectory along $\mathcal{P}$ and the upper bound is set to 1 corresponding to a reference position movement equal to the maximal achievable linear displacement of the robot. Similar to a tunnel-following MPC scheme [19], we impose a constraint on the tracking error, $\varepsilon_i = \|r(s_i) - h(\bar{x}_i)\|_2$, such that the robot position is in a ρ-neighborhood of the reference position[2]. The optimization problem for the MPC is proposed as follows:

$$\min_z \qquad -c_s s_N + c_e \varepsilon_N^T \varepsilon_N \tag{6a}$$

$$\text{s.t.} \qquad \xi_0 = [x^T, 0]^T \tag{6b}$$

$$i \in \mathcal{N}: \qquad \xi_{i+1} = f_\xi(\xi_i, z_i) \tag{6c}$$

$$z_i \in \mathcal{Z} \tag{6d}$$

$$\xi_{i+1} \in \Xi \tag{6e}$$

$$\varepsilon_{i+1} < \rho \tag{6f}$$

where $\mathcal{N} = \{0, .., N-1\}$. The positive scalars c_s and c_e are tuning parameters. The cost function is designed to motivate a solution where the robot moves in a forward direction of $\mathcal{P}$. This is done by maximizing the final path coordinate, s_N, or equivalently since $s_N = \sum_{i \in \mathcal{N}} \Delta s_i$, maximizing path increment at each time instance. That is, the reference position is desired to move along $\mathcal{P}$ at a fast rate. This in turn drives the robot position in the same direction due to the coupling effect of the tracking error constraint (6f). At the same time, (6f) restricts the reference position from diverging from the robot position at robot configurations when the linear velocity in the tangential direction of $\mathcal{P}$ is limited. To boost forward motion further, and to provide converging behavior to p_g when $r_N = p_g$, a penalizing term on the final tracking error is also included. The cost function can be tailored for the robot at hand to favor certain behaviors. For instance, a regularization term on (changes of) the control input can be included to provide a smoother trajectory. Moreover, a terminal cost to reach a desired final orientation can be incorporated.

The control law is given by

$$u = \bar{u}_0^* \tag{7}$$

where $\bar{u}_0^*$ is extracted from the initial control input, z_0^*, of the optimal solution, z^*, to (6). Although no explicit soft

---

[2]In contrast to [19] we apply strict, and not soft, constraints on the tracking error. This can be done and still ensure existence of solution from the design of the reference path. In particular, since $r(0) \in \mathcal{P}^0 \subset \mathbb{B}(p, \rho)$.

or hard constraints regarding the obstacles are used in the MPC formulation, obstacle avoidance is achieved as stated by the following theorem. This is obtained by ensuring a close tracking, $\|\varepsilon\|_2 < \rho$, of the path which is at least at a distance ρ from any obstacle.

Theorem 1: The trajectory for a robot with dynamics (2) under control law (7) is collision-free with respect to the obstacles $\mathcal{O}_k$, i.e. $p_k \in \mathcal{F}_k, \forall k$.

Proof: According to Assumption 1 and 2 it suffices to show that $p_{k+1} \in \mathcal{F}_k$ at any sampling instance k. First, we show existence of a solution to (6) at all times. Define the trivial solution, z', as $z_i' = [\bar{u}'^T, 0]^T, \forall i \in \mathcal{N}$ where $\bar{u}' \in \mathcal{U}$ s.t. $f(x_k, \bar{u}') = x_k$. Obviously, this satisfies constraint (6d). From (6b)-(6c) we get $\xi_{i+1} = [x_k^T, 0]^T, \forall i \in \mathcal{N}$. With current robot state $x_k \in \mathcal{X}$, we can conclude that (6e) is satisfied. Moreover, this gives $\varepsilon_{i+1} = r_k^0 - p_k, \forall i \in \mathcal{N}$. By construction, $r_k^0 \in \mathcal{P}_k^0 \subset \mathbb{B}(p_k, \rho_k)$ and it follows that $\varepsilon_{i+1} < \rho_k, \forall i \in \mathcal{N}$, satisfying constraint (6f). Hence, z' is a feasible solution. Any solution, z, to (6) satisfies $\varepsilon_{i+1} < \rho_k, \forall i \in \mathcal{N}$. This implies $h(\bar{x}_{i+1}) \in \mathcal{P}_k \oplus \mathbb{B}(\mathbf{0}, \rho_k), \forall i \in \mathcal{N}$. Since $\mathcal{P}_k \subset \mathcal{F}_k^\rho$ we thus have $h(\bar{x}_1) \in \mathcal{F}_k$. Applying control law (7), $u_k = \bar{u}_0$, leads to $x_{k+1} = f(x_k, \bar{u}_0) = \bar{x}_1$ from (6b)-(6c). Hence, $p_{k+1} = h(\bar{x}_1) \in \mathcal{F}_k$. ∎

The MPC works as a bridge to incorporate the robot constraints and find an admissible control sequence resulting in a smooth path within the clearance distance to the reference path. From this perspective, a short horizon is sufficient, e.g. 2-3 samples. However, when the orientation for a non-holonomic robot is poorly aligned with the reference path direction, the MPC may yield a solution where the robot is standing still if the constraints prohibits the robot from realigning sufficiently fast within the horizon. The horizon should thus be adapted for the admissible robot reorientation abilities, e.g. angular velocity limit for a unicycle robot.

V. Implementation aspects[3]

To implement the motion control scheme presented in Section IV in a real-time application, possible adjustments are here presented. The adjustments aim at reducing the computational complexity and obtaining a fixed upper bound on computation time.

1) Maximum environment modulation time: For a fixed upper bound on computation time, Algorithm 1 is terminated after a specified maximum time. If the algorithm is terminated prematurely before $\mathcal{O}^\star$ has been generated (before line 9), $\mathcal{O}^\star$ is set such that $\mathcal{F}^\star = \mathcal{F}^\rho$ where any non-starshaped polygon in $\mathcal{O}^\rho$ is divided into sub-obstacles by convex decomposition. In this way, $\mathcal{F}^\star$ is a star world and positively invariant for the dynamics (4).

2) Maximum reference generation time: In practice, the reference path is computed by discrete integration of (4) over an interval $s \in [0, N]$. The integration step size depend on current proximity to $\mathcal{O}^\star$ in order to guarantee that no obstacle boundary is penetrated. Similar to the environment modulation, a maximum computation time is used to terminate the

---

[3]Code is available at https://github.com/albindgit/starworld_tunnel_mpc

simulation when exceeded. In case the path generation is terminated prematurely, at $s' < N$, the path is extended with a static final position, i.e. $r(s) = r(s'), \forall s \in [s', N]$.

3) Reference buffering: It is practical to reuse the computed reference path from previous sampling instance, appropriately shifted, if it is still collision-free and within a distance ρ from the robot position, i.e. if $\mathcal{P}_{k-1} \subset \mathcal{F}_k^\star$ and $\mathcal{P}_{k-1} \cap \mathbb{B}(p_k, \rho_k) \neq \emptyset$. The reference path integration is then initialized at the final position of $\mathcal{P}_{k-1}$.

4) Reference approximation: To simplify calculation of ε_i, the discrete reference path, $\mathcal{P}$, is replaced by a function approximation, $\hat{r}(s)$, e.g. using polynomial regression. The approximation is biased to enforce $\hat{r}(0) = r(0)$ such that Theorem 1 is not compromised. To account for approximation errors, the upper bound for the tracking error, $\varepsilon_i = \|\hat{r}(s_i) - h(\bar{x}_i)\|_2$, in (6f) is adjusted to $\rho - \epsilon$, where ϵ is the maximum approximation error

$$\epsilon = \max_{s \in [0,N]} \|\hat{r}(s) - r(s)\|_2. \qquad (8)$$

It is assumed that a sufficiently flexible function representation is used such that $\epsilon < \rho$.

VI. RESULTS

We consider two scenarios: 1) a static scene with variations, and 2) a moving obstacle in a static scene with corridors. The first one illustrates the environment modulation procedure and convergence while the second one illuminates the properties of the MPC. Considering unicycle robot dynamics and using forward-Euler discretization with sampling time, Δt, the discrete robot model is

$$f(x, u) = x + \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \Delta t, \quad h(x) = \begin{bmatrix} p^x \\ p^y \end{bmatrix}, \qquad (9)$$

where $x = [p^x, p^y, \theta]^T$ are the Cartesian position [m] and orientation [rad] of the robot. The control inputs $u = [v\ \omega]^T$ are the linear and angular velocities which are bounded by $[0, 1.5]$ m/s and $[-1.5, 1.5]$ rad/s, respectively. The sampling rate is $\Delta t = 0.2$s and as function approximation of the reference path, a polynomial of degree 10 is used. For a smooth trajectory, the stage cost is extended with a regularization term on control input variation, $\sum_{i \in \mathcal{N}} \Delta \bar{u}_i^T R \Delta \bar{u}_i$. Here, $\Delta \bar{u}_i = \bar{u}_i - \bar{u}_{i-1}$ is the control variation, with $\bar{u}_{-1} = u_{k-1}$ being the previously applied control input and R being a positive-definite 2×2 matrix. The numerical values for the control parameters are $\bar{\rho} = 0.3$, $\gamma = 0.5$, $N = 5$, $c_s = 500$, $c_e = 100$, and $R = \text{diag}(250, 2.5)$.

In Fig. 2 a static scene with five intersecting obstacles is shown. As seen in Fig. 2b-2d, the resulting starshaped obstacle is depending on the robot position, goal position and other obstacles not in the combined cluster. When the starshaped obstacle can be extended with the convex hull, the path to the goal is more direct compared to if it is a concave obstacle. In each case, however, the guiding vector field has the goal as a global attractor.

In Fig. 3 a scenario with three static polygons forming two corridors and a moving circular obstacle is depicted.
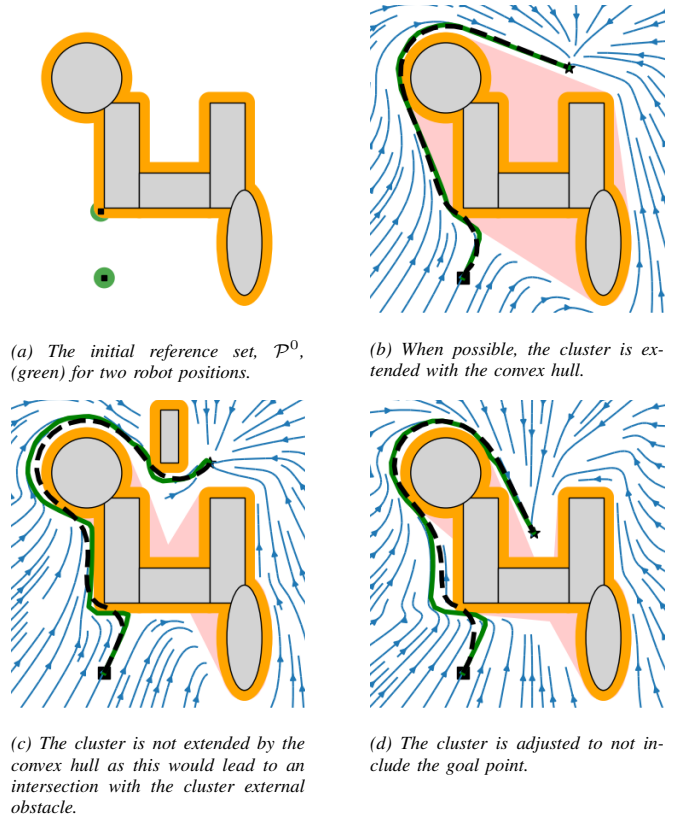

(a) The initial reference set, $\mathcal{P}^0$, (green) for two robot positions.

(b) When possible, the cluster is extended with the convex hull.

(c) The cluster is not extended by the convex hull as this would lead to an intersection with the cluster external obstacle.

(d) The cluster is adjusted to not include the goal point.

Fig. 2: Each obstacle in $\mathcal{O}$ (grey) is dilated with ρ to form $\mathcal{O}^\rho$ (yellow). Next, all intersecting obstacles are combined and extended with the starshaped hull (convex hull if possible) to form $\mathcal{O}^\star$ (red). The resulting robot path from initial position (square) to goal position (star) is shown as dashed black line. The vector field (blue arrows) and resulting path from initial position (green line) for the reference dynamics (4) are also shown.

The robot starts on one side of the corridors while the goal position is placed on the opposite side. As seen, the robot initially enters the lower corridor to take the most direct path towards the goal but reroutes its path to the upper corridor as the lower gets blocked. In situations where the robot is well aligned with the reference path (e.g. Fig. 3a and 3c), the path speed can be set close to maximal such that $s_N \approx N$. That is, the last reference point, $\hat{r}(s_N)$, and thus the predicted robot position according to the coupling constraint (6f), is close to the end of the reference path $\mathcal{P}$. During the rotation in the lower corridor (Fig. 3b), the kinematics of the robot in combination with the tracking error constraint prevent the path speed to be maximal such that $s_N \ll N$, i.e. $\hat{r}(s_N)$ is not at the end of $\mathcal{P}$, and the robot is moving more slowly, see also Fig. 4.

VII. CONCLUSION

This paper proposed a motion control scheme for robots operating in a Cartesian plane containing a collection of dynamic, possibly intersecting, obstacles. The method combines previous work by the authors to adjust the workspace into a scene of disjoint obstacles with a closed form dynamical system formulation to generate a receding horizon path. An MPC controller is used to compute an admissible control sequence yielding attracting behavior towards the goal with close enough path tracking to ensure collision avoidance.
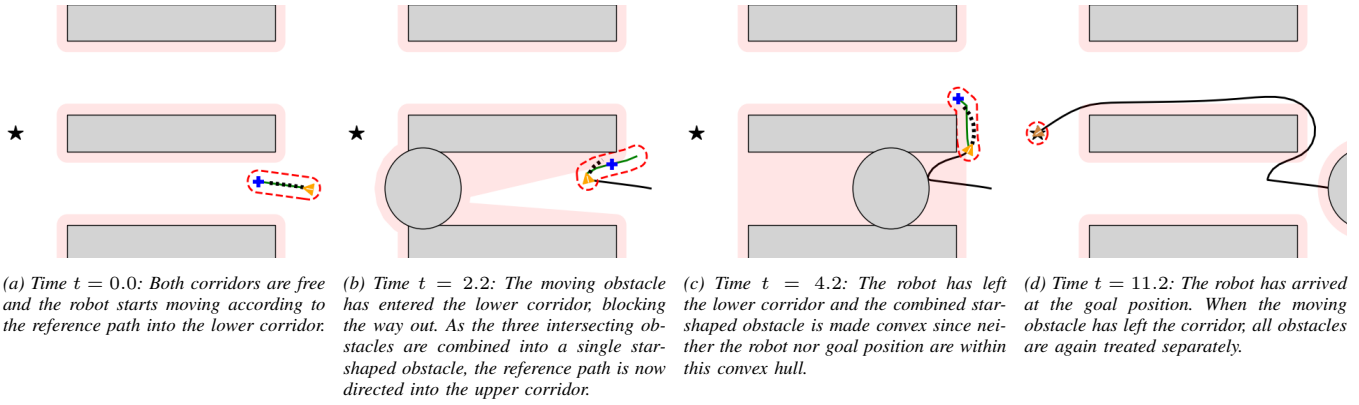
(a) Time $t = 0.0$: Both corridors are free and the robot starts moving according to the reference path into the lower corridor.

(b) Time $t = 2.2$: The moving obstacle has entered the lower corridor, blocking the way out. As the three intersecting obstacles are combined into a single starshaped obstacle, the reference path is now directed into the upper corridor.

(c) Time $t = 4.2$: The robot has left the lower corridor and the combined starshaped obstacle is made convex since neither the robot nor goal position are within this convex hull.

(d) Time $t = 11.2$: The robot has arrived at the goal position. When the moving obstacle has left the corridor, all obstacles are again treated separately.

Fig. 3: A unicycle robot in a double corridor scenario with a circular moving obstacle entering one corridor. The obstacles, $\mathcal{O}$, are shown as grey regions, the dilated starshaped obstacles, $\mathcal{O}^\star$, are shown as red regions, the goal position is depicted as black star and the yellow triangle indicates the current robot position and orientation. The reference path, $\mathcal{P}$, is shown as green line with boundary of accompanying tunnel region, $\mathcal{P} \oplus \mathbb{B}(0, \rho)$, depicted with dashed red line. The final reference point in the MPC horizon, $\hat{r}(s_N)$, is shown with blue cross and the predicted robot trajectory for the MPC solution, $h(\bar{x}_i)$, $i \in \mathcal{N}$, is shown as dotted black line. The travelled path is shown as black solid line.

(a) Linear velocity, v [m/s].
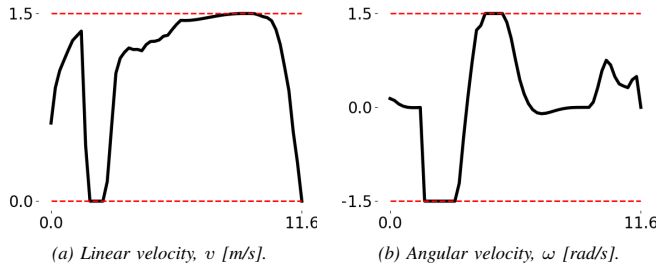
(b) Angular velocity, ω [rad/s].

Fig. 4: The applied control signals in the double corridor scenario. First, the robot accelerates until the lower corridor is blocked, at which point it decelerates to full stop and completes a U-turn. Next, it turns around the center wall into the upper corridor and accelerates to and maintains full speed. Finally, it decelerates to reach the final goal position.

As the method relies on conservative treatment of the obstacle regions, a natural drawback is the possible gap closing in narrow passages. While conditions for global convergence of the reference path can be derived, it might be lost by use of the MPC depending on the horizon and robot constraints. In future work, we aim at achieving convergence guarantees for the full motion control scheme. In the current formulation, environment boundaries are not considered. As robots commonly are restricted to some specific area, e.g. a room or building, it is of interest to include specification of a bounded workspace as well.

REFERENCES

[1] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 500–505, 1985.

[2] M. Ginesi, D. Meli, A. Calanca, D. Dall'Alba, N. Sansonetto, and P. Fiorini, "Dynamic movement primitives: Volumetric obstacle avoidance," in *2019 19th International Conference on Advanced Robotics (ICAR)*, pp. 234–239, 2019.

[3] S. Stavridis, D. Papageorgiou, and Z. Doulgeri, "Dynamical system based robotic motion generation with obstacle avoidance," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 712–718, 2017.

[4] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[5] S. G. Loizou, "Closed form navigation functions based on harmonic potentials," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 6361–6366, 2011.

[6] S. Paternain, D. E. Koditschek, and A. Ribeiro, "Navigation functions for convex potentials in a space with convex obstacles," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 2944–2959, 2018.

[7] C. Connolly, J. Burns, and R. Weiss, "Path planning using Laplace's equation," in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 2102–2106 vol.3, 1990.

[8] H. Feder and J.-J. Slotine, "Real-time path planning using harmonic potentials in dynamic environments," in *Proceedings of International Conference on Robotics and Automation*, vol. 1, pp. 874–881 vol.1, 1997.

[9] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," in *2008 American Control Conference*, pp. 4609–4614, 2008.

[10] L. Huber, A. Billard, and J.-J. Slotine, "Avoidance of convex and concave obstacles with convergence ensured through contraction," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1462–1469, 2019.

[11] L. Huber, J.-J. Slotine, and A. Billard, "Avoiding dense and dynamic obstacles in enclosed spaces: Application to moving in crowds," *IEEE Transactions on Robotics*, pp. 1–10, 2022.

[12] A. Dahlin and Y. Karayiannidis, "Creating star worlds: Reshaping the robot workspace for online motion planning," *IEEE Trans. Robotics*, pp. 1–16, 2023.

[13] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[14] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 972–983, 2021.

[15] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 07 2019.

[16] I. Sánchez, A. D'Jorge, G. V. Raffo, A. H. González, and A. Ferramosca, "Nonlinear model predictive path following controller with obstacle avoidance," *Journal of Intelligent and Robotic Systems*, vol. 102, may 2021.

[17] J. ji, A. Khajepour, W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, pp. 1–1, 01 2016.

[18] T. Faulwasser and R. Findeisen, "Nonlinear model predictive control for constrained output path following," *IEEE Transactions on Automatic Control*, vol. 61, no. 4, pp. 1026–1039, 2016.

[19] N. van Duijkeren, *Online Motion Control in Virtual Corridors - for Fast Robotic Systems*. Phd thesis, KU Leuven, [Online]. Available: https://lirias.kuleuven.be/retrieve/527169, 2019.

[20] G. Hansen, I. Herburt, H. Martini, and M. Moszyńska, "Starshaped sets," *Aequationes mathematicae*, vol. 94, 12 2020.