Reactive Path Coordination Based on Time-Scaled Collision Cone

Arun Kumar Singh[1] and Quang Cuong Pham [2]
Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798

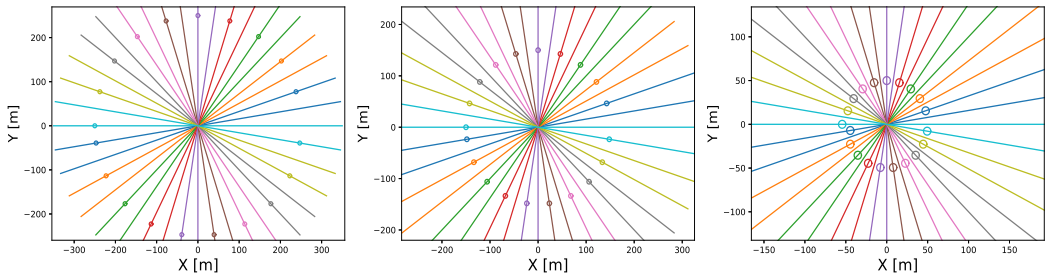
I. Introduction

There are many applications where multiple autonomous (or semi-autonomous) agents share a common workspace. For example, in applications like search and rescue operations and surveillance multiple fixed wing unmanned aerial vehicles (UAVs) are often deployed to navigate along strategically designed paths. Similarly, due to increased air-traffic, multiple commercial aircrafts are now routinely present in a given airspace at any time instant. A key requirement in any application involving multiple agents is to ensure collision avoidance. One of the efficient approaches for collision avoidance is to appropriately coordinate the motions of the agents along pre-specified paths, resulting in the so-called *path-coordination* problem [1]. The problem of *path-coordination* is usually solved as an optimization problem which takes the form of either mixed-integer [1], [2], [3], [4], [5], non-linear [6] or a convex problem [7] although sampling-based planners like [8, 9] have also been proposed. These cited approaches are designed as an off-line planning framework and their extension to reactive on-line setting is not straightforward. The most straightforward approach for the reactive setting would be to solve the optimizations in [1]-[7] over a shorter horizon (small segments of the given paths). However, we have observed in our studies that such straightforward extension proves to be insufficient. In particular, we have observed two potential bottlenecks. Firstly, since [1]-[7] model collision as the overlap of agents' footprints at a given time instant, the choice of the length of the horizon becomes very critical. As shown in Fig.1(a)-1(c), choosing a very small horizon

---

[1] Research Fellow, ATMRI, School of Mechanical and Aerospace Engineering
[2] Assistant Professor, School of Mechanical and Aerospace Engineering

may mean that collisions are detected only when agents are very close to each other and this in turn may lead to deadlocks. Secondly, a smaller horizon may not even lead to any computational gain. For example in conflicts like that shown in Fig.1(a)-1(c), collisions with all the agents needs to be considered even while considering a relatively small horizon. This in turn may prevent any reduction of binary variables in the mixed integer formulation [1]-[5]. Further, even in the convex approach of [7], the number of agents and conflicts considered is one of the key factors deciding the computation time.



(a) (b) (c)

Fig. 1 A receding horizon implementation of the *path coordination* framework proposed in [4]. The shape of the agents are approximated as discs which are shown as circles.

A. Overview of the Proposed Approach

The proposed approach is built on the concept of collision cone (CC) [10] which models collision in the combined position-velocity space. Thus, on straight line paths, all imminent collisions can be modeled by just knowing the current position and velocity of the agents. On curvilinear paths, the collision modeling relies on piece-wise linear approximation. The use of the concept of CC allows for the development of one-step receding horizon like framework wherein the next instant velocity of the agents are continuously updated based on their current positions and velocities.

In our earlier works [11], [12] we introduced the concept of *time scaled collision cone* (TSCC) which is a projection of CC constraints along a given path. In this paper, we present several improvements over these works to achieve reactive on-line path coordination. Firstly, we formulate TSCC over the joint velocity space of all the agents. For a system of n agents, the newly formulated TSCC models the n dimensional velocity space. In contrast, our earlier works [11], [12] only considered a

single agent at a time and thus modeled only a one-dimensional velocity space. Secondly, we show that the intersection space of reachable velocities with joint TSCC can simplified to a convex set defined by linear inequalities. This is an important result since only a few frameworks like [7] have previously attempted to provide a convex approximation to the difficult *path coordination* problem. In contrast to [7], the proposed formulation does not require a priori knowledge of priorities in which agents enter and depart from the collision zones. Rather, a feasible priority is computed automatically along with the coordination plans. Further, even for a given priority, the proposed framework computes less conservative coordination plans than [7], is computationally faster and consequently more suitable for reactive navigation.

We would like to point out that although CC or its analogous velocity obstacles (VO) [13] have been used extensively for reactive collision avoidance [14], the setting where the paths of the agents are fixed is not that well studied in the context of reactive navigation. Besides our earlier works cited above, we are only aware of [15] which tackles this problem. The technical approach followed in [15] is very different. However, similar to [11], [12], it also considers a single agent at a time and furthermore works under the assumption of instantaneous velocity change. In contrast, our proposed formulation is much more general as it considers the joint velocity space of all the agents and incorporates hard velocity and acceleration bounds.

II. Preliminaries

A. Original Collision Cone

Consider a pair of agents at time t_0 and assume that their footprints do not overlap at the current position. A pair of trajectories $\mathbf{x}_i(t), \mathbf{x}_j(t)$ is said to be collision-free $\forall t \in (t_0, \infty)$ if it satisfies the following constraint.

$$\frac{(\mathbf{r}_{ij}^T \mathbf{v}_{ij})^2}{\|\mathbf{v}_{ij}\|^2} - \|\mathbf{r}_{ij}\|^2 + R_{ij}^2 \leq 0, R_{ij} = R_i + R_j, \mathbf{r}_{ij} = \begin{bmatrix} x_i(t_0) - x_j(t_0) \\ y_i(t_0) - y_j(t_0) \end{bmatrix}, \mathbf{v}_{ij} = \begin{bmatrix} \dot{x}_i(t_0) - \dot{x}_j(t_0) \\ \dot{y}_i(t_0) - \dot{y}_j(t_0) \end{bmatrix}. \quad (1)$$

Inequality (1) is called the collision cone constraint [10] and it assumes that the velocities of the agents remain constant at $(\dot{x}_i(t_0), \dot{y}_i(t_0))$ and $(\dot{x}_j(t_0), \dot{y}_j(t_0))$ respectively $\forall t \in (t_0, \infty)$. Under the assumption of instantaneous velocity change, the solution space of (1) directly characterizes

the space of straight line paths and their corresponding velocity profiles which a pair of agents can employ for collision avoidance. It is worth pointing out that collision between a pair of agents can be detected by noting whether the l.h.s of (1) is greater than 0 along with $(\mathbf{r}_{ij}^T \mathbf{v}_{ij}) \leq 0$. Consequently, thereafter satisfaction of (1) would be sufficient for collision avoidance.

B. Time Scaling

Let us consider a path $\mathbf{x}(s) = (x(s), y(s))$, where $s \in [s_0 \;\; s_f]$ is the path variable. The space of velocities and accelerations that can be achieved at each point along the path can be parameterized in terms of a function $\frac{ds}{dt}$ in the following manner.

$$\dot{\mathbf{x}}(t) = \mathbf{x}^{'}(s)\frac{ds}{dt}, \;\; \ddot{\mathbf{x}}(t) = \mathbf{x}^{''}(s)(\frac{ds}{dt})^2 + \mathbf{x}^{'}(s)\frac{d^2s}{dt^2}, \frac{ds}{dt} = \dot{s}(s), \frac{d^2s}{dt^2} = \ddot{s}(s). \tag{2}$$

The function $\frac{ds}{dt}$ is called the scaling function and decides the transformation of the path $\mathbf{x}(s)$ to trajectory $\mathbf{x}(t)$ in some time interval $t \in [t_0 \;\; t_f]$. Please note in (2), the path derivatives $x^{'}(s), y^{'}(s)$ etc. are known and thus the space of velocities and accelerations are completely determined by the choice of scaling function.

III. Time Scaled Collision Cone (TSCC) Based Collision Avoidance

In our previous work [11], [12], we formulated TSCC for a single agent assuming either all agents compute their motions independently [11] or that there is only a single decision making agents and all other agents are non-reactive dynamic obstacles [12]. Thus, TSCC in these earlier works characterized a single dimensional velocity space. Here, we extend TSCC to characterize the joint space of collision free velocities of a pair of agents in collision. Thus, for a system of n agents, TSCC over all the pair of agents would characterize a n dimensional space of collision free velocities. To this end, consider Fig.2 which shows two agents whose straight line paths, $\mathbf{x}_i(s)$, $\mathbf{x}_j(s)$ entail a collision zone around the paths' intersection. At current time t_0, the position of the agents are given by arc length s_0 on their respective paths. Let arc length, $s_c > s_0$ define a look-ahead point on each path. Let the respective agents be associated with scaling functions $\dot{s}_i(s)$ and $\dot{s}_j(s)$ for the path segment between s_0 and s_c. Due to the application of these scaling functions, the agents move from

the arc length s_0 to s_c in the time interval $[t_0 \;\; t_i^c]$ and $[t_0 \;\; t_j^c]$ respectively. The space of velocities that can be achieved at arc length s_c by the respective agents are characterized using (2) in the following manner.

$$(\dot{x}_i(t_c^i), \dot{y}_i(t_c^i)) = \dot{s}_i(s_c)(x_i^{'}(s_c), y_i^{'}(s_c)), (\dot{x}_j(t_c^j), \dot{y}_j(t_c^j)) = \dot{s}_j(s_c)(x_j^{'}(s_c), y_j^{'}(s_c)). \tag{3}$$

Now, according to collision cone concept, if the agents are collision free in the some time interval $[\max(t_i^c, t_j^c), t_f)$, then the pair of position and velocities, $(x_i(s_c), y_i(s_c), \dot{s}_i(s_c)x_i^{'}(s_c), \dot{s}_i(s_c)y_i^{'}(s_c))$ and $(x_j(s_c), y_j(s_c), \dot{s}_j(s_c)x_j^{'}(s_c), \dot{s}_j(s_c)y_j^{'}(s_c))$ should satisfy collision cone constraint (1). Consequently, we obtain the following form for TSCC.

$$\frac{((\mathbf{r}_{ij}^s)^T\mathbf{v}_{ij}^s)^2}{\|\mathbf{v}_{ij}^s\|^2} - \|\mathbf{r}_{ij}^s\|^2 + R_{ij}^2 \leq 0, \mathbf{r}_{ij}^s = \begin{bmatrix} x_i(s_c) - x_j(s_c) \\ \\ y_i(s_c) - y_j(s_c) \end{bmatrix}, \mathbf{v}_{ij} = \begin{bmatrix} \dot{s}_i(s_c)x_i^{'}(s_c) - \dot{s}_j(s_c)x_j^{'}(s_c) \\ \\ \dot{s}_i(s_c)y_i^{'}(s_c) - \dot{s}_j(s_c)y_j^{'}(s_c) \end{bmatrix}. \tag{4}$$

In (4), the look ahead points defined by s_c are known and thus consequently, $\mathbf{r}_{ij}^s$ is also known for a given pair of paths. Inequalities (4) thus, can be represented in a compact form as the following generic quadratic inequality.

$$A_{ij}\dot{s}_i(s_c)^2 + B_{ij}\dot{s}_i(s_c)\dot{s}_j(s_c) + C_{ij}\dot{s}_j(s_c)^2 \leq 0. \tag{5}$$

Where, A_{ij}, B_{ij} and C_{ij} are functions of $x_i(s_c), x_i^{'}(s_c), x_j(s_c), x_j^{'}(s_c), y_i(s_c), y_i^{'}(s_c)$.... etc and are thus known for a given set of paths. Extension of (5) to multiple agents is straightforward and from here on, we assume such a construction has been performed for a system of n agents. Few important points regarding (5) are worth pointing out. Firstly, its dependency on relative position and velocities is same as the original collision cone. Secondly, for a pair of agents, $\dot{s}_i(s_c)$, $\dot{s}_j(s_c)$ models the priority among them. For example, $\dot{s}_i(s_c) > \dot{s}_j(s_c)$ means that agent i leads while passing through point defined by s_c. Finally, (5) represents an algebraic (not differential) quadratic inequality with respect to variables $\dot{s}_i(s_c), \dot{s}_j(s_c)$.

A. Role of lookahead point

The role of lookahead point defined by s_c in TSCC (4) is to relax the requirement of instantaneous velocity change associated with the original collision cone (1). Essentially, what we have done here is that instead of computing velocities which is safe between the arc lengths s_0 and s_f, we compute those which ensure safety between the arc lengths s_c and s_f. The path segment between s_0 and s_c provides the space for the achievement of collision free velocities. Naturally, there is an implicit requirement here that the agents do not collide between the path segment defined by s_0 and s_c while attempting to reach the collision free velocities. This requirement can be met by construction in the following manner. From the current position of the agents given by s_0 we look for a lookahead point given by s_c on their respective paths such that the footprints of the agents do not overlap anywhere in the path segment between s_0 and s_c.

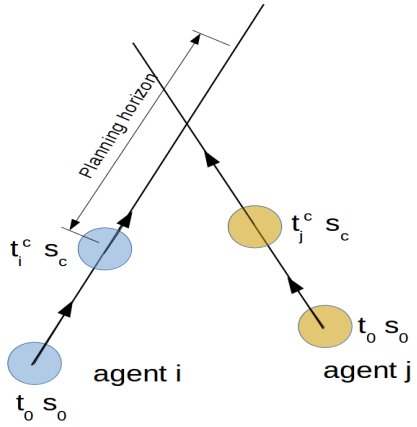


Fig. 2 An example of conflict between a pair of agents used to define TSCC constraints (4).

B. Optimization

In this section, we compute a scaling function $\dot{s}_i(s)$ valid over the path segment defined by s_0 and s_c for the i^{th} agent. Beyond s_c, the scaling function just remains constant at the value $\dot{s}_i(s_c)$. As a result of the time scaling transformation, each agent would traverse the path segment in a different time interval $[t_0 \;\; t_i^c]$ leading to collision free path coordination. To this end, we propose the following

optimization problem.

$$\arg\min_{\dot{s}_i(s)} J_2 = \sum_i (\dot{s}_i(s_c) - \dot{s}_i^{pref}), s_i^{pref} = \frac{v_i^{pref}}{\sqrt{\dot{x}_i(t_i^c)^2 + \dot{y}_i(t_i^c)^2}}. \tag{6a}$$

$$v_i^{min} \leq \sqrt{\dot{x}_i(t_i^c)^2 + \dot{y}_i(t_i^c)^2} \leq v_i^{max}. \tag{6b}$$

$$s_i(s_0) = 1, . \tag{6c}$$

$$\frac{a_i^{min}}{\sqrt{2}} \leq \ddot{x}_i(t_0) \leq \frac{a_i^{max}}{\sqrt{2}}, \frac{a_i^{min}}{\sqrt{2}} \leq \ddot{y}_i(t_0) \leq \frac{a_i^{max}}{\sqrt{2}}. \tag{6d}$$

$$A_{ij}s_i(s_c)^2 + B_{ij}s_i(s_c)s_j(s_c) + C_{ij}\dot{s}_j(s_c)^2 \leq 0, \forall i \neq j, \forall i = 1, 2, 3....n.. \tag{6e}$$

The cost function in (6a) seeks to compute such scaling functions which minimizes the deviation of the i^{th} agent from its preferred velocity profile, v_i^{pref}. Inequalities (6b) enforces the constraints that the forward velocity resulting from time scaling transformation should respect the minimum (v_i^{min}) and maximum (v_i^{max}) velocity bounds at time t_i^c or equivalently at points defined by s_c. In our formulation, v_{min} is strictly greater than zero to account for the fact that both fixed wing UAVs and commercial aircrafts have well defined minimum velocity below which it cannot maintain flight. The equality (6c) ensures that the velocity profile resulting from the time scaling transformation has continuity with the velocity at initial time instant t_0 or alternatively at the initial point defined by s_0. Note that if the scaling function $\dot{s}_i(s)$ is linear and the current velocities of the agents are within bounds, then (6b) and (6c) in conjunction ensures that the velocity bounds are satisfied at all the points between s_0 and s_c. Inequalities (6d) enforces the acceleration bounds. For computational reasons, we have split the acceleration bounds (a_i^{min}, a_i^{max}) into equivalent bounds for the separate x and y components. Inequalities (6e) is same as (5) but now written explicitly for a system of n agents.

IV. Simplification of Optimization

We start with the convexification of TSCC constraints (4) or (5) and then subsequently show that its intersection space with velocity and acceleration bounds can be described in terms of linear inequalities. Following [16], we approximate $\dot{s}_i(s), \forall s \in [s_0\ s_c]$ as a linear positive function to ensure that the transformation between s and t is monotonically increasing. Further, $\ddot{s}_i(s)$ is approximated

as constant. Further, for the ease of exposition, we introduce the change of variables (7), [16]. The third equality in (7) can also be intuitively thought as a finite difference approximation of $\ddot{s}$ over the path segment defined by s_0, s_c. In particular, note $\ddot{s} \approx \frac{\dot{s}(t_c)-\dot{s}(t_0)}{t_c-t_0} = \frac{\dot{s}(s_c)-\dot{s}(s_0)}{(s_c-s_0)\frac{2}{\dot{s}(s_c)+\dot{s}(s_0)}}$, where we have used the fact that $\dot{s}$ at time t_0, t_c will correspond to the value at the points on the path defined by s_0, s_c. Further, the time interval $t_c - t_0 = (s_c - s_0)\frac{2}{\dot{s}(s_c)+\dot{s}(s_0)}$ [16].

$$\dot{s}_i(s_0)^2 = z_i(s_0), \dot{s}_i(s_c)^2 = z_i(s_c), \ddot{s}_i(s) \approx \frac{\dot{s}_i(s_c)^2 - \dot{s}_i(s_0)^2}{2(s_c - s_0)}. \tag{7}$$

A. Convexification of TSCC Constraints

It is clear that the (5) could either be convex or non-convex depending on the sign of A_{ij}, B_{ij} and C_{ij}, which in turn would depend on the path of each agent. We now show that for a large number of relative configuration of the agents, (5) has a natural convex decomposition while for the rest, a convexification scheme could be adopted. The analysis presented below is for a pair of agents. Extension to multiple agents can be obtained by repeating the process over all the pairs.

Case 1: $A_{ij} \geq 0$, $C_{ij} \leq 0$ Let us start by defining a variable $\dot{s}_{ij} = \frac{\dot{s}_i(s_c)}{\dot{s}_j(s_c)}$, with respect to which (5) assumes the following form.

$$A_{ij}\dot{s}_{ij}^2 + B_{ij}\dot{s}_{ij} + C_{ij} \leq 0. \tag{8}$$

Since, $A_{ij} \geq 0$, (8) represents a single variable convex quadratic inequality. Thus, its solution space would be of the following form

$$\dot{s}_{ij} \in [\dot{s}_{min}\dot{s}_{max}], \dot{s}_{min} = \min\left(\frac{-B_{ij} \pm \sqrt{(B_{ij}^2 - 4A_{ij}C_{ij})}}{2A_{ij}}\right), \dot{s}_{max} = \max\left(\frac{-B_{ij} \pm \sqrt{(B_{ij}^2 - 4A_{ij}C_{ij})}}{2A_{ij}}\right). \tag{9}$$

It is straightforward to convert (9) to the original set of variables to obtain (10a). Finally, using (7), we have (10b).

$$\dot{s}_i(s_c) \geq \dot{s}_{min}\dot{s}_j(s_c), \quad \dot{s}_i(s_c) \leq \dot{s}_{max}\dot{s}_j(s_c). \tag{10a}$$

$$z_i(s_c) \geq (\dot{s}_{min})^2 z_j(s_c), \;\; z_i(s_c) \leq (\dot{s}_{max})^2 z_j(s_c). \tag{10b}$$

Case 2: $A_{ij} \leq 0$, $C_{ij} \geq 0$ The procedure for this case is similar to that described for the previous case. The small difference being that we start by defining $\dot{s}_{ji} = \frac{\dot{s}_j(s_c)}{\dot{s}_i(s_c)}$ to obtain the following reformulation of (5)

$$A_{ij} + B_{ij}\dot{s}_{ji} + C_{ij}\dot{s}_{ji}^2 \leq 0. \Rightarrow \dot{s}_i(s_c)\dot{s}_{min} \geq \dot{s}_j(s_c), \;\; \dot{s}_i(s_c)\dot{s}_{max} \leq \dot{s}_j(s_c). \tag{11}$$

Inequality (11) can eventually be reduced to the following linear inequalities.

$$(\dot{s}_{min})^2 z_i(s_c) \geq z_j(s_c), \;\; (\dot{s}_{max})^2 z_i(s_c) \leq z_j(s_c). \tag{12}$$

$$\dot{s}_{min} = \min\left(\frac{-B_{ij} \pm \sqrt{(B_{ij}^2 - 4A_{ij}C_{ij})}}{2C_{ij}}\right), \dot{s}_{max} = \max\left(\frac{-B_{ij} \pm \sqrt{(B_{ij}^2 - 4A_{ij}C_{ij})}}{2C_{ij}}\right). \tag{13}$$

Case 3: $A_{ij} \geq 0$, $C_{ij} \geq 0$ This case is superset of the previous two cases and thus, any one of the approaches described previously can be followed.

Case 4: $A_{ij} \leq 0$, $C_{ij} \leq 0$, $B_{ij} \geq 0$ This represents an instant where the constraint (5) is non-convex and thus is the most difficult. Recall (7), to represent (5) in the following form.

$$A_{ij} z_i(s_c) + B_{ij}\sqrt{z_i(s_c) z_j(s_c)} + C_{ij} z_j(s_c) \leq 0. \tag{14}$$

$$(A_{ij} - \frac{\sqrt{z_{j0}}}{\sqrt{z_{i0}}} B_{ij}) z_i(s_c) + (C_{ij} - \frac{\sqrt{z_{i0}}}{\sqrt{z_{j0}}} B_{ij}) z_j(s_c) \leq 0. \tag{15}$$

It can be seen from (14) that the first and third term is convex (in $z_i(s_c)$ and $z_j(s_c)$ respectively), while the second term is purely concave with respect to both $z_i(s_c)$ and $z_j(s_c)$. Thus, we linearize the second term around some initial guess z_{i0}, z_{j0}. This gives way to the linear inequality (15) in terms of $z_i(s_c), z_j(s_c)$. It should be noted that the solution space of (15) is contained in that of (14). In other words, (15) is a more conservative constraint than (14) [17].

B. Velocity and Acceleration Bounds

Using (2), the forward velocity for the i^{th} agent obtained through time scaling transformation can be described as $\sqrt{\dot{x}_i(t_c^i)^2 + \dot{y}_i(t_c^i)^2} = \dot{s}_i(s_c)\sqrt{x_i^{'}(s_c)^2 + y_i^{'}(s_c)^2}$. Thus, velocity bounds can be formulated using (7) in the following manner.

$$\Rightarrow (v_i^{min})^2 \leq z_i(s_c)(x_i^{'}(s_c)^2 + y_i^{'}(s_c)^2) \leq (v_i^{max})^2 \tag{16}$$

Similarly, using (2), (7), the acceleration bound constraints take the following form [16].

$$\begin{aligned} \frac{a_i^{min}}{\sqrt{2}} \leq z_i(s_0)x_i^{''}(s_0) + x_i^{'}(s_0)\frac{z_i(s_c) - z_i(s_0)}{2(s_c - s_0)} \leq \frac{a_i^{max}}{\sqrt{2}}. \\ \frac{a_i^{min}}{\sqrt{2}} \leq z(s_0)y_i^{''}(s_0) + y_i^{'}(s_0)\frac{z_i(s_c) - z_i(s_0)}{2(s_c - s_0)} \leq \frac{a_i^{max}}{\sqrt{2}}. \end{aligned} \tag{17}$$

Summary: The discussions presented in this section has shown that the TSCC constraints and the velocity and acceleration bounds can all be represented as linear inequalities in terms of variable $z_i(s_c), z_i(s_0)$. Let these inequalities be compactly represented as $\mathbf{A}_{inq}z - \mathbf{b}_{ineq} \leq 0$, where $z = (z_1 s_c, z_1(s_0), z_2(s_c), z_2(s_0)...)$. The optimization (6a)-(6c) can then be presented in the following simple form.

$$\arg\min_{z_i(s_c)} J_2 = \sum_i (z_i(s_c) - (s_i^{pref})^2)^2. \tag{18a}$$

$$\mathbf{A}_{inq}z - \mathbf{b}_{ineq} \leq 0. \tag{18b}$$

$$z_i(s_c) \geq 0. \tag{18c}$$

$$z_i(s_0) = 1. \tag{18d}$$

As can be seen, (18a)-(18d) is a convex QP problem. The variables in the optimization are $z_i(s_c)$ as a slight algebraic manipulation can remove the equalities (18d) and consequently $z_i(s_0)$. The time scaling function can be easily recovered through $\dot{s}_i(s_0) = \sqrt{z_i(s_0)}, \dot{s}_i(s_c) = \sqrt{z_i(s_c)}$.

V. Simulation Results

A. Benchmark Conflict Scenarios

In this section, we present implementation results of our proposed formulation on different benchmark scenarios. All the agents are cruising with a velocity, $v_i^{cruise} = 10m/s$ when the collisions

are detected. The velocity bounds are kept as $v_i^{min} = 5m/s, v_i^{max} = 15m/s$ while acceleration bounds were fixed at $a_i^{min} = -3m/s^2, a_i^{max} = 3m/s^2$. It is worth pointing out that existing path coordination frameworks like [2], [4] which are applied for coordination of fixed wing UAVs or commercial aircrafts assume capability of instantaneous velocity change (infinite acceleration). In contrast, we incorporate practical bounds on both velocities and accelerations in our simulations. Simulation videos can be found in `https://www.youtube.com/watch?v=Of6LMnV5CsE`. The coordination framework essentially consists of solving optimization (18a)-(18d) in a receding horizon manner wherein at each iteration, after obtaining the scaling functions, the forward velocities and the positions of the agents along their paths are updated. Consequently, at next iteration, these updated positions and velocities are taken as current boundary values and the optimization is again solved. The process continues till all the agents have reached their goal position.

Agents along the circumference of a circle: The conflict scenario shown in Fig.3(a) is similar to that addressed in the MIP based path-coordination framework [1]. Here, 20 agents modeled as circular disks of radius $4.5m$ are placed along the circumference of a circle of radius $313m$. Their straight line paths create a common collision zone around the center of the circle. Note that in contrast to [1], our implementation, imposes the constraint that $v_i^{min} > 0$. That is, the forward velocity is strictly positive and consequently the agents cannot stop in place to avoid collisions. We believe that this condition leads to a more difficult coordination problem.

Agents along the circumference of a semi-circle: Fig. 3(d) show 20 agents placed along the circumference of a semi-circle with a common collision zone around the center of the circle. The radius of the circle as well as the circular disks representing the agents are same as the previous conflict scenario.

Rectangular Grid: Fig.3(e) show the conflict scenario with two groups of 10 agents each. The straight line paths of the agents create a rectangular grid like structure with multiple collision zones.

Conflict Along Curved Paths: This conflict scenario shown in Fig.3(f) is similar to the semi-circle conflict discussed above. However, now the agents move along curved paths instead of straight lines. For this case, we resort to piece-wise straight line approximation which are constantly refined within our receding horizon framework. Further, additional constraints on turn rate was also incorporated

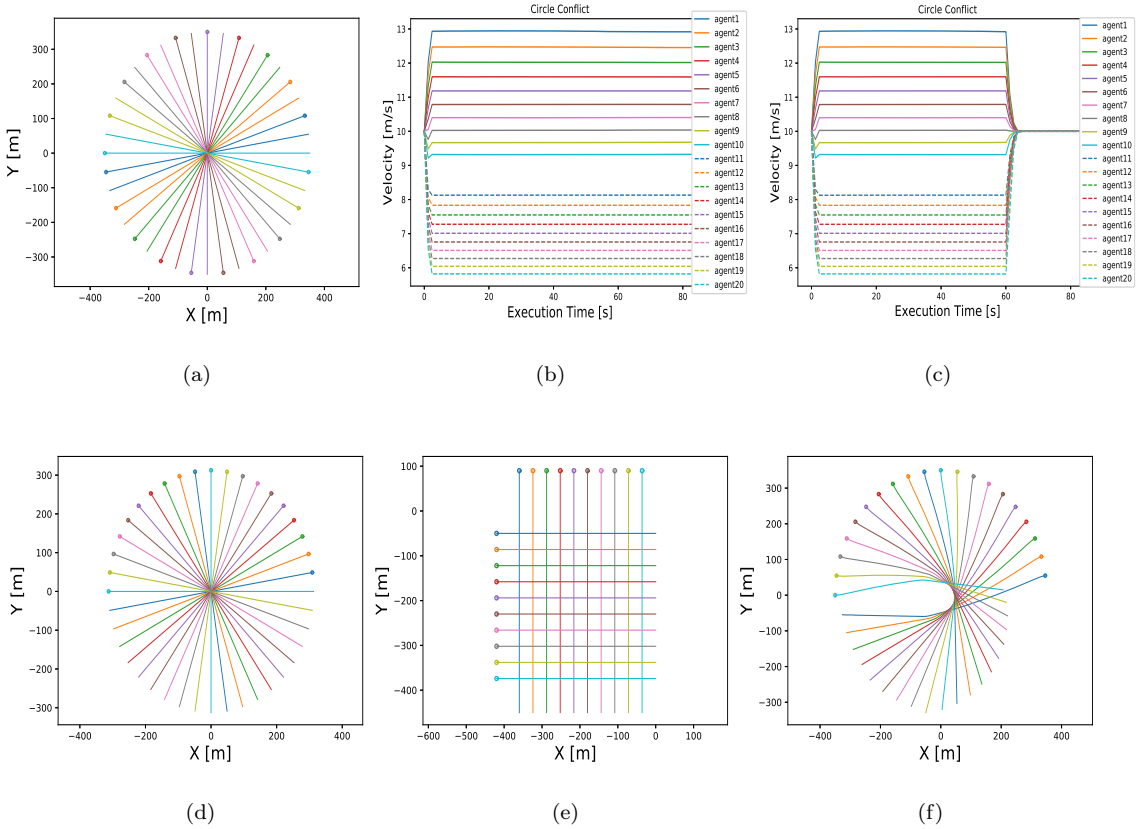


Fig. 3 Conflict scenarios considered and obtained velocity profiles.

which could also be converted to linear inequality in (18a)-(18d).

B. Velocity Profiles

Due to space constraints, we present velocity plots for only one conflict scenario.

The velocity profiles of the agents during collision avoidance depends on the choice of v_i^{pref}. That shown in Fig.3(b) are obtained with $v_i^{pref} = v_i^{current}$, where, $v_i^{current}$ represents the current velocity of the i^{th} agent. Similar choice of v_i^{pref} has also been reported in [18]. We also experimented with a different choice of v_i^{pref}. To this end, consider the following.

$$v_i^{pref} = v_i^{current}, f_{ij}^{coll}(.) > 0, v_i^{pref} = v_i^{cruise}, f_{ij}^{coll}(.) \leq 0 \tag{19}$$

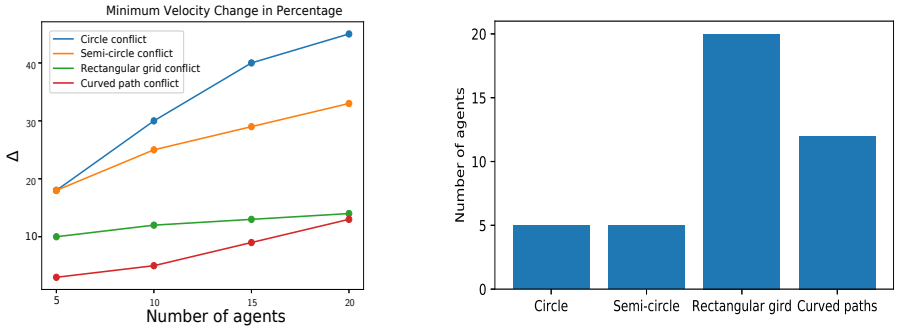
Where, v_i^{cruise} is some preferred cruise velocity for the agents and $f_{ij}^{coll}(.)$ is a function which detects the collision state of a pair of agents based on (1). Thus, a value greater than 0 implies the agents are on a collision course, while a value less than or equal to 0 implies the converse. Essentially, the

above cost function checks whether all agents have been out of collision zone for last k iterations of receding horizon formulation and if so, allows them to go back to their cruise velocities. In practical applications like air-traffic conflict resolution, the cruise velocity is designed based on metrics like fuel efficiency and thus, it is advantageous for the aircrafts to come back to their cruise velocities after coming out of the collision zone. Velocity profiles shown in Fig.3(c) is obtained with $v_i^{cruise} = 10m/s$.

C. Velocity Requirements for Path Coordination.

Let the velocity bounds for the agents be described as $[v_i^{cruise} - \Delta,\ v_i^{cruise} + \Delta]$. In this section, we empirically evaluate the minimum Δ required for collision free coordination in benchmark conflict scenarios presented in Fig.3. To this end, consider Fig.4(a) which presents Δ as a percentage of $v_i^{cruise} = 10m/s$ for varying number of agents. As can be seen, for difficult circle and semi-circle conflict scenarios, the requirement is around 45%. However for a more realistic scenario like rectangular grid, the minimum Δ required is significantly lower, around 13% for 20 agents. Moreover, the increase in Δ with the number of agents is also small. This is because, in this conflict scenario not all agents are simultaneously in conflict with each other. For the conflict scenario along the curved paths, the geometry of the paths actually assist in collision avoidance and thus here the minimum Δ required is the least.

The Δ requirements presented above are moderate and easily achievable for autonomous agents like commercially available UAVs. However, in applications like air traffic conflict resolution the flexibility in change of magnitude of forward velocity is limited due to operational constraints [19]. According to [19], a commercial passenger aircraft's forward velocity is recommended to be reduced and increased by maximum of 6% and 3% respectively. With these bounds, we evaluated the performance of our proposed formulation on the benchmark conflict scenarios presented in figure 3. The results are summarized in Fig.4(b).



(a) (b)

Fig. 4 (a):Forward velocity change requirement in percentage for different conflict scenarios. (b): Maximum number of agents that can be de-conflicted for the velocity change bounds of -6% **and** $+3\%$

VI. Comparisons with the State of the Art

A. Review of Convex Path Coordination of [7]

Let, $\mathbf{x}_i(s)$ be the path of the i^{th} agent. Let, $t_i(s_k)$ represent the time at which the i^{th} agent is at position $\mathbf{x}_i(s_k)$. Similarly, let $v_i(s_k), v_j(s_k)$, and $a_i(s_k), a_j(s_k)$ respectively represent the forward velocities and accelerations of the agents at the same point on the path. We formulate the following optimization problem for the path coordination based on [7], [20].

$$\sum_{i=1}^{i=n} \sum_{s_k=s_0}^{s_k=s_f} a_i(s_k)^2. \tag{20a}$$

$$\mathbf{p}_i(s_{k+1}) = \mathbf{A}\mathbf{p}_i(s_k) + \mathbf{b}a_i(s_k). \forall i \tag{20b}$$

$$t_j(s^{entry}) \geq t_i(s^{exit}). \forall i, \forall j = i+1, i+2.. \tag{20c}$$

$$v_i(s_k) \leq v_i^{max}, \forall i, \forall s_k \in [s_0 \;\; s_f]. \tag{20d}$$

$$a_i(s_k) \leq a_i^{max}(s_k), a_i(s_k) \geq a_i^{min}(s_k) \forall i, \forall s_k \in [s_0 \;\; s_f]. \tag{20e}$$

$$\mathbf{p}_i(s_k) = \begin{bmatrix} t_i(s_k) \\ v_i(s_k) \end{bmatrix}), \; \mathbf{A} = \begin{bmatrix} 1 & d_s \\ 0 & 1 \end{bmatrix}), \; \mathbf{b} = \begin{bmatrix} 0 \\ d_s \end{bmatrix}),$$

The cost function, (20a) penalizes acceleration inputs and encourages the agents to continue moving with their current velocities. This cost function is slightly different from the original formulation of [7], [20] but has been brought here for fair comparison with our proposed formulation. The equality

(20b) models the evolution of the agents along the given path. In particular, at what time, a given agent occupies a particular position on the path. Here, d_s is the resolution of discretization which can be thought as an arc length between positions $\mathbf{x}_i(s_k)$ and $\mathbf{x}_i(s_{k+1})$. The inequality (20c) models the collision avoidance constraints, where $s^{entry} \in [s_0 \;\; s_f]$ and $s^{exit} \in [s_0 \;\; s_f]$ denote the start and end of the collision zone. Inequality (20c) essentially forces the agent j to enter the collision zone only when the agent i has exited it. The inequalities (20d) and (20e) are typical velocity and acceleration bound constraints. Please note that $a_i^{max}(s_k)$, $a_i^{min}(s_k)$ are related but at the same time slightly different from the acceleration bounds presented in (6d). For details refer [7](and equation (16) therein).

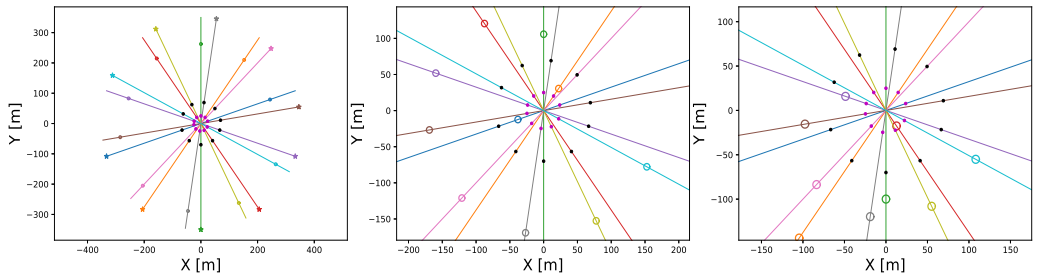
Planning Horizon: The computational complexity of the optimization (20a)-(20e) depends on the length of the planning horizon and the resolution of discretization d_s as the number of variables in the optimization depends on the product of these two parameters. For both these parameters, we experimented with many values till we found the smallest value for which the optimization (20a)-(20e) resulted in a feasible solution.

Table 1 Summary of Computation Times in seconds

Conflict Scenario	Murgovski et al .	Each iteration (Proposed)	Total time (Proposed)
Circle: 5 agents	0.54	0.004	0.24
Circle: 10 agents	2.87	0.011	0.52
Circle: 15 agents	39.18	0.02	1.4
Semi-Circle: 5 agents	0.14	0.0037	0.20
Semi-Circle: 10 agents	3.03	0.009	0.37
Semi-Circle: 15 agents	32.43	0.018	0.765

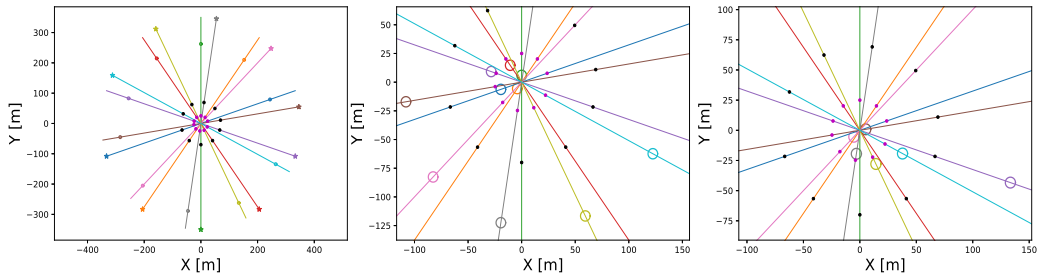
B. Comparative Results

We compare our proposed formulations with the convex path coordination of [7] in terms of (i) computational time and (ii) quality of coordination maneuver obtained. We use the conflict scenarios presented in Fig.3(a), 3(d) as test cases as they represent a difficult benchmark case while at the



(a) (b) (c)

Fig. 5 Snapshots of collision avoidance behavior obtained through the path coordination approach of [7].



(a) (b) (c)

Fig. 6 Snapshots of collision avoidance behavior obtained through the proposed formulations.

same consists of only a single collision zone. Moreover, these conflict scenarios require consideration of collision avoidance among all the agents even while considering a smaller path horizon.

Computational Time: Table 1 contrasts the computational time for our proposed formulation with the path coordination approach of [7] for a given planning horizon. Since, our formulation incrementally builds the path coordination plans by repeatedly solving optimization (18a)-(18d) in a receding horizon manner, we present both the computational time for each iteration as well as that required to plan for the entire considered horizon. All computations were performed on a laptop with 12GB RAM and 2.5GHz processor. Python based framework CVXOPT [21] was used to solve the optimizations.

As can be seen, our formulation is significantly faster and the difference increases sharply as the number of agents increase. From the table 1, it is also clear that the approach of [7] can be adapted for reactive setting for only small number of agents (typically around 5). It should be noted from

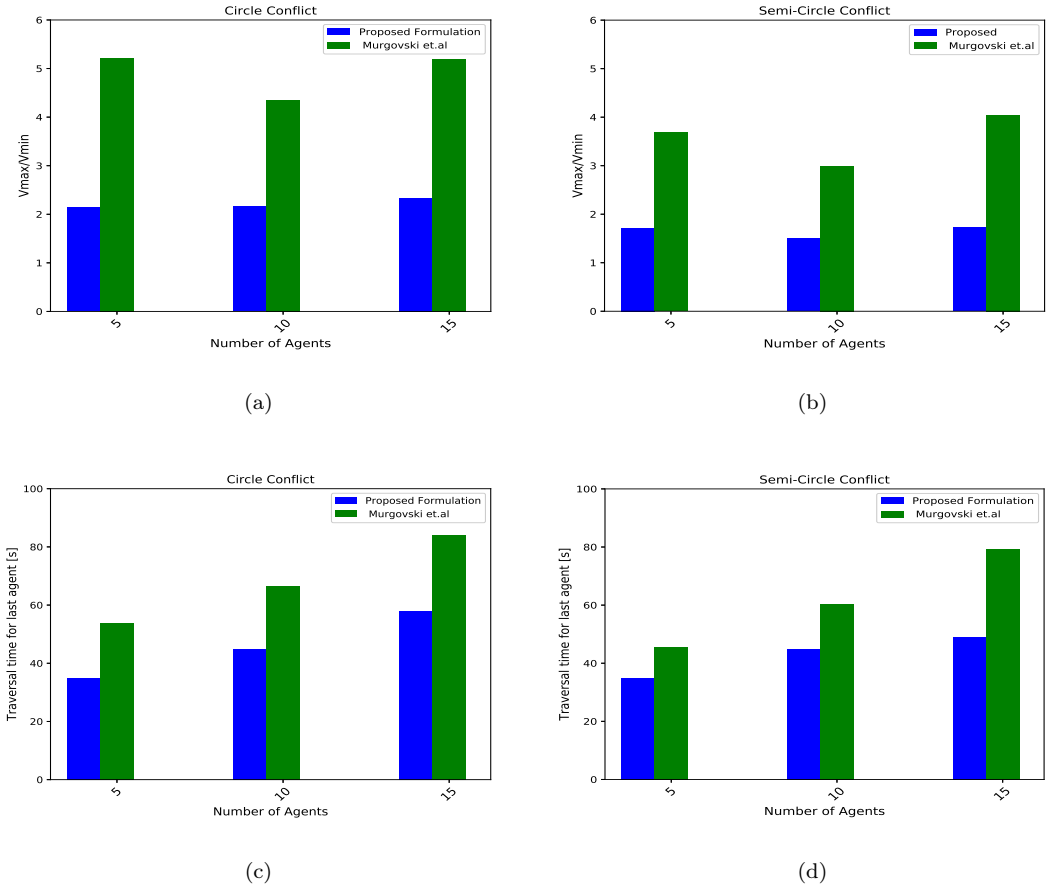


Fig. 7 Figures (a) and (b) presents a comparison of $\frac{\max(v_1,v_2,v_3\ldots v_n)}{\min(v_1,v_2,v_3\ldots v_n)}$. Figures (c) and (d) presents traversal time for the last agent.

table 1 that each iteration of the proposed receding horizon formulation can be solved in few tens of milliseconds. Thus, the agents can adapt their motions on the fly. In fact the receding horizon formulation can act as an implicit feedback for the agents.

Quality of Coordination Maneuver: Fig.5(a)-5(c) provides snapshots of coordination obtained following the approach of [7] while Fig.6(a)-6(c) provides the same obtained from our proposed formulation. All the agents are cruising with a velocity of $10m/s$ when they detect mutual conflicts with each other. The following key contrasts can be easily observed. Firstly, note Fig.5(a)-5(c) where the start and end of the collision zone is represented by the points shown in magenta and black respectively. The approach of [7] requires only one agent to be present in the collision zone at any given instant (recall inequality (20c)). This is a highly conservative requirement which induces unnecessarily high clearance between the agents (Fig.5(b), 5(c)). In contrast, our proposed

formulation allows agents to come close to each other provided they have collision avoiding velocities. As a result more than one agent can simultaneously reside in the collision zone at any given instant (Fig. 6(b), 6(c)).

Secondly, the conservativeness of [7] puts a higher demand for change in magnitude of forward velocities to achieve collision avoidance. To understand this further, consider Fig.7(a)-7(b) which quantifies $\frac{\max(v_1,v_2,v_3...v_n)}{\min(v_1,v_2,v_3...v_n)}$, that, is the ratio of maximum and minimum velocity required for achieving collision avoidance. As can be seen, our proposed formulation achieves collision free path coordination with a much smaller ratio than the approach presented in [7]. This in turn signifies that the agents incur much smaller deviation from their cruise velocities. As mentioned earlier, in many applications, departure from the cruise velocities incurs a physical cost. For example, an air traffic conflict resolution, this may impact fuel efficiency. Even for fixed wing UAVs, there is generally a cruise velocity which results in least drag and maximum efficiency.

Finally, our proposed formulation results in path coordination with smaller traversal times which can be quantified as the time taken by the last agent to exit the collision zone. Fig.7(c) and 7(d) quantifies the traversal times obtained with our proposed formulation as well as that obtained with the approach presented in [7].

VII. Conclusions

Frameworks for path coordination can be extremely useful for many applications involving deployment of multiple agents. However, the computational nature of the existing frameworks essentially restricts them to an off-line motion planning methodology and have limited their utility in real time applications wherein motion plans needs to be computed on the fly. In this paper, we have proposed a distinctively different approach based on a novel concept called time scaled collision cone (TSCC). As shown, our proposed formulation essentially boils down to solving a convex QP problem in a receding horizon manner. The computational time required to solve each instance of the QP was found to vary linearly with number of agents, n (refer table 1). For 20 agents, the time required was around 40 ms on a laptop with 12GB RAM and 2.5GHz processor. We have extensively compared our formulation with the current state of the art convex path coordination

approach of [7] and shown that our formulation is not only faster but also provides less conservative solutions. The proposed centralized formulation provides one key advantage-it allows for search in joint velocity space of all the agents. Thus, collision free coordination in some conflict scenarios was obtained with as less as 10% change in cruise velocity of the agents.

References

[1] Peng, J. and Akella, S., "Coordinating multiple robots with kinodynamic constraints along specified paths," *The International Journal of Robotics Research*, Vol. 24, No. 4, 2005, pp. 295–310.

[2] Gonçalves, V. M., Pimenta, L. C., Maia, C. A., and Pereira, G. A., "Coordination of multiple fixed-wing UAVs traversing intersecting periodic paths," in "Robotics and Automation (ICRA), 2013 IEEE International Conference on," IEEE, 2013, pp. 849–854.

[3] Altché, F., Qian, X., and de La Fortelle, A., "Time-optimal coordination of mobile robots along specified paths," in "Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on," IEEE, 2016, pp. 5020–5026.

[4] Cafieri, S. and Durand, N., "Aircraft deconfliction with speed regulation: new models from mixed-integer optimization," *Journal of Global Optimization*, Vol. 58, No. 4, 2014, pp. 613–629.

[5] Alonso-Ayuso, A., Escudero, L. F., and Martín-Campo, F. J., "On modeling the air traffic control coordination in the collision avoidance problem by mixed integer linear optimization," *Annals of Operations Research*, Vol. 222, No. 1, 2014, pp. 89–105.

[6] Abichandani, P., Benson, H. Y., and Kam, M., "Multi-vehicle path coordination under communication constraints," in "American Control Conference, 2008," IEEE, 2008, pp. 650–656.

[7] Murgovski, N., de Campos, G. R., and Sjöberg, J., "Convex modeling of conflict resolution at traffic intersections," in "Decision and Control (CDC), 2015 IEEE 54th Annual Conference on," IEEE, 2015, pp. 4708–4713.

[8] Švestka, P. and Overmars, M. H., "Coordinated path planning for multiple robots," *Robotics and autonomous systems*, Vol. 23, No. 3, 1998, pp. 125–152.

[9] Siméon, T., Leroy, S., and Laumond, J.-P., "Path coordination for multiple mobile robots: a resolution-complete algorithm," *Robotics and Automation, IEEE Transactions on*, Vol. 18, No. 1, 2002, pp. 42–49.

[10] Chakravarthy, A. and Ghose, D., "Obstacle avoidance in a dynamic environment: A collision cone approach," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 28, No. 5, 1998, pp. 562–574.

[11] Singh, A. K. and Krishna, K. M., "Reactive collision avoidance for multiple robots by non linear time scaling," in "Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on," IEEE, 2013, pp. 952–958.

[12] Gopalakrishnan, B., Singh, A. K., and Krishna, K. M., "Time scaled collision cone based trajectory optimization approach for reactive planning in dynamic environments," in "Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on," IEEE, 2014, pp. 4169–4176.

[13] Fiorini, P. and Shiller, Z., "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, Vol. 17, No. 7, 1998, pp. 760–772.

[14] Snape, J., Van den Berg, J., Guy, S. J., and Manocha, D., "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, Vol. 27, No. 4, 2011, pp. 696–706.

[15] García, S., Slawinski, E., Mut, V., and Penizzotto, F., "Collision avoidance method for multi-operator multi-robot teleoperation system," *Robotica*, pp. 1–18.

[16] Hauser, K., "Fast interpolation and time-optimization with contact," *The International Journal of Robotics Research*, Vol. 33, No. 9, 2014, pp. 1231–1250.

[17] Boyd, S., "Sequential convex programming," *Lecture Notes, Stanford University.*

[18] Van Den Berg, J., Guy, S. J., Lin, M., and Manocha, D., "Reciprocal n-body collision avoidance," in "Robotics research," Springer, pp. 3–19, 2011.

[19] Bonini, D., Dupré, C., and Granger, G., "How ERASMUS can support an increase in capacity in 2020," in "Proceedings of the 7th International Conference on Computing, Communications and Control Technologies: CCCT," , 2009.

[20] Riegger, L., Carlander, M., Lidander, N., Murgovski, N., and Sjöberg, J., "Centralized MPC for autonomous intersection crossing," in "Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on," IEEE, 2016, pp. 1372–1377.

[21] Andersen, M. S., Dahl, J., and Vandenberghe, L., "CVXOPT: A Python package for convex optimization, version 1.1. 6," *Available at cvxopt. org*, Vol. 54.