Time Scaled Collision Cone based Trajectory Optimization Approach for Reactive Planning in Dynamic Environments.

Bharath Gopalakrishnan[1], Arun Kumar Singh [1], K.Madhava Krishna[1]

***Abstract*— The current paper proposes a trajectory optimization approach for navigating a non-holonomic wheeled mobile robot in dynamic environments. The dynamic obstacle's motion is not known and hence is represented by a band of predicted trajectories . The trajectory optimization can account for large number of predicted obstacle trajectories and seeks to avoid each predicted trajectory of every obstacle in the sensing range of the robot. The two primary contributions of the proposed trajectory optimization are (1): A computationally efficient method for computing the intersection space of collision avoidance constraints of large number of predicted obstacle trajectories. (2): A optimization framework to connect the current state to the solution space in time optimal fashion.**

The intersection/solution space computation is build on our earlier proposed concept of *time scaled collision cone*, which can be solved in closed form to obtain a set of formulae. These formulae describe how much and in what manner the temporal specification of a trajectory needs to be changed to avoid a given set of dynamic obstacles. This allows us to quickly evaluate solution space of time scaled collision cone over various candidate trajectories, thus reducing the problem of computing the intersection space to that of generating multiple homotopic trajectories. The optimization framework used to connect the current state to the solution space in time optimal fashion is based on the concept of non-linear time scaling, which induces a *difference of convex* form structure. Thus, on the theoretical side, we show that the various components of the proposed framework are computationally simple and involves solving sets of linear equations and using state of the art convex programming techniques. On the practical side we show that the proposed planner performs better than sampling based planners which treat dynamic obstacles as static over a short duration of time

I. Introduction

Trajectory optimization approaches for motion planning has gathered tremendous interests in recent years and are seen as an alternative to sampling based planners [12], [7]. The attractive aspect of trajectory optimization is that it gives an one shot solution characterizing path and motion profile along it while satisfying constraints on states and control and optimizing a cost function. However the relative ease with which trajectory optimization can be solved depends on the nature of robot's dynamics and that of constraints. The problem of navigating a non-holonomic robot among dynamic obstacles is associated with constraints which are highly non-linear and non-convex. The problem becomes further challenging when the obstacles' motion is not known and hence multiple predicted trajectories for each obstacle have to be incorporated [9], resulting in large number of constraints. To realise the full potential of trajectory optimization on such problems, we propose an approach for efficiently solving the collision avoidance constraints.

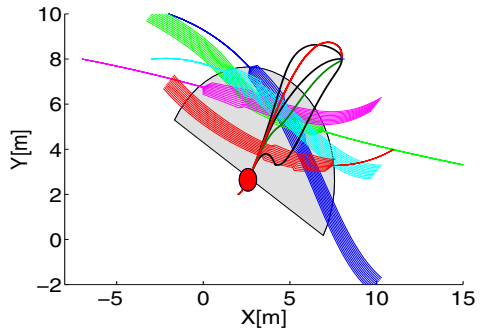

Fig. 1. Summary of the methodology proposed in this paper. The obstacles' motion is represented as band of predicted trajectories. A trajectory optimization approach is proposed which relies on efficient computation of intersection space of collision avoidance constraints resulting from large number of predicted obstacle trajectories. We build on the concept of *time scaled collision cone* which can be solved in closed form to obtain formulae which describes collision avoidance manoeuvre along a trajectory. Hence it is possible to quickly evaluate *time scaled collision cone* over various candidate trajectories. In the figure the original trajectory of the robot is shown in red and the various candidate trajectories are shown in black. The best solution space and hence the best trajectory can be chosen to minimize a cost function. In the above figure best trajectory is shown in green. Finally an optimization framework is proposed which connects the current state to the solution space in time optimal fashion.

At the crux of the proposed methodology is our earlier proposed concept of *time scaled collision cone* [13], which as the name suggests is a time scaling based reformulation of the concept of collision cone [4]. *Time scaled collision cone* provides a set of collision avoidance constraints along a given trajectory for a given set of dynamic obstacles. In other words, it describes how to avoid collisions by just increasing/decreasing the speed i.e changing the temporal specification of the trajectory. Important aspect however is the fact these collision avoidance constraints can be solved in closed form to obtain a set of formulae. The symbolic form of solution space allows us to quickly evaluate it over various homotopic candidate trajectories. Thus the complex problem of computing the intersection space of non-linear collision avoidance constraints is converted to a problem of generating multiple candidate trajectories and evaluating the symbolic formulae depicting the solution space over it. As shown in section IV, for a non-holonomic robot, generating multiple candidate trajectories amounts to solving a set of linear equations. Moreover we provide a systematic method

[1]The authors are with Robotics Research Center, IIIT-Hyderabad India, bharath.gopalakrishnan@research.iiit.ac.in, arunkumar.singh@research.iiit.ac.in, mkrishna@iiit.ac.in

of generating these candidate trajectories. We propose a simple cost function which evaluates all the solution spaces along various candidate trajectories and chooses one resultant based on collision avoidance scenario. To the best of author's knowledge such characterization of solution spaces has not been reported in literature and we believe it can be pivotal in implementing efficient reactive navigation methodologies on robots with low computational power and memory.

Finally, we propose an optimization framework which allows the robot to reach the solution space of collision avoidance in time optimal fashion, subject to given bounds on velocity and acceleration. We leverage our previously proposed concept of *non-linear time scaling* [14] to induce a simplified *difference of convex form* [3] structure in the optimization framework.

II. Related Work

Many dedicated approaches like [16], [1] for reactive navigation among dynamic obstacles has been proposed based on the concept of collision cone [4] and velocity obstacle [5]. As stated in [10], these purely reactive navigation schemes may sometimes show suboptimal behaviour. One possible line of improvement is to combine reactive navigation concepts based on velocity obstacles/collision cone with trajectory optimization techniques. The current work accomplishes precisely that by using the mathematical relations provided by concept of collision cone as constraints in the trajectory optimization.

Motion planning in dynamic environments has also been addressed in the purview of sampling based planners. Some approaches like [6] plan by treating dynamic obstacles as static over a short time horizon and sample in the (x, y, θ) configuration space. In contrast works like [9], [11] and [17] are more rigorous and include the time dimension by searching in (x, y, θ, t) space. The current proposed work also includes a search but not in the space of discrete configuration space parameters but in the space of trajectories. As a result the final trajectory is guaranteed to be kinematically feasible. Moreover as shown later, the concept of computing solution space along candidate trajectories allows for efficient exploitation of time dimension for collision avoidance.

The propose work is more closely related to [10]. The collision avoidance constraints in [10] are included as penalty term in the objective function. Thus avoidance is not ensured if the optimization runs into local minima. In contrast the characterization of solution space described in the previous section allows us to consider collision avoidance as hard constraints. Moreover [10] do not include time optimality as they consider trajectories of fixed duration.

A. Layout of the Paper

The rest of the paper is organised as follows. Section III describes how uncertainty in an obstacle's motion is modelled. Section IV introduces concept of time scaled collision cone and derives symbolic expressions describing it's solution space. The section also elucidates the relative ease with which the *time scaled collision cone* can be evaluated over multiple candidate trajectories. Section V describes how the temporal specification of a particular chosen candidate trajectory is modified in accordance with the solution space of collision cone. Section VI presents simulation results

III. Modelling Obstacle's Motion

Real life scenarios have uncertainty associated with obstacle's motion and we include this information in the planning framework by representing the obstacle's motion with a band of most probable trajectories. To derive this band we assume a motion model for the obstacle and used the measured velocity information in the EKF prediction step to evolve it's motion. Similar approaches has been reported in [9]. An example of resulting trajectories is shown in 2. The mean trajectory is shown as black dotted line. The most probable trajectories as shown in 2 are sampled from the covariance matrix, but within a bounded uncertainty.

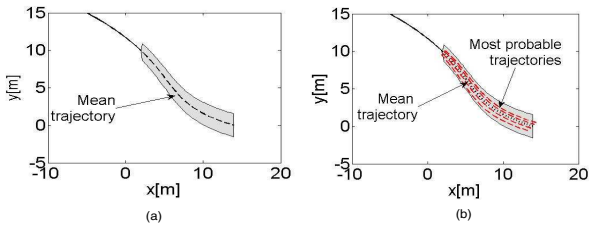

Fig. 2. The first figure(from the left),shows an obstacle path with bounded uncertainty (grey region),the second figure shows the samples (in red) at different areas in the obstacle's uncertainty region, each sample is an unique obstacle.The sampled paths (in red) which are between $\pm 2\sigma$ from the mean(dotted black lines) are shown in the third figure.

IV. Time Scaling based Collision Avoidance

Time scaling involves changing the current time scale, t to new time scale, τ in the trajectory definition, $\mathbf{X}(t) = (x(t), y(t))^T$. Such transformations does not change the path of the robot, but brings the following changes in the velocity and acceleration profile of the trajectory.

$$\dot{\mathbf{X}}(\tau) = \dot{\mathbf{X}}(t)\frac{dt}{d\tau}, \ddot{\mathbf{X}}(\tau) = \ddot{\mathbf{X}}(t)(\frac{dt}{d\tau})^2 + \dot{\mathbf{X}}(t)\frac{d^2t}{d\tau^2} \quad (1)$$

$\frac{dt}{d\tau}$ is called the scaling function and decides the transformation between the time scales. The objective is to compute such scaling function which leads to collision avoidance along a given trajectory. Since collision avoidance can be understood as reaching a particular velocity at a particular instant of time, the objective reduces to computing the range of values that the scaling function can take at a given instant.

To this end, let t_0 denote the time when the robot detects collision with N dynamic obstacles and t_c denote the minimum time in which the distance between any of the obstacles and the robot is expected to be less than some safety threshold based on predicted obstacle trajectories. The state of the i^{th} obstacle at time t_c will be denoted as $(x_i, y_i, \dot{x_i}, \dot{y_i})$. The state of the robot at time t_c based on it's current trajectory will be $(x, y, \dot{x}, \dot{y})$. By denoting $\frac{dt}{d\tau}(t_c) = s$, the set of velocities that the robot can reach by scaling transformation at time t_c can be denoted as $(s\dot{x}, s\dot{y})$.

The solution space of s can be obtained by computing the intersection space of following N *time scaled collision cone* constraints. Derivations leading to the following constraint can be found in [13].

$$(x-x_i)^2+(y-y_i)^2-R_i^2 - \frac{((s\dot{x}-\dot{x}_i)(x-x_i)+(s\dot{y}-\dot{y}_i)(y-y_i))^2}{(s\dot{x}-\dot{x}_i)^2+(s\dot{y}-\dot{y}_i)^2} \geq 0 \quad \forall i \in 1,2,...N \tag{2}$$

(2) represents N inequalities in the form $a_i s^2+b_i s+c_i > 0$ and general form of the solution space will depend on which of the following three cases arise.

• Case 1. $a_i > 0 \ \forall i = 1,2,3...N$

The solution space of s in this case will be of the form

$$s \in [0, \min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}] \cup [\max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}, \infty) \tag{3}$$

If $\min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2c_i}\} < 0$, the solution space is of the form

$$[\max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}, \infty) \tag{4}$$

No feasible solution exists if $\max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\} < 0$. In this case solution space will only comprise of negative values of s.

• Case 2. $a_i < 0 \ \forall i = 1,2,3...N$

The solution space is of the form

$$[\max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}, \min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}] \tag{5}$$

If $\max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\} < 0$, the solution space is of the form

$$[0, \min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\}] \tag{6}$$

No feasible solution exists in this case if $\min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\} < 0$

• Case 3. $a_i > 0$ for M constraints and $a_i < 0$ for $N-M$ constraints.

For clarity let coefficients arising out of M constraints be indexed with i while that arising from $N-M$ constraints be indexed with j. Then the solution space of s will in the following form

$$s \in [s_{jmax}, s_{imin}] \cup [s_{imax}, s_{jmin}], \tag{7}$$

where

$$s_{jmax} = \max\{-b_j + \frac{\sqrt{b_j^2-4a_jc_j}}{2a_j}\} \tag{8}$$

$$s_{imin} = \min\{-b_i - \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\} \tag{9}$$

$$s_{jmin} = \min\{-b_j - \frac{\sqrt{b_j^2-4a_jc_j}}{2a_j}\} \tag{10}$$

$$s_{imax} = \max\{-b_i + \frac{\sqrt{b_i^2-4a_ic_i}}{2a_i}\} \tag{11}$$

Solution space will be null if

$$s_{jmax} > s_{imin}, \text{and}, s_{imax} > s_{jmin} \tag{12}$$

The solution space of s given by equations (3)-(7) determines the solution space of collision avoiding velocities reachable through time scaling. We next describe the analogous case of avoiding collisions with a band of predicted obstacle trajectories.

A. Collision Avoidance with band of Trajectories.

We assume that the uncertainty around a particular dynamic obstacle can be captured by representing it's motion by a band of predicted trajectories. We build on the formulations described above to compute the solution space of velocities which avoids multiple band of trajectories. Let the robot at any particular instant detect collision with N obstacles and let each band consists of L predicted trajectories for describing obstacle's motion. Hence computing the solution space of collision avoiding velocities would involve finding the intersection of $L.N$ quadratic inequalities of the form of (2). The solution space would be given by either of (3) ,(5) or (7) depending on which particular case arise. The computations can be further simplified by noting that the sign of a_i usually remains the same for every trajectory in a particular band. Hence it is easier to first compute the intersection of inequality (2) over each band separately and then compute the overall intersections with multiple bands.

As an example consider figure 3(a) and 3(c) which shows the robot simultaneously detecting collisions with 3 obstacles. Each dynamic obstacle's motion is represented by a set of 11 probable trajectories. Hence effectively, the robot is required to compute the velocities for avoiding 33 dynamic obstacles. The solution space for each band and the overall intersection of solution space over multiple bands, along the current robot trajectory are shown in figure 3(b) and 3(d). In figure 3(b) the solution space for first band is in the form of (5), while the solution for the second and third band is in the form of (3). Hence the resultant solution space is in the form given by (7). Similarly for the obstacle configuration shown in 3(c), the solution space of each band and the resultant solution space is in the form given by (3).

The relative ease with which the solution space for multiple band of obstacle trajectories can be obtained allows us to quickly evaluate the solution space of s over multiple candidate trajectories and choose one which minimizes a cost metric. This is described in the next section.

B. Generating Candidate trajectories and Evaluating Time Scaled Collision Cone over it

Figure 4(a) and 4(b) shows various candidate trajectories generated around the original trajectory (shown in red). To understand how these candidate trajectories are generated, consider the following unicycle model of a non-holonomic robot.

$$\dot{x} = v\cos\theta, \dot{y} = v\sin\theta, u_1 = \dot{v}, u_2 = \ddot{\theta} \tag{13}$$

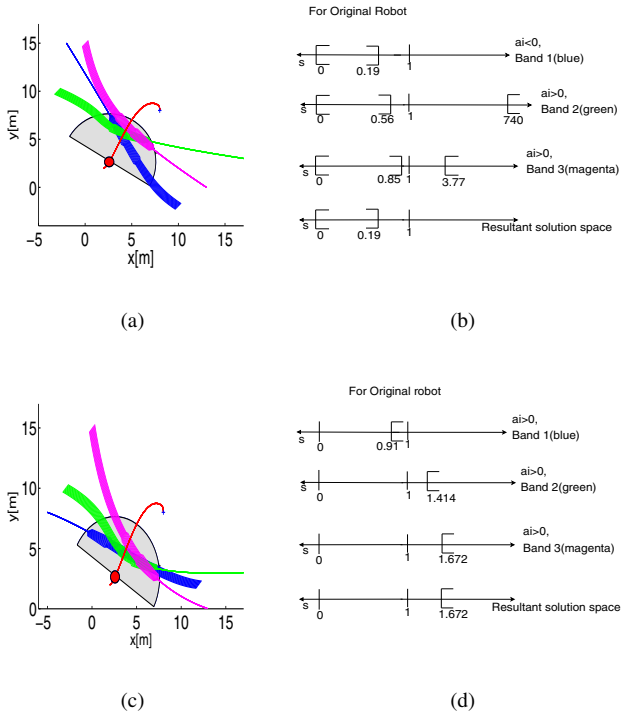

Fig. 3. (a) and (c) shows scenarios where three uncertain dynamic obstacles are simultaneously detected by the robot. Each obstacle's motion is described by 11 probable trajectories. Hence effectively in both the cases, the robot needs to compute velocities for avoiding 33 dynamic obstacles. The solution space of s along the current robot trajectory are shown in (b) and (d).

where $v = \sqrt{\dot{x}^2 + \dot{y}^2}$ represents the linear velocity of the robot from the robot's local reference frame aligned with the longitudinal axis. θ represents the heading of the robot. u_1 and u_2 are the control inputs. System represented by (13) is differentially flat [15] which means that a subset of the states can be chosen as flat outputs and all other states and control variable can be expressed as an algebraic function of the flat outputs. We choose here x and θ as flat outputs and parametrize them as the following time dependent polynomial functions.

$$x(\alpha_k, t) = \sum_{k=0}^{r} B_k^r(t)\alpha_k, \tan\theta(t) = \zeta(\beta_k, t) = \sum_{k=0}^{r} B_k^r(t)\beta_k \quad (14)$$

$$B_k^r = \binom{r}{k}(1 - \frac{t - t_0}{t_f - t_0})^{r-k}(\frac{t - t_0}{t_f - t_0})^k, \forall t \in [t_0, t_f]$$

By (13) we have

$$\dot{y} = \dot{x}\zeta, \Rightarrow y(\alpha_k, \beta_k, t) = \sum_{k=0}^{r} f_k(\alpha_k, t)\beta_k \quad (15)$$

f_i are functions of only parameters α_k and t. The coefficients α_k, β_k can be obtained by solving the following sets of linear equations.

$$\begin{cases} x(t_0) = x_0, x(t_f) = x_f, \dot{x}(t_0) = \dot{x_0}, \dot{x}(t_f) = \dot{x}_f \\ \zeta(t_0) = \zeta_0, \zeta(t_f) = \zeta_f, \dot{\zeta}(t_0) = \dot{\zeta_0}, \dot{\zeta}(t_f) = \dot{\zeta}_f \\ y(t_0) = y_0, y(t_f) = y_f \\ x(t_c) = x_c, y(t_c) = y_c \end{cases} \quad (16)$$

The first three equalities in (16) represents the initial and final boundary conditions while the last equality is used to control the topology of the paths by varying the intermediate point x_c, y_c. For specifying x_c, y_c, we choose a point, x_{co}, y_{co} at time t_c on the original trajectory and then gradually perturb it. The various magnitude of perturbations results in different candidate trajectories shown in figure 4(a) and 4(b).

The *time scaled collision cone* is solved along candidate trajectories to obtain the solution space of s. The variation in the solution space itself dictates the direction of perturbation for the candidate trajectories. For example in figure 4(a), first the candidate trajectories 1 and 2 are evaluated. It can be seen from figure 4(c) that while candidate 1 results in improvement of solution space ($s \in$ [0,0.22]) over the original trajectory ($s \in$ [0,0.19]), the solution space decreases along trajectory 2 ($s \in$ [0,0.182]) . The subsequent perturbations made by the trajectory planner is based on this information and hence candidate trajectory 3 and 4 are obtained by perturbing the path along the same direction as trajectory 1. The gradual increase in the solution space can be seen from figure 4(c). Similarly for the obstacle configuration shown in figure 4(b) first the candidate trajectory 1 is evaluated which leads to decrease in the solution space. Hence the perturbation direction is reversed and candidate trajectories 2, 3 and 4 are obtained which shows gradual improvement in the solution space of s.

Among multiple candidate trajectories, one which minimizes the following cost function is chosen.

$$J_{cost} = w_1\Delta_t + w_2\Delta_p \quad (17)$$

where

$$\begin{cases} \Delta_t = s_{min}, s_{min} > 1 \\ \Delta_t = \frac{1}{s_{min}}, s_{min} < 1 \\ s_{min} = \min \frac{s\sqrt{\dot{x}(t_c)^2 + \dot{y}(t_c)^2}}{v_o(t_c)} \end{cases} \quad (18)$$

$$\Delta_p = \sqrt{(\frac{x_c}{x_{co}})^2 + (\frac{y_c}{y_{co}})^2} \quad (19)$$

In (17), Δ_t is the cost associated with changing the speed i.e accelerating or de-accelerating. s_{min} is a value which measures the minimum deviation from the original velocity profile v_o at time when the solution space is computed i.e t_c. Consequently, $s_{min} > 1$ implies acceleration while $s_{min} < 1$ implies de-acceleration and the magnitude of s_{min} gives a measure of the magnitude of acceleration or de-acceleration. Δ_p is the cost associated with changing the path of the robot and is expressed as the norm of the ratio of perturbed coordinates to the original. w_1 and w_2 are the weights associated with the respective costs. The weights w_1 and w_2 are chosen according to the particular collision avoidance scenario. For example in tight spaces, large deviations from the original path may not be allowed and hence w_2 is chosen much higher than w_1. Similarly if the current velocity of the robot is close to the maximum limit, then candidate trajectories which demands acceleration for collision avoidance has higher w_1.

As an example consider the scenario shown in figure 4(a) and the associated solution space of candidate trajectories shown in figure 4(c). Both candidate trajectory 1, 3 and 4 are acceptable solutions. However in tight spaces, w_1 is set higher such that the planner chooses trajectory 1. Now consider figure 4(d) which shows the solution spaces of candidate trajectories for the scenario shown in figure 4(b). The solution space is such that the robot needs to accelerate to avoid collisions. If the current velocity of the robot is very close to it's maximum limit, then the trajectory planner chooses candidate trajectory 4, even it is deviated a lot from the original path.

In the next section, we describe how to construct a time optimal scaling function which comply with the solution space of s.

V. Optimal Construction of Scaling Function

For collision avoidance the scaling function should comply with the solution space of s i.e the following constraint should be enforced on the scaling function.

$$s_a \leq \frac{dt}{d\tau}(t_c) \leq s_b \tag{20}$$

s_a and s_b defines the boundary of the solution space of s. The profile of the scaling function before and after time t_c is free and hence we utilise this redundancy to create a time optimal scaling function, subject to given velocity and acceleration bounds. To this end, we enforce the following constraints.

$$\|\dot{\mathbf{X}}(\tau)\|_2 \leq v_{max} \Rightarrow \frac{dt}{d\tau}\|\dot{\mathbf{X}}(t)\|_2 \leq v_{max}, |\dot{\theta}(\tau)| \leq \dot{\theta}_{max} \tag{21}$$

$$\frac{dt}{d\tau}(t_o) = s_0, \frac{dt}{d\tau}(t_f) = s_f \tag{22}$$

$$|\ddot{\mathbf{X}}(\tau)| \leq \ddot{\mathbf{X}}_{max} \Rightarrow |\ddot{\mathbf{X}}(t)(\frac{dt}{d\tau})^2 + \dot{\mathbf{X}}(t)\frac{d^2t}{d\tau^2}| \leq \ddot{\mathbf{X}}_{max} \tag{23}$$

$$|\ddot{\theta}(t)(\frac{dt}{d\tau})^2 + \dot{\theta}(t)\frac{d^2t}{d\tau^2}| \leq \ddot{\theta}_{max} \tag{24}$$

(21) states that the total and the angular velocity profile resulting from the scaling function should always satisfy the velocity bounds. Similarly (23) and (24) represents the component wise acceleration bound constraints for linear and angular acceleration components. (22) maintains continuity between the scaled and unscaled velocity at the initial and final boundary points.

(21)-(22) are functional constraints and to solve them in a typical parameter optimization setting, we approximate $\frac{dt}{d\tau}$ by some parametric functions. Since time cannot reverse itself, the scaling function should be positive definite function i.e $\frac{dt}{d\tau} > 0$. Hence an appropriate choice is to represent the scaling function as combination of exponential functions in the form pe^{-qt}. Some other choices for the scaling function like the B-Spline are also presented in the literature [2], [8]. But B-spline functions are not naturally monotonic and hence to ensure positive definiteness, extra conditions have to be enforced on it's control points [2]. In sharp contrast

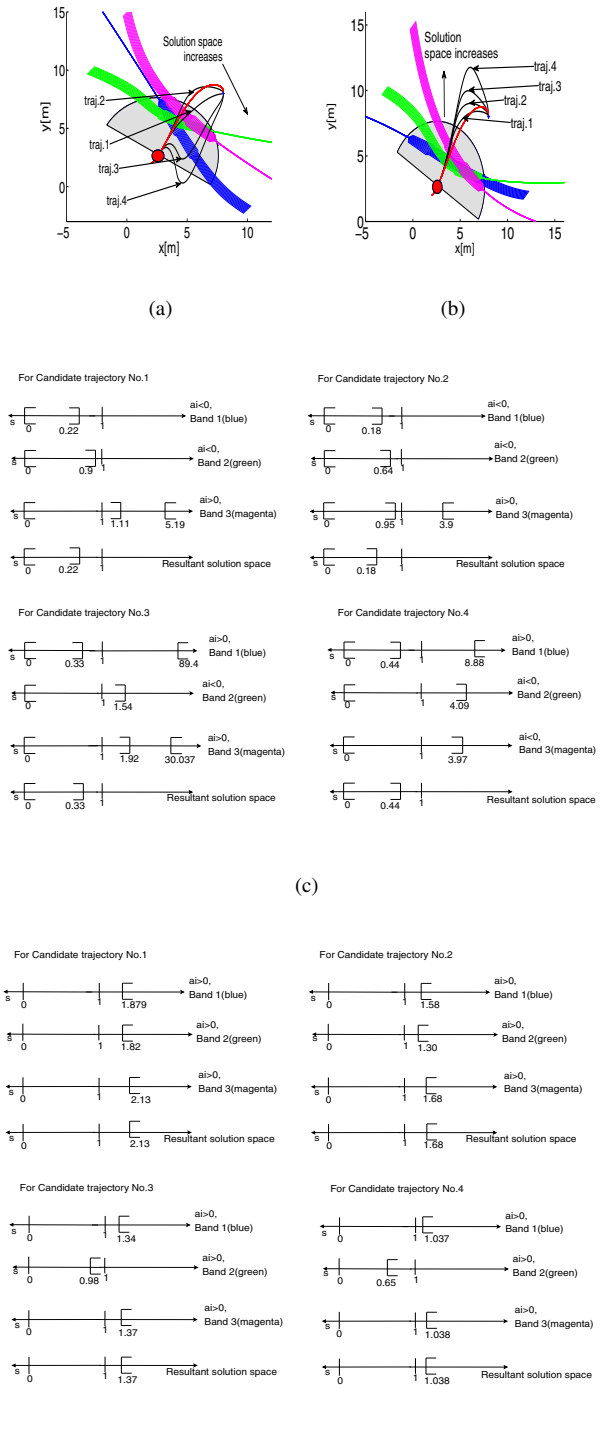

Fig. 4. (a) and (b): The solution space of s is computed along various candidate trajectories. Candidate trajectories are obtained by sequential perturbations of the original trajectory. The variation of solution space of s with perturbations itself dictates the direction of perturbations. In (a) candidate trajectory 1 leads to improvement in solution space over the original trajectory while candidate trajectory 2 leads to reduction in solution space. Hence the planner aligns subsequent perturbations along the direction which produced trajectory 1. The gradual increase in solution space with perturbation is shown in (c) and (d) for the scenarios shown in (a) and (b) respectively

the monotonicity of the exponentials can be exploited to induce natural positive definiteness. Moreover as shown later, the exponential functions leads to a relatively simple optimization structure where most of the constraints are linear.

As shown in figure 5, $\frac{dt}{d\tau}$ is expressed as a combination of exponential functions in the form pe^{-qt}, where p and q are constants and t is the original time scale. The figure shows the interval $[t_0, t_f]$ divided into $n+1$ subintervals by gird points $t_1, t_2, t_3....t_n$. At any subinterval i.e between any two grid points a different exponential function determined by parameters $p_1, q_1, p_2, q_2...p_n, q_n, p_{n+1}, q_{n+1}$ is defined.

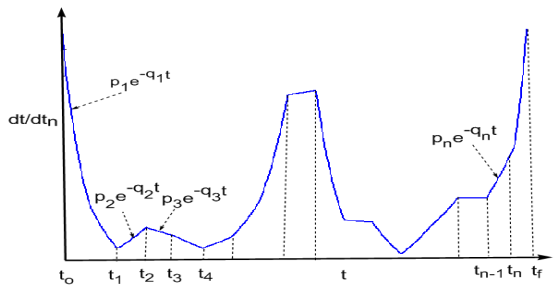

Fig. 5. Representation of scaling function $\frac{dt}{d\tau}$ as a combination of exponential functions. The interval $[t_o, t_f]$ is dicretized into $n+1$ subintervals by the grid points $t_1, t_2, t_3....t_n$ and a different exponential function is fitted in each subinterval.

To derive the general form of $\frac{dt}{d\tau}$ at any subinterval, we start with the first subinterval $[t_0, t_1]$. The initial boundary value of $\frac{dt}{d\tau}$ is s_0. To satisfy the initial boundary condition, we must have

$$\frac{dt}{d\tau}(t_0) = p_1 e^{-q_1 t_0} = s_0 \Rightarrow p_1 = s_0 e^{q_1 t_0} \tag{25}$$

Using (25) $\frac{dt}{d\tau}$ in the interval $[t_0, t_1]$ can be represented in the following form

$$\frac{dt}{d\tau}(t) = s_0 e^{q_1(t_0 - t)}, \forall t \in [t_0, t_1) \tag{26}$$

Similarly from figure 5 , it can be seen that in the interval $[t_1, t_2]$, $\frac{dt}{d\tau}$ is defined as $p_2 e^{-q_2 t}$. To ensure continuity between the scaling functions at the adjacent interval we must have

$$\begin{aligned} \frac{dt}{d\tau}(t_1) = p_2 e^{-q_2 t_1} &\Rightarrow s_0 e^{q_1(t_0 - t_1)} = p_2 e^{-q_2 t_1} \\ &\Rightarrow p_2 = s_0 e^{q_1(t_0 - t_1) + q_2 t_1} \end{aligned} \tag{27}$$

Using (27), $\frac{dt}{d\tau}$ in the interval $[t_1, t_2]$ can be defined as

$$\frac{dt}{d\tau}(t) = s_0 e^{q_1(t_0 - t_1) + q_2(t_1 - t)}, \forall t \in [t_1, t_2) \tag{28}$$

Following the same procedure as above $\frac{dt}{d\tau}$ at the n^{th} subinterval i.e between grid points t_{n-1} and t_n can be represented as

$$\frac{dt}{d\tau}(t) = s_0 e^{q_1(t_0 - t_1) + q_2(t_1 - t_2)...q_n(t_{n-1} - t)}, \forall t \in [t_{n-1}, t_n) \tag{29}$$

$\frac{dt}{d\tau}$ at any arbitrary grid point can be represented using (29) as

$$\frac{dt}{dt_n}(t_n) = s_0 e^{q_1(t_0 - t_1) + ...q_n(t_{n-1} - t_n)} \tag{30}$$

Using (29) it is easy to verify that $\frac{dt^2}{d\tau^2}$ at n^{th} subinterval can be expressed as

$$\begin{aligned} &\frac{dt^2}{d\tau^2}(t) = -q_n e^{2q_1(t_0 - t_1) + ...2q_{n+1}(t_n - t)}, \forall t \in [t_{n-1}, t_n) \\ &= -q_n (\frac{dt}{d\tau})^2 \end{aligned} \tag{31}$$

Hence $\frac{dt^2}{d\tau^2}$ at any arbitrary grid point can be written as

$$\frac{dt^2}{d\tau^2}(t_n) = -q_n (\frac{dt}{d\tau})^2(t_n) \tag{32}$$

Now evaluating the constraints (20) and (21) at the grid points, substituting corresponding expression of $\frac{dt}{d\tau}$ and taking logarithmic transformation, results in following n inequalities.

$$C = \begin{cases} \log(\frac{dt}{d\tau}) - \log(s_b) \leq 0 \\ -\log(\frac{dt}{d\tau}) + \log(s_a) \leq 0 \end{cases} \tag{33}$$

$$C_{vi} = \log(\frac{dt}{d\tau}(t_i)) + \log(\|\dot{\mathbf{X}}(t_i)\|_2) - \log(v_{max}) \leq 0, \quad \forall i = 1, 2, 3....n \tag{34}$$

$$C_{\theta i} = \log(\frac{dt}{d\tau}(t_i)) + \log(|\theta(\dot{t}_i)|) - \log(\dot{\theta}_{max}) = 0 \tag{35}$$

Similarly final boundary condition constraint (22) and acceleration constraints (23) and (24) can be reformulated in the following manner. Only x component constraint is shown. The constrains for the $y-$component, C_{ayi} and the angular component $C_{a\theta_i}$ will have similar form.

$$C_{eq} = \log(\frac{dt}{dt_n}(t_f)) - \log(s_f) = 0 \tag{36}$$

$$C_{axi} = \log(\frac{dt}{d\tau}) + 0.5\log(|\ddot{x}(t_i) - q_i \dot{x}(t_i)|) - 0.5\log(\ddot{x}_{max}) \leq 0, \forall i = 1, 2, ..n \tag{37}$$

To solve (33)-(37) for the parameter q_i (since p_i has already been eliminated), we fit them as constraints in the following optimization problem.

$$\begin{aligned} \min J = \sum_{i=1}^{i=n} [\log(\frac{dt}{d\tau}\|\dot{\mathbf{X}}(t_i)\|_2) - \log(v_{max})]^2 \\ s.t, C \leq 0, C_{vi} \leq 0, C_{\theta i} \leq 0 \\ C_{axi} \leq 0, C_{ayi} \leq 0, C_{a\theta i} \leq 0, C_{eq} = 0 \end{aligned} \tag{38}$$

The optimization problem (38) has a fairly simple structure. The objective function is convex quadratic in terms of parameters q_i and seeks to bring the scaling function as close as possible to the v_{max} profile. This translates to higher velocities along the path which in turn optimizes time. The constraints C, C_{vi}, $C_{\theta i}$ and C_{eq} are linear in terms of parameters q_i, since logarithm of an exponential function is linear. Similarly the first term in the acceleration constraints C_{axi}, C_{ayi} and $C_{a\theta_i}$ are linear. The second term in the

acceleration constraints can be reformulated into a purely concave non-linearity, thus putting the optimization (38) in a *difference of convex form* which can be solved efficiently through *sequential convex programming* [3].

Figure 6 shows the time optimal scaling function created for the candidate trajectory 4 shown in figure 4(c). It can be inferred from it's solution space that $s_a = 0$ and $s_b = 0.44$. $t_o = 4.0$ and $t_c = 4.599$ in this particular case. So a scaling function is constructed from t_o to $t_f = 12.220$ such that it assumes a value from the solution space of s at time t_c. It can be seen from figure 6 that the velocity bound constraints and acceleration bound constraints are perfectly satisfied. A truly time optimal solution is characterized by saturation of atleast one acceleration component when the velocity profile is not at the maximum limit. This is followed quite closely in figure 6. The quality of solution can be further improved by choosing a finer resolution of discretization but at the cost of increased computational time. Hence a trade-off is required between responsiveness in dynamic environments and optimality of the planned motions.

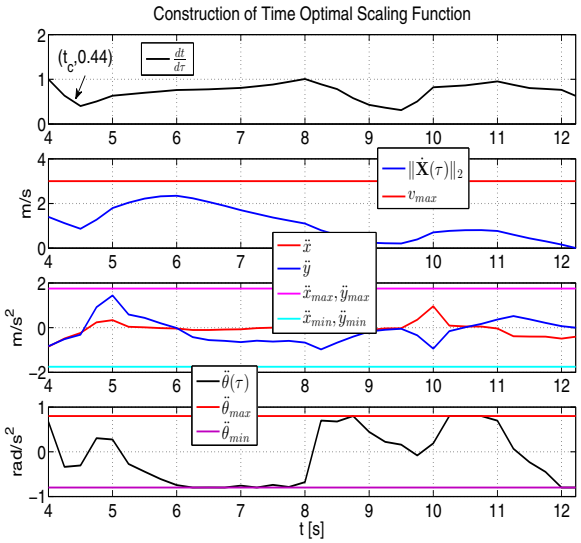

Fig. 6. Figure shows construction of time optimal scaling function. The scaling function is constrained to assume a value from the solution space of s at time t_c. As can be seen that atleast one component of resulting scaling function is near saturation at any-time which is a characteristic of time optimal motions

VI. Additional Results and Discussions

In this section, we present some additional results over those presented in section IV, highlighting the effectiveness of the proposed methodology.

Figure 7(a) shows robot encountering collision with a dynamic obstacle in a constrained space. As before 11 predicted trajectories for the obstacle were considered, which means that the robot needs to compute the solution space for 11 obstacles. Since the collision is head on, the solution space of s along the current trajectory is null. Moreover the constrained space poses restriction in perturbation directions. Hence the resultant trajectory produced by the planner includes reversals for avoiding the obstacle. The collision scenario and the particular avoidance behaviour is similar to that reported in ([9], section IV), but the proposed trajectory optimization is able to reproduce it for high speed trajectories. The time optimal velocity profiles along the collision avoiding trajectory is shown in accompanying video.

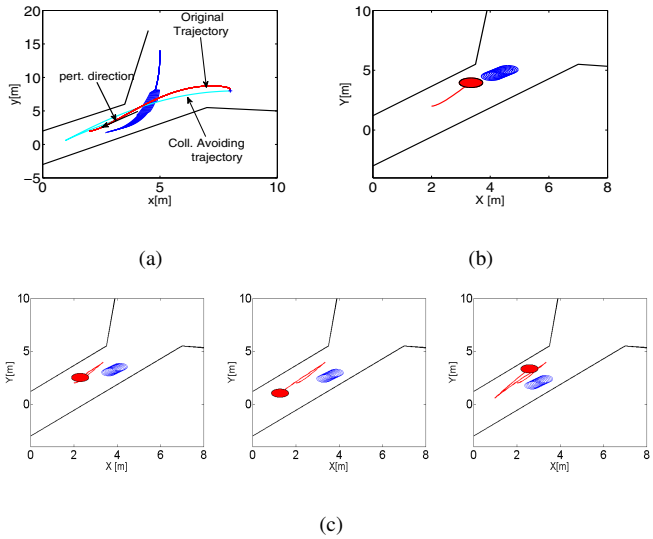

Fig. 7. A challenging scenario where dynamic obstacle is to be avoided in tight corridors. The planner can account for corridor constraints by restricting it's perturbation directions. As shown in (a), the final trajectory incorporates reversals for collision avoidance. (b)-(c) shows snapshots of robot performing collision avoidance.

A. Computational Aspects of the Proposed Trajectory Optimization

The fixed cost of the planner arises out of time optimal framework (section V) which has a MATLAB run time of 30 ms on a standard PC with an Intel Core 2 Duo CPU 2.93 Ghz with 2.0 GB of RAM. But this step needs to be performed only once, when the final selection among all the candidate trajectories has been made. The cost of computing the solution space depends on the number of evaluation of candidate trajectories which in turn depends on the obstacle configuration and number of obstacles. On an average a standard MATLAB implementation allows evaluation of 30 candidate trajectories in 15 ms time interval. Table I summarizes the success rate of the planner in generating a feasible solution space for 100 random scenerios, comprising of simultaneous collision with varied number of obstacles. A solution space is deemed to be feasible, if it can be reached without violating the given velocity and acceleration bounds (section V).

TABLE I

Success Rate of Planner in percentage		
No. of Predicted traj.	15ms time limit	30 ms time limit
11 (1 obst.)	83	94
22 (2 obst.)	76	89
33 (3 obst)	66	83.8
44 (4 obst.)	53.3	79
55 (5 obst.)	45	70

B. Comparison between Proposed and Sampling based planning

Figure 8 shows the comparison between the trajectories obtained from the proposed trajectory optimization approach and the sampling based planners like [6] which considers dynamic obstacles as static obstacles for a short time horizon. These planners perform sampling over (x, y) or (x, y, θ) space. Hence when large number of predicted obstacle trajectories are considered, the resultant robot's trajectory either takes a large detour as shown in figure 8 or resorts to reversal (rightmost figure). To obtain high quality trajectories, its imperative to include time in the search space as done in [11] and [17], although at the cost of increased computational time. In the proposed work the variable s and it's solution space precisely and explicitly states how to exploit the time dimension for collision avoidance. Hence proposed trajectory optimization approach is inherently associated with the same temporal reasoning associated with [11] and [17]. But in contrast to these works, we search in the space of candidate trajectories and since perturbations along various directions can be parameterized, we are effectively searching in a one dimensional space of magnitude of perturbation.

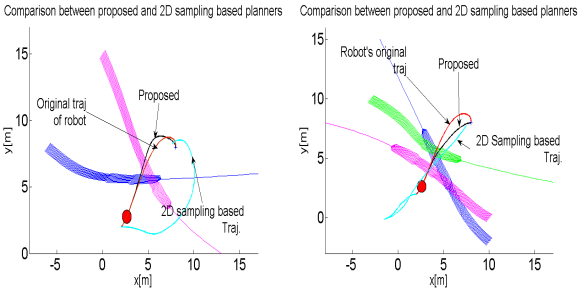

Fig. 8. Comparison between the proposed trajectory planner and 2D (x, y) sampling based approaches like [6] which treats dynamic obstacles as static over a short time horizon. As can be seen that trajectories resulting from 2D sampling based approaches either takes a large detour or resorts to reversals. In sharp contrast, the proposed trajectory planner is based on the concept of collision cone which incorporates second order information like velocity of the obstacles. Moreover by computing solution spaces along candidate trajectories we efficiently exploit the time dimension for collision avoidance. As a result the trajectories resulting from the current planner are very close to the original trajectory.

VII. Conclusions and Future Work

In this paper we presented a trajectory optimization methodology for reactive motion planning in dynamic environments. The methodology was built on top of an efficient framework for computing the intersection/solution space of non-linear, non-convex collision avoidance constraints. The process of computing the solution space was reduced to generating multiple homotopic candidate trajectories and evaluating a symbolic formulae over it.

We also presented a time optimal framework which ensures connection between the current state and the solution space in time optimal fashion. The use of *non-linear time scaling* induced a *difference of convex form* in the time optimization framework which can be solved efficiently through *sequential convex programming*.

There are various future lines of work. The most obvious extension would be to consider obstacles of arbitrary shapes. The collision cone concept for arbitrary shaped obstacles already exists in [4]. We are extending the methodology to Ackerman steered vehicles. To this end we have already developed the methodology to generate multiple candidate trajectories with turn radius constraints.

References

[1] J. Alonso-Mora, M. Rufli, R. Siegwart, and P. Beardsley. Collision avoidance for multiple agents with joint utility maximization. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2833–2838. IEEE, 2013.

[2] Y. Bouktir, M. Haddad, and T. Chettibi. Trajectory planning for a quadrotor helicopter. In *Control and Automation, 2008 16th Mediterranean Conference on*, pages 1258–1263. IEEE, 2008.

[3] S. Boyd. Sequential convex programming. *Lecture Notes, Stanford University*, 2008.

[4] A. Chakravarthy and D. Ghose. Obstacle avoidance in a dynamic environment: A collision cone approach. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 28(5):562–574, 1998.

[5] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, 1998.

[6] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier. Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1056–1062. IEEE, 2008.

[7] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4569–4574. IEEE, 2011.

[8] F. Kanehiro, W. Suleiman, F. Lamiraux, E. Yoshida, and J.-P. Laumond. Integrating dynamics into motion planning for humanoid robots. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 660–667. IEEE, 2008.

[9] M. Likhachev and D. Ferguson. Planning long dynamically feasible maneuvers for autonomous vehicles. *The International Journal of Robotics Research*, 28(8):933–945, 2009.

[10] C. Park, J. Pan, and D. Manocha. Itomp: Incremental trajectory optimization for real-time replanning in dynamic environments. In *ICAPS*, 2012.

[11] M. Phillips and M. Likhachev. Sipp: Safe interval path planning for dynamic environments. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5628–5635. IEEE, 2011.

[12] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 489–494. IEEE, 2009.

[13] A. K. Singh and K. M. Krishna. Reactive collision avoidance for multiple robots by non linear time scaling. In *Decision and Control, 2013. CDC 2013. IEEE International Conference on*. IEEE.

[14] A. K. Singh, K. M. Krishna, and S. Saripalli. Planning trajectories on uneven terrain using optimization and non-linear time scaling techniques. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3538–3545. IEEE, 2012.

[15] C. P. Tang, P. T. Miller, V. N. Krovi, J.-C. Ryu, and S. K. Agrawal. Differential-flatness-based planning and control of a wheeled mobile manipulatortheory and experiment. *Mechatronics, IEEE/ASME Transactions on*, 16(4):768–773, 2011.

[16] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. In *Robotics Research*, pages 3–19. Springer, 2011.

[17] J. P. Van Den Berg and M. H. Overmars. Roadmap-based motion planning in dynamic environments. *Robotics, IEEE Transactions on*, 21(5):885–897, 2005.