Applications of Convex Optimisation in Robotic Motion Planning

Athira Krishnan R
ai22resch11001@iith.ac.in

Tadipatri Uday Kiran Reddy
ee19btech11038@iith.ac.in

Abstract

In this project we try to implement path optimizer and velocity profiles for a car like (4 wheeled) robot configuration. Many works have evolved in and around this problem area, but most of them focus on building a path rather than tracking the path. Through this work we extend the work to three broad spectrum, path generation for known obstacles, path optimization and velocity optimization. In path generation we try to incorporate convex concepts to generate a collision free path. Path optimization would ensure that planned path is optimal considering the robot dimensions and its environment. Velocity optimizer would ensure we track the path to reach destination in an optimal time of travel considering the vehicle dynamics.

1. Introduction

As the field of robotics started to take an intelligent form with advancements in AI and computational capabilities. Convex optimization is a domain that can be applied to get feasible solutions for the robot in its motion planning for a faster traversal, obstacle avoidance and many more. In motion planning area, there are handful of intelligent algorithms (Dijkastra, A*, RRT, etc) developed to find the optimal path for a given destination. These algorithms make use of advanced data structures such as Graphs to compute optimal path in an efficient way. Whereas robots actuator takes velocity as input signal. It is essential to compute the velocities for a given path at each instant. Stephen, etal. [1] proposed velocity optimization with minimum time for different robot configurations. The paper limits in focus on dynamic obstacles. Zhijie, etal [3] proposed extending the capability of elastic band path planning algorithm to dynamic state. CES has shown to improve smoothness of the trajectory and reduce the time taken to navigate. Some works [2] optimize and as also plan the path if any obstacles are found in the way. We would like to initially investigate the convexity of the problem statement, further add such constraints like collision avoidance and re-planning of the path dynamically with changes in environment.

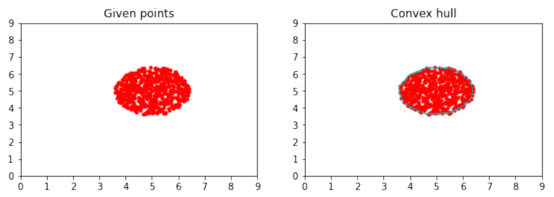

Figure 1. (Left) Obstacle (Right) Convex Hull of Obstacle region

2. Methodology

Goal of this work is divided as three subgoals:

1. Path generation with known obstacle.
2. Path optimization.
3. Velocity optimization.

Each one them are detailed in sub sections.

2.1. Path generation

Path planning or path finding algorithm is a well studied topic with major contributors towads robotics as A*, D*, TEB, RRT, etc. But they are generally formulated in a non-convex setup. Adding convexity is recently being tested with many models. In [], path is assumed to be known from a TEB planner. In our work, we have tried building a obstacle free path from pre-defined start to target locations $\epsilon \mathbf{R}^2$. A known obstacle $\mathbf{O} \epsilon \mathbf{R}^2$, exist in the environment whose boundary is formulated as its convex hull as shown in Figure.1. Path generation algorithm followed is detailed in algorithm 20. Points in the 2D environment $\epsilon \mathbf{C}$, are then randomnly sampled and its absence in obstacle hull is verfied.

2.2. Path optimization

Path generated may not be the optimal path as we randomnly sample the points from the environment $\mathbf{C}$. Following the work [3], in this section we first form bubbles at each way point to know the collision free region around the waypoint. Bubble formation step is detailed in algorithm17. In our implementation we have assumed a polygonal (circular) obstacle region. So bubbles are formed with

Algorithm 1 Path generation with static obstacle

```
1: procedure PLANPATH($pose, target, obstacles$)
2:     $tolerace = 1$
3:     while $||target - start||_2 > tolerace$ do
4:         $point = Getpoint$
5:         $waypoint.append(point)$
6: procedure GETPOINT($pose, target, obstacles$)
7:     threshold=5
8:     start=pose
9:     $hull = ConvexHull(obstaclepoints)$
10:    if $||target - start||_2 > threshold$ then
11:        $r = threshold$
12:        $point = start + (random * r))$
13:    else
14:        $r = ||target - start||_2$
15:        $point = target$
16:    $hullnew = ConvexHull(obstaclepoints, point)$
17:    while $(hullnew == hull)$ do
18:        $point = start + (random * r))$
19:        $hullnew = ConvexHull(obstaclepoints, point)$
20:    return point
```

$rmin = robotradius = 2m$ and $rmax = 5m$. Then we perform bisection method until a feasible r is obtained. Incase the problem is infeasible, then we maintain r as $rmin$ and P **P** as an adjusted point such that the distance is maintained as $rmin$ to nearest obstacle point. Path optimization problem P is defined by,

$$\begin{aligned} &\text{minimize} && 0 \\ &\text{subject to} && Ai \geq 0, ||P - Onear||_2 >= 2r \end{aligned} \tag{1}$$

where Onear is the nearest obstacle point to the current point in waypoint and Ai is the new center to replace Pi **P** such that distance to Onear is atleast rmin. Here objective is to minmize (0) as we are trying to solve the feasibility problem of finding collision free radius. Costraints $Ai \geq 0$ represents non-negativity constraint for selecting center of bubbles and $||P - Onear||_2 >= 2r$ clearly is a convex set. Thus the problem is a convex optimisation problem.

Now path is optimized such that each new waypoint resides within the collision free bubbles formed from each waypoint. According to [3], path optimization problem is formulated as a convex optimization problem with quadratic objective and constraints ($QCQP$). Dynamic constraints on the shape of a trajectory are mimicked by the bending stiffness of the band. Let Q[k] and Q[k+1] be points in bubbles k and k + 1, with k = 1, . . . , n 1. We consider an artificial tensile force F[k] between Q[k] and Q[k+1] given by,

$$F[k] = Q[k+1] - Q[k] \tag{2}$$

Algorithm 2 Bubble generation

```
1: procedure PLANPATH($point, nextpoint$)
2:     $rmin = 2$
3:     $rmin = 5$
4:     $tolerance = 1$
5:     r=0.5(rmax+rmin)
6:     while $rmax - rmin > tolerace$ do
7:         $r = 0.5(rmax + rmin)$
8:         $solveconvexproblem$
9:         if optimal then
10:            $center = Pi$
11:            $r_min = r$
12:            if $r$ then
13:                $center = Ai$
14:                $r = rmin$
15:        else
16:            $center = Pi$
17:            $r_max = r$
```

Balancing force at point Q[k] , k = 1, . . . , n is given by,

$$N[k] = F[k+1] - F[k] \tag{3}$$

Considering the elastic stretching forces, vehicle dynamics (velocity and turning radius), the path optimization problem is modelled as follows,

$$\begin{aligned} &\text{minimize} && ||2 * Q[k] - Q[k-1] - Q[k+1]||^2 \\ &\text{subject to} && Q[1] = P[1], \\ &&& Q[n] = P[n] \\ &&& Q[2] = P[1] + d.(v[1]/||v[1]||) \\ &&& Q[n-1] = P[n] - d.(v[n-1]/||v[n-1]||) \\ &&& ||2Q[k] - Q[k-1] - Q[k+1]|| <= \\ &&& max((d^2/Rmin), \alpha[k].(d/v[k])^2) \end{aligned} \tag{4}$$

where k is the iteration index and n is the number of way points in the path. Here even though the objective function (square of norm of affine) and constraint set (equality , inequality/ half spaces) are convex in terms of variable Q. Objective function being quadratic form, problem is formulated as $QCQP$.

2.3. Velocity optimization

In the previous section we computed a path which can robustly adjust the original path to avoid obstacles. In problem of velocity optimisation, since the path is fixed all of this boils down to is minimizing the time taken for traversing the path.

2.3.1 Dynamics

By applying newton law's of motion on vehicle with p, degrees of freedom and **r** actuators. Position vector is represented as $q \in \mathcal{R}^p$ ans control inputs as $u \in \mathcal{R}^r$. We consider dynamics of second order form.

$$R(q)u = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + d(q) \tag{5}$$

Here *R* is the control matrix, *M* is mass matrix, *C* is centrifugal matrix and *d* is force which is dependent on position vector.

2.3.2 Path interpretation

Since our optimisation problem is minimising the time domain, so we would like to represent everything in a generalised position space.

$$s(\theta(t)) = q(t), t \in [0, T] \tag{6}$$

Where $\theta(0) = 0$ and $\theta(1) = T$.

2.3.3 Optimisation problem

$$\begin{aligned}
\min \quad & T \\
\text{s.t.} \quad & R(q(t))u(t) = M(q(t))\ddot{q(t)} + C(q(t))\dot{q(t)}^2 + d(q(t)) \\
& s(\theta(t)) = q(t) \\
& (\dot{q(t)}^2, \ddot{q(t)}, u(t)) \in \mathcal{C}(q(t)) \\
& t \in [0, T]
\end{aligned} \tag{7}$$

For brevity let us introduce some more variables to make the optimisation problem simpler.

$$\begin{aligned}
\min \quad & \int_0^1 b(\theta)^{-1/2} d\theta \\
\text{s.t.} \quad & \tilde{R}(\theta)u(\theta) = \tilde{m}(\theta) + \tilde{c}(\theta)b(\theta) + \tilde{d}(\theta) \\
& b^{'}(\theta) = 2a(\theta), \theta \in [0, 1] \\
& (a(\theta)), b(\theta), u(\theta)) \in \tilde{\mathcal{C}}(s(\theta))
\end{aligned} \tag{8}$$

Where,

$$\tilde{R}(\theta) = R(s(\theta))$$
$$\tilde{m}(\theta) = M(s(\theta))s^{'}(\theta)$$
$$\tilde{c}(\theta) = M(s(\theta))s^{''}(\theta) + \tilde{C}(\theta) = C(s(\theta))s^{'^2}(\theta)$$
$$\tilde{d}(\theta) = d(s(\theta))$$
$$\tilde{\mathcal{C}} = \{(a(\theta), b(\theta), u(\theta)) | (s^{'}(\theta)^2 b(\theta),$$
$$s^{'}(\theta)a(\theta) + s^{''}(\theta)b(\theta), u(\theta)) \in C(s(\theta))\}$$

2.3.4 Discrete optimisation problem

It is important to discretize the optimisation to implement for practical implementations. Discretizing the problem will increase the throughput of the solver. We apply 1st-order approximation of taylor series on function *b*.

$$b(\theta) = b(\theta_i) + (\theta - \theta_i)\left(\frac{b_{i+1} - b_i}{\theta_{i+1} - \theta_i}\right) \tag{9}$$

$$\implies \int_0^1 b(\theta)^{-1/2} d\theta = 2\sum_{i=0}^{N-1}\left(\frac{\theta_{i+1} - \theta_i}{{b_{i+1}}^{1/2} + {b_i}^{1/2}}\right) \tag{10}$$

In our problem we have lot of derivatives w.r.t θ_i, which in turn is discontinuous thus we should take the derivatives to be constant in the interval of $[\theta_i, \theta_{i+1}]$. To avoid discontinuty at sampling points we take the average .

$$a_i = a\left(\frac{\theta_i + \theta_{i+1}}{2}\right)$$
$$u_i = u\left(\frac{\theta_i + \theta_{i+1}}{2}\right)$$
$$b\left(\frac{\theta_i + \theta_{i+1}}{2}\right) = \frac{b_i + b_{i+1}}{2}$$

$$\implies \tilde{R}(\overline{\theta_i})u_i = \tilde{m}(\overline{\theta_i}) + \tilde{c}(\overline{\theta_i})\frac{b_i + b_{i+1}}{2} + \tilde{d}(\overline{\theta_i}) \tag{11}$$

$$\implies b_i - b_{i-1} = 2a_i d\theta \tag{12}$$

$$\implies s^{'}(\overline{\theta_i}) = \frac{s(\theta_i) - s(\theta_{i-1})}{d\theta} \tag{13}$$

$$\implies s^{''}(\overline{\theta_i}) = \frac{s(\theta_{i-2}) - 2s(\theta_{i-1}) + s(\theta_i)}{d\theta^2} \tag{14}$$

Final optimisation problem is,

$$\begin{aligned}
\min \quad & 2\sum_{i=0}^{N-1}\left(\frac{\theta_{i+1} - \theta_i}{{b_{i+1}}^{1/2} + {b_i}^{1/2}}\right) \\
\text{s.t.} \quad & \tilde{R}(\overline{\theta_i})u_i = \tilde{m}(\overline{\theta_i}) + \tilde{c}(\overline{\theta_i})\frac{b_i + b_{i+1}}{2} + \tilde{d}(\overline{\theta_i}) \\
& b_i - b_{i-1} = 2a_i d\theta \\
& (a(\theta)), b(\theta), u(\theta)) \in \tilde{\mathcal{C}}(s(\theta))
\end{aligned} \tag{15}$$

Constraints are Affine and thus are convex but Objective is a non-linear function. Let's use epigraph form to verify the convexity of objective function.

$$g_i \geq \frac{2d\theta}{b_{i+1}^{1/2} + b_i^{1/2}}$$

$$\implies \frac{2d\theta}{g_i} - b_{i+1}^{1/2} - b_i^{1/2} \leq 0 \tag{16}$$

Clear the above function is convex, since range of objective is always positive, $g_i > 0$ thus $\frac{1}{g_i}$ is convex and functions of

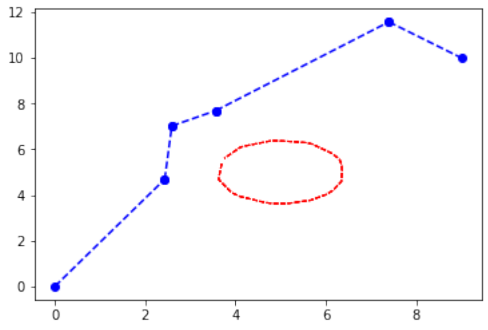
Figure 2. Generated path from [0,0] to [9,10]

form $-\sqrt{x}$ are indeed convex. And finally sum of convex function is convex. Now we see that the above optimisation problem is convex. Coding the same objective function to *CVXPY* would raise a *DCP ERROR*, this occurs because the function doesn't comply to the rule set of solver. We transform our optimisation problem to epigraph form so that *DCP rules* are satisfied.

$$\begin{aligned} \min \quad & \sum_{i=0}^{N-1} g_i \\ \text{s.t.} \quad & \frac{2d\theta}{g_i} - b_{i+1}^{1/2} - b_i^{1/2} \leq 0 \\ & \text{Constraints of Equation (15)} \end{aligned} \tag{17}$$

We can perform transformation of variables on $b_i^{1/2}$ and convert this problem with linear objective and quadratic constraints. This is clearly a *SOCP*.

3. Implementation Results

We have implemented [1] path optimizer and velocity optimizer using *CVXPY*. For implementation, a 2D environment is assumed. Path generated from [0,0] to [9,10] through the random sampling avoiding the obstacle hull is shown in Figure.2. The path is passed to bubble generation stage resulting in Figure.3. The resultant path from optimizer is shown in Figure.4

Through path optimization we could reduce path length by **26.7%** the length of the generated path (=14.255m) to length of optimized path (=10.436) with a optimal cumulative balancing cost measured as 5.36. All distance measures are assumed to be in meter.

Now we pass this optimized path to velocity optimizer. The obtained traversal path time is **9.56s**. Figure 6 are the optimal force inputs $(flong, flat)$. Figure 5 shows the true path along with simulated path which is obtained by observing the trajectory of car with optimal inputs(force). We see that path is smooth and also very close to the true path. In figure 7 and 8 show velocity and acceleration profiles during the trajectory.

[1]https://github.com/Athirakr94/robot-nav-convex

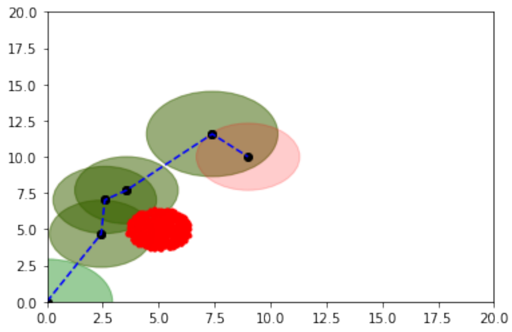
Figure 3. Collision free (with obstacle) bubbles formed around each waypoint

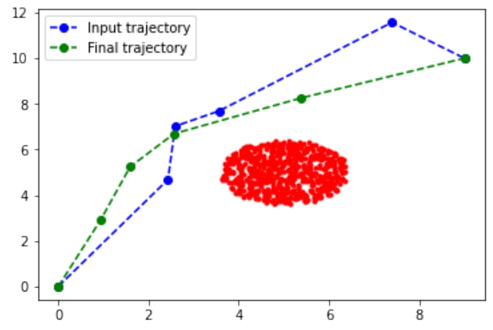

Figure 4. Generated path(blue) and Optimized path (green)

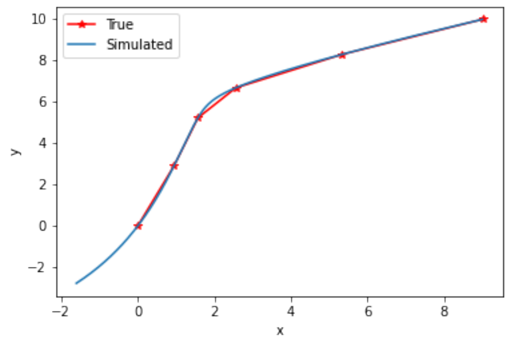

Figure 5. Traversed path

4. Conclusions

Using convex optimization tools we could understand recreate the model discussed in [3] and [1], the problem of optimal robot motion planning for a known configuration. The same can be extended to another robot configuration by changing the vehicle dynamics considered. These models are capable of finding the optimal smooth trajectory and velocity profile to be followed. This makes the approach a potential alternative future work on robot motion planning

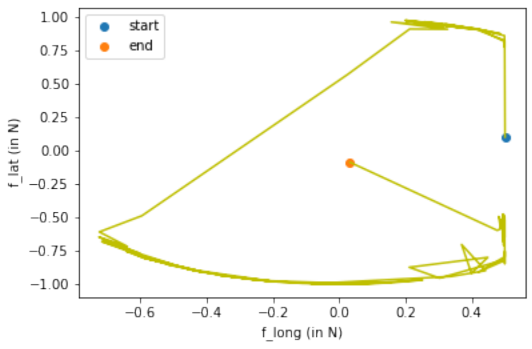

Figure 6. Control Inputs,i.e, Force

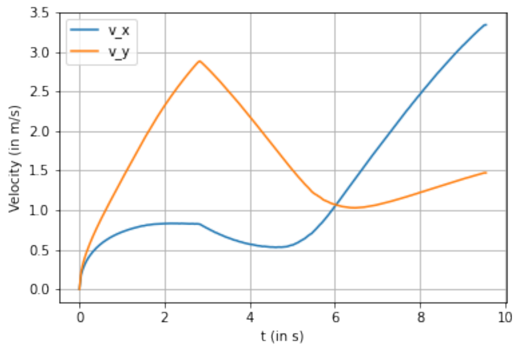

Figure 7. Velocity profile in trajectory

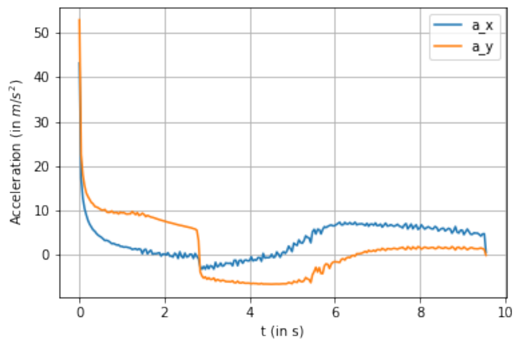

Figure 8. Acceleration profile in trajectory

in dynamic environment.In scenario with dynamic obstacles, task from re planning to its tracking would be expected to happen in real time to avoid any potential collision.

References

[1] Thomas Lipp and Stephen Boyd. Minimum-time speed optimisation over a fixed path. *International Journal of Control*, 87(6):1297–1311, 2014. 1, 4

[2] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014. 1

[3] Zhijie Zhu, Edward Schmerling, and Marco Pavone. A convex optimization approach to smooth trajectories for motion planning with car-like robots. In *2015 54th IEEE conference on decision and control (CDC)*, pages 835–842. IEEE, 2015. 1, 2, 4