# Functional Gradient Optimization for Path Planning in Dynamic Environments

Aashi Manglik

December 2017

## 1 Introduction

Robotic path planning in presence of moving obstacles like pedestrians is challenging because it requires prediction of their future trajectories and then accordingly adapt the pre-planned path, for example, wait, move to a side or overtake. To facilitate this human-like behavior, we need to add time to robot's state in addition to the configuration space variables. But this inclusion of time dimension will increase the search space and thus the planner will face huge difficulty to provide a solution in real-time. To avoid this, most real-time approaches treat obstacles as static for a specified time window and keep re-planning as they move. These approaches may result in unnecessary detours or no solution as shown in Figure 1. Phillips and Likhachev [2] developed a planner that considers collision time intervals to search for an optimal path in grid similar to A*. Planning with time interval (i.e., time period for a configuration with no collisions) instead of each timestep resulted in much lesser number of states and made their algorithm feasible to use in real-time. But this is a grid-based approach and we aim to find a solution for continuous domain without the need of discretizing the environment into cells. To plan a path in continuous space, sampling-based planners like RRT (Rapidly Exploring Random Trees) and RRT* are previously used in robotic applications but extending them to include time-based reasoning is challenging and will significantly increase their planning time.

In this work, we formulate path planning as a trajectory optimization problem where the goal is to find a trajectory that minimizes a given cost function while satisfying the given constraints. Covariant Hamiltoninan Optimization for Motion Planning (CHOMP) [1] utilizes covariant gradient descent to minimize obstacle and smoothness cost functionals. Their cost functional is designed for static environments, here we made an attempt to propose a dynamic cost functional and thus handle dynamic obstacles in the environment.

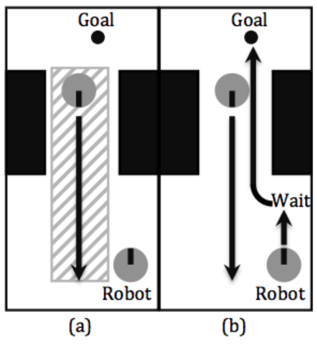


Figure 1: (a) If the moving obstacle is considered to be static, no solution to reach the goal. (b) If trajectory of obstacle is considered, planning with time could find a solution by waiting and then proceeding [2]

## 2 Modifying a Path using Optimization Theory

The standard form for an optimization problem may be given as follows:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & F(x) \qquad\qquad x \in \mathbb{R}^n \\ \text{subject to} \quad & f_i(x) \le 0,\ i = 1, \ldots, m. \\ & h_i(x) = 0,\ i = 1, \ldots, n. \end{aligned}$$

Optimizing a path cannot be directly expressed in above form as an arbitrary path cannot be described by a point in some finite-dimensional space $\mathbb{R}^n$. Instead, as Quinlan [3] proposed, it can be stated as:

$$\begin{aligned} \underset{\xi}{\text{minimize}} \quad & F[\xi] \qquad\qquad \xi : [0,1] \rightarrow C \\ \text{subject to} \quad & \xi(\tau) \in C_{free},\ \tau \in [0,1] \\ & \xi(0) = q_{start},\ \xi(1) = q_{goal}, \\ & \xi \text{ is a smooth function.} \end{aligned}$$

Here, C is a configuration space, $C_{free}$ is collision-free configuration space and $\xi$ represents the path. $\tau$ is a stepping parameter that monotonically increases as we traverse the trajectory. It is 0 at the initial position and 1 at the goal.
Path is a geometric object and its optimization can be seen as optimizing its shape. Reparametrizing a path does not change its shape and hence we require that the objective functional F should be independent of the particular parametrization of the path. If we define F as length of the path, it is invariant to parametrization. As the path is constrained to be collision-free, the minimum length path will then consist of straight line segments as in visibility graph. To improve the shape of path, a positive cost is associated with each configuration in free space and the objective functional F is defined as path length weighted by its cost denoted by $c$.

$$F[\xi] = \int_0^1 c(\xi(\tau)) \left\| \frac{d\xi(\tau)}{d\tau} \right\| d\tau \tag{1}$$

## 3 CHOMP

CHOMP [4] is a functional gradient optimization algorithm that minimizes a cost functional to give smooth collision-free trajectories. The objective functional has two components: a smoothness term that measures the shape of trajectory and an obstacle term (same as Eq. 1) that measures its proximity to obstacles.

$$U[\xi] = F_{obs}[\xi] + \lambda F_{smooth}[\xi] \tag{2}$$

$$F_{smooth}[\xi] = \frac{1}{2} \int_0^1 \left\| \frac{d\xi(\tau)}{d\tau} \right\|^2 d\tau \tag{3}$$

$$F_{obs}[\xi] = \int_0^1 c(\xi(\tau)) \left\| \frac{d\xi(\tau)}{d\tau} \right\| d\tau \tag{4}$$

To implement CHOMP, we use a uniform discretization which samples $n$ waypoints from the trajectory function $\xi(\tau)$ over equal steps of length $\Delta\tau$. $\xi \approx (q_1^T, q_2^T, ..., q_n^T)^T \in \mathbb{R}^{n\times d}$, with $q_0$ and $q_{n+1}$ as the starting and ending points of the trajectory respectively. $d$ is the dimension of configuration space. Using this waypoint parametrization, we can write the smoothness objective given in equation 3 as a series of finite differences:

$$F_{smooth}[\xi] = \frac{1}{2(n+1)} \sum_{i=0}^{n} \left\| \frac{q_{i+1} - q_i}{\Delta\tau} \right\|^2 = \frac{1}{2} \|K\xi + e\|^2$$

Here, $K$ is a finite differencing matrix and $e$ handles boundary conditions $q_0$ and $q_{n+1}$. If $A = K^T K$, $b = K^T e$ and $c = \frac{1}{2} e^T e$, then

$$F_{smooth}[\xi] = \frac{1}{2} \xi^T A \xi + \xi^T b + c \tag{5}$$

The Euclidean inner product in the discrete trajectory space can be written as:

$$\langle \xi_1, \xi_2 \rangle = \xi_1^T \xi_2$$

To measure the closeness or similarity between two trajectories, we can compute:

$$\|\xi_1 - \xi_2\| = \sqrt{\langle \xi_1, \xi_2 \rangle}$$

Under this metric, closeness of two trajectories can be misleading as their cost $U$ could be very different. We will thus define an alternative Euclidean product whose induced metric is sensitive to the cost $U$ imposed in the trajectory space.

$$\langle \xi_1, \xi_2 \rangle_A = \xi_1^T A \xi_2 = \|\xi_1 - \xi_2\|_A^2$$

Here, $A$ is the same symmetric positive-definite matrix derived in eq 5. It measures the total amount of acceleration in the trajectory.

CHOMP iteratively minimizes the objective functional $U$ given by eq 2 over the space of trajectories by minimizing a regularized first order Taylor expansion of $U$ around the current path $\xi_i$:

$$\xi_{i+1} = \arg\min_{\xi} \quad U[\xi_i] + \bar{\nabla} U[\xi_i]^T(\xi - \xi_i) + \frac{\eta}{2}\left\|\xi - \xi_i\right\|_A^2 \tag{6}$$

Here, $\eta$ is the regularization parameter. By differentiating the term on right hand side of equation 6 with respect to $\xi$ and setting it to zero, we get the following update rule:

$$\implies \xi_{i+1} = \xi_i - \frac{1}{\eta}A^{-1}\bar{\nabla}U[\xi_i] \tag{7}$$

## 4 Optimizing in Space-Time

Till now we only considered the shape of trajectory, not its timing. In particular, we know the desired positions of sampled waypoints but at what time a given waypoint should be reached is unknown. To facilitate this, we will introduce a time trajectory $t : [0,1] \to \mathbb{R}^+$ as a smooth function mapping $\tau \in [0,1]$ to time $t(\tau) \in \mathbb{R}^+$ [1]. From now on, $()'$ refers to $\frac{d}{d\tau}$. Obstacle and smoothness functionals given in equations 3 and 4 can be replaced as:

$$F_{smooth}[\xi, t] = \frac{1}{2}\int_0^1 \left(\left\|\frac{d\xi(\tau)}{d\tau}\right\|^2 + \gamma\left\|\frac{dt(\tau)}{d\tau}\right\|^2\right) d\tau \tag{8}$$

$$F_{obs}[\xi, t] = \int_0^1 c(\xi, t)\frac{\|\xi'\|}{t'}d\tau \tag{9}$$

The integrand inside obstacle functional $F_{obs}$ is the product of cost at a waypoint multiplied by the speed at that point. Through gradient descent we minimize $F_{obs}$. This implies that if the robot cannot escape a high cost region, it should slow down. This seems consistent with how people navigate in crowd, they slow down in order to avoid collisions with other people.
The cost functional $c$ and objective functional $U$ are functions of both configuration space and time. Update equations change as follows:

$$(\xi_{i+1}, t_{i+1}) = \arg\min_{\xi, t}\ U[\xi_i, t_i] + \bar{\nabla}U^{\xi}[\xi_i, t_i]^T(\xi - \xi_i) + \bar{\nabla}U^{t}[\xi_i, t_i]^T(t - t_i) + \frac{\eta_1}{2}\left\|\xi - \xi_i\right\|_A^2 + \frac{\eta_2}{2}\left\|t - t_i\right\|_B^2 \tag{10}$$

$$\xi_{i+1} = \xi_i - \frac{1}{\eta_1}A^{-1}\bar{\nabla}U^{\xi}[\xi_i, t_i] \tag{11}$$

$$t_{i+1} = t_i - \frac{1}{\eta_2}B^{-1}\bar{\nabla}U^{t}[\xi_i, t_i] \tag{12}$$

We computed the functional gradients of objective functional $U[\xi, t]$ with respect to $\xi$ and $t$ by applying Euler-Lagrange equations.

$$F_{obs}[\xi, t] = \int_0^1 f(\xi, \xi', t, t')d\tau$$

$$\bar{\nabla}F_{obs}^{\xi}[\xi, t] = \frac{\partial f}{\partial \xi} - \frac{d}{d\tau}\frac{\partial f}{\partial \xi'}$$

$$\bar{\nabla}F_{obs}^{t}[\xi, t] = \frac{\partial f}{\partial \xi} - \frac{d}{d\tau}\frac{\partial f}{\partial \xi'}$$

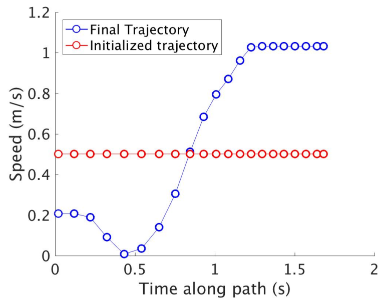
After substituting $f$ from eq 9, we get

$$\bar{\nabla}F_{obs}^{\xi}[\xi, t] = \frac{\|\xi'\|}{t'}\left[(I - \hat{\xi}'\hat{\xi}'^T)\nabla_{\xi}c - \hat{\xi}'^T\nabla_t c - ck + c\frac{\xi'}{\|\xi'\|^2}\frac{t''}{t'}\right]$$

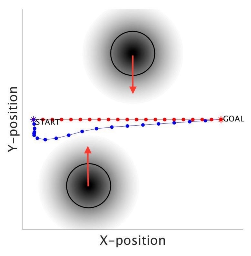
$$\bar{\nabla}F_{obs}^{t}[\xi, t] = \frac{\|\xi\prime\|}{t'^2}\left[\xi'\nabla_{\xi}c + 2t'\nabla_t c + c\frac{\xi'^T\xi''}{\|\xi''\|^2} - 2c\frac{t''}{t'}\right]$$

Here, $k = \frac{(I - \hat{\xi}'\hat{\xi}'^T)\xi''}{\|\xi'\|^2}$

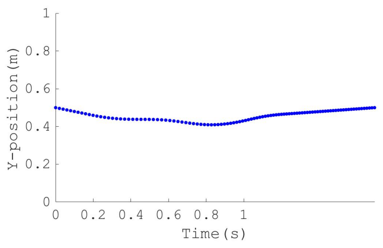
$$\bar{\nabla}F_{smooth}^{\xi}[\xi, t] = -\xi''(\tau) \quad \bar{\nabla}F_{smooth}^{t}[\xi, t] = -\gamma t''(\tau) \tag{13}$$



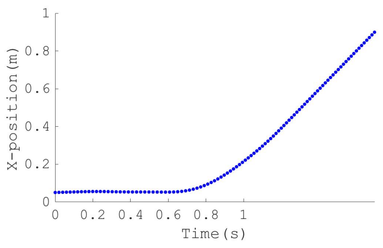
(a)



(b)



(c)



(d)

Figure 2: Test Case 1

### 4.1 Monotonicity constraints in time

$t$ is a sequence of $n+1$ waypoints, i.e., $t = [T_1, T_2, ...., T_N]$ with $T_0 = 0$ and $T_{N+1} = T_{goal}$. The monotonicity constraint on $t$ is formulated as a set of linear inequality constraints:

$$\Delta t = (T_j - T_{j-1}) \geq \epsilon_t; \forall j = 1, 2, ..., n+1 \tag{14}$$

In the test case presented in figure 2, $n = 101$ and $\epsilon_t = 8 \times 10^{-3}$. $n$ and $\epsilon_t$ need to be approximately chosen such that the instantaneous speed for final trajectory can remain close to walking speed. A lagrangian is constructed by adding these constraints to the objective functional and the Lagrangian multipliers are computed by applying KKT optimality conditions.

## 5 Cost function

Let $\xi = (x, y)$ be the robot's coordinates at time $t$. Consider an obstacle in the environment whose initial position at time $t = 0$ is $(x_0, y_0)$ and it continues to move with a constant velocity $(v_x, v_y)$ during the planning horizon. The Euclidean distance between robot and the obstacle at any time $t$ within planning horizon is then given as follows:

$$D(\xi, t) = \sqrt{(x - x_0 - v_x t)^2 + (y - y_0 - v_y t)^2} \tag{15}$$

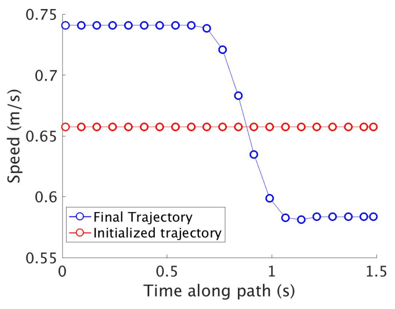
For dynamic environment, $D(\xi, t)$ is the distance from any point in configuration space to surface of the nearest obstacle at time t. The cost $c(\xi, t)$ is then defined as:

$$c(\xi, t) = \begin{cases} -D(\xi, t) + \frac{\epsilon}{2} & D(\xi, t) \leq 0 \\ \frac{1}{2\epsilon}(D(\xi, t) - \epsilon)^2 & 0 < D(\xi, t) \leq \epsilon \\ 0 & D(\xi, t) > \epsilon \end{cases}$$

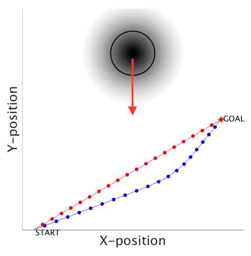
Here, $\epsilon$ is the desired minimum distance from the surface of any obstacle.

## 6 Experimentation and Results

We tested the trajectory optimization explained above on multiple scenarios, two of which are presented here. In first test case shown in Figure 2, there are two dynamic obstacles shown as black circles in 2a. The left obstacle is going up with a constant speed of 0.5m/s while the right one is going down. Figure 2a shows the



(a)



(b)

Figure 3: Test Case 2

speed adaptation. Initially the speed is very low and even goes to 0. This indicates that the robot tries to wait for the two obstacles to pass first and then proceeds ahead with increasing speed to reach its goal. The speed saturates to 1m/s which is close to the average walking speed of pedestrians. In second scenario shown in figure 3, the obstacle has a slow speed of 0.3m/s and the robot overtook it by slightly increasing its speed.

## 7 Conclusions

Since this approach is based on gradient descent, the solution can converge to a bad local minima. The resulting path is sensitive to parameters such as number of waypoints, minimum time difference between two consecutive waypoints ($\epsilon_t$) and initialization of time and position trajectories. In case of more number of obstacles, the cost function is highly non-convex and convergence to the optimal path becomes less likely. At present the proposed cost function is naive based on the assumption that the obstacles can be represented as circular discs which is unsuitable when we have walls and other static obstacles. In static environments, the obstacles are represented through occupancy grid or map. Creating a occupancy map with time dimension in 3D $(x, y, t)$ will be both memory and computationally intensive.

## References

[1] A. Byravan, B. Boots, S. S. Srinivasa, and D. Fox. Space-time functional gradient optimization for motion planning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6499–6506, May 2014.

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

[3] S. Quinlan. *Real-Time Modification of Collision-Free Paths.* Thesis, 1994.

[4] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *2009 IEEE International Conference on Robotics and Automation*, May 2009.