CHOMP Notes

Jennifer King

March 19, 2013

CHOMP formalization

Define a trajectory as a function which maps time to a configuration in configuration space:

$$\xi : [0, 1] \rightarrow \mathcal{C} \subset \mathbb{R}^d$$

Then define an objective functional which maps trajectories to a real number:

$$\mathcal{U} : \Xi \rightarrow \mathbb{R}$$

For CHOMP, the objective functional represents a tradeoff between two complementary aspects of motion planning. The first penalizes a trajectory based on dynamic criteria such as velocity, acceleration or jerk. The second penalizes a trajectory based on proximity to obstacles. This is represented in the following equation:

$$\mathcal{U}[\xi] = \mathcal{F}_{obs}[\xi] + \lambda \mathcal{F}_{smooth}[\xi]$$

where

$$\mathcal{F}_{obs} : \Xi \rightarrow \mathbb{R}$$

$$\mathcal{F}_{smooth} : \Xi \rightarrow \mathbb{R}$$

and λ is a weighting factor which represents the tradeoff between the two sometimes conflicting functional values.

We can define the smoothness functional as the integration across the trajectory of any function which represents dynamic criteria. One option, shown here, is to use squared velocity norms:

$$\mathcal{F}_{smooth} = \frac{1}{2} \int_0^1 \left\| \frac{d}{dt} \xi(t) \right\|^2 dt \tag{1}$$

Now define a function x as follows:

$$x : \mathcal{C} \times \mathcal{B} \rightarrow \mathbb{R}^3$$

which maps a configuration and body point to a point in the workspace. If we then have a cost function:

$$c : \mathbb{R}^3 \to \mathbb{R}$$

which assigns a cost to each point in the workspace, we can define the obstacle objective function:

$$\mathcal{F}_{obs}[\xi] = \int_0^1 \int_{\mathcal{B}} c(x(\xi(t), u)) \left\| \frac{d}{dt} x(\xi(t), u) \right\| du dt \tag{2}$$

If we think about the arc length (s) between two points, a and b, for a curve parameterized by t:

$$s = \int_a^b \sqrt{\frac{dx}{dt}^2 + \frac{dy}{dt}^2}$$

Then its easy to see that the $\mathcal{F}_{obs}$ function multiplies the cost by the arc length rather than just doing a line integral. This ensures that the obstacle objective is invariant to retiming the trajectory. Figure **??** visualizes this point.

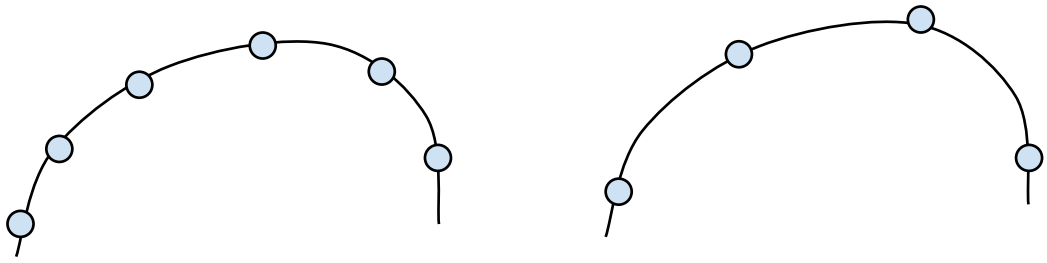

Figure 1: The trajectories on the left and right have same length. If cost was based on integrating across trajectory points, the left would have much higher cost. Using arc length makes cost of each trajectory the same.

We want to use a traditional gradient descent algorithm to minimizes our object function $\mathcal{U}[\xi]$:

$$\xi_{t+1} = \xi_t - \eta_i \bar{\nabla} \mathcal{F}[\xi]$$

To do this we must define the concept of functional gradient. The functional gradient is the perturbation ϕ that causes maximal increase to the function. Where:

$$\phi : [0, 1] \to \mathcal{C} \subset \mathbb{R}^d$$

In other words we want to find:

$$\bar{\nabla} \mathcal{U} = \arg\max_{\phi} \mathcal{U}[\xi + \phi]$$

It can be seen from the above equation that the outcome of the maximization is heavily dependent on the size of ϕ. Because ϕ represents a normalized vector, this is equivalent to saying the outcome of the maximization is strongly dependent on our choose of norm. **TODO: Check this statement to be sure the understanding is correct.** One option is the Euclidean norm. For a vector $x \subset \mathbb{R}^n = (x_1, ..., x_n)$:

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$$

For a trajectory:

$$\|\xi\| = \sqrt{\int_0^1 \xi(t)dt}$$

Using this as our norm, the Euclidean function gradient for a function $\mathcal{F}[\xi] = \int v(\xi(t))dt$ is:

$$\bar{\nabla}\mathcal{F}[\xi] = \frac{\partial v}{\partial \xi} - \frac{d}{dt}\frac{\partial v}{\partial \xi'} \tag{3}$$

This is the Euler-Lagrange equation.
Now consider equation 1. Let:

$$v(\xi(t)) = \left\| \frac{d}{dt}\xi(t) \right\|^2$$

Then **TODO: Verify this**:

$$\frac{\partial v}{\partial \xi} = 0$$

$$\frac{\partial v}{\partial \xi'} = 2\xi'(t)$$

$$\frac{d}{dt}2\xi'(t) = 2\frac{d^2}{dt^2}\xi(t)$$

By our definition:

$$\mathcal{F}_{smooth} = \frac{1}{2}\int v(\xi(t))dt$$

So using equation 3:

$$\bar{\nabla}\mathcal{F}_{smooth} = \frac{1}{2}\left(-2\frac{d^2}{dt^2}\xi(t)\right) = -\frac{d^2}{dt^2}\xi(t)$$

Next we consider equation 2. Let:

$$v(\xi(t)) = \int_{\mathcal{B}} c(x(\xi(t), u)) \left\| \frac{d}{dt}x(\xi(t), u) \right\| du$$

TODO: Work out this math like we did the other

Note: We formulated $\mathcal{F}_{smooth}$ and $\mathcal{F}_{obs}$ so that they did not rely on trajectory parameterization. Thus none has been specified. However, in order to do gradient descent, we had to pick a norm. We picked Euclidean norm. Because of this we have now created a dependency between our algorithm formulation and the choice of representation of the trajectory. Specifically, we assume the trajectory function is represented in terms of a basis of Dirac delta functions so that the i^{th} component is $\langle f, \delta_t \rangle = \int f(\tau)\delta(t-\tau)d\tau = f(t)$.

TODO: Explain this more

TODO: Double check that these next two paragraphs accurately represent the content of section 3.3 of the journal paper

The formulation above uses Euclidean norm specifically. However this is not required. What is required is that the norm we define over our space must depend solely on the dynamical quantities of the trajectory (in engish, only on the trajectory and its derivatives). In other words, we want to define a norm as follows:

$$\|\xi\|_A^2 = \int \sum_{n=1}^{k} \alpha_n (D^n \xi(t))^2 dt$$

Here D^n is an n^{th} order derivative operator and $\alpha_n \subset \mathbb{R}$ is a constant. If $k = 1$ and $\alpha_1 = 1$ then:

$$\|\xi\|_A^2 = \int \xi'(t)^2 dt$$

We can also define an inner product on two trajectories:

$$\langle \xi_1, \xi_2 \rangle = \int \sum_{n=1}^{k} \alpha_n (D^n \xi_1(t))(D^n \xi_2(t)) dt$$

We use the A operator only to differentiate this norm and dot product from the standard Euclidean norm. A must be constructed such that $A = D^T D$ where D is a consituent differential operator **TODO: What is this?** Then:

$$\langle f, Af \rangle = \int (Df(t))^2 dt$$

Identifying Parameterization

In order to numerically perform a function gradient, a specific parameterization of the trajectory must be selected. One choice is to sample from the trajectory at uniform time steps Δt: $\xi = \left[q_1^T, ..., q_n^T\right]^T$. We let q_0 represent the start configuration and q_{n+1} represent the goal configuration. Then the smoothness functional can be represented as:

$$\mathcal{F}_{smooth} = \frac{1}{2(n+1)} \sum_{t=1}^{n+1} \left\| \frac{q_{t+1} - q_t}{\Delta t} \right\|^2$$

We can generate an n by n differencing matrix K:

$$K = \begin{bmatrix} 1 & 0 & 0 & ... & 0 \\ 0 & -1 & 1 & ... & 0 \\ . & . & . & . & . \\ 0 & 0 & ... & -1 & 1 \\ 0 & 0 & ... & 0 & -1 \end{bmatrix}$$

Then if we define a vector e to handle endpoints conditions:

$$e = \begin{bmatrix} -q_0 & 0 & ... & 0 & q_{n+1} \end{bmatrix}^T$$

We can rewrite our smoothness function:

$$\mathcal{F}_{smooth} = \frac{1}{2} \|K\xi + e\|^2$$

Multiplying this out we get:

$$\mathcal{F}_{smooth} = \frac{1}{2}(\xi^T K^T K\xi + 2\xi^T K^T e + e^T e) = \frac{1}{2}\xi^T A\xi + \xi^T b + c$$