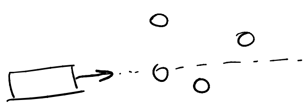
Collision Avoidance documentation

Friday, January 25, 2019 6:10 PM

Idea:

Use gradient descent to determine the ideal steering angle



Robot's environment → model each landmark as a Gaussian function

$$J = \underset{\theta}{\operatorname{argmin}} \left[\int_x P(\theta, x)\, dx + \frac{1}{2} \| \theta - \delta \|^2 \right]$$

basically, we want to pick the trajectory associated with the steering angle θ that has the fewest obstacles (we're integrating the probability of there being an obstacle at arc length x on trajectory θ

We also want to steer as little as possible which is why we add the regularizer $\frac{1}{2} \| \theta - \delta \|^2$, δ is our current steering angle.

The robot should also attempt to steer towards its goal as well, so after this step we run the standard controller.

$$\theta_{t+1} \leftarrow \theta_t + \gamma \nabla J$$

$$\nabla J = \frac{dJ}{d\theta} = \frac{d}{d\theta} \left(\int_x P(\theta, x)\, dx + \frac{1}{2} \| \theta \|^2 \right)$$

$$= \frac{d}{d\theta} \left(\int_x P(\theta, x)\, dx \right) + \| \theta \|$$

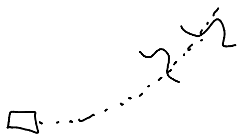
$$= \int_x \frac{d}{d\theta} \left(P(\theta, x) \right) dx + \| \theta \|$$

$\frac{d}{d\theta} \left(P(\theta, x) \right)$ is the derivative of a Gaussian whose mean

... all the obstacles along

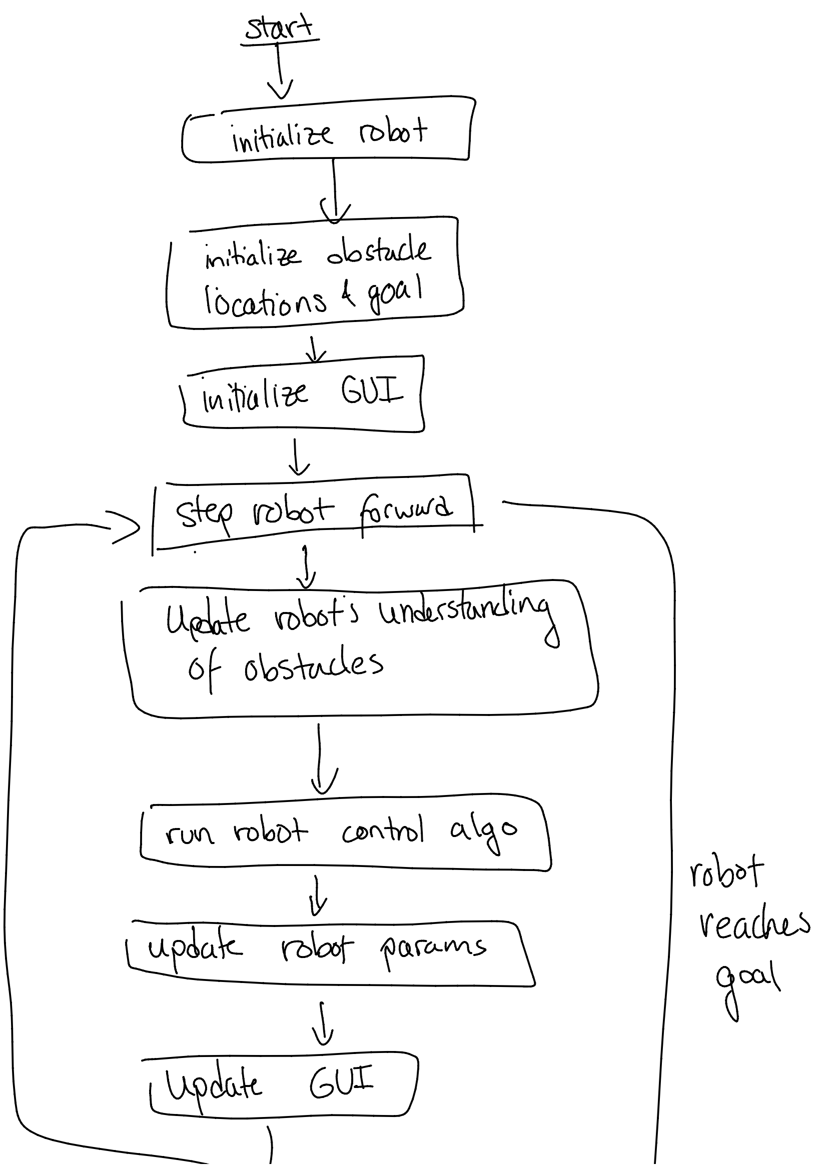
$\frac{d}{d\theta}(P(\theta, x))$ is the derivative of a Gaussian whose mean is an obstacle, so basically take all the obstacles along that trajectory and add their derivatives up

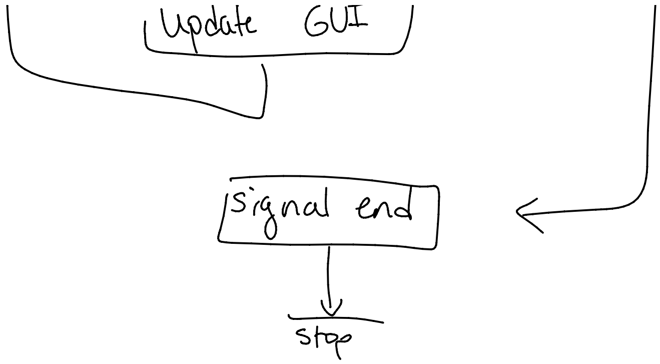
we also need to consider the following:



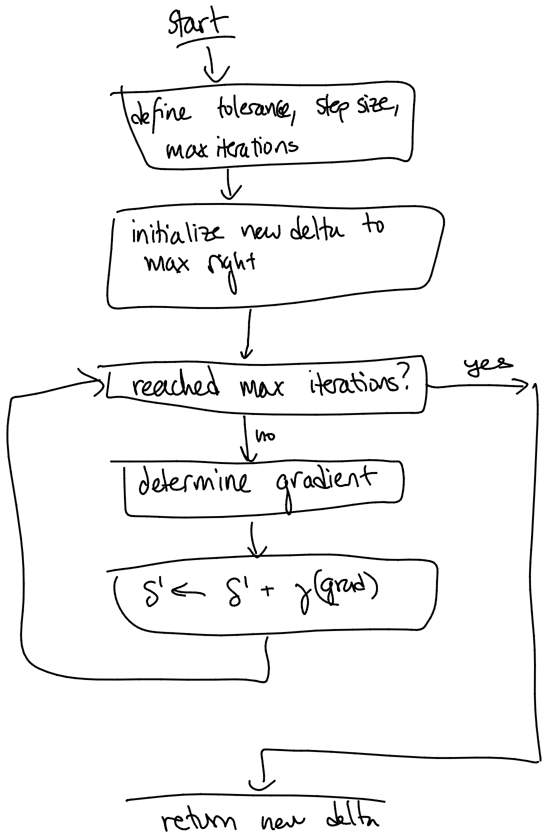
in this situation it's ok because the 2 obstacles will cancel each other out. Therefore, we also need to add on a scaling term for the distance between the buggy and the obstacle, prioritizing closer obstacles.

MATLAB Sim:





Collision Avoidance Algo



determining the gradient:

1) let r = radius of circle created with steering angle δ

2) use circle formula $x^2 + y^2 = r^2$ on each relative landmark to figure out if it's on the path.

3) if it's within the tolerance, consider it in the way

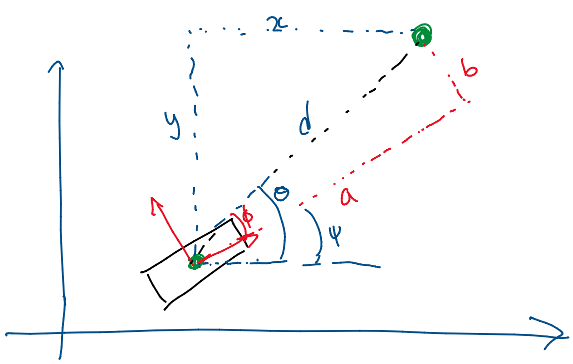
3) if it's within the tolerance, consider it in the way

4) Use the difference as the input to the derivative of the normal fn.

5) Calculate derivative & scale according to distance away (priority to closer objects)

6) add the difference between δ' and δ for the regularizer

Getting relative coordinates:



We know blue, want red

$$\phi = \theta - \psi$$

$$a = d\cos(\phi)$$

$$b = d\sin(\phi)$$