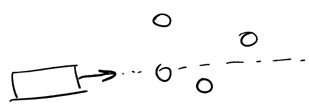
Collision Avoidance documentation

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

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

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 $\frac{1}{2}\|\theta\|^2$

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

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

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

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

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

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

[illegible] 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

MATLAB Sim: