Coordinates And Mobile Robots

Spring 2020

Hasan Poonawala

Robot Configurations

Robot	Configuration Manifold	Coordinates
Point mass	$\mathbb{R}^3$	(x, y, z)
Pan-Tilt Camera	$\mathcal{S}^2$	(θ, ϕ)
Differential-Drive Robot	$\mathbb{R}^2 \times S^1 \quad [SE(2)]$	(x, y, θ)
Elbow Manipulator	$\mathcal{S}^1 \times \mathcal{S}^1 \quad$ [Torus]	(q_1, q_2)
Quadrotor	$\mathbb{R}^3 \times SO(3) \quad [SE(3)]$	(d, R)
Serial-Link Robot Arm	$\left(\mathcal{S}^1\right)^n$	$(q_1, q_2, \ldots, q_n)$

Maps Between Manifolds

Consider a map

$$f\colon M_1 \to M_2,$$

where M_1 and M_2 are any two manifolds.

Maps Between Manifolds

Consider a map

$$f\colon M_1 \to M_2,$$

where M_1 and M_2 are any two manifolds.

M_1	M_2	Properties of f	Interpretation
M	$\mathbb{R}^m$	continuous, bijective, inverse is continuous	Coordinates
$\mathbb{R}$	M	domain is an interval	Trajectory in M, domain is time
M	M	continuous bijection	Coordinate transformation
Eg.: $\mathbb{R}^n$	$\mathbb{R}^n$	$f = R^{-1}(p - d)$	Coordinate transformation

Maps Between Manifolds

Consider a map

$$f : M_1 \to M_2,$$

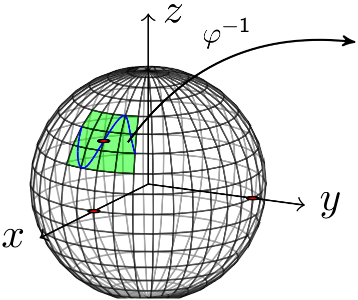
where M_1 and M_2 are any two manifolds.

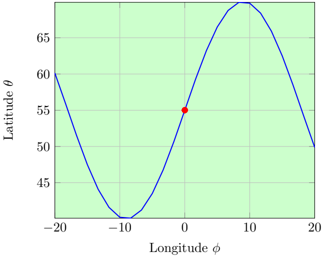
M_1	M_2	Properties of f	Interpretation
M	$\mathbb{R}^m$	continuous, bijective, inverse is continuous	Coordinates
$\mathbb{R}$	M	domain is an interval	Trajectory in M, domain is time
M	M	continuous bijection	Coordinate transformation
Eg.: $\mathbb{R}^n$	$\mathbb{R}^n$	$f = R^{-1}(p - d)$	Coordinate transformation

Example: Consider a circle drawn on the Sphere $\mathcal{S}^2$.

The Map Is Not The Territory

The green square in $\mathbb{R}^2$ becomes a patch on the sphere.

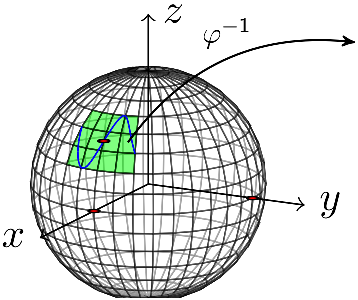


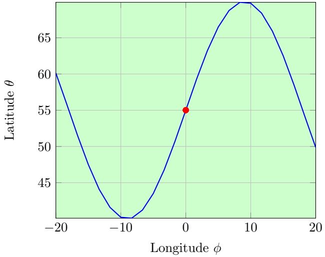


Rigid motions of the green square in $\mathbb{R}^2$ do not lead to rigid motions of the patch

The Map Is Not The Territory

The green square in $\mathbb{R}^2$ becomes a patch on the sphere.





Rigid motions of the green square in $\mathbb{R}^2$ do not lead to rigid motions of the patch

Example: What happens as the square moves upwards in $\mathbb{R}^2$?

We compute using coordinates in $\mathbb{R}^n$. It's our responsibility to handle the manifold structure underlying our robots' configurations.

Over to Youtube

- DARPA Grand Challenge (ca. 2004)
- DARPA Urban Challenge (ca. 2007)
- DARPA Robotics Challenge (ca. 2015)

Mobile Robot Simulation

For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame

Mobile Robot Simulation

For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame
2. A Quadrotor equipped with range-sensing capabilities

Mobile Robot Simulation

For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame
2. A Quadrotor equipped with range-sensing capabilities
3. A mechanism for the quadrotor to detect and locate landmarks within a distance from the quadrotor

Mobile Robot Simulation

For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame
2. A Quadrotor equipped with range-sensing capabilities
3. A mechanism for the quadrotor to detect and locate landmarks within a distance from the quadrotor

Mobile Robot Simulation

For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame
2. A Quadrotor equipped with range-sensing capabilities
3. A mechanism for the quadrotor to detect and locate landmarks within a distance from the quadrotor

Given this simulation, you will also implement algorithms that

1. Solves path-planning problems given initial and goal poses

Mobile Robot Simulation

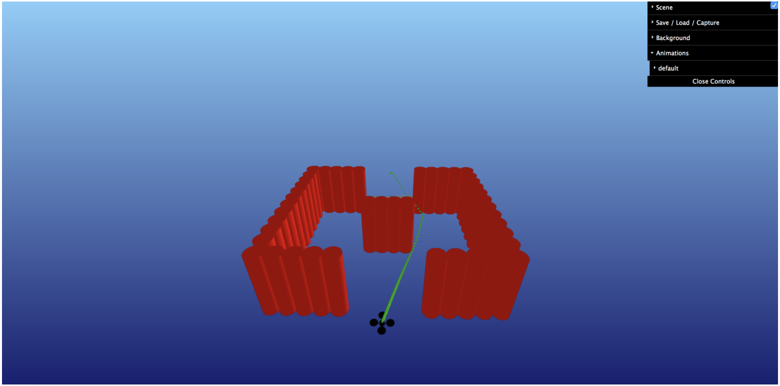
For the next three weeks, your goal is to develop a simulated environment containing

1. A variable set of spherical / cylindrical obstacles/landmarks, each equipped with a coordinate frame
2. A Quadrotor equipped with range-sensing capabilities
3. A mechanism for the quadrotor to detect and locate landmarks within a distance from the quadrotor

Given this simulation, you will also implement algorithms that

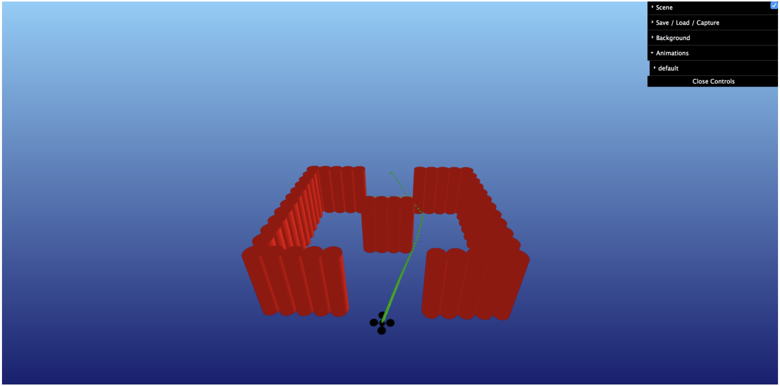
1. Solves path-planning problems given initial and goal poses
2. Implements state-estimation algorithms such as the Extended Kalman Filter

Example



(`Quadrotor_Maze.jl` on Canvas)

Example



(`Quadrotor_Maze.jl` on Canvas)
(This code depicts what I want, but is not a valid solution)

Information & Resources

This is a group assignment: group sizes between 3 and 4 only.

Use `github`, `bitbucket`, or other `git`-compatible online repositories to collaborate and eventually ‘submit’ the assignment.

- Learn version control (`git`) and unix skills online.
- Some code I played with. (Uploaded to Canvas)
- Original papers.

Relevant Planning Topics

1. Graph Search: DFS, BFS, $A^\star$, Djikstra's Algorithm
2. Probabilistic Road Maps
3. Rapidly Exploring Random Trees

Path Planning

A trajectory $q(t)$ is a mapping from time t (subset of $\mathbb{R}$) to the configuration manifold $\mathcal{Q} \ni q$.

Path Planning

A trajectory $q(t)$ is a mapping from time t (subset of $\mathbb{R}$) to the configuration manifold $\mathcal{Q} \ni q$.

A path is the image of that map.

Path Planning

A trajectory $q(t)$ is a mapping from time t (subset of $\mathbb{R}$) to the configuration manifold $\mathcal{Q} \ni q$.

A path is the image of that map.

A trajectory with arbitrary time of travel becomes a path.
For example trajectory

$$q_1(t) = t, q_2(t) = t$$

results in a path that can be expressed in $\mathbb{R}^2$ as $q_1 = q_2$.

Path Planning

A trajectory $q(t)$ is a mapping from time t (subset of $\mathbb{R}$) to the configuration manifold $\mathcal{Q} \ni q$.

A path is the image of that map.

A trajectory with arbitrary time of travel becomes a path.
For example trajectory

$$q_1(t) = t, q_2(t) = t$$

results in a path that can be expressed in $\mathbb{R}^2$ as $q_1 = q_2$.

The path/trajectory planning problems can be cast as an optimization problem.

Path Planning

Suppose we can measure the 'cost' of a trajectory by a function cost.

Path Planning

Suppose we can measure the 'cost' of a trajectory by a function cost.

We want to find an optimal solution $q^*(t)$ of the problem:

$$
\begin{aligned}
\min_{q(t)} \quad & \mathrm{cost}(q(t)) \\
\text{subject to} \quad & \\
& \text{Robot doesn't destroy itself or things} \\
& \text{Other concerns}
\end{aligned}
$$

Path Planning

Suppose we can measure the 'cost' of a trajectory by a function cost.

We want to find an optimal solution $q^*(t)$ of the problem:

$$\begin{array}{ll} \min\limits_{q(t)} & \mathrm{cost}(q(t)) \\ \text{subject to} & \\ & \text{Robot doesn't destroy itself or things} \\ & \text{Other concerns} \end{array}$$

This version of the problem doesn't worry about control.
Out pops $q^*(t)$ and we try and use path following or trajectory tracking controllers.

Workflow

1. Start with finding a path
2. Then, attach time to the path to get a trajectory

Workflow

1. Start with finding a path
2. Then, attach time to the path to get a trajectory

1. Finding paths:
   (a) Figure out the Obstacle-free configuration space (difficult, use over-approximations of robot and obstacles)
   (b) Sample points in free space (easy)
   (c) Connect points in free space (depends)
   (d) Find a sequence of points from start to goal (Search)

Workflow

1. Start with finding a path
2. Then, attach time to the path to get a trajectory

1. Finding paths:
   (a) Figure out the Obstacle-free configuration space (difficult, use over-approximations of robot and obstacles)
   (b) Sample points in free space (easy)
   (c) Connect points in free space (depends)
   (d) Find a sequence of points from start to goal (Search)

   c. How to connect?

   (i) Potential Field + random walk
   (ii) Probabilistic Road Maps
   (iii) Rapidly-exploring Random Trees

Workflow

1. Start with finding a path
2. Then, attach time to the path to get a trajectory

1. Finding paths:
   (a) Figure out the Obstacle-free configuration space (difficult, use over-approximations of robot and obstacles)
   (b) Sample points in free space (easy)
   (c) Connect points in free space (depends)
   (d) Find a sequence of points from start to goal (Search)

   c. How to connect?

   (i) Potential Field + random walk
   (ii) Probabilistic Road Maps
   (iii) Rapidly-exploring Random Trees

   d. How to Search?

Workflow

1. Start with finding a path
2. Then, attach time to the path to get a trajectory

1. Finding paths:
   (a) Figure out the Obstacle-free configuration space (difficult, use over-approximations of robot and obstacles)
   (b) Sample points in free space (easy)
   (c) Connect points in free space (depends)
   (d) Find a sequence of points from start to goal (Search)

   c. How to connect?
      (i) Potential Field + random walk
      (ii) Probabilistic Road Maps
      (iii) Rapidly-exploring Random Trees

   d. How to Search?
2. Polynomial/Parabolic Blends