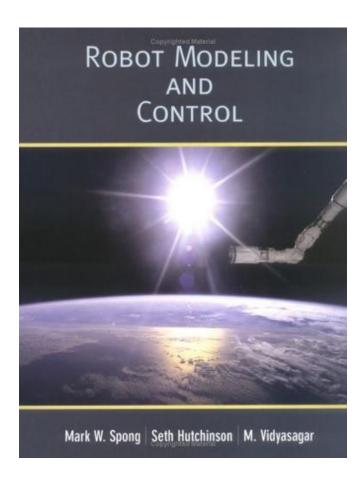
MEAM 520 Lecture 18: Potential Fields

Cynthia Sung, Ph.D.

Mechanical Engineering & Applied Mechanics

University of Pennsylvania

Today: Trajectory Planning with Potential Fields



Chapter 5: Trajectory Planning

• Read Sec. 5.2

Lab 4: Jacobians and Velocity Kinematics

MEAM 520, University of Pennsylvania

October 23, 2020

This lab consists of two portions, with a pre-lab due on Friday, October 30, by midnight (11:59 p.m.) and a lab report due on Friday, November 6, by midnight (11:59 p.m.). Late submissions will be accepted until midnight on Monday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Payza to request an extension if you need one due to a social situation.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, at consult outsides owners such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. When you get stuck, post a question on Pisuza or go to office hours!

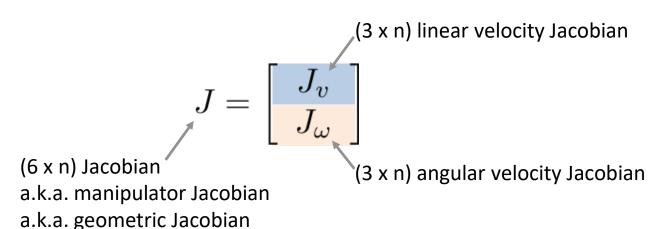
Individual vs. Pair Programming

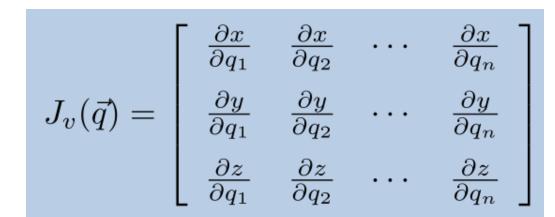
Work closely with your partner throughout the lab, following these guidelines, which were adapted from "All I really needed to know about pair programming I learned in Kindergarten," by Williams and Kessler, Communications of the ACM, May 2000. This article is available on Canavas under Files / Resources.

- . Start with a good attitude, setting aside any skepticism, and expect to jell with your partner
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot)
 while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every 30 minutes, even if one partner is much more experienced than the other. You may want to set a timer to help you remember to switch.
- If you notice an error in the equation or code that your partner is writing, wait until they finish the line to correct them.
- $\bullet\,$ Stay focused and on-task the whole time you are working together.
- Take a break periodically to refresh your perspective
- $\bullet \ \ {\rm Share \ responsibility \ for \ your \ project; \ avoid \ blaming \ either \ partner \ for \ challenges \ you \ run \ into.}$
- Recognize that working in pairs usually takes more time than working alone, but it produces better work, deeper learning, and a more positive experience for the participants.

Lab 4 due Nov. 6

Previously: Manipulator Jacobian





forward velocity kinematics

$$\xi = J(q)\dot{q}$$
 (n x 1) joint velocities (6 x 1) body velocity (6 x n) Jacobian

$$J_{\omega} = \begin{bmatrix} \rho_1 \hat{\mathbf{z}} & \rho_2 \mathbf{R}_1^0 \hat{\mathbf{z}} & \rho_3 \mathbf{R}_2^0 \hat{\mathbf{z}} & \cdots & \rho_n \mathbf{R}_{n-1}^0 \hat{\mathbf{z}} \end{bmatrix}$$

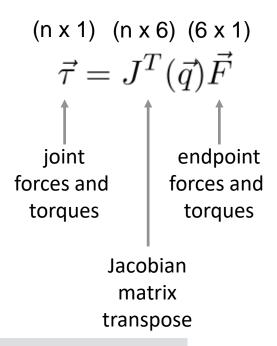
$$ho_i={0
m \ for \ prismation}{1
m \ for \ revolute}$$

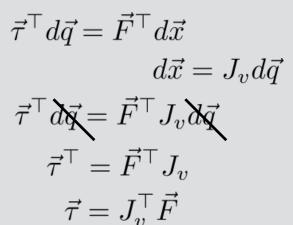
inverse velocity kinematics

$$\dot{q} = J^{-1}\xi$$

Derivation

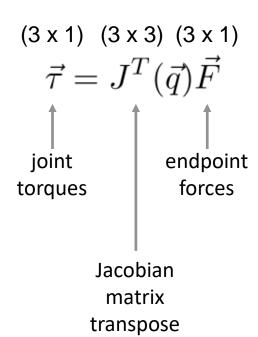
Previously: Static Force/Torque Relationships





Simplest to think about for a 3-DOF robot with all revolute joints.

We want to output a force at the tip.



Previously: Manipulability

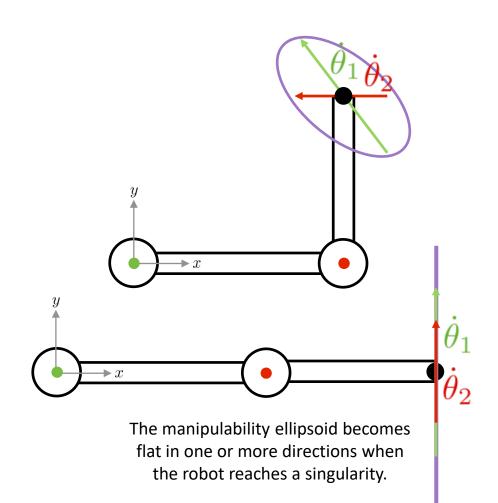
For a specific configuration, the Jacobian scales the input (joint velocities) to the output (body velocity)

$$\xi = J(q)\dot{q}$$

If you put in a joint velocity vector with unit norm, you can calculate in which direction and how fast the robot's end-effector will translate and rotate.

3D manipulability ellipsoids:

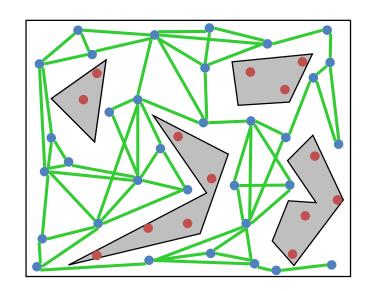
a geometrical representation of all the possible tip velocities (linear or angular) for a normalized joint velocity input.

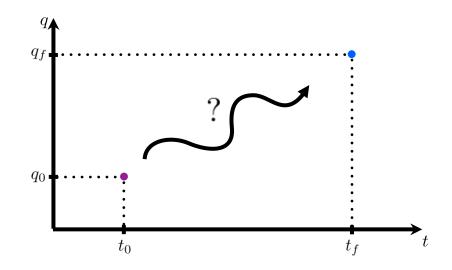


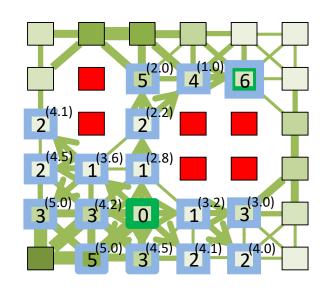
Previously: Path Planning

Planning strategy:

- 1. Convert your free C-space into a graph/roadmap
- 2. Find a path from q_{start} to a node q_a that is in the roadmap
- 3. Find a path from q_{goal} to a node q_b that is in the roadmap
- 4. Search the roadmap for a path from q_a to q_b







Finding the Free C-Space is Hard

Without requiring an explicit representation of the configuration space obstacle or free configuration space,

find a path from a starting configuration $\,q_s\,$

to a final configuration $\,q_f\,$

such that the robot does not collide with any obstacle as it traverses the path.

Artificial Potential Fields

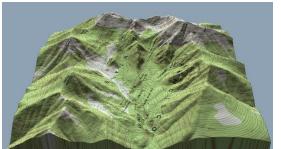
Treat the robot as a point particle in the configuration space.

The robot feels forces from an artificial potential field U defined across its configuration space.

We design U to attract the robot to the desired final configuration and repel it from the boundaries of obstacles.

We want one global minimum at goal with no local minima.

This is often really difficult to construct!



$$U(q) = U_{\text{att}}(q) + U_{\text{rep}}(q)$$

Total

Attraction: field at the goal

Repulsion: potential low potential high potential near obstacles

Optimization problem: find the global minimum in Ustarting from q_s

Use gradient descent

$$\tau(q) = -\nabla U(q)$$

Joint effort

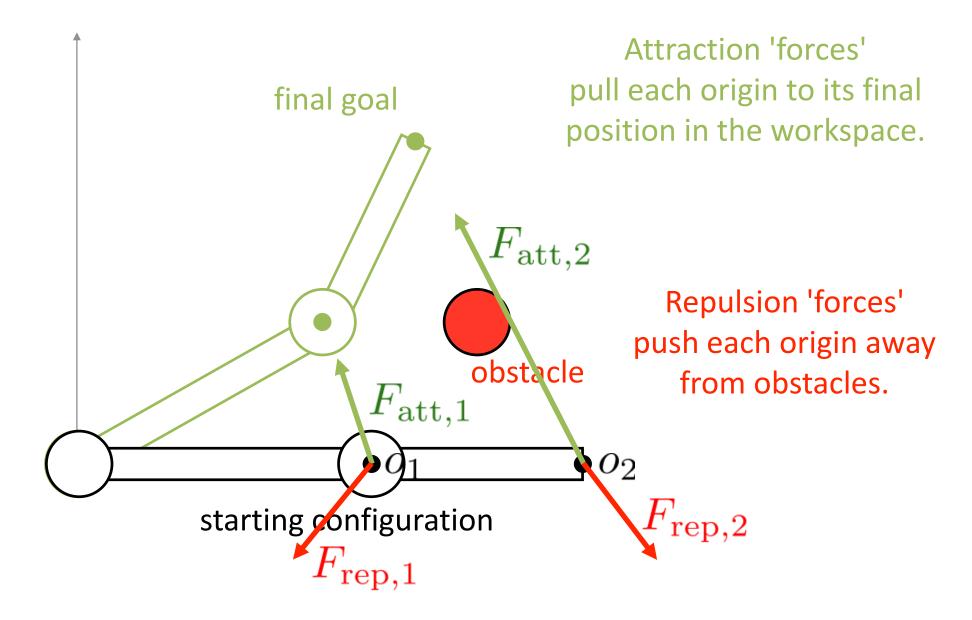
The negative of the gradient of the total potential field: go downhill!

Constructing potential fields directly on the configuration space is difficult because the geometry is complex and you need to know shortest distances to obstacles.

Instead, we define our potential fields directly on the workspace of the robot.

We create a workspace potential field for each DH frame origin (except frame 0) so that it is attracted to its goal location and repelled from obstacles.

Forces on Frame of Origins



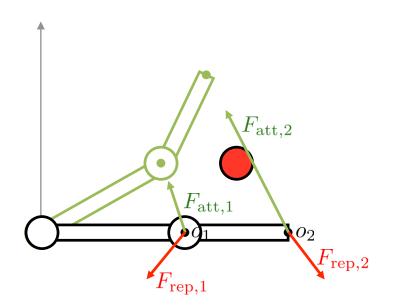
Attractive Field

A simple attractive potential field is the **conic well potential**:

$$U_{\text{att},i}(q) = ||o_i(q) - o_i(q_f)||$$
$$F_{\text{att},i}(q) = -\nabla U_{\text{att},i}(q)$$

$$F_{\text{att},i}(q) = -\frac{(o_i(q) - o_i(q_f))}{||(o_i(q) - o_i(q_f))||}$$

Unit magnitude everywhere, pointing at the final goal.
Discontinuity at the goal can cause instability.



Attractive Field

The most common attractive potential field is the **parabolic well potential**:

$$U_{\mathrm{att},i}(q) = \frac{1}{2}\zeta_i||o_i(q) - o_i(q_f)||^2$$

$$F_{\mathrm{att},i}(q) = -\nabla U_{\mathrm{att},i}(q)$$

$$F_{\mathrm{att},i}(q) = -\zeta_i\left(o_i(q) - o_i(q_f)\right)$$

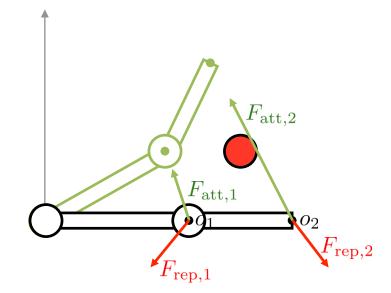
$$\uparrow \qquad \uparrow$$

$$\downarrow \qquad \downarrow$$

$$\downarrow \qquad \uparrow$$

$$\downarrow \qquad \downarrow$$

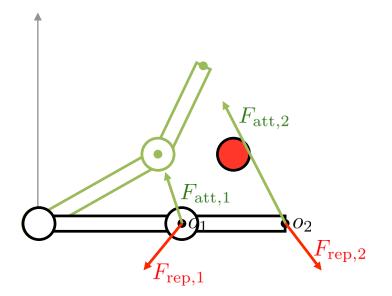
$$\downarrow$$



Attractive Field

The gradient of the **parabolic well potential** is very large far from the goal, which causes very large initial attractive forces.

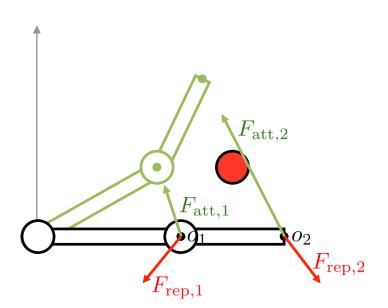
A solution is to use a conic potential far from the goal and transition (smoothly) to a parabolic potential closer to the goal.

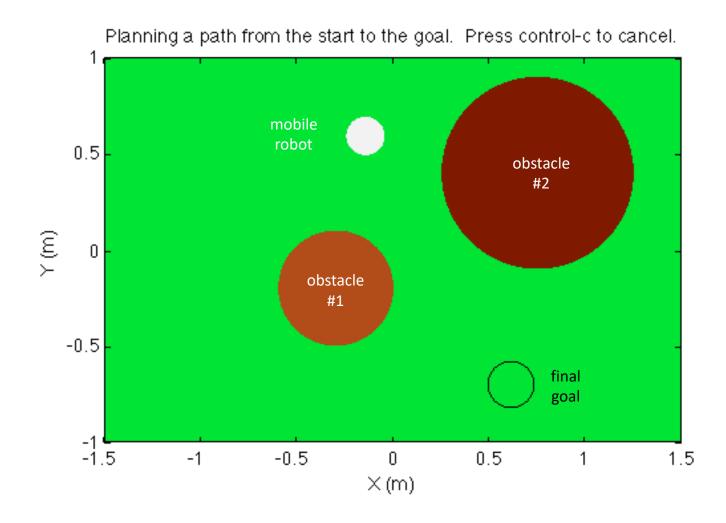


Attractive force has a constant magnitude when more than a given distance away.

Simple Demo

Mobile robot (PP) in the plane.

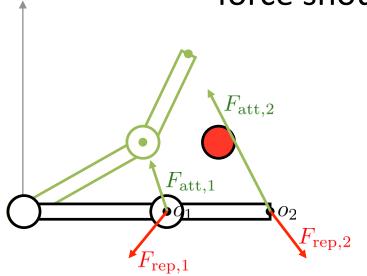




We need to prevent collisions between the robot and all of the obstacles.

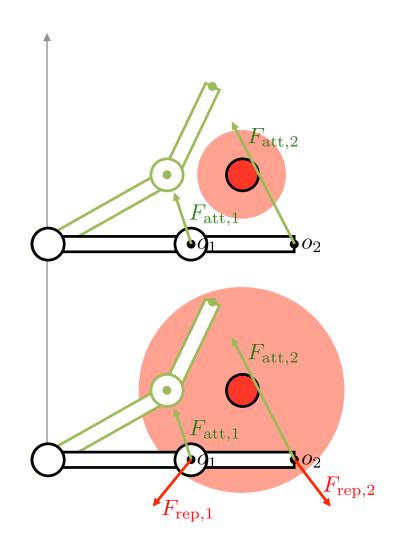
Define a workspace repulsive potential field for each frame origin.

Push very hard on the robot when close: force should go to infinity to prevent collisions.



Don't push on the robot when it is farther than a certain distance from the obstacle.

The most common repulsive potential field is as follows:



$$U_{\mathrm{rep},i}(q) = 0 \,\, \mathrm{when} \,\,
ho_i(q) >
ho_0$$
 shortest distance distance of between o_i and influence of the the obstacle obstacle

$$\begin{aligned} &\text{when } \rho_i(q) \leq \rho_0 \\ &U_{\mathrm{rep},i}(q) = \frac{1}{2} \eta_i \left(\frac{1}{\rho(o_i(q))} - \frac{1}{\rho_0} \right)^2 \\ &\text{repulsive field strength} \end{aligned}$$

What is the force on the border of the region of influence?

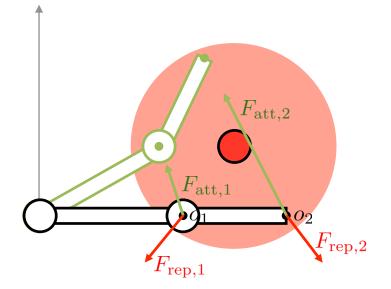
when
$$\rho_i(q) > \rho_0$$

$$F_{\text{rep},i}(q) = 0$$

What is the force on the border of the obstacle? infinity

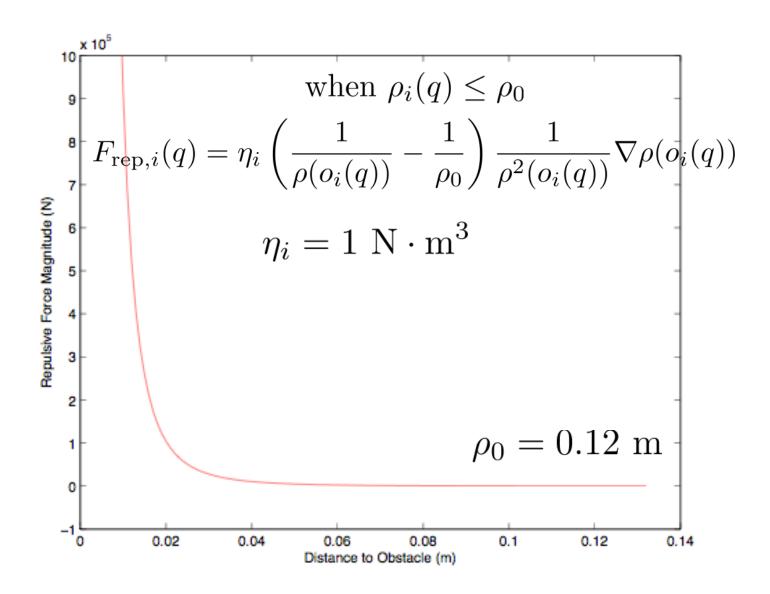
when
$$\rho_i(q) \leq \rho_0$$

$$F_{\text{rep},i}(q) = \eta_i \left(\frac{1}{\rho(o_i(q))} - \frac{1}{\rho_0} \right) \frac{1}{\rho^2(o_i(q))} \nabla \rho(o_i(q))$$



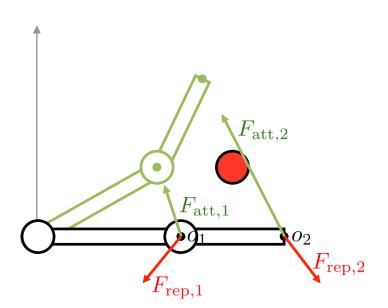
if the obstacle is convex and b is the point on obstacle boundary closest to o_i :

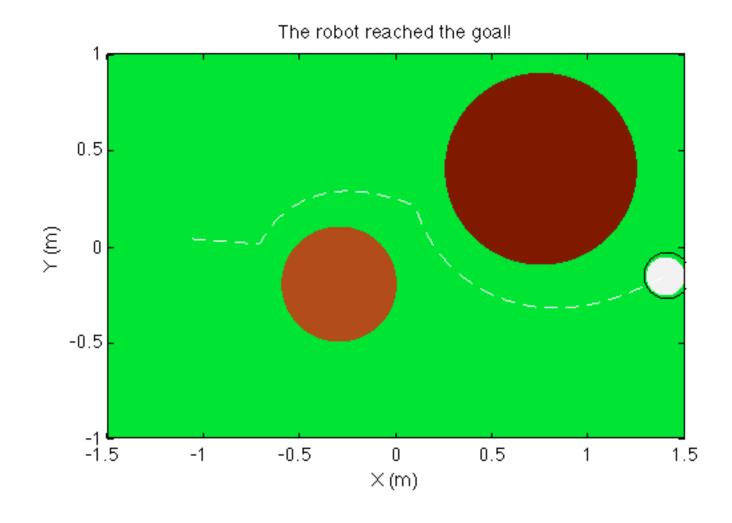
$$\nabla \rho(o_i(q)) = \frac{o_i(q) - b}{||o_i(q) - b||}$$



Simple Demo

Mobile robot (PP) in the plane.





Procedure

while the robot is not at the final goal:

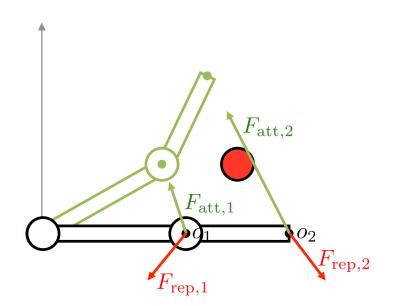
calculate the attractive force on each origin

calculate the repulsive forces on each origin (one force for each obstacle)

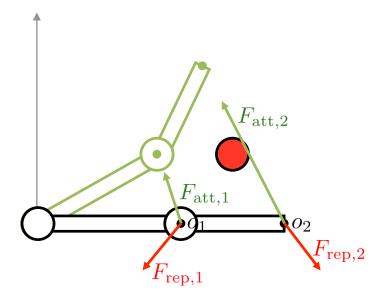
convert each workspace force to the equivalent joint-space efforts (torques) using J_v^T

sum all the joint efforts together

take a fixed-magnitude step in joint space in direction of joint efforts to obtain new joint values check for collisions and goal



Procedure



fixed-size

1.
$$q^0 \leftarrow q_s, i \leftarrow 0$$
 step in joint

2. IF
$$||q^{i} - q_{f}|| > \epsilon$$

$$q_{s}, t < 0 \text{ step in joint}$$

$$||q^{i} - q_{f}|| > \epsilon$$

$$q^{i+1} \leftarrow q^{i} + \alpha^{i} \frac{\tau(q^{i}) \text{ efforts}}{||\tau(q^{i})||}$$

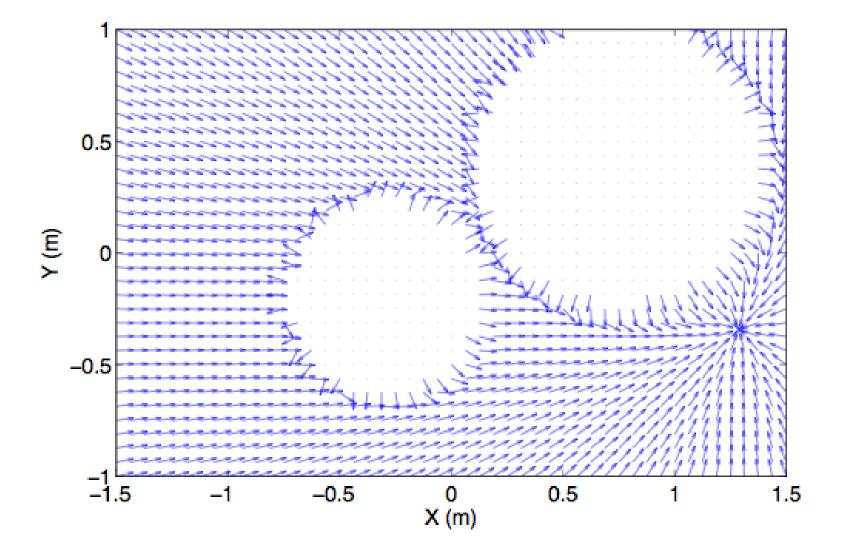
$$i \leftarrow i + 1$$

ELSE return
$$\langle q^0, q^1, \dots, q^i \rangle$$

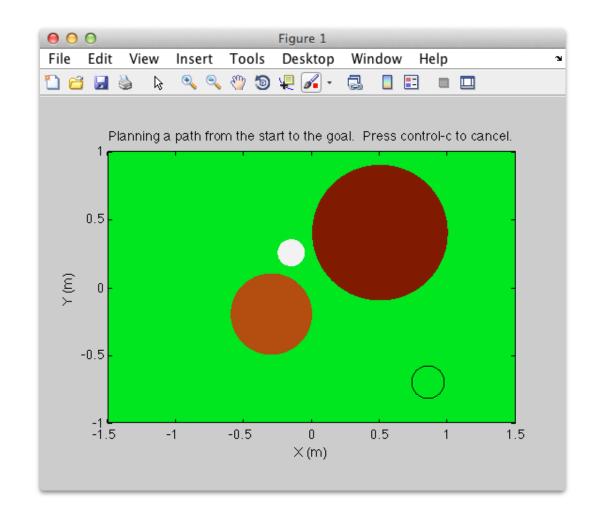
3. GO TO 2

```
116
        % Move the robot while it is more than epsilon from the goal.
117 -
       while (norm(probot-pf) > epsilon)
118
119
            % Calculate the attractive force on the robot. We are using a
120
            % parabolic well potential, which always pulls the robot to the
            % goal with a magnitude that is linearly related to the distance
121
122
            % between the robot and the final position.
            Fa = -zeta * (probot - pf); Calculate attractive force
123 -
124
125
            % Convert attractive force to joint-space efforts. In this case,
126
            % they will be the same because Jv is the identity for a PP robot
127
            % with orthogonal joints.
                                        Calculate joint efforts needed to
128 -
            Jv = eye(2);
                                        create the attractive force
            taua = Jv'*Fa:
129 -
130
131
            % Calculate the repulsive forces on the robot.
132
            % Calculate distance between the robot and obstacle 1.
133
            dobs1 = norm(probot - pobs1) - rrobot - robs1;
134 -
135
            if (dobs1 > rho1)
136 -
137
                % The robot is outside this obstacle's region of influence, so
138
                % the force is zero.
139 -
                Fr1 = [0; 0];
                                Calculate repulsive force from obstacle 1
140 -
            else
141
                % The robot is inside this obstacle's region of influence, so
                % the repulsive force must push the robot away.
142
                Fr1 = eta1 * ((1/dobs1) - (1/rho1)) * (1 / dobs1^2) * ...
143 -
144
                    (probot - pobs1)/norm(probot-pobs1);
145 -
            end
```

```
147
            % Calculate distance between the robot and obstacle 2.
            dobs2 = norm(probot - pobs2) - rrobot - robs2;
148 -
149
            if (dobs2 > rho2) Calculate repulsive force from obstacle 2
150 -
151
                % The robot is outside this obstacle's region of influence, so
152
                % the force is zero.
153 -
                Fr2 = [0: 0]:
154 -
            else
155
                % The robot is inside this obstacle's region of influence, so
156
                % the repulsive force must push the robot away.
157 -
                 Fr2 = eta2 * ((1/dobs2) - (1/rho2)) * (1 / dobs2^2) * (probot - p)
158 -
            end
159
            % Convert repulsive force to joint-space efforts. In this case,
160
            % they will be the same because Jv is the identity for a PP robot
161
            % with orthogonal joints.
162
                                          Calculate joint efforts needed to
163 -
            taur1 = Jv'*Fr1:
                                          create the two repulsive forces
            taur2 = Jv'*Fr2:
164 -
165
166
            % Sum the torques due to the attractive and repulsive forces
167
            % together. This is actually forces for our robot, but we are
168
            % following SHV naming conventions.
            tau = taua + taur1 + taur2; Sum all the joint efforts together
169 -
170
171
            % Calculate the change in position as a scaled version of the net
172
            % torque.
            probot = probot + alpha * tau / norm(tau);
173 -
                                                           Calculate a new pose for the robot exactly alpha
174
                                                           away in joint space in the direction of the summed
175
            % Store this robot position.
                                                          ioint effort.
            probothistorv(:,end+1) = probot;
176 -
```

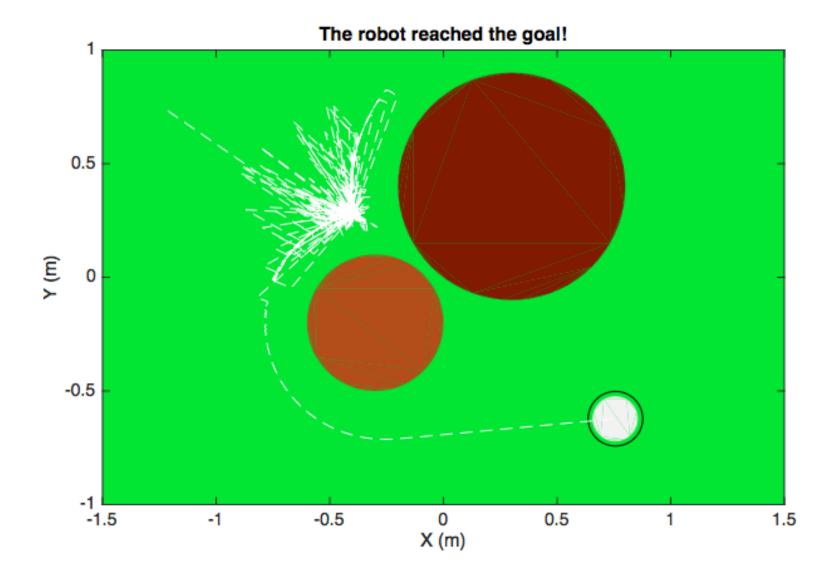


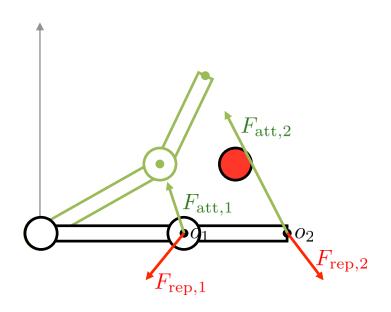
The robot can get stuck at a local minimum.



SHV 5.3 explains how to escape local minima: detect lack of motion and do a random walk.

```
189
            % Check for local minima if requested.
190 -
            if (checkForLocalMinima)
191
               % Get the size of the robot's history; we can't check three
192
193
               % steps back in the robot's history if it doesn't yet have
194
               % four steps.
195 -
                s = size(probothistory);
196
197
               % If there are more than four elements in the robot history.
198 -
                if (s(2) > 4)
199
                    % Check the current position against each of the last
200
                    % three positions, and compare the distance to the
201
                    % threshold epsilonm.
202 -
                    if (((norm(probot-probothistory(:,end-1)) < epsilonm) && ...</pre>
                            ((norm(probot-probothistory(:,end-2)) < epsilonm))) && ...
203
204
                            (norm(probot-probothistory(:,end-3)) < epsilonm))</pre>
205
                        % The robot is probably at a local minimum.
206 -
                        disp('The robot is at a local minimum. Do a random walk.')
207
208
                        % Remember where the robot is now.
209 -
                        localmin(1) = probot(1);
                        localmin(2) = probot(2);
210 -
211
212
                        % Move the robot to the middle of one of the
213
                        % obstacles so that we can choose a new non-colliding
214
                        % position as often as is needed.
215 -
                        probot(1) = xobs1:
216 -
                        probot(2) = yobs1;
217
218
                        % Check if the proposed new position is inside either
219
                        % of the obstacles. If it is, this is not a good
220
                        % position, so we should pick a new one.
                        while ((norm(pobs1 - probot) < (rrobot + robs1)) || ...</pre>
221 -
222
                                (norm(pobs2 - probot) < (rrobot + robs2)))</pre>
223
                            % Perturb the robot's position by a fixed amount
224
                            % in a random direction. This is like a one-step
225
                            % random walk, but in a random direction rather
226
                            % than always positive or negative on each
227
                            % coordinate.
228 -
                            theta = 2*pi*rand(1);
229 -
                            probot(1) = localmin(1) + v * cos(theta);
230 -
                            probot(2) = localmin(2) + v * sin(theta);
231 -
                        end
232
233
                        % Store this new position in the robobot's history.
234 -
                        probothistory(:,end+1) = probot;
235
236
                        % Increase the size of the random walk over time to
237
                        % try to escape bigger local minima.
238 -
                        v = v * 1.02;
239 -
                    end
240 -
                end
241 -
            end
```





How do we apply the attractive and repulsive forces to a robotic arm?

You need to formulate the linear velocity Jacobian *for each origin*.

$$\vec{ au}_1 = J_{v,1}^{ op} \vec{F}_1 \qquad \vec{ au}_2 = J_{v,2}^{ op} \vec{F}_2$$

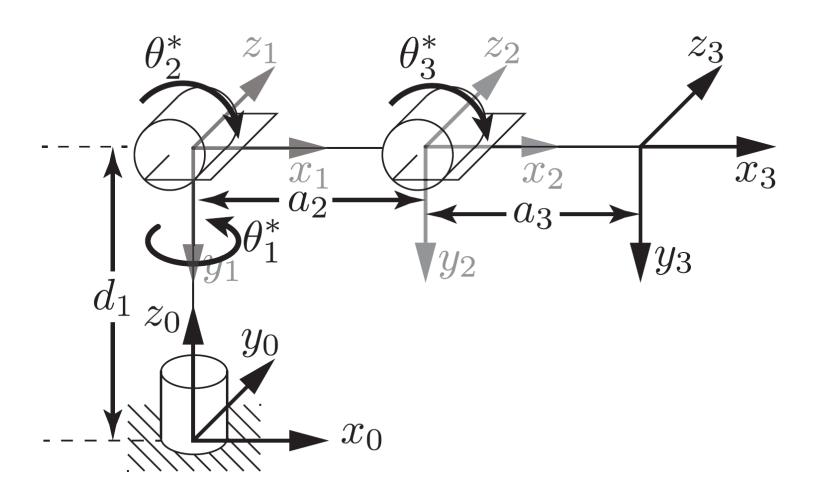
Columns for joints after that origin will be zeros.

Sum the 'forces' on that origin and determine the joint velocities.

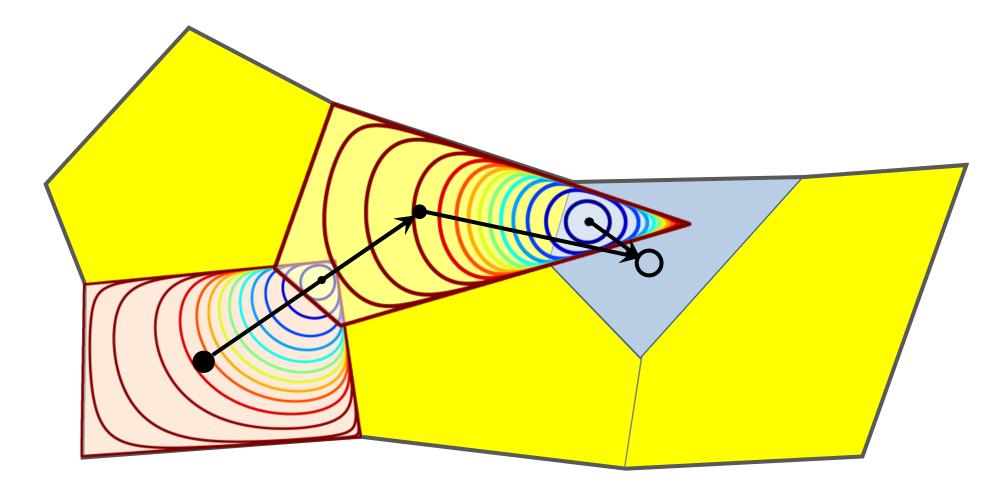
Note that this approach may still allow collisions along the links: you would need to add a floating repulsion point to guarantee no collisions.

This is not necessary for Lab 5

Example: Articulated RRR Arm



Combining planning methods



Next Time: Paper Reading

2007 IEEE International Conference on Robotics and Automation Roma, Italy, 10-14 April 2007

FrE4.2

Grasping POMDPs

Kaijen Hsiao and Leslie Pack Kaelbling and Tomás Lozano-Pérez

I. INTRODUCTION

in order to manipulate objects, for example, these strategies the objects in the world is not exactly known at the outset be exactly known. In such cases, traditional open-loop plans most likely to be encountered.

even extended with simple feedback) are not reliable.

It is useful to distinguish between modes of uncertainty situations with unmodelable uncertainty, such as insertion have large effects on the necessary directions of applied forces, and the available sensors can gain little or no information about those surfaces. When the uncertainty is unmodelable, we must fall back to strategies such as "wigeling" the key, which are highly successful without ever

Modelable uncertainty, on the other hand, typically occurs a good high-level model of the situation, but significant ining uncertainty about the pose or shape of the mug. Based on sensor feedback, it can reason about whether the and choose actions that will both gather more information about the situation and make progress toward a desired grasp with a multi-fingered hand.

This research was supported in part by DARPA IPTO Contract FAST50-05-2-05-9, "Effective Rayesian Transfer Learning", and in part by the Singapor-MIT Alliance agreement dated 1140/98. Computer Science and Artificial Intelligence Laboratory, Manuschusetts Institute of Technology, 32 Vassar Sixest, Cambridge, MA 02139 (k)histion, 1 pk, 1-12] Recards 11.mit. edu

1-4244-0602-1/07/\$20.00 @2007 IEEE.

Abstract—We provide a method for planning under un-certainty for robotic manipulation by partitioning the con-figuration space into seed or regions that a reduced under formation are missed to see the reduced under the seed of the reduced under the reduced under the seed of the reduced under the reduced under the reduced under the which can be solved to yield optimal control policies under uncertainty. We demonstrate the approach on simple grasping executable solution: Can construct the jobly robotal, efficiently and the seed of the configurations consistant of the reduced to the variety of all possible configurations consistant of the variety of all possible configurations consistant.

maning motions for robots with many degrees of freedom through free space [10], [9], [13]. These methods enable robots to more through complex environments, as long as they are set in contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects to the manifest of the contact with the objects of the contact with the objects of the contact with the objects of the contact with the contact with the contact with the objects of the contact with the wever, as soon as the robot needs to contact the world, succeed in every possible situation. We can choose plans that optimize a variety of different objective functions involving do not apply. The fundamental problem with planning for motion in contact is that the configuration of the robot and of execution, and, given the resolution of sensors, it cannot savings through a focus on the parts of the space that are

By building an abstraction of the underlying continuous It is useful to distinguish between modes of uncertainty that can be effectively modeled, and those that cannot. In tional simplification, making it feasible to compute solutions of keys into locks, very fine-grained details of the surfaces to real problems. Concretely, we will use methods of model minimization to create an abstract model of the underlying actions under uncertainty as a partially observable Markov decision process [25].

II. BACKGROUND AND APPROACH

The approach we outline here applies to any domain in at a coarser scale. In attempting to pick up a mug, for example, a robot with vision or range sensing might have and there is non-trivial uncertainty in the configuration. In robot arm and hand performing pick-and-place operations We assume that the robot's position in the global frame is reasonably well known, but that there is some uncertainty about the relative pose and/or shape of the object to be manipulated. Additionally, we assume that there are tactile and/or force sensors on the robot that will enable it to perform compliant motions and to reasonably reliably detec when it makes or loses contacts. We frame this problem primarily as a planning problem. That is, we assume that a reasonably accurate model of the task dynamics and sensors is known, and that the principal uncertainty is in the configuration of the robot and the state of the objects in

Hsiao, K and Kaelbling, LP, and Lozano-Pérez, T. "Grasping POMDPs." ICRA

Lab 4: Jacobians and Velocity Kinematics

MEAM 520. University of Pennsylvania

This lab consists of two portions, with a pre-lab due on Friday, October 30, by midnight (11:59 p.m.) and a lab report due on Friday, November 6, by midnight (11:59 p.m.). Late submissions will be accepted until midnight on Monday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Pizzza to request an extension if you need one due to a special situation.

You may talk with other students about this assignment, ask the teaching team questions, use a calculate and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions cted of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct When you get stuck, post a question on Piazza or go to office hours!

Individual vs. Pair Programming

Work closely with your partner throughout the lab. following these guidelines, which were adapted from Wall I really needed to know about pair programming I learned in Kindergarten," by Williams and Kessler Communications of the ACM, May 2000. This article is available on Canvas under Files / Resources.

- · Start with a good attitude, setting aside any skepticism, and expect to jell with your partne
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot) while the other is continuously reviewing the work (thinking and making suggestion
- . Change driving reviewing roles at least every 30 minutes, even if one partner is much more experienced than the other. You may want to set a timer to help you remember to switch

- · Share responsibility for your project; avoid blaming either partner for challenges you run into
- Recognize that working in pairs usually takes more time than working alone, but it produces better

Lab 4 due Nov. 6