HOW TO PLAN AMIDST UNCERTAINTY

Lecture 18: Markov Decision Processes and Bellman Value Iteration
Today’s Listening Outline

- Reviewing to motivate expansion
- Working a Motivating Example
- Generalizing the Skill
- Practicing Solution Transfer + Discussing with Neighbors
- Inquiring into Extension
- Bellman Backups
- Summary of Discoveries

Rising Intensity
Rising Cognitive Engagement
we planned a trajectory of states \( x(t) \) over time

we planned a motion of states \( x(t) \) over time driven by our limited controls \( u(t) \) and constrained space \( C_{\text{free}} \)

we have full control of our state

we have limited control of our state
**Motion Planning Algorithms**

- **Reachability Tree**
- **Random Trees (RRT)**
- **Search on Visibility Graphs**

**Controlling the Continuous Dynamics**
\[
\frac{dx(t)}{dt} = f(x(t), u(t))
\]

**Controlling the Discrete Dynamics**
\[
x_{t+1} = f(x_t, u_t)
\]
we controlled future states through our dynamics $f(x_t, u_t)$

we have certainty about what will happen from our actions

we control future states’ distribution through our dynamics $P(x_{t+1} | x_t, u_t)$

we are uncertain about what will happen next
Use *Bellman Backups on MDPs* to plan amidst uncertainty*

Class Outline

5m • Review: Motion Planning collides with Accidents

20m • Motivating Example: Avoiding plane crashes

15m • Learning a Skill: Describing uncertain tasks with MDPs

20m • Exercise: Analyzing self-driving cars with MDPs

15m • Expanding our skill: Strategizing valuables moves via Bellman Backups

5m • Recap: Connecting it all together for your takeaway
Today’s Listening Outline

0min

Reviewing to motivate expansion

Rising Intensity

Reviewing to motivate expansion

Working a Motivating Example

Generalizing the Skill

Practicing Solution Transfer

+ Discussing with Neighbors

Rising Cognitive Engagement

Inquiring into Extension

Bellman Backups

Summary of Discoveries

20min

40min

60min

80min
Use **Bellman Backups on MDPs** to plan amidst uncertainty

Today's Learning Goal:

- Review: Motion Planning collides with Accidents
- Motivating Example: Avoiding plane crashes
- Learning a Skill: Describing uncertain tasks with MDPs
- Exercise: Analyzing self-driving cars with MDPs
- Expanding our skill: Strategizing valuables moves via Bellman Backups
- Recap: Connecting it all together for your takeaway
Traffic Alert and Collision Avoidance System (TCAS)

Sensor Measurements

Resolution Advisory

IF (ITF.A LT G.ZTHR)
THEN IF(ABS(ITF.VMD) LT G.ZTHR)
THEN SET ZHIT;
ELSE CLEAR ZHIT;
ELSE IF (ITF.ADOT GE P.ZDTHR)
THEN CLEAR ZHIT
ELSE
ITF.TAUV = -ITF.A/ITF.ADOT;
IF (ITF.TAUV LT TVTHR AND
((ABS(ITF.VMD) LT G.ZTHR) OR
(ITF.TAUV LT ITF.TRTRU))
THEN SET ZHIT
ELSE CLEAR ZHIT
IF (ZHIT EQ $TRUE AND
ABS(ITF.ZDINT) GT P.MAXZDINT
THEN CLEAR ZHIT

Surveillance
Advisory Logic
Display

Slide Credit: Mykel Kochenderfer
PROCESS Reversal_modeling:
  . Default modeled separation for current RA is 0 if current RA is numeric.
  . Set own altitude and own rate to own tracked altitude and own tracked rate.
  . IF (even does not follow this RA):
    . THEN Model separation achieved remaining RA not followed:
      . THEN CLEAR flag indicating the sense of the RA after a reversal.
      . ELSE SET flag indicating the sense of the RA after a reversal.
      . IF (modeled separation achieved by continuing current RA greater than 1.2
        . THEN CLEAR reversed flag in ITT.
        . ELSE Begin own is assumed to follow an RA:
          . THEN model response to current RA:
            . model maximum displayable rate for climb if current rate exceeds
              minimum displayable rate or minimum displayable rate for descent if
              current rate is less than minimum displayable rate.
            . IF (tracked response lies modeled response to RA direction AND
              time since RA less than a parameter time AND
              own’s rate has not changed by more than a parameter rate since
              the RA was first issued)
              THEN set own altitude and own rate to modeled altitude and rate
              for use in reversal modeling.
              . Model separation achieved by continuing current RA.
              . Set delay time to generate of pilot delay time remaining for last advisory to a
                new target, and the pilot prompt reaction time.
              . IF (considering a reversal from a descend RA to a climb RA):
                . THEN set own goal rate to zero or own tracked rate (or minimum
                  displayable rate, whichever is less) and own climb rate.
                . ELSE (own too close to ground to descend)
                  THEN set own goal rate to zero.
                  . ELSE set own goal rate to lower or own tracked rate (or minimum
                    displayable rate, whichever is greater) and own descent rate.
                  IF (vertical close, low VMD geometry was not the reason for this reversal):
                    . THEN if (modeled crossing OR (model crossing AND own crossing
                      from above OR at same rate and own modeled rate are opposite in sign)
                      . THEN use rate to bound to modeled rate.
                      . ELSE use rate to bound to forward rate.
                      IF (own’s tracked vertical rate to modeled rate):
                        . THEN model separator:
                          . adjust, goal rate, own altitude, own rate, acceleration response; sense other
                          reversed. if modeled rate exceeds modeled rate (1 ITT entry)
                          . THEN CLEAR reversed flag in ITT.
                          . ELSE Begin own is assumed to follow an RA:
                          . THEN Reversal_modeling.
                          END Reversal_modeling;

RESOLUTION HIGH LEVEL LOGIC
6-P22

Traffic Alert and Collision Avoidance System (TCAS) RTCA DO-185B (1799 total pages / 440 is pseudocode)
Why is it hard?

State Uncertainty
Imperfect sensor information leads to uncertainty in position and velocity of aircraft

Dynamic Uncertainty
Variability in pilot behavior makes it difficult to predict future trajectories of aircraft

Multiple Objectives
System must carefully balance both safety and operational considerations

Slide Credit: Mykel Kochenderfer
# Simplified MDP

## State space
- Relative altitude
- Own vertical rate
- Intruder vertical rate
- Time to lateral NMAC
- State of advisory

## Action space
- Clear of conflict
- Climb > 1500 ft/min
- Climb > 2500 ft/min
- Descend > 1500 ft/min
- Descend > 2500 ft/min

## Dynamic model
- Head-on, constant closure
- Random vertical acceleration
- Pilot response delay (5 s)
- Pilot response strength (1/4 g)
- State of advisory

## Reward model
- NMAC (-1)
- Alert (-0.01)
- Reversal (-0.01)
- Strengthen (-0.009)
- Clear of conflict (0.0001)
Simplified MDP

State space
- Relative altitude
- Own vertical rate
- Intruder vertical rate
- Time to lateral NMAC
- State of advisory

Dynamic model
- Head-on, constant closure
- Random vertical acceleration
- Pilot response delay (5 s)
- Pilot response strength (1/4 g)
- State of advisory

Action space
- Clear of conflict
- Climb > 1500 ft/min
- Climb > 2500 ft/min
- Descend > 1500 ft/min
- Descend > 2500 ft/min

Reward model
- NMAC (-1)
- Alert (-0.01)
- Reversal (-0.01)
- Strengthen (-0.009)
- Clear of conflict (0.0001)

Slide Credit: Mykel Kochenderfer
Optimized Logic
Both Own and Intruder Level

<table>
<thead>
<tr>
<th>Metric</th>
<th>TCAS</th>
<th>ACAS X</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMACs</td>
<td>169</td>
<td>3</td>
</tr>
<tr>
<td>Alerts</td>
<td>994,317</td>
<td>690,406</td>
</tr>
<tr>
<td>Strengthens</td>
<td>40,470</td>
<td>92,946</td>
</tr>
<tr>
<td>Reversals</td>
<td>197,315</td>
<td>9,569</td>
</tr>
</tbody>
</table>

Slide Credit: Mykel Kochenderfer
Use *Bellman Backups on MDPs* to plan amidst uncertainty

Today’s Learning Goal:

- Review: Motion Planning collides with Accidents
- Motivating Example: Avoiding plane crashes
- Learning a Skill: Describing uncertain tasks with MDPs
- Exercise: Analyzing self-driving cars with MDPs
- Expanding our skill: Strategizing valuables moves via Bellman Backups
- Recap: Connecting it all together for your takeaway
Temporal Models: Markov Chains

\[ X_0 \rightarrow X_1 \rightarrow X_2 \rightarrow \ldots \]

\( X_t \) is state at time \( t \)

Markov Property:
- future states only rely on current state

\[
\begin{array}{c|cc}
X & S & R \\
\hline
S & .9 & 1 \\
R & .2 & .8 \\
\end{array}
\]
Markov Assumptions and Common Violations

Markov Assumption postulates that past and future data are independent if you know the current state.

What are some common violations?

• Unmodeled dynamics in the environment not included in state
  • E.g., moving people and their effects on sensor measurements in localization

• Inaccuracies in the probabilistic model
  • E.g., error in the map of a localizing agent or incorrect model dynamics

• Approximation errors when using approximate representations
  • E.g., discretization errors from grids, Gaussian assumptions

• Variables in control scheme that influence multiple controls
  • E.g., the goal or target location will influence an entire sequence of control commands
Uncertainty in Robot Motion

- **Markov Decision Processes (MDPs)** model the robot and environment assuming full observability
  - $P(z|x)$: deterministic and bijective
  - $P(x'|x,u)$: may be nondeterministic
  - Must incorporate uncertainty into the planner and generate actions for a range of possible states

- A **policy** for action selection is defined for all states the robot may encounter
Robot Values

• Robot actions are driven by goals
  • E.g., reach configuration, balance for as long as possible

• Often, we want to reach goal while optimizing some cost
  • E.g., minimize time / energy consumption, obstacle avoidance

• We express both costs and goals in a single function, called the payoff function

\[
\text{ex} \quad r(x,u) = \begin{cases} 
100, & \text{if reach goal} \\
-1, & \text{otherwise}
\end{cases}
\]

→ evaluated at each time step
Policies

• We want to devise a scheme that generates actions to optimize the future payoff *in expectation*

• Policy: $\pi : x_t \rightarrow u_t$
  • Maps states to actions
  • Can be low-level reactive algorithm or a long-term, high-level planner
  • May or may not be deterministic

• Typically, we want a policy that optimizes future payoff, considering optimal actions over a planning (time) horizon
Expected Cumulative Payoff

$$R_T = \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^t r_{t+1} \right]$$

deck of cards

discount factor \( \gamma \in [0, 1] \)

1) Greedy case: \( T = 1 \)
   \( \rightarrow \) opt next payoff

2) Finite Horizon: \( 1 \leq T < \infty \), \( (\gamma \leq 1) \)
   \( \rightarrow \) opt \( R_T \) for set time window

3) Infinite Horizon: \( T = \infty \), \( (\gamma < 1) \)
   \( \rightarrow \) opt \( R_\infty \) for all time

if \( |r| < r_{\max} \), discounting guarantees \( R_\infty \) is finite:

$$R_\infty \leq r_{\max} + \gamma r_{\max} + \gamma^2 r_{\max} + \ldots = \frac{r_{\max}}{1-\gamma}$$
<table>
<thead>
<tr>
<th><strong>State space</strong></th>
<th><strong>Action space</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Relative altitude</td>
<td>• Clear of conflict</td>
</tr>
<tr>
<td>• Own vertical rate</td>
<td>• Climb &gt; 1500 ft/min</td>
</tr>
<tr>
<td>• Intruder vertical rate</td>
<td>• Climb &gt; 2500 ft/min</td>
</tr>
<tr>
<td>• Time to lateral NMAC</td>
<td>• Descend &gt; 1500 ft/min</td>
</tr>
<tr>
<td>• State of advisory</td>
<td>• Descend &gt; 2500 ft/min</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Dynamic model</strong></th>
<th><strong>Reward model</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Head-on, constant closure</td>
<td>• NMAC (-1)</td>
</tr>
<tr>
<td>• Random vertical acceleration</td>
<td>• Alert (-0.01)</td>
</tr>
<tr>
<td>• Pilot response delay (5 s)</td>
<td>• Reversal (-0.01)</td>
</tr>
<tr>
<td>• Pilot response strength (1/4 g)</td>
<td>• Strengthen (-0.009)</td>
</tr>
<tr>
<td>• State of advisory</td>
<td>• Clear of conflict (0.0001)</td>
</tr>
<tr>
<td>State space</td>
<td>Action space</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td><img src="image1" alt="State space diagram" /></td>
<td><img src="image2" alt="Action space diagram" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dynamic model</th>
<th>Reward model</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Dynamic model" /></td>
<td>$R^T = \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^t r_{t+1} \right]$</td>
</tr>
</tbody>
</table>

- $\gamma$ is the discount factor.
<table>
<thead>
<tr>
<th>State space</th>
<th>Action space</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In decision-making papers:</strong></td>
<td><strong>In decision-making papers:</strong></td>
</tr>
<tr>
<td>$X$</td>
<td>$U$</td>
</tr>
<tr>
<td>$S$</td>
<td>$A$</td>
</tr>
<tr>
<td><strong>Dynamic model</strong></td>
<td><strong>Reward model</strong></td>
</tr>
<tr>
<td>$T$</td>
<td>$J$</td>
</tr>
</tbody>
</table>

In reinforcement learning papers:

- $S$ and $A$ represent state space and action space, respectively.
- $R$ represents the reward.

**Dynamic model**:

- $T$ represents the transition matrix.

**Reward model**:

- $J$ represents the reward function.

In control theory:

- $J = E \left[ \sum_{t=0}^{T} \gamma^t r_t \right]$ when the discount factor $\gamma$ is most of the time.
- $J = E \left[ \sum_{t=0}^{T} \gamma^t r_t \right]$ when negative.
A Markov Decision Process (MDP) is a combo of five items:

\((S, A, T, \gamma, R)\)

Note the parallels to Lecture 14’s deterministic and unrewarding:

\((X, U, f(\cdot, \cdot))\)
Summary

- Discussed decision-making (planning) schemes and how they fit into the robotics pipeline.
- Defined the MDP model for decision-making, including robot goals, costs, payoff, and policies.
- Defined Expected Cumulative Payoff, which plays a key role in optimizing actions over planning horizons.
- Next: Look at how to algorithmically determine the policy that optimizes this pay off.
EXERCISE CHALLENGE:
What are some states we need to plan around?
What are actions can the car take?
What are some competing desirable goals?
Use **Bellman Backups on MDPs** to plan amidst uncertainty

Today's Learning Goal:

- Review: Motion Planning collides with Accidents
- Motivating Example: Avoiding plane crashes
- Learning a Skill: Describing uncertain tasks with MDPs
- Exercise: Analyzing self-driving cars with MDPs
- Expanding our skill: Strategizing valuables moves via Bellman Backups
- Recap: Connecting it all together for your takeaway
Who is Richard Bellman?

- Richard Bellman (August 26, 1920 – March 19, 1984) was an American applied mathematician
- Developed the concept of dynamic programming while working at the RAND corporation
- Best known for the **Bellman equation** (a.k.a. dynamic programming equation) is a necessary condition for optimality associated with the mathematical optimization method known as dynamic programming
- **Hamilton–Jacobi–Bellman equation** gives a necessary and sufficient condition for optimality of a control with respect to a loss function
- There is a strong connection to the Hamilton–Jacobi equation from classical physics was first drawn by Rudolf Kalman

Credit: Wikipedia
Markov Decision Processes (MDPs)

\((S, A, T, \pi, R)\)

model determined by itself
MDP Policies

• Policies map states to actions

\[ \pi : X \rightarrow U \]

• We want to find a policy that maximizes future pay off

\[ \text{suppose } T = 1 : \pi_1(x) = \arg\max_u r(x,u) \]

• Generally, we want to find the sequence of actions that optimize the expected cumulative discounted future payoff

• We write the Value Function for given \( x \):

\[ v_1(x) = \gamma \max_u r(x,u) \]
Value Functions

For longer time horizons, we define $V(x)$ recursively:

$$V_1(x) = \gamma \max_u r(x, u)$$

From prev slide:

$$V_1(x) = \gamma \max_u r(x, u)$$

$T = 2$: select action that maximizes sum of $V_i$ and 1-step payoff:

$$\pi_2(x) = \arg\max_u \left[ r(x, u) + \int U_i(x') p(x' \mid u, x) dx' \right]$$

$$V_2(x) = \gamma \max_u \left[ r(x, u) + \int U_i(x') p(x' \mid u, x) dx' \right]$$

For finite $T$:

$$\pi_T(x) = \arg\max_u \left[ r(x, u) + \int U_{T-1}(x') p(x' \mid u, x) dx' \right]$$

$$V_T(x) = \gamma \max_u \left[ r(x, u) + \int U_{T-1}(x') p(x' \mid u, x) dx' \right]$$
Value Functions
Value Functions

• In the infinite time horizon, we tend to reach equilibrium:

\[ V_\infty(x) = \gamma \max_u \left[ r(x, u) + \int V_\infty(x') \rho(x' | x, u) \, dx' \right] \]

• This is the Bellman Equation
  • Satisfying this is necessary and sufficient for an optimal policy
Computing the Value Function

Value Fn approximation: first init some guess for $\hat{V}$

$\hat{V} \leftarrow r_{\text{min}}$

successively update for increasing horizons:

$\hat{V}(x) \leftarrow \gamma \max_u \left[ r(x, u) + \int \hat{V}(x') p(x'|u, x) dx' \right]$

Value iteration converge if $\gamma < 1$

Given $\hat{V}(x)$, policy is found:

$\pi(x) = \arg\max_u \left[ r(x, u) + \int \hat{V}(x') p(x'|u, x) dx' \right]$

Often discrete:

$\pi(x) = \arg\max_u \left[ r(x, u) + \sum_{x'} \hat{V}(x') p(x'|u, x) \right]$
Example

\[ \hat{V}(x_i) = \gamma \max_u \left[ r(x_i, u) + \sum \hat{V}(x_j) p(x_j | u, x_i) \right] \]

vector rep:
\[ \hat{V} = [\hat{V}(x_1) \ \hat{V}(x_2) \ \hat{V}(x_3)] \]

init \[ \hat{V} = [0 \ 0 \ 0 \ 0] \]

iter 1 \[ \hat{V} = [\gamma(R + 0) \ \gamma(0 + 0) \ \gamma(0 + 0)] \]

iter 2 \[ \hat{V} = [\gamma^2 R \ \gamma(0 + \gamma R) \ \gamma(0 + 0)] \]

iter 3 \[ \hat{V} = [\gamma R \ \gamma^2 R \ \gamma^3 R] \]

- \( u_i \in \{1, 2, 3\} \)
- \( P(x_i | u, x) \) is deterministic \( \in \{1, 0, 3\} \)
- \( r(x_i, u) = R \) 
  o.w. 0
- only get payoff if run into door \( \rightarrow \) terminal condition
Grid world example

- States: cells in $10 \times 10$ grid
- Actions: up, down, left, right
- Transition model: 0.7 chance of moving in intended direction, uniform in other directions
- Reward:
  - two states with cost
  - two terminal states with rewards
    - $-1$ for wall crash
- Discount is 0.9
<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.41</td>
<td>0.74</td>
<td>0.96</td>
<td>1.18</td>
<td>1.43</td>
<td>1.71</td>
<td>1.98</td>
<td>2.11</td>
<td>2.39</td>
<td>2.09</td>
</tr>
<tr>
<td>0.74</td>
<td>1.04</td>
<td>1.27</td>
<td>1.52</td>
<td>1.81</td>
<td>2.15</td>
<td>2.47</td>
<td>2.58</td>
<td>3.02</td>
<td>2.69</td>
</tr>
<tr>
<td>0.86</td>
<td>1.18</td>
<td>1.45</td>
<td>1.76</td>
<td>2.15</td>
<td>2.55</td>
<td>2.97</td>
<td>3</td>
<td>3.69</td>
<td>3.32</td>
</tr>
<tr>
<td>0.84</td>
<td>1.11</td>
<td>1.31</td>
<td>1.55</td>
<td>2.45</td>
<td>3.01</td>
<td>3.56</td>
<td>4.1</td>
<td>4.53</td>
<td>4.04</td>
</tr>
<tr>
<td>0.91</td>
<td>1.2</td>
<td>1.09</td>
<td>-3</td>
<td>2.48</td>
<td>3.53</td>
<td>4.21</td>
<td>4.93</td>
<td>5.5</td>
<td>4.88</td>
</tr>
<tr>
<td>1.1</td>
<td>1.46</td>
<td>1.79</td>
<td>2.24</td>
<td>3.42</td>
<td>4.2</td>
<td>4.97</td>
<td>5.85</td>
<td>6.68</td>
<td>5.84</td>
</tr>
<tr>
<td>1.06</td>
<td>1.41</td>
<td>1.7</td>
<td>2.14</td>
<td>3.89</td>
<td>4.9</td>
<td>5.85</td>
<td>6.92</td>
<td>8.15</td>
<td>6.94</td>
</tr>
<tr>
<td>0.92</td>
<td>1.18</td>
<td>0.7</td>
<td>-7.39</td>
<td>3.43</td>
<td>5.39</td>
<td>6.67</td>
<td>8.15</td>
<td>10</td>
<td>8.19</td>
</tr>
<tr>
<td>1.09</td>
<td>1.45</td>
<td>1.75</td>
<td>2.18</td>
<td>3.89</td>
<td>4.88</td>
<td>5.84</td>
<td>6.92</td>
<td>8.15</td>
<td>6.94</td>
</tr>
<tr>
<td>1.07</td>
<td>1.56</td>
<td>2.05</td>
<td>2.65</td>
<td>3.38</td>
<td>4.11</td>
<td>4.92</td>
<td>5.83</td>
<td>6.68</td>
<td>5.82</td>
</tr>
</tbody>
</table>

Converged!
\[
\gamma = 0.9
\]

<table>
<thead>
<tr>
<th></th>
<th>0.41</th>
<th>0.74</th>
<th>0.96</th>
<th>1.18</th>
<th>1.43</th>
<th>1.71</th>
<th>1.98</th>
<th>2.11</th>
<th>2.39</th>
<th>2.09</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.74</td>
<td>1.04</td>
<td>1.27</td>
<td>1.52</td>
<td>1.81</td>
<td>2.15</td>
<td>2.47</td>
<td>2.58</td>
<td>3.02</td>
<td>2.69</td>
<td></td>
</tr>
<tr>
<td>0.86</td>
<td>1.18</td>
<td>1.45</td>
<td>1.76</td>
<td>2.15</td>
<td>2.55</td>
<td>2.97</td>
<td>3.32</td>
<td>3.69</td>
<td>3.32</td>
<td></td>
</tr>
<tr>
<td>0.84</td>
<td>1.11</td>
<td>1.31</td>
<td>1.55</td>
<td>2.45</td>
<td>3.01</td>
<td>3.56</td>
<td>4.1</td>
<td>4.53</td>
<td>4.04</td>
<td></td>
</tr>
<tr>
<td>0.91</td>
<td>1.2</td>
<td>1.09</td>
<td>-3</td>
<td>2.48</td>
<td>3.53</td>
<td>4.21</td>
<td>4.93</td>
<td>5.5</td>
<td>4.88</td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td>1.46</td>
<td>1.79</td>
<td>2.24</td>
<td>3.42</td>
<td>4.2</td>
<td>4.97</td>
<td>5.85</td>
<td>6.68</td>
<td>5.84</td>
<td></td>
</tr>
<tr>
<td>1.06</td>
<td>1.41</td>
<td>1.7</td>
<td>2.14</td>
<td>3.89</td>
<td>4.9</td>
<td>5.85</td>
<td>6.92</td>
<td>8.15</td>
<td>6.94</td>
<td></td>
</tr>
<tr>
<td>0.92</td>
<td>1.18</td>
<td>0.7</td>
<td>-7.39</td>
<td>3.43</td>
<td>5.39</td>
<td>6.67</td>
<td>8.15</td>
<td>10</td>
<td>8.19</td>
<td></td>
</tr>
<tr>
<td>1.09</td>
<td>1.45</td>
<td>1.75</td>
<td>2.18</td>
<td>3.89</td>
<td>4.88</td>
<td>5.84</td>
<td>6.92</td>
<td>8.15</td>
<td>6.94</td>
<td></td>
</tr>
<tr>
<td>1.07</td>
<td>1.56</td>
<td>2.05</td>
<td>2.65</td>
<td>3.38</td>
<td>4.11</td>
<td>4.92</td>
<td>5.83</td>
<td>6.68</td>
<td>5.82</td>
<td></td>
</tr>
</tbody>
</table>

\[
\gamma = 0.5
\]

<table>
<thead>
<tr>
<th></th>
<th>-0.28</th>
<th>-0.13</th>
<th>-0.12</th>
<th>-0.11</th>
<th>-0.09</th>
<th>-0.04</th>
<th>0.08</th>
<th>0.31</th>
<th>0.07</th>
<th>-0.19</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.13</td>
<td>-0.01</td>
<td>0</td>
<td>0.02</td>
<td>0.07</td>
<td>0.18</td>
<td>0.46</td>
<td>1.11</td>
<td>0.45</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td>-0.12</td>
<td>-0.01</td>
<td>-0.02</td>
<td>-0.24</td>
<td>0.05</td>
<td>0.19</td>
<td>0.47</td>
<td>1.12</td>
<td>0.48</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>-0.13</td>
<td>-0.02</td>
<td>-0.27</td>
<td>-5.12</td>
<td>-0.23</td>
<td>0.08</td>
<td>0.2</td>
<td>0.46</td>
<td>0.54</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>-0.12</td>
<td>-0.01</td>
<td>-0.04</td>
<td>-0.28</td>
<td>0.02</td>
<td>0.11</td>
<td>0.28</td>
<td>0.65</td>
<td>1.39</td>
<td>0.53</td>
<td></td>
</tr>
<tr>
<td>-0.12</td>
<td>-0.02</td>
<td>-0.06</td>
<td>-0.51</td>
<td>0.05</td>
<td>0.26</td>
<td>0.64</td>
<td>1.55</td>
<td>3.72</td>
<td>1.49</td>
<td></td>
</tr>
<tr>
<td>-0.13</td>
<td>-0.04</td>
<td>-0.53</td>
<td>-10.19</td>
<td>-0.33</td>
<td>0.5</td>
<td>1.39</td>
<td>3.72</td>
<td>10</td>
<td>3.74</td>
<td></td>
</tr>
<tr>
<td>-0.14</td>
<td>-0.03</td>
<td>-0.07</td>
<td>-0.51</td>
<td>0.04</td>
<td>0.25</td>
<td>0.63</td>
<td>1.55</td>
<td>3.72</td>
<td>1.49</td>
<td></td>
</tr>
<tr>
<td>-0.28</td>
<td>-0.14</td>
<td>-0.15</td>
<td>-0.18</td>
<td>-0.1</td>
<td>-0.01</td>
<td>0.16</td>
<td>0.54</td>
<td>1.32</td>
<td>0.43</td>
<td></td>
</tr>
</tbody>
</table>
Use **Bellman Backups on MDPs** to plan amidst uncertainty

Today’s Learning Goal:

- Review: Motion Planning collides with Accidents
- Motivating Example: Avoiding plane crashes
- Learning a Skill: Describing uncertain tasks with MDPs
- Exercise: Analyzing self-driving cars with MDPs
- Expanding our skill: Strategizing valuables moves via Bellman Backups
- Recap: Connecting it all together for your takeaway
Takeaways to Remember

- Our motion plans can be robust to uncertain (and some unmodeled) dynamics using Markov Decision Processes:
  
  - State space $s_t \in S$
  
  - Action space $a_t \in A$
  
  - Transition probabilities $T(s_{t+1} \mid s_t, a_t)$
  
  - Reward and Discount $r(s, a)$ and $\gamma$
  
- Dynamic programming follows waves as we plan ahead from our goals

$$\hat{V}(x) = \gamma \max_u r(x, u) + \int_X \hat{V}(x') p(x' \mid u, x) dx'$$