CSE 591: Human-aware Robotics

Instructor: Dr. Yu (“Tony”) Zhang

Location & Times: CAVC 359, Tue/Thu, 9:00--10:15 AM
Office Hours: BYENG 558, Tue/Thu, 10:30--11:30AM

Nov 3, 2016

Slides adapted from Alan Fern (OSU), Henry Kautz, Geoff Hollinger

This set of slides borrows from various online sources; it is used for educational purposes only.
Challenges in human-aware robotics

- **Perception of humans**
  Human recognition, human tracking, and activity recognition

- **Modeling of humans**
  Goal and intent recognition, human decision and behavioral models, expectation, model learning

- **Human-aware decision making**
  Human-aware planning, reinforcement learning and inverse reinforcement learning.

- **Human-robot interface**
  Command recognition, gesture recognition

"Sit, boy, sit! Sit, I say, Si... Oh, forget it."
Modeling of Humans

Human modeling

Human-aware planner

Human models

Plan generation

Robot models

Human teammate

Observations
Decision-Theoretic Assistance

Don’t just recognize!
Jump in and help..

Allows us to also talk about POMDPs

A Decision-Theoretic Model of Assistance, JAIR
Intelligent Assistants

• Many examples of AI techniques being applied to assistive technologies

• Intelligent Desktop Assistants
  ▶ Calendar Apprentice (CAP) (Mitchell et al. 1994)
  ▶ Travel Assistant (Ambite et al. 2002)
  ▶ CALO Project
  ▶ Tasktracer
  ▶ Electric Elves (Hans Chalupsky et al. 2001)

• Assistive Technologies for the Disabled
  ▶ COACH System (Boger et al. 2005)
Not So Intelligent

• Most previous work uses problem-specific, hand-crafted solutions
  ▶ Lack ability to offer assistance in ways not planned for by designer

• **Our goal:** provide a general, formal framework for intelligent-assistant design

• **Desirable properties:**
  ▶ Explicitly reason about models of the world and user to provide flexible assistance – Human modeling
  ▶ Handle uncertainty about the world and user
  ▶ Handle variable costs of user and assistive actions

• We describe a model-based decision-theoretic framework (MDP) that captures these properties
What is a Markov Chain?

- Finite number of discrete states
- Probabilistic transitions between states
- Next state determined only by the current state
  - This is the Markov property

![Diagram of Markov Chain]

Rewards: S1 = 10, S2 = 0
What is a Hidden Markov Model?

- Finite number of discrete states
- Probabilistic transitions between states
- Next state determined only by the current state
- We’re unsure which state we’re in
  - The current state emits an observation

Rewards: $S_1 = 10$, $S_2 = 0$

Do not know state:
- $S_1$ emits $O_1$ with prob 0.75
- $S_2$ emits $O_2$ with prob 0.75
What is a Markov Decision Process?

- Finite number of discrete states
- Probabilistic transitions between states and controllable actions in each state
- Next state determined only by the current state and current action
  - This is still the Markov property

Rewards: $S1 = 10$, $S2 = 0$
What is a Partially Observable Markov Decision Process?

- Finite number of discrete states
- Probabilistic transitions between states and controllable actions
- Next state determined only by the current state and current action
- We’re unsure which state we’re in
  - The current state emits observations

Rewards: S1 = 10, S2 = 0

Do not know state:
- S1 emits O1 with prob 0.75
- S2 emits O2 with prob 0.75
# A Very Helpful Chart

<table>
<thead>
<tr>
<th>Markov Models</th>
<th>Do we have control over the state transitions?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="chart.png" alt="Chart" /></td>
</tr>
</tbody>
</table>

- **YES**
  - Markov Chain
  - MDP (Markov Decision Process)
- **NO**
  - HMM (Hidden Markov Model)
  - POMDP (Partially Observable Markov Decision Process)
POMDP versus MDP

**MDP**
- +Tractable to solve
- +Relatively easy to specify
- -Assumes perfect knowledge of state

**POMDP**
- +Treats all sources of uncertainty uniformly
- +Allows for information gathering actions
- -Hugely intractable to solve optimally
Time for Some Formalism

- **POMDP model**
  - Finite set of states: $s_1, \ldots, s_n \in S$
  - Finite set of actions: $a_1, \ldots, a_m \in A$
  - Probabilistic state-action transitions: $p(s_i | a, s_j)$
  - Reward for each state/action pair*: $r(s, a)$
  - Conditional observation probabilities: $p(o | s)$

- **Belief state**
  - Probability distribution over world states: $b(s) = p(s)$
  - Action update rule: $b'(s) = \sum_{s'} \in S p(s | a, s') \cdot b(s')$
  - Observation update rule: $b'(s) = p(o | s) \cdot b(s)/k$
Belief States

- If we have $k$ state variables, $2^k$ states
- A “belief state” is a probability distribution over states
  - Non-deterministic
    - We just know the states for which the probability is non-zero
    - $2^{2^k}$ belief states
  - Stochastic
    - We know the probability distribution over the states
    - Infinite number of probability distributions
  - A complete state is a special case of belief state where the distribution is “dirac-delta”
    - i.e., non-zero only for one state

In blocks world,
Suppose we have blocks A and B and they can be “clear”, “on-table” “On” each other

- A state: A is on table, B is on table, both are clear, hand is empty

- A belief state :
  A is either on B or on Table
  B is on table. Hand is empty

→ 2 states in the belief state
Actions and Belief States

• Two types of actions
  • Standard actions: Modify the distribution of belief states
    • Doing “C on A” action in the belief state gives us a new belief state (with C on A on B OR C on A; B clear)
    • Doing “Shake-the-Table” action converts the previous belief state to (A on table; B on Table; A clear; B clear)
      ▼ Notice that actions reduce the uncertainty!

• Sensing actions
  • Sensing actions observe some aspect of the belief state
  • The observations modify the belief state distribution
    • In the belief state above, if we observed that two blocks are clear, then the belief state changes to {A on table; B on table; both clear}
    • If the observation above is noisy (i.e., we are not completely certain), then the probability distribution just changes so more probability mass is centered on the {A on table; B on Table} state.

A belief state:
- A is either on B or on Table
- B is on table. Hand is empty
POMDP as Belief-State MDP

- Equivalent belief-state MDP
  - Each MDP state is a probability distribution (continuous belief state $b$) over the states of the original POMDP
  - State transitions are products of actions and observations
    \[
    b'(s') = p(s' | a, o, b) = \frac{p(o | s', a, b) \cdot p(s' | a, b)}{p(o | a, b)}
    \]
    \[
    p(o | s', a, b) = p(o | s')
    \]
    \[
    p(s' | a, b) = \sum_{s \in S} p(s' | a, s) \cdot b(s)
    \]
    \[
    p(o | a, b) = \sum_{s' \in S} p(o | s') \cdot p(s' | a, b)
    \]
  - Rewards are expected rewards of original POMDP
    \[
    R(a, b) = \sum_{s \in S} r(a, s) \cdot b(s)
    \]
Our First POMDP Solving Algorithm

- Discretize the POMDP belief space
  - Solve the resulting belief-space MDP using
    - Value iteration
    - Policy iteration
    - Any MDP solving technique
- Why might this not work very well?

![Diagram showing a line segment with an annotation](image-url)
Value Iteration for POMDPs

- Until someone figured out
  - The value function of POMDPs can be represented as max of linear segments
    - Each vector typically called “alpha vector”: $\alpha_i \cdot b$
    - This is piecewise-linear-convex

$$V^*_t(b) = \max_{\sigma \in \Gamma_t} \sum_{s \in S} b(s) \alpha^\sigma(s) = \max_{\alpha \in \mathcal{V}_t} \sum_{s \in S} b(s) \alpha(s).$$

$$\alpha^\sigma = [V^\sigma(s_0), V^\sigma(s_1), \ldots, V^\sigma(s_N)].$$
Value Iteration for POMDPs

- Basic idea
  - Calculate value function vectors for each action (horizon 1 value function)
    - Keep in mind we need to account for observations
  - Continue looking forward (horizon 2, horizon 3)
  - Iterate until convergence
Value Iteration for POMDPs

- Example POMDP for value iteration
  - Two states: s1, s2
  - Two actions: a1, a2
  - Three observations: z1, z2, z3
  - Positive rewards in both states:
    \[
    R(s_1, a_1) = 1.0, \quad R(s_1, a_2) = 0 \\
    R(s_2, a_1) = 0, \quad R(s_2, a_2) = 1.5
    \]
Value Iteration for POMDPs

- Horizon 1 value function
  - Calculate immediate rewards for each action in belief space

\[ R(s_1, a_1) = 1.0, \quad R(s_1, a_2) = 0 \]
\[ R(s_2, a_1) = 0, \quad R(s_2, a_2) = 1.5 \]

\[ b = [0.25 \ 0.75] \]
Value Iteration for POMDPs

- Need to transform value function with observations
Value Iteration for POMDPs

- Each action from horizon 1 yields new vectors from the transformed space

Value function and partition for taking action a1 in step 1
Value Iteration for POMDPs

- Each action from horizon 1 yields new vectors from the transformed space
Value Iteration for POMDPs

- Combine vectors to yield horizon 2 value function

Combined a1 and a2 value functions
Value Iteration for POMDPs

- Combine vectors to yield horizon 2 value function (can also prune dominated vectors)
Value Iteration for POMDPs

- Iterate to convergence
  - This can sometimes take many steps
- Course reading also gives horizon 3 calculation
  - “POMDPs for Dummies” by Tony Cassandra

Horizon 2 value function with pruning

Horizon 3 value function with pruning
Value Iteration for POMDPs

- After all that...
- The good news
  - Value iteration is an exact method for determining the value function of POMDPs
  - The optimal action can be read from the value function for any belief state
- The bad news
  - Time complexity of solving POMDP value iteration is exponential in:
    - Actions and observations
  - Dimensionality of the belief space grows with number of states
Solving POMDPs

- Exact value iteration
- Policy iteration
- Witness algorithm, HSVI
- Greedy solutions
Decision-Theoretic Assistance

Don’t just recognize!
Jump in and help..

Allows us to also talk about POMDPs
An Episodic Interaction Model

Action set $U$

User

Goal

Action set $A$

Assistant

Each user and assistant action has a cost

Objective: minimize expected cost of episodes
Example: Grid World Domain

World states:
(x,y) location and door status

Possible goals:
Get wood, gold, or food

User actions:
Up, Down, Left, Right, noop
Open a door in current room
(all actions have cost = 1)

Assistant actions:
Open a door, noop
(all actions have cost = 0)
World and User Models

• Model world dynamics as a Markov decision process (MDP)
• Model user as a stochastic policy

Given: model, action sequence
Output: assistant action
Optimal Solution: Assistant POMDP

- Can view as a POMDP called the **assistant POMDP**
  - Hidden State: user goal
  - Observations: user actions and world states
- Optimal policy gives mapping from observation sequences to assistant actions
  - Represents optimal assistant
- Typically intractable to solve exactly
Approximate Solution Approach

- Online actions selection cycle
  1) Estimate posterior goal distribution given observation
  2) Action selection via myopic heuristics
Approximate Solution Approach

- Online actions selection cycle
  1) Estimate posterior goal distribution given observation
  2) Action selection via myopic heuristics
Goal Estimation

- Given
  - $P(G \mid O_t)$: goal posterior at time $t$ initially equal to prior $P(G)$
  - $P(U_t \mid G, W_t)$: stochastic user policy
  - $O_{t+1}$: new observation of user action and world state

It is straightforward to update goal posterior at time $t+1$

$$P(G \mid O_{t+1}) \propto P(G \mid O_t) \cdot P(U_t \mid G, W_{t+1})$$

must learn user policy

Goal posterior given observations up to time $t$

new observation

Updated goal posterior

$P(G \mid O_t) \quad W_t \quad W_{t+1} \quad P(G \mid O_{t+1})$

Current State
Learning User Policy

- Use Bayesian updates to update user policy $P(U|G, W)$ after each episode
  - **Problem:** can converge slowing, leading to poor goal estimation
  - **Solution:** use strong prior on user policy derived via planning

- Assume that user behaves “nearly rational”
  - Take prior distribution on $P(U|G, W)$ to be bias toward optimal user actions

- Let $Q(U,W,G)$ be value of user taking action $U$ in state $W$ given goal $G$
  - Can compute via MDP planning
  - Use prior $P(U | G, W) \propto \exp(Q(U,W,G))$
\( Q(U,W,G) \) for Grid World
Approximate Solution Approach

- Online actions selection cycle
  1) Estimate posterior goal distribution given observation
  2) Action selection via myopic heuristics
**Action Selection: Assistant POMDP**

- Assume we know the user goal $G$ and policy
  - Can create a corresponding **assistant MDP** over assistant actions
  - Can compute $Q(A, W, G)$ giving value of taking assistive action $A$ when users goal is $G$

- Select action that maximizes expected (myopic) value:

$$Q(A, W) = \sum_G P(G \mid O_t) \cdot Q(A, W, G)$$

If you just want to recognize, you only need $P(G \mid O_t)$
If you just want to help (and know the goal), you just need $Q(A, W, G)$
Summary of Assumptions

• Model Assumptions:
  ▶ World can be approximately modeled as MDP
  ▶ User and assistant interleave actions (no parallel activity)
  ▶ User can be modeled as a stationary, stochastic policy
  ▶ Finite set of known goals

• Assumptions Made by Solution Approach
  ▶ Access to practical algorithm for solving the world MDP
  ▶ User does no reason about the existence of the assistance
  ▶ Goal set is relatively small and known to assistant
  ▶ User is close to “rational”