
Spring 2019

Planning
Overview

• Planning
  – AI planning
  – Path planning

• References
  – Planning algorithms – LaValle, 2006
    http://planning.cs.uiuc.edu/
  – Motion planning – Kelly, 2013
  – AI the modern approach – Russell & Norvig 3rd edition
Planning

Classical planning
AI planning
Path planning
Motion planning
Robot planning
…
Define planning
Define planning

“Devise a plan of action to achieve one’s goal” [Russell & Norvig]

Find a sequence of actions to take to reach goal state from initial state
Example: planning

Find a sequence of actions to take to reach goal state from initial state

Initial state: B I G

Actions: right right

Goal state: B I G
Blocks world example

Predicates:
(Block X): X is a block
(On X Y): X is on Y
(Clear X): There is nothing on X.

Operators:
P ^ Q: Both P and Q are true.
P ∨ Q: Either P or Q is true.
~P: P is not true.

Initial state

G
B
I
Blocks world example

Predicates:
(Block X): X is a block
(On X Y): X is on Y
(Clear X): There is nothing on X.

Operators:
P ∧ Q: Both P and Q are true.
P ∨ Q: Either P or Q is true.
~P: P is not true.

Initial state: (Block G)^ …
Exercise 1: define state

Initial state:
Goal state:
Exercise 1: define state

Initial state: $(\text{On } G \space B)^{\land} (\text{On } B \space I)^{\land} (\text{On } I \space \text{Table})^{\land} (\text{Clear } G)$

Goal state: $(\text{On } B \space I)^{\land} (\text{On } I \space G)^{\land} (\text{On } G \space \text{Table})^{\land} (\text{Clear } B)$
Action changes the world

• Action can be taken when **pre-conditions** are satisfied
• Action has **effects** (or **post-conditions**)

Example:

`Action (MoveToBlock ?b ?from ?to)`
Precond: …
Effect: …
Exercise 2: define action

Action (MoveToBlock ?b ?from ?to)
Precond:

Effect:

(MoveToBlock B I G):
Move block B from I to G
Exercise 2: define action

Action (MoveToBlock ?b ?from ?to)

[Modified from Fig 10-3, Russell & Norvik]

(MoveToBlock B I G):
Move block B from I to G
Exercise 3: take action

(MoveToBlock B I G)

Precond: (On B I)^(Clear B)^(Clear G)^
     (Block B)^(Block G)^(!= I G)^(!= B I)^(!= B G)

Effect: (Clear I)^^(On B G)^~(Clear G)^~(On B I)

Initial state: (On G Table)^^(On I Table)^^(Clear G)^^(Clear B)^^(On B I)

Goal state: (On G Table)^^(On I Table)^(Clear B)^^(Clear I)^^(On B G)
Exercise 3: take action

(MoveToBlock B I G)
Precond: (On B I)^(Clear B)^(Clear G)^
   (Block B)^(Block G)^^(≠ I G)^^(≠ B I)^^(≠ B G)
Effect: (Clear I)^(On B G)^~(Clear G)^~(On B I)

Initial state: (On G Table)^^(On I Table)^(Clear G)^^(Clear B)^^(On B I)^
   ^^(Clear I)^^(On B G)
Goal state: (On G Table)^^(On I Table)^(Clear B)^^(Clear I)^^(On B G)
Planning search space

Initial state

Goal state
Planning and scheduling

• What is the difference between planning and scheduling?

• Scheduling is resource allocation, ideally in an optimal way
Motion Planning

“Convert high-level specification of tasks from humans into low-level descriptions of how to move” [LaValle’06]

Slides are based on Planning Algorithms (LaValle’06)
Desiderata for planners

• Sound
  – Feasible – vehicle constraints
  – Admissible – avoid obstacles

• Complete
  – If any solution exists, it will be found
  – Otherwise, report failure

• Optimal: If more than one solutions exist the best will be generated
Discrete feasible planning

Problem formulation

• $X$: a set of states
• $U(x)$: a set of actions available in state $x$ in $X$
• $f(x,u)$: state transition function for every state $x$, $u$ in $X$
• $x_I$: Initial state
• $x_G$: Goal state
Search graph

- **Is_goal?** (state $s$) $\rightarrow$ yes | no
- **Get_actions** (state $s$) $\rightarrow$ set of actions $A$
- **Next_state** (state $s$, action $a$) $\rightarrow$ state $s'$
States of state

• Unvisited
• Dead
• Alive
Search

Backward, bidirectional

```plaintext
FORWARD-SEARCH
1 Q.Insert(x_I) and mark x_I as visited
2 while Q not empty do
3 \( x \leftarrow Q.GetFirst() \)
4 if \( x \in X_G \)
5 \( \text{return SUCCESS} \)
6 forall \( u \in U(x) \)
7 \( x' \leftarrow f(x, u) \)
8 if \( x' \) not visited
9 \( \text{Mark } x' \text{ as visited} \)
10 \( Q.Insert(x') \)
11 else
12 \( \text{Resolve duplicate } x' \)
13 \( \text{return FAILURE} \)  [LaValle Fig. 2.4]
```

Priority Queue

- Breadth-first
- Depth-first
- Best-first
- Iterative deepening
- Dijkstra
- A* (A-star)
Forward search

\( s_I \)

\( Q \)
Forward search

Q

Is goal (s)? Return SUCCESS

Is empty (Q)? Return FAIL
Forward search

Apply each action in state \( s \) and get resulting state \( s' \)

Haven’t visited \( (s') \)?

Sort states according to...
Breadth-first
Depth-first
Optimal planning

• Cost optimization
• Efficient search
Dijkstra’s algorithm

• Sort according to cost-to-come to the state so far
• Nonnegative costs
A*

- Cost-to-come $C(x) + \text{Cost-to-go } G(x)$
- Admissible (underestimate) heuristics
- $G(x) = 0$: Dijkstra
Value iteration

• Fixed-length plans (length=K)
• $O(|U|^K)$
• Value iteration
  – Cost “to go” or
  – Cost “to come”
  – Dynamic programming
Backwards value iteration

K-step plan
Stages 1, ..., K, F
Actions $u_1, ..., u_K$
States $x_1, ..., x_K, x_F$
State transition $f(x,u)$

Cost of taking action $u_i$ in state $x_i$

Accumulated cost-to-go from stage $k$ to $F$ under optimal plan

$G^*_k(x_k) = \min_{u_k, ..., u_K} \left\{ \sum_{i=k}^{K} l(x_i, u_i) + l_F(x_F) \right\}$

Number of actions

$F = K+1$; $x_F$ denotes final state;
$l_F(x_G) = 0$, $l_F(x) = \infty$ for all $x \neq x_G$

[LaValle Ch. 2]
Backward value iteration

\[ G^*_K(x_K) = \min_{u_K} \left\{ l(x_K, u_K) + l_F(x_F) \right\} \]

\[ G^*_K(x_K) = \min_{u_K} \left\{ l(x_K, u_K) + G^*_F(f(x_K, u_K)) \right\} \]

\[ G^*_K(x_K) = \min_{u_K} \left\{ l(x_K, u_K) + G^*_{K+1}(x_{K+1}) \right\} \]

\[ l_F = G^*_F \]

\[ x_F = f(x_K, u_K) \]
Backward value iteration

\[
G_k^*(x_k) = \min_{u_k, \ldots, u_K} \left\{ \sum_{i=k}^{K} l(x_i, u_i) + l_F(x_F) \right\}
\]

\[
G_k^*(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + \min_{u_{k+1}, \ldots, u_K} \left\{ \sum_{i=k+1}^{K} l(x_i, u_i) + l_F(x_F) \right\} \right\}
\]
Backward value iteration

\[
G^*_k(x_k) = \min_{u_k, \ldots, u_K} \left\{ \sum_{i=k}^{K} l(x_i, u_i) + l_F(x_F) \right\}
\]

\[
G^*_k(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + \min_{u_{k+1}, \ldots, u_K} \left\{ \sum_{i=k+1}^{K} l(x_i, u_i) + l_F(x_F) \right\} \right\}
\]

\[
G^*_k(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + G^*_{k+1}(x_{k+1}) \right\}
\]
Fixed-length plans (length=K)

- $O(|U|^K)$
- Value (= cost to go) iteration
  - Dynamic programming
    - $O(K|X||U|)$
You are in NSH and want to be at UC in exact 3 stages with minimum cost

(3 stages means 3 actions, 4 states, i.e., K=3)
Example
Example
Example
Example

\[ G^*_k(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + G^*_{k+1}(x_{k+1}) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>G^*_4</td>
<td>\infty</td>
<td>\infty</td>
<td>\infty</td>
<td>0</td>
</tr>
<tr>
<td>G^*_3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G^*_2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G^*_1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Forward value iteration

Optimal cost to come

\[ C_k^*(x_k) = \min_{u_1, \ldots, u_{k-1}} \left\{ l_I(x_1) + \sum_{i=1}^{k-1} l(x_i, u_i) \right\} \]

\[ C_k^*(x_k) = \min_{u_k^{-1} \in U^{-1}(x_k)} \left\{ C_{k-1}^*(x_{k-1}) + l(x_{k-1}, u_{k-1}) \right\} \]

\( l_I(x_I) = 0, \quad l_I(x) = \infty \) for all \( x \neq x_I \)
Example

\[ C_k^*(x_k) = \min_{u_k^{-1} \in U^{-1}(x_k)} \left\{ C_{k-1}^*(x_{k-1}) + l(x_{k-1}, u_{k-1}) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C_1</td>
<td>$\infty$</td>
<td>0</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>C_2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C_3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C_4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

\[ C^*_k(x_k) = \min_{u_k^{-1} \in U^{-1}(x_k)} \left\{ C^*_{k-1}(x_{k-1}) + l(x_{k-1}, u_{k-1}) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C^*_1)</td>
<td>(\infty)</td>
<td>(0)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td>(C^*_2)</td>
<td>(?)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(C^*_3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(C^*_4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

\[ C_k^*(x_k) = \min_{u_k^{-1} \in U^{-1}(x_k)} \left\{ C_{k-1}^*(x_{k-1}) + l(x_{k-1}, u_{k-1}) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C_1</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>C_2</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C_3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C_4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

\[ C_{k}^{*}(x_k) = \min_{u_{k}^{-1} \in U^{-1}(x_k)} \left\{ C_{k-1}^{*}(x_{k-1}) + l(x_{k-1}, u_{k-1}) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C*&lt;sub&gt;1&lt;/sub&gt;</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>C*&lt;sub&gt;2&lt;/sub&gt;</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>C*&lt;sub&gt;3&lt;/sub&gt;</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>C*&lt;sub&gt;4&lt;/sub&gt;</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>
Any length optimal planning

[States $X$, Actions $U$, State transition function $f$, initial state $s_I$, goal state $s_G$] +

\[
L(\pi_K) = \sum_{k=1}^{K} l(x_k, u_k) + l_F(x_F)
\]

Additive cost function for any K-step plan $\pi_K$

Termination action $u_T$ in action set $U(x)$

$l(x, u_T) = 0; f(x, u_T) = x$

Taking $u_T$ doesn’t accumulate cost nor change state
Any length optimal planning

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

Optimal cost to go

\[ u^* = \arg\min_{u \in U(x)} \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

Optimal action
Example

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G^*_0 )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_{-1} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G^*_{-2} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G^*_{-3} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G^*_{-4} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G^*_{-5} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination action \( u_T \)

1/15/19
Example

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G^*_0)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>0</td>
</tr>
<tr>
<td>(G^*_1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination action \(u_T\)
Example

$$G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\}$$

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G^*_0$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>0</td>
</tr>
<tr>
<td>$G^*_1$</td>
<td>$\infty$</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$G^*_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G^*_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G^*_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G^*_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination action $u_T$
**Example**

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_0^*)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>0</td>
</tr>
<tr>
<td>(G_{-1}^*)</td>
<td>(\infty)</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Termination action \(u_T\)
Example

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G^*_0)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>0</td>
</tr>
<tr>
<td>(G^*_{-1})</td>
<td>(\infty)</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>(G^*_{-2})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_{-3})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_{-4})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(G^*_{-5})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination action \(u_T\)

1/15/19

CMU 16-785: Integrated Intelligence in Robotics

Jean Oh 2019
Example

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G_0^* )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>( G_{-1}^* )</td>
<td>( \infty )</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( G_{-2}^* )</td>
<td>( \infty )</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G_{-3}^* )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G_{-4}^* )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G_{-5}^* )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination action \( u_T \)

1/15/19
Example

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

<table>
<thead>
<tr>
<th></th>
<th>WH</th>
<th>NSH</th>
<th>GHC</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G^*_0 )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_1 )</td>
<td>( \infty )</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_2 )</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_3 )</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_4 )</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( G^*_5 )</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Termination action \( u_T \)

1/15/19
Navigation planning

• Costs are revealed during execution
• Implicit constraint on state transition: robots cannot teleport
• D* (D-star): Dynamic, backward variation of Dijkstra (cost to go) [Stentz’94]

D* path planner

NASA Mars Curiosity Rover

**D**

- Backward Dijkstra; sort Q according to $G_{\text{best}}(x)$
- $G_{\text{best}}(x)$: lowest cost to go while in Queue
- $G_{\text{cur}}(x)$: current cost to go (possibly higher than $G_{\text{best}}$ due to dynamic updates)
- $G_{\text{via}}(x, x')$: cost to go from $x$ by traveling via $x'$

\[
x' = f(x, u)
\]

\[x \quad u \quad x' \quad ... \quad S_G\]
\( x \leftarrow \arg\min_{x \in Q} (G_{\text{best}}(x)) \)

If \( G_{\text{best}}(x) < G_{\text{cur}}(x) \)
for all \( x' = f(x,u) \)
\begin{align*}
&\text{If } G_{\text{via}}(x,x') < G_{\text{cur}}(x) \text{ & } G_{\text{cur}}(x') \leq G_{\text{best}}(x) \\
&\quad G_{\text{cur}}(x) := G_{\text{via}}(x,x') \text{ & } \pi(x) = u
\end{align*}
for all \( x' \) that can reach \( x \) in 1 action, i.e., \( x = f(x',u') \)
\begin{align*}
&\text{If } x' \text{ is unvisited} \\
&\quad \pi(x') := u' \\
&\quad \text{insert } x' \text{ onto } Q \text{ with cost } G_{\text{via}}(x',x)
\end{align*}
If \( \pi(x') = u' \) but \( G_{\text{via}}(x',x) \neq G_{\text{cur}}(x') \)
\begin{align*}
&\quad \text{insert } x' \text{ onto } Q \text{ with cost } G_{\text{via}}(x',x)
\end{align*}
If \( \pi(x') \neq u' \) but \( G_{\text{via}}(x',x) < G_{\text{cur}}(x') \)
\begin{align*}
&\quad \text{if } G_{\text{cur}}(x) = G_{\text{best}}(x) \text{ then } \pi(x') := u' \text{ & insert } x'
\end{align*}
onto \( Q \)
\begin{align*}
&\quad \text{else insert } x \text{ onto } Q \text{ with } G_{\text{cur}}(x)
\end{align*}
If \( x' \) is dead,
\begin{align*}
&\text{if } \pi(x') \neq u', G_{\text{via}}(x,x') < G_{\text{cur}}(x), \text{ and } G_{\text{cur}}(x) > G_{\text{best}}(x) \\
&\quad \text{then insert } x' \text{ back onto } Q \text{ with cost } G_{\text{cur}}(x')
\end{align*}
If $G_{\text{best}}(x) < G_{\text{cur}}(x)$ then the cost of $x$ is actually higher than thought (underestimated)
If $G_{\text{best}}(x) < G_{\text{cur}}(x)$ then the cost of $x$ is actually higher than thought.

\[ \pi(x') = u' \text{ but } G_{\text{via}}(x',x) \neq G_{\text{cur}}(x') \]
1) $G_{\text{best}}(x) = G_{\text{cur}}(x)$

If $G_{\text{best}}(x) < G_{\text{cur}}(x)$ then the cost of $x$ is actually higher than thought.

$\pi(x') \neq u'$ but $G_{\text{via}}(x',x) < G_{\text{cur}}(x')$
If $G_{\text{best}}(x) < G_{\text{cur}}(x)$ then the cost of $x$ is actually higher than thought.

2) $G_{\text{best}}(x) \neq G_{\text{cur}}(x)$

$\pi(x') \neq u'$ but $G_{\text{via}}(x',x) < G_{\text{cur}}(x')$
Motion Planning

"Convert high-level specification of tasks from humans into low-level descriptions of how to move" [LaValle’06]

Natural language $\rightarrow$ GoTo ($X,Y$) $\rightarrow$ D* $\rightarrow$ Waypoints ($x_1,y_1$) ($x_2,y_2$) …
Next

• Preparing for data-driven approaches
• Dataset analysis