

Assignment #2

Tracking and Discrete Search

Due: Start of Lecture, Week 9 (2pm, Thursday 25 September 2008)

1 Overview

The goal of this assignment is to test your understanding of Bayesian tracking and discrete search algorithms. While this assignment does not directly ask you to implement or trace all the algorithms discussed in class, it may be advantageous for you to do so.

You should feel free to ask the Lecturer questions and discuss the assignment with others, but you should not copy someone else's work. (e.g. You can get someone to show you how to answer a question, but you should write your answer up yourself from what you remember at least 30 minutes after they've left.)

Total marks for this assignment: 90. It will be scaled to 15% of the comp3431 grade.

2 Bayesian Tracking

You have been working for a company developing a computer game for young girls. In this game an animated ballerina must dance across the screen at one point. You decide to use motion capture to record a real ballerina and use that to animate your model.

The motion capture system works using computer vision. An actress is dressed in a black leotard with small brightly coloured balls attached to it. The actress is then captured on video tape as she performs the required motion.

Your job is to design the system that will track the ballerina's motion (track the small coloured balls attached to the actress in three dimensions). You decide to build your own system using a Bayesian Filter rather than purchasing a commercial system.

The sensors available to you are a set of 10 video cameras that can see the stage from all directions. Each small ball is differently coloured in such a way that they can be individually identified by the computer vision system. Unfortunately the system cannot detect the distance to the balls from a single camera image. There are 30 of these small coloured balls attached to the actress. The video cameras are all synchronised (the images are taken at precisely the same instant in time) and record at 100 frames per second.

You have previously measured the locations of the cameras so that you can project a virtual ray out of any camera through any pixel in that camera's image plane to determine a line in space along which the shown object must lie. There is an error in this process which can be roughly approximated as Gaussian noise.

The animation team has already developed a motion model for humanoid figures. Given a particular state (no specific ‘action’ is required) it will return a probability distribution over possible next states. This motion model can be adjusted to fit your particular state representation as required. This motion model is highly non-linear over longer periods of time.

2.1 Question 2.1 (5 marks):

Describe the state space you would choose for this problem. Please explain briefly why you chose that state space.

2.2 Question 2.2 (6 marks):

Bayesian tracking requires representing a probability distribution over the state space. List three different methods for representing a probability distribution for Bayesian tracking (they do not need to be efficient for this particular problem).

2.3 Question 2.3 (5 marks):

How would you represent a probability distribution over your chosen state space for this particular problem? Why?

2.4 Question 2.4 (5 marks):

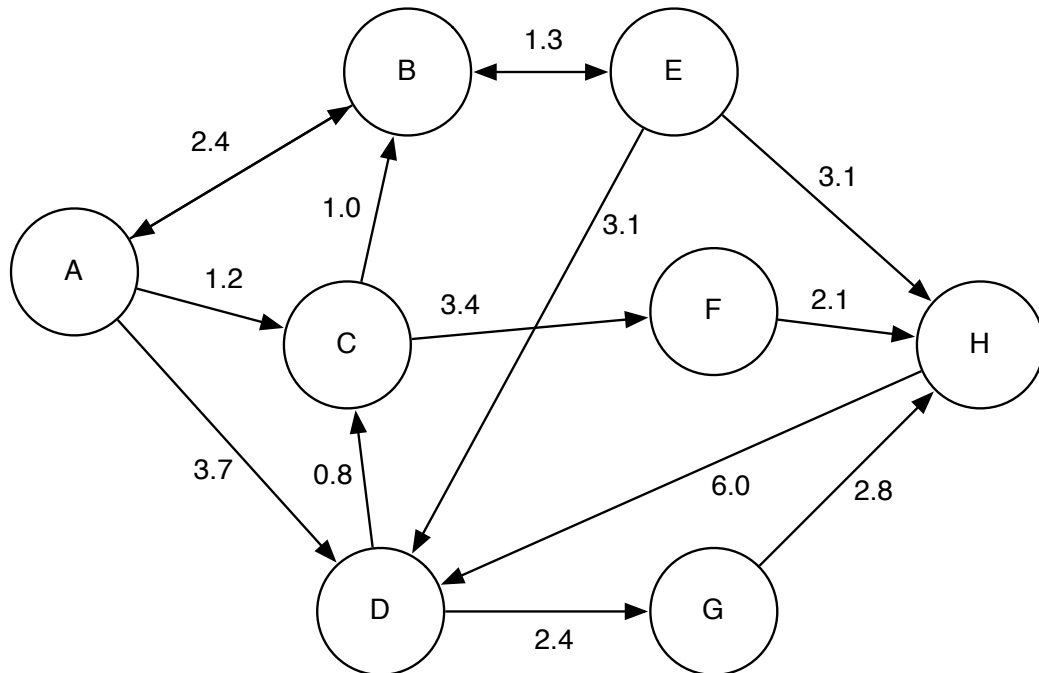
While debugging you decide to use a one-dimensional Kalman filter on a small subspace of the full state space (Aside: you shouldn’t assume this is a hint about the previous question). You remember from class that the pointwise product of two Gaussians can be calculated as follows: $G_1(\mu_1, \sigma_1^2).G_2(\mu_2, \sigma_2^2) = C.G_3(\mu_3, \sigma_3^2)$ where $\mu_3 = \frac{\sigma_2^2 \cdot \mu_1 + \sigma_1^2 \cdot \mu_2}{\sigma_1^2 + \sigma_2^2}$ and $\sigma_3^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2}$.

There is one ball that has a Gaussian prior state distribution with a mean $\mu = 27$ and a variance $\sigma^2 = 100$. You receive an observation of this ball’s position at location 31, the variance of observations is known to be about $\sigma^2 = 144$.

What is the mean and variance of the belief state distribution after incorporating this observation?

3 Discrete Search

In the following questions we’ll be considering this graph:



With the following heuristic costs to node H:

Node	Distance
A	6.0
B	4.1
C	5.4
D	4.5
E	2.4
F	1.9
G	2.5
H	0.0

You should assume that for any node, if we need its children, we can iterate through them in alphabetical order. When asked to trace a discrete search algorithm, you should list the state of the fringe data structure just before a node is removed from it, you should also list what node is removed, and any state associated with that node. This should be repeated from when the start node is added to the fringe, until the algorithm is complete.

3.1 Question 3.1 (5 marks):

Write the pseudo-code for a generalised discrete search algorithm. Your algorithm should print out the final path between the start and goal nodes.

3.2 Question 3.2 (5 marks):

Trace the behaviour of Depth First Search on the graph starting at node A, with a goal of node H.

3.3 Question 3.3 (5 marks):

Trace the behaviour of Breadth First Search on the graph starting at node A, with a goal of node H.

3.4 Question 3.4 (5 marks):

Trace the behaviour of Best First Search on the graph starting at node A, with a goal of node H.

3.5 Question 3.5 (5 marks):

Trace the behaviour of Uniform Cost Search on the graph starting at node A, with a goal of node H.

3.6 Question 3.6 (5 marks):

Trace the behaviour of A* Search on the graph starting at node A, with a goal of node H.

3.7 Question 3.7 (5 marks):

Give an admissible heuristic estimate for the minimum number of search steps from each node to H. Your heuristic should assign non-zero values to some nodes.

3.8 Question 3.8 (5 marks):

Use your heuristic to perform Best First Search on the graph. How does use of this heuristic compare with the cost-heuristic above?

3.9 Question 3.9 (10 marks):

Prove that A* with an admissible heuristic returns a lowest cost path through a graph.

4 Partial Order Planning

Here is a description of an SNLP planning operator (for the Towers of Hanoi domain):

```

(defstep
  :action ' (move-disk ?disk ?below-disk ?new-below-disk)
  :precond ' ((on ?disk ?below-disk)
              (clear ?disk)
              (clear ?new-below-disk)
              (smaller ?disk ?new-below-disk) ;handles pegs!
              )
  :add ' ((clear ?below-disk)
          (on ?disk ?new-below-disk))
  :delete ' ((on ?disk ?below-disk)
             (clear ?new-below-disk))
  :equals ' ((not (?new-below-disk ?below-disk))
             (not (?new-below-disk ?disk))
             (not (?below-disk ?disk))
             ))

```

There are also the following parts of the state description that never change. Note that because there are a lot of them, I've used a modified set notation to describe them. The first set says that all discs are smaller than all pegs:

$$\forall D \in \{\text{discA}, \text{discB}, \text{discC}, \text{discD}\}, \forall P \in \{\text{peg1}, \text{peg2}, \text{peg3}\}, (\text{smaller } D P).$$

The next part describes the order of every pair of discs. A disc with a later letter is smaller than a disc with an earlier letter – discA is the largest, and they get smaller from there.

$$\forall D_a, D_b \in \{A, B, C, D\}, \text{ if } D_b < D_a \text{ then } (\text{smaller disc} D_b \text{ disc} D_a).$$

Finally, we can describe the parts of the state that actually change:

Start state:

```

(on discA peg1) (on discB discA) (on discC discB)
(on discD discC) (clear discD) (clear peg2) (clear peg3)

```

Goal state:

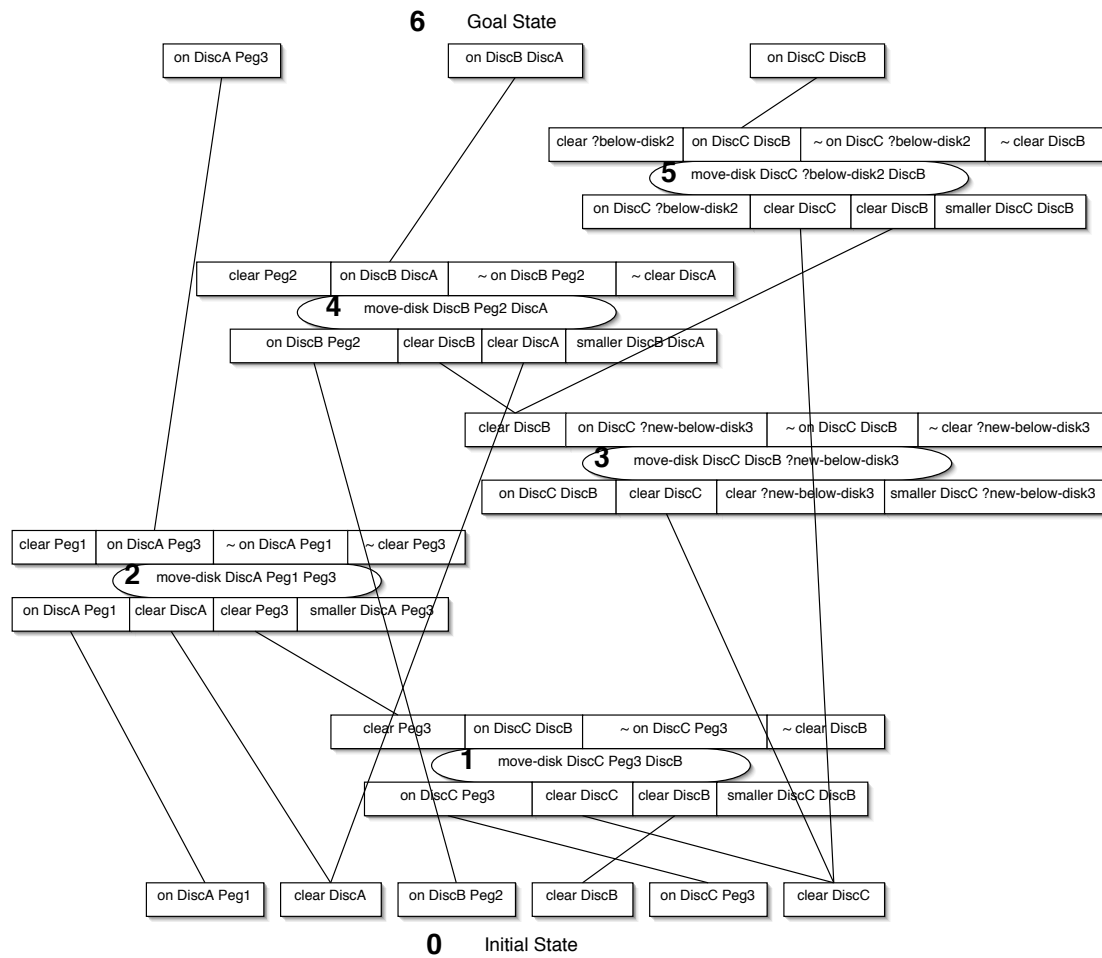
```

(on discA peg3) (on discB discA) (on discC discB)
(on discD discC) (clear discD) (clear peg2) (clear peg1)

```

When describing a state, I have left out the (smaller $X Y$) parts. You may do likewise.

Consider the following incomplete partial-order plan. Note that the plan steps, initial state and goal state are numbered for ease of reference, and the numbers do *not* constrain the ordering. Also note that we can specify a link by plan step number and precondition; e.g. 1, (clear DiscC) specifies a link from the initial state to the first move. Finally, note that this plan uses the same domain as above, but a different start state and goal.



4.1 Question 4.1 (5 marks):

Given current constraints, list the numbered entities (plan steps + initial state + goal state) that plan step 4 could appear immediately before.

4.2 Question 4.2 (4 marks):

List the links that plan step 4 threatens, and all ways of resolving those threats.

4.3 Question 4.3 (10 marks):

List the links that plan step 3 threatens, and all ways of resolving those threats.