

Robotics

Control II

Robot control Architectures

- For any reasonably complex task, need control architecture for robot
 - feedback control is fine if
 - there is only one thing the robot needs to do
 - solid reliable error measurement is available
 - most of time more than one thing to do
 - find the goal
 - avoid the walls
 - don't run out of battery
 - don't tip over
 - How do we decide which of this is more important now?

Robot Control Architectures

- Robot Control Architecture
 - Control Architecture
 - set of guiding principles for which goals are more important and when
 - set into program form
 - or burned/wired into hardware.

Languages for programming robots

- No best language for programming robots
 - some use python, why?
 - some robots use C
 - some C++
 - many use several languages depending on the task.

classes of Architecture

- The types of robot Architectures used now and in the past
 - 1)deliberative control
 - 2)reactive control
 - 3)hybrid control
 - 4)Behavior based control
 - many group 2 and 4 together
 - roughly in order of development

Time

- Time is very important in robotics
 - real time reactivity
 - what is “real time”?

Time

- Time is very important in robotics
 - real time reactivity
 - what is “real time”?
 - whatever time frame is needed to get the job done
 - not running off a cliff
 - finding your way in minimal traffic to a nearby town
 - different architectures treat time differently
 - deliberative
 - long time lag/frame
 - reactive
 - all instantaneous time frame
 - hybrid
 - some long time, some instant time tasks.

Modularity

- Modularity important for evaluation of robot architecture
 - how well do the pieces fit together
 - can we replace one with something else that gives similar outputs?

Next

- look at robot paradigms
- fit in representation
-

Search

- Often in planning search is used
 - especially if we have a graph
 - use standard graph search techniques
 - or AI extensions
 - chess branching factor of ~ 20
 - tic tac toe branching factor < 9
 - moving in the world?

Search

- Often in planning search is used
 - especially if we have a graph
 - use standard graph search techniques
 - or AI extensions
 - chess branching factor of ~ 20
 - tic tac toe branching factor < 9
 - moving in the world?
 - depends on how you represent it, but variable branching some quite high.

Search

- complexity of search
 - depth first search running time
 - $O(b^d)$
 - b : branching factor
 - d : depth of graph
 - breadth first search
 - $O(b^d)$
 - space is of course much worse for breadth first
 - depth first not complete.

So we need to optimize

- Optimize search
 - associate costs with each arc in the graph
 - find least cost path
 - what are good cost metrics?

Common Cost Metrics

- distance
- time
- power requirements
- etc
- often best at one is same as best at others
 - but sometimes not. like when?

Planning costs

- for small search graphs
 - search is pretty easy
 - for larger graphs/open worlds
 - search gets costly
- so what?
 - sit and plan?
 - open loop control? act without going through plan all of the time?
 - need to operate in real time

Drawback: time

- Sense data is integrated into the world representation
 - “I robot am here”
 - that is over there
 - etc
- then have to plan
- both of these are expensive.

Space considerations

- Robots are usually limited in hardware
 - representation takes up space
 - big graphs
 - lots of objects to identify
 - even breadth first search space is huge

Information Maintenance

- Back at time we said needed to update representation
 - what if we don't get it all right?
 - representation doesn't match the world
 - some not to be helped
 - accident shuts down the road
 - some needs to be updated
 - construction of science building changes layout of area.

Necessity of plans

- Plans are needed in every cycle of a true deliberative system.
 - slowing robot down