Building Robots That Act Like People

Overall Topic: Robotics, Robot-Human Interaction

Why?

Robots are not good at acting like people, and we may not want them to be good at it! Or do we.

To test models of how people make decisions to act

To improve robot-human interactions
- Get people to enjoy the interaction
- More efficiently communicate from robot -> person and person -> robot
- More efficiently filter information

To improve robot capabilities
- People are good at perceiving, interacting with, and manipulating the world
- "People" perhaps can be more broadly interpreted as "Living creatures"
- If robots do something close to the right thing most of the time that's pretty good

Intelligent robots would let us do things, either as individuals or as a collective entity, that we cannot do now:
- Exploration in hazardous environments
- Exploration in physically unreachable or distant environments
- Assisted living devices

What do robots need to be able to do?

Robots need to be able to do very human things in order to function autonomously in the world
- A robot arm in a manufacturing plant is not a very smart device, in general
- An autonomous mobile robot, or mobile ambulance needs to make decisions on the fly

Intelligent systems must be situated, embodied, and motivated
- Situated: the robot must live in a rich sensory environment and be able to sense it
  - An agent with no sensing ability is a special purpose tool
- Embodied: the robot must be able to modify its environment
  - An agent without embodiment is just an observer (e.g. a web crawler)
- Motivated: the robot must be motivated to interact and modify its environment
  - A robot without motivation is a power tool

So once we give our robots the physical ability to sense and act (sensors & motors), we need to build the brains that give them the ability to perceive (not just sense), manipulate (not just act) and a reason to do both.

Whether the brains are implemented as analog circuits, digital circuits, or computer software, they all implement control loops and algorithms that generate actions based on current perceptions and the robot's inner physical & informational state.

Robotics (broadly) is about all aspects of the robot, but is particularly about how we generate the algorithms that enable robots to do the right thing in a given situation.

Perceive

The world is rich in sensory data, but raw data is not useful

- Perception is about transforming raw data into information, and then into knowledge

One example of the perception process that is relvant for interactive robots is face detection

- What is a face?
  - Color
  - Shape
  - Features and their arrangement
- We can approach face detection from any of these directions, or combine approaches

Color-based detection

- First you need a model of face color
- People have skin, pigment, and blood making up their skin color
  - (For now we discount makeup or other acoutrements)
- It turns out that varying the amount of pigment does not change the normalized color
  - So we want to work in 2D normalized color space: {r, g} = {R, G} / (R + B + G)
- Then we need to geta model of the normalized color of skin
  - Histograms represent arbitrary probability distributions nicely
  - So we generate 2D histograms based on images of people
  - Color camera calibration!
    - or the robot functions under a limited illumination environment)
- Then we need to find skin in the image
  - Use the histogram to generate a probability image from an RGB color image
  - Connect high probability pixels into regions
  - Test the regions for size (and possibly shape)
  - Test the regions for slight gradients (not to small, not too large)
- Return regions that fit the parameters as potential faces (knowledge)
  - Use knowledge about the camera (camara spatial calibration) to identify a ray

So now we have some knowledge, we need to store it and generate an appropriate response.

Another example of a relevant perception process is localization. How does the robot take raw sensor readings and calculate an estimate of where it is in the world?

Particle filters for localization

- Given a map...
- Given a set of sensor readings, what is the probability you are anywhere on the map?
  - There is a probability distribution representing that information
  - We could try to represent it explicity, or implicitly
    - Explicit representations of continuous arbitrary probability functions is hard
    - Implicit representations are computational, but not hard
- Use a particle filter to represent the probability distribution
  - Each particle is a sample of the distribution
  - Each particle represents a potential robot location
  - The particles should gravitate to high probability locations (more interesting)
  - Initial: generate a set of random particles, each of which represents a location
  - Algorithm: take a sensor reading, update the probabilities, resample the particles

Represent

In order to act intelligently, a robot must maintain some kind of state about the world and its own intentionality and actions.

- Brooks argues the world should be its own state (no explicit world state representations)
- Trying to undertake sequences of actions is difficult with also have an internal state
  - The part of the world with the relevant state information may not be perceivable
  - Some actions may not leave perceivable effects on the world
    - Transferring information to another person or robot, for example

Let's continue the example of the robot who has detected a face. Now we have to figure out what to do about it based on our current state (which includes where the robot is located).

It turns out that there are many ways of representing the world and building intentionality into robots. None of them are completely satisfying, and none of them are completely general yet. But some are very common.

State machines:

- State machines are a common method of building both intentionality and complex sequences of actions into robots
  - Each state represents a belief about the state of the world and the robots intentions
  - Output actions are a function of the internal state and the external inputs
  - What the next state will be is also a function of the current state and external inputs
  - State machines can be simple or complex, depending upon the task
    - Most of main loops and sub-loops that represent different sub-tasks
- How about a simple state machine for the task of finding a registration desk
  - Startup state
  - Wander & look state
  - Ask person for help state
  - Sighted registration desk state
  - At registration desk state

So if the robot is wandering and looking and sees a person, it moves to the "ask person" state

Act

Finally, a robot must be able to generate a response that permits it to act intelligently.

Acting intelligently includes how it moves and how it interacts with people (speech/face)

Generating actions can be simple or complex, and the resulting behavior of the robot does not necessarily give away the nature of the underlying process.

Obstacle avoidance:

- Take into account all of the current sensor information (sonar, infrared, bump sensors)
- Generate a push left, right, forward, and backwards based on the sensors
- Generate pushes based on the desired location of the robot (or just give a push forward)
- Combine the pushes into a set of motor commands
- Execute the motor commands

Such subsymbolic action selection may be overriden by a symbolic safety system

Organizing the flow of information

We need to organize the flow of information from perception through representation to action.

- There is a lot of information

Option #1: Monolithic process

- Collect all of the sensory data
- Generate knowledge
- Modify the internal state representations
- Generate an action plan
- Execute the actions

Option #2: Behavior-based architectures

- Create many different processes (virtual or actual)
- Each process collects the raw data it needs
- Each process generates an action plan
- An arbitration method determines which action occurs
  - Subsumption: priority based method where one process chooses
  - Fuzzy controller: merge actions based on weighted priorities

Option #3: Multi-layer architecture (Bryson)

- Bottom layer processes mimic the behavior based architectures and produce action plans
  - Bottom layer processes always have a suggested action based on sensor data
- Top layer processes are symbolic systems that also generate plans
  - Top layer processes may not always have suggested actions
- Middle layer process acts as a filter for determining the final action plan
  - Considers the top and bottom layer inputs
  - Determines the final action plan that gets executed

Multi-layer architectures combine the strengths of symbolic and behavior-based architectures without sacrificing real-time responsiveness or requiring symbolic representations to act.