Introduction to Robotics

Talk the Talk

What is a robot?

"I can't define a robot, but I know one when I see one."
-Joseph Engelberger

A robot is a machine built for real-world functions that is computer-controlled…

…maybe.

Right: Roomba microprocessor (from HowStuffWorks)

Who's to say?

- Many devices with varying degrees of autonomy are called robots.
- Many different definitions for robots exist.
- Some consider machines wholly controlled by an operator to be robots.
- Others require a machine be easily reprogrammable.

Japan?[1]

- *Manual-Handling Device*: controlled by operator
- *Fixed-Sequence Robot*: mechanical action sequence
- *Variable-Sequence Robot*: as 2 but modifiable
- *Playback Robot*: imitates human actions
- *Numerical Control Robot*: run by movement program
- *Intelligent Robot*: reactive to environment

1: Japanese Industrial Robot Association

America and Europe?

- "a programmable, multifunction manipulator…"
  -RIA[2]
- "an independently acting and self controlling machine…"
  -ECM[3]

2: Robotics Institute of America
3: European Common Market

Robot Classes

- *Manipulators*: robotic arms. These are most commonly found in industrial settings.
- *Mobile Robots*: unmanned vehicles capable of locomotion.
- *Hybrid Robots*: mobile robots with manipulators.

(Images from AAAI and HowStuffWorks, respectively)

Robot Components

- Body
- Effectors
- Actuators
- Sensors
- Controller
- Software

Robot::Body

- Typically defined as a graph of *links* and *joints*:

A link is a part, a shape with physical properties.

A joint is a constraint on the spatial relations of two or more links.

Types of Joints

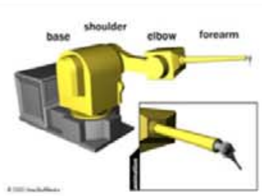
Respectively, a *ball joint*, which allows rotation around x, y, and z, a *hinge joint*, which allows rotation around z, and a *slider joint*, which allows translation along x.

These are just a few examples…

Degrees of Freedom

- Joints constraint free movement, measured in "Degrees of Freedom" (DOFs).
- Links start with 6 DOFs, translations and rotations around three axes.
- Joints reduce the number of DOFs by constraining some translations or rotations.
- Robots classified by total number of DOFs

6-DOFs Robot Arm



How many DOFs can you identify in your arm?

Robot::Effectors

- Component to accomplish some desired physical function
- Examples:
  - Hands
  - Torch
  - Wheels
  - Legs
  - Trumpet?

(Image from http://www.toyota.co.jp/en/special/robot/)

Roomba Effectors

- What are the effectors of the Roomba?

Roomba Effectors

- What are the effectors of the Roomba?

Vacuum, brushes, wheels

Robot::Actuators

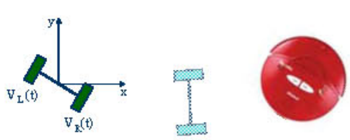
- Actuators are the “muscles” of the robot.
- These can be electric motors, hydraulic systems, pneumatic systems, or any other system that can apply forces to the system.

Roomba Actuators

- The Roomba has five actuators, all electric motors:
  - Two drive wheels
  - One drives the vacuum
  - One drives the spinning side brush
  - One drives the agitator (spinning brush underneath)

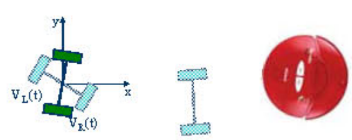
Differential Steering

- The Roomba uses a differential steering system to turn and move forward. Each wheel is controlled by a distinct motor. Here, the Roomba rotates and moves forward.



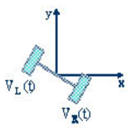
Differential Steering

- The Roomba uses a differential steering system to turn and move forward. Each wheel is controlled by a distinct motor. Here, the Roomba rotates and moves forward.



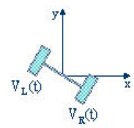
Differential Steering

- The Roomba uses a differential steering system to turn and move forward. Each wheel is controlled by a distinct motor. Here, the Roomba rotates and moves forward.



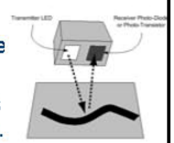
Differential Steering!

- The Roomba uses a differential steering system to turn and move forward. Each wheel is controlled by a distinct motor. Here, the Roomba rotates and moves forward.



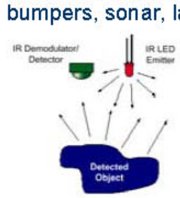
Robot::Sensors

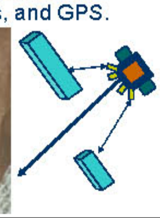
- Allow for perception.
- Sensors can be active or passive
- *Active* – derive information from environment's reaction to robot's actions, e.g. bumpers and sonar.
- *Passive* – observers only, e.g. cameras and microphones .



Sensor Classes

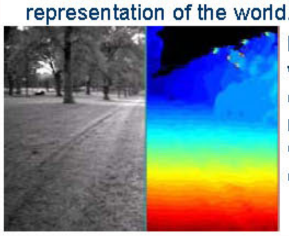
- *Range finders*: these sensors are used to determine distances from other objects, e.g. bumpers, sonar, lasers, whiskers, and GPS.





Sensor Classes

- *Imaging sensors*: these create a visual representation of the world.

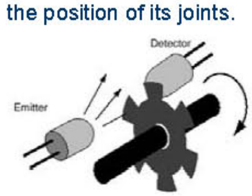


Here, a stereo vision system creates a depth map for a Grand Challenge competitor.

From NOVA, www.pbs.org

Sensor Classes

- *Proprioceptive* sensors: these provide information on the robot's internal state, e.g. the position of its joints.



Shaft decoders count revolutions, allowing for configuration data and *odometry*.

Odometry

- Odometry is the estimation of distance and direction from a previously visited location using the number of revolutions made by the wheels of a vehicle.
- Odometry can be considered a form of "Dead Reckoning*," a more general position estimation based on time, speed, and heading from a known position.

*The Oxford English Dictionary does not recognize "deductive reasoning" as the basis of "dead reckoning"

Odometry

- Odometry is good for short term, relative position estimation.
- However, uncertainty grows, shown by *error ellipses*, without bound.
- This is due to systematic and non-systematic errors.

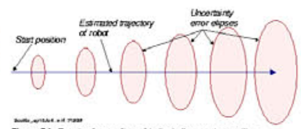


Figure 5.1: Growing "error ellipses" indicate the growing position uncertainty with odometry. (Adapted from [Tonouchi et al., 1994].)

Odometry, Non-systematic Errors

- These errors can rarely be measured and incorporated into the model.
- Error causes include uneven friction, wheel slippage, bumps, and uneven floors.

Odometry, Systematic Errors

- Errors arising from general differences in model and robot behavior that can be measured and accounted for in the model, a process known as *calibration*.
- Two primary sources:
  - Unequal wheel diameters – lead to curved trajectory
  - Uncertainty about wheel base – lead to errors in turn angle

Odometry, Position Updates

- With calibration, model behavior becomes more similar to observed behavior. However, estimation uncertainty still grows without bound.
- Position updates reduce uncertainty.

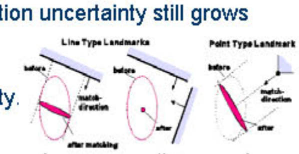


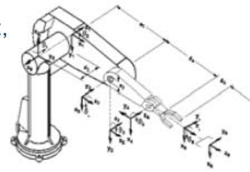
Figure 8.10: Different features can reduce the size of the robot's uncertainty ellipse in one or two directions.
a, c: Walls and corners reduce uncertainty in one direction
b.: Two adjacent walls at right angles reduce uncertainty in two directions.
(Courtesy of [Bauer and Rencken, 1995]).

Kinematics

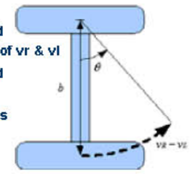
- The calculation of position via odometry is an example of *kinematics*.
- Kinematics is the study of motion without regard for the forces that cause it.
- It refers to all time-based and geometrical properties of motion.
- It ignores concepts such as torque, force, mass, energy, and inertia.

Forward Kinematics

- Given the starting configuration of the mechanism and joint angles, compute the new configuration.
- For a mechanism robot, this would mean calculating the position and orientation of the end effector given all the joint variables.



Kinematics of Differential Steering



Derivation:

$dx/dt = v(t)\cos(\theta(t))$ **X component of speed**

$dx/dt = [(v_R - v_L)/2]\cos(\theta(t))$ **Speed is average of vr & vl**

$dy/dt = v(t)\sin(\theta(t))$ **Y component of speed**

$dy/dt = [(v_R + v_L)/2]\sin(\theta(t))$

$d\theta/dt = (v_R - v_L)/b$ **Arc change over radius**

Integrate all:

$\theta(t) = (v_R - v_L)t/b - \theta_0$

$x(t) = x_0 + \frac{b(v_R + v_L)}{2(v_R - v_L)}\left[\sin((v_R - v_L)t/b + \theta_0) - \sin(\theta_0)\right]$

$y(t) = y_0 - \frac{b(v_R - v_L)}{2(v_R - v_L)}\left[\cos((v_R - v_L)t/b + \theta_0) - \cos(\theta_0)\right]$

This is the turn radius for a circular trajectory: $(b/2)(v_R + v_L)/(v_R - v_L)$

Kinematics of Differential Steering

- The above model has an asymptote when

$$v_R - v_L \approx 0$$

- When this occurs, special handling is required.
- Or a simpler model can be used:

$\bar{s} = (s_R + s_L)/2$

$\theta = (s_R - s_L)/b + \theta_0$

$x = \bar{s}\cos(\theta) + x_0$

$y = \bar{s}\sin(\theta) + y_0$

Here, SR and SL are measured right and left velocities. This approximates movement as a "point-and-shoot."

Kinematics of Differential Steering

- Simpler approximations are often used when onboard computing power is lacking (or programmers are lazy!).
- However, the error grows quicker.
- A slightly better approximation:

$\bar{s} = (s_R + s_L)/2$

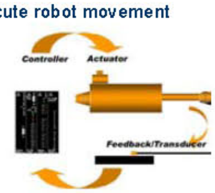
$\theta = (s_R - s_L)/2b + \theta_0$

$x = \bar{s}\cos(\theta) + x_0$

$y = \bar{s}\sin(\theta) + y_0$

Robot::Controller

- Controllers direct a robot how to move.
- There are two controller paradigms
  - Open-loop controllers execute robot movement without feedback.
  - Closed-loop controllers execute robot movement and judge progress with sensors. They can thus compensate for errors.



Controller, Open-loop

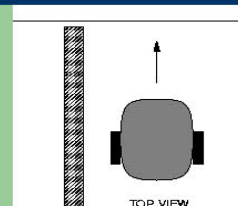
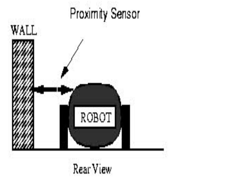


Figure 11.1: Driving along a Wall Edge

- Goal: Drive parallel to the wall.
- Feedback: None.
- Result: Noisy movement, due to slippage, model inaccuracy, bumps, etcetera is likely to cause the robot t veer off the path.

Controller, Closed-loop



Using a Proximity Sensor to Measure Distance to a Wall

- Goal: walk parallel to the wall.
- Feedback: a proximity sensor
- Result: the robot will still veer away or toward the wall, but now it can compensate.

Trajectory Error Compensation

- If a robot is attempting to follow a path, it will typically veer off eventually. Controllers design to correct this error typically come in three types:
  - *P controllers* provide force in negative **p**roportion to measured error.
  - *PD controllers* are P controllers that also add force proportional to the first derivative of measured error.
  - *PID controllers* are PD controllers that also add force proportional to the integral of measured error.

Roomba Control

- The movement of the Roomba can be hard-coded ahead of time as an example of open-loop control.
- A path can be converted to Roomba wheel movement commands via *inverse kinematics*.

Inverse Kinematics

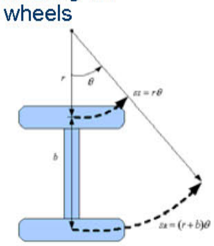
- Inverse Kinematics is the reverse of Forward Kinematics. (!)
- It is the calculation of joint values given the positions, orientations, and geometries of mechanism's parts.
- It is useful for planning how to move a robot in a certain way.

Kinematics^{-1} of Differential Steering

- Vehicles using differential steering will go in a straight line if both wheels receive the same power.
- If both wheels turn at constant, but different, speeds, the vehicle follows a circular path
- Distances traveled: $s_L = r\theta$

  $s_R = (r + b)\theta$

  $s_M = (r + b/2)\theta$



Kinematics^{-1} of Differential Steering

- This calculation ignores acceleration, but it can be used to calculate how to move a device using a differential steering system, such as a Roomba, along a path that consists of lines and arcs.

Potential Field Control

- Potential field control is similar to the hill-climbing algorithm.
- Given a goal position in a space, create an impulse to go from any position in the space toward the goal position.
- Add Repulsive forces wherever there are obstacles to be avoided.
- This does not require path planning.

Potential Field Soccer

- 1 moves toward the blue goal.
- 1 avoids 7, 6, and 8.
- Teammates generate attractive fields.

(image from http://www.itee.uq.edu.au/~dball/roborops/about_robots.html)

Reactive Control

- Given some sensor reading, take some action.
- This is the robotics version of a *reflex agent design*.
- It requires no model of the robot or the environment.
- Maze exiting:
  - Keep Moving forward.
    - If bump, turn right.

Robot::Software Architecture

- Previous control methods include *deliberative methods* and *reactive methods*.
  - Deliberative methods are model-driven and involve planning before acting.
  - Reactive methods is sensor-driven and behavior must emerge from interaction.
- *Hybrid architectures* are software architectures combining deliberative and reactive controllers.
  - An example is path-planning and PD control.

Three-Layer Architecture

- The most popular hybrid software architecture is the three-layer architecture:
  - *Reactive layer* – low-level control, tight sensor-action loop, decisions cycles (DCs) order of milliseconds.
  - *Executive layer* – directives from deliberative layer sequenced for reactive layer, representing sensor information, localization, mapping, DCs order of seconds.
  - *Deliberative layer* – generates global solutions to complex tasks, path planning, model-based planning, analyze sensor data represented by executive layer, DCs order of minutes.

Robot Ethics

Asimov's Three^H^H^H^H^H Four Laws:

0th) A robot may not harm humanity, or, by inaction, allow humanity to come to harm.

1st) A robot may not injure a human being or, through inaction, allow a human being to come to harm.

2nd) A robot must obey orders given it by human beings except where such orders would conflict with the First Law.

3rd) A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

(Image from http://www.bmc.riken.jp/%7ERI-MAN/index_jp.html)