ENGINEERING 5

Lecture 8: Forward and Inverse Kinematics &
Servo Motors

Professor Erik Cheever

Course web page:
http://www.swarthmore.edu/NatSci/echeever1/Class/e5/E5Index.html

E5 Lab is in Hicks 213, **not** in Trotter

Remember…

- Thursday 10/29: 3rd MatLab is due (Animating T puzzle)
- Thursday 11/5: 4th MatLab is due (robot prelims – this week's lab)
- Thursday 11/12: Demonstrate robot arm
- Wizards: Tuesday 10/27, 8:00-10:00 p.m & Wednesday 10/28, **7:00-10:00 p.m**
  Don't wait until the last minute!
- Complete learning styles inventory and record results
- Start thinking about your project (last 3 weeks of class). Some ideas are at "Projects" link on left side of course web page page (including last year's projects).
- You need to have a project picked out by November 12th.

- E5 Lab is in Hicks 213, **not** in Trotter

The big picture…

- Today:
  - Forward kinematics for Robot arm
  - Inverse kinematics for Robot arm
  - Controlling servo motors
  - Professor Fred Orthlieb, Mechanical Engineering
- Subsequent weeks: controlling motors, and designing, build, and program both a robotic arm with laser pointer.

- Projects!!!

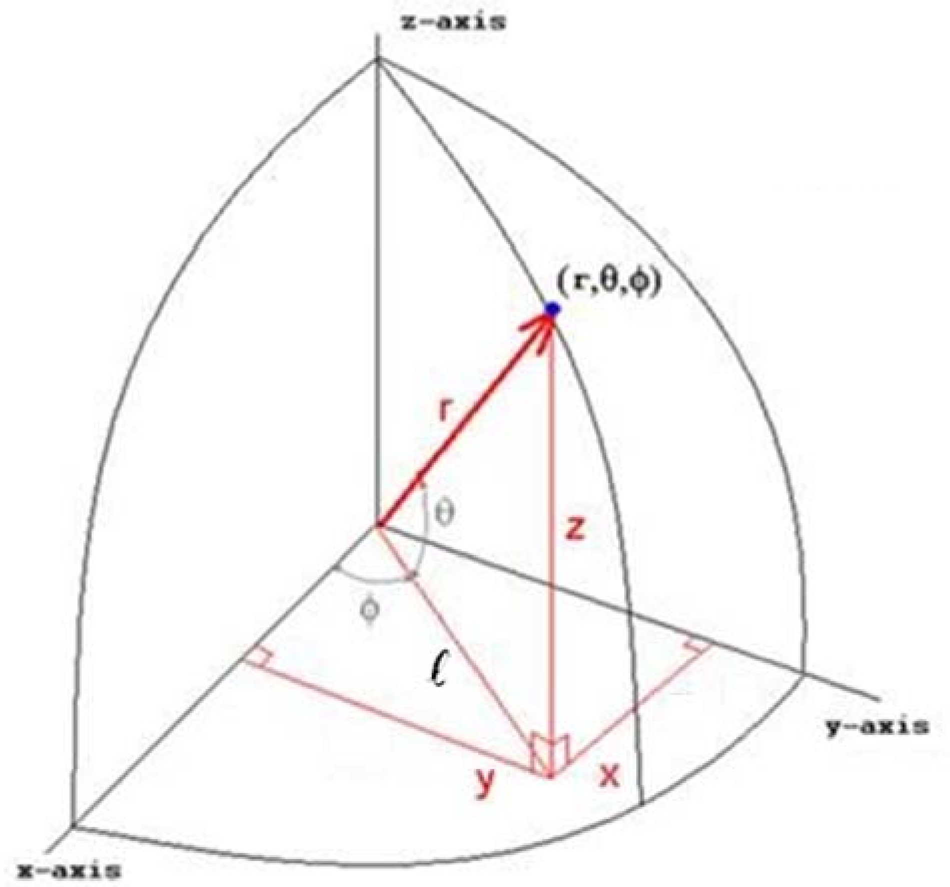
- E5 Lab is in Hicks 213, **not** in Trotter

Forward and Inverse Kinematics

Mathematical description of the robot arm geometry

E5 Lab is in Hicks 213, **not** in Trotter

Spherical ⟶ Cartesian Coordinates



Adapted from: http://rbrundritt.spaces.live.com/blog/cns!E7DBA9A4BFD458C5!280.entry

$$z = r\sin(\theta)$$

$$\ell = r\cos(\theta)$$

$$x = \ell\cos(\phi) = r\cos(\theta)\cos(\phi)$$

$$y = \ell\sin(\phi) = r\cos(\theta)\sin(\phi)$$

Given r, θ and φ we can easily find x, y and z. (Forward kinematics)

Given x and y, we could find θ and φ with some difficulty.
(Inverse kinematics)

… but this is not the problem we want to solve.

2-D Polar ⟶ Cartesian w/ offset

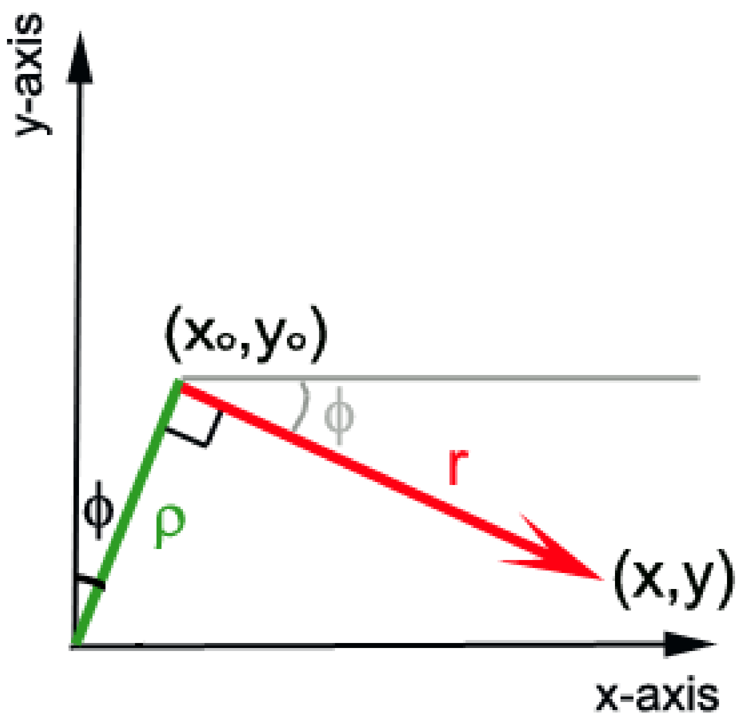
$$x_0 = \rho \sin(\phi)$$

$$y_0 = \rho \cos(\phi)$$

Angle of "r" from horizontal is equal to φ.

$$x = x_0 + r\sin(\phi) = \rho\sin(\phi) + r\sin(\phi)$$

$$y = y_0 + r\sin(\phi) = \rho\cos(\phi) + r\sin(\phi)$$



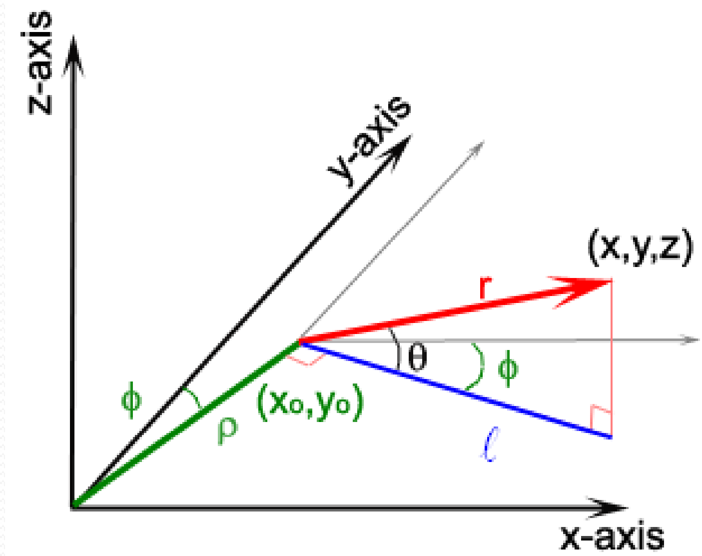
Forward Kinematics (1)

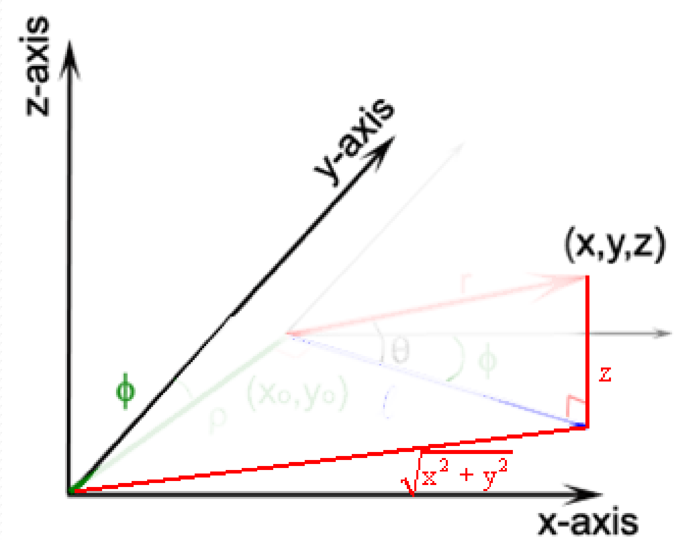
We know θ, φ and ρ.
We want to know x, y, z

Pythagoras tells us: $x^2 + y^2 + z^2 = \rho^2 + r^2$

$$r = \sqrt{x^2 + y^2 + z^2 - \rho^2}$$

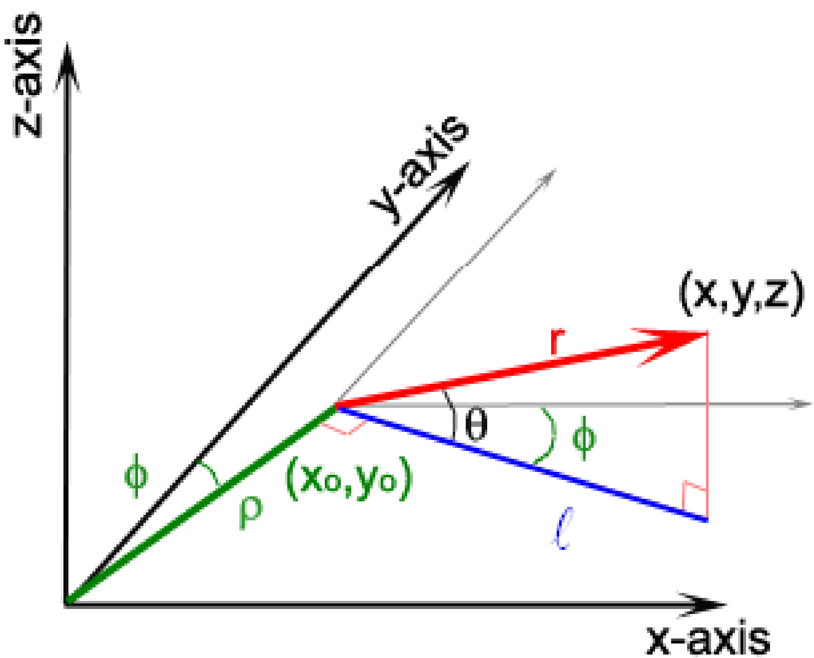
$$z = r\sin(\theta)$$





Forward Kinematics (2)

We know θ, φ and ρ.
We want to know x, y, z



$$x_0 = \rho \sin(\phi)$$

$$y_0 = \rho \cos(\phi)$$

$$x = x_0 + \ell \cos(\theta)$$

$$x = \rho \sin(\phi) + \ell \cos(\phi)$$

$$x = \rho \sin(\phi) + r \cos(\theta) \cos(\phi)$$

$$y = y_0 - \ell \sin(\theta)$$

Note sign in this expression for y

$$y = \rho \cos(\phi) - \ell \sin(\phi)$$

$$y = \rho \cos(\phi) - r \cos(\theta) \sin(\phi)$$

Inverse Kinematics (1)

So… using forward kinematics we can determine x, y and z, given the angles φ and θ and the length of the arm, ρ.

$$r = \sqrt{x^2 + y^2 + z^2 - \rho^2} \qquad x = \rho\sin(\phi) + r\cos(\theta)\cos(\phi)$$

$$z = r\sin(\theta) \qquad y = \rho\cos(\phi) - r\cos(\theta)\sin(\phi)$$

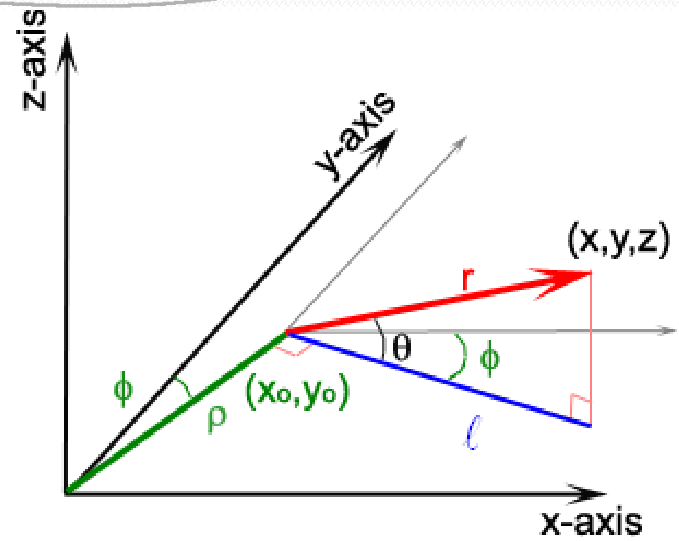
But… forward kinematics is not enough. Generally with a robot, we know where we want the robot to be (x,y), and need to find the angles.

This process is called inverse kinematics.

Problem statement: we know x, y, and z (these are inputs) and we know ρ (determined by geometry of robot).

We want to find φ and θ.

Inverse Kinematics (2)



We know x, y, z and ρ.

Solving for θ. (this is relatively easy)

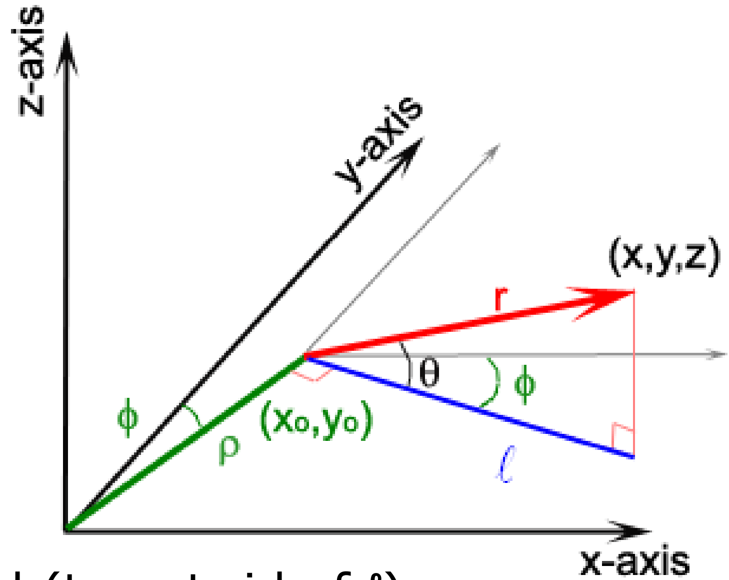
We know: $z = r\sin(\theta)$

$$r = \sqrt{x^2 + y^2 + z^2 - \rho^2}$$

So: $$\theta = \arcsin\left(\frac{z}{r}\right)$$

…and we have solved for one of our two angles.

Inverse Kinematics (3)



We know x, y, z and ρ.

Solving for φ. (this is harder)

Start with: $x = \rho\sin(\phi) + \ell\cos(\phi)$

$$y = \rho\cos(\phi) - \ell\sin(\phi)$$

Multiply x by sin(φ) and y by cos(φ) and add (to get rid of ℓ):

$$x\sin(\phi) = \rho\sin^2(\phi) + \ell\cos(\phi)\sin(\phi)$$

$$y\cos(\phi) = \rho\cos^2(\phi) - \ell\sin(\phi)\cos(\phi)$$

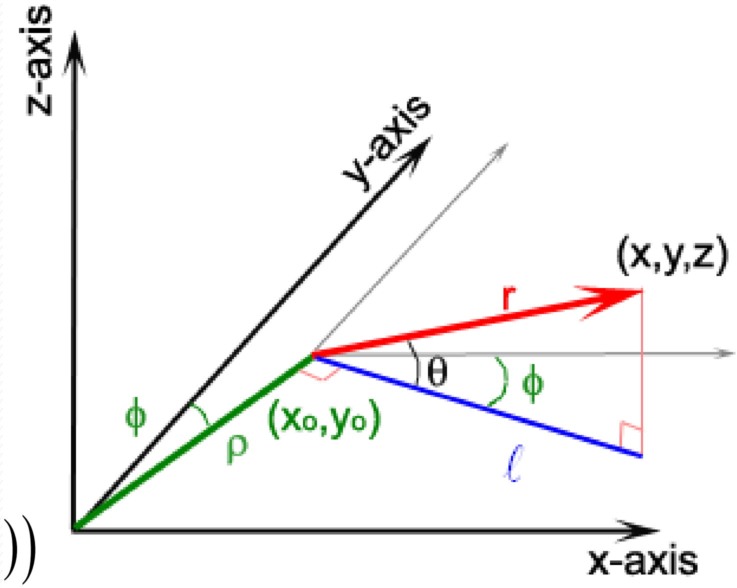
$$x\sin(\phi) + y\cos(\phi) = \rho\sin^2(\phi) + \cancel{\ell\cos(\phi)\sin(\phi)} + \rho\cos^2(\phi) - \cancel{\ell\sin(\phi)\cos(\phi)}$$

$$x\sin(\phi) + y\cos(\phi) = \rho\left(\sin^2(\phi) + \cos^2(\phi)\right)$$

$$x\sin(\phi) + y\cos(\phi) = \rho$$

We're almost done…

Inverse Kinematics (4)



… continued from previous

$$x\sin(\phi) + y\cos(\phi) = \rho$$

Use trigonometric identity:

$$a\sin(\alpha) + b\cos(\alpha) = \sqrt{a^2 + b^2}\sin(\alpha + \mathrm{atan2}(b, a))$$

$$a = x \qquad b = y \qquad \alpha = \phi$$

$$x\sin(\phi) + y\cos(\phi) = \sqrt{x^2 + y^2}\sin(\phi + \mathrm{atan2}(y, x)) = \rho$$

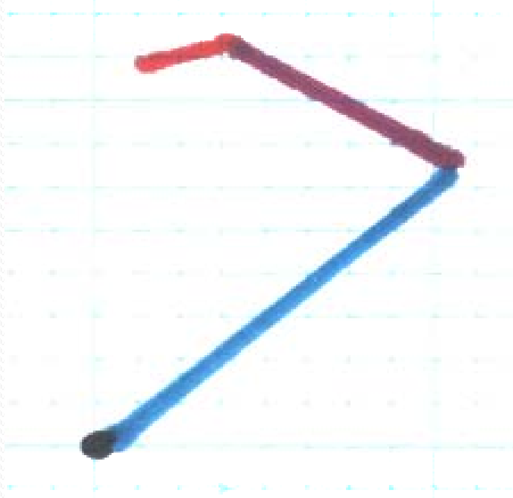
$$\sin(\phi + \mathrm{atan2}(y, x)) = \frac{\rho}{\sqrt{x^2 + y^2}}$$

$$\phi + \mathrm{atan2}(y, x) = \mathrm{asin}\left(\frac{\rho}{\sqrt{x^2 + y^2}}\right)$$

so…

$$\phi = \mathrm{asin}\left(\frac{\rho}{\sqrt{x^2 + y^2}}\right) - \mathrm{atan2}(y, x)$$

What if we add a third joint?



Are there any difficulties to the forward kinematics problem?

Are there any difficulties to the reverse kinematics problem?

Inverse kinematics is generally harder than forward kinematics.

Moving the Robot Arm

A Brief Introduction to Servo Motors

E5 Lab is in Hicks 213, **not** in Trotter

Servo Motors (1)

- Output shaft of motor turns to angle specified by input pulses.
- We send a stream of pulses to servo through wires connected to it.
- Width of pulses defines angle of output shaft.

http://www.lynxmotion.com/images/Products/Full/hsr-5980.jpg

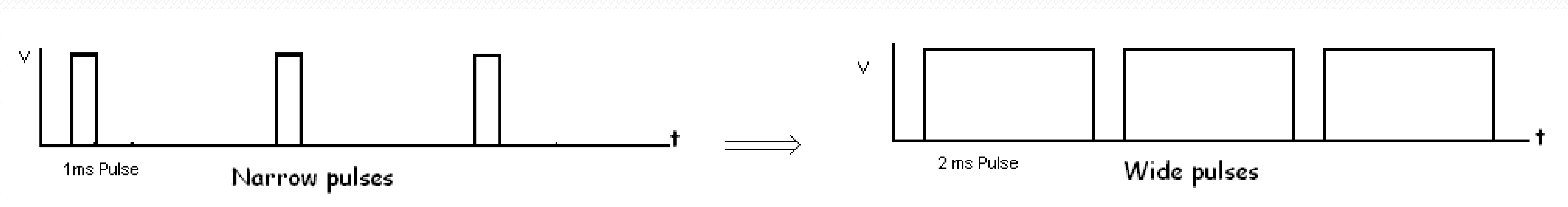
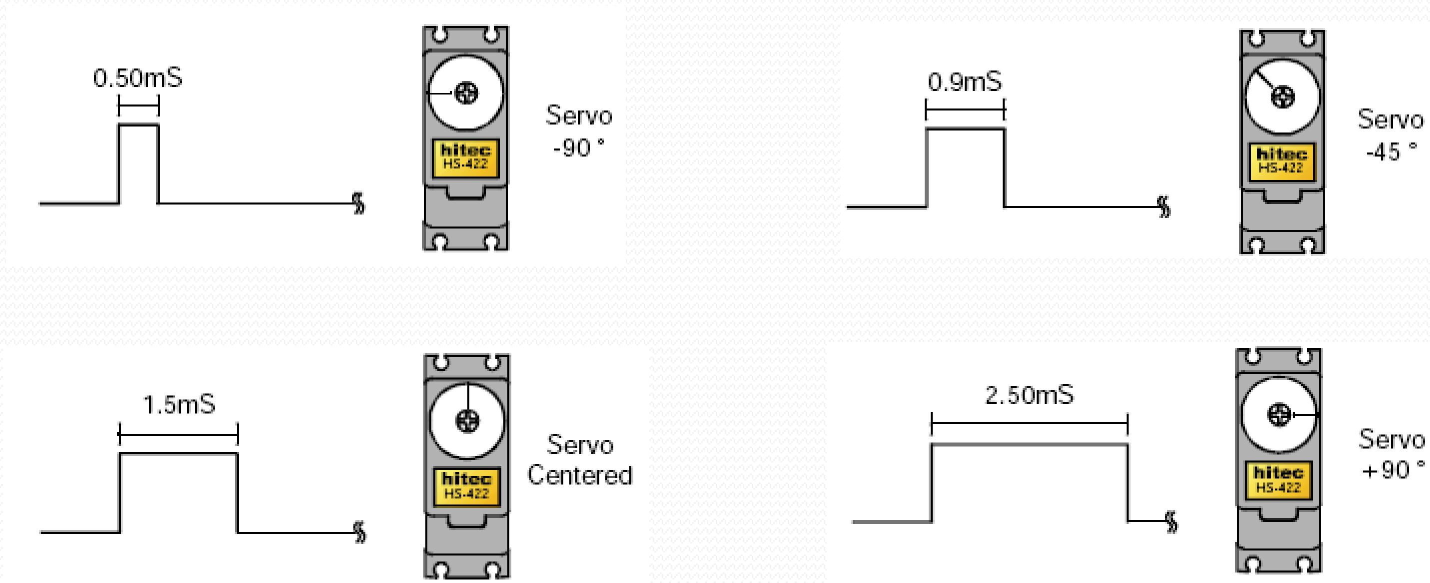


Image adapted from: http://www.electronicsteacher.com/robotics/images/pulsewidthdiagram.gif

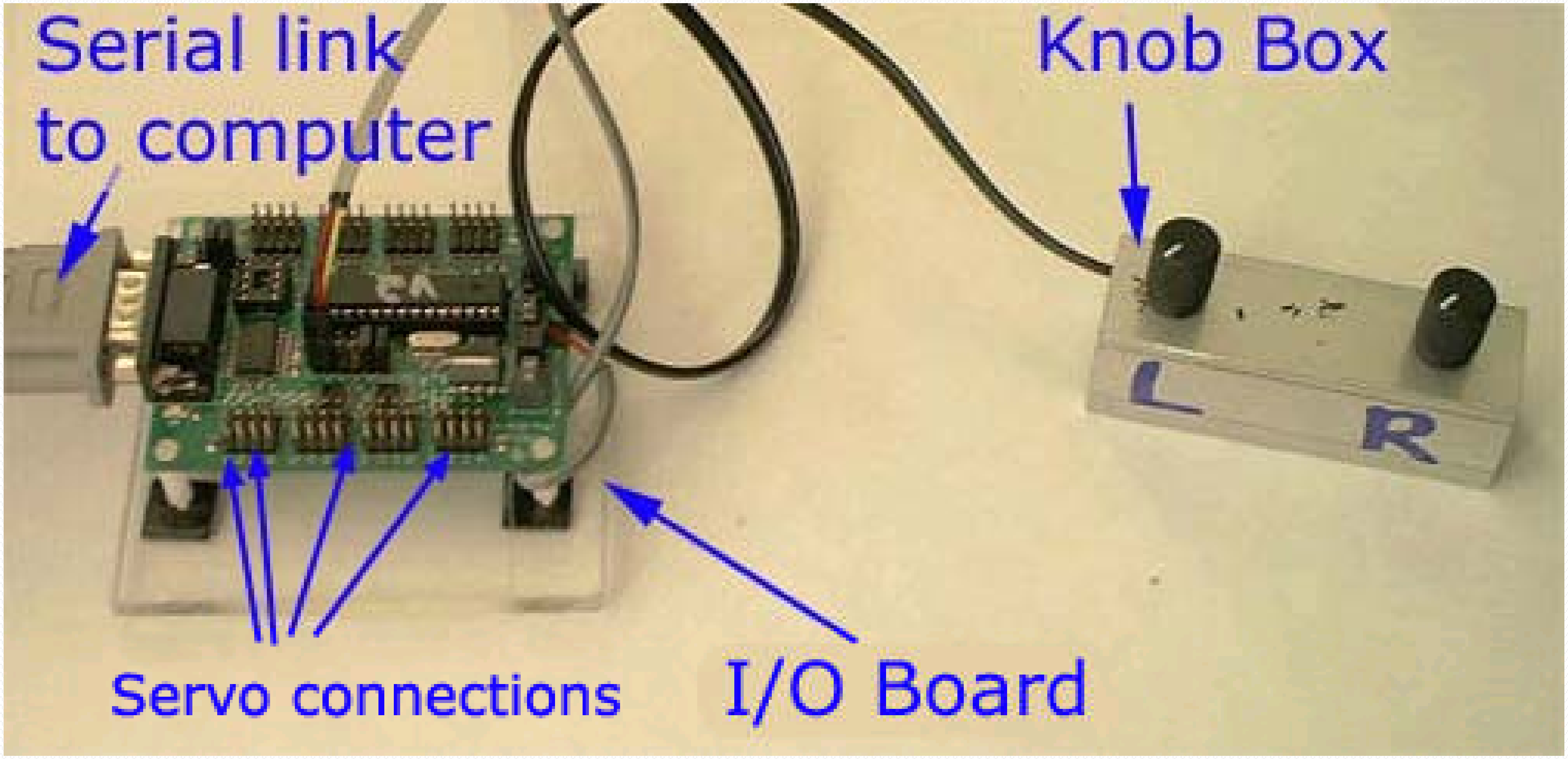
Servo Motors (2)

Relatively precise, but not accurate



Images adapted from: http://www.lynxmotion.com/images/data/ssc-32.pdf

Connecting the Servo



Computer sends commands to I/O Board, which sends pulses to motors.

Controlling the Servo

- Open a serial port connection
- Send a string of characters to the I/O board
- The I/O board controls the motors by varying the width of pulses sent to the servo

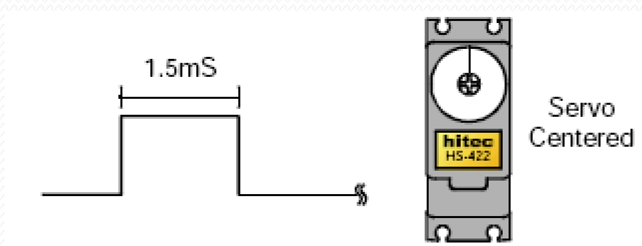
Control Format

- #<ch>P<pw>...#<ch>P<pw>T<time><cr>
- <ch> = channel number (0-31)
- <pw> = pulse width in microseconds (500-2500)
- <time> = time in milliseconds for move (optional)
- <cr> = carriage return http://www.youtube.com/watch?v=Pp2IJIk5qvc

Control Format Examples

#<ch>P<pw>...#<ch>P<pw>T<time><cr>

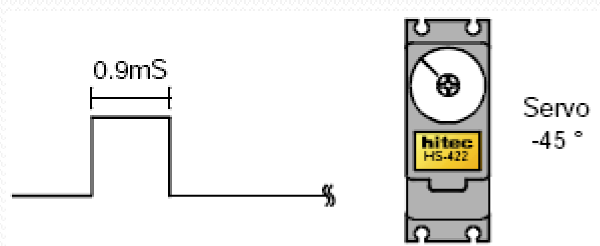
- Sending the string: '#1P1500T200<cr>'
  - ch=1, pw=1500, time=200
  - This will set the pulsewidth on motor 1 to 1500 microseconds, and will make the change over 200 milliseconds. (1500 ms ≈ 0°)



Another Control Format Examples

#<ch>P<pw>...#<ch>P<pw>T<time><cr>

- Sending the string: ‘#1P900T300<cr>’
  - ch=1, pw=900, time=300
  - This will set the pulsewidth on motor 1 to 900 microseconds, and will make the change over 300 milliseconds. (900 ms ≈ -45°)



Control Format Example Again

#<ch>P<pw>...#<ch>P<pw>T<time><cr>

- Sending the string: '#1P1200#5P2100T1500<cr>'
  - ch=1, pw=1200
ch=5, pw=2100
time=1500
  - This will set
    - the pulsewidth on motor 1 to 900 microseconds,
    - the pulsewidth on motor 5 to 2100 microseconds,
    - and will make the change over 1500 milliseconds.

Some MatLab Background

Open a serial connection (done only once).

```
s=instrfind;    %Find any serial links (we can have only 1)
delete(s);      %... and delete.

%Create a new serial communications link
s=serial('COM1','Baudrate',115200,'Terminator','CR');
fopen(s);                                  %... and open it
```

Send a string to I/O board.

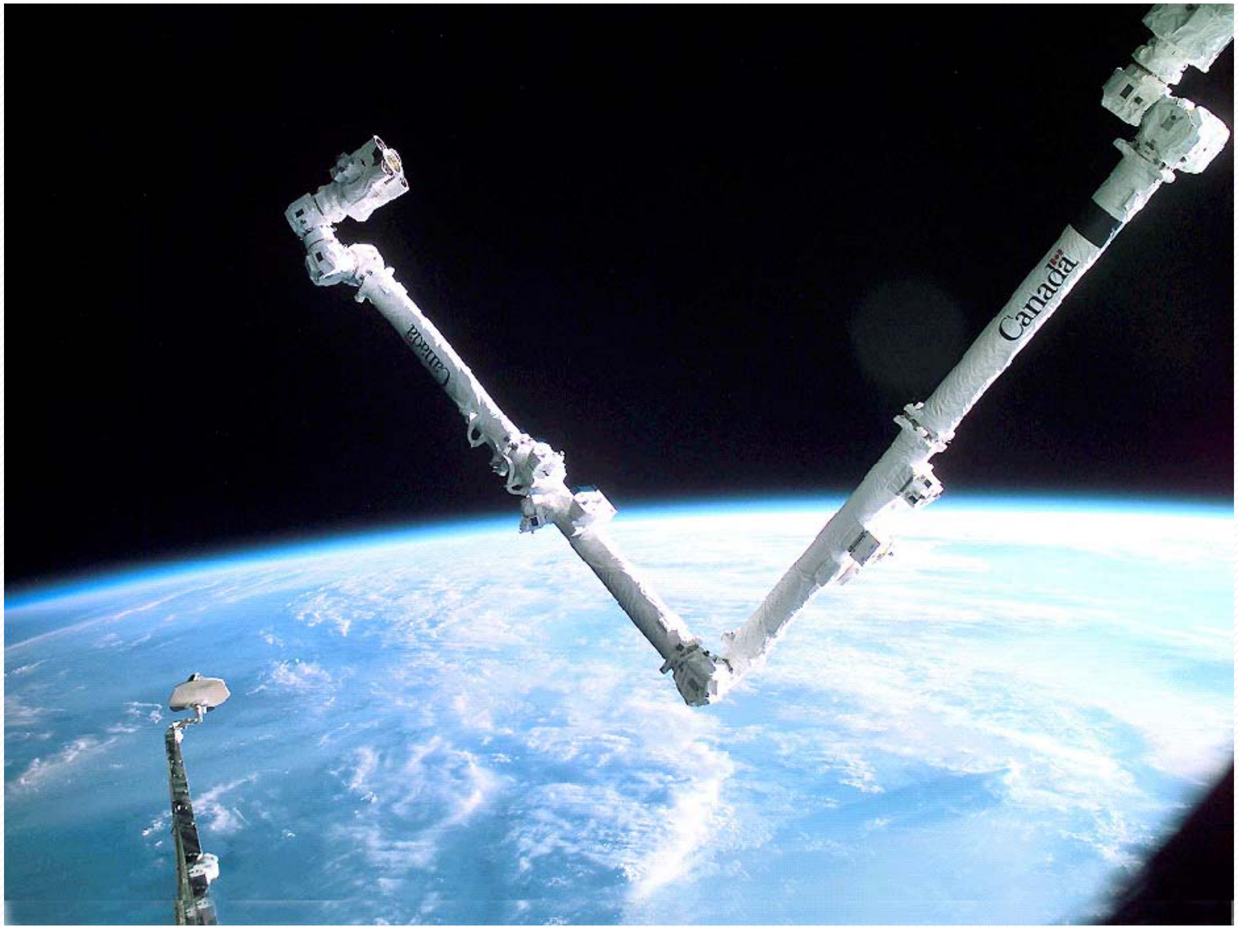
```
% set pulsewidth on motor 1 to 1000 microseconds (over 200 mS)
fprintf(s,‘#1P1000T200');
```

So… what?

- Given the desired y, z position of the laser we can calculate the necessary angles.
- Given the angles, we can determine the appropriate pulse widths and turn our motors
- So… we can direct the laser beam at will.

- This is your task for the next two lab periods (and probably a bit more time) – we will demonstrate them in lab in two weeks
- Lab is in Hicks 213, **not** in Trotter!!!!!!!!!!!!!!!

Swarthmore alums in robotics (2)



http://spaceflight.nasa.gov/gallery/video/shuttle/sts-111/mpg/sts111ani7a.mpg

Swarthmore alums in robotics (3)

Heather writes:

- We use:
  - Forward kinematics – someone has delivered a set of 7 joint angles to our group to use for analysis – where does that place the end of the arm?
  - Inverse kinematics – we know we want the LEE on the end of the arm at a certain position and orientation – how do we need to set the joint angles in the simulation to place the LEE there?
  - Matrix math – converting between reference frames such as inertial, space station structural frame, free flying vehicle structural frame, end effector frame, etc.
  - Matlab – great for writing scripts to analyze large batches of data, or for making plots to visualize our results and present them to management.

Swarthmore alums in robotics (4)

Ani Hsieh, class of 1999. Swarm Robotics.

http://kumar.cis.upenn.edu/movies/icra08_manipulate.flv

Chris Lee, class of 1993. Big Dog.

http://www.youtube.com/watch?v=cHJJQ0zNNOM

Friday the 13th and MatLab

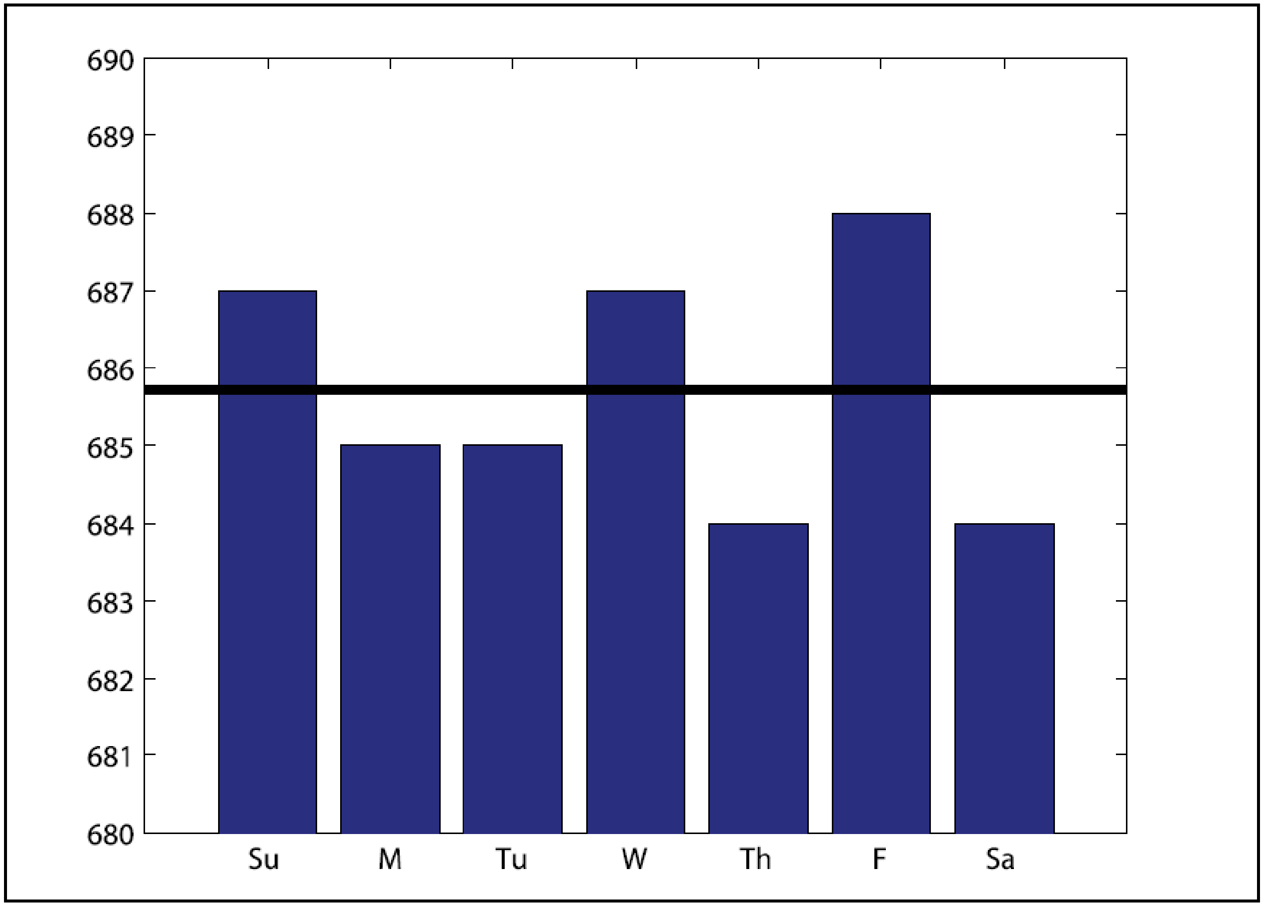


FIGURE 1. *MATLAB plot showing that the 13th day of a month is more likely to be a Friday than any other day of the week.*

http://www.mathworks.com/company/newsletters/news_notes/2008/news-notes-en.pdf