Lecture 7: Even More MatLab, &
Forward and Inverse Kinematics

Professor Erik Cheever

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

- Thursday 10/22: 2nd MatLab lab is due (Solving the T puzzle).

- Thursday 10/29: 3rd MatLab lab is due (Animating the T puzzle – this week’s lab)

- Tuesday & Wednesday, 10/20 & 10/21: Wizards available (Trotter 201) from 8:00-10:00 p.m. specifically for E5. Don’t wait until the last minute!

- 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).
A volunteer opportunity

Professor Macken works with kids from Chester on Engineering projects (bridges in the fall, solar cars in the spring).

You must have Wednesday afternoons after 3:45 free.

It begins this Wednesday and goes 4 weeks.

Contact Professor Macken, nmacken1, x8073
A volunteer opportunity
The big picture...

- Today:
  - Control of flow in MatLab
  - Forward kinematics for Robot arm
  - Inverse kinematics for Robot arm
  - Professor Lynne Molter, Electrical Engineering
- Read sections 4.3.4-5 and 5.2.1-3 of text
- Following three weeks: controlling motors, and designing, build, and program both a robotic arm with laser pointer.

- Last three weeks of class – individual projects. Some ideas on “Projects” link on left side of course web page (including last year’s projects).
Control of flow in MatLab

- So far we have used MatLab as a fancy calculator
- The real power comes when we can have MatLab:
  - Perform operations many times consecutively:
    - for... loop
  - Make decisions:
    - if...then...else...
    - if...then...elseif...then...elseif... ... else...
    - while... loop
The for... loop (1)

Syntax:

```
for variable=start:finish,  
  ...commands...  
end
```

Example:

```
for i=1:10,  
  disp(i)  
end
```
The for... loop (2)

Syntax:
for variable=start:increment:finish,
  ...commands...
end

Example:
for i=1:4:10,
  disp(i)
end

Output:
  1
  5
  9
For loop example (0)

First define a shape:

```matlab
%% Define a shape
theta=0:10:360;   %theta is spaced around a circle (0 to 360).
r=1;              %The radius of our circle.
%Define a circular magenta patch.
x=r*cosd(theta);
y=r*sind(theta);
myShape=patch(x,y,'m');
axis(20*[-1 1 -1 1],'square');  grid on;
```
Make the shape move:

```matlab
T=0.25;                         %Delay between images
for x0=-15:2.5:15,
    set(myShape,'Xdata',x0+x); %Translate by x0
    pause(T);                 %Wait T seconds
end
```

For loop example (1)
For loop example (2)

Make the shape move (another method):

```matlab
%% Move it across the screen (different technique)
T=0.25;                           %Delay between images
x0=linspace(-15,15,20);

for i=1:length(x0),
    set(myShape,'Xdata',x0(i)+x);  %Translate by x0
    pause(T);                      %Wait T seconds
end
```
For loop example (2)

Make the shape move (another method):

```matlab
T=0.25; % Delay between images
x0=linspace(-15,15,20);

for i=1:length(x0),
    set(myShape,'Xdata',x0(i)+x); % Translate by x0
    pause(T); % Wait T seconds
end
```

Note: This week in lab you will use the “RotTrans” function (from last lab) to animate the parts of the T puzzle as they move into place.
For loop example (3)

Make the shape move by giving it velocity:

```matlab
%% Move it by giving it a velocity in x
vx=5;       % velocity in meters/sec.
dt=0.25;    % time step

x0=-10;
for t=0:dt:4,
    set(myShape,'Xdata',x+x0);        % Translate by x0
    pause(dt);                        % Wait dt seconds
    x0=x0+vx*dt;                      % Update x position
end
```
For loop example (4)

Give it velocity in two directions:

```matlab
%% Move it by giving it a velocity in x and y
vx=5;       %velocity in meters/sec.
vy=10;
dt=0.25;    %time step

x0=-10;
y0=-10;
for t=0:dt:3,
    set(myShape,'Xdata',x+x0,'Ydata',y+y0); %Translate by x0,y0
    pause(dt); %Wait dt seconds
    x0=x0+vx*dt;       %update x position
    y0=y0+vy*dt;       %update y position
end
```
For loop example (5)

Add gravity:

```matlab
vx=5;   %velocity in meters/sec.
vy=15;
dt=0.2; %time step

x0=-15;
y0=0;
for t=0:dt:4,
    set(myShape,'Xdata',x+x0,'Ydata',y+y0);   %Translate by x0, y0
    pause(dt); %Wait dt seconds
    x0=x0+vx*dt;      %update x position
    y0=y0+vy*dt;      %update y position
    vy=vy-9.8*dt;     %update y velocity
end
```

end
Making Decisions in Matlab: **if**

**Syntax:**

```matlab
if expression,
    ... statements ...
end
```

*expression* has to evaluate to TRUE or FALSE

**Example:**

```matlab
if a>b,
    disp('a is greater than b');
end

if (a>b) & (a>0),
    disp('a>b and a is positive');
end
```
Logical expressions

(expression has to evaluate to TRUE or FALSE

- \(a > b\) TRUE when \(a\) is \textit{greater} than \(b\), otherwise FALSE
- \(a < b\) TRUE when \(a\) is \textit{less} than \(b\), otherwise FALSE
- \(a \geq b\) TRUE when \(a\) is \textit{greater} than or equal to \(b\), otherwise FALSE
- \(a = b\) TRUE when \(a\) is \textit{equal} to \(b\), otherwise FALSE
- \(a \neq b\) TRUE when \(a\) is \textit{not} equal to \(b\), otherwise FALSE
- \((a > 0) \text{ and } (b > 0)\) TRUE when \(a\) is greater than 0 \textit{and} \(b\) is greater than zero, otherwise FALSE
- \((a > 0) \text{ or } (b > 0)\) TRUE when \(a\) is greater than 0 \textit{or} \(b\) is greater than zero, otherwise FALSE
- \((a > 0) \text{ or } \sim(b > 0)\) TRUE when \(a\) is greater than 0 \textit{or} \(b\) is \textit{not} greater than zero, otherwise FALSE

Be careful if \(a\) or \(b\) are matrices, surprising results can occur!

Note: this list is not exhaustive, check your text for more.
Making Decisions: \texttt{if} \ ... \ \texttt{else}

\textbf{Syntax:}
\begin{verbatim}
if expression,
    ... statements ...
else
    ... statements
end
\end{verbatim}

\textbf{Example:}
\begin{verbatim}
if a>b,
    disp('a is greater than b');
else
    disp('a is less than or equal to b');
end
\end{verbatim}
Making Decisions: \textbf{if} \ldots \textbf{elseif} \ldots \textbf{else}

\textbf{Syntax:}
\begin{verbatim}
if expression1,
    ... statements ...
elseif expression2,
    ... statements ...
else
    ... statements ...
end
\end{verbatim}

\textbf{Example:}
\begin{verbatim}
if a>b,
    disp('a is greater than b');
elseif a<b,
    disp('a is less than b');
else
    disp('a is equal to b');
end
\end{verbatim}
Example: if...elseif...else (1)

```matlab
plot([-15 -15 15 15 -15],[15 -15 -15 15 15],'b','LineWidth',2);
for t=0:dt:10,
    set(myShape,'Xdata',x+x0,'Ydata',y+y0); %Translate by x0, y0
    pause(dt); %Wait dt seconds
    x0=x0+vx*dt; %update x position
    y0=y0+vy*dt; %update y position
    vy=vy-9.8*dt; %update y velocity
    if x0>=15-r, %check for collision with right side
        vx=-vx; %if so, reverse x velocity
        x0=15-r; %reset x position so it touches wall
    elseif x0<=-15+r, %check for collision with left side...
        vx=-vx;
        x0=-15+r;
    end
    if y0>=15-r, %check for top
        vy=-vy;
        y0=15-r;
    elseif y0<=-15+r, %check for bottom
        vy=-vy;
        y0=-15+r;
    end
end
```
Example: if...elseif...else

\[ p = 0.7; \quad \text{% coefficient of restitution} \]

\[
\text{for } t=0:dt:15, \\
\quad \text{set(myShape,'Xdata',}x+x0,'Ydata',}y+y0); \quad \text{% Translate by x0, y0} \\
\quad \text{pause(dt);} \quad \text{% Wait dt seconds} \\
\quad x0=x0+vx*dt; \quad \text{% update x position} \\
\quad y0=y0+vy*dt; \quad \text{% update y position} \\
\quad vy=vy-9.8*dt; \quad \text{% update y velocity} \\
\quad \text{if } x0\geq15-r, \quad \text{% check for right wall} \\
\quad \quad vx=-vx*p; \quad \text{% ...if hit, reverse (and decrease) velocity} \\
\quad \quad x0=15-r; \\
\quad \text{elseif } x0\leq-15+r, \quad \text{% left wall} \\
\quad \quad vx=-vx*p; \\
\quad \quad x0=-15+r; \\
\quad \text{end} \\
\quad \text{if } y0\geq15-r, \quad \text{% top} \\
\quad \quad vy=-vy*p; \\
\quad \quad y0=15-r; \\
\quad \text{elseif } y0\leq-15+r, \quad \text{% bottom} \\
\quad \quad vy=-vy*p; \\
\quad \quad y0=-15+r; \\
\quad \text{end} \\
\text{end} \]
Making Decisions: **while**...

**Syntax:**
```
while expression,
  ... statements ...
end
```

**Example:**
```
i=1;
while i<10,
  disp(i);
  i=i+4;
end
```

**Output:**
```
1
5
9
```

Note: this is equivalent to the **for**...**loop** used earlier
Example: while...

```matlab
while (abs(vx)>2) | (abs(vy)>2) | (y0<-15+2*r),
    set(myShape,'Xdata',x+x0,'Ydata',y+y0);   %Translate by x0, y0
    pause(dt);   %Wait dt seconds
    x0=x0+vx*dt;   %update x position
    vy=vy-9.8*dt;   %update y velocity
    if x0>=15-r,   %check for right wall
        vx=-vx*p;   %if hit, reverse (and decrease) velocity
        x0=15-r;
    elseif x0<=-15+r, %left wall
        vx=-vx*p;
        x0=-15+r;
    end
    if y0>=15-r,   %top
        vy=-vy*p;
        y0=15-r;
    elseif y0<=-15+r, %bottom
        vy=-vy*p;
        y0=-15+r;
    end
end
```
The laser pointer project

Given a simple robotic arm with a laser, how can you get it to point to a specified location on the wall.

To start, we must understand the geometry of the problem.
Trig review – basic functions

\[ x_0 = \ell \cos(\theta) \quad \cos(\theta) = \frac{x_0}{\ell} \]
\[ y_0 = \ell \sin(\theta) \quad \sin(\theta) = \frac{y_0}{\ell} \]
\[ \tan(\theta) = \frac{y_0}{x_0} \]

Inverse functions
\[ \theta = \arccos\left(\frac{x_0}{\ell}\right) \]
\[ \theta = \arcsin\left(\frac{y_0}{\ell}\right) \]
\[ \theta = \arctan\left(\frac{y_0}{x_0}\right) \]

But, be careful with arctangent…
Trig review – arctangents

Consider a point in the first quadrant, \((x_0, y_0)\).

\[
\tan(\theta) = \frac{y_0}{x_0} = \text{slope of line}
\]

\[
\gg \text{atan}(6/3) \times 180/\pi \\
\text{ans} = 63.4349
\]

\[
\gg \text{atan2}(6,3) \times 180/\pi \\
\text{ans} = 63.4349
\]

Now consider a point in the third quadrant, \((x_0, y_0)\).

\[
\gg \text{atan}(-6/-3) \times 180/\pi \\
\text{ans} = 63.4349
\]

\[
\gg \text{atan2}(-6,-3) \times 180/\pi \\
\text{ans} = -116.5651
\]
Trig review – Pythagoras

\[ x_0^2 + y_0^2 = \ell^2, \quad \text{Pythagoras' theorem} \]

\[ x_0 = \ell \cos(\theta) \quad y_0 = \ell \sin(\theta) \]

\[ \ell^2 \cos^2(\theta) + \ell^2 \sin^2(\theta) = \ell^2 \]

\[ \cos^2(\theta) + \sin^2(\theta) = 1 \]
Spherical → Cartesian Coordinates

\[ z = r \sin(\theta) \]
\[ \ell = r \cos(\theta) \]
\[ x = \ell \cos(\phi) = r \cos(\theta) \cos(\phi) \]
\[ y = \ell \sin(\phi) = r \cos(\theta) \sin(\phi) \]

Given \( r, \theta \) and \( \phi \) we can easily find \( x, y \) and \( z \).
(Forward kinematics)

Given \( x \) and \( y \), we could find \( \theta \) and \( \phi \) with some difficulty.
(Inverse kinematics)

... but this is not the problem we want to solve.
2-D Polar $\rightarrow$ Cartesian w/ offset

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

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

Angle of “r” from horizontal is equal to $\phi$.

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

$$y = y_0 + r \sin(\phi) = \rho \cos(\phi) + r \sin(\phi)$$
Forward Kinematics (1)

We know $\theta$, $\phi$ and $\rho$.
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 \( \theta, \phi \) and \( \rho \).
We want to know \( x, y, z \)

\[
\begin{align*}
    x_0 &= \rho \sin(\phi) \\
    y_0 &= \rho \cos(\phi) \\
    x &= x_0 + \ell \cos(\theta) \\
    x &= \rho \sin(\phi) + \ell \cos(\phi) \\
    x &= \rho \sin(\phi) + r \cos(\theta) \cos(\phi) \\
    y &= y_0 - \ell \sin(\theta) \\
    y &= \rho \cos(\phi) - \ell \sin(\phi) \\
    y &= \rho \cos(\phi) - r \cos(\theta) \sin(\phi) 
\end{align*}
\]

Note sign in this expression for \( y \)
Inverse Kinematics (1)

So... using forward kinematics we can determine x, y and z, given the angles \( \phi \) and \( \theta \).

\[
x = \rho \sin(\phi) + r \cos(\theta) \cos(\phi) \quad y = \rho \cos(\phi) - r \cos(\theta) \sin(\phi)
\]

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 \( \rho \) (determined by geometry of robot).

We want to find \( \phi \) and \( \theta \).
Inverse Kinematics (2)

We know $x$, $y$, $z$ and $\rho$.

**Solving for $\theta$.** (this is relatively easy)

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

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 \( \sin(\phi) \) and y by \( \cos(\phi) \) and add (to get rid of \( \ell \)):
\[ x \sin(\phi) = \rho \sin^2(\phi) + \ell \cos(\phi) \sin(\phi) \]
\[ y \cos(\phi) = \rho \cos^2(\phi) - \ell \sin(\phi) \cos(\phi) \]
\[ x \sin(\phi) + y \cos(\phi) = \rho \sin^2(\phi) + \ell \cos(\phi) \sin(\phi) + \rho \cos^2(\phi) - \ell \sin(\phi) \cos(\phi) \]
\[ x \sin(\phi) + y \cos(\phi) = \rho \left( \sin^2(\phi) + \cos^2(\phi) \right) \]
\[ x \sin(\phi) + y \cos(\phi) = \rho \]

We’re almost done...
Inverse Kinematics (4)

... continued from previous

\[ x \sin(\phi) + y \cos(\phi) = \rho \]

Use trigonometric identity:

\[ \alpha = \phi \]

\[ a = x \quad b = y \quad \alpha = \phi \]

\[ x \sin(\phi) + y \cos(\phi) = \sqrt{x^2 + y^2} \sin(\phi + \tan^{-1}(y, x)) = \rho \]

\[ \sin(\phi + \tan^{-1}(y, x)) = \frac{\rho}{\sqrt{x^2 + y^2}} \]

\[ \phi + \tan^{-1}(y, x) = \arcsin\left(\frac{\rho}{\sqrt{x^2 + y^2}}\right) \quad \text{so...} \]

\[ \phi = \arcsin\left(\frac{\rho}{\sqrt{x^2 + y^2}}\right) - \tan^{-1}(y, x) \]
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.