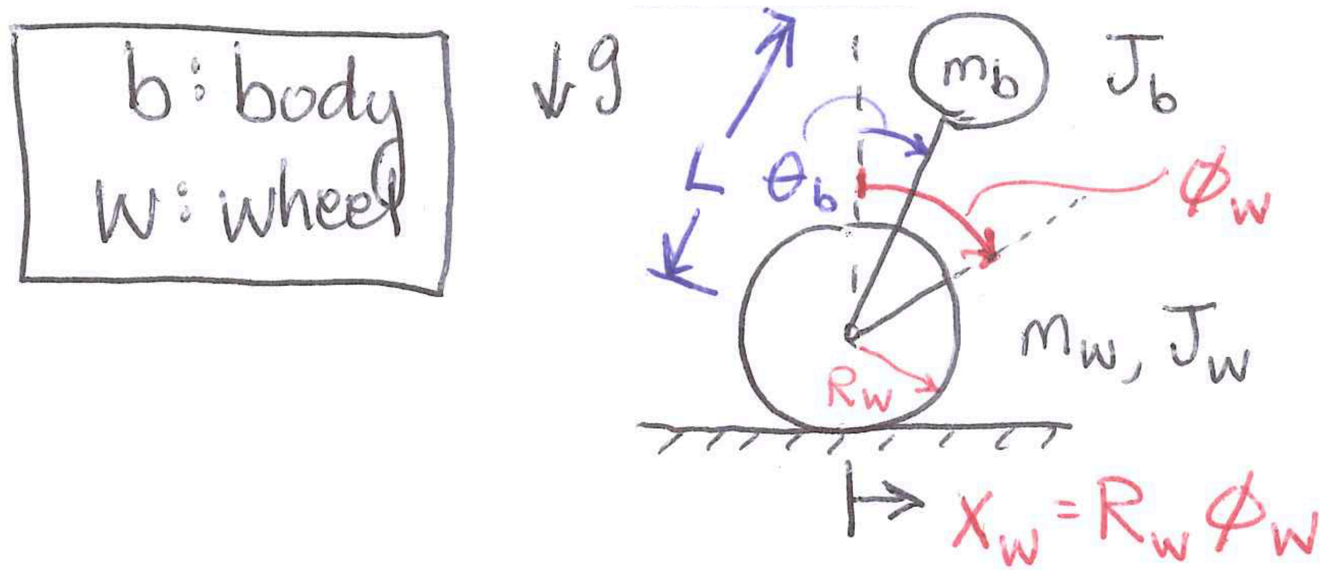
MATLAB’s “Symbolic Toolbox”

Symbolic differentiation, to get Lagrangian EOMs

Use this document as a guide for HW 4

Balancing Inverted Pendulum Robot :



(...aka the "Segway" style system)

Here's an approach:

Note: I used "q" instead of "little xi" for G.C.'s here… Don't get confused. Various references may use either one. Roboticists like to use "q".

1. **Define needed variables and constraints:**
   a) Define the Generalized Coordinates (G.C.'s), q_1 through q_n.
   b) Define geometry of masses and inertia elements.
   c) Differentiate, to define velocities (for kinetic energy).
2. **Define kinetic energy, T**
3. **Define potential energy, V**
4. **Then the Lagrangian is: L = T – V**
5. **The lefthand of the i^{th} EOM is:**
6. **Righthand side of each EOM is "Big Xi".**

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \Xi_i$$

Here is annotated matlab code to generate EOM's…

- **Requires an additional m-file called:**

fulldiff.m

Fulldiff.m performs the "chain rule" for differentiation of variables with time derivatives.

0. **First, define symbolic variables you will need…**

0. First, define symbolic variables you will need...

```
% segway_eom.m
%
% "Segway-Style" Inverted Pendulum: Equations of Motion.
% (See slides from Lecture 11 for a visual description of the system.)
% Katie Byl, Nov. 2, 2017. ECE/ME 179D, UCSB.

clear all; format compact % compact produces single-spaced output

% Define symbolic variables in matlab:
syms phiw thetab L mb Jb mw Jw Rw g b tau
```

Modify "segway_eom.m" for HW 4 part a

1. **Define needed variables and constraints:**
   a) **Define the Generalized Coordinates (G.C.'s), q_1 through q_n.**
   b) **Define geometry of masses and inertia elements.**
   c) **Differentiate, to define velocities (for kinetic energy).**

Roboticists like to use "q" (for joints), instead of "xi".

```
% 1a. GC's (generalized coordinates), and their derivatives:
GC = [{phiw},{thetab}];% Using ABSOLUTE angles here
dphiw = fulldiff(phiw,GC);% time derivative. GC are variables (over time)
dthetab = fulldiff(thetab,GC);

% 1b. Geometry of the masses/inertias, given GC's are freely changing...
xw = Rw*phiw;
xb = xw+L*sin(thetab);
yw = 0;
yb = L*cos(thetab);

% 1c. Define any required velocity terms (for masses):
dxw = fulldiff(xw,GC);
dxb = fulldiff(xb,GC);
dyb = fulldiff(yb,GC);
```

1. **Define needed variables and constraints:**
    a) **Define the Generalized Coordinates (G.C.'s), q_1 through q_n.**
    b) **Define geometry of masses and inertia elements.**
    c) **Differentiate, to define velocities (for kinetic energy).**

2. **Define kinetic energy, T**
3. **Define potential energy, V**
4. **Then the Lagrangian is: L = T – V**

```
% 2. Kinetic Energy:
T = (1/2)*(mw*dxw^2 + Jw*dphiw^2 + mb*(dxb^2 + dyb^2) + Jb*dthetab^2)

% 3. Potential Energy:
V = mb*g*yb

% 4. Lagrangian:
L = T-V
```

2. **Define kinetic energy, T**
3. **Define potential energy, V**
4. **Then the Lagrangian is: L = T – V**

5. The lefthand of the ith EOM is:
6. Righthand side of each EOM is "Big Xi".

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \Xi_i$$

```
% 5. EOMs:
eq1 = fulldiff(diff(L,dphiw),GC) - diff(L,phiw)
eq2 = fulldiff(diff(L,dthetab),GC) - diff(L,thetab);
eq2 = simplify(eq2)

% 6. Xi: non-conservative terms
Xi1 = tau - b*(dphiw-dthetab)% Motor torque tau, and back emf damping b
Xi2 = -tau + b*(dphiw-dthetab)% (equal and opposite to above)
```

Equations of Motion (EOM's) are then:

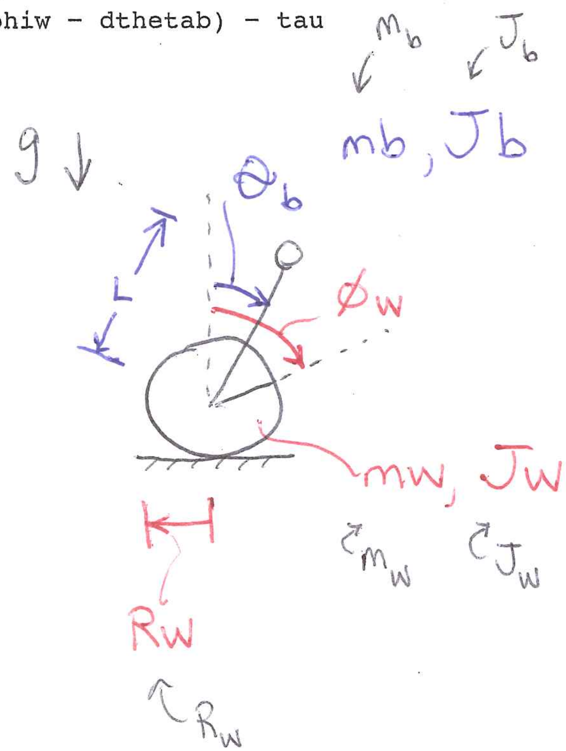
eq1 = Xi1

eq2 = Xi2

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \Xi_i$$

1. **The lefthand of the i^th EOM is:**
2. **Righthand side of each EOM is "Big Xi".**

```
Output from segway_eom.m :
(Katie Byl, 2012)
-----------------------------------------------------

T =
(Jb*dthetab^2)/2 + (Jw*dphiw^2)/2 + (mb*((Rw*dphiw + L*dthetab*cos(thetab))^2 + L^2*dthetab^2*sin(thetab)^2))/2 +
(Rw^2*dphiw^2*mw)/2

V =
L*g*mb*cos(thetab)

L =
(Jb*dthetab^2)/2 + (Jw*dphiw^2)/2 + (mb*((Rw*dphiw + L*dthetab*cos(thetab))^2 + L^2*dthetab^2*sin(thetab)^2))/2 +
(Rw^2*dphiw^2*mw)/2 - L*g*mb*cos(thetab)

eq1 =
- L*Rw*mb*sin(thetab)*dthetab^2 + d2phiw*(Jw + Rw^2*mb + Rw^2*mw) + L*Rw*d2thetab*mb*cos(thetab)

eq2 =
Jb*d2thetab + L^2*d2thetab*mb - L*g*mb*sin(thetab) + L*Rw*d2phiw*mb*cos(thetab)

Xi1 =
tau - b*(dphiw - dthetab)

Xi2 =
b*(dphiw - dthetab) - tau
```



$\theta_b \rightarrow$ thetab	$\phi_w \rightarrow$ phiw
$\dot{\theta}_b \rightarrow$ dthetab	$\dot{\phi}_w \rightarrow$ dphiw
$\ddot{\theta}_b \rightarrow$ d2thetab	$\ddot{\phi}_w \rightarrow$ d2phiw