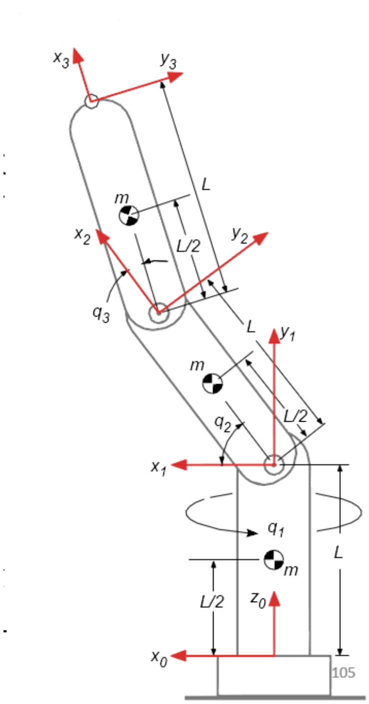
Derivation of equation of motion for the RRR robot arm

These below are the steps used to obtain the equations of motion for the 3 degree of freedom RRR robot arm to use them to make the simulation in *Mathematica*.
This was written in version 10.02 and the simulation uses Manipulate. You can adjust the speed of the simulation, the torque applied to each joint, and the damping at each joint.
Nasser M. Abbasi
4/2/2015

Forward kinematics

The following diagram (thanks to lecture notes by Professor Zinn, UW Madison) shows the geometry of the problem. In the derivation below, I used x_i in place of q_i. There are 3 degrees of freedom. (RRR robot arm)



The first step is to determine the transformation matrices between each frame and the base.

In[35]:=
```
(T01 = {{Cos[x1[t]], 0, Sin[x1[t]], 0},
    {Sin[x1[t]], 0, -Cos[x1[t]], 0}, {0, 1, 0, L1}, {0, 0, 0, 1}}) // MatrixForm
```

Out[35]//MatrixForm=

$$\begin{pmatrix} \text{Cos[x1[t]]} & 0 & \text{Sin[x1[t]]} & 0 \\ \text{Sin[x1[t]]} & 0 & -\text{Cos[x1[t]]} & 0 \\ 0 & 1 & 0 & \text{L1} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In[36]:=
```
T01c = T01; T01c[[1 ;; 3, 4]] = T01c[[1 ;; 3, 4]]/2;
MatrixForm[T01c]
```

Out[37]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[x1[t]] & 0 & \text{Sin}[x1[t]] & 0 \\ \text{Sin}[x1[t]] & 0 & -\text{Cos}[x1[t]] & 0 \\ 0 & 1 & 0 & \frac{L1}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In[38]:=
```
(T12 = {{Cos[x2[t]], -Sin[x2[t]], 0, L2 Cos[x2[t]]}, {Sin[x2[t]],
    Cos[x2[t]], 0, L2 Sin[x2[t]]}, {0, 0, 1, 0}, {0, 0, 0, 1}}) // MatrixForm
```

Out[38]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[x2[t]] & -\text{Sin}[x2[t]] & 0 & \text{L2 Cos}[x2[t]] \\ \text{Sin}[x2[t]] & \text{Cos}[x2[t]] & 0 & \text{L2 Sin}[x2[t]] \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In[39]:=
```
T12c = T12;
T12c[[1 ;; 3, 4]] = 1/2 T12c[[1 ;; 3, 4]];
MatrixForm[T12c]
```

Out[41]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[x2[t]] & -\text{Sin}[x2[t]] & 0 & \frac{1}{2}\text{L2 Cos}[x2[t]] \\ \text{Sin}[x2[t]] & \text{Cos}[x2[t]] & 0 & \frac{1}{2}\text{L2 Sin}[x2[t]] \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In[42]:=
```
(T23 = {{Cos[x3[t]], -Sin[x3[t]], 0, L3 Cos[x3[t]]}, {Sin[x3[t]],
    Cos[x3[t]], 0, L3 Sin[x3[t]]}, {0, 0, 1, 0}, {0, 0, 0, 1}}) // MatrixForm
```

Out[42]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[x3[t]] & -\text{Sin}[x3[t]] & 0 & \text{L3 Cos}[x3[t]] \\ \text{Sin}[x3[t]] & \text{Cos}[x3[t]] & 0 & \text{L3 Sin}[x3[t]] \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In[43]:=
```
T23c = T23;
T23c[[1 ;; 3, 4]] = 1/2 T23c[[1 ;; 3, 4]]
MatrixForm[T23c]
```

Out[44]= $\left\{\frac{1}{2}\text{L3 Cos}[x3[t]], \frac{1}{2}\text{L3 Sin}[x3[t]], 0\right\}$

Out[45]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[x3[t]] & -\text{Sin}[x3[t]] & 0 & \frac{1}{2}\text{L3 Cos}[x3[t]] \\ \text{Sin}[x3[t]] & \text{Cos}[x3[t]] & 0 & \frac{1}{2}\text{L3 Sin}[x3[t]] \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now that we found the transformation matrices between each frame, we find the transformation between each frame and the base

```
In[46]:= (T02 = T01.T12) // MatrixForm
Out[46]//MatrixForm=
 Cos[x1[t]] Cos[x2[t]]   -Cos[x1[t]] Sin[x2[t]]   Sin[x1[t]]   L2 Cos[x1[t]] Cos[x2[t]]
 Cos[x2[t]] Sin[x1[t]]   -Sin[x1[t]] Sin[x2[t]]   -Cos[x1[t]]  L2 Cos[x2[t]] Sin[x1[t]]
      Sin[x2[t]]               Cos[x2[t]]              0        L1 + L2 Sin[x2[t]]
          0                        0                   0                1
```

```
In[47]:= (T03 = T02.T23) // MatrixForm
Out[47]//MatrixForm=
 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] - Cos[x1[t]] Sin[x2[t]] Sin[x3[t]]   -Cos[x1[t]] Cos[x
 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] - Sin[x1[t]] Sin[x2[t]] Sin[x3[t]]   -Cos[x3[t]] Sin[x
            Cos[x3[t]] Sin[x2[t]] + Cos[x2[t]] Sin[x3[t]]                          Cos[x2
                                  0
```

```
In[48]:= (T02c = T01.T12c) // MatrixForm
Out[48]//MatrixForm=
 Cos[x1[t]] Cos[x2[t]]   -Cos[x1[t]] Sin[x2[t]]   Sin[x1[t]]   1/2 L2 Cos[x1[t]] Cos[x2[t]]
 Cos[x2[t]] Sin[x1[t]]   -Sin[x1[t]] Sin[x2[t]]   -Cos[x1[t]]  1/2 L2 Cos[x2[t]] Sin[x1[t]]
      Sin[x2[t]]               Cos[x2[t]]              0        L1 + 1/2 L2 Sin[x2[t]]
          0                        0                   0                 1
```

```
In[49]:= (T03c = T02.T23c) // MatrixForm
Out[49]//MatrixForm=
 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] - Cos[x1[t]] Sin[x2[t]] Sin[x3[t]]   -Cos[x1[t]] Cos[x
 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] - Sin[x1[t]] Sin[x2[t]] Sin[x3[t]]   -Cos[x3[t]] Sin[x
            Cos[x3[t]] Sin[x2[t]] + Cos[x2[t]] Sin[x3[t]]                          Cos[x2
                                  0
```

obtain the needed vectors

```
In[50]:= z00 = {0, 0, 1};

In[51]:= z01 = T01[[1 ;; 3, 3]]
Out[51]= {Sin[x1[t]], -Cos[x1[t]], 0}

In[52]:= z02 = T02[[1 ;; 3, 3]]
Out[52]= {Sin[x1[t]], -Cos[x1[t]], 0}

In[53]:= o01 = T01[[1 ;; 3, 4]]
Out[53]= {0, 0, L1}

In[54]:= o02 = T02[[1 ;; 3, 4]]
Out[54]= {L2 Cos[x1[t]] Cos[x2[t]], L2 Cos[x2[t]] Sin[x1[t]], L1 + L2 Sin[x2[t]]}
```

In[55]:=
```
o03 = T03[[1 ;; 3, 4]]
```

Out[55]=
```
{L2 Cos[x1[t]] Cos[x2[t]] + L3 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] -
  L3 Cos[x1[t]] Sin[x2[t]] Sin[x3[t]], L2 Cos[x2[t]] Sin[x1[t]] +
  L3 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] - L3 Sin[x1[t]] Sin[x2[t]] Sin[x3[t]],
 L1 + L2 Sin[x2[t]] + L3 Cos[x3[t]] Sin[x2[t]] + L3 Cos[x2[t]] Sin[x3[t]]}
```

Find the linear veclocity Jacobian of origin of end effector

In[56]:=
```
r3 = T03[[1 ;; 3, 4]]
```

Out[56]=
```
{L2 Cos[x1[t]] Cos[x2[t]] + L3 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] -
  L3 Cos[x1[t]] Sin[x2[t]] Sin[x3[t]], L2 Cos[x2[t]] Sin[x1[t]] +
  L3 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] - L3 Sin[x1[t]] Sin[x2[t]] Sin[x3[t]],
 L1 + L2 Sin[x2[t]] + L3 Cos[x3[t]] Sin[x2[t]] + L3 Cos[x2[t]] Sin[x3[t]]}
```

In[65]:=
```
jv3 = Transpose[{ D[r3, x1[t]], D[r3, x2[t]], D[r3, x3[t]]}]
```

Out[65]=
```
{{-L2 Cos[x2[t]] Sin[x1[t]] - L3 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] +
   L3 Sin[x1[t]] Sin[x2[t]] Sin[x3[t]], -L2 Cos[x1[t]] Sin[x2[t]] -
   L3 Cos[x1[t]] Cos[x3[t]] Sin[x2[t]] - L3 Cos[x1[t]] Cos[x2[t]] Sin[x3[t]],
  -L3 Cos[x1[t]] Cos[x3[t]] Sin[x2[t]] - L3 Cos[x1[t]] Cos[x2[t]] Sin[x3[t]]},
 {L2 Cos[x1[t]] Cos[x2[t]] + L3 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] -
   L3 Cos[x1[t]] Sin[x2[t]] Sin[x3[t]], -L2 Sin[x1[t]] Sin[x2[t]] -
   L3 Cos[x3[t]] Sin[x1[t]] Sin[x2[t]] - L3 Cos[x2[t]] Sin[x1[t]] Sin[x3[t]],
  -L3 Cos[x3[t]] Sin[x1[t]] Sin[x2[t]] - L3 Cos[x2[t]] Sin[x1[t]] Sin[x3[t]]},
 {0, L2 Cos[x2[t]] + L3 Cos[x2[t]] Cos[x3[t]] - L3 Sin[x2[t]] Sin[x3[t]],
  L3 Cos[x2[t]] Cos[x3[t]] - L3 Sin[x2[t]] Sin[x3[t]]}}
```

```
v3 = jv3.{x1'[t], x2'[t], x3'[t]}
```

```
{(-L2 Cos[x2[t]] Sin[x1[t]] - L3 Cos[x2[t]] Cos[x3[t]] Sin[x1[t]] +
     L3 Sin[x1[t]] Sin[x2[t]] Sin[x3[t]]) x1′[t] +
  (-L2 Cos[x1[t]] Sin[x2[t]] - L3 Cos[x1[t]] Cos[x3[t]] Sin[x2[t]] -
     L3 Cos[x1[t]] Cos[x2[t]] Sin[x3[t]]) x2′[t] +
  (-L3 Cos[x1[t]] Cos[x3[t]] Sin[x2[t]] - L3 Cos[x1[t]] Cos[x2[t]] Sin[x3[t]]) x3′[t],
 (L2 Cos[x1[t]] Cos[x2[t]] + L3 Cos[x1[t]] Cos[x2[t]] Cos[x3[t]] -
     L3 Cos[x1[t]] Sin[x2[t]] Sin[x3[t]]) x1′[t] +
  (-L2 Sin[x1[t]] Sin[x2[t]] - L3 Cos[x3[t]] Sin[x1[t]] Sin[x2[t]] -
     L3 Cos[x2[t]] Sin[x1[t]] Sin[x3[t]]) x2′[t] +
  (-L3 Cos[x3[t]] Sin[x1[t]] Sin[x2[t]] - L3 Cos[x2[t]] Sin[x1[t]] Sin[x3[t]]) x3′[t],
 (L2 Cos[x2[t]] + L3 Cos[x2[t]] Cos[x3[t]] - L3 Sin[x2[t]] Sin[x3[t]]) x2′[t] +
  (L3 Cos[x2[t]] Cos[x3[t]] - L3 Sin[x2[t]] Sin[x3[t]]) x3′[t]}
```

```
v3 = Simplify[v3]
{- (L2 Cos[x2[t]] + L3 Cos[x2[t] + x3[t]]) Sin[x1[t]] x1′[t] - Cos[x1[t]]
    ((L2 Sin[x2[t]] + L3 Sin[x2[t] + x3[t]]) x2′[t] + L3 Sin[x2[t] + x3[t]] x3′[t]),
  Cos[x1[t]] (L2 Cos[x2[t]] + L3 Cos[x2[t] + x3[t]]) x1′[t] - Sin[x1[t]]
    ((L2 Sin[x2[t]] + L3 Sin[x2[t] + x3[t]]) x2′[t] + L3 Sin[x2[t] + x3[t]] x3′[t]),
  (L2 Cos[x2[t]] + L3 Cos[x2[t] + x3[t]]) x2′[t] + L3 Cos[x2[t] + x3[t]] x3′[t]}

o01c = T01c[[1 ;; 3, 4]]
{0, 0, L1/2}

o02c = T02c[[1 ;; 3, 4]]

o03c = T03c[[1 ;; 3, 4]]


T02[[1 ;; 3, 4]]

jv1c = {Cross[z00, o01c], {0, 0, 0}, {0, 0, 0}} // Transpose

(jv2c = {Cross[z00, o02c], Cross[z01, o02c - o01], {0, 0, 0}} // Transpose) //
  Simplify // MatrixForm

(jv3c = {Cross[z00, o03c], Cross[z01, o03c - o01], Cross[z02, o03c - o02]} //
     Transpose) // Simplify // MatrixForm

(jw1 = {z00, {0, 0, 0}, {0, 0, 0}} // Transpose) // MatrixForm

(jw2 = {z00, z01, {0, 0, 0}} // Transpose) // MatrixForm

(jw3 = {z00, z01, z02} // Transpose) // MatrixForm

R01 = T01[[1 ;; 3, 1 ;; 3]];
R02 = T02[[1 ;; 3, 1 ;; 3]];
R03 = T03[[1 ;; 3, 1 ;; 3]];
I1 = {{1 / 12 m1 (3 * r1^2 + L1^2), 0, 0},
    {0, m1 * r1^2 / 2, 0}, {0, 0, 1 / 12 m1 (3 * r1^2 + L1^2)}};
I2 = {{m2 * r2^2 / 2, 0, 0}, {0, 1 / 12 m2 (3 * r2^2 + L2^2), 0},
    {0, 0, 1 / 12 m2 (3 * r2^2 + L2^2)}};
I3 = {{m3 * r3^2 / 2, 0, 0}, {0, 1 / 12 m3 (3 * r3^2 + L3^2), 0},
    {0, 0, 1 / 12 m3 (3 * r3^2 + L3^2)}};
massMatrix = m1 Transpose@jv1c . jv1c + m2 Transpose@jv2c . jv2c +
   m3 Transpose@jv3c . jv3c + Transpose@jw1.R01 .I1 .Transpose@R01 .jw1 +
   Transpose@jw2.R02 .I2 .Transpose@R02 .jw2 +
   Transpose@jw3.R03 .I3 .Transpose@R03 .jw3;
MatrixForm[Simplify@massMatrix]
```

Coriolis terms

```
d[i_, j_, k_] := D[massMatrix[[i, j]], ToExpression["x" <> ToString[k] <> "[t]"]];
b[i_, j_, k_] := 1/2 (d[i, j, k] + d[i, k, j] - d[j, k, i])
B0 = {{2 b[1, 1, 2], 2 b[1, 1, 3], 2 b[1, 2, 3]},
   {2 b[2, 1, 2], 2 b[2, 1, 3], 2 b[2, 2, 3]}, {2 b[3, 1, 2], 2 b[3, 1, 3], 2 b[3, 2, 3]}};
MatrixForm[Simplify@B0]
```

centrifugal terms

```
C0 = {{b[1, 1, 1], b[1, 2, 2], b[1, 3, 3]},
   {b[2, 1, 1], b[2, 2, 2], b[2, 3, 3]}, {b[3, 1, 1], b[3, 2, 2], b[3, 3, 3]}};
MatrixForm[Simplify@C0]
```

Gravity

```
gvector = {0, 0, -g};
G0 = -(Transpose@jv1c.(m1 gvector) +
     Transpose@jv2c.(m2 gvector) + Transpose@jv3c.(m3 gvector));
MatrixForm[Simplify[G0]]

eqOfMotion = Simplify[massMatrix.{x1''[t], x2''[t], x3''[t]} +
   B0.{x1'[t] x2'[t], x1'[t] x3'[t], x2'[t] x3'[t]} +
   C0.{(x1'[t])^2, (x2'[t])^2, (x3'[t])^2} + G0]
```