Forward Kinematics

You Start with Separate Pieces, all Defined in their Own Local Coordinate System

Hook the Pieces Together, Change Parameters, Things Move
(All Children Understand This)

Where do the Pieces Move To?

Positioning Part #1 With Respect to Ground

1. Rotate by $\theta_1$
2. Translate by $T_{1G}$

Write it

$[M_{1G}] = [T_{1G}] [R_{\theta_1}]$

Say it

Why Do We Say it Right-to-Left?

Because computer graphics has adopted the convention that
the coordinates are multiplied on the right side of the matrix.
Positioning Part #2 With Respect to Ground

1. Rotate by \( \Theta_2 \)
2. Translate the length of part 1
3. Rotate by \( \Theta_1 \)
4. Translate by \( T \)

\[
[M_{2/G}] = [T_{1/G}] *[R_{G_1}] *[T_{2_1}] *[R_{G_2}]
\]

\[
[M_{2/G}] = [M_{3/G}] *[M_{2/1}]
\]

Positioning Part #3 With Respect to Ground

1. Rotate by \( \Theta_3 \)
2. Translate the length of part 2
3. Rotate by \( \Theta_2 \)
4. Translate the length of part 1
5. Rotate by \( \Theta_1 \)
6. Translate by \( T \)

\[
[M_{3/G}] = [T_{1/G}] *[R_{G_1}] *[T_{2_1}] *[R_{G_2}] *[T_{3_2}] *[R_{G_3}]
\]

\[
[M_{3/G}] = [M_{3/G}] *[M_{2/1}] *[M_{3/2}]
\]

Sample Program

```c
float DrawMechanism( float \[ \Theta \] 1, float \[ \Theta \] 2, float \[ \Theta \] 3 )
{
    glPushMatrix();
    glRotatef( \[ \Theta \] 1, 0., 0., 1.);
    glColor3f( 1., 0., 0. );
    DrawLinkOne( );
    glTranslatef( LENGTH_1, 0., 0.);
    glRotatef( \[ \Theta \] 2, 0., 0., 1.);
    glColor3f( 0., 1., 0. );
    DrawLinkTwo( );
    glTranslatef( LENGTH_2, 0., 0.);
    glRotatef( \[ \Theta \] 3, 0., 0., 1.);
    glColor3f( 0., 0., 1. );
    DrawLinkThree( );
    glPopMatrix();
}
```

Sample Program, using OpenGL’s Built-in Transformation Concatenation

Sample Program

```c
float DrawLinkOne( )
{
    glBegin( GL_QUADS );
    glVertex2f(    -BUTT, -THICKNESS/2 );
    glVertex2f( LENGTH_1, -THICKNESS/2 );
    glVertex2f( LENGTH_1,  THICKNESS/2 );
    glVertex2f(    -BUTT,  THICKNESS/2 );
    glEnd( );
}
```

```c
float DrawLinkTwo( )
{
    glBegin( GL_QUADS );
    glVertex2f(    -BUTT, -THICKNESS/2 );
    glVertex2f( LENGTH_1, -THICKNESS/2 );
    glVertex2f( LENGTH_1,  THICKNESS/2 );
    glVertex2f(    -BUTT,  THICKNESS/2 );
    glEnd( );
}
```

```c
float DrawLinkThree( )
{
    glBegin( GL_QUADS );
    glVertex2f(    -BUTT, -THICKNESS/2 );
    glVertex2f( LENGTH_1, -THICKNESS/2 );
    glVertex2f( LENGTH_1,  THICKNESS/2 );
    glVertex2f(    -BUTT,  THICKNESS/2 );
    glEnd( );
}
```
Sample Program

```c
glViewport(100, 100, 500, 500);
glMatrixMode(GL_PROJECTION);
gluPerspective(90., 1.0, 1., 10.);
glMatrixMode(GL_MODELVIEW);
gluLookAt(eyex, eyey, eyez,
centerx, centery, centerz,
upx, upy, upz);
DrawMechanism(θ1, θ2, θ3);
gluPopMatrix();
do = FALSE;
while( ! done )
{
    Determine θ1, θ2, θ3
    glPushMatrix();
    gluLookAt(eyex, eyey, eyez,
        centerx, centery, centerz,
        upx, upy, upz);
    DrawMechanism(θ1, θ2, θ3);
    gluPopMatrix();
}
```

Another Great Example of Hierarchical Transformations