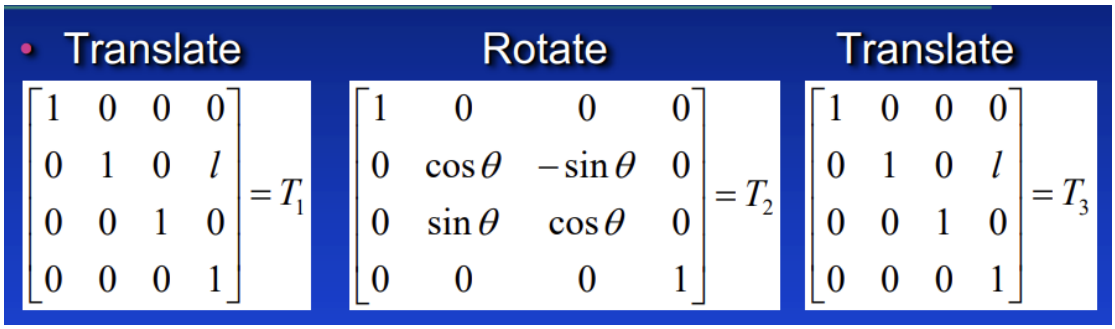
Move the figure through a trajectory given by 7 control points, similar to lab1.
The leg motion is achieved by sinusoidal interpolation, left leg for example:

```
// First Translate Matrix
GLfloat LT1[16] = { 1, 0, 0, 0,            //column 1
                    0, 1, 0, 0,               //column 2
                    0, 0, 1, 0,               //column 3
                    0, -1, 0, 1 };         //column 4

//Rotation Matrix by Z axix
GLfloat LAngle = (sin(4 * 3.14 * t - 3.14 / 2) * 3.14) / 4;     //animate rotation
GLfloat LT2[16] = { cos(LAngle), sin(LAngle), 0, 0,            //column 1
                   -sin(LAngle), cos(LAngle), 0, 0,                        //column 2
                    0, 0, 1, 0,                                                    //column 3
                    0, 0, 0, 1 };                                                      //column 4

// Second Translate Matrix
GLfloat LT3[16] = { 1, 0, 0, 0,            //column 1
                    0, 1, 0, 0,               //column 2
                    0, 0, 1, 0,               //column 3
                    0, 0, 0.3, 1 };           //column 4
```

Basic Structure:



LAngle for sinusoidal interpolation, it calculates a middle point between start point and end point, making a smooth transition between rotating angles.
Sine wave formula:

$$y(t) = A \cdot sin(2\pi ft + \varphi)$$

Using Matrix4Mult4 to multiply the two 4*4 matrix and store them in the respective leg matrix, resulting as:
-LLeg translate along y axis
-Rotate LAngle about axis (where you animate the rotation)
-LLeg translate along y axis
which is the transformation that describes B in A coordinate system

```
//T3*T2*T1
Matrix4Mult4(M, LT2, Left);
Matrix4Mult4(Left, LT1, Left);
Matrix4Mult4(Left, LT3, Left);

//Show left leg
glLoadMatrixf(Left);
glScalef(0.3f, 2.0f, 0.3f);
glutSolidCube(1.0);
```