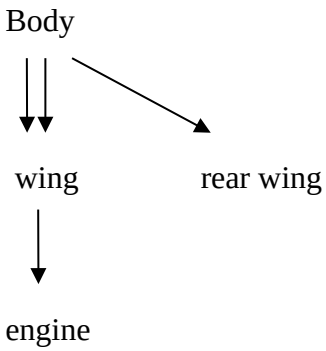
Ans. to Tut 3

Qn 1

a)

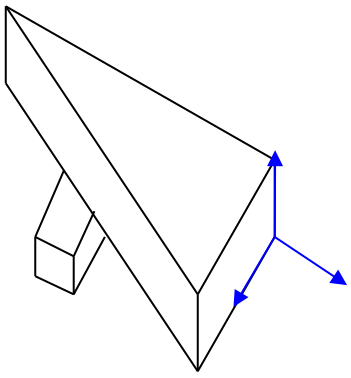


b) $\mathbf{M}_{leftwing \leftarrow engine} = \mathbf{T}(-20,0,0)^{-1} = \mathbf{T}(20,0,0)$

$\mathbf{M}_{body \leftarrow leftwing} = \mathbf{T}(-5,0,0)^{-1} = \mathbf{T}(5,0,0)$

$\mathbf{M}_{body \leftarrow rearwing} = \mathbf{T}(0,-10,40)^{-1} = \mathbf{T}(0,10,-40)$

Observe that the right wing with engine can be obtained by physically reflecting the left wing with engine, i.e. applying $\mathbf{RF}_x$ to vertices in *wing ( )* will give the following object:



Right wing

Note that the coordinate system is the original coordinate system of the left wing, since the reflection is a physical reflection, and so

$$\mathbf{M}_{body \leftarrow rightwing} = \mathbf{T}(5,0,0)^{-1} = \mathbf{T}(-5,0,0)$$

Hence we write the OpenGL program as follows:

```
/*  draw both the left and the right wing,  with engines */
void draw_wing ()
{
        glPushMatrix ( );

        wing ();
        glTranslatef (20,0,0);
        engine ();

        glPopMatrix ( );
}


/* draw the airplane */
void airplane (void)
{
        glMatrixMode (GL_MODELVIEW);  // set current matrix to identity
        glLoadIdentity ( );

        // draw body
        body ();

        glPushMatrix ();                 // store the current matrix

        // draw left wing with engine
        glTranslatef (5,0,0);            //   M body←leftwing
        draw_wing ();

        glPopMatrix ();                  // retrieve the original current matrix
                                         // associated with body
        glPushMatrix ();

        // draw right wing with engine

        glTranslatef (-5,0,0);           //    M body←rightwing

        // you can consider the next four lines as a function draw_right_wing
        glPushMatrix ();

        glScalef      (-1,1,1);          //   draw the right wing
        draw_wing ();
```

```
        glPopMatrix ();

        glPopMatrix ();                 // retrieve the original current matrix
                                        // associated with body
        glPushMatrix ();

       // draw rear wing
        glTranslatef (0,10,-40);        //   M_{body←rearwing}
        rear_wing ();
    }
```

Qn 2

a)
```
    void box (float length, width, height)
    {
        glPushMatrix ( );
        glScalef      (length, width, height);
        glTranslatef (0, 0.5, 0);
        glutSolidCube (1);

        glPopMatrix  ( );
    }
```

b) i) $\mathbf{M}_{la\leftarrow ua} = \mathbf{T}(15,65,0)\mathbf{R}_z(-90^o)$ (rule 2)

Alternatively,

$\mathbf{M}_{la\leftarrow ua} = [\mathbf{T}(65,-15,0)\mathbf{R}_z(90^o)]^{-1} = \mathbf{R}_z(-90^o)\mathbf{T}(-65,15,0)$ (rule 1)

Note that both will give the same 4×4 composite transformation matrix. You can verify this.

ii) $\mathbf{M}_{ua\leftarrow tf} = \mathbf{T}(-15,80,0)\mathbf{R}_z(30^o)\mathbf{S}(0.5,0.5,0.5)$

Note that it is much simpler to work with coordinate system ua, treat ua as an object and find the transformation from ua to tf, i.e. use method 2.

iii) $\mathbf{M}_{ua\leftarrow bf} = \mathbf{T}(15,80,0)\mathbf{R}_z(-30^o)\mathbf{S}(0.5,0.5,0.5)$

Note similar situation to ii).

```
    void robotic_hand ( );
    {
        glMatrixMode (GL_MODELVIEW);  // set current matrix to identity
        glLoadIdentity ( );

        glRotatef (30, 0, 0, 1);      // the lower arm rotates
```

```
box (30, 80,50);                    // draw lower arm

glTranslatef (15, 65, 0);     //  M_{la←ua}
glRotatef     (-90, 0, 0, 1);

box (30, 80, 50);                   // draw upper arm

glPushMatrix ( );                   // store current matrix

glTranslatef (-15, 80, 0);      //  M_{ua←tf}
glRotatef (30, 0, 0, 1);
glScalef (0.5, 0.5, 0.5);

glRotatef (20, 0, 0, 1);        // the top finger rotates

box (30, 80, 50);

glPopMatrix ( );                    // restore current matrix

glTranslatef (15, 80, 0);       //  M_{ua←bf}
glRotatef (-30, 0, 0, 1);
glScalef (0.5, 0.5, 0.5)

 box (30, 80, 50);

}
```

c) The underlined code above.