OpenGL勉強会
第二回

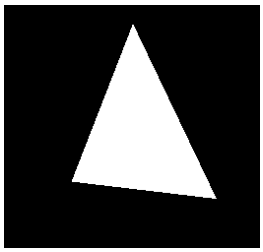
慶應義塾大学大学院
システムデザイン・マネジメント研究科
立山　義祐

今回学ぶ関数

- gluPerspective(画角、アスペクト比、クリップ指定);
- gluLookAt(カメラ位置、見る対象の点、上方向);
- glTranslate3f(並行移動位置)
- glRotate3f(角度、回転軸ベクトル)

gltest3.c (Rotating a Object)

```
#include <GL/glut.h>
#endif

float g_r = 0.0f;

void
display(void)
{
 int i;

 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(1.0f, 1.0f, 1.0f);
 glLoadIdentity();
 glTranslatef(0.0, 0.0, -5.0f);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glBegin(GL_TRIANGLES);
 glVertex3f(1.0f, -.5f, 0.0f);
 glVertex3f(0.0f, 1.0f, 0.0f);
 glVertex3f(-1.0f, -.5f, 0.0f);
 glEnd();
 glutSwapBuffers();
}
```

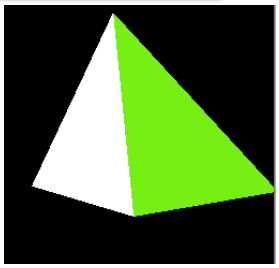


```
void
resize(int w, int h)
{
 glViewport(0, 0, w, h);
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluPerspective(30.0, (double)w/(double)h, 1.0f,
       100.0f);
 //glTranslatef(0.0f, 0.0f, -5.0f);
 //gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0,
       0.0);
 glMatrixMode(GL_MODELVIEW);
}

void
init(void)
{
 glClearColor(0.0, 0.0, 0.0, 0.0);
}
```

```
void
idle(void)
{
 g_r = g_r + 10.0f;
 if (g_r >= 360.0f) {
  g_r = 0.0f;
 }
 glutPostRedisplay();
 usleep(50000);
}

int
main(int ac, char **av)
{
 glutInit(&ac, av);
 glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
 glutCreateWindow(av[0]);
 glutDisplayFunc(display);
 glutReshapeFunc(resize);
 glutIdleFunc(idle);
 init();
 glutMainLoop();
 return 0;
}
```


gltest4.c (Depth Test & 3D object)

```
void
display(void)
{
 int i;

 glClear(GL_COLOR_BUFFER_BIT |
      GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 //glTranslatef(0.0, 0.0, -5.0f);
 gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0,
      0.0, 1.0, 0.0);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glBegin(GL_TRIANGLES);
 // face 1
 glColor3f(1.0f, 1.0f, 1.0f); // White
 glVertex3f(1.0f, -.5f, 1.0f);
 glVertex3f(0.0f, 1.0f, 0.0f);
 glVertex3f(-1.0f, -.5f, 0.0f);

 // face 2
 glColor3f(1.0f, 0.0f, 0.0f); // Red
 glVertex3f(-1.0f, -.5f, 0.0f);
 glVertex3f(0.0f, 1.0f, 0.0f);
 glVertex3f(1.0f, -.5f, -1.0f);

 // face 3
 glColor3f(0.0f, 1.0f, 0.0f); // Green
 glVertex3f(1.0f, -.5f, -1.0f);
 glVertex3f(0.0f, 1.0f, 0.0f);
 glVertex3f(1.0f, -.5f, 1.0f);

 glEnd();

 glutSwapBuffers();
}
```



gltest5.c (Builtin 3D Object: Teapot)

```
void
display(void)
{
 int i;

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 //gluLookAt(0.0, 50.0, -10.0, 0.0, 0.0, -10.0, 0.0, 0.0,
     1.0);

 //glTranslatef(0.0, 0.0, -10.0f);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glColor4f(1.0f, 1.0f, 1.0f, 1.0f);
 glutSolidTeapot(1.0f);

 glutSwapBuffers();
}
```


gltest6.c (Lighting & Other Builtin Objects)

```
void
init(void)
{
 //glClearColor(1.0, 1.0, 1.0, 0.0);
 glClearColor(0.0, 0.0, 0.0, 0.0);

 glEnable(GL_LIGHTING);
 glEnable(GL_LIGHT0);
}
```

```
void
display(void)
{
 int i;

 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

 //glTranslatef(0.0, 0.0, -10.0f);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glColor4f(1.0f, 0.0f, 0.0f, 1.0f); // useless
#if 0
 glScalef(1.0f, 1.0f, 1.0f);
 //glutSolidTetrahedron(); // 3
 //glutSolidOctahedron(); // 8
 glutSolidDodecahedron(); // 12
 //glutSolidIcosahedron(); // 20
#else
 glutSolidTeapot(1.0f);
#endif

 glutSwapBuffers();
}
```

gltest7.c (Material Color)

```
void
display(void)
{
 int i;
 float color[] = {1.0f, 0.0f, 0.0f, 1.0f};

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
//gluLookAt(0.0, 0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 // gluLookAt(0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
     1.0);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);
 glColor4f(1.0f, 0.0f, 0.0f, 1.0f); // useless
 glMaterialfv(GL_FRONT_AND_BACK,
     GL_DIFFUSE, color);
#if 0
 //glScalef(1.0f, 1.0f, 1.0f);
 //glutSolidTetrahedron(); // 3
 //glutSolidOctahedron(); // 8
 glutSolidDodecahedron(); // 12
 //glutSolidIcosahedron(); // 20
#else
 glutSolidTeapot(1.0f);
#endif

 glutSwapBuffers();
}
```

gltest7.c (視点を変える)

```
void
display(void)
{
 int i;
 float color[] = {1.0f, 0.0f, 0.0f, 1.0f};

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 //gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0,
     0.0);
//gluLookAt(0.0, 0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 gluLookAt(0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0);
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);
 glColor4f(1.0f, 0.0f, 0.0f, 1.0f); // useless
 glMaterialfv(GL_FRONT_AND_BACK,
     GL_DIFFUSE, color);
#if 0
 //glScalef(1.0f, 1.0f, 1.0f);
 //glutSolidTetrahedron(); // 3
 //glutSolidOctahedron(); // 8
 glutSolidDodecahedron(); // 12
 //glutSolidIcosahedron(); // 20
#else
 glutSolidTeapot(1.0f);
#endif

 glutSwapBuffers();
}
```

gltest8.c (回る2つの物体)

```
void
display(void)
{
 int i;
 float red[] = {1.0f, 0.0f, 0.0f, 1.0f};
 float green[] = {0.0f, 1.0f, 0.0f, 1.0f};

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 //gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 gluLookAt(0.0, 0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 //gluLookAt(0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0);
```

```
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, red);
 glutSolidTetrahedron();

 glTranslatef(0.0, 0.0, 10.0f);

 //glRotatef(g_r * 2.0f, 0.0f, 1.0f, 0.0f);
 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, green);
 glutSolidTeapot(1.0f);

 glutSwapBuffers();
}
```


gltest9.c (回る3つの物体)

```
void
display(void)
{
 int i;
 float red[] = {1.0f, 0.0f, 0.0f, 1.0f};
 float green[] = {0.0f, 1.0f, 0.0f, 1.0f};
 float purple[] = {1.0f, 0.0f, 1.0f, 1.0f};

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 //gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 gluLookAt(0.0, 0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 //gluLookAt(0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0);
```

```
  glRotatef(g_r, 0.0f, 1.0f, 0.0f);

  glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, red);
  glutSolidTetrahedron();

  glTranslatef(0.0, 0.0, 10.0f);

  glRotatef(g_r * 2.0f, 0.0f, 1.0f, 0.0f);
  glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, green);
  glutSolidTeapot(1.0f);

  glTranslatef(0.0, 0.0, 3.0f);

  glRotatef(g_r, 0.0f, 1.0f, 0.0f);
  glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, purple);
  glutSolidTorus(0.5f, 1.0f, 32, 32);

  glutSwapBuffers();
 }
```

gltest10.c (回る3つの物体: push/pop matrix)

```
void
display(void)
{
 int i;
 float red[] = {1.0f, 0.0f, 0.0f, 1.0f};
 float green[] = {0.0f, 1.0f, 0.0f, 1.0f};
 float purple[] = {1.0f, 0.0f, 1.0f, 1.0f};

 glClear(GL_COLOR_BUFFER_BIT |
     GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glEnable(GL_DEPTH_TEST);
 //gluLookAt(0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 gluLookAt(0.0, 0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
 //gluLookAt(0.0, 50.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0);
```

```
 glRotatef(g_r, 0.0f, 1.0f, 0.0f);

 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, red);
 glutSolidTetrahedron();

 glPushMatrix();
 glTranslatef(0.0, 0.0, 10.0f);

 glRotatef(g_r * 2.0f, 0.0f, 1.0f, 0.0f);
 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, green);
 glutSolidTeapot(1.0f);

 glPopMatrix();
 glTranslatef(0.0, 0.0, 3.0f);

 glRotatef(g_r, 0.0f, 1.0f, 0.0f);
 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, purple);
 glutSolidTorus(0.5f, 1.0f, 32, 32);

 glutSwapBuffers();
}
```