

© Professors d'IDI – Curs 2014-2015

Bloc_2: Transformacions Geomètriques i Models

Sessions i Objectius

- Sessió 1 –seccions 1 a 3- : Transformacions Geomètriques
 - Objectes glut
 - Entendre el funcionament de les transformacions geomètriques per: posicionar i animar objectes.
 - Utilització en OpenGL.
 - Exercici.
- Sessió 2 –seccions 4 i 5-:
 - Carregar models geomètrics (OBJ) i visualitzar en OpenGL.
 - Aplicació resum de conceptes: crear una escena concreta, poder girar l'escena, poder moure un dels objectes.

IDI- Q1 2014-2015

Primer exemple vist

```

...
void refresh (void) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glBegin(GL_TRIANGLES);
        glColor (1,0,0);
        glVertex3f(-0.5,0.0,0.0);
        glVertex3f(0.5,0.0,0.0);
        glVertex3f(0.0,0.5,0.0);
    glEnd();
    glutSwapBuffers(); }
int main(int argc, const char *argv[]) {
    glutInit(&argc, ((char **) argv);
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize(600,600);
    glutCreateWindow("IDI: Practiques OpenGL");
    glClearColor (0.5,0.5,0.5) => initGL
    glutDisplayFunc (refresh);
    ...
    glutMainLoop();
    return 0; }

```

1. Inicialitzacions glut

2. Registre *callbacks*

3. Bucle
processament
events

IDI- Q1 2014-2015

Repàs callbacks

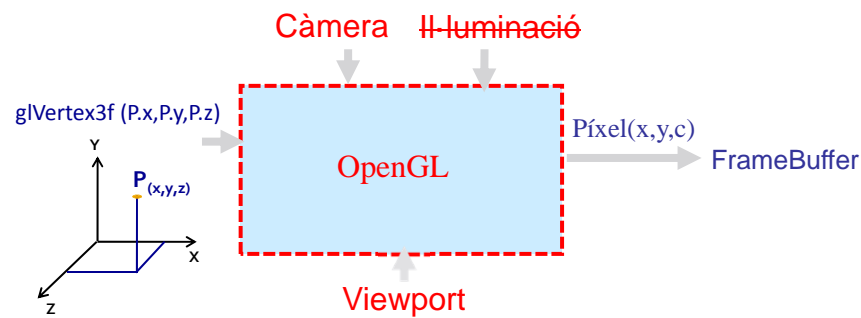
- void glutDisplayFunc (void (*funcName)void);
- void glutReshapeFunc (void (*func)(int width, int height));
- void glutKeyboardFunc (void (*func) (unsigned char key,
int x, int y));
- void glutMouseFunc (void (*func)(int button, int state,
int x, int y));
- void glutMotionFunc (void (*func) (int x,int y)
- ... altres que anirem veient

glutPostRedisplay(); Per marcar finestra per repintar

IDI- Q1 2014-2015

Repàs: OpenGL

■ Màquina d'estats

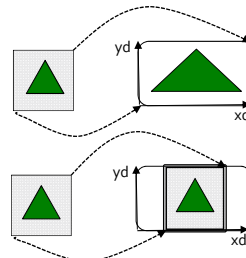
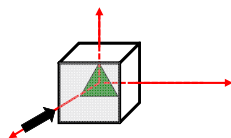


IDI- Q1 2014-2015

Funcions per modificar l'estat

- OpenGL té valors per defecte de les variables d'estat.
- De moment utilitzarem volum de visió per defecte:

Volum per defecte:
cub de $(-1, -1, -1)$ fins $(1, 1, 1)$



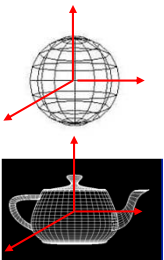
- Definició del *viewport* (`glViewport()`): mateixa relació d'aspecte que el *window* per a no tenir deformacions; per defecte tota la pantalla => registrar *callback* `glutReshapeFunc(resize)` per redefinir *viewport* (`glViewport(...)`) quan es modifica grandària finestra.

IDI- Q1 2014-2015

Secció 1: Objectes glut

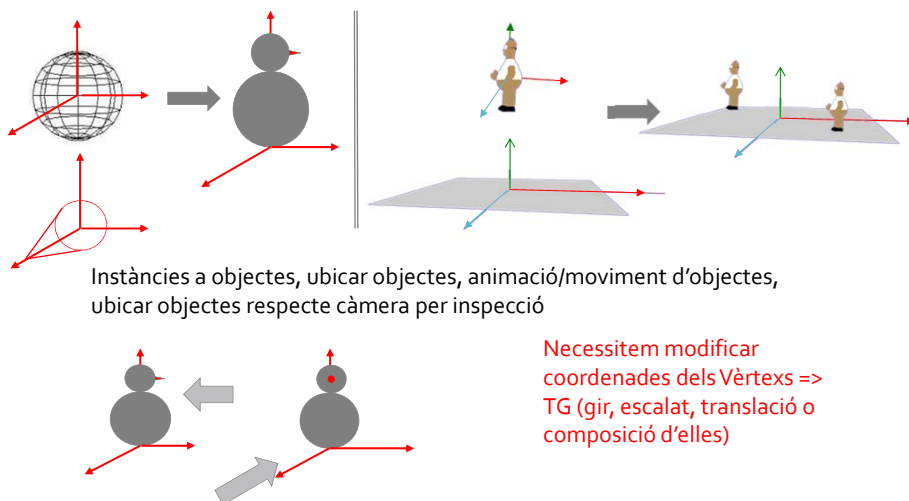
```
#include <GL/gl.h>
#include <GL/freeglut.h>
void refresh (void)
{ glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glBegin(GL_TRIANGLES);
    glVertex3f(-0.5,-0.5,0.0);
    glVertex3f(0.5,0.0,0.0);
    glVertex3f(0.0,0.5,0.0);
  glEnd();
  glutSwapBuffers();
}
int main(int argc, const char *argv[])
{ glutInit(&argc, ((char **) argv));
  glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGB);
  glutInitWindowSize(600,600);
  glutCreateWindow("IDI: Practiques OpenGL");
  glutDisplayFunc (refresh);
  glutMainLoop();
  return 0;
}
```

glutWireSphere(GLdouble radius, GLint slices, GLint stacks)
 glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks)
 glutWireTeapot(GLdouble size)
 ...



IDI- Q1 2014-2015

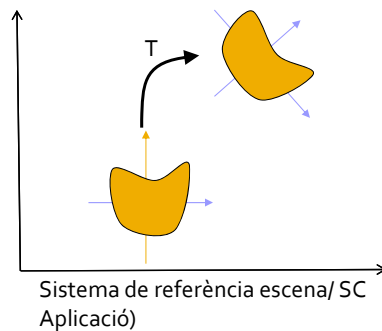
Secció 2: TG. Necessitat



IDI- Q1 2014-2015

Secció 2: TG. Repàs (mireu apunts racó)

- Transformacions geomètriques bàsiques: escalat, rotacions, translació



Transformació geomètrica →  Matriu 4x4 TG

IDI- Q1 2014-2015

Transformations geomètriques i OpenGL

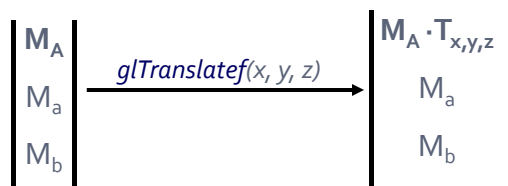
- OpenGL manté *dues piles de matrius* (la MODELVIEW i la PROJECTION). Només una pot estar activa per a ser modificada. Per defecte: la MODELVIEW

- La **matriu activa** M_A és sempre la del top de la **pila activa**

- Té instruccions per a **crear matrius de TG que afecten a la M_A** :

- `void glTranslatef(TYPE x, TYPE y, TYPE z);`
- `void glRotatef(TYPE angle, TYPE x, TYPE y, TYPE z);`
- `void glScalef(TYPE x, TYPE y, TYPE z);`

M_A
 M_2
 M_3



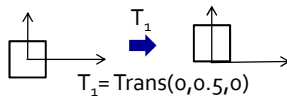
$M_A = M_A * TG$
Noteu l'ordre de la multiplicació!!!

IDI- Q1 2014-2015

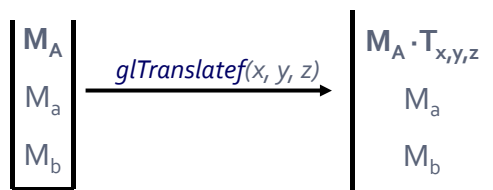
Transformations Geomètriques i OpenGL: exemple 1

- Quan s'envia a pintar un vèrtex V , li aplica primer la matriu del top de la pila MODELVIEW (el "mou") i després "el pinta".

$$V_transformat = M_A * V$$



```
glMatrixMode(GL_MODELVIEW);
glTranslatef(0, 0.5, 0);
glutWireCube(1);
```



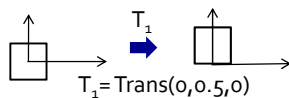
Segur? Quin valor M_A ?

IDI- Q1 2014-2015

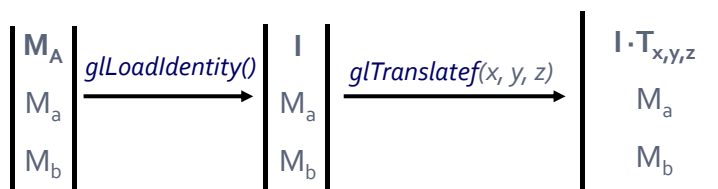
Transformations Geomètriques i OpenGL

- Quan s'envia a pintar un vèrtex V , li aplica primer la matriu del top de la pila MODELVIEW (el "mou") i després "el pinta".

$$V_transformat = M_A * V$$

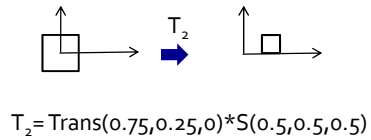


```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0.5, 0);
glutWireCube(1);
```



IDI- Q1 2014-2015

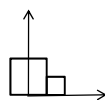
Transformations Geomètriques i OpenGL: Exemple 2



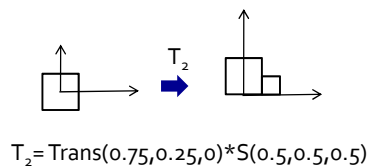
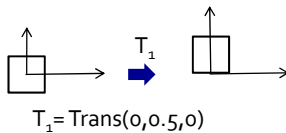
```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0.75, 0.25, 0);
glScalef(0.5, 0.5, 0.5);
glutWireCube(1);
```

IDI- Q1 2014-2015

TG i OpenGL: exemple 3



Escena a pintar utilitzant instàncies de glutWireCube(1)



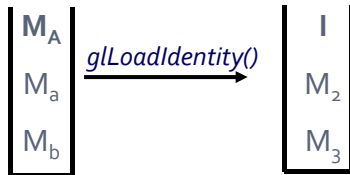
Comporta aplicar TG diferents als diferents objectes.

Com fer-ho amb OpenGL?

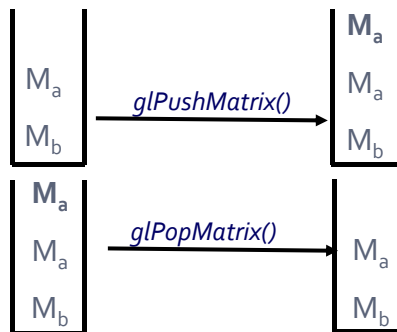
IDI- Q1 2014-2015

TG i OpenGL

- Substituir la matriu al top de la pila: (*glLoadIdentity*, *glLoadMatrix**)

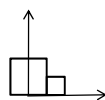


- Duplicar la matriu del top de la pila: *glPushMatrix()*
- Eliminar la matriu del top de la pila: *glPopMatrix()*

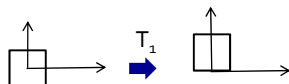


IDI- Q1 2014-2015

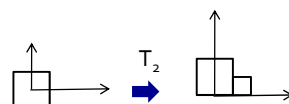
TG i OpenGL: exemple 3



Escena a pintar utilitzant instàncies de *glutWireCube()*



$$T_1 = \text{Trans}(0, 0.5, 0)$$



$$T_2 = \text{Trans}(0.75, 0.25, 0) * S(0.5, 0.5, 0.5)$$

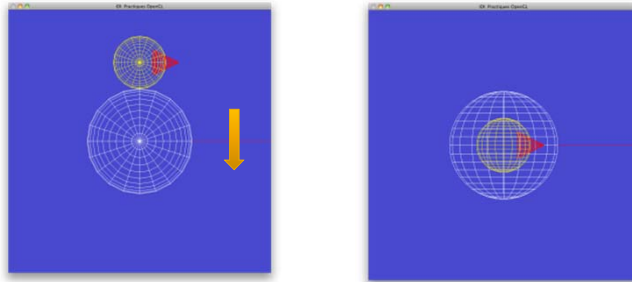
```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glPushMatrix();
glTranslatef(0, 0.5, 0);
glutWireCube(1);
glPopMatrix();
glPushMatrix();
glTranslatef(0.75, 0.25, 0);
glScalef(0.5, 0.5, 0.5);
glutWireCube(1);
glPopMatrix();
```

Ara gireu els dos cubs entorn de l'eix Y. Com?

IDI- Q1 2014-2015

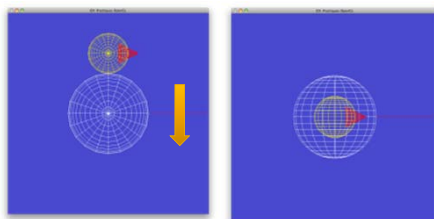
Girar tota l'escena

- Recordeu que càmera per defecte està en origen de coordenades, mirant en direcció de l'eix Z negatiu i és ortogonal i volum de visió (-1,-1,-1) a (1,1,1).
- Aplicant: $TG = G_x * G_y$ i movent ratolí per modificar angles => "diferents vistes"



IDI- Q1 2014-2015

Girar tota l'escena



Noteu que en cada "refresh" es comença la TG des de la Identitat (en un futur no serà així, ja ens ho trobarem ☺).

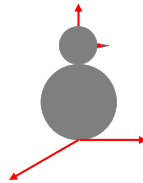
```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glRotated (alfa,1,0,0);
glRotated (beta,0,1,0);
glPushMatrix();
    pinta_ninot();
glPopMatrix();
```

```
glMatrixMode(GL_MODELVIEW);
// glLoadIdentity();
glPushMatrix();
glRotated (alfa,1,0,0);
glRotated (beta,0,1,0);
glPushMatrix();
    pinta_ninot();
glPopMatrix();
glPopMatrix();
```

IDI- Q1 2014-2015

Què heu de fer en 1ª sessió Bloc 2?

- Pintar algun objecte *glut* (secció 1)
- Utilitzar OpenGL per aplicar TG a un objecte (secció 2)
 - Entendre els paràmetres de les crides i composició d'operacions
 - Recordeu que aplica la matriu del top de la pila
 - Utilitzeu *callbacks* de teclat i ratolí per modificar TG; els reutilitzareu!!
- Utilitzar OpenGL per a aplicar diferents TG als diferents objectes de l'escena
 - Caldrà Push/Pop Matrius
- Crear una escena utilitzant objectes *glut* (secció 3) => TG per a crear el ninot i TG per a moure tot el ninot.



IDI- Q1 2014-2015