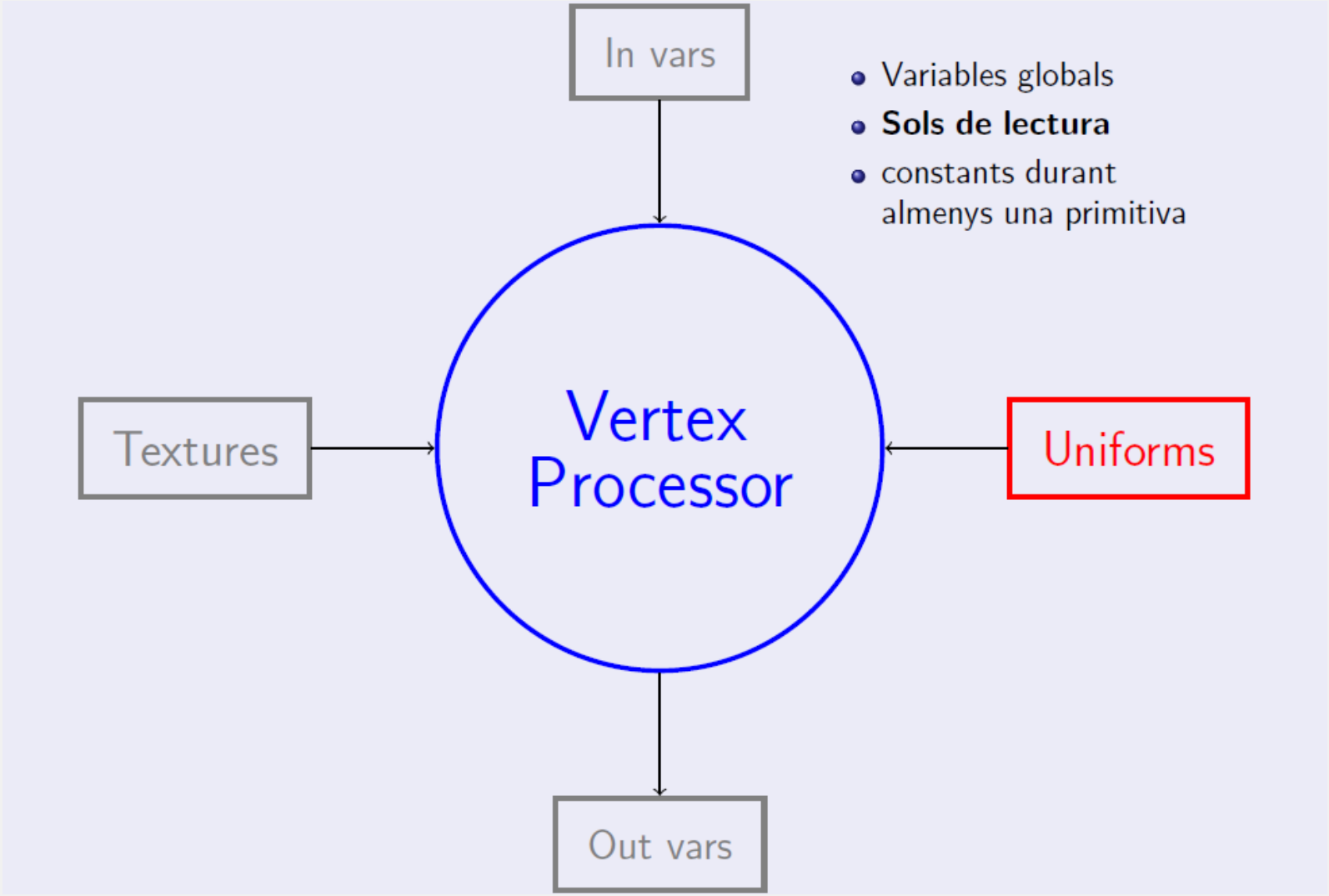
Laboratori OpenGL – Sessió 3

- Ús de *uniforms*
- Interacció directa a Qt
- Transformacions Geomètriques amb glm

Uniforms


In vars
Variables globals
Sols de lectura
constants durant almenys una primitiva
Textures
Vertex Processor
Uniforms
Out vars

Uniforms

- Al vertex shader:

```
# version 330 core

in vec3 vertex;
uniform float val;

void main ()
{
	gl_Position = vec4 (vertex * val, 1.0);
}
```

Uniforms

- Al codi cpp de QGLWidget:
  - Associar identificador al shader (només cal fer-ho un cop)

```
varLoc = glGetUniformLocation (program->programId (), “val”);
```

  - Donar valor al uniform (cal fer-ho cada cop que es vulgui canviar el valor del paràmetre *scl*)

```
glUniform1f (varLoc, scl);
// scl variable que conté el valor que es vol per “val”
```

Funcions OpenGL per a uniforms

GLint glGetUniformLocation (GLuint *program*, const GLchar **name*);

Obté la posició d'un uniform declarat al shader amb nom *name*

program : program al que està lligat el shader que conté el uniform

name : nom que identifica al uniform en el shader

void glUniform1f (GLint *location*, GLfloat *value*);

Especifica el valor *value* per al uniform identificat per *location*

location : identificador del uniform aconseguit amb glGetUniformLocation

value : valor que es passa cap al shader

Funcions OpenGL per a uniforms

Altres crides possibles:

glUniform{1|2|3|4}{f|i|ui} // nombre de paràmetres depenent de 1|2|3|4

1 – tipus float (f), int (i), unsigned int (ui), bool (f|i|ui)

2 – tipus vec2 (f), ivec2 (i), uvec2 (ui), bvec2 (f|i|ui)

3 – tipus vec3 (f), ivec3 (i), uvec3 (ui), bvec3 (f|i|ui)

3 – tipus vec4 (f), ivec4 (i), uvec4 (ui), bvec4 (f|i|ui)

glUniform{1|2|3|4}{f|i|ui}v (GLint *loc*, GLsizei *count*, const Type **value*);

{1|2|3|4} i {f|i|ui} – igual que crida anterior

count – nombre d'elements de l'array *value,* 1: un sol valor; >=1 array de valors

glUniformMatrix{2|3|4|2x3|3x2|2x4|4x2|3x4|4x3}fv
(GLint *loc*, GLsizei *count*, GLboolean transpose, const GLfloat **value*);

{2|3|4|2x3|3x2|2x4|4x2|3x4|4x3} – defineix les dimensions de la matriu

count – nombre de matrius de l'array *value*

transpose – si la matriu s'ha de transposar

Funcions OpenGL per a uniforms

Les que més usarem:

glUniform1{f|i|ui} // per a passar un únic valor

Exemple:

```
float scl = 0.5;
glUniform1f (varLoc, scl);
```

glUniform3fv // per a passar vectors de 3 components

Exemple:

```
glm::vec3 posLlum = glm::vec3 (1.0, 5.0, 0.0);
glUniform3f (posLlumLoc, &posLlum[0]);
```

glUniformMatrix4fv // per a passar les matrius de transformació

Exemple:

```
glm::mat4 TG = glm::mat4(1.0);
glUniformMatrix4fv (transLoc, 1, GL_FALSE, &TG[0][0])
```

Interacció directa amb Qt

- Per tal de tractar events de baix nivell en una aplicació OpenGL amb Qt cal re-implementar els mètodes virtuals corresponents:

```
virtual void mousePressEvent ( QMouseEvent * e )
virtual void mouseReleaseEvent ( QMouseEvent * e )
virtual void mouseMoveEvent ( QMouseEvent * e )
virtual void keyPressEvent ( QKeyEvent * e )
```

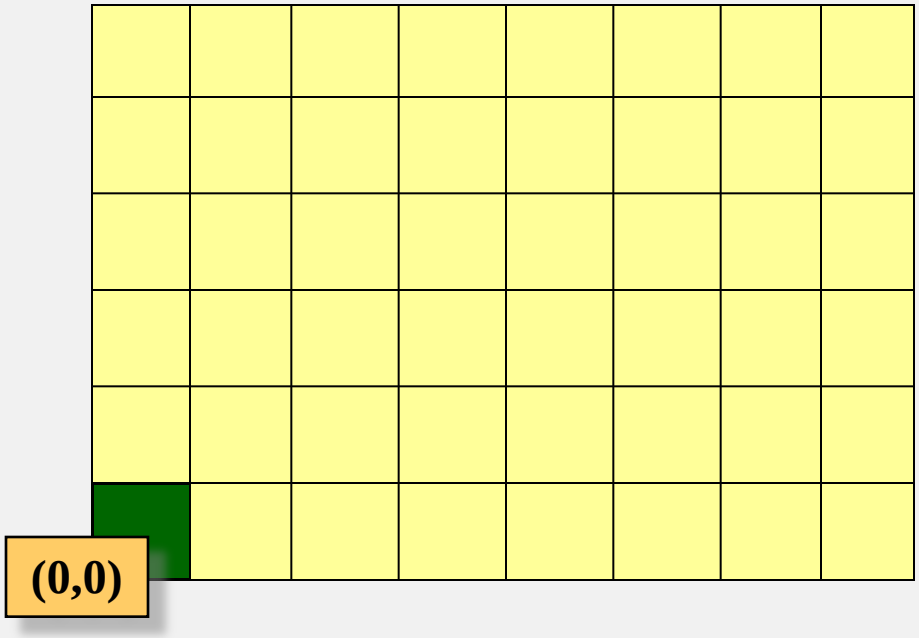
Interacció directa amb Qt

- Exemple d'implementació:

```
void MyGLWidget::keyPressEvent (QKeyEvent *e) {
  switch ( e->key() ) {
    case  Qt::Key_Escape :
          exit (0);
    case  Qt::Key_S :
          scl += 0.1;
          glUniform1f (varLoc, scl);
          updateGL ();
          break;
    case  Qt::Key_D :
          scl -= 0.1;
          glUniform1f (varLoc, scl);
          updateGL ();
          break;
    default: e->ignore (); // propagar al pare
  }
}
```

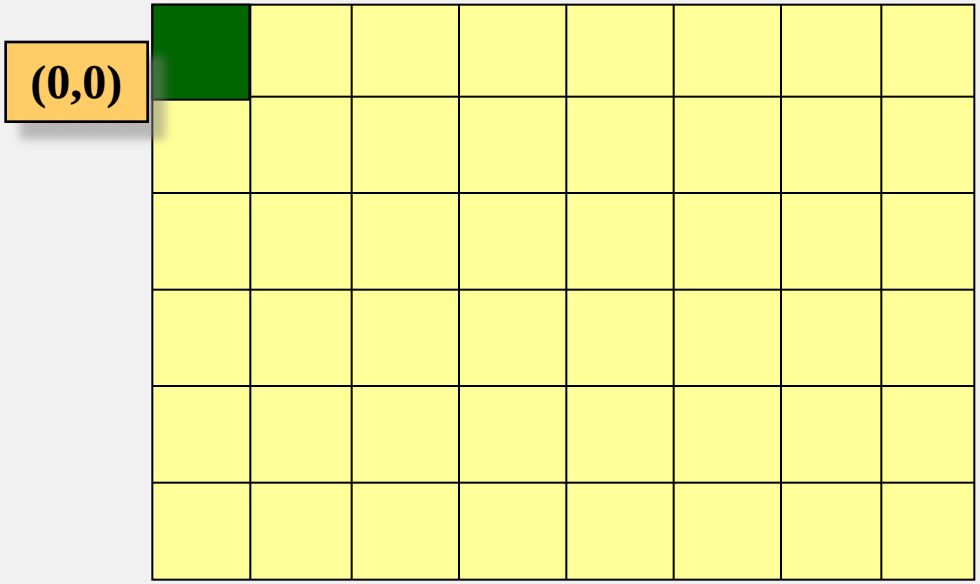
Consideració important

- OpenGL considera l'origen del SC de dispositiu a la cantonada inferior esquerra de la finestra gràfica.



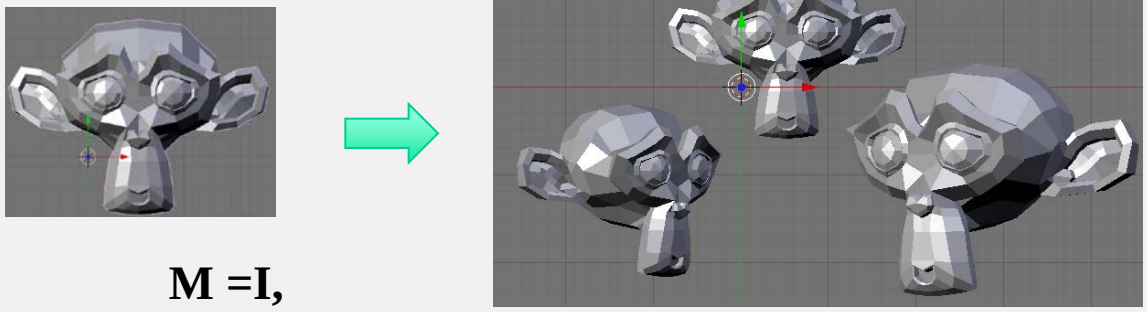
Consideració important

- Qt considera l'origen del SC de dispositiu a la cantonada superior esquerra de la finestra gràfica.

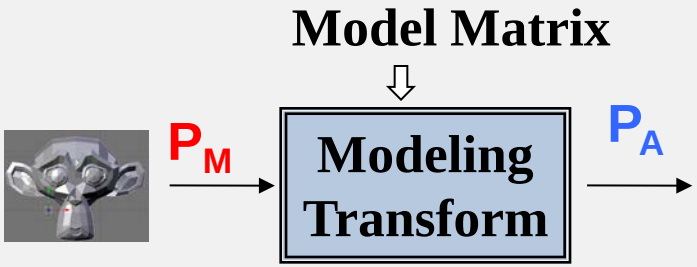


Matrius de transformació

- Hem de poder transformar els vèrtexs (pex, amb transformacions de model):

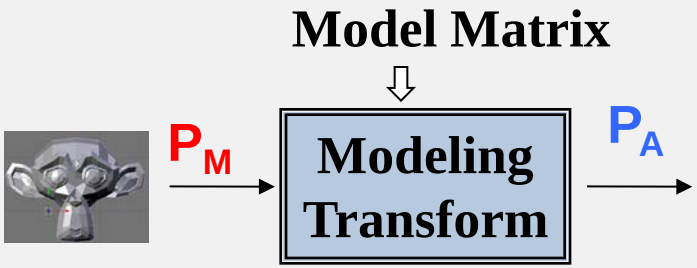


M =I,
TG_1, TG_2



Matrius de transformació

- Usarem glm per a construir la matriu de transformació:



- Exemple:

```
void MyGLWidget::modelTransform () {
    glm::mat4 TG;   // Matriu de transformació
    TG = glm::translate (glm::mat4(1.0), glm::vec3(-0.5, 0.5, 0.0));
    glUniformMatrix4fv (transLoc, 1, GL_FALSE, &TG[0][0]);
}
```

Matrius de transformació

- Mètodes de transformacions geomètriques de la glm:

  translate (glm::mat4 *m_ant*, glm::vec3 *vec_trans*);

  // acumula a la matriu anterior *m_ant* la matriu resultant de fer una
  // translació pel vector *vec_trans*

  scale (glm::mat4 *m_ant*, glm::vec3 *vec_scale*);

  // acumula a la matriu anterior *m_ant* la matriu resultant de fer un
  // escalat en cada direcció segons els factors *vec_scale*

  rotate (glm::mat4 *m_ant*, float *angle*, glm::vec3 *vec_axe*);

  // acumula a la matriu anterior *m_ant* la matriu resultant de fer una
  // rotació de *angle* radians al voltant de l'eix *vec_axe*

- Per a que els angles a usar a la rotació siguin en radians ens cal afegir al nostre codi (al fitxer MyGLWidget.h) el següent:

```
#define GLM_FORCE_RADIANS
```

Matrius de transformació

Per a poder usar l'exemple del mètode modelTransform() vist abans ens caldrà:

- al vertex shader:

```
in vec 3 vertex;
uniform mat4 TG;
void main () {
    gl_Position = TG * vec4 (vertex, 1.0);
}
```

- al nostre programa (en els llocs corresponents):

```
GLuint transLoc;
transLoc = glGetUniformLocation (program->programId(), "TG");
```

Exercicis sessió 3

El que cal que feu en aquesta sessió és:

1) Afegiu al vostre codi el uniform *scl* descrit en els exemples de codi vistos i feu, com hem vist, que amb les tecles ‘s’ i ‘d’ aquest valor del uniform augmenti o disminueixi respectivament.

2) Feu els exercicis que teniu al guió per a aquesta sessió.

   - L’exercici 5 demana un escalat no uniforme. Feu-lo usant el moviment del ratolí, de manera que quan el ratolí es mou d’esquerra a dreta s’incrementa el factor d’escala en X (i es decrementa en anar de dreta a esquerra), i quan el ratolí es mou de baix a dalt s’incrementa el fator d’escala en Y (i es decrementa en anar de dalt a baix).