constructora

angleY = angleX = 0.0;

modelHeight = 4; //alcada del model, depen d’aquest

initializeGL()

**viewTransform();**

**projectTransform();**

projectTransform()

glm::mat4 Proj; // Matriu de projecció

if (perspectiva)

Proj = glm::perspective(FOV, ra, radiEsc, 3.0f\*radiEsc);

else

Proj = glm::ortho(left, right, bottom, top, radiEsc, 3.0f\*radiEsc);

glUniformMatrix4fv (projLoc, 1, GL\_FALSE, &Proj[0][0]);

}

viewTransform()

glm::mat4 View; // Matriu de posició i orientació

View = glm::translate(glm::mat4(1.f), glm::vec3(0, 0, -distancia)); //Allunyem camara

View = glm::rotate(View, -angleX, glm::vec3(1, 0, 0));

View = glm::rotate(View, -angleY, glm::vec3(0, 1, 0));

View = glm::translate(View, -VRP); // Portem la camara al VRP

glUniformMatrix4fv (viewLoc, 1, GL\_FALSE, &View[0][0]);

}

resizeGL()

void MyGLWidget::*resizeGL* (int width, int height){

ra = (float)width/(float)height; // ra(v) = raw

//Recalculem el FOV per tal que no es retalli quan raw<1

if (width< height) {

FOV = 2 \* atan(tan(initFOV /2.f)/ra); //perspec

bottom = -radiEsc / ra; //ortho

top = radiEsc / ra; //ortho

}

if (width > height) {

left = -radiEsc \* ra; //ortho

right = radiEsc \* ra; //ortho

}

glViewport(0, 0, width, height);

projectTransform();

}

Mouse

En el void MyGLWidget::*mouseMoveEvent*(QMouseEvent \*e) afegir el calcul d angleX

angleX += (e->y() - yClick) \* M\_PI / 180.0;

.h

void maxSize();

void initCamera();

float initFOV, FOV, ra;

float left, right, top, bottom;

float distancia, maxHeight, modelHeight;

glm::vec3 VRP, sceneMax, sceneMin;

**float angleX;**

Noves funcions

void MyGLWidget::initCamera(){

viewTransform();

//Calcul de parametres inicials de camera perspectiva

initFOV = FOV = 2.0\*asin(radiEsc/distancia);

ra = 1.f;

//calcul de parametres inicials de camera ortogonal

right = top = radiEsc;

left = bottom = -radiEsc;

projectTransform();

}

void MyGLWidget::maxSize() {

//L'alçada del patricio max

maxHeight = modelHeight;

//Prenem 5 i -5 pqe son els maxims i els minim del terra (i de l'escena per tant)

sceneMax = glm::vec3(5, maxHeight, 5);

sceneMin = glm::vec3(-5, -1, -5); //alçada min = -1, que es on es situa el terra

radiEsc = glm::distance(sceneMax,sceneMin) / 2;

distancia = radiEsc\*2.f;

for (int i = 0; i < 3; ++i){

VRP[i] = (sceneMax[i] + sceneMin[i])/2.0;

}

}