Camera

1 Camera position

|  |
| --- |
| Vector3 cameraPos = new Vector3(0.0f, 0.0f, 0.3f); |

2. Camera direction

|  |
| --- |
| Vector3 cameraTarget = new Vector3(0.0f, 0.0f, 0.0f);  Vector3 cameraDirection = Vector3.Normalize(cameraPos - cameraTarget); |

3. Right axis

|  |
| --- |
| Vector3 up = new Vector3(0.0f, 1.0f, 0.0f);  Vector3 cameraRight = Vector3.Normalize(Vector3.Cross(up, cameraDirection)); |

4. Up axis

|  |
| --- |
| Vector3 cameraUp = Vector3.Cross(cameraDirection, cameraRight); |

Look At

OnRenderFrame()

|  |
| --- |
| ~~Matrix4 view = Matrix4.Identity;~~  ~~view = Matrix4.CreateTranslation(0.0f, 0.0f, -3.0f);~~  Matrix4 view;  view = Matrix4.LookAt(  new Vector3(0.0f, 0.0f, 3.0f),  new Vector3(0.0f, 0.0f, 0.0f),  new Vector3(0.0f, 1.0f, 0.0f)  ); |

|  |
| --- |
| float radius = 10.0f;  float camX = MathF.Sin((float)GLFW.GetTime()) \* radius;  float camZ = MathF.Cos((float)GLFW.GetTime()) \* radius;  Matrix4 view;  view = Matrix4.LookAt(  new Vector3(camX, 0.0f, camZ),  new Vector3(0.0f, 0.0f, 0.0f),  new Vector3(0.0f, 1.0f, 0.0f)  ); |

Link: <https://learnopengl.com/video/getting-started/camera_circle.mp4>

|  |
| --- |
| Vector3 cameraPos = new Vector3(0.0f, 0.0f, 0.3f);  Vector3 cameraFront = new Vector3(0.0f, 0.0f, -1.0f);  Vector3 cameraUp = new Vector3(0.0f, 1.0f, 0.0f); |

Isso aqui tem que ser variáveis fora da função

OnRenderFrame()

|  |
| --- |
| ~~view = Matrix4.LookAt(~~  ~~new Vector3(camX, 0.0f, camZ),~~  ~~new Vector3(0.0f, 0.0f, 0.0f),~~  ~~new Vector3(0.0f, 1.0f, 0.0f)~~  ~~);~~  view = Matrix4.LookAt(cameraPos, cameraPos + cameraFront, cameraUp); |

|  |
| --- |
| void processInput() {  float cameraSeep = 0.05f;  if(KeyboardState.IsKeyDown(Keys.W)) {  cameraPos += Vector3.Normalize(new Vector3(cameraFront.X, 0, cameraFront.Z)) \* cameraSpeed;  }  if(KeyboardState.IsKeyDown(Keys.S)) {  cameraPos -= Vector3.Normalize(new Vector3(cameraFront.X, 0, cameraFront.Z)) \* cameraSpeed;  }  if(KeyboardState.IsKeyDown(Keys.A)) {  cameraPos -= Vector3.Normalize(Vector3.Cross(cameraFront, cameraUp)) \* cameraSeep;  }  if(KeyboardState.IsKeyDown(Keys.D)) {  cameraPos += Vector3.Normalize(Vector3.Cross(cameraFront, cameraUp)) \* cameraSeep;  }  } |

|  |
| --- |
| if(KeyboardState.IsKeyDown(Keys.Space)) {  cameraPos += cameraUp \* cameraSpeed;  }  if(KeyboardState.IsKeyDown(Keys.LeftShift)) {  cameraPos -= cameraUp \* cameraSpeed;  } |

|  |
| --- |
| protected override void OnUpdateFrame(FrameEventArgs args) {  base.OnUpdateFrame(args);  [...]  processInput();  } |

|  |
| --- |
| float deltaTime = 0.0f;  float lastFrame = 0.0f; |

OnRenderFrame()

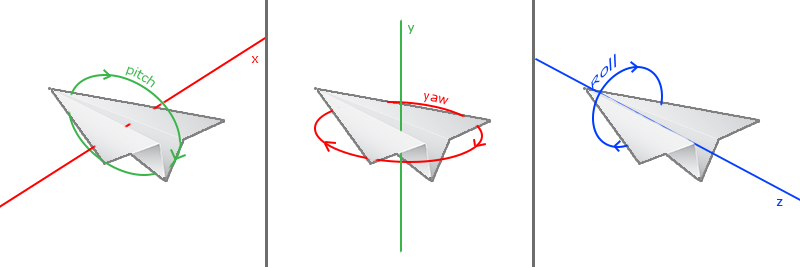
|  |
| --- |
| float currentFrame = (float)GLFW.GetTime();  deltaTime = currentFrame - lastFrame;  lastFrame = currentFrame; |

|  |
| --- |
| void processInput() {  float cameraSpeed = 2.5f \* deltaTime;  [...]  } |

Link: <https://learnopengl.com/video/getting-started/camera_smooth.mp4>

Look around

Euler angles



Tela de computador com luz verde

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente

|  |
| --- |
| void processInput() {  float cameraSpeed = 2.5f \* deltaTime;  [...]  } |

|  |
| --- |
| float yaw = -90.0f;  float pitch; |

Vamos fazer uma função chamada mouse\_callback() e chama-la em OnUpdateFrame()

|  |
| --- |
| void mouse\_callback() {  [...]  Vector3 direction;  direction.X = (float)Math.Cos(MathHelper.DegreesToRadians(yaw));  direction.Z = (float)Math.Sin(MathHelper.DegreesToRadians(yaw));  } |

Uma imagem contendo Diagrama

Descrição gerada automaticamente

|  |
| --- |
| direction.Y = (float)Math.Sin(MathHelper.DegreesToRadians(pitch)); |

|  |
| --- |
| direction.X = (float)Math.Cos(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  direction.Y = (float)Math.Sin(MathHelper.DegreesToRadians(pitch));  direction.Z = (float)Math.Sin(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch)); |

Mouse input

OnLoad()

|  |
| --- |
| CursorState = CursorState.Grabbed; |

|  |
| --- |
| void mouse\_callback(float xpos, float ypos) {  [...]  } |

OnUpdateFrame()

|  |
| --- |
| mouse\_callback(MouseState.X, MouseState.Y); |

Isso é pra fora

|  |
| --- |
| float lastX = 400.0f;  float lastY = 300.0f; |

mouse\_callback()

|  |
| --- |
| float xoffset = xpos - lastX;  float yoffset = ypos - lastY;  lastX = xpos;  lastY = ypos;  float sensitivity = 0.1f;  xoffset \*= sensitivity;  yoffset \*= sensitivity; |

|  |
| --- |
| yaw += xoffset;  pitch -= yoffset; |

|  |
| --- |
| if(pitch > 89.0f) {  pitch = 89.0f;  }  if(pitch < -89.0f) {  pitch = -89.0f;  } |

|  |
| --- |
| Vector3 direction;  direction.X = (float)Math.Cos(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  direction.Y = (float)Math.Sin(MathHelper.DegreesToRadians(pitch));  direction.Z = (float)Math.Sin(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  cameraFront = Vector3.Normalize(direction); |

|  |
| --- |
| void mouse\_callback(float xpos, float ypos) {  if(firstMouse) {  lastX = xpos;  lastY = ypos;  firstMouse = false;  }  [...]  } |

|  |
| --- |
| void mouse\_callback(float xpos, float ypos) {  if(firstMouse) {  lastX = xpos;  lastY = ypos;  firstMouse = false;  }  float xoffset = xpos - lastX;  float yoffset = ypos - lastY;  lastX = xpos;  lastY = ypos;  float sensitivity = 0.1f;  xoffset \*= sensitivity;  yoffset \*= sensitivity;  yaw += xoffset;  pitch -= yoffset;  if(pitch > 89.0f) {  pitch = 89.0f;  }  if(pitch < -89.0f) {  pitch = -89.0f;  }  Vector3 direction;  direction.X = (float)Math.Cos(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  direction.Y = (float)Math.Sin(MathHelper.DegreesToRadians(pitch));  direction.Z = (float)Math.Sin(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  cameraFront = Vector3.Normalize(direction);  } |

Zoom

|  |
| --- |
| float fov = 45.0f; |

|  |
| --- |
| void scrool\_callback(double xoffset, double yoffset) {  fov -= (float)yoffset;  if(fov < 1.0f) {  fov = 1.0f;  }  if(fov > 45.0f) {  fov = 45.0f;  }  } |

OnRenderFrame()

|  |
| --- |
| projection = Matrix4.CreatePerspectiveFieldOfView(MathHelper.DegreesToRadians(fov), 800.0f / 600.0f, 0.1f, 100.0f); |

|  |
| --- |
| protected override void OnMouseWheel(MouseWheelEventArgs e) {  base.OnMouseWheel(e);  scrool\_callback(e.OffsetX, e.OffsetY);  } |

Link: <https://learnopengl.com/video/getting-started/camera_mouse.mp4>

Camera Class

Basicamente o tutorial diz... se vira. Ou algo motivador do tipo “agora tente você mesmo”, sei la, eu não sei inglês.

Vamos la... cria a p\*\*\* da classe ae

|  |
| --- |
| namespace LearnOpenTK.src {  internal class Camera {  }  } |

|  |
| --- |
| using OpenTK.Graphics.OpenGL4;  using OpenTK.Mathematics;  using OpenTK.Windowing.GraphicsLibraryFramework;  namespace LearnOpenTK.src {  internal class Camera {  Vector3 cameraPos = new Vector3(0.0f, 0.0f, 0.3f);  Vector3 cameraFront = new Vector3(0.0f, 0.0f, -1.0f);  Vector3 cameraUp = new Vector3(0.0f, 1.0f, 0.0f);  float deltaTime = 0.0f;  float lastFrame = 0.0f;  bool firstMouse;  float lastX = 400.0f;  float lastY = 300.0f;  float yaw = -90.0f;  float pitch;  float fov = 45.0f;  public void uma\_funcao\_a\_e(Shader ourShader) {  Vector3 cameraTarget = new Vector3(0.0f, 0.0f, 0.0f);  Vector3 cameraDirection = Vector3.Normalize(cameraPos - cameraTarget);  Vector3 up = new Vector3(0.0f, 1.0f, 0.0f);  Vector3 cameraRight = Vector3.Normalize(Vector3.Cross(up, cameraDirection));  //Vector3 cameraUp = Vector3.Cross(cameraDirection, cameraRight);  float radius = 10.0f;  float camX = (float)Math.Sin((float)GLFW.GetTime()) \* radius;  float camZ = (float)Math.Cos((float)GLFW.GetTime()) \* radius;  Matrix4 view;  view = Matrix4.LookAt(cameraPos, cameraPos + cameraFront, cameraUp);  float currentFrame = (float)GLFW.GetTime();  deltaTime = currentFrame - lastFrame;  lastFrame = currentFrame;  Matrix4 projection;  projection = Matrix4.CreatePerspectiveFieldOfView(MathHelper.DegreesToRadians(fov), 800.0f / 600.0f, 0.1f, 100.0f);  int viewLoc = GL.GetUniformLocation(ourShader.ID, "view");  GL.UniformMatrix4(viewLoc, false, ref view);  int projectionLoc = GL.GetUniformLocation(ourShader.ID, "projection");  GL.UniformMatrix4(projectionLoc, false, ref projection);  }  public void processInput(KeyboardState input) {  float cameraSpeed = 2.5f \* deltaTime;  if(input.IsKeyDown(Keys.W)) {  //cameraPos += cameraSpeed \* cameraFront;  cameraPos += Vector3.Normalize(new Vector3(cameraFront.X, 0, cameraFront.Z)) \* cameraSpeed;  }  if(input.IsKeyDown(Keys.S)) {  //cameraPos -= cameraSpeed \* cameraFront;  cameraPos -= Vector3.Normalize(new Vector3(cameraFront.X, 0, cameraFront.Z)) \* cameraSpeed;  }  if(input.IsKeyDown(Keys.A)) {  cameraPos -= Vector3.Normalize(Vector3.Cross(cameraFront, cameraUp)) \* cameraSpeed;  }  if(input.IsKeyDown(Keys.D)) {  cameraPos += Vector3.Normalize(Vector3.Cross(cameraFront, cameraUp)) \* cameraSpeed;  }  if(input.IsKeyDown(Keys.Space)) {  cameraPos += cameraUp \* cameraSpeed;  }  if(input.IsKeyDown(Keys.LeftShift)) {  cameraPos -= cameraUp \* cameraSpeed;  }  }  public void mouse\_callback(float xpos, float ypos) {  if(firstMouse) {  lastX = xpos;  lastY = ypos;  firstMouse = false;  }  float xoffset = xpos - lastX;  float yoffset = ypos - lastY;  lastX = xpos;  lastY = ypos;  float sensitivity = 0.1f;  xoffset \*= sensitivity;  yoffset \*= sensitivity;  yaw += xoffset;  pitch -= yoffset;  if(pitch > 89.0f) {  pitch = 89.0f;  }  if(pitch < -89.0f) {  pitch = -89.0f;  }  Vector3 direction;  direction.X = (float)Math.Cos(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  direction.Y = (float)Math.Sin(MathHelper.DegreesToRadians(pitch));  direction.Z = (float)Math.Sin(MathHelper.DegreesToRadians(yaw)) \* (float)Math.Cos(MathHelper.DegreesToRadians(pitch));  cameraFront = Vector3.Normalize(direction);  }  public void scrool\_callback(double xoffset, double yoffset) {  fov -= (float)yoffset;  if(fov < 1.0f) {  fov = 1.0f;  }  if(fov > 45.0f) {  fov = 45.0f;  }  }  }  } |

Na classe principal

|  |
| --- |
| Camera camera = new Camera(); |

OnUpdateFrame()

|  |
| --- |
| camera.processInput(KeyboardState);  camera.mouse\_callback(MouseState.X, MouseState.Y); |

OnMouseWheel()

|  |
| --- |
| camera.scrool\_callback(e.OffsetX, e.OffsetY); |

OnRenderFrame()

|  |
| --- |
| camera.uma\_funcao\_a\_e(ourShader); |