How to Calculate the Positon and Rotation of Camera Using Projection Matrix

Yubaoliu89@gmail.com

# Purpose

Draw the Camera’s behavior in Unity 3D

# Preparation

## Camera.txt

For example:

0

0.000000 -1.000000 -0.000000 -0.050000

-1.000000 0.000000 0.000000 -0.002000

0.000000 0.000000 -1.000000 0.000000

Explanation:

The first line is image frame number. And the others are the projection Matrix. The first 3\*3 Matrix is the Rotation Matrix (R) and the last column is the Transform Vector (T).

This file is generated by OpenGL, and it is Right-hand based. However unity is Left-hand based system. Therefore the transform is necessary in the code.

# Algorithm

## Transformation form World Space to Camera Space

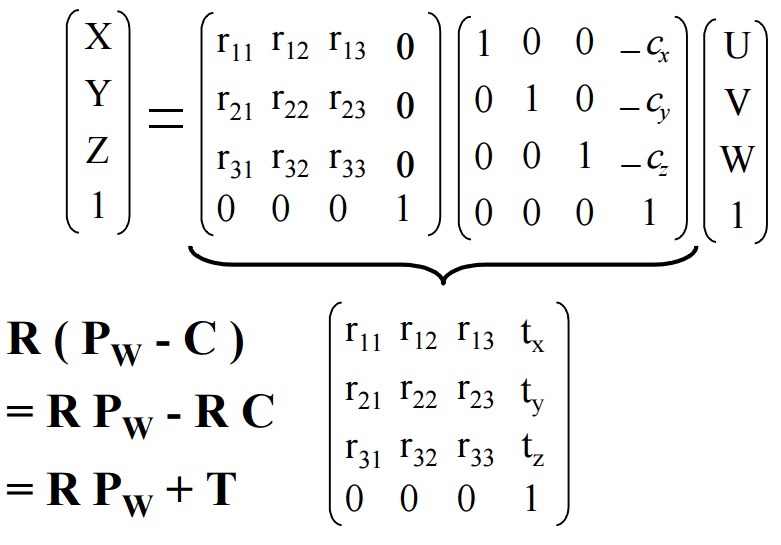


Figure 1 World coordinate to Camera Coordinate

In Figure 1, the variable ‘C’ is the Camera’s position in world-Coordinate Space, and ‘Pw’ is a point in World-Coordinate system. This function indicates the transform from World-Coordinate System to Camera-Coordinate system.

Therefore, the camera’s position is , and the rotation behavior of camera is

After that, we need to calculate the Euler angle according to Rc.

## Math Function

Table 1 atan2 in Unity

|  |
| --- |
| [Mathf](https://docs.unity3d.com/ScriptReference/Mathf.html).Atan2 public static float **Atan2**(float **y**, float **x**); Description Returns the angle in radians whose [Tan](https://docs.unity3d.com/ScriptReference/Mathf.Tan.html) is y/x.  Return value is the angle between the x-axis and a 2D vector starting at zero and terminating at (x,y).  **Note:** This function takes account of the cases where x is zero and returns the correct angle rather than throwing a division by zero exception.  using UnityEngine;  using System.Collections;  public class ExampleClass : [MonoBehaviour](https://docs.unity3d.com/ScriptReference/MonoBehaviour.html) {  public [Transform](https://docs.unity3d.com/ScriptReference/Transform.html) target;  void Update() {  [Vector3](https://docs.unity3d.com/ScriptReference/Vector3.html) relative = transform.InverseTransformPoint(target.position);  float angle = [Mathf.Atan2](https://docs.unity3d.com/ScriptReference/Mathf.Atan2.html)(relative.x, relative.z) \* [Mathf.Rad2Deg](https://docs.unity3d.com/ScriptReference/Mathf.Rad2Deg.html);  transform.Rotate(0, angle, 0);  }  }  Refer: <https://docs.unity3d.com/ScriptReference/Mathf.Atan2.html> |

# Source Code Example

## Unity C# example

|  |
| --- |
| Matrix4x4[] cameraPos = new Matrix4x4[1501];  int readFrame(string filePath)  {  StreamReader sr = new StreamReader(filePath, Encoding.Default);  string line;  Matrix4x4 RT = Matrix4x4.zero;  while ((line = sr.ReadLine()) != null)  {  sequence = Convert.ToInt32(line);  Debug.Log("sequence: " + sequence);  for (int i = 0; i < 3; i++)  {  string[] data = new string[4];  line = sr.ReadLine();  data = line.Split(' ');  for (int j = 0; j < 4; j++)  {  RT[i + j \* 4] = Convert.ToSingle(data[j]);  }  }  RT.m33 = 1;  cameraPos[sequence] = RT;  }  return 0;  } |

Figure 2 Read Each Frame using Unity C#

|  |
| --- |
| int calculateCameraRotateTransformFromProjectionMatrix(Matrix4x4 P, ref Matrix4x4 cameraRotation, ref Vector4 cameraPosition)//P is the projection matrix  {  Vector4 T = new Vector4(P.m03, P.m13, P.m23, P.m33);  Matrix4x4 R = Matrix4x4.zero;  R.m33 = 1.0f;  for (int i = 0; i < 3; i++)  {  for (int j = 0; j < 3; j++)  {  R[i + j \* 4] = P[i + j \* 4];  }  }  cameraRotation = R.inverse;  cameraPosition = R.inverse \* T \* (-1.0f);  return 0;  }  int computingEulerAnglesFromaRotationMatrix(Matrix4x4 R, ref Vector4 eulerangle)  {  float R11 = R.m00;  float R12 = R.m01;  float R13 = R.m02;  float R21 = R.m10;  float R22 = R.m11;  float R23 = R.m12;  float R31 = R.m20;  float R32 = R.m21;  float R33 = R.m22;  float thet1 = 0f, thet2 = 0f, psi1 = 0f, psi2 = 0f, phi1 = 0f, phi2 = 0f;  float thet = 0f, phi = 0f, psi = 0;  if (!(R31 + 1.0f < 0.0000000001f) || !(R31 - 1 < 0.0000000001f))  {  thet1 = (-1.0f) \* Mathf.Asin(R31);  thet2 = Mathf.PI - thet1;  psi1 = Mathf.Atan2(R32 / Mathf.Cos(thet1), R33 / Mathf.Cos(thet1));  psi2 = Mathf.Atan2(R32 / Mathf.Cos(thet2), R33 / Mathf.Cos(thet2));  phi1 = Mathf.Atan2(R21 / Mathf.Cos(thet1), R11 / Mathf.Cos(thet1));  phi2 = Mathf.Atan2(R21 / Mathf.Cos(thet2), R11 / Mathf.Cos(thet2));  //both 1 and 2 are valid, here use 1  eulerangle.x = Mathf.Rad2Deg \* psi1;  eulerangle.y = Mathf.Rad2Deg \* thet1;  eulerangle.z = Mathf.Rad2Deg \* phi1;  }  else  {  phi = 0;  if (R31 + 1.0f < 0.0000000001f)  {  thet = Mathf.PI / 2;  psi = phi + Mathf.Atan2(R12, R13);  }  else  {  thet = (-1.0f) \* Mathf.PI / 2.0f;  psi = (-1.0f) \* psi + Mathf.Atan2((-1.0f) \* R12, (-1.0f) \* R13);  }  eulerangle.x = Mathf.Rad2Deg \* psi;  eulerangle.y = Mathf.Rad2Deg \* thet;  eulerangle.z = Mathf.Rad2Deg \* phi;  }  return 0;  } |

Figure 3 Calculate Euler Angle and Position of Camera

|  |
| --- |
| int frameIndex = 0;  void Update()  {  Matrix4x4 projectionMatrix = cameraPos[++frameIndex % TotalFrameNumber];  Matrix4x4 cameraRotation = Matrix4x4.zero;  Vector4 cameraPosition = Vector4.zero;  calculateCameraRotateTransformFromProjectionMatrix(projectionMatrix, ref cameraRotation, ref cameraPosition);  transform.position = new Vector3(cameraPosition.x, cameraPosition.y, -cameraPosition.z);  Vector4 eulerangle = Vector4.zero;  computingEulerAnglesFromaRotationMatrix(cameraRotation, ref eulerangle);  transform.rotation = new Quaternion(-eulerangle.x, -eulerangle.y, eulerangle.z, 0);  transform.Rotate(Vector3.right, 180);  } |

Figure 4 Main Function

## MATLAB Example to Calculate the Euler Angle According to the Rotation Matrix

|  |
| --- |
| %calculate Euler Angle according to Rotate Matrix  yd = asind(-R(3,1));  if cos(yd) ~= 0  if R(1,1)/cosd(yd) > 1  zd = acosd(1);  elseif R(1,1)/cosd(yd) < -1  zd = acosd(-1);  else  zd = atand(R(2,1)/R(1,1));  end    if R(3,3)/cosd(yd) > 1  xd = acosd(1);  elseif R(3,3)/cosd(yd) < -1  xd = acosd(-1);  else  xd = asind(R(3,2)/cosd(yd));  end  else    end    %output: [-pi/2,pi/2]  r = [xd; yd; zd] |

Figure 5 MATLAB Example

The Euler Angle’s value is confined in [-pi/2, pi/2] using this method. From my personal view, I recommend the method described in Unity code.

# Attached Files

1. Computing\_Euler\_angles\_from\_a\_rotation\_matrix.pdf: How to calculate the Euler angle from rotation matrix.
2. calculate\_Euler\_Angle\_from\_Rotate\_Matrics .m: A MATLAB example to calculate Euler angle