Annya Paulina Verduzco Meza
A01650668

Datos dados

```
ILLUMINATION DATA
kar = 0.27716           kag = 0.28804           kab = 0.13353
kdr = 0.63956           kdg = 0.01195           kdb = 0.07031
ksr = 0.35586           ksg = 0.36302           ksb = 0.39225
Iar = 0.83657           Iag = 0.90372           Iab = 0.84783
Idr = 0.94256           Idg = 0.87530           Idb = 0.87470
Isr = 0.81019           Isg = 0.97526           Isb = 0.97796
alpha = 93
GEOMETRY DATA
Sphere radius:           SR =  0.53503
To obtain the Sphere Center (SC), do the following HOMOGENEOUS TRANSFORMATIONS:
Consider point A = ( 1.77695,   -1.64028,       -0.29728)
    1. A' is: translate point A by ( 0.96938,   -0.61295,       -0.01015)
    2. Consider the pivot P = ( 0.82048,        -0.80133,       -0.69552)
    3. SC is: rotate point A' by  1.0° around P with axis Z.
PoI inclination angle:    i = 150°
PoI azimuth angle:        a = 105°
SCENE DATA
Point light position: LIGHT = ( 3.43526,        -4.06723,       -4.63959)
Camera position:        CAM = ( 4.53517,        -4.48071,        2.87276)
```

Necesitamos encontrar el centro y el PoI (punto de interés).

Capturamos los datos dados para esta tarea en el código de Ilumination y nos aseguramos de agregar el script de Transformations a la escena, posteriormente, asignamos el script de Illumination a un objeto de 3d creado (esfera).

Ilumination.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Ilumination : MonoBehaviour
{
    Dictionary<string, Vector3> vec3data;
    Dictionary<string, float> floatdata;

    // Start is called before the first frame update
    void Start()
    {
        // Datos iniciales
        vec3data = new Dictionary<string, Vector3>();
        floatdata = new Dictionary<string, float>();

        vec3data.Add("LUZ", new Vector3(3.43526f, -4.06723f, -4.63959f));
        vec3data.Add("CAMARA", new Vector3(4.53517f, -4.48071f, 2.87276f));
        floatdata.Add("RADIO", 0.53503f);
        floatdata.Add("ALPHA", 93f);

        vec3data.Add("ka", new Vector3(0.27716f, 0.28804f, 0.13353f));
        vec3data.Add("kd", new Vector3(0.63956f, 0.01195f, 0.07031f));
        vec3data.Add("ks", new Vector3(0.35586f, 0.36302f, 0.39225f));

        vec3data.Add("Ia", new Vector3(0.83657f, 0.90372f, 0.84783f));
        vec3data.Add("Id", new Vector3(0.94256f, 0.87530f, 0.87470f));
        vec3data.Add("Is", new Vector3(0.81019f, 0.97526f, 0.97796f));

        vec3data.Add("A", new Vector3(1.77695f, -1.64028f, -0.29728f));
        vec3data.Add("tm", new Vector3(0.96938f, -0.61295f, -0.01015f));
        vec3data.Add("P", new Vector3(0.82048f, -0.80133f, -0.69552f));

        floatdata.Add("iAngle", 150f);
        floatdata.Add("aAngle", 105f);

        // Calculamos la traslación
        Vector4 transform = new Vector4(vec3data["A"].x, vec3data["A"].y, vec3data["A"].z, 1);
        Matrix4x4 tm = Transformations.TranslateM(vec3data["tm"].x, vec3data["tm"].y, vec3data["tm"].z);
        transform = tm * transform;

        // Rotamos respecto a un punto P
        Matrix4x4 rm = Transformations.RotateM(1.0f, Transformations.AXIS.AX_Y);
        Matrix4x4 siP = Transformations.TranslateM(vec3data["P"].x, vec3data["P"].y, vec3data["P"].z);
        Matrix4x4 noP = Transformations.TranslateM(-vec3data["P"].x, -vec3data["P"].y, -vec3data["P"].z);
        // Es importante el orden de las matrices
        // Centro de la esfera
        Vector4 result = noP * rm * siP * transform;

        // Convertimos el ángulo a radianes
        float iAngle = floatdata["iAngle"] * Mathf.Deg2Rad;
        float aAngle = floatdata["aAngle"] * Mathf.Deg2Rad;

        // El punto de interes
        // Utilizamos la formúla de la clase para obtener el punto de interes
        float x = result.x + floatdata["RADIO"] * Mathf.Sin(iAngle) * Mathf.Sin(aAngle);
        float y = result.y + floatdata["RADIO"] * Mathf.Cos(iAngle);
        float z = result.z + floatdata["RADIO"] * Mathf.Sin(iAngle) * Mathf.Cos(aAngle);

        vec3data.Add("CENTRO", result);
        Vector3 PoI = new Vector3(x, y, z);
        vec3data.Add("PoI", PoI);

        Vector3 n = vec3data["PoI"] - vec3data["CENTRO"];
        vec3data.Add("N", n);

        Vector3 v = vec3data["CAMARA"] - vec3data["PoI"];
        vec3data.Add("V", v);

        Vector3 l = vec3data["LUZ"] - vec3data["PoI"];
        vec3data.Add("L", l);
```

Line 52, Column 60

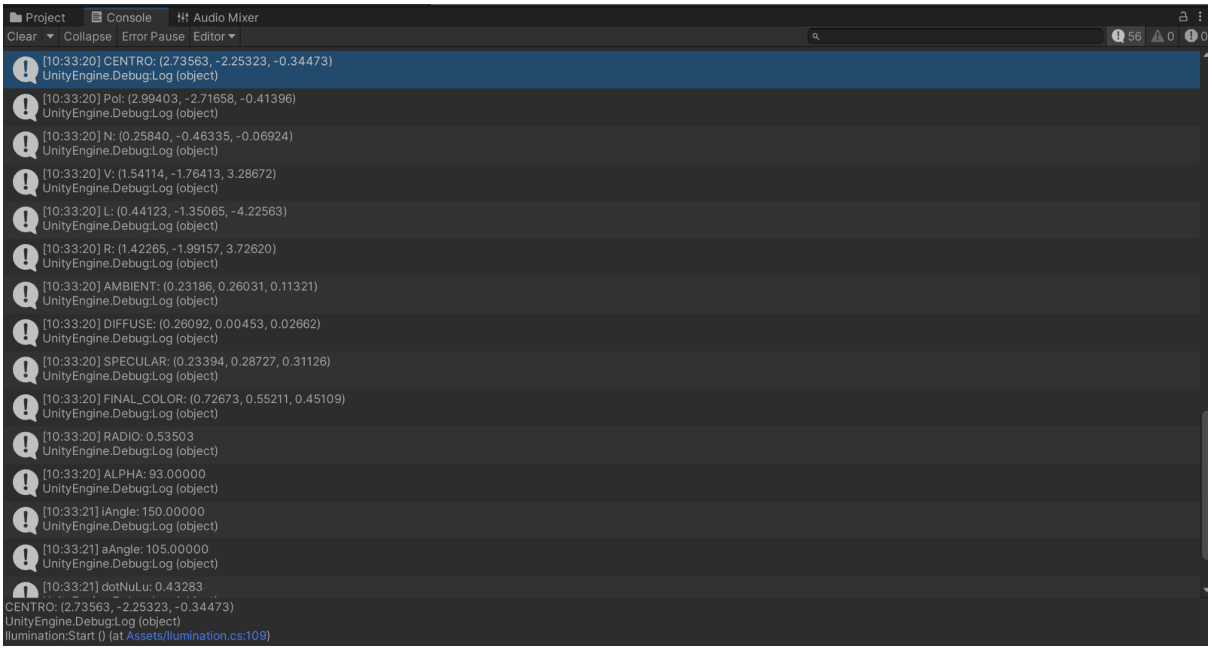
El script de Transformations nos permite trasladar el punto A respecto a TM.

Se realiza la rotación del punto dado A’ respecto a P (pivot). Para esto se definen las 3 matrices necesarias y se multiplican por el vector temporal.
Obtenemos los ángulos de x, y y z encontramos PoI.

Se realizan los cálculos para los 3 componentes de RGB: difusa, ambiental y especular.

n(Pol-Cntro) v(Cam-Pol) l(Luz-Pol).

A continuación se muestran los resultados encontrados del centro y PoI:



Además del resultado del color final, bastante similar al resultado dado por el profesor:
Resultados

A01650668 ANNYA_PAULINA	Ir	Ig	Ib
	0.74494	0.57454	0.47540