1. **Monitoring scene**
2. Motion Simulation:

if (State == 1)

{

if (HuBangBanRotAngle > HuBangBanShouAngle)

{

HuBangBanFunction(HuBangBanRotAngle);

HuBangBanRotAngle -= 1;

HBBRotangle = XiaoZhou\_DingLiang\_HuBangBan.transform.eulerAngles.z - 360;

}

}

1. Named Variables:

public Text x\_zuobiao;

public Text y\_zuobiao;

public Text z\_zuobiao;

public Text pianhang;

public Text fuyang;

public Text henggun;

public Text houliangan;

public Text yanhuliang;

public Text dingliang;

public GameObject dz;

public GameObject hlg;

public GameObject yhl;

public GameObject dl;

1. Variable Assignment:

x\_zuobiao.text = xPos.ToString("F2");

y\_zuobiao.text = yPos.ToString("F2");

z\_zuobiao.text = zPos.ToString("F2");

pianhang.text = PHjiao.ToString("F2");

fuyang.text = FYjiao.ToString("F2");

henggun.text = HGjiao.ToString("F2");

houliangan.text = HLG.ToString("F2");

yanhuliang.text = YHG.ToString("F2");

dingliang.text = DL.ToString("F2");

1. Pose optimization
2. data reading:

string filePath = "data.csv"; // CSV file path (relative to "Resources" folder)

TextAsset csvFile = Resources.Load<TextAsset>(filePath);

string[] lines = csvFile.text.Split('\n');

int rowCount = lines.Length;

positions = new Vector3[rowCount];

rotations = new Vector3[rowCount];

for (int i = 0; i < rowCount; i++)

{

string[] values = lines[i].Split(',');

float x = float.Parse(values[0]);

float y = float.Parse(values[1]);

float z = float.Parse(values[2]);

positions[i] = new Vector3(x, y, z);

float rx = float.Parse(values[3]);

float ry = float.Parse(values[4]);

float rz = float.Parse(values[5]);

rotations[i] = new Vector3(rx, ry, rz);

1. data writing(MATLAB):

matrix = zeros(matrixSize); % Create a zero matrix of size matrixSize

for col = 1:matrixSize(2)

% Gets the mean and variance of the current column

meanValue = meanValues(col);

varValue = varValues(col);

% Column vectors that generate a normal distribution

columnVector = normrnd(meanValue, sqrt(varValue), matrixSize(1), 1);

% Assigns the column vector to the corresponding column of the matrix

matrix(:, col) = columnVector;

end

disp(matrix); % Displays the generated matrix

% Write the matrix to an existing CSV file (overwrite existing content)

filePath = 'C:\Users\fff\Desktop\unity\QIUJIE\Assets\Resources\data.csv.csv';

csvwrite(filePath, matrix);

1. Line Renderer:

public void dizuo1biaoxianshi()

{

//imagejinggao.SetActive(true);

dizuo1biao.SetActive(!dizuo1biao.activeSelf);

}

public void dizuo2biaoxianshi()

{

//imagejinggao.SetActive(true);

dizuo2biao.SetActive(!dizuo2biao.activeSelf);

}

1. Collision scene
2. Visualization:

public void OnButtonClick()

{

if (loopCount < positions.Length & loopCount < weizhi.Length)

{

Vector3 JZZJ\_Position = new Vector3(weizhi[loopCount].x, weizhi[loopCount].y, weizhi[loopCount].z);

Quaternion JZZJ\_Rotation = Quaternion.Eulerxuanzhuan[loopCount].y,xuanzhuan[loopCount].x, xuanzhuan[loopCount].z);

Vector3 HDZJ\_Position = new Vector3(positions[loopCount].x, positions[loopCount].y , positions[loopCount].z);

Quaternion HDZJ\_Rotation = Quaternion.Euler rotations[loopCount].y, rotations[loopCount].x, rotations[loopCount].z);

GameObject instance1 = Instantiate(prefab1, JZZJ\_Position, JZZJ\_Rotation);

GameObject instance2 = Instantiate(prefab2, HDZJ\_Position, HDZJ\_Rotation);

}

}

1. Collision solution:

if (collider1.bounds.Intersects(collider2.bounds))

{

collisionCount++;

}

float collisionProbability = (float)collisionCount / n \* 100;

Debug.Log(collisionProbability);

3.Bayesian algorithmfor Normal Distribution Data (MATLAB code)

meanC = meanA - meanB;

varC = varA + varB;

meanC1 = meanA - meanb;

varC1 = varA + varb;

varD = (varA .\* varC) ./ (n \* varA + varC);

meanD = (varC.\*meanA+n\*varA.\*meanC)./(n\*varA+varC);

varD1 = (varA .\* varC1) ./ (n \* varA + varC1);

meanD1 = (varC1.\*meanA+n\*varA.\*meanC1)./(n\*varA+varC1);

disp(meanC);

disp(meanD);

disp(meanD1);