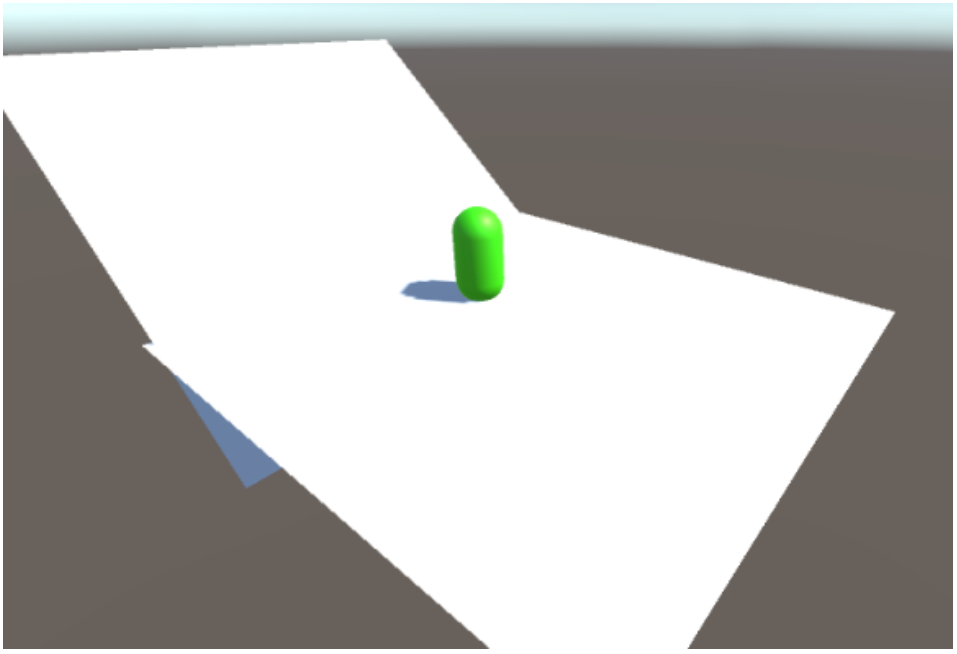
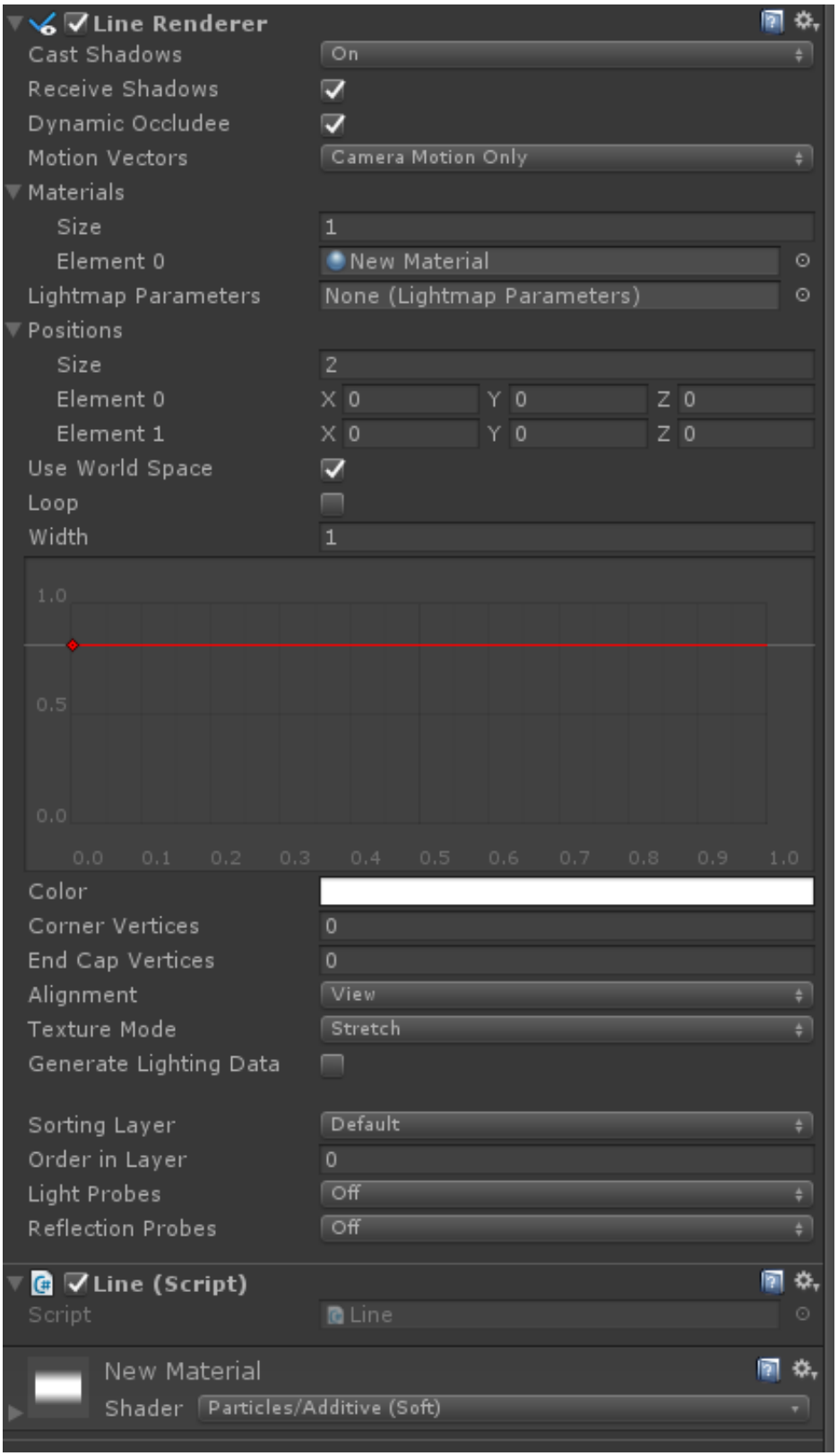
点击地面控制人物爬坡



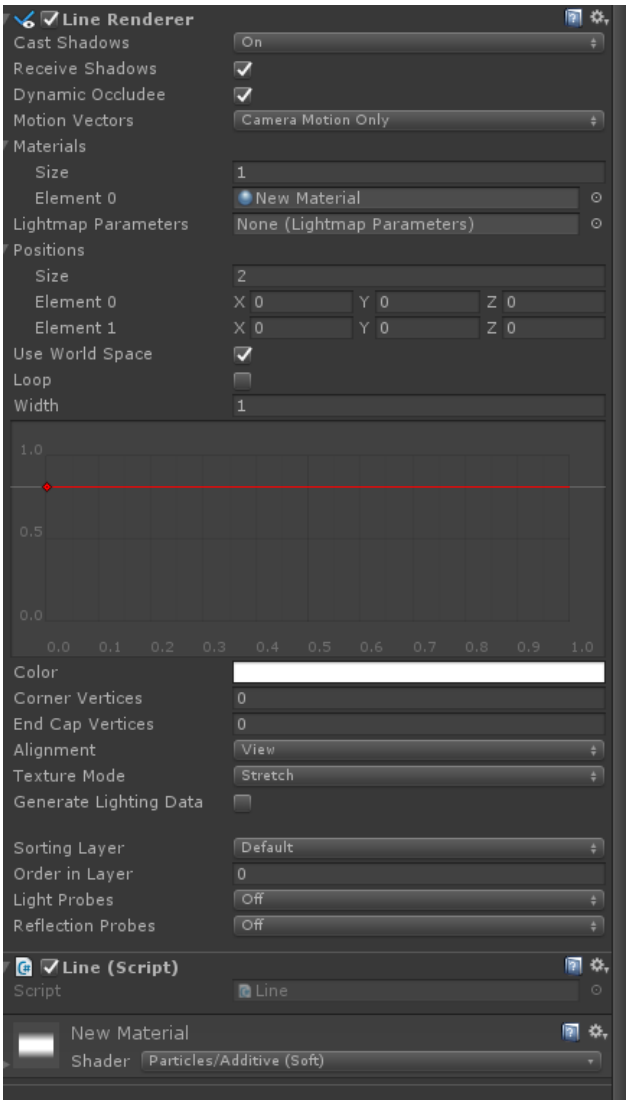



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SopleMove : MonoBehaviour {
    private Vector3 target;
```

```
    private bool move;

    void Update () {
        if (Input.GetMouseButtonDown(0))
        {
            Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
            RaycastHit hit;
            if (Physics.Raycast(ray, out hit, 300))
            {
                if (hit.collider.CompareTag("terrain"))
                {
                    target.Set(hit.point.x, transform.position.y, hit.point.z);
                    move = true;
                }
            }
        }

        if (move)
        {
            target.Set(target.x, transform.position.y, target.z);
            transform.LookAt(target);
            transform.Translate(Vector3.forward * 0.1f);

            //设置第二条射线 检测人物应该出现在的Y值
            Ray ray1 = new Ray();
            ray1.origin = transform.position+Vector3.up;
            ray1.direction = Vector3.down;

            RaycastHit hit;
            Physics.Raycast(ray1, out hit, 100, 1 << LayerMask.NameToLayer("UI"));
            if (hit.collider == null)
                return;

            transform.position = new Vector3(transform.position.x, hit.point.y, transform.position.z);
            if (Vector3.Distance(transform.position, target) < 0.1f)
                move = false;
        }
    }
```

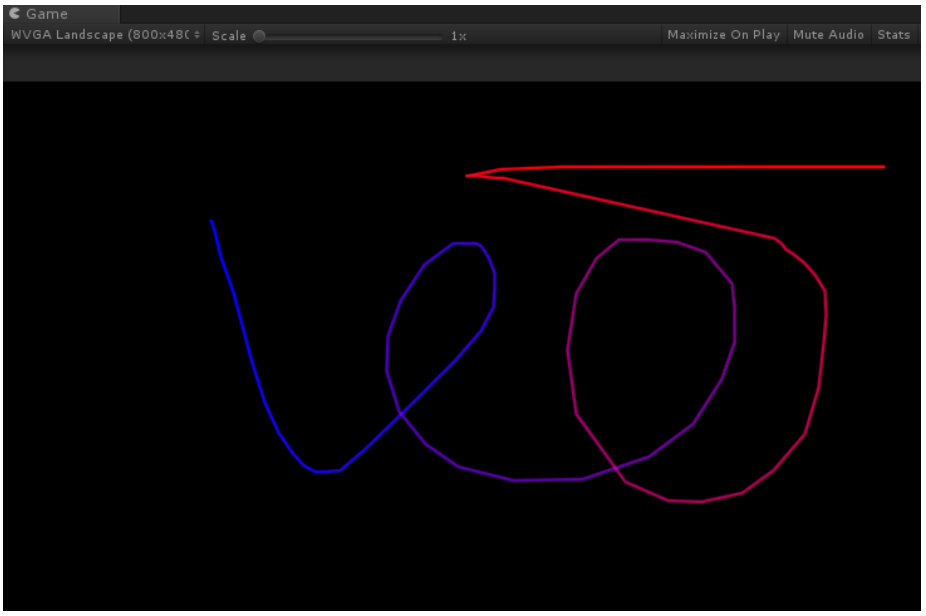
```
45 }
46
```

LineRenderer 线性渲染器



Shader选择Paticles/Additive（soft）

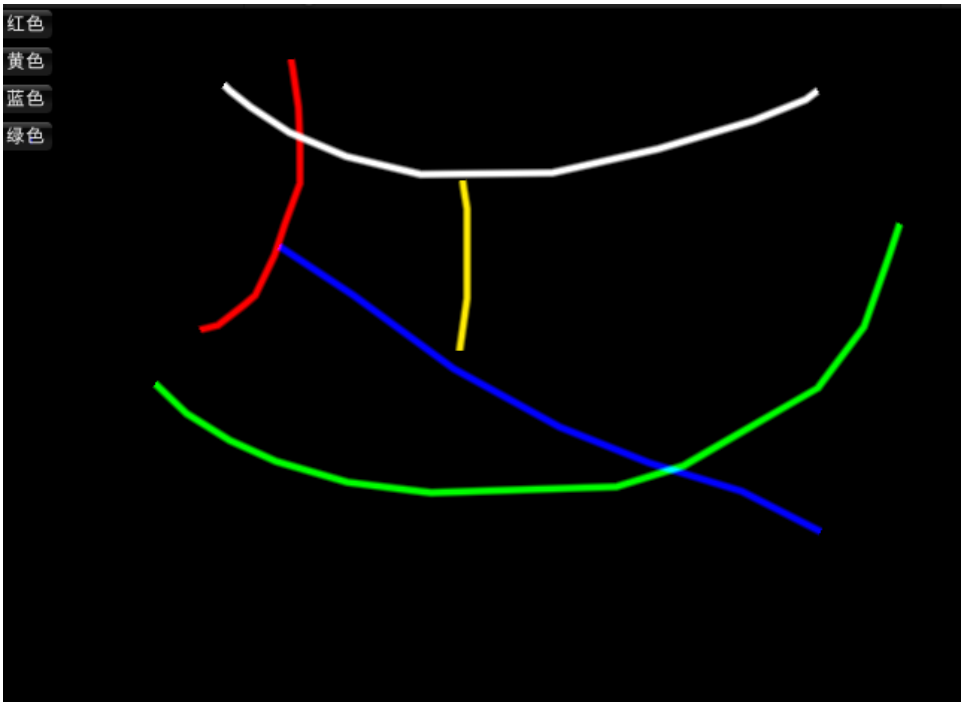
使用lineRenderer在场景中绘制线




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

public class Line : MonoBehaviour {

    private LineRenderer lr;
    private List<Vector3> points = new List<Vector3>();
    private bool beginDraw;
    void Start () {
        lr = GetComponent<LineRenderer>();
        lr.startColor = Color.blue;//设置线的起始颜色
        lr.endColor = Color.red;//设置线的结束颜色 (颜色渐变)
    }

    void Update () {
        if (Input.GetMouseButtonDown(0))
        {
            beginDraw = true;
        }
        if (Input.GetMouseButtonUp(0))
        {
            beginDraw = false;
        }

        if (beginDraw)
        {
            //保持z轴为正   这样才能保证2D物体可以被摄像机照射到和渲染
```

```
            Vector3 pos = new Vector3(Input.mousePosition.x,
Input.mousePosition.y, 1.0f);
            //把屏幕坐标转化为世界坐标
            pos = Camera.main.ScreenToWorldPoint(pos);
            points.Add(pos);
            Draw();
        }
    }

    void Draw()
    {
        //设置线渲染器的点的个数
        lr.positionCount = points.Count;
        //通过for循环把所有的点赋值给线渲染器
        for (int i = 0; i < points.Count; i++)
        {
            lr.SetPosition(i, points[i]);
        }
    }
}
```

使用预设在画面中绘制多条线



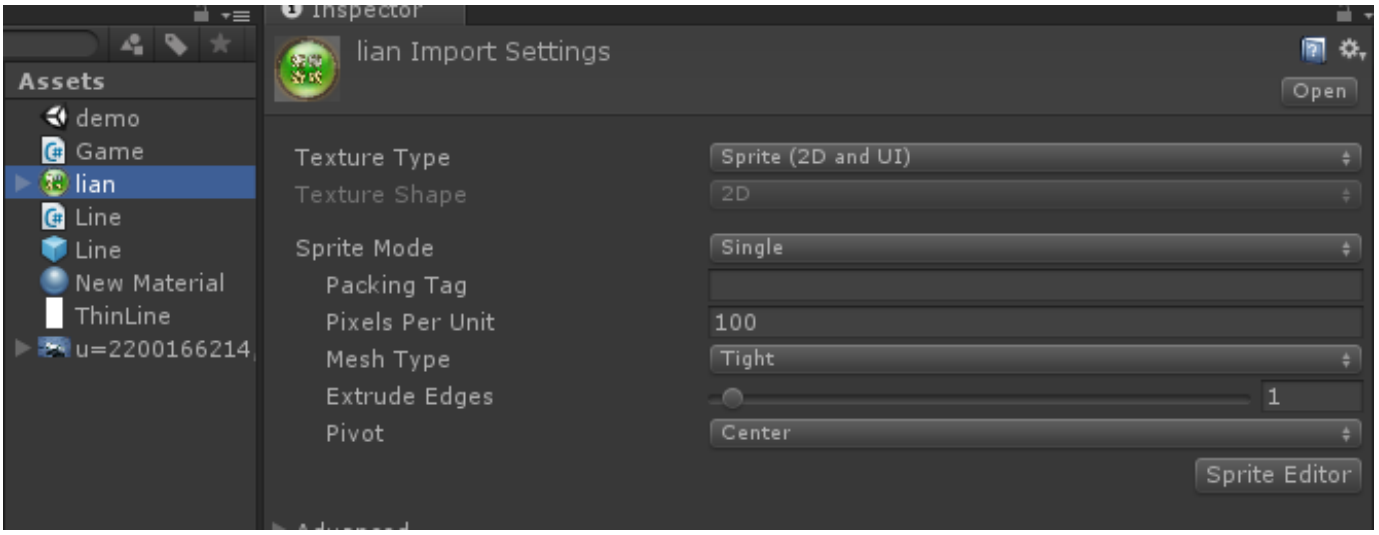

```
//绑定在场景中的空物体上
```

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

public class Game : MonoBehaviour {
    public GameObject linePrefab;
    private GameObject lineObj;
    private bool beginDraw;

    private Color colorData = Color.white;
    void Update () {
        if (Input.GetMouseButtonDown(0))
        {
            beginDraw = true;
            lineObj=Instantiate(linePrefab);
        }
        if (Input.GetMouseButtonUp(0))
        {
            beginDraw = false;
        }

        if (beginDraw)
        {
            Vector3 position = new Vector3(Input.mousePosition.x,
Input.mousePosition.y, 1.0f);
            position = Camera.main.ScreenToWorldPoint(position);

            Line line = lineObj.GetComponent<Line>();

            line.points.Add(position);
            line.lr.startColor = colorData;
            line.lr.endColor = colorData;
            line.Draw();
        }
    }

    void OnGUI()
    {
        if (GUILayout.Button("红色"))
        {
            colorData = Color.red;
        }
```

```
        if (GUILayout.Button("黄色"))
        {
            colorData = Color.yellow;
        }
        if (GUILayout.Button("蓝色"))
        {
            colorData = Color.blue;
        }
        if (GUILayout.Button("绿色"))
        {
            colorData = Color.green;
        }
    }
}

```

```
//绑定在线渲染器预制体上
public class Line : MonoBehaviour
{
    public LineRenderer lr;
    public List<Vector3> points = new List<Vector3>();

    void Awake()
    {
        lr = GetComponent<LineRenderer>();
    }

    public void Draw()
    {
        lr.positionCount = points.Count;

        for (int i = 0; i < points.Count; i++)
            lr.SetPosition(i, points[i]);
    }
}
```

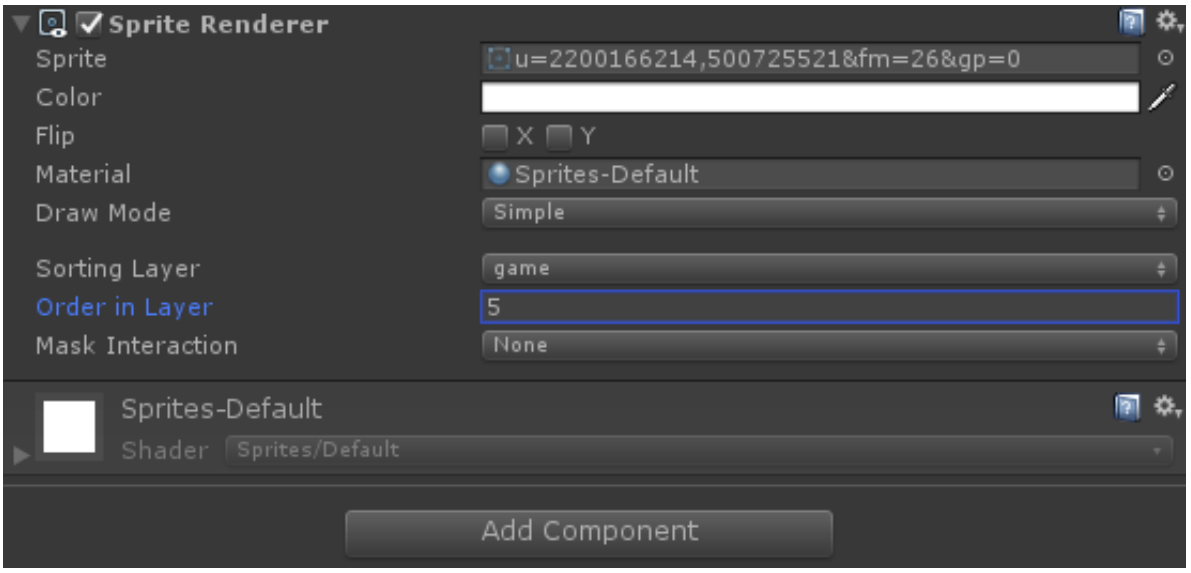
Native2D 系统



所有的图片类型 选择Sprite类型

SpriteRenderer 精灵渲染器

当图片添加到场景中 自动添加组件



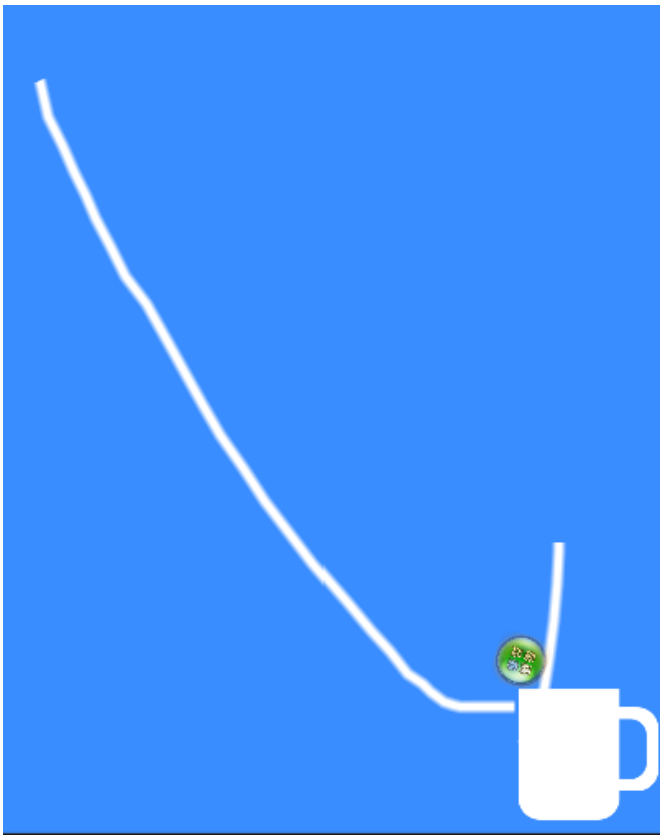
SortingLayer 排序层

默认Default的层级是最低的 由上往下依次递增

层级高的图片覆盖层级低的图片

如果两个图片在同一个Layer下 显示顺序由OrderInLayer决定

使用代码给线添加边缘碰撞器



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

public class Line : MonoBehaviour
{
    public LineRenderer lr;
    public List<Vector3> points = new List<Vector3>();

    private EdgeCollider2D edge;

    void Awake()
    {
        lr = GetComponent<LineRenderer>();
        edge = GetComponent<EdgeCollider2D>();
    }

    public void Draw()
    {
        List<Vector2> edgePoints=new List<Vector2>();
        lr.positionCount = points.Count;

        for (int i = 0; i < points.Count; i++)
```

```
        {
            lr.SetPosition(i, points[i]);
            edgePoints.Add(new Vector2(points[i].x,points[i].y));
        }

        if(points.Count>1)
            edge.points = edgePoints.ToArray();
    }
}
```