北京林业大学 2023--2024 学年第 2 学期实验任务书

课程名称：三维动画原理与制作　　开课学院：信息学院

考试班级：数媒 21-1、2　　命题人：杨　猛

实验环境：Unity 3D 2019 或以上　实验学时：2

实验题目(范围)：实验 2　图表关键帧动画

请详细说明该设计的方案、内容、要求、进度等

严禁剽窃、抄袭等作弊行为！

实验目的：

1. 掌握 Unity 的 Build-In 实验环境。
2. 实际应用插值动画的思想。
3. 熟悉 Unity 中常见的插值计算。
4. 通过动画进行函数的可视化

实验内容：

1. Unity 3D 的动画编程方法，主要包括：
   （1） 熟悉 Unity 3D，创建新项目的方法。
   （2） 熟悉 Unity 3D 脚本编辑方法。
   （3） 重点熟练掌握 Unity 3D 的动画编程方法。
2. 依照“**实验方法**”中步骤依次进行实验操作，并给所添加/修改的每一句代码添加注释。

实验环境要求：

1. 实验环境：建议使用 Unity 3D 2019 或更高版本，可在 https://unity.cn/releases 路径下载。

实验方法：（注：以下演示步骤均在 Unity2019.4.17f1c1 环境中完成）

1. 建立 Unity 项目，初始化实验环境

打开 UnityHub，选择，新建项目，新项目->3D 核心模板->创建项目 （图 1）。

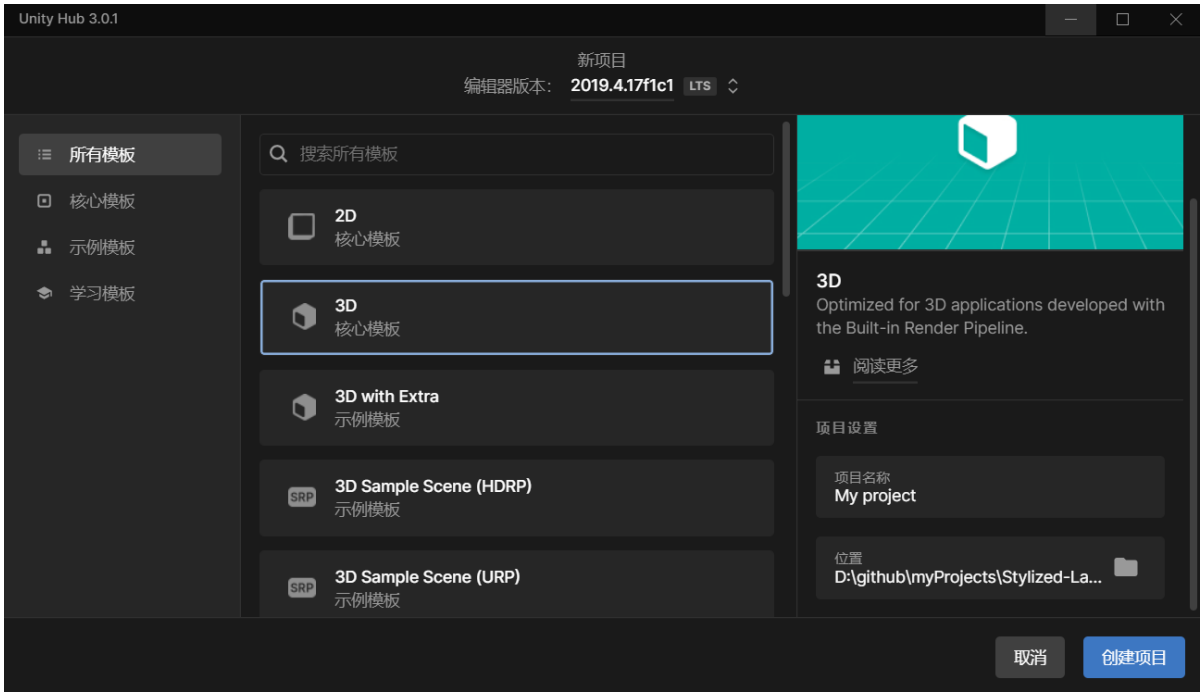

图 1 新建项目菜单

将 Assets 中的 Scenes 文件夹删除（如图 2 所示）。

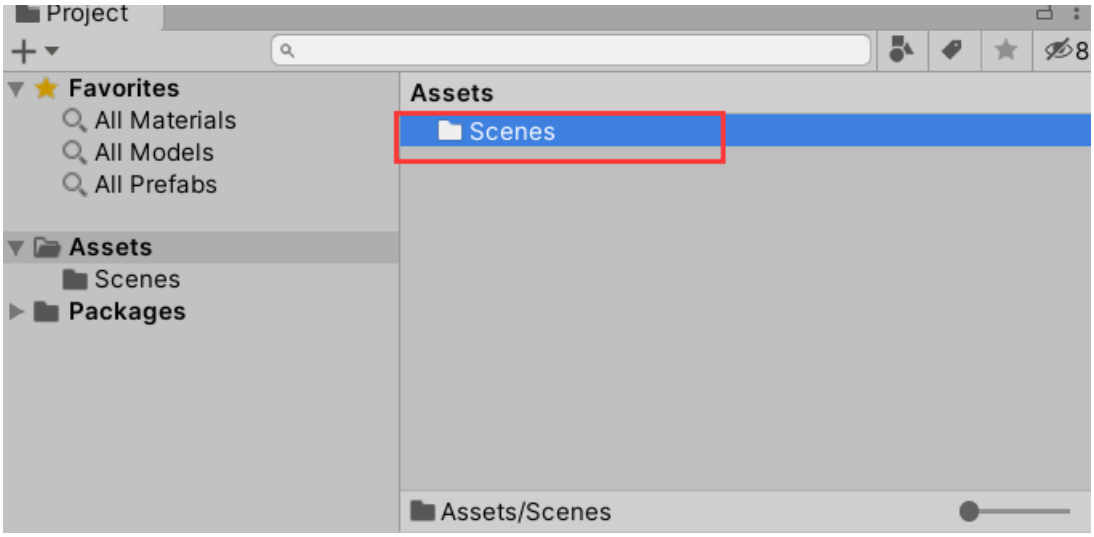

图 2 删除 Scene 文件夹

右键 Create->Scene 新建一个场景，命名为实验 1，然后双击打开（如图 3）。

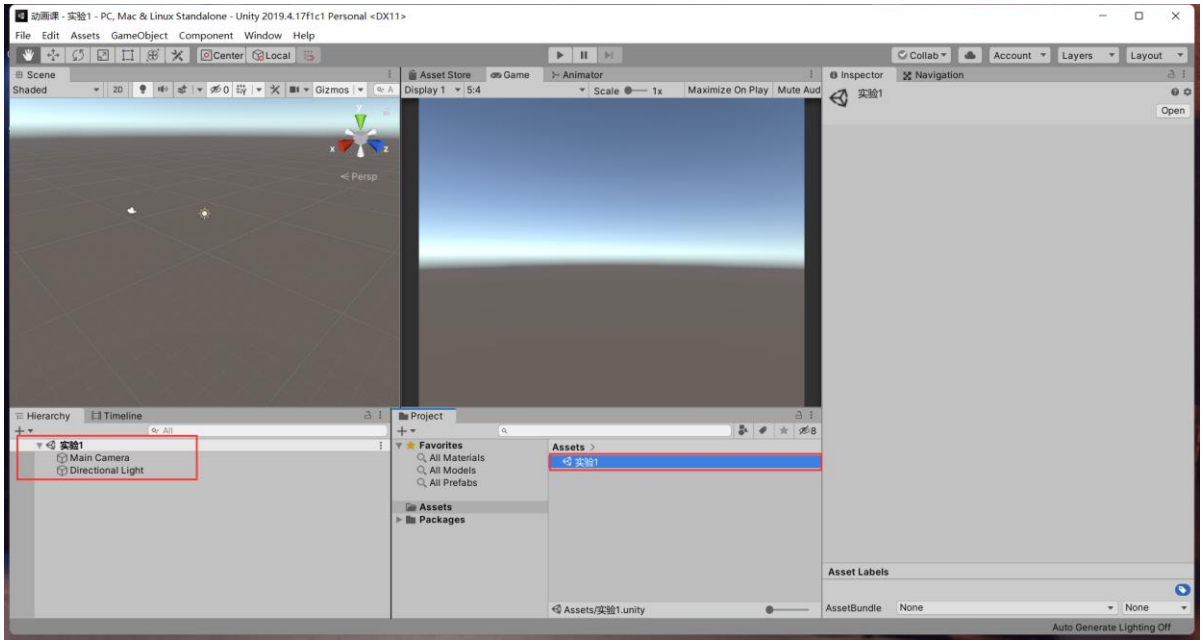

图 3 打开新的场景

新建一个在场景中新建一个立方体，并将其转化为预制件，然后将场景中的立方体删除

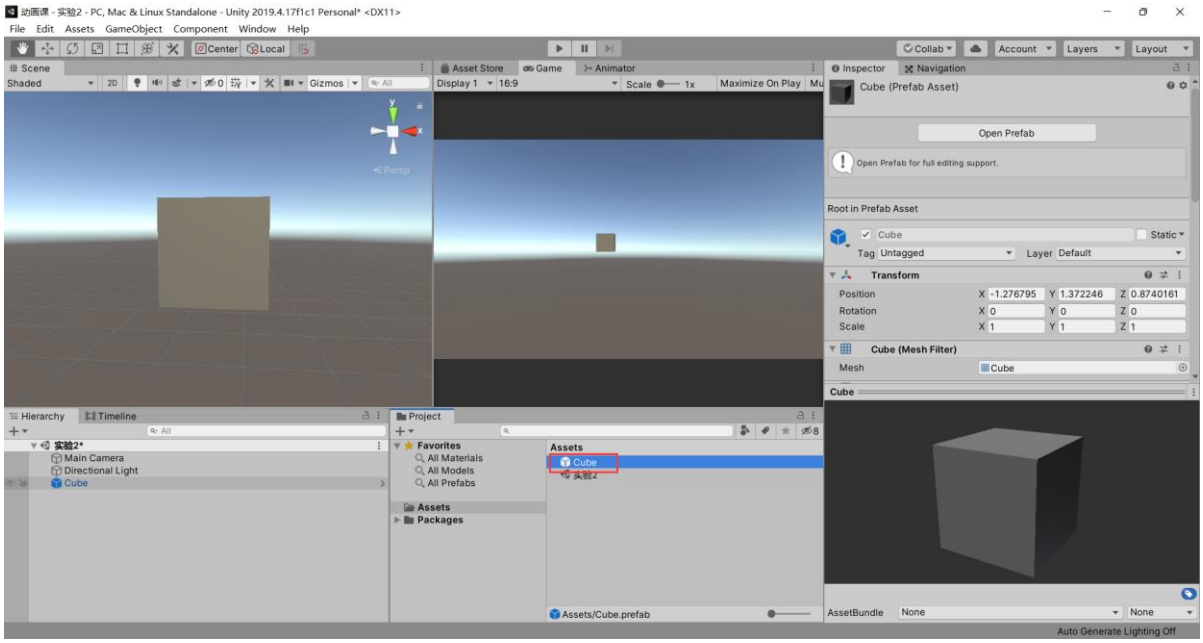

图 4 新建预制件

2. 新建动画脚本

如图 5 所示，鼠标右键 Create->C# Script,新建一个 C#脚本，命名为 Graph

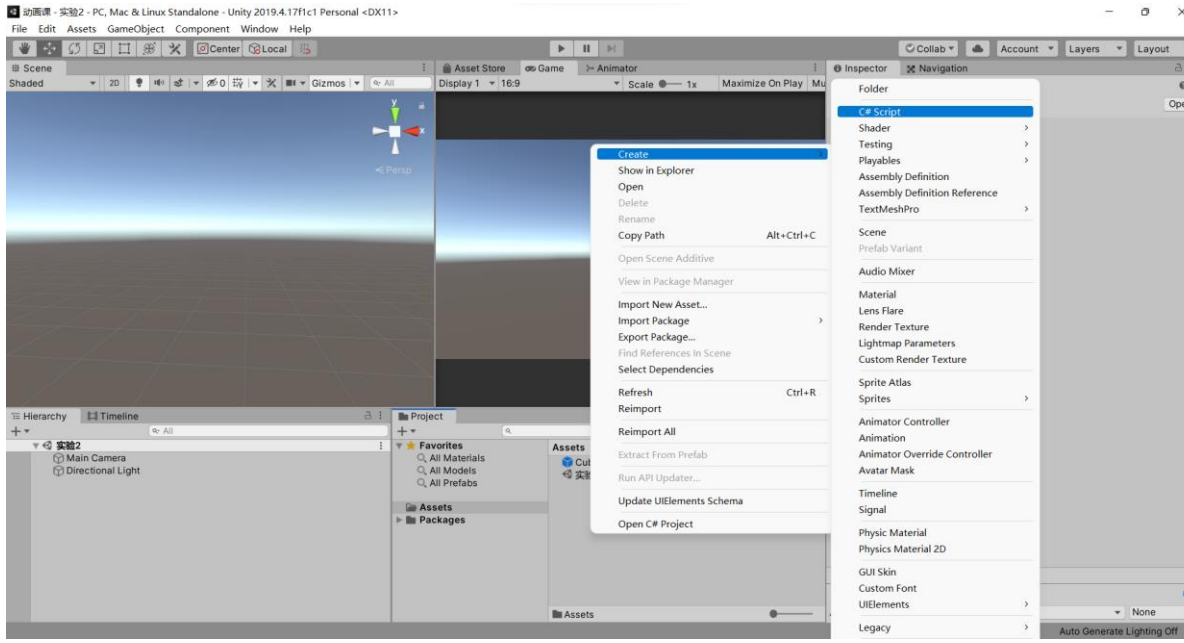

图 5 新建动画脚本

声明一个枚举变量 Function，用于计算函数值

```
public enum Function
{
    Wave
}

[SerializeField] private Function func = Function.Wave;  Unchanged
```

图 6 添加枚举变量

新声明三个个变量，用于后续实现图表，pointPrefab 用于图表的点，resolution 表示图表的精度，points 表示图表上离散点组成的数组

```
[SerializeField] Transform pointPrefab;  Unchanged

[SerializeField, Range(10, 50)] int resolution = 10;  Unchanged

Transform[] points;
```

图 7 声明图表所需变量

在 Void Start()函数中进行图表的初始化，理解 step 和 scale 的关系，将点的位置控制在（-1，1）之间，另外理解 point 的本地坐标以及与 transform 之间的父子关系。

```
void Start () {
    float step = 2f / resolution;
    var scale :Vector3 = Vector3.one * step;
    points = new Transform[resolution];
    for (int i = 0; i < points.Length; i++) {
        Transform point = points[i] = Instantiate(pointPrefab);
        point.localScale = scale;
        point.SetParent(transform, worldPositionStays: false);
    }
}
```

图 8 初始化图表数组

声明函数 float GetFunction（float x， float t）用来获取在 x 处，t 时间图表的 y 值，**下图为示例函数，在提交实验报告时需要添加一到两个新的函数**，产生新的图表。

```
float GetFunction(float x, float t)
{
    switch(func)
    {
        case Function.Wave:
            float d = Mathf.Abs(x);
            float y = Mathf.Sin(f:Mathf.PI * (4f * d - t));
            return y / (1f + 10f * d);
        default:
            return 0;
    }
}
```

图 9 获取函数值

最后，在 void Update（）函数中进行图表的更新，update 函数会在每一帧调用一次，可以在这个函数中进行图表的动画实现，理解在 Update 函数中更新 y 值的意义。

```
void Update () {
    float time = Time.time;
    float step = 2f / resolution;
    float v = 0.5f * step - 1f;
    for (int i = 0; i < points.Length; i++) {

        float x = (i + 0.5f) * step - 1f;
        points[i].localPosition = new Vector3(x, y: GetFunction(x, t: time), z: 0);
    }
}
```

图 10 在 Update 函数中进行图表的更新

3. 实现图表动画

在场景中新建空物体并命名为 Graph，将 Graph 脚本添加到这个物体上，并将一开始生成的预制体添加到脚本上。

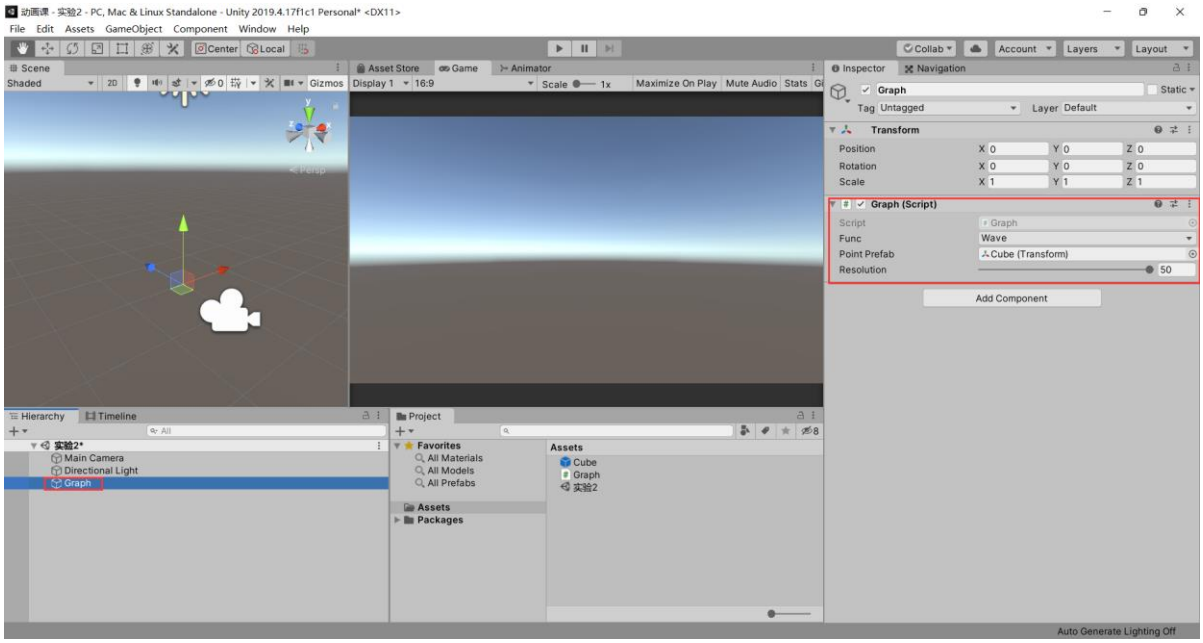

图 11 新建 Graph 对象

运行 Unity，可以看到下图的效果

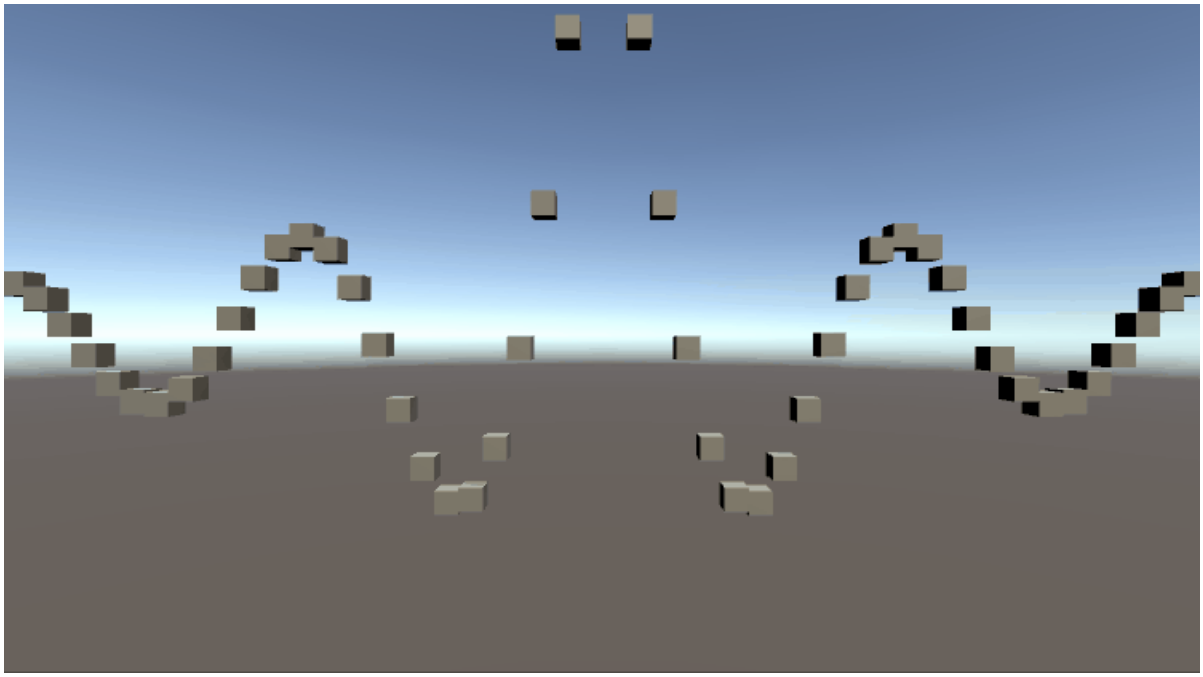

图 12 图表动画

4. 丰富显示效果

在 Assets 中，鼠标右键 Create->Shader->UnlitShader,新建一个着色器。

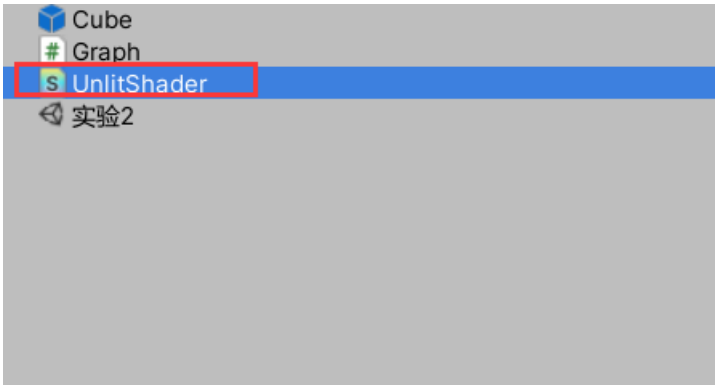

图 13 新建着色器

打开该着色器，将 Properties 中的变量删除。

```
Properties
{
    //_MainTex ("Texture", 2D) = "white" {}
}
```

图 14 删除不需要变量

然后再把所有的相关变量按照下图注释删除，这些变量在本次实验中不需要使用，只需要在片元着色器中进行简单的代码即可。

```
struct appdata
{
    float4 vertex : POSITION;
    //float2 uv : TEXCOORD0;
};

struct v2f
{
    float2 uv : TEXCOORD0;
    //UNITY_FOG_COORDS(1)
    float4 vertex : SV_POSITION;
};

//sampler2D _MainTex;
//float4 _MainTex_ST;

v2f vert (appdata v)
{
    v2f o;
    o.vertex = UnityObjectToClipPos(v.vertex);
    //o.uv = TRANSFORM_TEX(v.uv, _MainTex);
    //UNITY_TRANSFER_FOG(o,o.vertex);
    return o;
}
```

图 15 删除不需要的变量

将 appdata 结构体中变量如下图所示更改，vertex 是模型坐标，worldPos 是世界坐标。

```
struct appdata
{
    float4 vertex : POSITION0;
    float4 worldPos : POSITION1;
};
```

图 16 appdata 变量

将传入片元着色器的 v2f 结构体中的变量如下图所示修改，vertex 是裁剪空间的坐标，worldPos 是世界坐标。

```
struct v2f
{
    float4 vertex : SV_POSITION;
    float4 worldPos : POSITION1;
};
```

图 17 v2f 变量

简单更改顶点和片元着色器，让正方体的颜色根据世界坐标改变，理解世界坐标，模型坐标，裁剪空间的坐标的区别。

```
v2f vert (appdata v)
{
    v2f o;
    o.worldPos = mul(unity_ObjectToWorld,v.vertex);
    o.vertex = UnityObjectToClipPos(v.vertex);
    return o;
}

fixed4 frag (v2f i) : SV_Target
{
    fixed4 col = fixed4(i.worldPos.xyz, 1.0)*0.5+0.5;

    return col;
}
```

图 18 顶点着色器和片元着色器

最终的结果应该如图 21 所示：

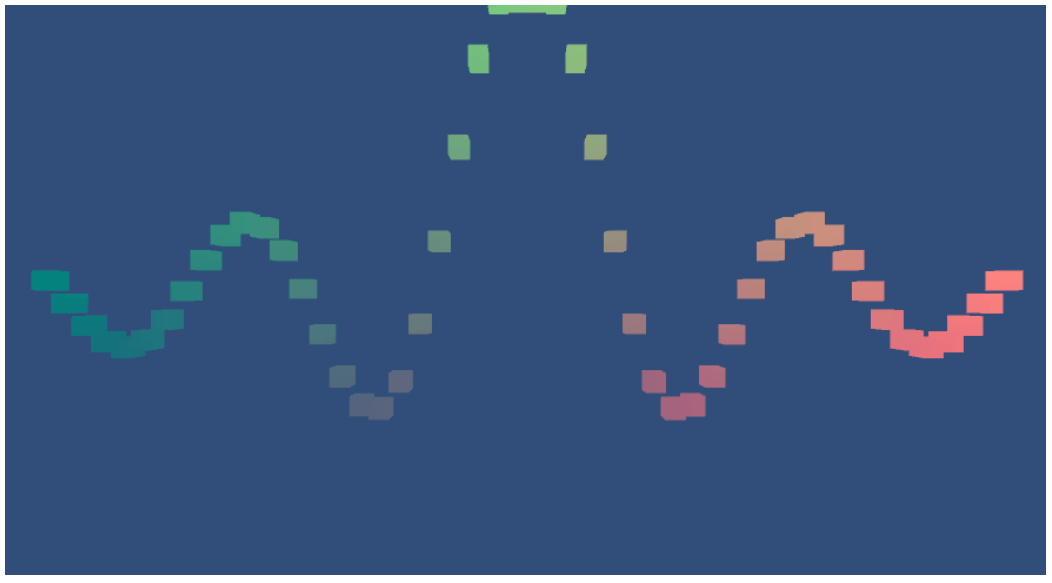

图 19 最终结果（gif 动图）

结论分析：

列举实验中遇到的问题、解决的方法，总结实验的收获和体会以及尚存在的问题。

实验要求：

1. 确认机器已经安装 Unity2019（或更高版本）；参见“**实验环境要求**”。
2. 根据“**实验方法**”中步骤依次实践计算机动画编程方法，并达到**熟练掌握**程度。
3. 给实验中添加/修改的代码逐句添加注释，可参考“参考书目”中教材或网络上资料。
4. 提交材料：本次实验(实验 2)需要填写实验报告并且提交*.Unitypackage文件。实验报告内容包括实验步骤、结果图、注释后的代码以及“**结论分析**”中所述内容等。实验报告必须是在 **Microsoft Office Word** 中进行格式排版后的报告。
5. 命名规则：学号_姓名，注意中间为“下划线”，**未按规则命名者按未交作业处理**。
6. 本课程**所有实验的报告**要求集成在一个文档中，并在课程结束后规定时间内提交即可，即课程结束前不必分开提交“实验 1”、“实验 2”等实验材料。
7. 提交地址：ftp://211.71.149.149/yang_meng/homework/相应目录下。

参考书目：

- 计算机动画算法与编程基础，雍俊海 著，2008.7 ，清华大学出版社。

教研室主任意见：

签字：_____ 年 月 日

学院负责人意见：

签字：_____ 年 月 日

注：此表一式两份，一份于考前交到考试中心，一份随学生课程设计材料上交学院。