Project 4 Motion Path Editing 技術文件

(一)、 讀取 BVH

開啟資料夾用的是套件

https://github.com/gkngkc/UnityStandaloneFileBrowser，選擇一個 BHV 檔案之後開時讀取。

一開始會先讀到 HIERARCHY，接下來就根據讀到的字串決定是什麼(ROOT、JOINT、OFFSET、CHANNELS…等)

整個 BVH 會由 BVH.cs 和 BVHJoint.cs 儲存結構。

```
string line = reader.ReadLine();
string[] inputs = line.Split(' ');
string jointName = "";
if (inputs[0] == "HIERARCHY")
{
    BVH bvh = Instantiate(bvhPrefab, Vector3.zero, Quaternion.identity);
    // 用來儲存遞迴結構
    List<string> jointNames = new List<string>();
    while (inputs[0] != "MOTION")
    {
        line = reader.ReadLine();
        line = line.Trim();
        line = Regex.Replace(line, @"\s+", " ");
        inputs = line.Split(' ');
        if (inputs[0] == "ROOT")
        {
            jointName = line.Split(' ')[1];
            bvh.AddRoot(inputs[1]);
            yield return null;
        }
```

```
else if (inputs[0] == "{")
    jointNames.Add(jointName);
// 偏移量
else if (inputs[0] == "OFFSET")
{
    for (int i = 1; i < inputs.Length; i++)
    {
        if (!IsFloat(inputs[i]))
        {
            Debug.Log("BVH ERROR!");
            yield break;
        }
    }
    Vector3 offset = new Vector3(Convert.ToSingle(inputs[1]), Convert.ToSingle(inputs[2]), Convert.ToSingle(inputs[3]));
    bvh.SetJointOffset(jointNames[jointNames.Count - 1], offset);
}
// 頻道
else if (inputs[0] == "CHANNELS")
{
    List<string> channels = new List<string>();
    for (int i = 0; i < Convert.ToInt32(inputs[1]); i++)
    {
        channels.Add(inputs[i + 2]);
        bvh.motionString.Add(jointName + " " + inputs[i + 2]);
    }
    bvh.SetJointChannels(jointNames[jointNames.Count - 1], channels);
}
```

```
// 新的關節
else if (inputs[0] == "JOINT")
{
    // 堆疊最上層為父關節
    jointName = inputs[1];
    bvh.AddJoint(inputs[1], jointNames[jointNames.Count - 1]);
    yield return null;
}
else if (inputs[0] == "End")
{
    jointName = jointNames[jointNames.Count - 1] + " End Site";
    bvh.AddJoint(jointName, jointNames[jointNames.Count - 1]);
}
// 右括號，減少一層
else if (inputs[0] == "}")
    jointNames.RemoveAt(jointNames.Count - 1);
else if (inputs[0] == "MOTION")
    ;
else
{
    Debug.Log("BVH ERROR! " + inputs[0]);
    yield break;
}
```

jointNames 用來儲存的 joint 的遞迴結構，然後就看讀到什麼就儲存資訊。

(二)、 BVH 動畫

依照 frame rate 去抓出 BVH 的 motion，再實際賦值給物件。但因為有用插植，所以 frame rate 會分更細。

```
public void UpdateToFrame(int frameNumber, float time)
{
    if (frameNumber >= frames.Count)
        return;
    Dictionary<int, float> frameData = frames[frameNumber];
    Vector3 next = GetRotation(frameNumber, frameData).eulerAngles;
    Vector3 position = GetPosition(frameNumber, frameData);
    Vector3 interpolated = next;
    // 線性插值
    if (frameNumber > 0)
    {
        Vector3 now = GetRotation(frameNumber - 1, frameData).eulerAn
        interpolated = now + time * (next - now);
    }
    transform.localPosition = position;
    transform.localRotation = Quaternion.Euler(interpolated);
}
```

每個 BVHJoint 會呼叫這個 function 來更新位置和旋轉。

```
public Vector3 GetPosition(int frameNumber, Dictionary<int, float> frameData)
{
    try
    {
        Vector3 position = transform.localPosition;
        foreach (KeyValuePair<int, float> pair in frameData)
        {
            if (pair.Key < 0)
                continue;
            string channel = channels[pair.Key];
            if (channel == "Xposition")
                position.x = pair.Value;
            else if (channel == "Yposition")
                position.y = pair.Value;
            else if (channel == "Zposition")
                position.z = pair.Value;
        }
        return position;
    }
```

GetPosition 依照 channel 順序抓出位置。

```
public Quaternion GetRotation(int frameNumber, Dictionary<int, float> frameData)
{
    Quaternion rotation = Quaternion.Euler(0, 0, 0);

    for (int i = 0; i < channels.Count; i++)
    {
        string channel = channels[i];
        float value = frameData[i];
        if (i == 0)
        {
            if (channel == "Zrotation")
                rotation = Quaternion.Euler(0, 0, value);
            else if (channel == "Xrotation")
                rotation = Quaternion.Euler(value, 0, 0);
            else if (channel == "Yrotation")
                rotation = Quaternion.Euler(0, value, 0);
        }
        else
        {
            if (channel == "Zrotation")
                rotation *= Quaternion.Euler(0, 0, value);
            else if (channel == "Xrotation")
                rotation *= Quaternion.Euler(value, 0, 0);
            else if (channel == "Yrotation")
                rotation *= Quaternion.Euler(0, value, 0);
        }
    }
    return rotation;
}
```

GetRotation 依照 BVH 中 Motion 的資料抓出旋轉。

(三)、 人物模型

BVH 每個關節都會對應到人物的關節，用這個對應關係來移動人物。

```
case "LeftUpLeg":
case "LeftHip":
    temp = HumanBodyBones.RightUpperLeg;
    break;
case "LeftLowLeg":
case "LeftKnee":
    temp = HumanBodyBones.RightLowerLeg;
    break;
case "LeftFoot":
case "LeftAnkle":
    temp = HumanBodyBones.RightFoot;
    break;
```

```
public void UpdateMotionPos(GameObject people)
{
    int CurrentIndex = 0;
    BVH bvhPeople = people.GetComponent<BVH>();
    BVHJoint[] joints = bvhPeople.joints.ToArray();

    for(int i = 0; i < joints.Length; i++)
    {
        Transform TempBonesTran = SearchHumanBoneTransformByName(joints[i].name);
        if (TempBonesTran == null)
        {
            Debug.Log("No match: " + joints[i].name);
            continue;
        }

        // 把 Motion 套上去
        Quaternion org = new Quaternion(MotionPos[CurrentIndex * 4],
            MotionPos[CurrentIndex * 4 + 1],
            MotionPos[CurrentIndex * 4 + 2],
            MotionPos[CurrentIndex * 4 + 3]);

        Joints[CurrentIndex++].transform.rotation = joints[i].transform.rotation * org;
    }
```

MotionPos 就是來自 BVH 的資料，從裡面抓出旋轉後實際給人物就可以讓人物作出 BVH 的動作。

(四)、 連接 BVH

從第一個動作的最後一個 frame 和要連接的動作的前幾個 frame 去找出最接近的，然後對這兩個 frame 做插值，接著將插值出來的 frame 和第二個動作的 frame 接到第一個動作後面。

(五)、 Fit 曲線

曲線繪製的方式是使用 Unity 的 LineRenderer，將 Hip 的座標輸入後，透過算法 Fit 成 BezierCurve。

```
public void calculateBezierCurve(Vector3 startPnt, Vector3 endPnt, Vector3 subPnt1, Vector3 subPnt2)
{
    Vector3 p0 = startPnt; //頭
    Vector3 p1 = subPnt1; //Control point
    Vector3 p2 = subPnt2; //Control point
    Vector3 p3 = endPnt; //尾
    FEPoint[0] = startPnt;
    FEPoint[1] = endPnt;

    List<Vector3> pnts = new List<Vector3>();
    float omt = 1f;
    for (int i = segmentPntCount; i > 0; i--)
    {
        omt = i / ((float)segmentPntCount);
        Vector3 currentPosition = p0 * Mathf.Pow(omt, 3) + p1 * (3 * omt * omt * (1f - omt)) + p2 * (3 * omt * (1f - omt) * (1f - omt)) + p3 * Mathf.Pow((1f - omt), 3);
        pnts.Add(currentPosition);
    }

    segmentRenderer.positionCount = pnts.Count;
    segmentRenderer.SetPositions(pnts.ToArray());
}
```

之後產出控制點。

```
public static Vector3[] GetBezierFitCurve(Vector3[] d, double error)
{
    Vector3 tHat1, tHat2;    /*  Unit tangent vectors at endpoints */

    tHat1 = ComputeLeftTangent(d, 0);
    tHat2 = ComputeRightTangent(d, d.Length - 1);
    double[] u = ChordLengthParameterize(d, 0, d.Length - 1); ;     /*  Parameter values for point  */
    return GenerateBezier(d, 0, d.Length - 1, u, tHat1, tHat2);
}
```

```
public void SetBezierFitPath(List<Vector3> pnts)
{
    // May need to reset these lists
    //controllPntObs = new List<GameObject>();
    DestroyObInList(controllPntObs);
    DestroyObInList(subControllPntObs);
    controllPnts.Clear();
    for (int i = 0; i < segments.Count; i++)
    {
        segments[i].Destroy();
        segments[i] = null;
    }
    segments.Clear();
    //subControllPntObs = new List<GameObject>();
    //controllPnts = new List<Vector3>();
    //segments = new List<LineSegment>();
    Vector3[] result = FitCurves.GetBezierFitCurve(pnts.ToArray(), bezierErrorPar);
    for (int i = 0; i < result.Length; i++)
    {
        result[i] = new Vector3(result[i].x, 0, result[i].z);
    }
    controllPntObs.Add(newControllPntOb(result[0]));
    subControllPntObs.Add(newSubControllPntOb(result[1], 0, controllPntObs[controllPntObs.Count - 1].transform));
    controllPntObs.Add(newControllPntOb(result[3]));
    subControllPntObs.Add(newSubControllPntOb(result[2], 0, controllPntObs[controllPntObs.Count - 1].transform));
    addSegment(pnts.Count, pnts, result[1], result[2]);
}
```

控制點部分，透過 PosController 控制，它會在控制點被點擊後，產出一個 XYZ 箭頭 Object，當按住滑鼠時，會計算要把控制點拖到 3D 中的哪個位置。

```
void Update()
{
    if (!isMoving)
    {
        // Not moving any object.
        if (Input.GetMouseButtonDown(0))
        {
            RaycastHit hit;
            Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
            if (Physics.Raycast(ray, out hit))
            {
                Transform objectHit = hit.transform;
                if (objectHit.CompareTag("ControllPnt"))
                {
                    Debug.Log(objectHit.gameObject.name);
                    GameObject thePivot = Instantiate(pivot, Vector3.zero, Quaternion.identity);
                    thePivot.transform.SetParent(objectHit);
                    isMoving = true;
                }
            }
        }
    }
```