大规模人群的轻量化处理技术在web会议中的应用

摘要

关键字

1.引言

2.相关工作

实例化渲染技术：THREEJS中提供的实例化渲染技术虽然在渲染效率上比较可观，但是不能够实现丰富的多样性，也没有处理骨骼动画方面的问题。

3.方法概括

轻量化处理的核心是资源重用，我们使用该技术必须首先明白哪些资源是可重用的，并且尽可以提高资源的重用度。另外为了效果要保留丰富的多样性，参数化调整每个对象，另外可以通过对不同资源进行搭配组合来提高多样性。

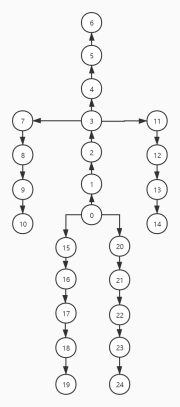
首先将模型划分为多个区域（在这个问题中是划分了头部、上身、下身三个区域），每个区域可以匹配不同贴图（在这个问题中使用了男性32套贴图，女性16套贴图），这样就可以通过不同贴图搭配各种效果（男性有32^3=32768种组合，女性有16^3=4086种组合，共36864种组合）。另外，可以通过对人物对象的高矮胖瘦进行设置，还可以对对象的色调进行编辑（这个问题中主要是对下身的裤子颜色进行设置），在骨骼动画方面，可以为每个人物对象设置不同的动画播放速度。

4.大规模人群的轻量化处理技术

4.1场景分析

大规模人群渲染需要存储的信息可以分为三级。第一级，所有对象都共用的数据，如模型的网格信息（网格点位置，UV等信息）；第二级，选择性使用的数据（如模型贴图），这类数据每个对象只需要其中的一部分数据（如模型贴图有多套可供选择，但每个对象只使用其中的一套）；第二级，描述对象的参数信息（如，贴图类型，动画播放速度，高矮胖瘦，色调），这些信息每个对象都可以不同，所以每个对象的这些参数信息都要单独存放。

4.2数据预处理

在项目中我们只需要实现听众鼓掌的动作，这个动作只涉及到手臂的8个骨骼，其它的17个骨骼的状态没有发生变化，所以首先可以将骨骼数据分成两部分，手臂处骨骼和固定动作的骨骼。

每个骨骼的变换矩阵因为最后一行是固定值，所以一个变换矩阵只需要存储12个float数据。有17个骨骼在动画播放的过程中的状态没有发生变化，所以这可以提前计算好这17\*12=204个数据。涉及到手臂的8个骨骼，由于我们在项目中只需要实现鼓掌这个简单的动作，经过测试使用16帧就可以获得比较好的动画效果，这是16帧的鼓掌动画中后8帧可以看作前8帧的倒放，所以我们只需要8个骨骼8帧分别的变换矩阵，这8个骨骼需要的数据量为8\*8\*12=768。将这些数据在预处理阶段直接计算好可以减少在客户端的计算量。

4.3 轻量化信息传输

将场景中需要的数据细化拆分，相同部件只传输一次。

分级传输，提高初始加载速度。

4.4 轻量化内存管理

为了确保复用的人物对象资源不被重复存储，需要将场景中的人群统一管理，我们用JS设计了以一个对象来管理整个的人群。

为了提高渲染效率，所有的人物在渲染时作为一个整体，一次性将人群所需的全部数据输入着色器。这就需要我们管理好每个人的所有参数信息（位置，大小，贴图类型，等等），渲染时一次性将这些信息传入shader。

4.4 轻量化场景渲染

骨骼动画数据的输入格式是32的float浮点数，经过测试发现骨骼动画数据实际上并不需要这么高的精度，所以我们设计了一种16位的浮点数格式，这样就使得传入着色器的骨骼数据量减少了一半。

项目中人物的贴图左右对称，所以可以只传入着色器左半部分贴图，这样传入着色器的贴图数据量就减少了一半。

5.效果

6.结论

多个小部件组合成的大型复杂模型

服务器端(轻量化处理)

模型分解->找出重用组件->外壳抽取

网络传输（提高传输速度）

优先传输先看到的部件,重用部件只传输一次

客户端（提高初始渲染速度，渲染质量）

重用部件使用实例化渲染

高精度模型处理（PM+LOD）

服务器端

将复杂模型简化，并记录简化过程中删除和修改的点以及操作的次序

网络传输

先传输基础模型，然后传输更新模型所需的数据

客户端

先将基础模型添加到场景中，然后不断更新模型

每更新到一定程度就记录下当前精度的模型数据

当彻底恢复成原模型后通过记录的数据实现LOD

提高初始加载速度

初始加载速度

PM处理

外壳提取

静态模型的轻量化处理

模型组件拆分

找出重用组件

外壳抽取

动态模型

细粒化调度机制流程

开始，轻量化预处理，找出重用部件，确定传输次序，完成

顶点着色器

一、输入的数据

模型顶点的信息：

顶点的原坐标

顶点的UV

顶点的对应骨骼编号

实例化对象的信息：

对象的变换矩阵

对象的动画播放速度和动画类型

对象的各部位贴图类型和色调

共用的unifrom类数据：

骨骼数据

时间

二、数据处理

判断顶点所在的区域

计算方法：

根据顶点在模型中的原坐标，判断顶点在头、上身、下身哪个部分

功能：

实现模型的分区，方便片元着色器的处理

实例化对象变换矩阵

计算方法：

将输入的对象变换矩阵信息解码为矩阵

功能：

用于单个实例化对象的放缩、移动、旋转

骨骼动画变换矩阵

计算方法：

找出当前点对应的骨骼编号

判断是否骨骼是否运动（静止骨骼的数据单独存放）

静止：根据骨骼编号找到对应的骨骼矩阵

运动：根据速度和时间计算帧序号

根据骨骼编号和帧序号找到对应的骨骼矩阵

功能：

控制动画的播放速度、种类

不同动画的搭配

通过骨骼放缩实现人物的形态差异

数据的存储和传输：

将静止骨骼的数据单独存放：

每帧变换矩阵都相同的骨骼，只存放一帧的数据

设计了专门用于传输骨骼的数据的浮点数类型：

新的16位浮点数类型所占空间比32位的float减少了一半。

将骨骼数据以纹理贴图的格式传入着色器。

1. 输出

顶点对应屏幕上的位置

向片元着色器输出

顶点所在的区域，

贴图类型，

色调

片元着色器

一输入

共用的unifrom类数据：

纹理贴图

顶点着色器输入的数据：

顶点所在的区域

贴图类型

色调

二数据处理

UV计算：由于人的纹理贴图左右对称，纹理贴图只传入了左半部分，要将UV都对应到左侧。

色调调整

三输出

对应像素的颜色

轻量化处理

大规模动态模型

方法：实例化渲染、参数化差异、模型资源分级复用

应用：听众人群

由多个小部件组合成的复合模型

方法：实例化渲染、外壳提取、视锥剔除

应用：会议厅模型（尤其是椅子、门和墙壁部分）

高精度模型处理

方法：PM、LOD、视锥剔除

应用：嘉宾模型