狭义相对论视觉效应模拟　项目报告

76 朱雨田　PB23050888

84 刘子潮　PB22000015

85 夏子汐　PB22000057

2024 年 6 月 9 日

一　项目介绍

自从爱因斯坦提出狭义相对论以来，就有许多学者对狭义相对论的视觉效应做出了研究。我们基于本课程提供的节点编程渲染框架，使用 C++ 和 OpenGL 实现了在狭义相对论条件下通过运动相机观察给定场景，包含静止和运动对象，其中包含运动物体的推迟时间变换、相机视角下光线方位和颜色的变换等，并进行了美观的渲染设计。我们还实现了上帝视角的模拟，和基于狭义相对论动力学的质点弹簧系统物理仿真。

分工情况

朱雨田：研究并修改框架底层代码，实现摄像机速度、光速、模型等物理量在不同模块间的共享；调整 OpenGL，添加必要的中间量传递；实现狭义相对论视觉模拟中的推迟时间部分；设计观察者的交互逻辑；设计渲染部分，添加更多渲染节点与 Shader，调整实现更好的视觉效果；制作、修改各种场景模型、纹理；录制展示视频。

刘子潮：推导狭义相对论视觉效应的理论和实现方法；实现相机视角下光线方位和颜色的变换；推导狭义相对论动力学和质点弹簧系统物理仿真的理论、实现质点弹簧系统的物理仿真并进行模拟结果的比较分析；文档原理和实现部分主要编写。

夏子汐：推导狭义相对论视觉效应的理论和实现方法；推导上帝视角理论、实现上帝视角并且对结果进行分析；进行了一些模型和场景的测试。

二　功能描述

相机的运动

(1) 相机通过 WASDQE 键进行加减速，且支持相机的平滑移动。

(2) 在 Relativity Console（如图2）中指定相机的最大速度。按住 Shift 移动的最大速度即是指定的最大速度，而松开 Shift 移动的最大速度则是指定的最大速度的一半与 8m/s 相比的较小值。

(3) 支持两种相机移动锁：一种是为相机指定速度，而不发生真正的移动，便于观察场景的几何变化；另一种是让相机发生运动，但对其他节点暴露的速度为 0，便于在移动时避免几何变换干扰判断。按下 Tab 键即可切换相机移动的锁定模式。

(4) UsdView 的菜单栏右侧会显示实时的锁定模式与相机速度（m/s 与 c 两种单位，如图1），中部显示目前设定的光速。

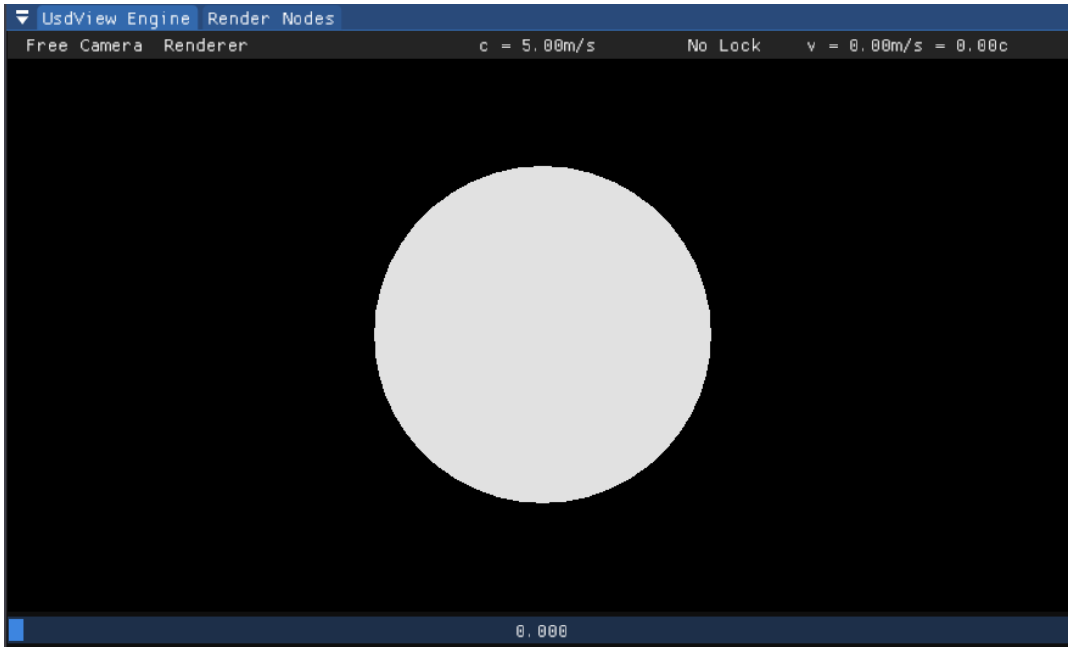

图 1: 相机界面

重要的相对论节点

1. 相对论控制台 在 Cmposition 中提供了控制相对论模拟基本参数的节点 Relativity Console（见图2）。可以控制的参数为:

(1) 光速: 单位 m/s。

(2) 相机最大速度: 单位是光速，建议不要超过或接近 1，否则视觉效果和模拟可能会出现不可预料的问题。

(3) 是否使用有限光速的变换: 建议打开，具体效果见原理叙述或者效果展示，这是光速有限的必然结果。

(4) 牛顿迭代次数: 控制求解推迟时间效应和求解上帝视角的迭代法迭代次数; 更大的迭代次数可以获得更好的效果，但是也会更加消耗性能，过小的迭代次数容易找不到解，导致网格不正常的变形和错误。

(5) 迭代的 “阻尼”: 迭代法求解可能会出现爆炸现象，如果对于每次迭代改变的量提供一个 damping 效果，可以缓解爆炸。如果出现爆炸现象，可以尝试增加牛顿迭代次数或者减小迭代 damping 系数。

(6) 是否使用上帝视角: 如果需要使用上帝视角模拟，请打开。

2. 相对论光栅化节点 在 Render node 中提供了相对论光栅化节点 Relativity Rasterize（如图3），默认给好了相对论着色器 Vertex Shader 和 Fragment Shader。这里的光栅化考虑了相对论波矢变换，即对于光线进行的相对论变换（方向和波长变换），但是没有考虑有限光速导致的推迟时间变换，请在使用这个节点时一定要打开 Console 中的 Use Limited-C Transform，以保证正确的视觉现象。可以调的参数为 Doppler Mix 和 Draw Grid，其中 Doppler Mix 取值范围为 $[0, 1]$，0 表示忽略多普勒效应，1 表示给出全部多普勒效应。如果多普勒效应过于强烈导致光谱变为红外/紫外光无法看见，可以尝试调低参数，但这并不是物理上正确的。打开 Draw Grid 可以画出一个 xOy 平面的网格，用来参照和观察物体的变形。

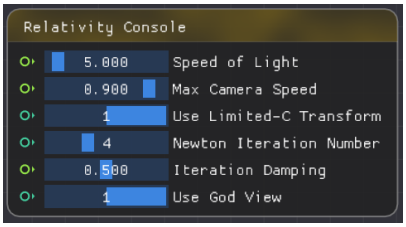

图 2: 相对论控制台

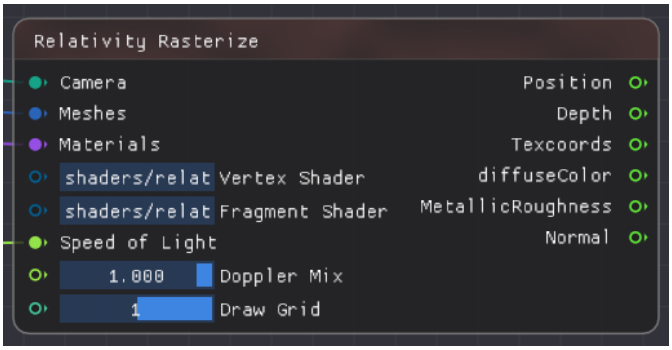

图 3: 相对论光栅化节点

3. 渲染后处理节点 如图4，在 Render node 中提供了 Post Process 节点和 Mix Color 节点。Post Process 节点接受一系列 G-Buffer 中的信息，并将其传入 Shader 中以对图像实现 SSAO、模糊、Bloom 等后处理效果；Mix Color 节点接受两个图像纹理，并用 Shader 所定义的混合模式进行混合，输出混合后的图像。一个实际后处理的例子如图14，见第 16 页。

图 4: 渲染后处理节点

对于后处理 Shader，我们提供了 `blur.fs` 用于实现单方向的高斯模糊，`bilateral_filter.fs` 用于实现纹理的双边滤波，`relativity_background.fs` 用于绘制主要展示场景的天空盒。

对于混合模式 Shader，我们提供了 `mix.fs`，`multiply.fs`，`add.fs`，分别对应常见图像处理软件中的 Alpha 混合（“正常”）、正片叠底、叠加三种混合模式。

节点相关参数的使用方法：

(1) Post Process 节点提供 4 个可以输入的参数，可以传入 Shader 中，在 Shader 中使用如下代码可以导入并使用参数：

Shader 输入参数

```
uniform float arg1;
uniform float arg2;
uniform float arg3;
uniform float arg4;
```

(2) Mix Color 节点的 Alpha 参数用于调整两种颜色的混合比例等自定义用途。

需要注意的是，这些节点与相对论视觉效应本身无关，我们只是为了更好的视觉效果与更方便地对比而设计这些节点。

狭义相对论视觉效应

节点图如图5所示。使用时请注意：

(1) 需要将 Console 中的 Use God View 关闭，同时接上 Relativity Rasterize 节点渲染。

(2) 支持多种方法导入物体：Read Usd 导入（Geometric Nodes 或者 Composition 部分）、动画处理后导入、物理仿真结果导入。但是在当前代码下，仅有“在 Geometric Nodes 中输出，且 Write Usd 节点中 Prim Path 为‘geometry’”的网格可以进行推迟效应的计算。如果通过其他方式导入了运动物体，或者希望所有网格都可以参与推迟效应计算，请删去 `RCore/hd_USTC_CG_GL/geometries/mesh.cpp` 的第 206 行：

mesh.cpp

```
206    if (path == "/geom/geometry")
```

并重新编译，这样会导致当载入大型模型（例如 `Sponza_subdivided.usda`）时性能开销增加。

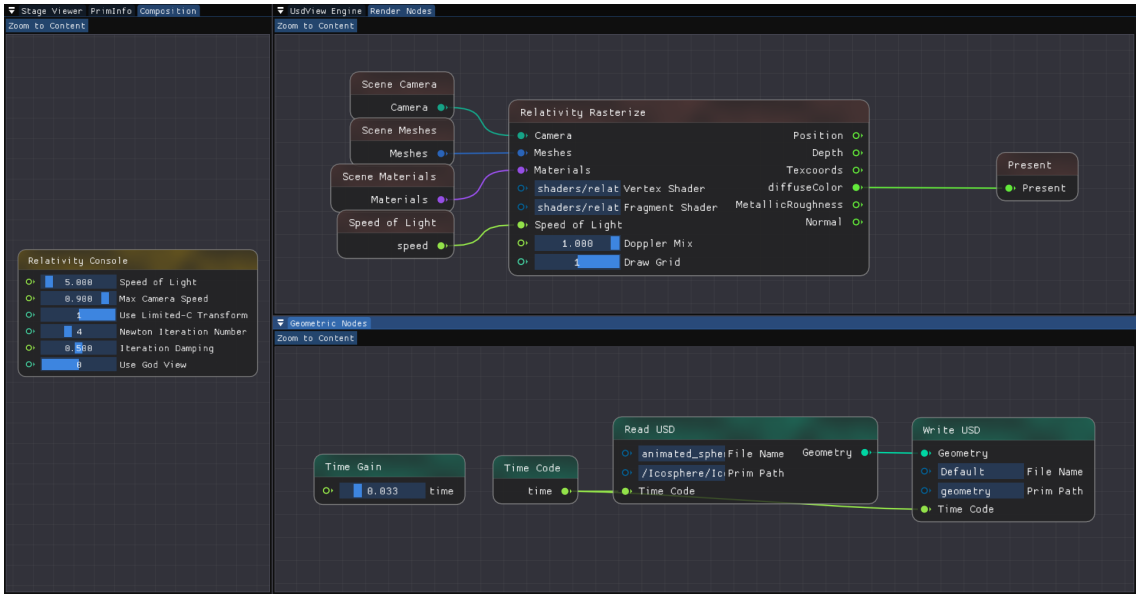

图 5: 相对论视觉效应节点图

上帝视角模拟

如图6，是一个最简单的上帝视角模拟节点图的例子。使用时需要注意：

(1) 打开 Use God View ，不使用相对论光栅化节点（不考虑光的传播），直接使用正常 Rasterize 方法渲染求得的网格。

(2) 请先跑一遍完整的时间，存储所有时刻的网格，再开始上帝视角的模拟。

(3) 由于实现机制的问题，要求场景里面的物体都是运动物体，而且必须在 UsdView Engine 正在播放的状态下进行摄像机的移动，才能够触发上帝视角模块。

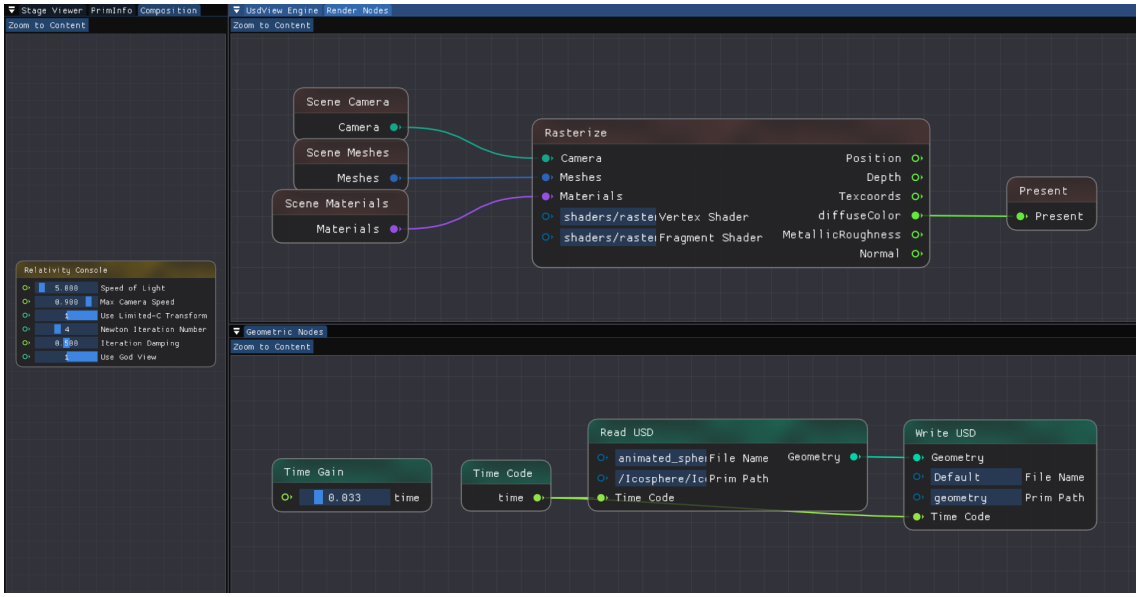

图 6: 上帝视角模拟节点图

狭义相对论质点弹簧系统物理仿真

如图7为进行含球体碰撞的质点弹簧系统物理仿真的节点图（均在 Geometric Nodes 中）。其中质点弹簧计算节点与作业 8 几乎相同，增加了光速的输入和能量修正功能 `enable energy correction`，设置为 0 表示不修正，设置为 1 表示使用切线修正，设置为 2 表示使用割线修正，推荐设置为 2，详细含义见后文原理和实现。结果输出到 Write USD 中，即为默认系中的图形，再经过 Relativity Rasterize 或者上帝视角，可以在相对论视角看到物理仿真结果。

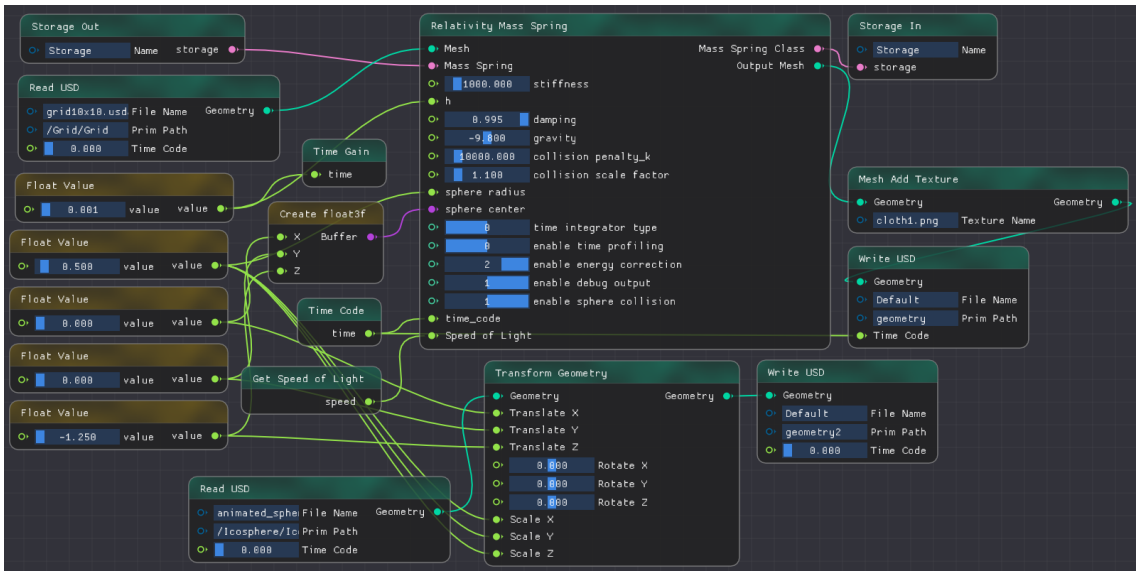

图 7: 相对论质点弹簧物理仿真节点图

三　效果展示

参见演示视频和物理仿真结果，文件地址见附录。

四　背景、原理和实现

狭义相对论的时空观

牛顿运动定律是基于实验结果总结的对宏观、低速物体的运动规律，适用性非常广泛。牛顿力学描述了惯性参考系中任意物体的运动规律，其满足伽利略相对性原理（对任意惯性参考系力学定律具有相同的形式），同一物理事件在两个惯性参考系中的时空坐标遵循伽利略变换（速度线性合成、时间平移）。但是有一些实验给出了不同的结论。粒子物理实验表明，当粒子能量持续增加时，其速度将趋于一个确定的值，这与牛顿力学中能量的计算不符。电磁学从各类电磁实验结果发展出来，总结为麦克斯韦方程组，其说明电磁波在真空中传播的速度为一常数 $c=\frac{1}{\sqrt{\varepsilon_0\mu_0}}$，这与牛顿力学中速度的线性合成相矛盾。种种新发现的理论与实验上的矛盾均展示了对宏观、高速物体牛顿运动定律不再适用。

爱因斯坦综合了前人对这样的矛盾的各类解读和尝试，提出了狭义相对论。狭义相对论假设：

(1) 空间是均匀的、各向同性的，时间是均匀的；

(2) 真空中的光速对于所有的观测者都具有相同的数值，与光源的运动无关；

(3) 物理定律在所有惯性参考系中具有相同的形式。

我们使用四维坐标来表达时空位置，定义逆变时空坐标（四维位矢）为 $x^\alpha=(x^0,x^1,x^2,x^3)=(ct,x,y,z)$，在闵可夫斯基空间中的度规矩阵

$$g_{\alpha\beta}=g^{\alpha\beta}=\begin{pmatrix}-1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}$$

定义协变时空坐标为 $x_\alpha=g_{\alpha\beta}x^\beta=(-x^0,x^1,x^2,x^3)=(-ct,x,y,z)$。则在此度规下可以证明两个惯性参考系间一般的线性变换 $x^{'}=\Lambda x+a$（四维形式为 $x^{'\alpha}=\Lambda^\alpha{}_\beta x^\beta+a^\alpha$）需要满足 $\Lambda^{\mathrm{T}}g\Lambda=g$（或者说 Λ 是洛伦兹群 $O(3,1)$ 的元素，这样的变换称作庞加莱变换）。若无平移，$a=0$，则为洛伦兹变换。通过无穷小洛伦兹变换可以得到与坐标系选择无关的洛伦兹变换为：

$$\begin{cases}ct^{'}=\gamma(ct-\vec{\beta}\cdot\vec{x})\\ \vec{x}^{'}_{\parallel}=\gamma(\vec{x}_{\parallel}-\vec{\beta}ct)\\ \vec{x}^{'}_{\perp}=\vec{x}_{\perp}\end{cases}\qquad\text{或者}\quad\begin{cases}ct^{'}=\gamma(ct-\vec{\beta}\cdot\vec{x})\\ \vec{x}^{'}=\vec{x}+\frac{\gamma^2}{\gamma+1}(\vec{\beta}\cdot\vec{x})\vec{\beta}-\gamma\vec{\beta}ct\end{cases}$$

其中带 $'$ 号表示的坐标系以速度 $\vec{v}=\vec{\beta}c$ 相对于不带 $'$ 号的坐标系运动，$\gamma=\frac{1}{\sqrt{1-\beta^2}}\geq 1$，下标 $\parallel$ 表示与速度方向平行的分量，$\perp$ 表示与速度方向垂直的分量（均是一个矢量）。令 $\vec{\beta}$ 反向，可以得到反变换：

$$\begin{cases}ct=\gamma(ct^{'}+\vec{\beta}\cdot\vec{x}^{'})\\ \vec{x}_{\parallel}=\gamma(\vec{x}^{'}_{\parallel}+\vec{\beta}ct^{'})\\ \vec{x}_{\perp}=\vec{x}^{'}_{\perp}\end{cases}\qquad\text{或者}\quad\begin{cases}ct=\gamma(ct^{'}+\vec{\beta}\cdot\vec{x}^{'})\\ \vec{x}=\vec{x}^{'}+\frac{\gamma^2}{\gamma+1}(\vec{\beta}\cdot\vec{x}^{'})\vec{\beta}+\gamma\vec{\beta}ct^{'}\end{cases}$$

可以证明正变换与反变换是相容的。注意到这个变换是关于原点不变的（t、$t^{'}$、$\vec{x}$、$\vec{x}^{'}$ 均为 0 时变换式成立），然而实际情形可能并非如此，故实际使用时应该将各量理解为相对值、变化量。一般洛伦兹变换的系数矩阵 $\Lambda=\begin{pmatrix}\gamma&-\gamma\vec{\beta}\\-\gamma\vec{\beta}&\overleftrightarrow{I}+\frac{\gamma^2}{\gamma+1}\vec{\beta}\vec{\beta}\end{pmatrix}$，其中 $\overleftrightarrow{I}$ 为单位张量。

洛伦兹变换指出同一物理事件在不同惯性参考系中发生的时间与位置均不一定相同，且呈现与惯性参考系间相对速度具有较为复杂关系的线性关系，打破了原有的简单的伽利略变换时空观。一个很重要的不同点就是同时的相对性，即在一个惯性参考系中同时发生的两个事件，在另一个相对运动参考系看来不同时发生。从洛伦兹变换出发，我们可以得到狭义相对论中最被人们熟知的两个效应：动钟变慢和动尺收缩。动钟变慢是指：对一个匀速运动的物体，此时 $\vec{x}^{'}=0$，则有 $t=\gamma t^{'}$，即在一段运动过程中，运动物体自身记录的时间 $t^{'}$ 比静止观察者记录的时间 t 要短。这是因为，两个钟的位置不同，因而从洛伦兹变换的角度时间也就不同，体现了时间间隔的相对性。动尺收缩是指：对一个沿着长边方向匀速运动的尺子，自身系中同时（$t^{'}=0$）获得尺子的两端的坐标，得到长度为 $x^{'}=l_0$（此处仅考虑这一维的运动），则在静止观察者看来，他要测量尺子的长度，则需要在同一时刻（$t=0$）获得尺子两端的坐标，代入变换得到 $x^{'}=\gamma x$，即测量到的长度 $l=x=\frac{l_0}{\gamma}<l_0$，测量到的长度比尺子自身的长度要短。这是因为，在静止观察者看来同时测量尺子两端，在运动尺子看来则不是同时测量，自然得不到正确的长度，这体现了空间间隔的相对性，其本质也是同时的相对性。

更多物理量的变换

基于闵可夫斯基空间的描述工具和一般的线性变换形式，我们可以拓展到其他四维标量、矢量和张量的变换，得到更多其他物理量的变换形式。定义四维速度 $u^{\alpha}=\frac{\mathrm{d}x^{\alpha}}{\mathrm{d}\tau}=\gamma(c,\vec{v})$，其中 τ 为固有时（自身系中经过的时间），则经过变换后可以整理得到狭义相对论的速度合成法则：$\begin{cases}\vec{\beta_{\parallel}}=\frac{\vec{\beta_{\parallel}^{'}}+\vec{\beta_0}}{1+\beta_0\beta_{\parallel}^{'}}\\ \vec{\beta_{\perp}}=\frac{\vec{\beta_{\perp}^{'}}}{\gamma(1+\beta_0\beta_{\parallel}^{'})}\end{cases}$，表示在相对静止系以 $\vec{\beta_0}$ 运动的惯性参考系中以 $\vec{\beta^{'}}$ 运动的物体在静止系中的速度 $\vec{\beta}$，$_{\parallel}$ 表示与 $\vec{\beta}$ 平行的分量。定义四维加速度 $w^{\alpha}=\frac{\mathrm{d}u^{\alpha}}{\mathrm{d}\tau}=(\gamma^4\vec{\beta}\cdot\vec{a},\gamma^4\vec{a}_{\parallel}+\gamma^2\vec{a}_{\perp})$，可以证明在自身系下的加速度 $\vec{a}^{'}$ 与静止观察者看到的加速度 $\vec{a}$ 之间的关系为 $\begin{cases}\vec{a}_{\parallel}=\frac{\vec{a}_{\parallel}^{'}}{\gamma^3}\\ \vec{a}_{\perp}=\frac{\vec{a}_{\perp}^{'}}{\gamma^2}\end{cases}$，静止观察者看到的加速度比自身系中的要小。定义四维动量 $p^{\alpha}=mu_{\alpha}=\gamma mc(1,\vec{\beta})=(\frac{\epsilon}{c},\vec{p})$，其中粒子的相对论能量 $\epsilon=\gamma mc^2=\frac{mc^2}{\sqrt{1-\beta^2}}$、相对论动量 $\vec{p}=\gamma m\vec{v}=\frac{m\vec{v}}{\sqrt{1-\beta^2}}$，所以 $p^{\alpha}=(\frac{\epsilon}{c},\vec{p})$。光子的能量 $\epsilon=c|\vec{p}|$，它的相对论四维动量矢量可以表示为 $p^{\alpha}=(|\vec{p}|,\vec{p})$。采用这样形式的四维动量可以使得其为四维矢量，且在低速情形下回归到经典力学的结果。电磁场也可以构造相应的四维矢量，写成协变的形式。在考察波动方程时，定义四维波矢 $k^{\alpha}=(\omega/c,\vec{k})=\omega/c(1,\hat{k})$，其中 ω 为频率，$\vec{k}$ 为三维的波矢，可以证明四维波矢为四维矢量，得到波矢的变换：

$$\begin{cases}\omega^{'}=\gamma(\omega-c\vec{\beta}\cdot\vec{k})\\ \vec{k}_{\parallel}^{'}=\gamma(\vec{k}_{\parallel}-\vec{\beta}\omega/c)\\ \vec{k}_{\perp}^{'}=\vec{k}_{\perp}\end{cases}\qquad\text{或者}\quad\begin{cases}\omega^{'}=\gamma(\omega-c\vec{\beta}\cdot\vec{k})\\ \vec{k}^{'}=\vec{k}+\frac{\gamma^2}{\gamma+1}(\vec{\beta}\cdot\vec{k})\vec{\beta}-\gamma\vec{\beta}\omega/c\end{cases}$$

从波矢的变换我们可以看出，运动的波源发出的波接收到时不仅频率发生了变化（多普勒频移），波矢的方向也发生了变化。将光当作波来处理，则光线在不同惯性参考系中方向会不同、频率会不同，即颜色会不同。

狭义相对论的视觉效应

动钟变慢和动尺收缩是广为人知的狭义相对论效应，然而若光速下降到一定的数值使得有可能可以通过视觉方法观察该情形下相对论性物体（β 不远小于 1）的运动，我们能看到动尺收缩的情况吗？答案是否定的，这是因为我们前文讨论的效应采用了“上帝视角”，即能获得该时刻物体的位置形成形象。然而，因为物体发出的光传播到眼睛或者摄像机需要一定的时间，且涉及到光线的传播，我们能看到的不是当前时刻物体的位置，而是若干时间前物体的位置，这导致狭义相对论下物体的视觉效应并非动尺收缩这样简单。我们可以通过

上述的四维波矢变换来实现视觉效应。我们假设世界空间为一个“默认惯性参考系”（默认系），相机在其中运动，产生相应的视觉效应。

从原理上，视觉效应主要来源于光线的方向变化和波长变化，过程以“光线”为中心，自然地可以尝试使用光线追踪的方式进行模拟。考虑渲染相机看到的图像，发出光线，先变换到默认系中，再与默认系中的场景求交；光线打到物体上，如果物体相对默认系运动，则需要再变换到物体（关于光线打到的位置）的自身系，再进行 BRDF 采样、随机选择出射方向，再变换回默认系，继续进行光线弹射的过程。与传统的光线追踪相比，主要的区别就在于在每一次发射或接收时均需要变换到物体自身系或默认系中，考虑频移和方向变化。然而，视觉效应还有另一来源，即物体在发出光线到相机接收光线的时间内运动了，故看到的物体应当是此前若干时间的物体的位置，需要作“推迟时间”的处理（见下），且对不同质点而言向前回溯的时间是不同的。使用光线追踪算法，在求交时就会遇到无法确定时刻和物体位置的问题，需要完全求解该时刻整个网格的形状并重新求交（在四维空间中，见下），求解是很困难的，可预见地会有大量的计算量。基于这样的考虑，我们决定采用光栅化的管线来实现视觉效应的模拟。

首先考虑在默认系中静止的物体，其在运动的相机的自身系中的视觉效应。从光追的思想来看就是将光线进行相应的偏折，与物体相交。从光栅的思想来看，主要考虑物体在空间中的位置，则可以假设物体发出打向摄像机的光线，这一光线在相机系中变换了角度，则等效于物体在相机系中移到了对应角度处。我们可以对 Mesh 中的每一个顶点进行空间位置的移动，即在 Vertex Shader 中通过传入相机的位置、速度等参数计算在相机参考系中物体的位置。可以推导发现：使用光线思想得到的物体的新方位与使用洛伦兹变换得到的位置的方位是一致的，这体现了相对论变换的一致性。在这样的变换下，物体间的遮挡关系仍然符合物理，则经过这样的顶点变换后我们就可以得到相机系中物体的位置。相对相机的位置矢量 $\vec{x}$ 的变换为：

$$\vec{x}^{'} = \vec{x} + \frac{\gamma^2}{\gamma+1}(\vec{\beta}\cdot\vec{x})\vec{\beta} - \gamma|\vec{x}|\vec{\beta}$$

此后，OpenGL 对新的 Mesh 进行光栅化，且我们通过 Normal Mapping 的步骤使得变换顶点位置后能计算出正确的法向。在 Fragment Shader 中，我们需要考虑颜色的变化，因为光栅化成许多像素，故需要对每个像素通过变换确定波长的变化倍数

$$\omega = \omega^{'}\gamma(1+\vec{\beta}\cdot\hat{k}^{'})$$

$$\lambda^{'} = \lambda\gamma(1+\vec{\beta}\cdot\hat{k}^{'})$$

注意 $\hat{k}^{'}$ 为相机系中光线射向相机的方向向量。知道了波长的变换倍数，我们需要得出已知 (r,g,b) 颜色的变换方式。相关内容见下文。这样就实现了对默认系中静止的物体的视觉效应的模拟。

再考虑在默认系中运动的物体，其在相机系中的视觉效应。我们需要考虑在此时刻相机接收到的光线是由物体在此前某时刻发出。除了考虑光线波长变化和方向变化外，还需要考虑光线传播所用的时间（两者都是光速有限的直接结果）。对于运动的物体，我们看到的其实是物体过去传递过来的光（例如说我们看到的太阳是八分钟前的）。

时空图可以帮助我们更好地理解狭义相对论。如图8，利用三维空间表示一个四维矢量，竖直轴表示时间（实际上为 ct，保持量纲相同），与之垂直的平面就代表了三维空间。由于运动速度不能超过光速，那么时间上的距离 $c\Delta t$ 一定大于空间上距离 $|\Delta\vec{x}|$。如果再考虑光的运动，一定有 $c\Delta t = |\Delta\vec{x}|$，各个方向发出的光组成了图中的“未来光锥”面（这里的面实际上是三维的，称作超曲面），锥面内的时空点就是可能经过的。同理考虑原点过去的运动，也在过去光锥内；照射到原点的光一定经过过去光锥。这说明中能够在某一时刻 t 到达摄像机的光线一定在”摄像机的过去光锥”上，如果物体有一个点在某一时刻 t^* 正好经过某处，这个四维坐标正好在过去光锥上（如图9），那么此时物体这个点 t^* 时刻发出的光可以在 t 时刻到达摄像机形成物体的像（四维空间中某个质点的运动轨迹称作世界线，这里的意思就是物体上面某点的世界线与摄像机的过去光锥相交）。设摄像机四维空间坐标为 (x,y,z,ct)，那么（过去/未来）光锥上面的点 (x^*,y^*,z^*,ct^*)，一

定满足 $c|t^*-t| = \sqrt{(x^*-x)^2+(y^*-y)^2+(z^*-z)^2}$，记时空间隔为 $\Delta s^2 = -c^2\Delta t^2+\Delta x^2+\Delta y^2+\Delta z^2$，那么由时空图可知，（过去/未来）光锥内的点与原点的时空间隔小于零，锥面上点与原点的时空间隔等于零，光锥外的点的时空间隔大于零。为了让后面求解方程拥有更好的性质，这里改写为 $\Delta s = -c|\Delta t| + |\Delta\vec{r}|$ $(|\Delta\vec{r}| = \sqrt{\Delta x^2+\Delta y^2+\Delta z^2})$，同样满足上面的条件。由于我们看到的是物体之前时刻的像，电动力学上把 t^* 称作推迟时间。

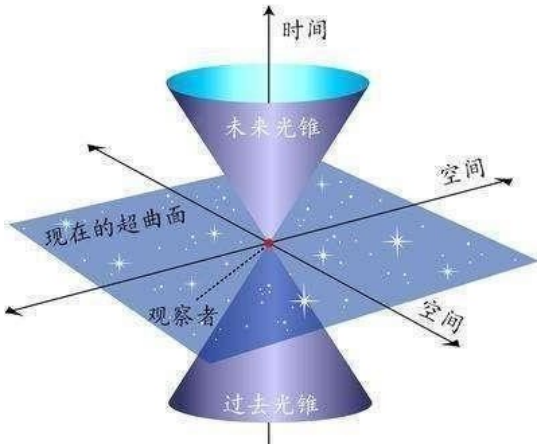

图 8: 四维时空图

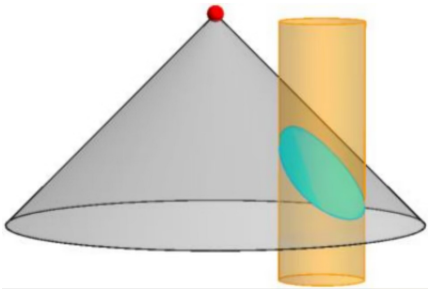

图 9: 过去光锥与世界线相交，相交面就是实际看到的三维网格

我们知道静止物体在相机系中任意观察时刻的位置，因为物体不运动，不涉及“推迟时间”的处理。对于运动物体，我们需要在每时每刻都获得物体的运动轨迹，求解方程，即

$$|\vec{x}(t^*) - \vec{x}_{\text{cam}}| = c(t-t^*)$$

则我们要求的物体位置为 $\vec{x}(t^*)$。注意到，对物理场景应有物体的速度不超过光速，故方程左侧的变化率不会大于右侧，这是一个单调函数的方程，可以使用二分法求解，也可以使用牛顿迭代法求解。对光速不太慢、物体速度不太快的情形，方程的导数较大，牛顿迭代法效率较好。求解得到物体每一个顶点的新位置后，此时就与上文静止物体的情形相同了，可以进行光栅化、颜色计算等步骤，此时颜色计算就采用两者的相对速度作为参考系之间的速度。这样就实现了对默认系中运动的物体的视觉效应的模拟。需要注意，对于运动物体，我们需要在每一次渲染时重新计算物体的位置，这是因为相机的位置是随着时间变化的。

颜色变换

我们需要对颜色的波长乘上一个倍数，颜色由 (r,g,b) 形式给出。为了追溯到波长，或者颜色对应的波谱，我们需要回到 sRGB 色域的定义（大部分 Windows 系统设备显示一般程序内容时均运行在 sRGB 色域上）。如图10和图11所示（来源维基百科`https://en.wikipedia.org/wiki/CIE_1931_color_space`），RGB 颜色值的定义如下：

$$R = \int_0^\infty I(\lambda)r(\lambda)\mathrm{d}\lambda, \quad G = \int_0^\infty I(\lambda)g(\lambda)\mathrm{d}\lambda, \quad B = \int_0^\infty I(\lambda)b(\lambda)\mathrm{d}\lambda$$

其中 $I(\lambda)$ 为频谱功率分布，此处我们需要将波长乘上一个倍数，则可以通过移动 $I(\lambda)$ 实现。理论上波长和方向变化可能导致光强变化，但保证变化前后指定区域一段物理事件的时间内能量守恒，频谱功率分布的高度也应该变化。

上述是从定义得到的计算方法，具有高的准确度，但是运算复杂、需要许多先验内容，极为不适合在 Shader 这样的轻量计算环境中使用。我们尝试简化颜色匹配函数，如图12所示（来源文献 Miha, Dragos, and Eugen

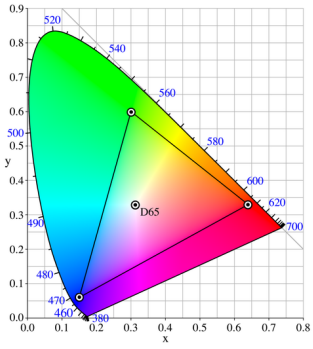

图 10: RGB 原色的色域

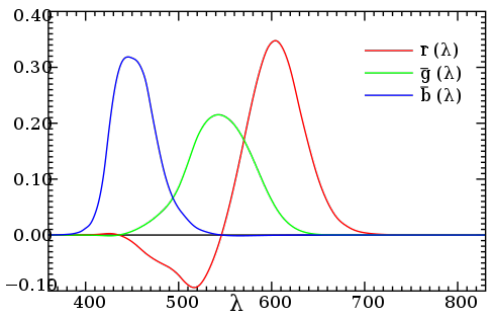

图 11: RGB 颜色匹配函数

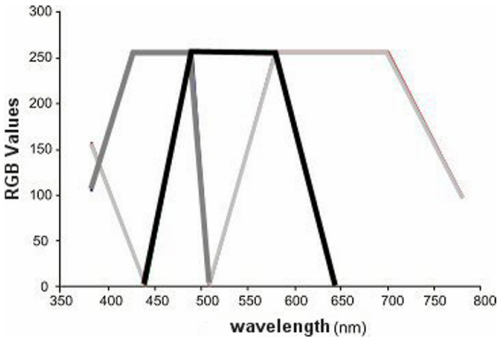

图 12: 简化 RGB 颜色匹配函数

Strajescu. "FROM WAVELENGTH TO RGB FILTER." University Politehnica of Bucharest Scientific Bulletin, Series D: Mechanical Engineering 69.2 (2007): 77-84.)，简化为分段线性的形式，适合在 Shader 中计算。此时每个频率就对应了一个颜色，采用 Stack Overflow 中提供的线性、增加衰减和 Gamma 修正的算法（来源 `https://stackoverflow.com/questions/1472514/convert-light-frequency-to-rgb`）。我们需要先将给定的 RGB 颜色转换为合适的波长，再进行波长变换，再还原为颜色。从曲线中就可以看出，许多 RGB 颜色无法使用单一波长表示，此处采用简化方法，即假设任意 RGB 颜色由两种波长对应的颜色线性相加构成，强度和颜色均线性相加，更改波长不改变光强。尽量还原到两个波长接近的颜色，这样与实际情况更为接近。在实践中，尤其当光速与运动速度相近时，波长很容易变换到紫外和红外区域，使得区域变黑，难以观察；为了实现更好的视觉效果，我们设计了参数 Doppler Mix 来控制多普勒效应的混合比例。

相机的运动

此处我们采用了原有的相机，设置成通过 WASDQE 键进行加减速的方式，使得相机的运动较为平滑。为了确保渲染能够正常进行，相机的最大速度限制为光速。相机的速度通过 `renderParam` 在每次 `usdview_engine` 全局 GUI 刷新的时候从 `renderDelegate` 的 `SetRendererSetting` 传入渲染的 `Scene Camera` 节点，从而被渲染节点和 Shader 利用。

在理论上，类似于牛顿力学，狭义相对论的框架仍然只适用于惯性参考系。然而，相机的运动不可能没有加速度，不是一个惯性参考系。对此，我们的解决方式是采用“瞬动标架”——在这一时刻与相机运动速度相同的惯性参考系——来进行变换和渲染。这一方式的合理性可以从四维时空世界线的角度来考虑。考虑到实际物理情形速度是不会突变的，故光线的变换也是连续的，场景是连续地变化的。在目前的离散时间渲染下，使用平滑的相机运动形式可以使得渲染结果更加连续，观感更佳。另外，根据洛伦兹变换，相机参考系中经过 Δt 时间，默认参考系应该经过 $\gamma\Delta t$ 时间，我们在相机参考系中观察，则操作系统时间不是默认参考系经过的时间。然而，这在目前的框架下要实现存在不小的难度，故我们目前仍将相机参考系中的时间和默认参考系中的时间混为一谈。这种折中对于视觉效果的影响不大，对运动情况可能会有一定的效应。

推迟时间的实现

为了在每帧都对每个顶点用牛顿迭代法求解函数 $f(t^*) = |\vec{x}(t^*) - \vec{x}_{\text{cam}}| - c(t - t^*)$ 的零点，我们需要能低开销地访问 $f(t^*)$ 在每个时间的值。为了在光栅化前对顶点做出变换，我们研究了框架的运行原理，并决定在光栅化对每个 Mesh 调用的的更新顶点函数，即 `RCore/hd_USTC_CG_GL/geometries/mesh.cpp` 中的 `Hd_USTC_CG_Mesh::Sync` 中实现顶点变换的过程。

从 `usdview_engine.cpp` 向 Render Params 传入指向存储每个时间戳物体位置的 Global Usd Stage 指

针后，我们可以在 `Sync()` 函数中取得这些数据。获得的数据本身是离散的，如果要获取 $f(t^*)$ 在任意 t^* 的取值，则需要对该函数进行插值。框架本身有对离散位置插值的功能，但是受限于框架实现，每次调用插值都需要对整个 Mesh 的所有顶点都进行一次插值，这样的时间开销是难以接受的。因此，我们手动实现了物体位置的线性插值。

具体地，我们先获取存储的 Time Samples 列表，在需要计算 t^* 时刻顶点位置时，用 STL 中的 `std::lower_bound` 函数取得 t^* 在时间戳列表中的最大下界，并进一步算出最小上界。获取到临近的两个时间戳的位置信息后，我们认为顶点在这段时间里做匀速直线运动，采用线性插值计算 t^* 时刻的位置，并直接计算速度。对于 $t < 0$ 的情形，我们将物体视为静止以计算顶点位置。

求导得 $f'(t^*) = \dfrac{(\vec{x} - \vec{x}_{\text{cam}})}{|\vec{x} - \vec{x}_{\text{cam}}|} \cdot \vec{v} + c$，采用牛顿迭代法 $t^*_{n+1} = t^*_n - k\dfrac{f(t^*_n)}{f'(t^*_n)}$，即可在几次迭代后算出顶点合理的新位置。其中 $0 < k \leq 1$ 为常数（也就是 Iteration damping），一般取 1，但可以合理调整以保证迭代的稳定性。

还需要注意的是，3D 动画的原理一般是对模型整体应用 Transform，而不是对每个顶点记录其每个时间戳的位置。本实现需要考虑修改每个顶点的位置，而不能对整个物体应用相同的 Transform，因此在读入每一时间戳顶点位置时就将变换应用在了顶点上，对世界坐标计算新位置，并将单位阵作为 Transform 传入光栅化顶点。

上帝视角的模拟

除了视觉效应外，我们还保留了相对论的经典情况（例如动尺收缩）的模拟，能够体现传统观点中相对论的现象。这种观点下忽略了光的传播，认为我们“看到”的某个时刻 t_0 的内容就是四维空间中 $t = t_0$ 的切片。例如说我们说“$t = t_0$ 时刻某个地方有一个物体”相比于“看到了一个物体”，就忽略了视觉中要求的光的传播。对应地我们只需要找到 $t = t_0$ 时刻的所有点，就可以组成上帝视角的三维空间。相机参考系相对默认系运动，那么相机系看到的应该是 $t' = t'_0$ 的点集合。假设相机这一点处 $t = t'$（见上，忽略两个参考系相机点时间流逝的差别），且默认参考系中相机的位置变换到相机系中的相机位置，那么“相对相机四维坐标的四维矢量”就满足洛伦兹变换。由于相机可以看到自己的“该时刻”，那么我们说此时空间中有的东西的相对四维矢量一定满足 $\Delta t' = 0$。根据公式，那么该点满足的方程为 $c\Delta t - \vec{\beta} \cdot \Delta\vec{x} = 0$。对 Δt 求导，得到 $c - \vec{\beta} \cdot \vec{v}$（相机运动速度与时间间隔（导数可以仅与点在默认系中时间 t 有关）无关，所以不含其他项），一定大于零，说明原来方程左边是一个单调递增函数，可以有唯一解。

求解方程的方法和上节一样，采用牛顿迭代法。但是其中有一些不同之处：

(1) 推迟时间一定小于真实时间，但是此处 Δt 没有正负限制，所以正确的求解上帝视角需要先把默认参考系所有时刻的网格求出来。从实际操作上来看，就是要先跑一遍，让 Global Usd Stage 中的 GeomMesh 里面包含了 0 - 250 时刻的所有有效的网格信息，然后再进行上帝视角模拟；

(2) 此处的 t 的求解结果很有可能小于 0 或者大于 250，通过 Time Samples 的查找得到的结果，其实是默认该物体在 $t < 0$ 和 $t > 250$ 都处于静止状态，且与运动状态的位形连续，这样就把世界线延拓到了 $(-\infty, +\infty)$ 上，可以保证方程有唯一解。规避物体“产生和消失”的近似对于相对论视觉效果的实现来说是能够极大提升可行性和简便性的。如果物体涉及在某个时刻产生，在某个时刻消失的情况，由于同时的相对性，物体上两个不同点在默认系中同时产生，在相机系中就不会同时产生，所以那个时刻看到的网格就应该是一个残缺的（同样适用于推迟时间计算上）。这就需要涉及到对 Mesh 的点的删除操作，如果 Mesh 不够精密，可能还需要在某些地方进行插值计算，改变其拓扑结构的操作。认为物体不会产生和消失可以有效规避这种操作。

(3) 由于之后的步骤是直接接入 Rasterize 节点，需要在这里对求解结果进行洛伦兹变换。变换矢量形式 $\Delta\vec{x}' = \Delta\vec{x} + \frac{\gamma^2}{\gamma+1}(\vec{\beta}\cdot\Delta\vec{x})\vec{\beta} - \gamma\vec{\beta}c\Delta t$，代码中实现了这一步。如果代入方程 $c\Delta t - \vec{\beta}\cdot\Delta\vec{x} = 0$，那么可以化简形式，变为 $\Delta\vec{x}' = \Delta\vec{x} - \frac{\gamma}{\gamma+1}(\vec{\beta}\cdot\Delta\vec{x})\vec{\beta}$。可以看到，如果 $\vec{\beta}$ 变为 $-\vec{\beta}$，那么结果 $\Delta\vec{x'}$ 不变！这也可以解释展示视频中向前和向后移动看到类似场景的现象。

狭义相对论物理仿真

上述狭义相对论的时空观解决了狭义相对论下物体的运动学问题，但没有给出动力学问题的解决途径。基于在光速趋于无穷时运动规律应当与牛顿运动定律一致的思想，可以推导出狭义相对论下物体的运动学方程：

$$\frac{\mathrm{d}\vec{p}}{\mathrm{d}t} = \vec{F}$$

与牛顿第二定律的动量形式一致，但此处动量表示相对论动量 $\vec{p} = \gamma m\vec{v}$。或者写成协变的形式 $\frac{\mathrm{d}p^\alpha}{\mathrm{d}\tau} = K^\alpha$，其中 K^α 为四维力，四维动量的形式见上文。如果了解了力的形式，可以使用动量来描述物体的运动，或更直接地写出三维的运动方程：

$$\begin{cases}\gamma^3 m\vec{a}_\parallel = \vec{F}_\parallel \\ \gamma m\vec{a}_\perp = \vec{F}_\perp\end{cases} \quad \text{或者} \quad \gamma m\vec{a} = \vec{F} - (\vec{\beta}\cdot\vec{F})\vec{\beta}, \vec{a} = \frac{1}{\gamma m}(\vec{F} - (\vec{F}\cdot\vec{\beta})\vec{\beta})$$

这个方程是在任一惯性系中的结果，$\vec{\beta}$ 表达粒子运动的速度，故我们如果想要进行相对论性的物理仿真，只需使用上式在默认系中进行，而不需要考虑洛伦兹变换。这些是相对论物理仿真的基础。

考虑质点弹簧系统（Mass Spring）的仿真，原理和推导参考作业 8。质点弹簧系统假设有质量无体积的质点之间由轻理想弹簧连接，从而模拟布料的运动。在相对论情形下，我们仍然假设物体受到的力不变：重力是所述空间引力、自转等的平均效果，可以取合适的值；线性弹性力是对物理上的弹簧在弹性限度内的实验规律，只与位移相关，故在相对论情形下应当在弹性限度内不改变力的形式。

最简单的半隐式方法直接计算力、加速度，从而得到下一时刻的速度，利用新速度更新新位置。则相对论情形下只需更改速度更新式为下式即可。

$$\vec{v}_{n+1} = \vec{v}_n + \vec{a}\Delta t = \vec{v}_n + h\cdot\frac{1}{\gamma}(\frac{\vec{F}}{m} - \frac{\vec{F}}{m}\cdot\vec{\beta}\vec{\beta})$$

隐式方法设出下一时刻的位置和和速度，形成方程

$$\begin{cases}\vec{x}^{n+1} = \vec{x}^n + h\vec{v}^{n+1} \\ \vec{v}^{n+1} = \vec{v}^n + \frac{h}{\gamma}(\mathbf{I} - \vec{\beta}\vec{\beta}^\top)\cdot\mathbf{M}^{-1}(\vec{f}_{\text{int}}(\vec{x}^{n+1}) + \vec{f}_{\text{ext}})\end{cases}$$

其中 $\mathbf{M} = m\mathbf{I}$。故 $\vec{x}^{n+1} = \vec{x}^n + h\vec{v}^n + \frac{h^2}{\gamma}(\mathbf{I} - \vec{\beta}\vec{\beta}^\top)\mathbf{M}^{-1}(-\nabla E(\vec{x}^{n+1}) + \vec{f}_{\text{ext}})$，定义 $\vec{y} = \vec{x}^n + h\vec{v}^n + \frac{h^2}{\gamma}(\mathbf{I} - \vec{\beta}\vec{\beta}^\top)\cdot\mathbf{M}^{-1}\vec{f}_{\text{ext}}$。则方程化为

$$\frac{\gamma}{h^2}\mathbf{M}(\vec{x}^{n+1} - \vec{y}) + (\mathbf{I} - \vec{\beta}\vec{\beta}^\top)\cdot\nabla E(\vec{x}^{n+1}) = 0$$

原方法将该方程化为梯度为 0 的方程，从而转化为最小化能量函数的问题，使用最优化方法求解。此处需要考察方程能否化为梯度的形式，可以通过验证梯度的散度为 0 来判断。$\mathbf{I} - \vec{\beta}\vec{\beta}^\top$ 与坐标无关（此处采用 $\vec{v}_n$ 计算，并非严格隐式，否则非线性，更加难以处理），不参与 ∇ 运算。记 $\vec{x} = \vec{x}^{n+1} \in \mathbf{R}^{3n\times 1}$，判断 $\nabla E(\vec{x})\cdot\vec{\beta}\vec{\beta}$ 是否无旋：

$$\nabla\times(\nabla E(\vec{x})\cdot\vec{\beta}\vec{\beta}) = \nabla(\nabla E(\vec{x})\cdot\vec{\beta})\times\vec{\beta} = \vec{\beta}\cdot\nabla(\nabla E(\vec{x}))\times\vec{\beta}$$

弹簧的能量 $E=\sum \frac{1}{2}k(||\vec{x}-\vec{y}||-L)^2$，$\nabla E=\sum k(\vec{x}-\vec{y})(1-\frac{L}{||\vec{x}-\vec{y}||})$

$$\nabla(\nabla E)=\sum k(\mathbf{I}-\frac{L}{||\vec{x}-\vec{y}||^3}(||\vec{x}-\vec{y}||^2\mathbf{I}-(\vec{x}-\vec{y})(\vec{x}-\vec{y})^\top))$$

又 $\vec{\beta}\cdot\mathbf{I}\times\vec{\beta}=\vec{\beta}\times\vec{\beta}=\vec{0}$。因此

$$\vec{\beta}\cdot\nabla(\nabla E(\vec{x}))\times\vec{\beta}=\sum kL\vec{\beta}\cdot\frac{(\vec{x}-\vec{y})(\vec{x}-\vec{y})^T}{||\vec{x}-\vec{y}||^3}\times\vec{\beta}$$

不一定为 0。原矢量方程无法转换为梯度方程，无法变为最小化能量函数的形式，故不能使用原有的最优化方法求解。

如果不管此处的不正确性，仍然继续按照此前的方式进行，设存在一个最小化的函数 $\min_{\vec{x}} g(\vec{x})$，其中

$$\nabla g(\vec{x})=\frac{\gamma}{h^2}\mathbf{M}(\vec{x}-\vec{y})+(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\nabla E(\vec{x})$$

使用牛顿法迭代一次：$\vec{x}^{n+1}=\vec{x}^n-(\nabla^2 g)^{-1}\nabla g$，其中

$$\nabla^2 g=\frac{\gamma}{h^2}\mathbf{M}+(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\nabla^2 E(\vec{x})$$

使用相同的方法计算 Hessian 矩阵，得 $\nabla^2 g=\frac{\gamma}{h^2}\mathbf{M}+(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{H}$，注意 γ、$\mathbf{I}-\vec{\beta}\vec{\beta}^\top$ 在计算每一个顶点的时候都是不同的，应分别处理。$\vec{x}^{n+1}=\vec{x}^n-(\nabla^2 g)^{-1}\nabla g(\vec{x}^n)=\vec{x}^n+\Delta\vec{x}$，方程组为：

$$\nabla^2 g\Delta\vec{x}=-\nabla g$$

求解该方程组即可得到位移，从而确定下一时刻的位置和速度。

加速方法使用 Local-Global 优化方法，需要使用 g ，而这是不存在的。但若按照步骤进行：首先进行 Local Step：$\vec{d_i}=L_i\frac{\vec{x}_i}{\|\vec{x}_i\|}$。再考虑 Global Step：也是梯度形式的

$$\nabla g(\vec{x})=\frac{\gamma}{h^2}\mathbf{M}(\vec{x}-\vec{y})+(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\nabla E(\vec{x})$$

其中 $E=\frac{1}{2}\vec{x}^\top\mathbf{L}\vec{x}-\vec{x}^\top\mathbf{J}\vec{d}$，$\nabla E=\mathbf{L}\vec{x}-\mathbf{J}\vec{d}$，$\mathbf{L}=\left(\sum_{i=1}^s k_i\vec{A}_i\vec{A}_i^\top\right)\otimes\mathbf{I}_3$，$\mathbf{J}=\left(\sum_{i=1}^s k_i\vec{A}_i\vec{S}_i^\top\right)\otimes\mathbf{I}_3$，$(\vec{A}_i)_j=1$(i 弹簧左端点为 j)$,-1$(i 弹簧右端点为 j)$,0$(其他)，$(\vec{S}_i)_j=1(i=j),0(i\neq j)$。故求解的方程为

$$\gamma\mathbf{M}(\vec{x}-\vec{y})+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)(\mathbf{L}\vec{x}-\mathbf{J}\vec{d})=0$$

$$(\gamma\mathbf{M}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{L})\vec{x}=\gamma\mathbf{M}\vec{y}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{J}\vec{d}$$

求解该方程就可以得到各点的新位置。对固定点，最好采用投影矩阵来处理：$\vec{x}'=\mathbf{K}\vec{x}$，$\vec{x}=\mathbf{K}^T\vec{x}'+\vec{b}$，$\mathbf{K}$ 降维、$\mathbf{K}^T$ 升维，$\vec{b}$ 表示固定点，则

$$\mathbf{K}(\gamma\mathbf{M}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{L})\mathbf{K}^T\vec{x}'=\mathbf{K}(\gamma\mathbf{M}\vec{y}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{J}\vec{d})-\mathbf{K}(\gamma\mathbf{M}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{L})\vec{b}$$

注意到，这可以写出矩阵方程 $\mathbf{A}\vec{x}'=\vec{B}$ 的形式，但不再像经典情形 $\mathbf{A}$ 是运动过程的不变量，故无法实现相同矩阵的复用，无法实现加速求解。考虑能否近似地适用，设 $\mathbf{A}$ 可以表示为常矩阵乘上一个标量的形式(此时才能从式子中自由地提出和移动)，$\mathbf{A}=\mathbf{K}(\gamma\mathbf{M}+h^2(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{L})\mathbf{K}^T=\gamma\{\mathbf{K}(\mathbf{M}+\frac{h^2}{\gamma}(\mathbf{I}-\vec{\beta}\vec{\beta}^\top)\mathbf{L})\mathbf{K}^T\}$。速度变大，$\gamma\to\infty$，$\mathbf{A}\approx\gamma\mathbf{K}\mathbf{M}\mathbf{K}^T$；而速度很小时，$\gamma\to 0$，$\mathbf{A}\approx\mathbf{K}(\mathbf{M}+h^2\mathbf{L})\mathbf{K}^T$。可见这两种情形下 $\mathbf{A}$ 之间的差异无法用常数刻画。仅在近似情形：$m\gg kh^2$，$\mathbf{A}\approx\gamma\mathbf{K}\mathbf{M}\mathbf{K}^T$，此时可以使用该方法求解。鉴于实际仿真时常取 $m=1/N,k=1000,h=0.01,kh^2=0.1$，故该方法不适用。

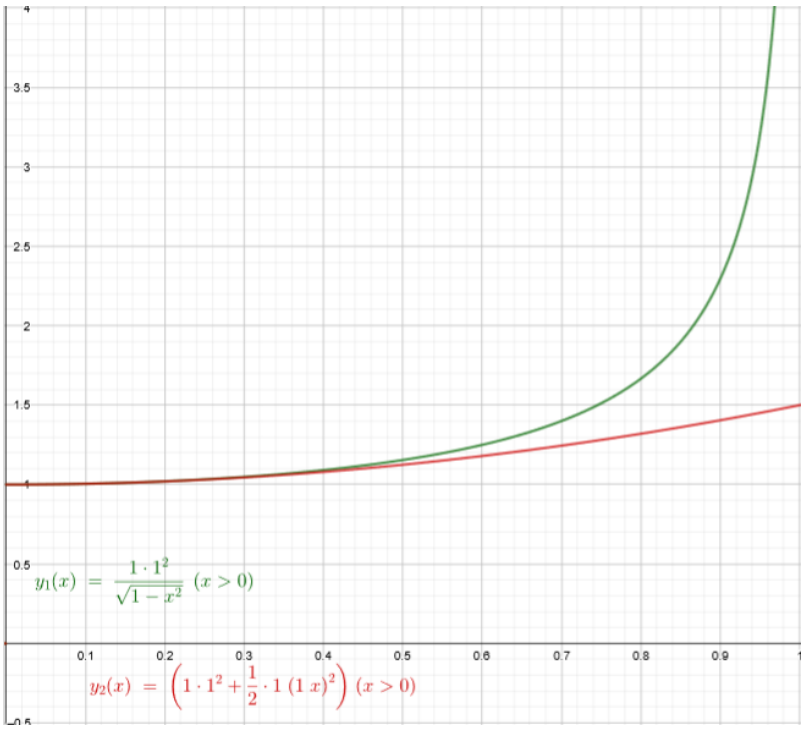

图 13: 相对论能量和经典能量的比较，无量纲化取 $m = 1$，$c = 1$，横轴为 β。

综上所述，将 Mass Spring 的几种处理方法进行相对论性改造，半隐式方法完美可行；隐式方法存在理论上的问题，但因为本来就不够严格，可以根据步骤进行计算；加速方法无法起到加速的作用。可以预见的是，如果场景的相对论性强，则对时间步长要求很高，且容易出现能量爆炸的现象。下面的模拟均采用默认的物体参数:`k = 1000, damping = 0.995, g = -9.8, collision penalty = 10000, collision scale factor = 1.1`。若取时间步长为 0.0001、光速为 100，可以在一段时间内获得尚可的结果，但是抖动很大；若取时间步长为 0.001、光速为 10，中途就出现了错误的解，使得网格的结果消失，参见 `cloth_semi_rela_0_10_0.001.gif`、`cloth_semi_rela_0_100_0.0001.gif`。可以证明相对论能量 $\epsilon = \gamma mc^2$ 严格大于经典的动能加上静止能量 $mc^2 + \frac{1}{2}mv^2$，且相对论能量在速度接近光速时随着速度的增加而急剧增加，如图13所示，这也是物体加速到光速的困难所在。然而上述仿真过程基于位移和速度，当速度与光速数量级接近时，很容易出现能量的急剧增加，需要合适的能量守恒修正。

对半隐式方法，我们直接计算加速度，从而改变速度和位置。设 n 时间步的总能量为 $E_n = \sum \frac{1}{2}k(||\vec{x}_1 - \vec{x}_2|| - L)^2 + \sum(\gamma mc^2 - \vec{f}_{\text{ext}} \cdot \vec{x})$，如果 $n+1$ 时间步的总能量为 E_{n+1}，我们考虑等比地减小计算出的加速度。实际模拟时，$\vec{x}$ 均为有限的量，因为 $\vec{v}$ 有限。近似修正后不改变 $\vec{x}$，能量的增加 ΔE 来源于速度项 $\vec{v}$ 的改变和势能的改变。使用得到的新位置计算出新势能，从而由能量守恒得到应有的动能和动能应该变化的大小 ΔE

$$\Delta E = \sum \Delta(\gamma)mc^2 = \sum \frac{\vec{\beta} \cdot \mathrm{d}\vec{\beta}}{(1-\beta^2)^{\frac{3}{2}}}mc^2 = \sum \frac{\vec{\beta} \cdot \vec{a}\mathrm{d}t}{(1-\beta^2)^{\frac{3}{2}}}mc$$

使用当前计算出的加速度计算上式右侧的结果，与应当变化的大小作比较，即可得到加速度应减小的比例。

这个算法实际上是利用图13的切线进行估计，从曲线中可以看出达到同样的能量时，切线对应的速度结果会比真实值大，且前面取直接计算 $\vec{x}$ 的结果作为势能估计，在很多情况下给动能留下了偏大的空间，能量仍然会不断增长。实际情况也是如此，仍然可能会出现爆炸的现象。考虑到曲线的特性，我们也可以进行割线估计，即计算出下一步的动能，通过线性变换动能差值达到目标值，类似地分析可以得到此时速度是偏小的，结合偏大的动能差值，实际能量与原能量的关系不确定，但是相差不大，达到了稳定能量的目的。注意到我

们计算的能量是所有粒子能量之和，不遵循简单的曲线关系，且各点加速度不一定均使速度增大，能量单调性不能保证，上述方法也不能保证动能一定会减小或者锁定在限定范围内。我们不应简单地进行线性的变化，因为会存在某些特定的情形计算出来的能量差值与目标相差甚远，这可能是爆炸的前兆。

若取可用动能增量和目前动能增量的比值作为比例，且在绝对值大于 1 截断到绝对值为 1 的情形；此时进行半隐式方法的物理仿真，会发现光速较小时，布料在第一次摆动的过程中能量几乎控制不变，效果很好，但是在摆动到最高处时出现了可用动能增量为正而目前动能增量为负的情形：此时原本应当使速度按照加速度计算结果减小，而上述修正过程却使得加速度反向，这与真实的运动是不符的，无论如何减小时间步长都无法阻止爆炸现象，参见 cloth_semi_rela_1_10_0.0001.gif、cloth_semi_rela_1_100_0.0001.gif。故在修正时，我们应该牢记保证能量不增加的目的，分别处理目前动能增量为正、负的情形。最后我们给出的能量修正方法为：对动能减小或不变的情形，不改变计算结果，即使势能要求动能减到更小值；对动能增加的情形，如果势能要求动能应减小，此时令加速度为 0，因为从物理上至少不应该改变方向，如果势能要求动能增加，则按比例处理，不高于 1。这样的处理方法不能严格保证能量不增加（即便前文所述方法也无法完全保证，因为有势能的不确定性和多粒子的叠加），但是实际操作时观察到每次能量的变化不太大，总体而言能量基本守恒。不过，由于半隐式方法自身的不稳定性，在模拟较长时间后仍然会出现爆炸现象，且该现象无法使用上述的能量修正手段加以弥补，因为此刻速度的方向已经不正确，只会使得网格不断向不正确的方向扩张，参见 cloth_semi_rela_2_10_0.0005.gif、cloth_semi_rela_2_100_0.001.gif。想要获得更精确的结果，就需要不断减小时间步长。

对隐式方法，理论上可以使用更大的时间步长，可以类似地进行两种能量修正方法，只需要将其中涉及到的加速度改为 $a = \frac{\Delta\vec{x}/h - \vec{v}}{h}$。在隐式方法中需要求解 $(\mathbf{I} - \vec{\beta}\vec{\beta}^T)\mathbf{He}$，需要注意不能使用 Eigen::MatrixXd 中提供的乘法，因为发现会出现意外的极大的数的结果。最好使用矩阵乘法的定义进行，且因为仅为 3×3 矩阵，不影响运行效率，且保障了结果的正确性。如果不进行能量修正，模拟会出现爆炸现象，表现为矩阵的最小特征值逐步减小直至小于 0，此后方程就无解了，经过正定化后就爆炸了。如果进行切线的能量修正，与不修正结果相近，也会出现明显的爆炸现象。基于割线估计的修正方法效果最好，没有出现爆炸的情况。参见 cloth_imp_rela_0_10_0.001.gif、cloth_imp_rela_1_10_0.001.gif、cloth_imp_rela_2_10_0.001.gif。割线估计的方法效果最好，还可以模拟极低光速的情形，参见 cloth_imp_rela_2_1_0.001.gif。

给出非相对论性模拟的结果，参见 cloth_semi_0.001.gif、cloth_imp_0.001.gif。总体而言，半隐式方法观感上能量更大、变化更剧烈，相对论情形和非相对论情形差异不显著。隐式方法观感上能量稍小、变化较小，光速较小时效应比较明显，即加速较为困难，但此时割线能量修正的算法会使得能量不断减小，最后像是有阻尼力一样。理论上，只要取足够小的时间步长，可以使用割线能量修正的隐式、半隐式算法取得较好的模拟效果。

在这个框架下还可以使用碰撞惩罚力模拟布料与物体的碰撞，参见 cloth_ball_imp_rela_2_10_0.001.gif、cloth_ball_imp_rela_2_10_0.001_2.gif。还可以模拟其他弹簧-质点类型的物体的运动，如模拟一个弹性球与一个刚性球间的碰撞，参见 ball_ball_imp_rela_2_10_0.001.gif。

修正方法可以做到更加精细。切线修正和割线修正的方法仅分别使用切线和割线作近似，实际上我们可以以需要衰减的比例作为参数构成一函数，问题即求解函数达到指定能量差值时的衰减比例。这转化为一求零点问题，可以使用二分法、牛顿迭代法等处理。需要注意，因为具有多粒子，该函数的形状和单调性都不确定，使用前述方法需要注意确保结果落在合理的范围，可以取切线修正和割线修正的结果为范围。

原则上我们可以利用狭义相对论动力学方程进行更多系统的物理仿真，包括流体仿真等。考虑到狭义相对论能量与速度的关系，在光速较小时需要极小的时间步长才能实现较好的模拟，且需要增加适当的能量守恒修正算法。针对低光速情形的物理仿真，为了平衡时间和效果，应考虑从能量为自变量的角度入手发展新的算法。

渲染的取舍与完善

当光速低至能够与物体的运动速度相比时，观察者视角中会因为光线传播需要时间而畸变的不仅有物体的几何，还有物体的阴影。上文已经提到，畸变后物体的几何可以通过牛顿迭代求解函数零点以得到一个物理正确的结果，但对于阴影，几何的变化就要复杂得多。要决定一个片元是否为一个光源的阴影，需要确认光线从光源传播至片元所在位置的过程中是否被运动物体遮挡。一个简单的想法是，对光源应用和观察者相类似的时间推迟顶点变换后再计算出对应的 Shadow Map。但是，该方法的不足之处在于，光线从片元所在位置传播至观察者也是需要时间的。这意味着每个片元都需要获取其对应 t^* 时刻的 Shadow Map 以决定其在此时是否为阴影，而这对 Shader 来说仍然是不现实的。

考虑到该项目只是作为相对论视觉仿真的演示，而不追求精确的阴影，我们对阴影的处理做出了取舍，即不计算基于光源的阴影，而只使用 Bling-Phong 光照和 SSAO 结合渲染光影效果。为了让这样的渲染效果更为可信，本项目演示场景仅采用了一个位置较高的点光源，使环境光遮蔽成为场景阴影的主要构成，最终得到了较为美观的结果。

为了得到更好的视觉效果，本项目还对 Render Nodes 的节点体系做了进一步完善。考虑到可能的后处理需要，我们新增了 Post Process 节点，将 G-Buffer 中的信息与四个自定义参数传入可自由编写的 Fragment Shader 中。该节点是原 SSAO 节点的进一步扩展，在能够兼容 SSAO Shader 的基础上还可以调用其他 Shader 实现更多后处理效果。例如，我们编写了双边滤波（Bilateral Filter）Shader 以对 SSAO 生成的结果进行降噪；测试场景的天空盒同样也是在后处理过程经过的 `relativity_background.fs` 中计算得出。画面模糊、色调偏移、Bloom 等常见的后处理效果也可以在这一框架下方便地实现，但并未在该项目中得到应用。

我们还编写了 Mix Color 后处理节点。该节点通过将两张图像与一个 Alpha 参数传入 Fragment Shader，简单地实现了图像混合模式的自定义，以借此实现多种效果。例如，SSAO 节点不需传入原图像并输出加入 SSAO 后的图像，而可以仅读入 G-Buffer、输出纯 AO 结果，再与 Deferred Lighting 输出的图像以正片叠底模式合并。这一新的流程使得纯 AO 结果可以在经过双边滤波降噪后再应用于原图像。除此之外，将相对论光栅化输出的结果与传统光栅化输出的结果以透明度模式混合，可以在同一屏幕中比较物体的位置与相对论变换后的位置，效果甚佳。

考虑到主测试场景中所有的物体均以棋盘格作为纹理，我们将纹理过滤模式切换为了最邻近插值，避免了棋盘格纹理由于线性插值出现的模糊现象，得到了尖锐的边缘，在视觉效果上改善了许多。

一个完整的渲染节点图如图14。

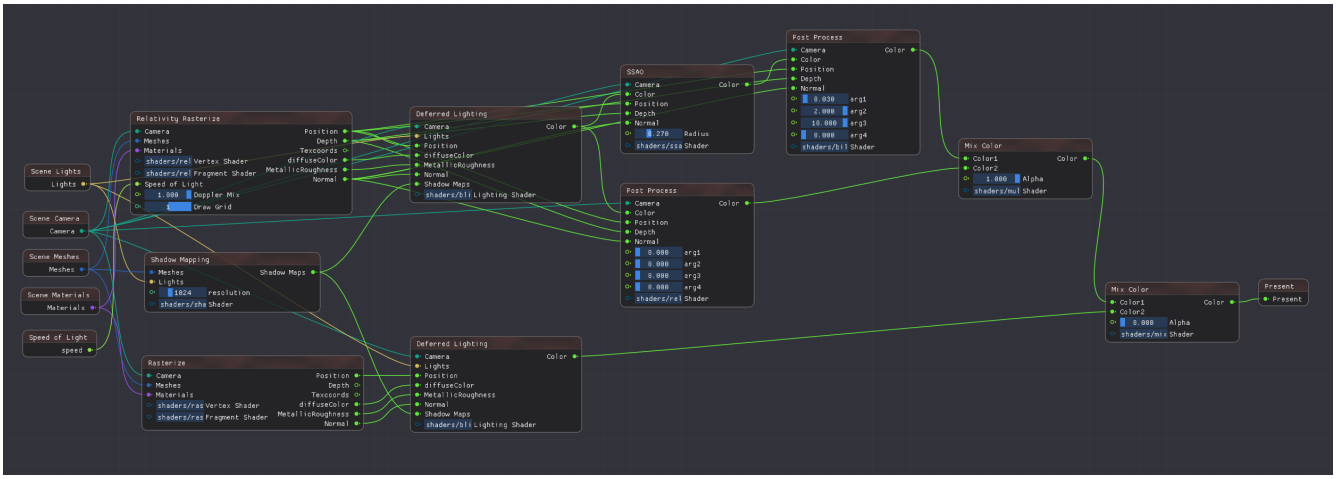

图 14: 完整的渲染节点图

场景布置

本项目的主要测试场景灵感很大程度上来源于视频“爱因斯坦未能正确理解的相对论视觉效应”(`https://www.bilibili.com/video/BV1JY411L7xm`)。受该视频启发，本项目的测试场景中物体均采用了棋盘格纹理，以便于观察相对论效应下物体几何的畸变。同时，我们也用 Blender 制作了与该视频类似的运动方块、旋转球体模型及其他更多模型，以测试物体的相对论畸变与多普勒效应。

为了提供良好的位置参考，测试场景中布置了 $5\text{m} \times 5\text{m}$ 大小的棋盘格地板。同时，地板也会参与相对论畸变的计算，以避免穿模现象。除此之外，相对论光栅化节点中还会调用 OpenGL 接口绘制一组不受相对论变换影响的地面网格，以更清晰地观察相对论几何畸变的程度。

该项目的实现中相对论畸变均是对顶点的变换，因此顶点较为稀疏的网格将无法正确得到“扭曲”的效果，在与其他物体交互时还会出现穿模的现象。这意味着测试场景中所有的曲面都要得到充分的细分。主测试场景的地面、立方体均是预先细分后的高模；本项目还用 Blender 对 `Sponza.usda` 模型中的墙壁、地板进行了进一步细分得到了 `Sponza_subdivided.usda` 模型，以满足几何变换的需要。

为了避免场景过于单调，本项目用后处理的方式为场景添加了一个渐变的天空盒。当场景中物体较远时，物体的颜色会与天空盒进行混合以达到雾效，这使得地面网格的视觉效果也更加自然。

五　项目总结

在本项目中，我们成功地实现了：

(1) 静态场景的相对论视觉效应模拟；

(2) 动态场景推迟方程求解和相对论视觉效应模拟，且可以达到近乎实时的效果，形成了完整的相对论视觉模拟方法；

(3) 上帝视角的模拟；

(4) 以质点弹簧为例的相对论动力学物理仿真。

本项目仍有许多不足之处，如：

(1) 相机参考系是“瞬时共动”参考系，如果物体加速度很大，那么狭义相对论就不适用了；

(2) 没能够区分相机的固有时和默认系时间；

(3) 不能处理物体产生和消失的情况，因为要涉及 Mesh 拓扑的改变；

(4) 光栅化结果受到 Mesh 精度的影响，无法实现高精度的变换；

(5) 多普勒频移带来的颜色变化在物理上不准确，仅仅是简单的线性近似的结果；

(6) 上帝视角的模拟流程不够友善，需要额外的操作；

(7) 质点弹簧系统物理仿真中对能量守恒的修正不够精确，不能保证能量变化在合理范围内。

通过本项目的实践，我们对狭义相对论有了更加深入的理解，对狭义相对论下的视觉效应有了更加直观的感受。同时，我们对计算机图形学课程中许多学到的内容有了进一步的巩固和应用，对节点编程渲染框架底层运作的原理有了大致的认识，对一个大型工程的程序运行过程更加清晰。在项目的实现过程中，我们也遇到了许多困难，如对于框架的运行的理解、上帝视角的理论困难、物理仿真中的能量守恒问题、渲染的取舍处理等。解决这些问题的过程中，我们开拓了思路，学到了更多知识。

六　致谢

感谢计算机图形学课程的沈鹏飞、李喆昊、吴汶政三位助教提供了本项目的框架支持，并为对框架的修改调整提供了悉心指导。尤其感谢沈鹏飞助教线下协助指导实现获取历史节点位置的功能。

感谢 MIT Game Lab 制作的游戏 A Slower Speed of Light 为物体的几何畸变与多普勒效应的期望效果提供了参考；感谢 B 站 Up 主“两颗熟李子”制作的相对论视觉效应系列视频，为测试场景的设计与期望的展示效果提供了灵感参考。

七　附录

代码文件见 `\Framework3D\`。

演示视频见 `\videos\`。

物理仿真结果见 `\results\`。

使用的模型和纹理见 `\models\`、`\textures\`。