作业 8 质点弹簧系统
GAMES101, 2020 年春季

教授: 闫令琪

计算机图形学与混合现实研讨会

GAMES：Graphics And Mixed Environment Seminar

发布日期为北京时间 2020 年 4 月 24 日 (星期五) 晚上10：00

截止日期为北京时间 2020 年 5 月 01 日 (星期五) 晚上10：00

注意:

- 任何更新或更正都将发布在论坛上，因此请偶尔检查一下。
- 论坛链接：http://games-cn.org/forums/forum/graphics-intro/。
- 你必须独立完成自己的作业。
- 你可以在论坛上发布帖子求助，但是发布问题之前，请仔细阅读本文档。
- 在截止时间之前将你的作业提交到 SmartChair 上。

---

1 总览

1.1 连接绳子的约束

在 `rope.cpp` 中, 实现 Rope 类的构造函数。这个构造函数应该可以创建一个新的绳子 (Rope) 对象，该对象从 start 开始，end 结束，包含 `num_nodes` 个节点。也就是如下图所示：

```
Rope(start, end, 3, ...)

  0------------0-------------0
  ^                         ^
start                      end
```

每个结点都有**质量**，称为质点；质点之间的线段是一个**弹簧**。通过创建一系列的质点和弹簧，你就可以创建一个像弹簧一样运动的物体。

pinned_nodes 设置结点的索引。这些索引对应结点的固定属性 (pinned attribute) 应该设置为真（他们是静止的）。对于每一个结点，你应该构造一个 **Mass** 对象，并在 **Mass** 对象的构造函数里设置质量和固定属性。（请仔细阅读代码，确定传递给构造函数的参数）。你应该在连续的两个结点之间创建一个弹簧，设置弹簧两端的结点索引和弹簧系数 **k**，请检查构造函数的签名以确定传入的参数。

运行 `./ropesim`。你应该可以看到屏幕上画出绳子，但它不发生运动。

1.2 显式/半隐式欧拉法

胡克定律表示弹簧连接的两个质点之间的力和他们之间的距离成比例。也就是：

$$\mathbf{f_{b\to a}} = -k_s \frac{\mathbf{b}-\mathbf{a}}{||\mathbf{b}-\mathbf{a}||}(||\mathbf{b}-\mathbf{a}|| - l)$$

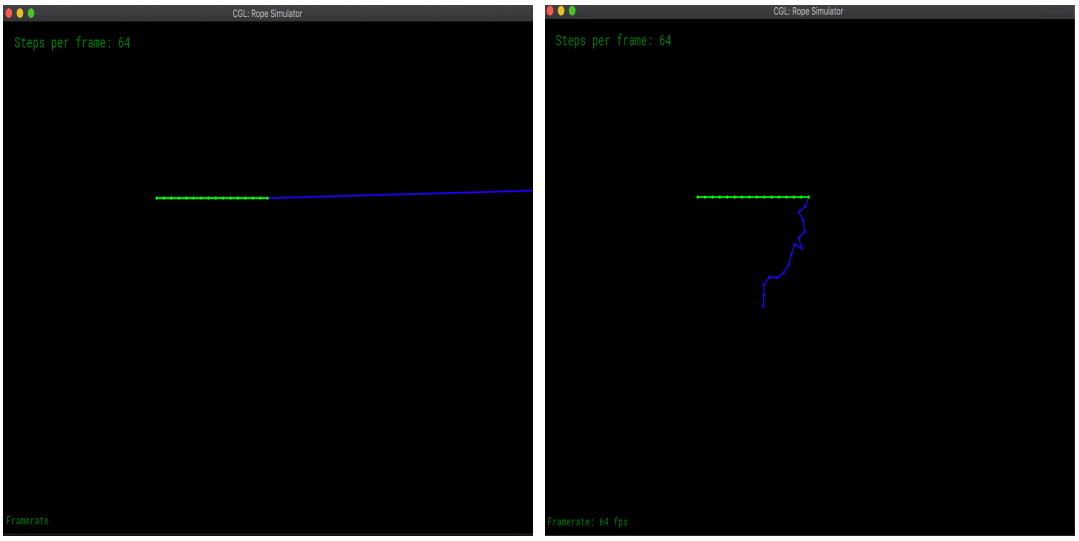
在 `Rope::simulateEuler` 中, 首先实现胡克定律。遍历所有的弹簧，对弹簧两端的质点施加正确的弹簧力。保证力的方向是正确的！对每个质点，累加所有的弹簧力。

一旦计算出所有的弹簧力，对每个质点应用物理定律：

```
F=ma
v(t+1) = v(t) + a(t) * dt
x(t+1) = x(t) + v(t) * dt   // For explicit method
x(t+1) = x(t) + v(t+1) * dt // For semi-implicit method
```

运行./ropesim。仿真应该就开始运行了，但是只有 3 个结点，看起来不够多。在 application.cpp 文件的最上方，你应该可以看到欧拉绳子和 Verlet 绳子的定义。改变两个绳子结点个数（默认为 3 个），比如 16 或者更多。

运行 ./ropesim -s 32 来设置仿真中每帧不同的仿真步数。尝试设置较小的值和较大的值（默认值为 64）。



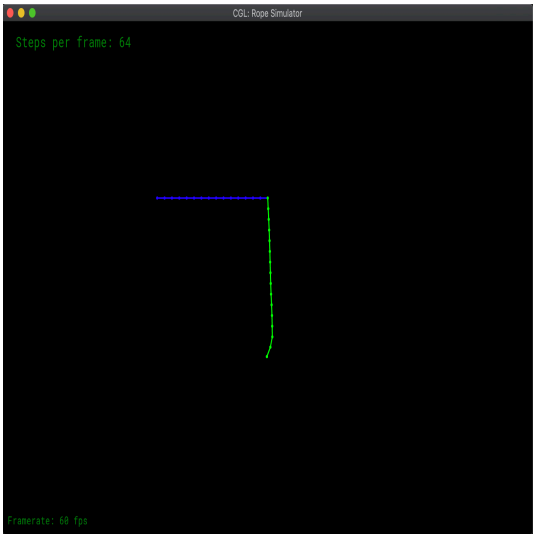
1.3 显式 Verlet

Verlet 是另一种精确求解所有约束的方法。这种方法的优点是只处理仿真中顶点的位置并且保证四阶精度。和欧拉法不同，Verlet 积分按如下的方式来更新下一步位置：

```
x(t+1) = x(t) + [x(t) - x(t-1)] + a(t)*dt*dt
```

除此之外，我们可以仿真弹簧系数无限大的弹簧。不用再考虑弹簧力，而是用解约束的方法来更新质点位置：只要简单的移动每个质点的位置使得弹簧的长度保

持原长。修正向量应该和两个质点之间的位移成比例，方向为一个质点指向另一质点。每个质点应该移动位移的一半。

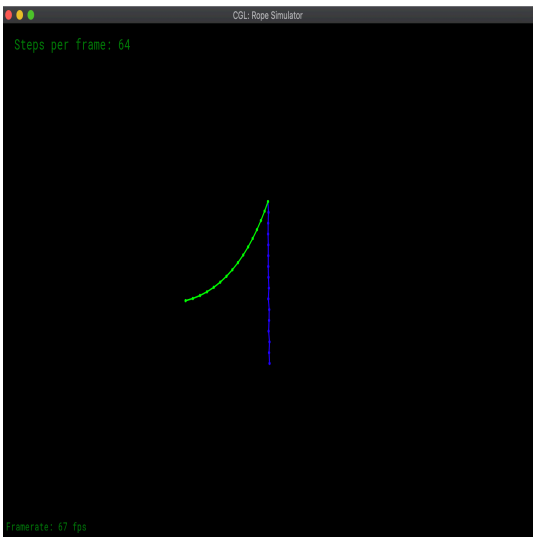


只要对每个弹簧执行这样的操作，我们就可以得到稳定的仿真。为了使运动更加平滑，每一帧可能需要更多的仿真次数。

1.4 阻尼

向显示 Verlet 方法积分的胡克定律中加入阻尼。现实中的弹簧不会永远跳动-因为动能会因摩擦而减小。阻尼系数设置为 0.00005, 加入阻尼之后质点位置更新如下：

```
x(t+1) = x(t) + (1 - damping_factor) * [x(t) - x(t-1)] + a(t) * dt * dt
```



1.5

你应该修改的函数是:

- rope.cpp 中的 `Rope::rope(...)`
- rope.cpp 中的 `void Rope::simulateEuler(...)`
- rope.cpp 中的 `void Rope::simulateVerlet(...)`

2 开始编写

2.0.1 依赖

本次作业需要预先安装 OpenGL, Freetype 还有 RandR 这三个库。可以通过以下命令进行安装：

```
$ sudo apt install libglu1-mesa-dev freeglut3-dev \\
        mesa-common-dev
$ sudo apt install xorg-dev #会自动安装 libfreetype6-dev
```

2.0.2 开始吧

请下载工程的代码框架并通过下面的命令创建工程：

```
$ mkdir build
$ cd build
$ cmake ..
$ make
```

之后，你应该可以使用命令**./ropesim** 来运行仿真。

3 评分和提交

评分:

- [5 分] 提交的格式正确，包含所有必须的文件。代码可以编译和运行。
- [5 分] 连接绳子约束，正确的构造绳子
- [5 分] 半隐式欧拉法
- [5 分] 显式欧拉法
- [10 分] 显式 Verlet
- [5 分] 阻尼
- [-2 分]惩罚分数:
  未删除 /build, /.vscode 和 assignment8.pdf。
  未按格式建立 /images，缺少结果图片。
  未提交或未按要求完成 README.md。
  代码相关文件和 README 文件不在你提交的文件夹下的第一层。

提交:

- 当你完成作业后，请清理你的项目，记得在你的文件夹中包含 CMakeLists.txt 和所有的程序文件 (无论是否修改);

- 提交的代码中应该包含带有阻尼的半隐式欧拉法和显式 Verlet，并且将显式欧拉法注释掉。
- 再添加一个 README.md 文件写清楚自己完成了上述得分点中的哪几点 (如果完成了，也请同时提交一份结果图片)，并简要描述你在各个函数中实现的功能;
- 最后,将上述内容打包,并用“姓名 Homework8.zip”的命名方式提交到 SmartChair 平台。

平台链接：http://smartchair.org/GAMES2020Course-YLQ。