王蕙钰 2100018733

可视计算与交互概论 Lab4 报告

2024.12.9

Task 1

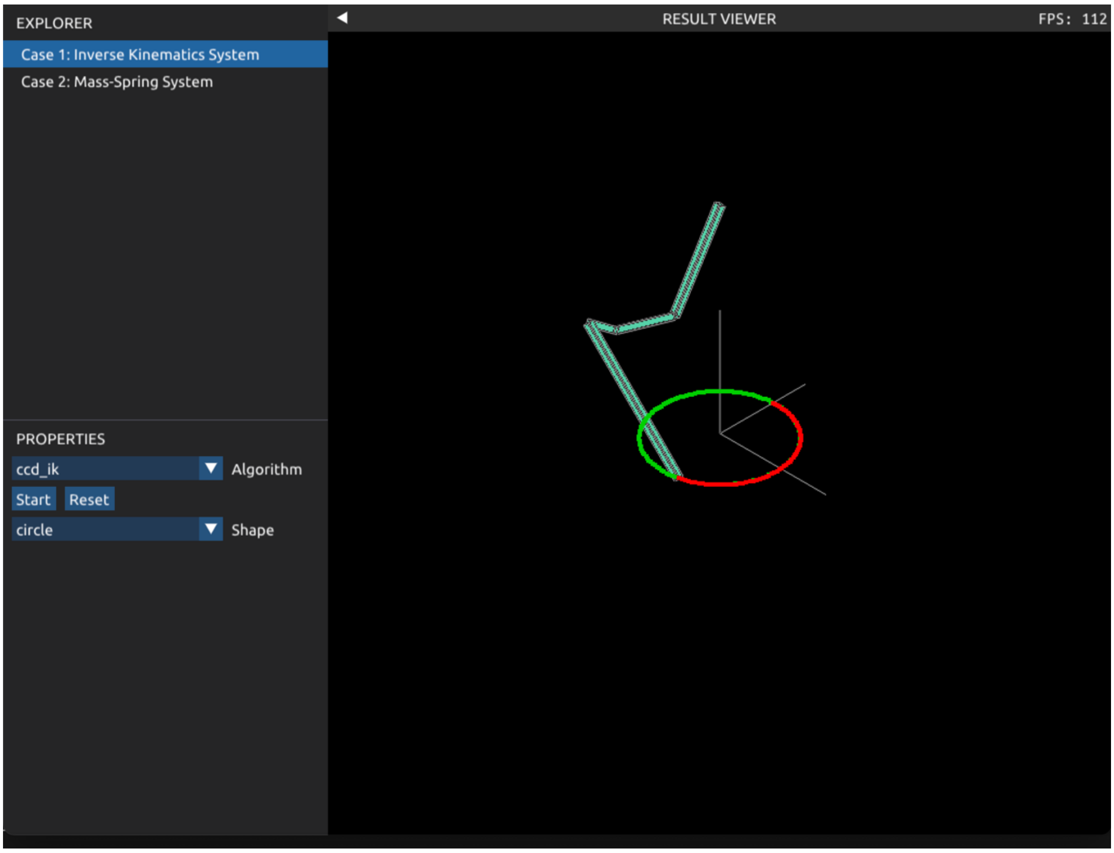
问题：

1. 如果目标位置太远，无法到达，IK 结果会怎样？
   会朝着目标位置方向伸到最远
2. 比较 CCD IK 和 FABR IK 所需要的迭代次数。
   ```
   CCDIKIteration: 100
   CCDIKIteration: 100
   CCDIKIteration: 100
   CCDIKIteration: 100
   CCDIKIteration: 100
   CCDIKIteration: 100
   CCDIKIteration: 100
   IKIteration: 3
   IKIteration: 3
   IKIteration: 3
   IKIteration: 3
   IKIteration: 3
   ```
   输出画圆时两种方法所需要的循环次数，发现 CCD 每次都达到了上限 100 次，而 FABR 小于 5 次就实现收敛，FABR 所需迭代次数要少得多。
3. （选做，只提供大概想法即可）由于 IK 是多解问题，在个别情况下，会出现前后两帧关节旋转抖动的情况。怎样避免或是缓解这种情况？
   可以通过对关节角度变化的速度进行限制，避免每帧之间的关节旋转过快产生突变。

实现思路：

1. 前向运动学
   根据：（1）GlobalRotation(child) = GlobalRotation(parent) * LocalRotation(child)
   （2）GlobalPosition(child) = GlobalPosition(parent) + GlobalRotation(parent) * LocalOffset(child)
   依次计算即可。
2. CCD
   每次循环都从最后一个关节开始遍历，每一次移动一个关节，目标是让现在关节、最后一个关节 endEffector、目标位置 endPosition 共线，直到二者足够接近。
   遍历中对于每个关节都计算现在关节到最后一个关节的向量和现在关节到目标位置的向量->计算所需旋转->更新该关节的旋转：新增旋转*原旋转量。

效果：



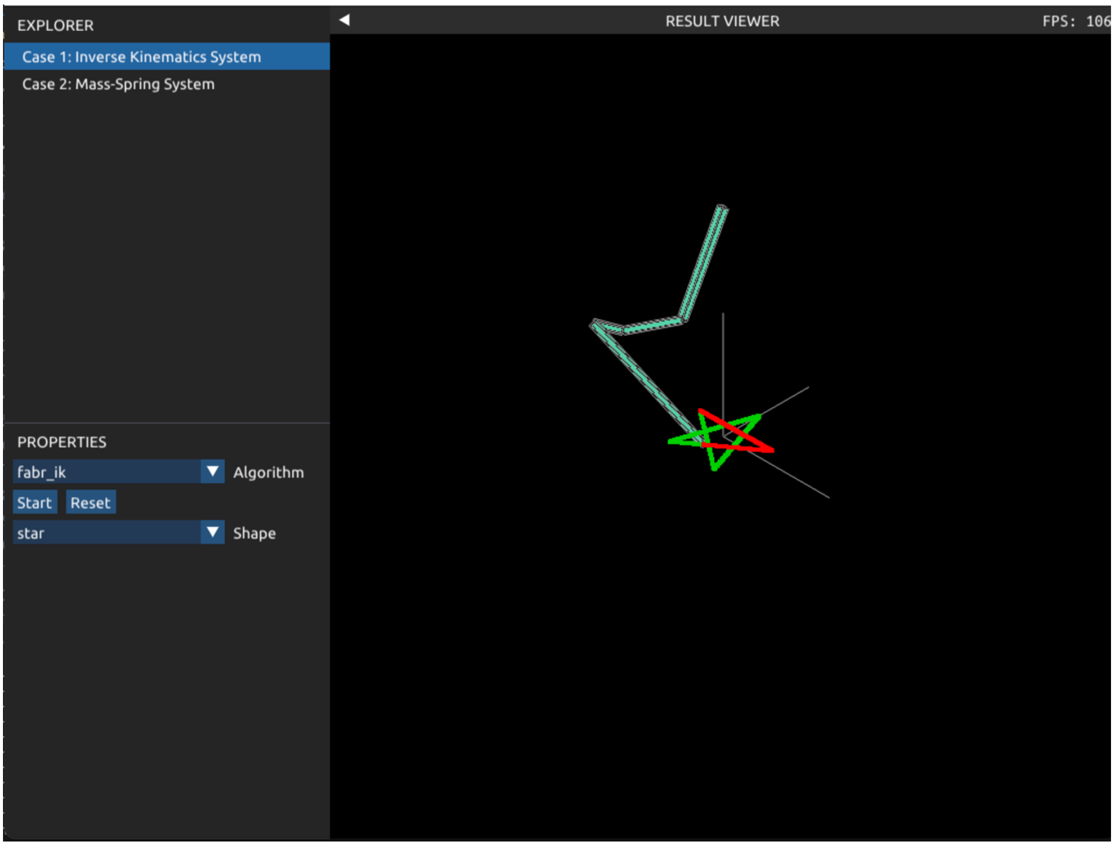
3. FABR

每次循环都包含：从后往前遍历和从前往后遍历关节。

从后往前遍历：将最后一个关节移动到目标位置；从倒数第二个关节开始，第 i 个关节移动到连接其原位置和第 i+1 个关节所在位置的直线上，距离第 i+1 个关节的长度为骨骼长度。因此每次只要计算方向和偏移量即可更新坐标。

从前往后遍历：将 root 移动回固定位置，从第二个关节开始移动，同理第 i 个关节移动到连接其原位置和第 i-1 个关节所在位置的直线上，距离第 i-1 个关节的长度为骨骼长度。每次只要计算方向和偏移量即可更新坐标。

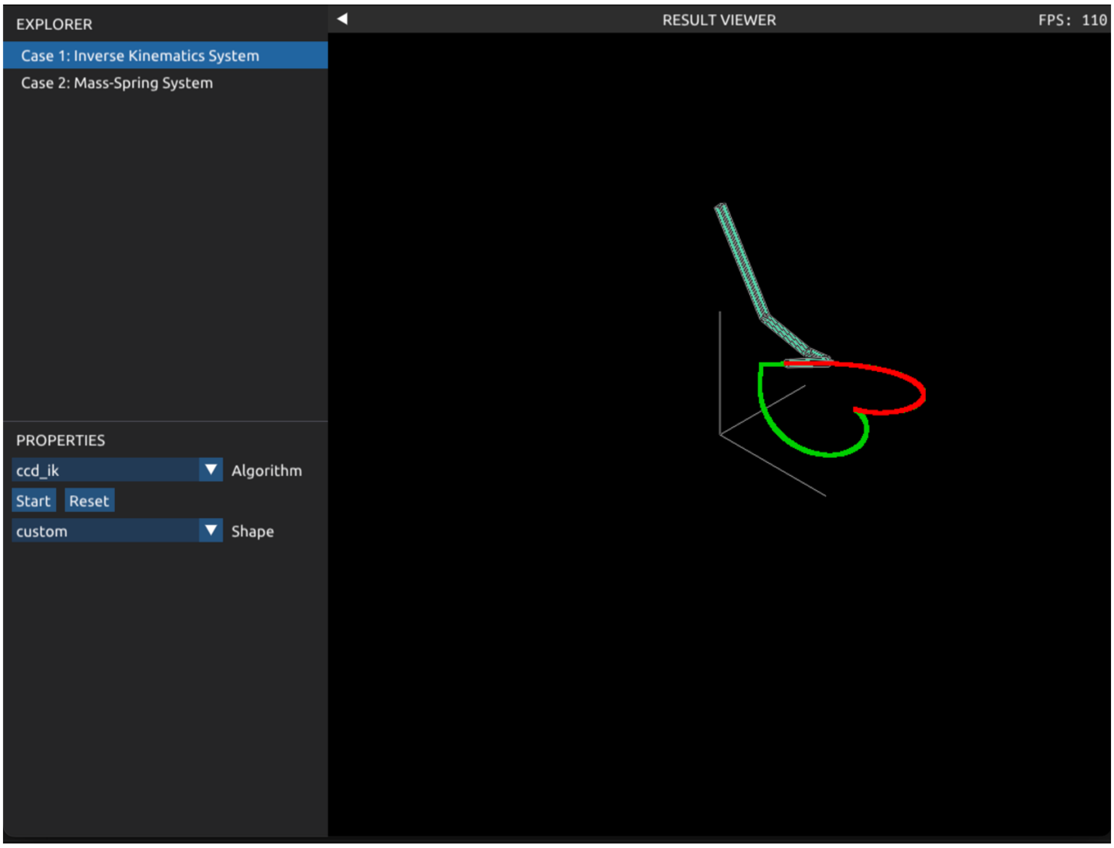
效果：



4. 绘制自定义曲线

   绘制心形曲线，函数来自 [https://mathworld.wolfram.com/HeartCurve.html 中第 7 个心形曲线](https://mathworld.wolfram.com/HeartCurve.html)。

   效果：



Task 2:

隐式欧拉:

- Implicit Euler:
  - $\mathbf{x}(t_{n+1}) = \mathbf{x}(t_n) + \underline{\mathbf{v}(t_{n+1})\Delta t}$
  - $\mathbf{v}(t_{n+1}) = \mathbf{v}(t_n) + \underline{\frac{1}{m}\mathbf{f}(t_{n+1})\Delta t}$

没有更准确

根据讲义推导即求解如下问题:

它等价于求解如下的优化问题 (读者可以利用目标函数关于 $\mathbf{x}$ 的梯度等于 $\mathbf{0}$ 证明等价性):

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \underbrace{\frac{1}{2h^2}\|\mathbf{x} - \mathbf{y}^k\|_{\mathrm{M}}^2}_{\text{惯性}} + \underbrace{E(\mathbf{x})}_{\text{弹性}} \tag{17.19}$$

PPT 上给出了使用牛顿法求解的步骤:

Algorithm 2: Newton Solver with Backtracking Line Search

$\mathbf{x}^{(1)} := \mathbf{y};$

$g(\mathbf{x}^{(1)}) := \mathtt{evalObjective}(\mathbf{x}^{(1)})$

for $k = 1, \ldots, \mathrm{numIterations}$ **do**

$\quad \nabla g(\mathbf{x}^{(k)}) := \mathtt{evalGradient}(\mathbf{x}^{(k)})$

$\quad \nabla^2 g(\mathbf{x}^{(k)}) := \mathtt{evalHessian}(\mathbf{x}^{(k)})$

$\quad \delta\mathbf{x}^{(k)} := -\nabla^2 g(\mathbf{x}^{(k)})^{-1} \nabla g(\mathbf{x}^{(k)})$

$\quad \alpha := 1/\beta$

$\quad$ **repeat**

$\quad\quad \alpha := \beta\alpha$

$\quad\quad \mathbf{x}^{(k+1)} := \mathbf{x}^{(k)} + \alpha\delta\mathbf{x}^{(k)}$

$\quad\quad g(\mathbf{x}^{(k+1)}) := \mathtt{evalObjective}(\mathbf{x}^{(k+1)})$

$\quad$ **until** $g(\mathbf{x}^{(k+1)}) \leq g(\mathbf{x}^{(k)}) + \gamma\alpha\left(\nabla g(\mathbf{x}^{(k)})\right)^{\mathsf{T}} \delta\mathbf{x}^{(k)};$

end

29 / 61

正如讲义上所说，“对于本节中的弹簧质点系统，g（x）性质足够好，认为只需要一步牛顿法即可求得最小值点”。即求解如下方程：

$$\mathbf{H}_g(\mathbf{x}^k)(\mathbf{x}^{k+1} - \mathbf{x}^k) = -\nabla g(\mathbf{x}^k) \tag{17.23}$$

其中 g(x)和 Hg（x）定义如下：

$$\begin{aligned} \nabla g(\mathbf{x}^k) &= \frac{1}{h^2}\mathbf{M}(\mathbf{x}^k - \mathbf{y}^k) + \nabla E(\mathbf{x}^k) \\ \mathbf{H}_g(\mathbf{x}^k) &= \frac{1}{h^2}\mathbf{M} + \mathbf{H}(\mathbf{x}^k) \end{aligned} \tag{17.24}$$

具体步骤：

1. 参考显示欧拉写法，定义时间步 h；在循环中更新每一步状态
2. 构造此时刻 X0，V0 矩阵
3. 参考牛顿法，定义 y

记 $\mathbf{y}^k = \mathbf{x}^k + h\mathbf{v}^k + h^2\mathbf{M}^{-1}\mathbf{f}_{\mathrm{ext}}$，

4. 求 Hg（x0）：根据定义分为 2 个部分。

a) 第一部分：M 是质量的对角阵。依次写入即可。

b) E（x0）的海瑟矩阵，根据讲义计算如下

$$
\begin{aligned}
\mathbf{H}_e &:= \frac{\partial^2 E_{ij}(\mathbf{x}^k)}{\partial \mathbf{x}_i^2} = k_{ij}\frac{(\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^\top}{\|\mathbf{x}_i-\mathbf{x}_j\|^2} + k_{ij}\left(1-\frac{l_{ij}}{\|\mathbf{x}_i-\mathbf{x}_j\|}\right)\left(\mathbf{I}-\frac{(\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^\top}{\|\mathbf{x}_i-\mathbf{x}_j\|^2}\right) \\
&\frac{\partial^2 E_{ij}(\mathbf{x}^k)}{\partial \mathbf{x}_i \partial \mathbf{x}_j} = -\mathbf{H}_e \\
&\frac{\partial^2 E_{ij}(\mathbf{x}^k)}{\partial \mathbf{x}_j^2} = \mathbf{H}_e
\end{aligned} \tag{17.27}
$$

那么这个弹簧关于所有质点坐标的海瑟矩阵可以写成如下形式：

$$
\mathbf{H}_{ij}(\mathbf{x}^k) = \begin{bmatrix} & \vdots & & \vdots & \\ \cdots & \mathbf{H}_e & \cdots & -\mathbf{H}_e & \cdots \\ & \vdots & & \vdots & \\ \cdots & -\mathbf{H}_e & \cdots & \mathbf{H}_e & \cdots \\ & \vdots & & \vdots & \end{bmatrix} \tag{17.28}
$$

将 $\mathbf{H}_{ij}(\mathbf{x}^k)$ 划分成 $n \times n$ 个 3×3 的块，则第 i 行第 i 列与第 j 行第 j 列的块为 $\mathbf{H}_e$，而第 i 行第 j 列与第 j 行第 i 列的块为 $-\mathbf{H}_e$，其余块均为零矩阵．总能量的海瑟矩阵即为所有

具体计算参考下图：

- $H_{ii} = k(1-\ell_0/\ell)$，因 $\ell = \ell_0 = 1$，仅耦合项有贡献。
- $H_{ij} = k \cdot \frac{\ell_0}{\ell^3} \cdot x_{01,i} \cdot x_{01,j}$。

5. 求 g（x0）的梯度，参考上文定义分为两项
   a) 第一项直接计算即可
   b) 第二项计算弹性势能的梯度，参考下

- **公式：**

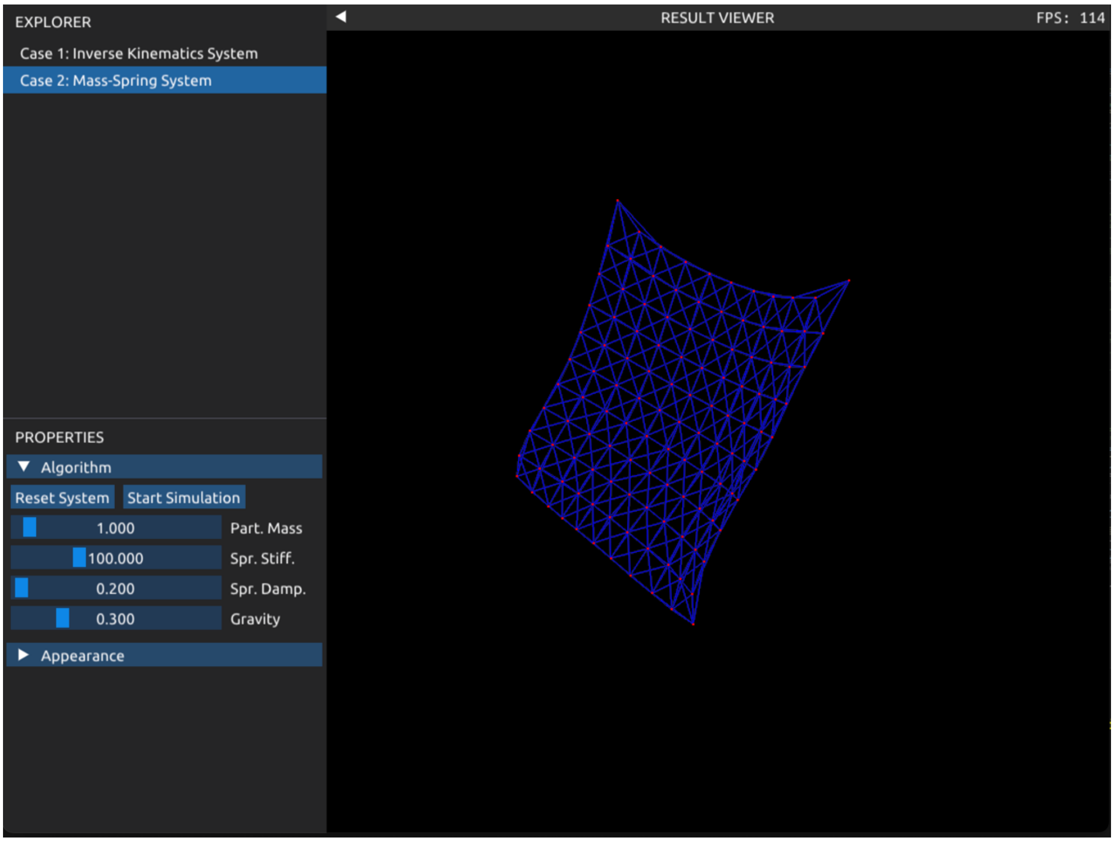
$$\mathbf{f}_{\text{spring}} = k \cdot (\ell - \ell_0) \cdot \mathbf{e}$$

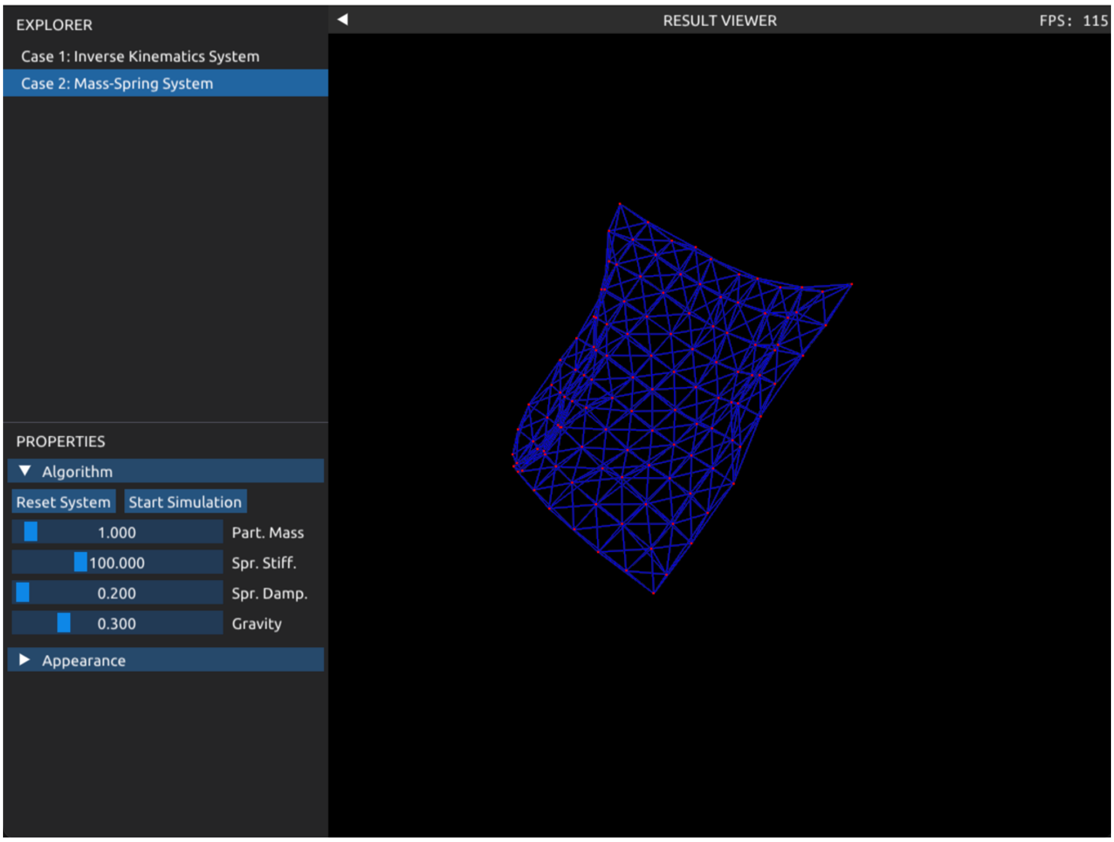
其中：

- k 是弹簧刚度。
- $\ell = \|\mathbf{x}_{p1} - \mathbf{x}_{p0}\|$ 是弹簧当前长度。
- ℓ_0 是弹簧静止长度。
- $\mathbf{e} = \frac{\mathbf{x}_{p1}-\mathbf{x}_{p0}}{\ell}$ 是弹簧方向的单位向量。

6. 解方程 Hg(x0) * (x1 – x0) = - g(x0)的梯度，使用提供的函数求解
7. 将数据从 Eigen 转换回 glm 并输出 position 和 velocity。

效果如下：


EXPLORER
Case 1: Inverse Kinematics System
Case 2: Mass-Spring System
PROPERTIES
Algorithm
Reset System
Start Simulation
1.000
Part. Mass
100.000
Spr. Stiff.
0.200
Spr. Damp.
0.300
Gravity
Appearance
RESULT VIEWER
FPS: 114


EXPLORER
Case 1: Inverse Kinematics System
Case 2: Mass-Spring System
PROPERTIES
Algorithm
Reset System
Start Simulation
1.000
Part. Mass
100.000
Spr. Stiff.
0.200
Spr. Damp.
0.300
Gravity
Appearance
RESULT VIEWER
FPS: 115