2023학년도 1학기

컴퓨터애니메이션실습

컴퓨터 애니메이션

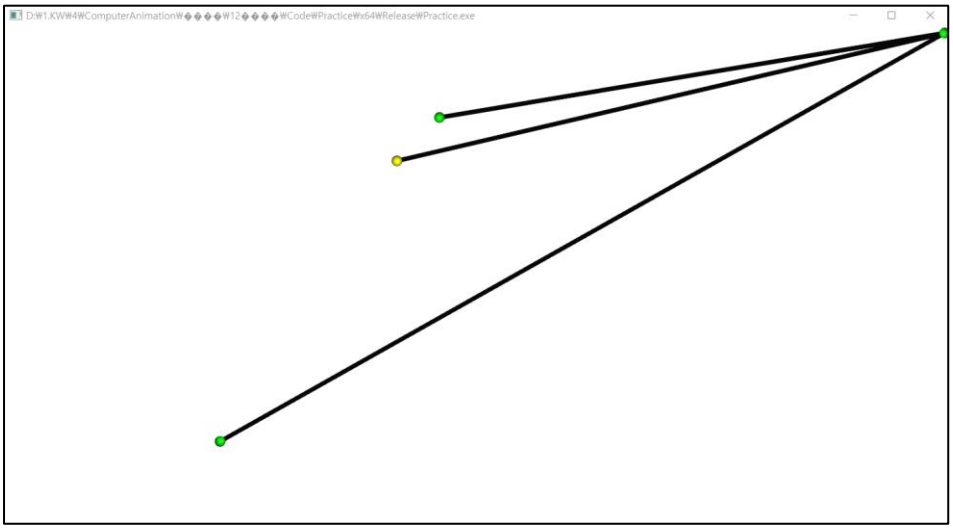
실습 보고서

Self-Scoring Table

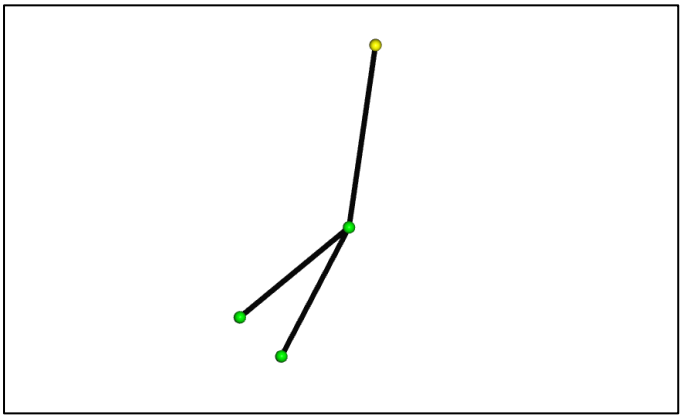
	P1	P2	E1
Score	1	1	1

정보융합학부 2018204058 김민교

P1 - Time Integration with Euler's method

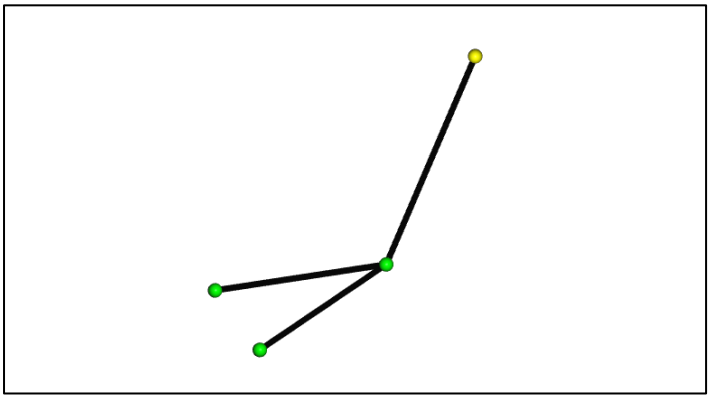


Euler's method로 `N_SUBSTEPS` 값을 1로 주고 실행한 결과이다. 시스템이 엉망이 된 것을 볼 수 있다.

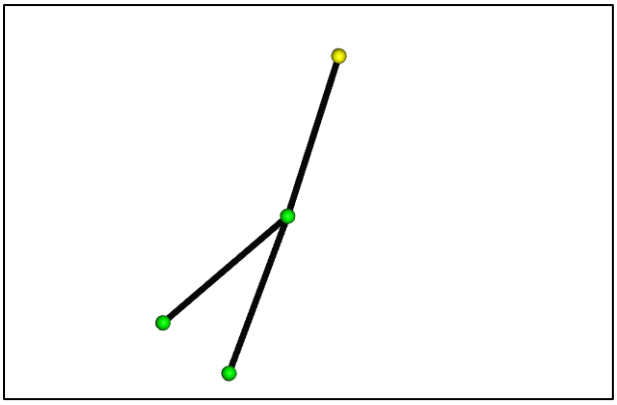
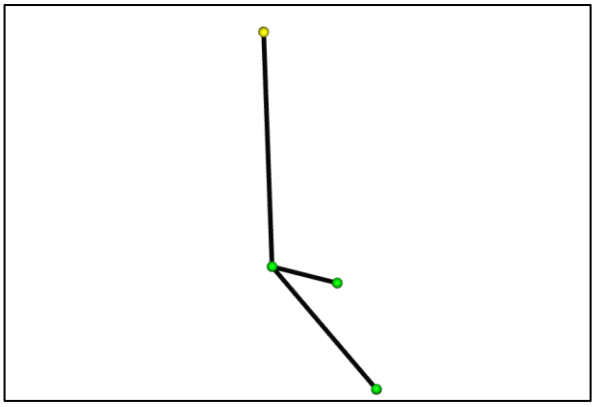


Euler's method로 `N_SUBSTEPS` 값을 9로 주고 실행한 결과이다. 1을 주었을 때 보다 훨씬 나은 모습이다.

P2 - Time integration with the Modified Euler's method



Modified Euler's method로 `N_SUBSTEPS` 값을 1로 주고 실행한 결과이다. Euler's method를 사용했을 때 보다 훨씬 나은 모습이다.

왼쪽은 spring constant를 8까지 늘린 경우이고, 오른쪽은 spring constant를 0.1로 줄인 경우이다. spring constant를 늘리면 spring이 거의 늘어나지 않는다

사진상으로 차이가 분명히 나는 게 드러나지 않아 첨부하지 않았지만, damping coefficient를 늘리니 아주 천천히 떨어졌다. 이는 우리가 원하는 결과가 아니다. point에 damping이 일어나는 게 아니고 spring에 damping이 일어나야 한다.

E1 - Point damping vs damped springs

point damping을 할 때에는, 중력을 받을 때에도 느려졌지만, damped spring을 적용하니 중력과 상관없이 스프링 힘에만 적용이 됐다. 중력은 이제 damping 계수와 상관없이 똑같이 작용한다.

$$\frac{d\mathbf{v}_i(t)}{dt} = \frac{\mathbf{f}_i(t) - \sum_{j \in N_i} c_{ij} \left(\mathbf{v}_{ij}(t) \cdot \mathbf{d}_{ij}(t) \right) \mathbf{d}_{ij}(t)}{m_i}$$

[구현 코드]

```
279     if (usePointDamping)
280     {
281         // Damping spring
282         for (int i = 0; i < nEdges; i++)
283         {
284             Vector3f x_ij = x[e1[i]] - x[e2[i]];
285             Vector3f n_ij = x_ij/x_ij.norm();              // Current length
286             Vector3f v_ij = v[e1[i]] - v[e2[i]];
287 
288             Vector3f x_ji = x[e2[i]] - x[e1[i]];
289             Vector3f n_ji = x_ji / x_ji.norm();            // Current length
290             Vector3f v_ji = v[e2[i]] - v[e1[i]];
291 
292             Vector3f f_damped_i = damping * v_ij.dot(n_ij) * n_ij;
293             Vector3f f_damped_j = damping * v_ji.dot(n_ji) * n_ji;
294 
295             f[e1[i]] -= f_damped_i;
296             f[e2[i]] -= f_damped_j;
297         }
298     }
```

총 힘에서 이 damped spring force를 빼준다.