

3. 뒤는 공을 이해하라

목차

1. 평면과의 충돌
(COLLISIONS WITH PLANES)
2. 충돌 반응
(COLLISION RESPONSE)
3. 튀는 공 구현하기
(IMPLEMENTING A BOUNCING BALL)
4. 다각형 기하학
(POLYGONAL GEOMETRY)
5. 점-다각형 충돌
(POINT-POLYGON COLLISION)
6. 특별한 경우 : 삼각형 상호작용
(A SPECIAL CASE: TRIANGLE INTERSECTION)

1. 평면과의 충돌



Image used by permission
Acushnet Company, MA

- 실제 충돌은 매우 짧은 시간에 걸쳐 적용된다.
- 또한 딱딱한 두 물체가 충돌하더라도 각 물체에는 약간의 변형이 일어나며, 이를 다시 복원하는 과정에서 물체가 분리된다.

1. 평면과의 충돌



Image used by permission
Acushnet Company, MA

- 그러나 애니메이션 갱신은 이 과정에 비해 너무 긴 시간이 요구되어 실제로는 변형없이 “즉시” 튀어 나오는 것처럼 보인다.
- 따라서 물리적 기반 애니메이션에서 강체 간의 충돌은 연속된 프로세스가 아닌, 순간적인 이벤트로 처리된다.

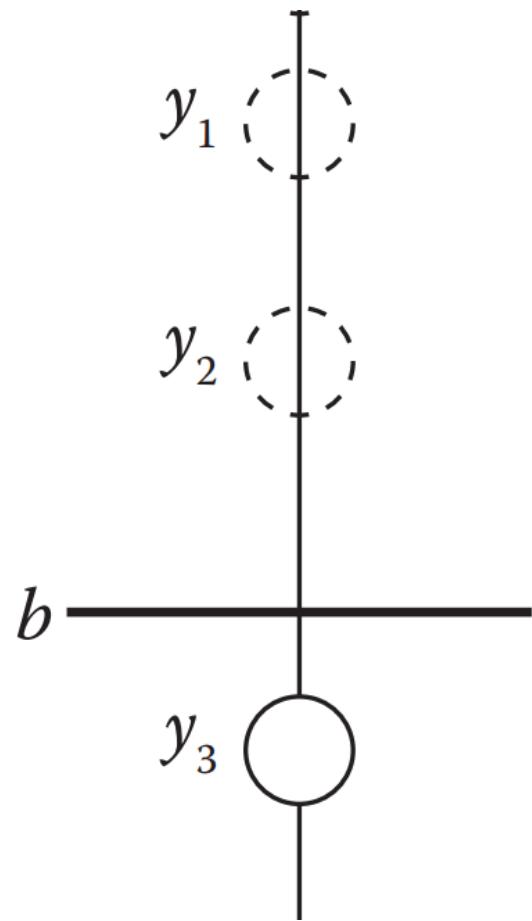
1. 평면과의 충돌

- 물리 기반 애니메이션에서의 충돌은 일반적으로 다음과 같은 3단계로 나누어 처리한다.
 1. 충돌 감지 : 충돌이 발생하였는가?
 2. 충돌 결정 : 충돌이 언제, 어디서 발생하였는가?
 3. 충돌 응답 : 충돌로 인한 결과는 무엇인가?

1. 평면과의 충돌

1.1 충돌 감지 – 1D 예제

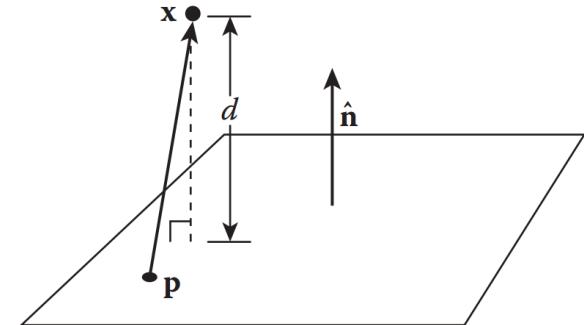
- 가정
 - y 가 중심인 움직이는 공
 - b 에 위치한 장벽
- 공과 장벽의 위치 관계는 $y-b$ 을 계산하면 판단할 수 있음
- $y-b$ 의 부호가 바뀌면, 충돌이 발생했다는 의미



1. 평면과의 충돌

1.1 충돌 감지 – 3D 예제

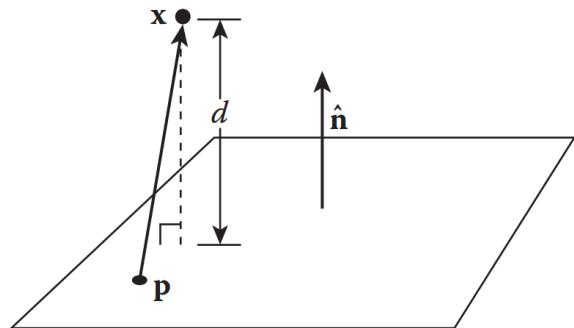
- 가정
 - 점 $x = (x, y, z)$
 - 평면 $Ax + By + Cz + D = 0$
- 평면식에 점 x 의 좌표를 대입하여 값이 0 이면 점은 평면 위에 있다고 할 수 있음.
- 평면 법선 벡터와 평면에 있는 점 p 를 사용하면 충돌 감지 및 결정에 더 유용함.



1. 평면과의 충돌

1.1 충돌 감지 – 3D 예제

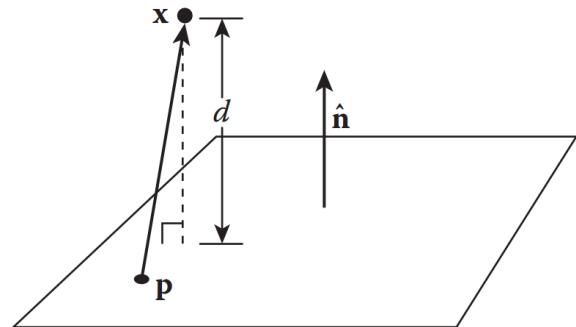
- 가정
 - 평면에 있는 점 $p = (p_x, p_y, p_z)$
 - 평면 법선 벡터 $\hat{n} = [n_x n_y n_z]^T$
- 이것들은 다음과 같은 평면 방정식의 형태를 만드는 데 사용될 수 있다.
 - $n_x x + n_y y + n_z z - \hat{n} \cdot p = 0$
 - $(x - p) \cdot \hat{n} = 0$



1. 평면과의 충돌

1.1 충돌 감지 – 3D 예제

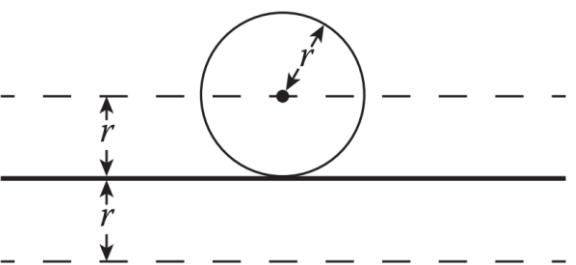
- $(x - p) \cdot \hat{n} = 0$ 에서 x 는 검사할 공간의 임의의 위치
 $(x - p)$ 는 p 에서 x 까지의 벡터
- 평면 법선 벡터 \hat{n} 과 $(x - p)$ 의 내적은 법선 방향으로의 길이를 나타내며, 이는 곧 점 x 와 평면 사이의 거리 d 를 의미한다.
- 따라서 d 부호가 변하면 충돌이 발생한 것



1. 평면과의 충돌

1.1 충돌 감지 – 반경 r 을 가지는 공

- 가정
 - 공이 평면과 충돌했을 때,
공의 중심과 평면 사이의 거리는 r
- 방향에 따라 다음 식을 적용하여 충돌 여부를 검사
 - 공이 양의 방향일 때 : $d = (x - p) \cdot \hat{n} - r$
 - 공이 음의 방향일 때 : $d = (x - p) \cdot \hat{n} + r$



1. 평면과의 충돌

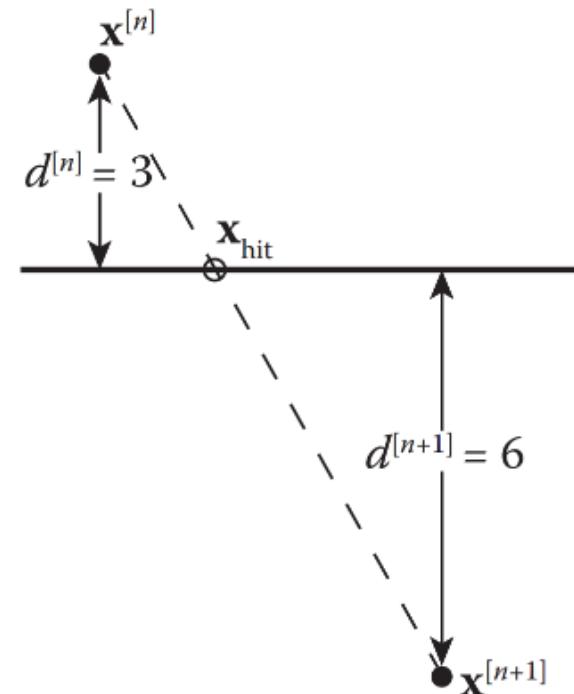
1.2 충돌 결정

- 어느 시간 단계(Time Step)에서 충돌이 발생했다면, 다음은 이 충돌이 언제, 어디서 발생했는지 정확히 파악하는 것이다.
- 오일러 통합을 사용하면, 시간 단계 전후에 있는 객체의 위치를 검사하고 선형 보간을 통해 충돌 위치를 결정 할 수 있다.

1. 평면과의 충돌

1.2 충돌 결정

- 가정
 - 시간 단계(h)동안 속도가 일정함
 - 가정에 따라, 시간은 이동한 거리에 직접 비례
 - $f = \frac{d^{[n]}}{d^{[n]} - d^{[n+1]}}$
 - $t_{hit} = t + fh$



1.

1.3 시
○ 중 중 중 시

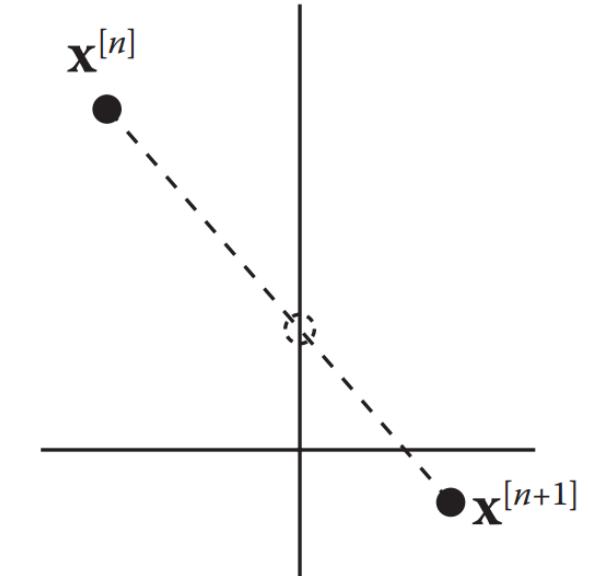
while $t < t_{max}$ **do** loop invariant: s is the state at time t
output state for step number n here

```
TimestepRemaining = h;  
Timestep = TimestepRemaining; try to simulate a full timestep  
while TimestepRemaining > 0 do  
     $\dot{s} = \text{GetDeriv}(s)$ ; determine accelerations  
  
    integrate from state s using derivative  $\dot{s}$  for time Timestep  
     $s_{\text{new}} = \text{Integrate}(s, \dot{s}, \text{Timestep});$   
  
if CollisionBetween( $s, s_{\text{new}}$ ) then  
    calculate first collision and reintegrate  
    Calculate  $f$ ; using Equation 3.2  
    Timestep =  $f$  Timestep;  
     $s_{\text{new}} = \text{Integrate}(s, \dot{s}, \text{Timestep});$   
  
     $s_{\text{new}} = \text{CollisionResponse}(s_{\text{new}});$   
end  
TimestepRemaining = TimestepRemaining - Timestep;
```

1. 평면과의 충돌

1.3 시뮬레이션 루프 업데이트

- 개체가 단일 시간 단계에서 여러 평면을 통과할 가능성도 있다.
- 이때는 가능한 모든 충돌을 감지하여 계산해야 한다.
- 발생 가능한 모든 충돌 중에서 f 의 최소값을 가진 충돌이 가장 먼저 발생하며, 이를 선택해야 한다.

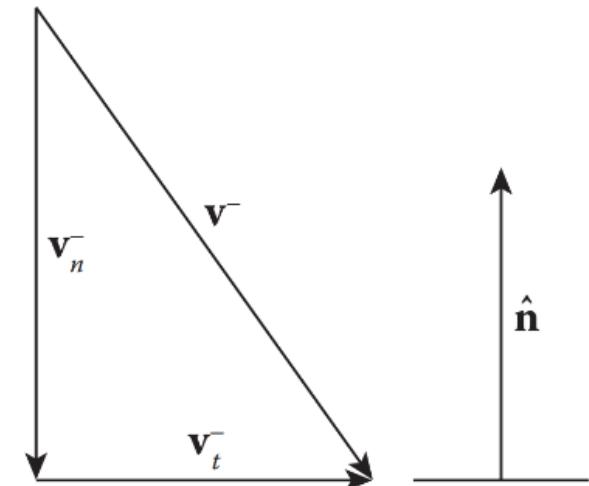


2. 충돌 반응

- 가정
 - 공은 회전하지 않음
 - 무한한 평면은 고정되어 있음
- 따라서 충돌의 결과로 공의 속도가 어떻게 변화하는지를 관찰하는 것이 목표
- 충돌 직전의 상태 : 위첨자 +
○ 충돌 직후의 상태 : 위첨자 -
- 충돌 직후의 공의 위치는 변하지 않으므로
 $x^- = x^+$
- v^- 로 v^+ 를 찾아야 함

2. 충돌 반응

- v^- 의 평면 법선 방향으로의 속도
 - $v_n^- = (v^- \cdot \hat{n})\hat{n}$
- v^- 의 접선 방향 속도
 - $v_t^- = v^- - v_n^-$
- 이 2개의 성분은 탄성 및 마찰 효과를 계산하기 위해 사용됨



2. 충돌 반응

2.1 탄성

- 가정
 - 충돌 에너지의 일부분이 충돌 과정에서 탄성 에너지로 변환됨
 - 따라서 평면과 충돌한 공은 속도가 줄어든 채 평면에서 멀어지게 되며, 이때 속도의 변화는 반발 계수 c_r 에 따라 달라짐
 - 반발계수 C_r 가 적용된 평면의 법선 방향으로의 속도
 - $v_n^+ = -c_r v_n^- = -c_r(v^- \cdot \hat{n})\hat{n}$

2. 충돌 반응

2.1 탄성

- 다양한 상황에서의 반발계수 c_r
 - 목재 표면에 부딪친 야구공 $c_r < 0.5$
 - 코트에 떨어진 테니스공 $0.73 < c_r < 0.76$
 - 코트에 떨어진 농구공 $0.82 < c_r < 0.88$

2. 충돌 반응

2.2 마찰

- 가정
 - 마찰은 접선 방향의 속도에 비례하여 물체의 접선 운동을 늦추는 역할
- 마찰계수 c_f : 0 ~ 1 사이의 값을 가지는 접선 속도 손실 비율
- 충돌 전후의 접선 속도의 관계
 - $v_t^+ = (1 - c_f)v_t^- = (1 - c_f)(v^- - v_n^-)$

2. 충돌 반응

2.2 마찰

- Coulomb 모델

- 두 고체 표면 간에 생기는 마찰력이 외견 접촉 면적 및 마찰 속도와 무관하며, 수직 하중에 비례하는 세가지 조건을 만족시키는 경우의 마찰을 말함 – 네이버 지식백과

- $\|F_t\| = \mu \|F_n\|$

- 다양한 상황에서의 Coulomb 마찰계수 μ

- 테플론 $\mu < 0.05$

- 나무 $0.25 < \mu < 0.5$

- 강철 $0.5 < \mu < 0.8$

2. 충돌 반응

2.2 마찰

- 가정
 - 충돌하는 동안 힘은 매우 작은 길이의 시간에 적용됨
 - 시간에 따른 힘의 적분은 운동량을 제공하므로 힘 ma 대신 운동량 mv 를 대입
- 접선 운동량의 변화는 물체가 표면의 법선 방향으로 충돌하는 운동량에 비례 따라서 마찰력은 $m\|v_n^-\|$ 에 비례

2. 충돌 반응

2.2 마찰

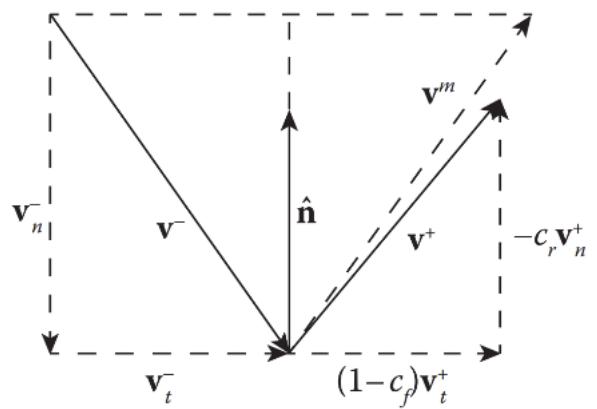
- 마찰력은 접선 운동을 정지시키는 데 필요한 힘보다 반드시 커야 하며,
마찰력으로 인한 접선 속도의 변화를 조사해
원래의 접선 속도보다 절대 커지지 않도록 해야 함
- 새로운 접선 속도 v_t^+
 - $v_t^+ = v_t^- - \min(\mu\|v_n^-\|, \|v_t^-\|)\hat{v}_t^-$

2. 충돌 반응

2.3 종합

○ 완전한 충돌 응답 프로세스 요약

- 충돌 직전의 속도 v^- 를 정규 및 접선 구성 요소인 v_n^- 과 v_t^- 로 나눔
- 탄성 응답 v_n^+ 과 마찰 응답 v_t^+ 을 계산
- 새로운 속도 $v^+ = v_n^+ + v_t^+$ 로 변경



3. 뒤는 공 구현하기

3.1 수치 정밀도

- 반올림 오류
 - 대부분의 숫자가 2진 부동 소수점 형식으로 표현되어 실수를 정확하게 표현할 수 없음
 - 대신 컴퓨터가 표현할 수 있는 가장 가까운 숫자로 반올림하여 사용함
 - 따라서 아주 정확한 정밀도를 요구하는 테스트에서 문제가 발생할 수 있음

3. 튀는 공 구현하기

3.1 수치 정밀도

○ 통합 오류

- 연속 시간 대신 이산 시간 단계를 사용하고,
시간 단계 내에서 부정확한 방법으로 모션을 나타낼 때 발생
- 또는 매끄러운 표면이 다각형에 가깝게 처리되거나,
슬리드가 복셀로 근사되는 기하학의 이산화에 의해 발생
- 소스에 관계없이 최종 결과는 “정확한” 수치 값을 가진다고 가정해서는 안됨

3. 튀는 공 구현하기

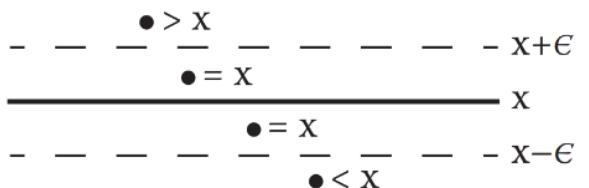
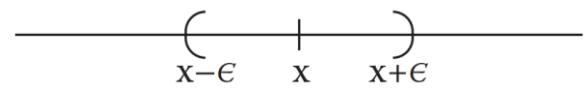
3.1 수치 정밀도

- 지금까지 보았던 시뮬레이션에서 발생할 수 있는 오류
 1. 평면으로부터의 거리 계산 – 반올림 오류
 2. 시간 단계 f 를 사용함에 따라 발생하는 오류
 1. 시간 단계 f 의 비율에 따라 오류가 발생
 2. f 를 사용한 충돌 위치 계산에도 오류가 발생
 3. 충돌 위치가 평면보다 약간 위 또는 약간 아래가 될 수 있음
- 계산된 충돌 위치가 평면보다 아래가 될 경우, 충돌 응답에서 평면의 “뒷면”과 충돌한 결과를 만들어 공이 평면을 통과해버릴 수 있음

3. 튀는 공 구현하기

3.1 수치 정밀도

- 공차(ϵ)
 - 수학 : 등차수열에서 연속한 두 항의 차이
 - 공학 : 실용상 허용되는 범위의 오차
- 공차를 사용한 조건 검사
 - $a = b \rightarrow |a - b| < \epsilon$
 - $a > b \rightarrow a - b > \epsilon$
 - $a < b \rightarrow a - b < -\epsilon$



3. 튀는 공 구현하기

3.1 수치 정밀도

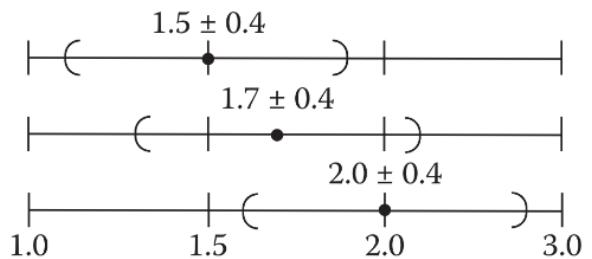
○ 공차의 한계

1. 좋은 공차를 설정하는 것

- 공차의 범위에 따라 분류가 잘못될 수 있음
- 경우에 따라 서로 다른 범위의 공차가 필요할 수 있음

2. 비이행성 발생(incidence intransitivity)

- 이행성 : $a=b$ 및 $b=c$ 일때, $a=c$
- 공차로 인해 이행성이 위반될 수 있음



3. 튕는 공 구현하기

3.2 정지 조건

- 충돌이 완전한 탄성이 아니라면, 공은 점점 낮게 튕겨 결국 멈춰야 함
- 지금까지의 공식을 적용하면 공의 속도는 줄어들지만 결코 멈추지는 않으며 지속적으로 떨리는 현상을 보여줌.
- 이는 수치 정확도 문제 및 컴퓨터 리소스의 낭비를 야기함

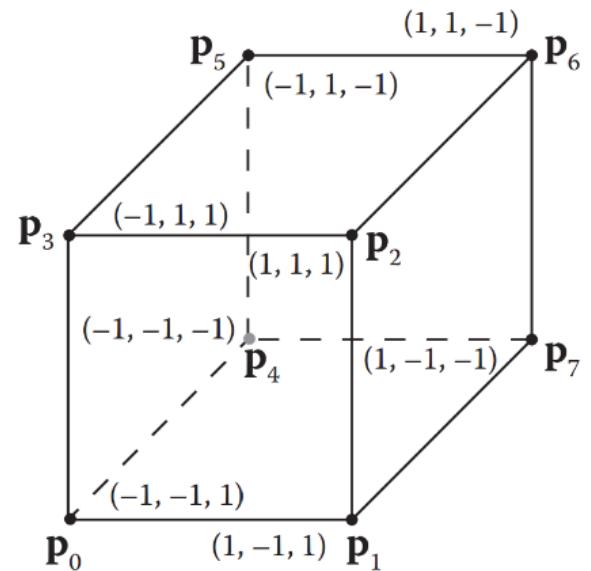
3. 튕는 공 구현하기

3.2 정지 조건

- 공의 정지를 결정하는 기준
 - $\|v\| < \epsilon_1$ (속도의 크기 $\|v\|$ 가 공차 범위 ϵ_1 보다 작은가?)
 - $d < \epsilon_2$ (충돌 거리 d 가 공차 범위 ϵ_2 보다 작은가?)
 - $F \cdot \hat{n} < \epsilon_3$ (힘 F 와 평면 법선 \hat{n} 의 내적이 공차 범위 ϵ_3 보다 작은가?)
 - $\|F_t\| < \mu \|F_n\|$ (접선 힘 $\|F_t\|$ 가 마찰력 $\mu \|F_n\|$ 보다 작은가?)

4. 다각형 기하학

- 유용한 시뮬레이터를 만들기 위해서는 유한 기하 객체와의 충돌을 탐지할 수 있어야 함
- 다각형 모델에서 다각형은 3D 좌표로 지정된 꼭지점과 이를 연결하여 모서리와 면을 형성함

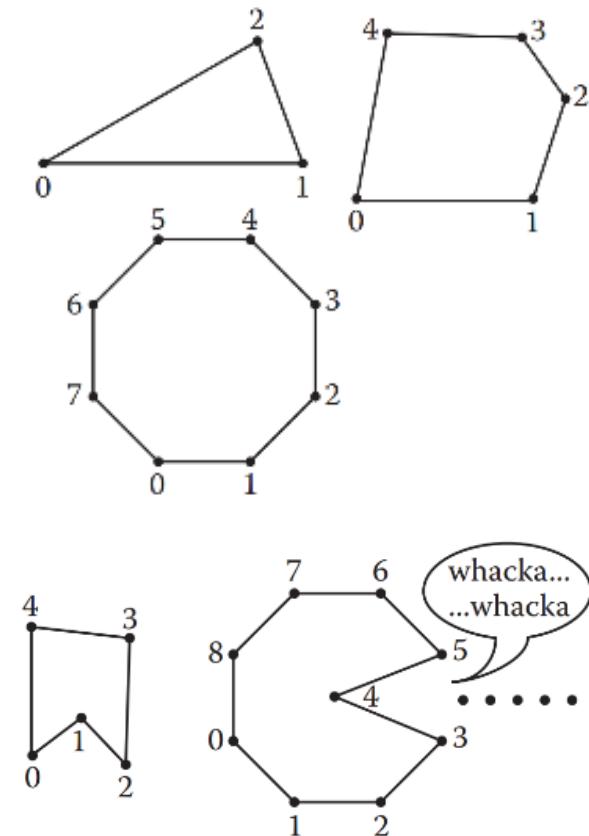


4. 다각형 기하학

- 튀는 공과 다각형 모델을 충돌시키기 위한 최소 조건
 - 기하학의 다각형과 움직이는 점의 교차를 감지하는 법
 - 각 교차점에서 표면 법선을 결정하는 법

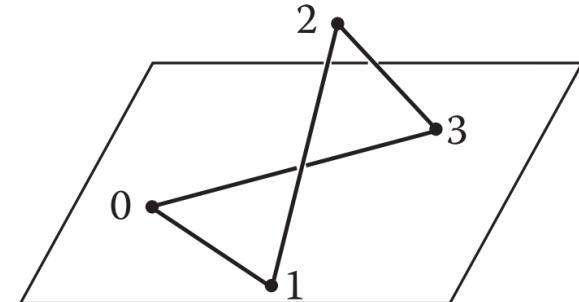
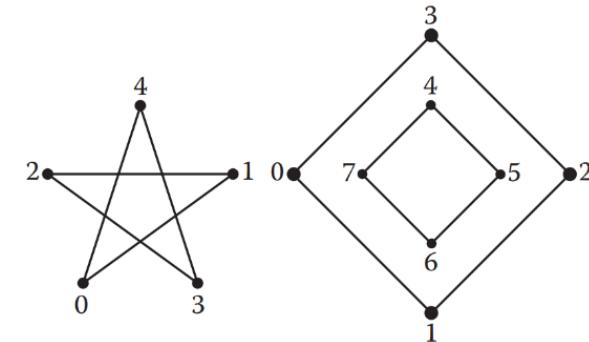
4. 다각형 기하학

- 단순한 다각형은 모서리가 순서대로 찍히지 않고, 교차하지 않는 모든 단일 평면에 있는 정점 집합
- 볼록 다각형은 내부 각이 180° 이하, 오목 다각형은 내부 각이 180° 이상



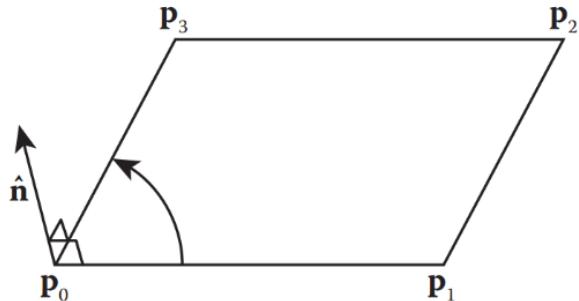
4. 다각형 기하학

- 우측의 별과 다이아몬드는 단순한 다각형이 아니며 여기서는 고려하지 않음
- 4개의 정점으로 구성된 2번째 그림은 다각형이 아니지만 두개의 삼각형으로 분리함으로서 점 대 면 충돌을 쉽게 계산할 수 있도록 함



4. 다각형 기하학

- 평면은 표면 법선 \hat{n} 과 평면에 있는 단일 점 p 에 의해 고유하게 정의됨
- 다각형은 정의에 따라 평면이므로, 임의의 평행하지 않은 모서리 2개의 외적을 취하고 정규화하여 표면 법선을 결정할 수 있음
- $\hat{n} = \frac{(p_1-p_0) \times (p_3-p_0)}{\|(p_1-p_0) \times (p_3-p_0)\|}$
- $p = p_0$

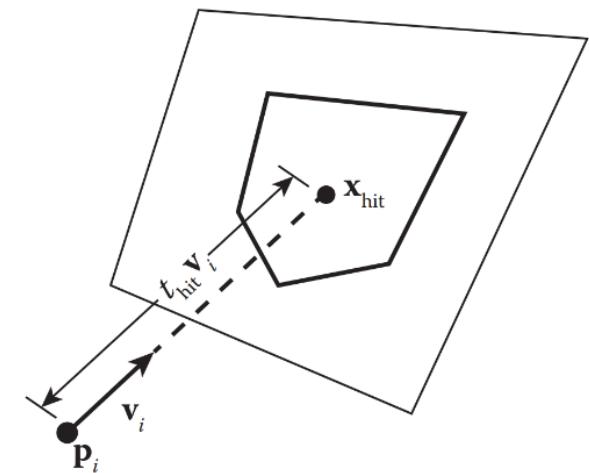


5. 점-다각형 충돌

- 움직이는 점과 다각형이 교차하는지 검사하기
 1. 다각형의 평면 방정식을 찾는다.
 2. 충돌 탐지 및 충돌 결정을 수행하여 점과 평면 사이의 교차점 X_{hit} 를 찾는다.
 3. X_{hit} 이 다각형의 내부 또는 외부에 있는 여부를 결정

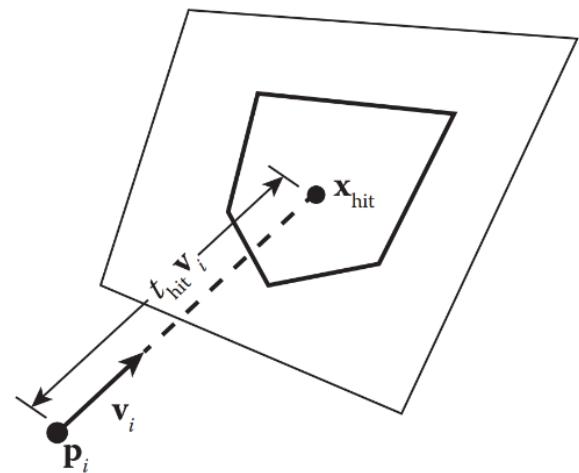
5. 점-다각형 충돌

- 가정
 - 시간 단계 동안 점의 속도가 일정함
- 점-평면 교차 검출하기
 - 평면 방정식 x 를 입자 방정식 p_i 의 현재 위치로 대체
 - 속도 벡터 v_i 또는 $x = p_i + t v_i$ 를 따라 시간 t 에서 전방 투영
- $t_{hit} = \frac{(p - p_i) \cdot \hat{n}}{v_i \cdot \hat{n}}$



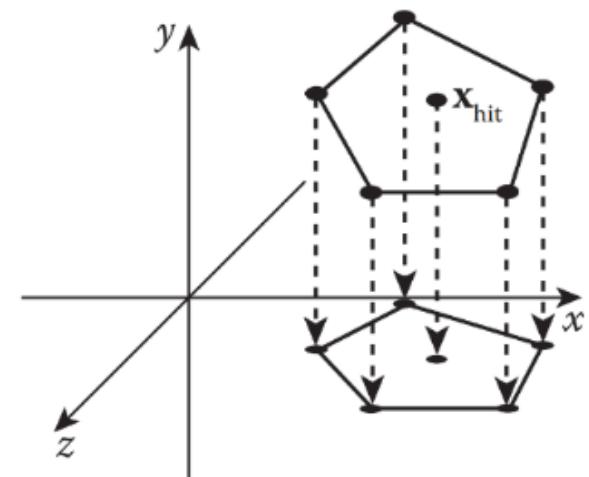
5. 점-다각형 충돌

- $0 \leq t_{hit} < h$ 이면 시간 단계 동안 t_{hit} 에 충돌 발생
- 시간 단계에서 충돌이 발생할 경우 충돌 위치
 - $x_{hit} = p_i + t_{hit}v_i$



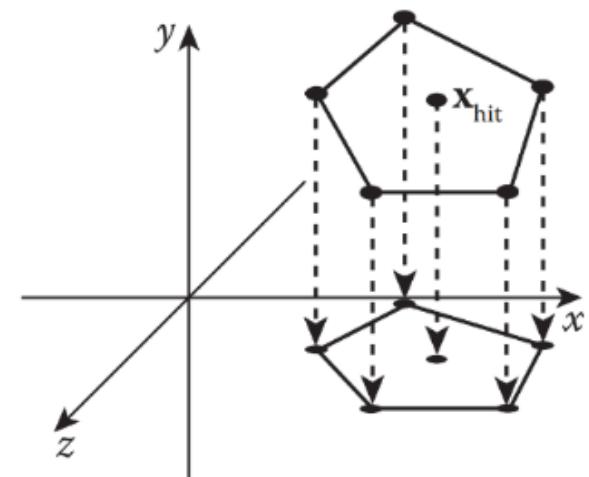
5. 점-다각형 충돌

- 내부-외부 테스트는 3D 문제 처럼 보이나, 다각형의 모든 정점과 x_{hit} 가 같은 평면에 있기 때문에 실제로는 평면임
- 다각형을 선으로 투영하지 않으려면, 다각형의 가장 큰 직교 투영을 갖는 좌표 평면에 다각형을 투영 해야 함



5. 점-다각형 충돌

- 최상의 좌표 평면은 표면 법선 \hat{n} 을 검사하여 결정할 수 있으며, 가장 큰 크기를 가진 \hat{n} 의 원소는 평면에 가장 수직인 법선의 방향을 결정
- 그에 따른 투영은 다음과 같음
 - $|\hat{n}_x|$ y – z 평면에서 가장 큰 투영 : $(x, y, z) \Rightarrow (y, z)$
 - $|\hat{n}_y|$ z – x 평면에서 가장 큰 투영 : $(x, y, z) \Rightarrow (z, x)$
 - $|\hat{n}_z|$ x – y 평면에서 가장 큰 투영 : $(x, y, z) \Rightarrow (x, y)$



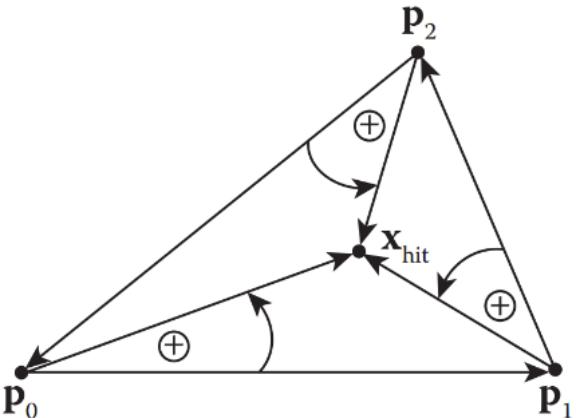
5. 점-다각형 충돌

○ 볼록 다각형에 대해 작동할 내부-외부 테스트 알고리즘

1. 모든 점을 2D 좌표 평면에 투영하여 2D 문제로 만든다
2. 각 꼭지점에 대한 2D 가장자리 벡터를 계산하고, 다음의 연속하는 정점으로 부터 이 정점을 뺀다.
3. x_{hit} 에서 꼭지점을 빼서 각 꼭지점에 대해 다른 2D 벡터를 계산한다.
4. 가장자리 벡터를 맨 위 행으로 사용하고 벡터를 x_{hit} 방향으로 맨 아래 행으로 하여 각 꼭지점에 대해 2×2 행렬을 형성한다.
5. 각 꼭지점에 대한 행렬의 행렬식을 계산한다.
6. 모든 결정자의 부호가 동일하다면, x_{hit} 은 다각형 안에 있어야 한다.

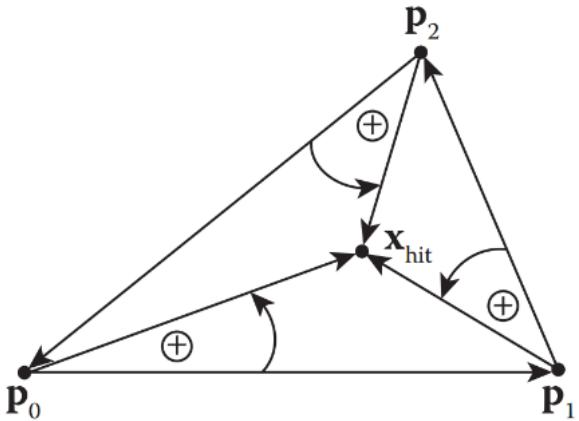
5. 점-다각형 충돌

- 해당 알고리즘은 위에서 계산된 결정 요인 각각이 가장자리 벡터와 x_{hit} 방향으로 벡터의 교차 곱의 z 구성 요소와 동일하다는 것을 인식하면 됨
- x_{hit} 과 함께 p_0 에서 p_1 까지의 가장자리 행렬식
- $$\begin{vmatrix} x_1 - x_0 & y_1 - y_0 \\ x_{hit} - x_0 & y_{hit} - y_0 \end{vmatrix} = (x_1 - x_0)(y_{hit} - y_0) - (y_1 - y_0)(x_{hit} - x_0)$$



5. 점-다각형 충돌

- 마찬가지로 p_0 에서 p_1 까지의 벡터와 p_0 에서 x_{hit} 까지의 벡터 간의 교차 곱이 3D로 확장되어 다음과 같은 동일한 결과를 얻음.
- $$\begin{bmatrix} x_1 - x_0 \\ y_1 - y_0 \\ 0 \end{bmatrix} \times \begin{bmatrix} x_{hit} - x_0 \\ y_{hit} - y_0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ (x_1 - x_0)(y_{hit} - y_0) - (y_1 - y_0)(x_{hit} - x_0) \end{bmatrix}$$
- x_{hit} 이 삼각형 내부에 있는 한, 이러한 교차 곱은 모두 평면에서 위 또는 안으로 향해야 한다.



6. 삼각형 상호 작용

- 삼각형이 항상 다각형인 이유
 - 항상 3개의 꼭지점이 하나의 평면에 있음
 - 가장자리가 교차할 수 없음
- 때문에 그래픽 시스템은 렌더링 또는 다각형이 필요한 계산을 수행하기 전에, 표면을 최대한 삼각형으로 쪼깝.

