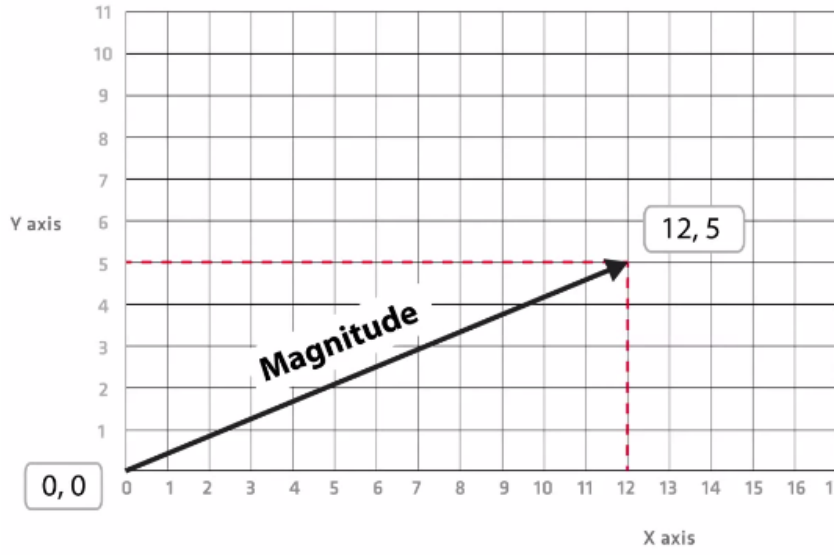
이번강좌는 유니티 공식 홈페이지의 Vector Math 튜토리얼 강좌를 번역하여 활용하였습니다.
http://unity3d.com/learn/tutorials/modules/beginner/scripting/vector-maths-dot-cross-products

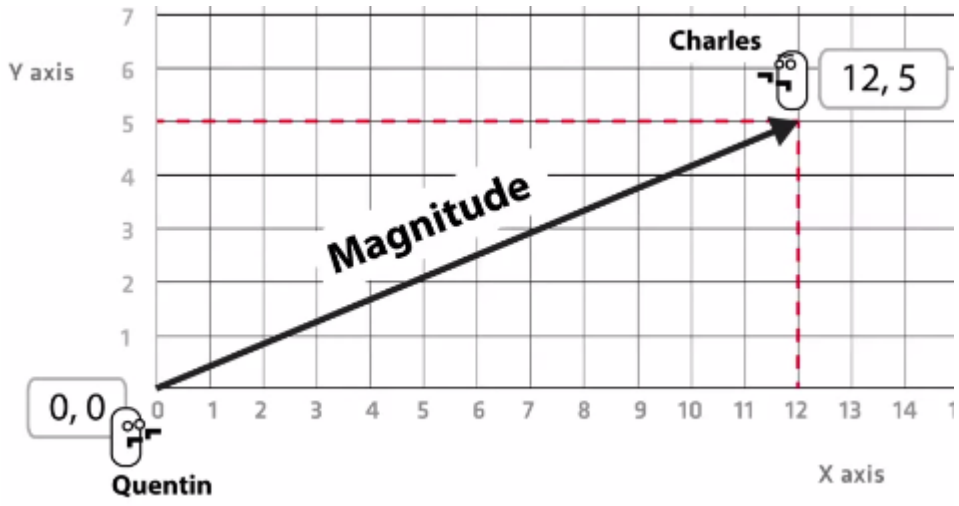
게임 개발에 있어서 벡터는, 위치, 방향 등의 계산을 활용할 때 많이 사용합니다. 이 강좌를 진행하면서 연산들을 우리말로 번역한 것들보다는 영어 원문 그대로 사용하고자 합니다. 왜냐하면 연산의 명칭이 매서드의 이름과 직결되기 때문에 굳이 번역하여 두번 공부하는 일을 줄이고자 함입니다.



벡터는 2점 사이의 라인을 그린 것을 의미합니다. 이 때 벡터의 길이를 Magnitude 라고 합니다. t시작은 심플하게 2D 벡터로 가봅시다.

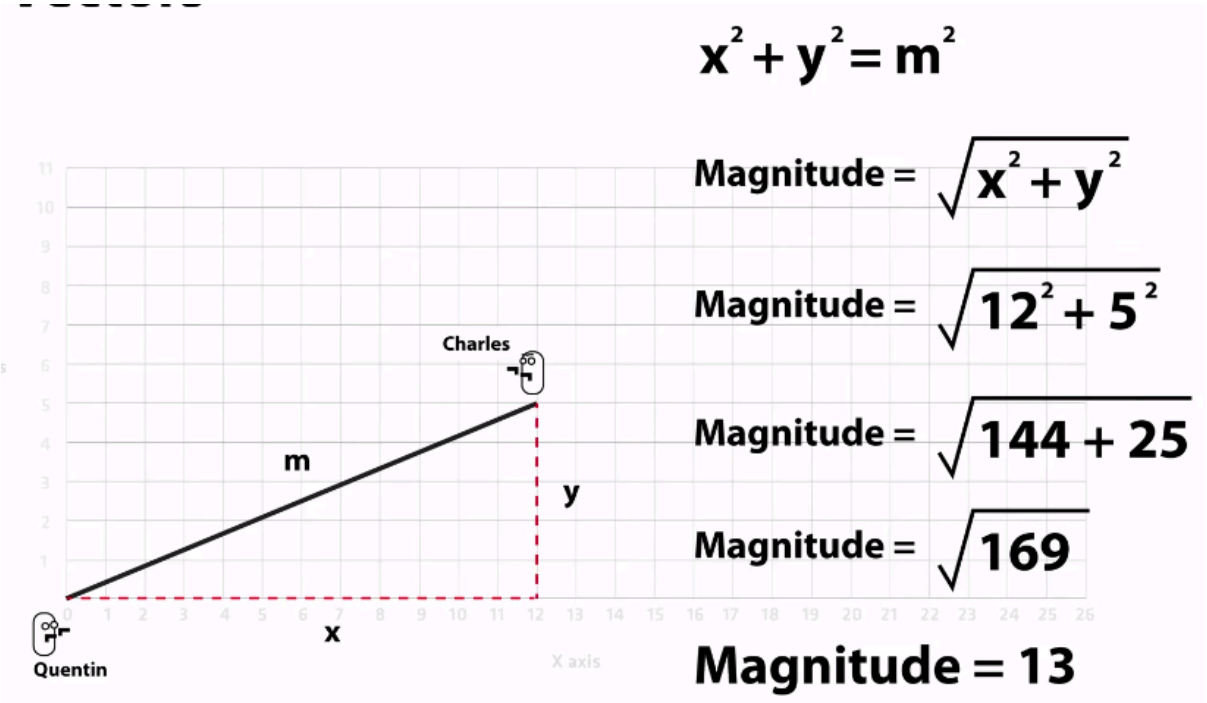
먼저 2D벡터는 원점(0,0)에서부터 시작하여 2D 공간의 어느점이든 연결되는 선을 말합니다. 이때 방향은 원점에서부터 목적지 점으로 뻗어나가는 방향을 말합니다.

이 벡터를 활용해 다음과 같은 일을 할 수 있습니다. 만약 맵상에 아래 그림과 같이 쿠엔틴과 찰스가 서로 총을 겨누고 있다고 가정합시다.



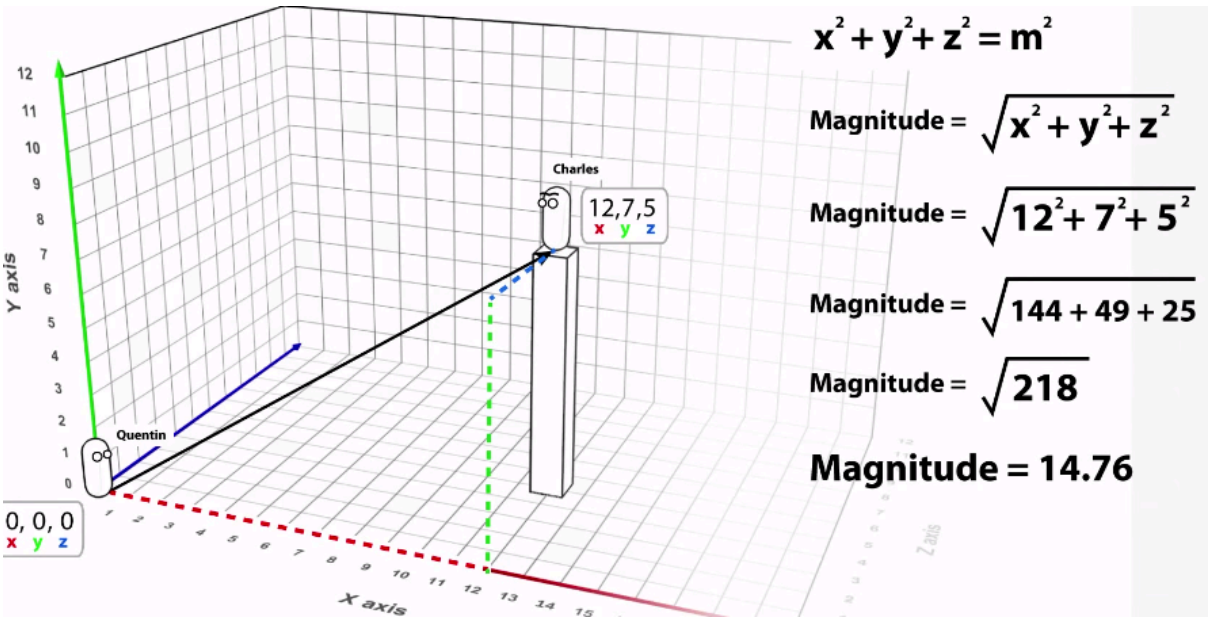
두 사람의 좌표는 (0,0), (12,5)입니다. 그리고 이 총은 12의 사정거리를 지닙니다.

Magnitude는 피타고라스의 정리에 의해서 다음과 같이 계산됩니다. (걱정하지 마세요 유니티가 알아서 계산합니다)



따라서 현상황에서 쿠엔틴은 총을 쏠 수 없습니다. 사정거리 밖이기 때문이죠.

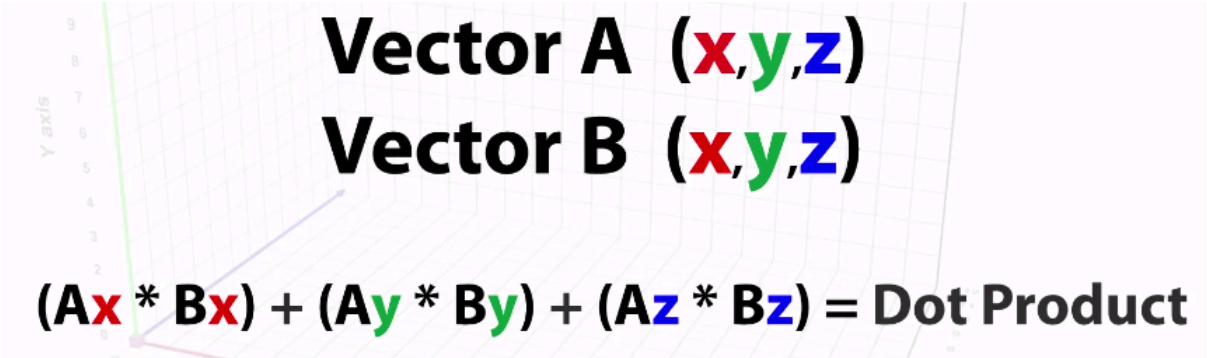
3차원 공간으로 옮겨도 그다지 달라지지는 않습니다.



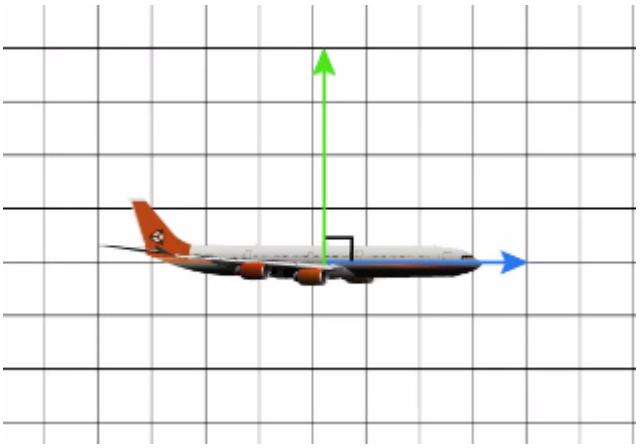
다만 이것을 사람이 직접 계산하지는 않습니다. 유니티의 벡터 클래스에는 당연하게도 Magnitude 매서드가 존재합니다.

이와 더불어서 매우 유용한 함수가 닷프로덕트와 크로스 프로덕트입니다.

닷 프로덕트는 2개의 벡터를 받아 곱한 후 이를 합산해주는 함수로 2개 벡터의 곱의 크기를 나타냅니다.

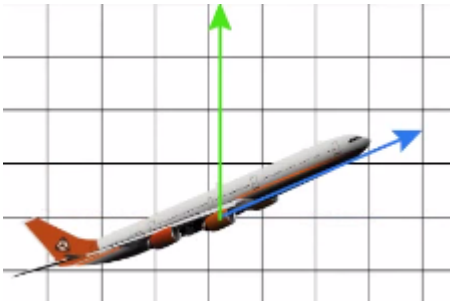


이 닷프로덕트의 값을 통해 2개의 벡터가 수직인지를 판별할 수 있습니다. 직각의 벡터는 닷 프로덕트의 값이 0이 나오게 되어 있습니다. 이를 통해 우리는 수평으 구할 수 있습니다. 다음과 같은 상황을 생각해보세요.



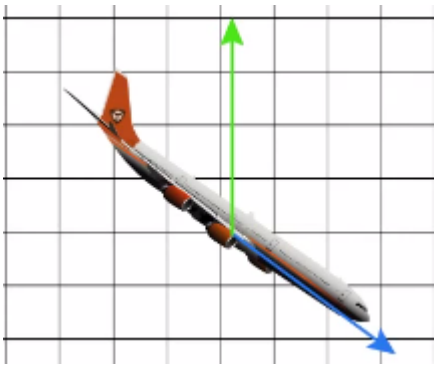
이 비행기의 중력 역방향 벡터인 초록색과 비행기의 전방 벡터인 푸른색 간의 닷프로덕트 값이 0이라면 비행기는 수평을 유지하고 있는 것이며 이때 비행기는 가장 적은 저항력을 가지게 됩니다.

만약 비행기가 아래와 같은 상황이라면 닷 프로덕트는 양수 값을 가지게 됩니다.



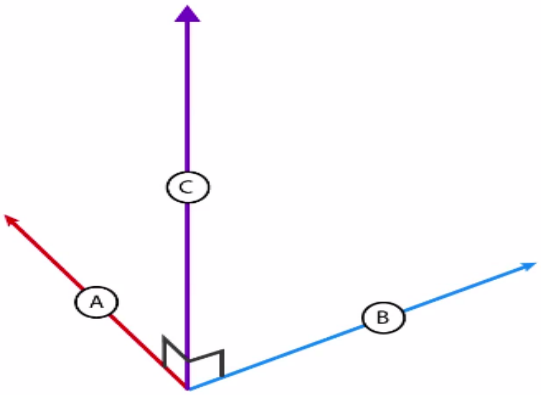
따라서 이때는 비행기가 솟아오르고 있다는 것을 알 소 있죠. 그리고 있대는 저항력이 추가되게 됩니다. 그리고 닷프로덕트의 결과값이 음수값을 가지게 된다는 것은 비행기가 아래 그림과 같은 상태가

되었다는 것을 말합니다.

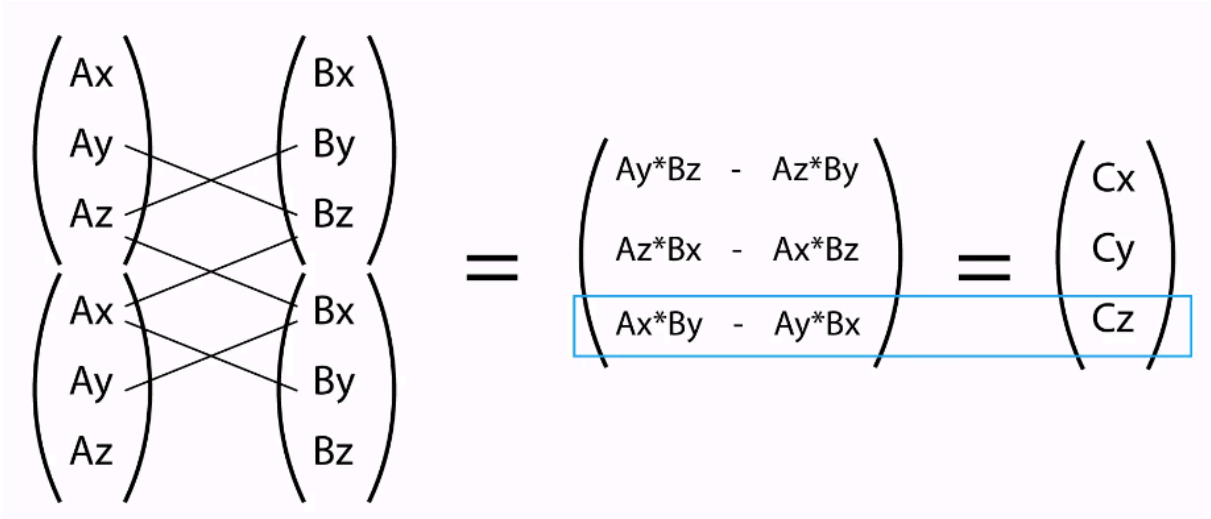


유니티는 이것을 계산하기 위한 Vector3.Dot(Vector A, Vector B) 매서드를 기본으로 제공하고 있습니다.

크로스 프로덕트는 닷 프로덕트와는 달리 단일값이 아닌 벡터를 결과로 만들어냅니다. 만약 아래 그림과 같은 A벡터와 B벡터를 곱을 해주게 되면 이 2개의 벡터와 직교하는 벡터가 생성됩니다.

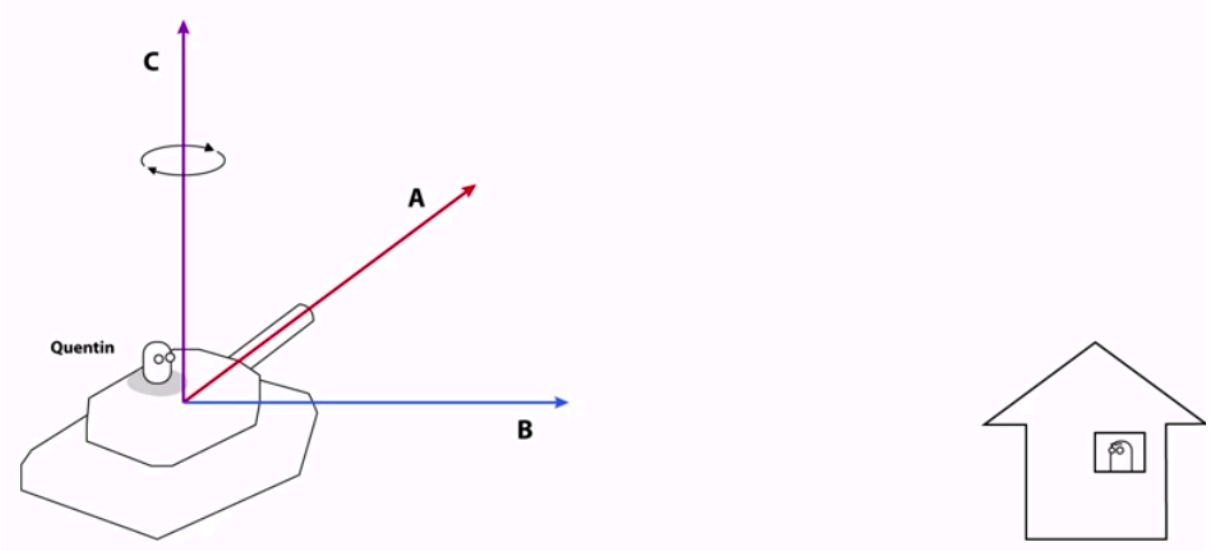


구하는 과정은 다음과 같으나 굳이 기억해야할 필요는 없습니다.



이또한 유니티에서 함수로 제공하기 때문이죠. Vector3.Cross(Vector A, Vector B) 입니다.

크로스 프로덕트는 다음과 같은 상황에서 쓰입니다.



탱크가 현재 A방향을 보고 있고 이 때 포탑을 B방향으로 회전시키려 할 때 C축을 기준으로 회전해야합니다.

이때 Transform.rotate 매서드를 활용해 C벡터에 값을 곱해나가면 해당 축을 기반으로 회전합니다.

또한 벡터는 방향을 잡을 때도 사용합니다. 예를 들어 A(2, 3) 의 위치에서 B(7, 2) 를 바라본다면

B에서 A를 빼주면 A가 B를 바라보는 벡터가 생기게 됩니다. 이 벡터를 normalized 연산을 통해 길이 1짜리 방향성만을 가지는 벡터를 가져오면 이것이 방향벡터가 됩니다. 주로 총알의 발사시 목적지 계산을 위해 사용됩니다.