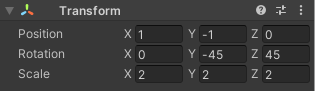
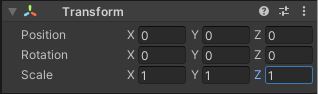
Unity’de Vektörlerin Kullanımına İlişkin Birkaç Örnek

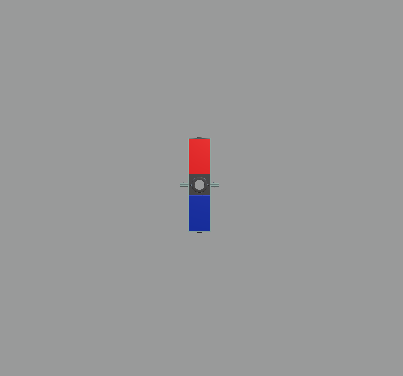
Bilindiği gibi temel Matematik konuları arasında vektörler oldukça önemli bir yer işgal etmektedirler.

Vektörler birbirinden çok farklı büyüklükleri ifade etmekte ve bu büyüklükler arasındaki ilişkileri analiz etmede yoğun bir şekilde kullanılmaktadır. Mekanik sistemlerde hareket ve kuvvet analizlerinde, Elektrik ve Elektronikte alternatif akım devre analizlerinde vb. bir çok alanda vektörler birçok hesaplamada hesaplamayı kolaylaştırmak için kullanılmaktadır.

Bilgisayar oyunlarında da nesnelerin transformasyonları ile ilgili olarak vektörler kullanılmaktadır. Nesnelerin pozisyonları ve rotasyonlarını ifade etmede ve buna bağlı olarak nesnelerin hareketlerini ifade etme ve hesaplamada, nesnelerin büyüklük bilgilerini ifade etme ve büyüklük değişimlerini hesaplamada da vektörler yoğun bir şekilde kullanılmaktadır.

Örneğin Unity’de (popüler oyun motorlarından biri) üç boyutlu bir nesnenin transformasyon bilgileri aşağıda görüldüğü gibi vektörlerle ifade edilir.



oyuncak içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Resim 1 Resim 2**

Resim 1’de sahneye 1:1 ölçeğinde alınmış bir nesne görülmektedir. Nesneye ait transformasyon bilgilerinin anlamı aşağıdaki gibidir.

Pozisyon : x = 0, y = 0, z = 0. (Vector3.Zero)

Rotasyon : x = 0, y = 0, z = 0. (Vector3.Zero)

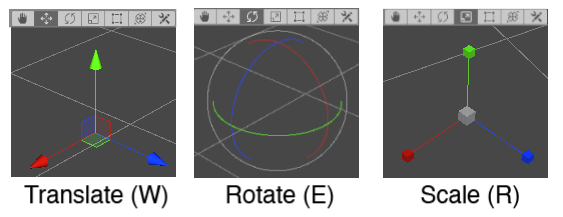
Ölçek : x = 1, y = 1, z = 1. (Vector3.One)

Resim 1’de ise nesne, orijinal rotasyon vektörleri değiştirilmiş haliyle görülmektedir. Nesneye ait transformasyon bilgilerinin anlamı aşağıdaki gibidir.

Pozisyon : x = 1, y = -1, z = 0. Nesne 1 birim sağa ve 1 birim aşağıya kaydırılmış

Rotasyon : x = 0, y = -45, z = 45. Nesne Y ekseninde 45 sağa ve Z ekseninde 45 derece sola döndürülmüş.

Ölçek : x = 1, y = 1, z = 1. Nesne her eksende (x, y, z) iki kat büyütülmüş



**Resim 3**

Unity’de nesnelerin transformasyonlarını sahneden değiştirme yöntemleri oldukça belirlenmiş yöntemlerle olmaktadır. Örneğin nesneyi 1 birim sağa kaydırmak için ya pozisyon vektörünün tutulduğu metin kutularından x değerini tutan kutuya 1 yazılabilir ya da pozisyon sekmesi seçili iken (Resim 3 1. resim) x eksenini temsil eden ok x ekseni yönünde bir birim sağa kaydırılır. Buna karşın oyun modunda nesnelerin transformasyon bilgilerini değiştirmenin oldukça fazla yol ve yöntemleri vardır. Bu yöntemler, yerleşik fonksiyonlara ek olarak geliştiricinin yaratıcı yetenekleri sayesinde oldukça çeşitli olabilmektedir.

Aşağıda Update fonksiyonu içine yazılmış transform manüplasyonuna yönelik birkaç örnek görülmektedir.

**1.**

Void Update()

{

transform.Translate (1, 0f, 0f); // *Nesneyi birim zamanda 1 birim sağa hareket ettir.*

}

**2.**

void Update()

{

*// Boşluk tuşuna basılınca nesneyi Y ekseni boyunca hareket ettir. (Zıplama)*

*// “Sağ ok” veya “D” tuşuna basılınca nesneyi sağa, “sol ok” veya “A” tuşuna basılınca nesneyi sola hareket ettir.*

*// “Yukarı ok” veya “W” tuşuna basılınca nesneyi ileri, “aşağı ok” veya “S” tuşuna basılınca nesneyi geri hareket ettir.*

Vector3 Vec = transform.localPosition;

Vec.y += Input.GetAxis("Jump") \* Time.deltaTime \* 20;

Vec.x += Input.GetAxis("Horizontal") \* Time.deltaTime \* 20;

Vec.z += Input.GetAxis("Vertical") \* Time.deltaTime \* 20;

transform.localPosition = Vec;

}

**3.**

void Update()

{

Vector 3 S3 = transform.localeScale;

bool x = Input.GetKey(KeyCode.UpArrow) || Input.GetKey(KeyCode.DownArrow) || Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.S);

bool y = Input.GetKey(KeyCode.LeftArrow) || Input.GetKey(KeyCode.RightArrow) || Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.D);

*// “Yukarı ok” yada “W” tuşuna basılınca S3 vektörünün “y” değerini (0-1 birim) arttır.*

*// “Aşağı ok” yada “S” tuşuna basılınca S3 vektörünün “y” değerini (0-1 birim) düşür.*

if (x) S3.y += Input.GetAxis("Vertical");

*// “Sağ ok” yada “D” tuşuna basılınca S3 vektörünün “x” değerini (0-1 birim) arttır.*

*// “Sol ok” yada “A” tuşuna basılınca S3 vektörünün “x” değerini (0-1 birim) düşür.*

if (y) S3.x += Input.GetAxis("Horizontal ");

*// Nesnenin ölçeğini S3 vektörüne eşitle*

transform.localeScale = S3;

}

**4.**

void Update() {

*//eğer nesnenin pozisyonu hedef nesnenin pozisyonundan farklıysa*

if (transform.position != target.transform.position)

{

// hedef nesnenin pozisyon bilgisini 10\*0.02 hızında takip edecek vektör bilgisi

Vector3 pos = Vector3.MoveTowards(transform.position, target.transform.position, 10 \* Time.deltaTime);

transform.position = pos;

}

}

Yukarıdaki örneklere ek olarak vektörlerle yapılan işlemler hem Unity ile kullanılan C#’ta hem de diğer birçok dilde yüzlerle, binlerle ifade edilecek kadar çoktur. Elbette yaratıcılık önemlidir ancak her dilde yerleşik fonksiyonlardan faydalanmak çalışmayı hızlandırabilir.