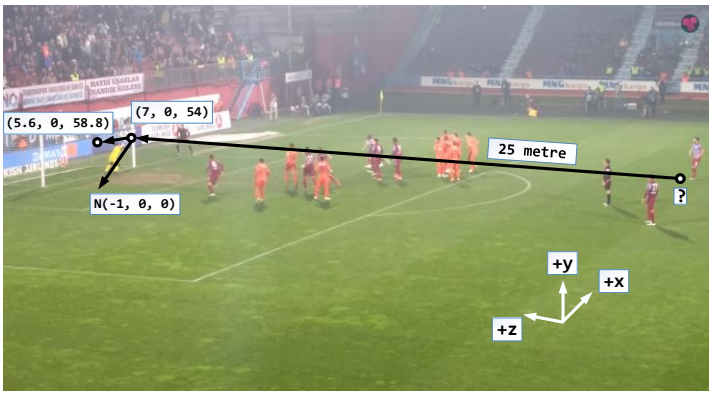
Karadeniz Teknik Üniversitesi
Bilgisayar Mühendisliği Bölümü
Öğr.Gör. Ömer ÇAKIR

BIL320 Bilgisayar Grafikleri I
Arasınav, 7.4.2015, 08:00, D1-2
Süre : 90 Dakika

CEVAPLAR

1. UEFA'nın twitter hesabında "free-kick master" olarak övülen Mehmet EKİCİ, bu sezon serbest vuruş gollerinden birini Başakşehir takımına atmıştı. Top sağ kale direğinden aynasal yansımaya göre sekip ağlarla buluşmuştu. Topun ağlarla buluştuğu nokta **(5.6, 0, 58.8)**, kale direğinden sektiği nokta **(7,0,54)**, direğin normali **(-1,0,0)**, şutu çektiği noktanın kale direğinden sektiği noktaya uzaklığı da **25** metre olsun. Mehmet'in şutu hangi noktadan çektiğini ışın izleme ile hesaplayınız. **(30P)**



R_d = (7, 0, 54) - (5.6, 0, 58.8)
= (1.4, 0, -4.8).normalize()
= (0.28, 0, -0.96)

R_{ref} = (-0.28, 0, -0.96)

Ekici = (7,0,54) + 25*(-0.28,0,-0.96)
= (0,0,30)

U0(0, 30, 160) U1(30, -30, 80) U2(-30, -30, 80)
V0(0, -30, 160) V1(30, 30, 80) V2(-30, 30, 80)

2. Yukarıda köşe noktaları verilen U ve V üçgenlerinin arkayüz (backface) olup olmadıklarını belirleyiniz. Dilediğiniz yöntemi kulllanabilirsiniz. Bakış noktası **(0,0,0)**'dadır. Görüntü düzleminin bakış noktasına uzaklığı 5 birimdir. Normalleri normalize etmeniz gerekmiyor. **(30P)**

$R_1 \times R_2 = (R_{1y}R_{2z}-R_{1z}R_{2y},\ R_{1z}R_{2x}-R_{1x}R_{2z},\ R_{1x}R_{2y}-R_{1y}R_{2x})$

$Normal_U$ = (0, 4800, -3600)
$toEye_{U0}$ = (0, -30, -160)
$Normal_U * toEye_{U0}$ = 432000 > 0
(backface değil)

$Normal_V$ = (0, 4800, 3600)
$toEye_{V0}$ = (0, 30, -160)
$Normal_V * toEye_{V0}$ = -432000 < 0
(backface)

```
Rotate = XMMatrixRotationY( XM_PIDIV4 );   //45° CW
Translate = XMMatrixTranslation(4.0f, 0.0f, 0.0f);
Scale = XMMatrixScaling( 0.5f, 0.5f, 0.5f );

g_World = Rotate * Translate * Rotate * Scale; //1
g_World = Rotate * Scale * Rotate * Translate; //2
g_World = Translate * Rotate * Scale * Rotate; //3
g_World = Scale * Rotate * Translate * Rotate; //4
g_World = Rotate * Translate * Scale * Rotate; //5
g_World = Rotate * Scale * Translate * Rotate; //6
```

3. Küçük küpe ait **1-6** arası rakamlarla temsil edilen yukarıdaki **g_World** matrisi setlemelerinin **I, II, III** ve **IV** ile temsil edilen aşağıdaki ekran görüntüleri ile doğru eşleştirildiği şıkkı işaretleyiniz? **(25P)**

Not → Bakış noktası **(0,4,-8)**'dedir.
Büyük küpün merkezi **(0,0,0)**'dadır, köşe noktaları **-1,+1** değerleri ile setlenmiştir.
45° CW : Saat yönünde 45 derece dönme işlemi.

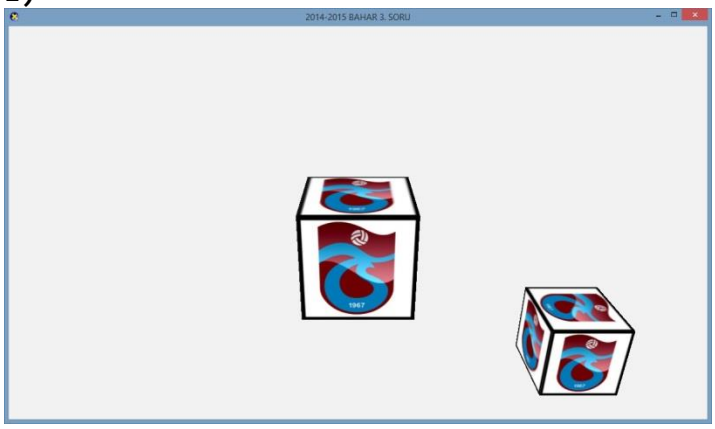
Yanlış cevaptan 5P kırılacaktır!

(A)	**I-1,5**	**II-3**	**III-2**	**IV-4,6**
(B)	**I-4,6**	**II-3**	**III-2**	**IV-1,5**
(C)	**I-1,4**	**II-5**	**III-2**	**IV-3,6**
(D)	**I-1,6**	**II-5**	**III-2**	**IV-3,4**
(E)	**I-1,6**	**II-3**	**III-2**	**IV-4,5**

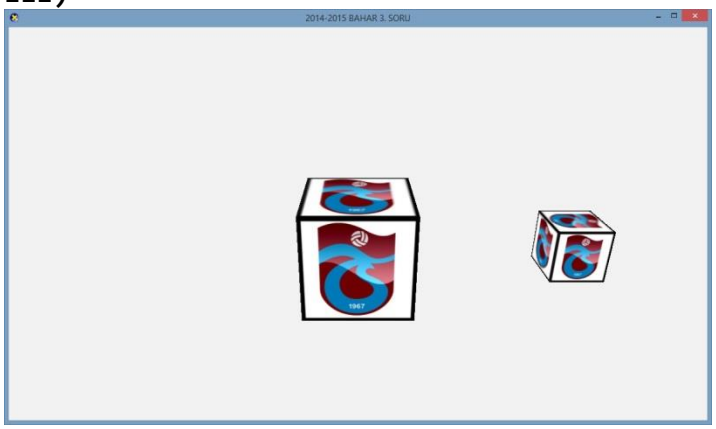
4. 3. soruda büyük küp ve küçük küp aynı vertex/index bufferları kullandığı halde farklı **World** matrisleri kullanılarak farklı büyüklükte ve konumda çizilebilmektedir. Bu iki küpün sanki farklı vertex/index bufferlar kullanıyorlarmış gibi aynı anda ekranda görüntülenmesini sağlayan nesnenin adı nedir? Bunu nasıl ve hangi fonksiyonu kullanarak yapmaktadır? **(15P)**

İki küpün aynı anda ekranda görünmesini SwapChain nesnesi sağlamaktadır.
Back buffera çizim sonrası SwapChain nesnesinin Present() fonksiyonu ile buffer içeriği ekranda görüntülenir.
İki küp de aynı vertex/index bufferları kullandıkları halde World matrisleri güncellenerek backbufferda farklı konumlarda ve büyüklüklerde çizildikten sonra Present() fonksiyonu ile back ve front bufferlar swap yapılarak ekranda görüntülenirler.

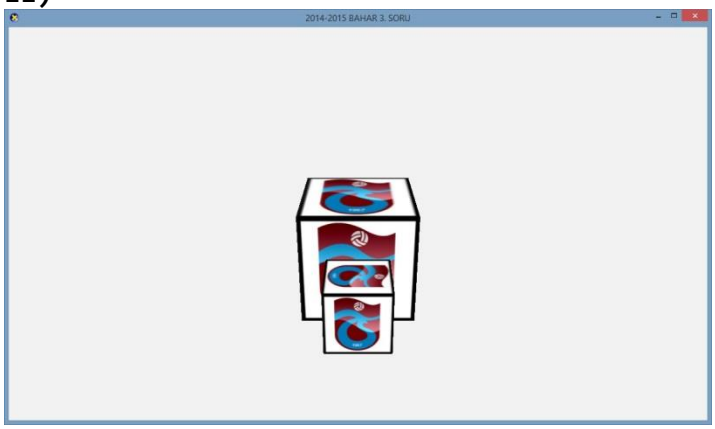
I)



III)



II)



IV)