**PROJE BAŞLIĞI: Nokta Bulutu İşleme**

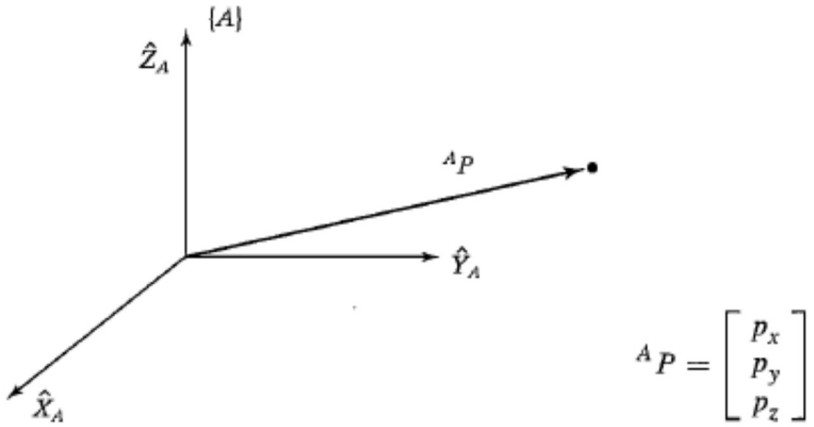
**Proje Açıklaması:**

Projenin kapsamı, nokta bulutları (Point Cloud) ve nokta bulutları üzerinde işlemleri kapsamaktadır. Şekil 1'de resmi verilen derinlik kamerası, görüş alanı içerisinde renkli piksel değerleri yanına derinlik bilgisi de sunmaktadır. Elde edilen 3B derinlik bilgisi kameraya göre tanımlı 3B noktalarla ifade edilmektedir. Bu noktaların oluşturduğu noktalar kümesine nokta bulutu adı verilir. Şekil 1'de bir tavşan yüzeyinden elde edilen nokta bulutunun gösterimi verilmektedir.

**Şekil 1.** Derinlik kamerası ve nokta bulutu

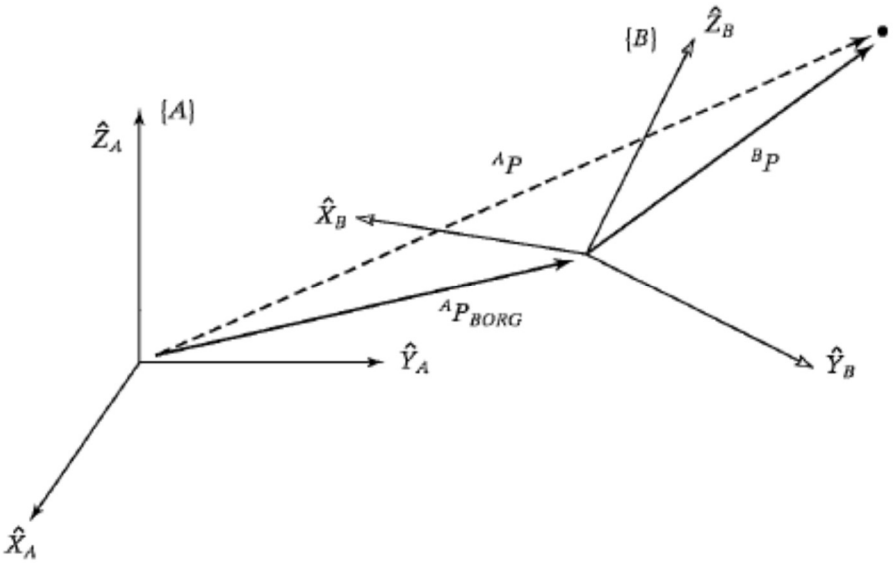
Bir kameradan alınan nokta bulutundaki her bir nokta, o kamera üzerinde tanımlı koordinat sistemine göre tanımlıdır. Örneğin, Şekil 2'de mavi renkli elips içinde gösterilen nokta bulutu $O_{C1}$ koordinat sistemine göre tanımlıdır. Yeşil renkli elips içinde gösterilen nokta bulutu $O_{C2}$ koordinat sistemine göre tanımlıdır. Bu iki nokta bulutunu birleştirip tek bir nokta bulutu olarak elde etmek için, her iki buluttaki noktaları ortak bir koordinat sistemine ($O_B$) göre tanımlı olacak şekilde dönüştürmek gerekir.

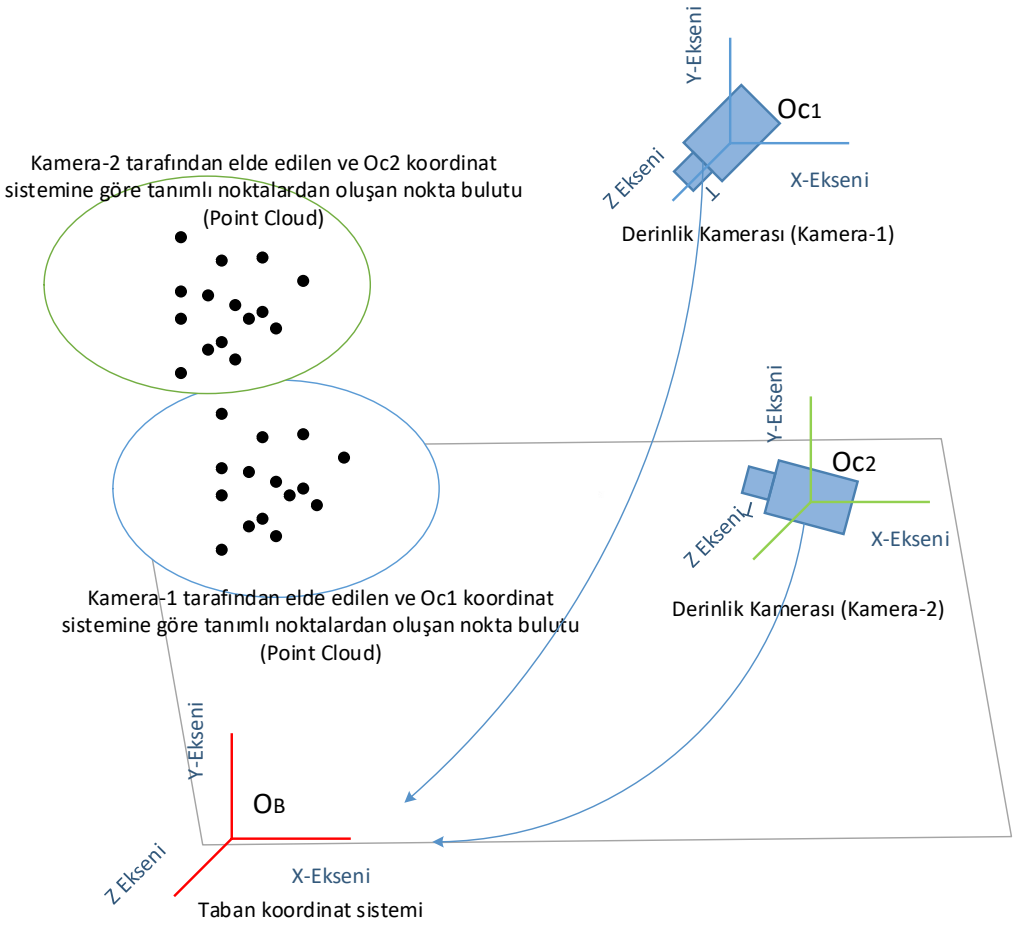
P noktasının, {A} koordinat eksenine göre gösterimi aşağıda verilmektedir.




$$ {}^AP = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} $$

Bir P noktasının, {A} ve {B} koordinat eksenlerine göre, gösterimi sırasıyla ${}^AP$ ve ${}^BP$ ile gösterilmektedir.



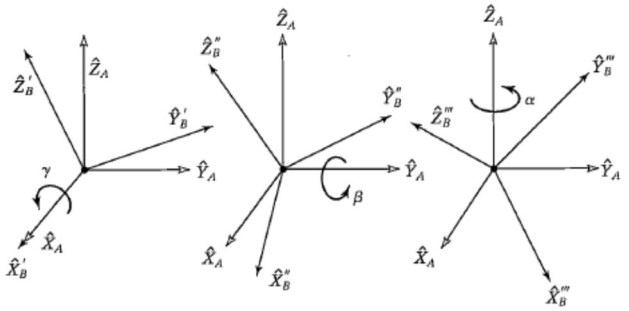




**Şekil 2.** İki farklı kameradan sağlanan nokta bulutları ve referans koordinat sistemlerinin sembolik gösterimi

${}^{B}P$ verildiğinde ${}^{A}P$ vektörünü bulmak için {B} 'den {A}'ya dönüşüm matrisi gerekir. Dönüşüm matrisi ${}^{A}T_{B}$, aşağıda gösterilmektedir. ${}^{A}_{B}R$, {B} 'den {A}'ya 3x3 rotasyon matrisidir.

$$\begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{A}_{B}R & & {}^{A}P_{BORG} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix}.$$

Dönüşüm matrisinin ${}^{A}T_{B}$ tersi, {A} 'dan {B}'ye dönüşüm matrisini verir. $\left({}^{A}T_{B}\right)^{-1} = {}^{B}T_{A}$. Koordinat eksenleri arasındaki rotasyon matrisi 3 açı değeri ile belirtilebilir. X-Y-Z dönüşümüne göre




$\gamma, \beta, \alpha$ açıları ile belirtilen rotasyon için rotaston matrisi

$${}^{A}_{B}R_{XYZ}(\gamma, \beta, \alpha) = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix}.$$

olarak hesaplanabilir.

**Proje İstekleri:**

Proje kapsamında, iki farklı kameradan elde edilen nokta bulutlarının taban koordinat sistemine dönüştürülmesi ve tek bir nokta bulutu olarak elde edilmesi istenmektedir. Ayrıca nokta bulutu üzerinde bazı filtreleme işlemlerinin yapılabilmesi isteniyor. Aşağıda geliştirilecek sınıflar UML olarak verilmektedir. Üye fonksiyonlar için zorunlu olanlar verilmiştir. Bunun dışında gereken ya da sizin eklemek istediğiniz üye fonksiyon ve üye verileri sınıflara dahil edebilirsiniz.

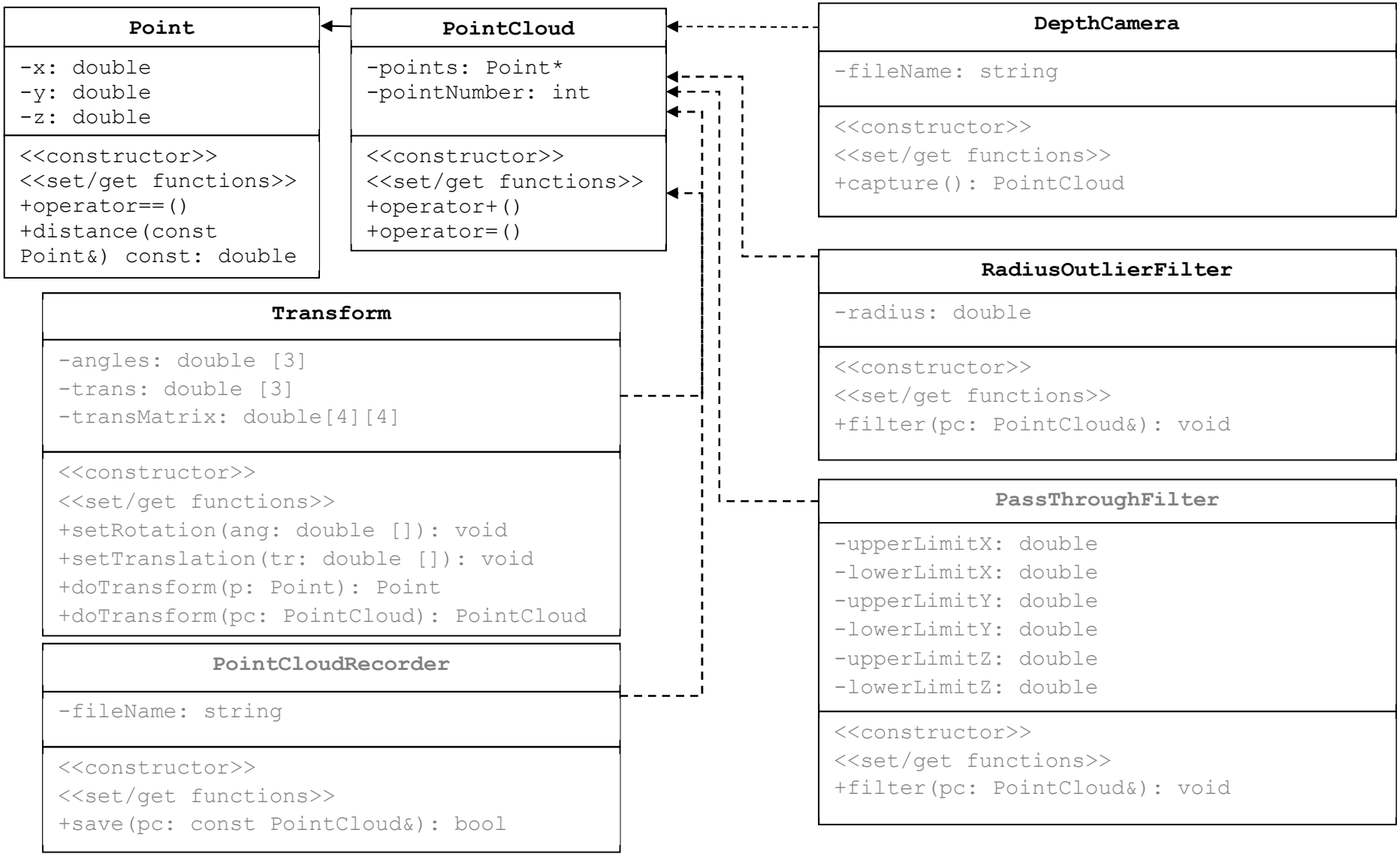
Şekil 3'de geliştirilmesi beklenen taslak UML sınıf diyagramı verilmektedir. Bu tasarımda, genel olarak sadık kalınmalı ancak üzerinde gerekli değişiklikler yapılabilir. Tasarımdaki sınıflar ve üye fonksiyonların açıklamaları kısaca şu şekildedir:

**Point Sınıfı:** Bu sınıf nokta bulutundaki 3B noktaların koordinatlarını tutar. Eşit operatörü (= =), iki noktanın eşit olup olmadığını denetler.

**PointCloud Sınıfı:** Sahip olduğu noktaları, dinamik olarak yaratılan bir Point dizisinde tutar. Dizinin boyutu, nesne yaratılırken constructor fonksiyonunda bir parametre olarak alınır. + operatörü, her iki nokta bulutunun sahip olduğu noktalara sahip tek bir nokta bulutunu döndürür. = operatörü, bir nokta bulutunun başka bir nokta bulutuna kopyalanmasını sağlar.

**Transform Sınıfı:** Açıklamlar kısmında anlatıldığı gibi, iki koordinat ekseni orijinleri arasındaki uzaklık (trans) ve rotasyon açılarını (angles) alır. Dönüşüm matrisini oluşturur (transMatrix). Daha sonrasında doTrans fonksiyonu ile alınan nokta ya da nokta bulutunu bu dönüşüme tabi tutarak dönüştürülmüş nokta ya da nokta bulutunu döndürür.

**DepthCamera Sınıfı:** İsmi (fileName) verilen dosyadan, capture fonksiyonu çağrıldığında noktaları okur ve yaratılan nokta bulutu nesnesine bu noktaları atar. Nokta bulutunu döndürür. Bu işlem kameranın bir benzetimidir. Noktalar bir kamera yerine dosyadan alınır.




**Şekil 3.** Geliştirilecek yazılımın taslak UML Sınıf Diyagramı

**RadiusOutlierFilter Sınıfı:** Bu nokta bulutunda filtreleme yapar. Filter fonksiyonu ile nokta bulutunu alır ve filtrelenmiş halini döndürür. Bu filtreleme işleminin algoritması şu şekildedir. Nokta bulutundaki her bir nokta için tek tek işlem yapılır. Noktaya, radius değerinden daha yakın başka bir nokta yok ise, bu nokta nokta bulutundan çıkarılır.

**PassThroughFilter Sınıfı:** Bu nokta bulutunda filtreleme yapar. Filter fonksiyonu ile nokta bulutunu alır ve filtrelenmiş halini döndürür. Bu filtreleme işleminde, noktanın x, y ve z değerlerinden en az birisi limitlerin dışında ise, bu nokta nokta bulutundan çıkarılır.

**PointCloudRecorder Sınıfı:** Bu nokta bulutlarının dosyaya kaydedilmesi için kullanılmaktadır. save fonksiyonu çağrıldığında, fileName ile ismi verilen dosya açılır, parametre olarak verilen nokta bulutundaki noktalar bu dosyaya kaydedilir.

Hem DepthCamera hemde PointCloudRecorder tarafından kullanılan metin tabanlı dosyada noktalar şu şekilde bulunmalıdır. Her satırda bir nokta olmak üzere, noktanın x, y, z bileşen değerleri aralarında boşlukla bulunacaktır.

```
X1 Y1 Z1
X2 Y2 Z2
X3 Y3 Z3
X4 Y4 Z4
.
.
.
Xn Yn Zn
```

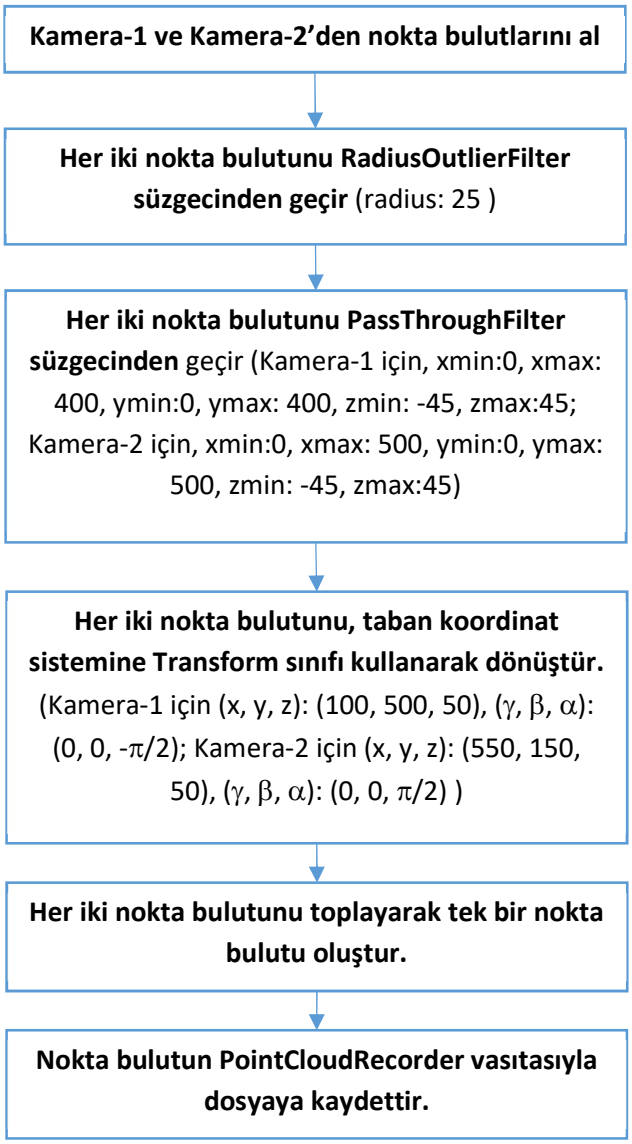
Nokta bulutunu 3B görselleştirmek için, CloudCompare (https://www.danielgm.net/cc/) programını bilgisayarınıza kurup, yukarıdaki formattaki dosyalarınızı bu programda açabilirsiniz. Nokta bulutunu 3B olarak görebilirsiniz.

**Kısım 1.**

Yukarıda verilen tasarıma uygun olarak bir Nokta Bulutu Kütüphanesi oluşturun. Her sınıf için doğruluğunu test etmek için bir test programı yazınız. Testler sırasında, Kısım 2'de verilen örnek değerlerden faydalanabilirsiniz.

**Kısım 2.**

Bir test uygulama programı yazınız. Bu test programında iki adet derinlik kamerası vardır. Derinlik kameralarından birincisinin (Kamera-1) koordinat ekseninin orijini, taban koordinat sistemine göre (x, y, z): (100, 500, 50) -translation- ve ($\gamma$, $\beta$, $\alpha$): (0, 0, -$\pi$/2) -rotation- olarak tanımlıdır. Derinlik kameralarından ikincisinin (Kamera-2) koordinat ekseninin orijini, taban koordinat sistemine göre (x, y, z): (550, 150, 50) -translation- ve ($\gamma$, $\beta$, $\alpha$): (0, 0, $\pi$/2) -rotation- olarak tanımlıdır. Kamera-1'den alınan nokta bulutu "camera1.txt" ve Kamera-2'den alınan nokta bulutu "camera2.txt" olacaktır.




## DOSYA EKLERİ:

**EK 1.** "camera1.txt" dosyası

**EK 2.** "camera2.txt" dosyası.