T.C.

ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

2021-2022 Eğitim-Öğretim Yılı

NESNE TABANLI PROGRAMLAMA I

PROJE RAPORU

Proje Başlığı

"Nokta Bulutu İşleme"

Projeyi Hazırlayanlar:

152120141056 Emre Satılmış

152120171106 Fatih Yünsel

152120191064 Bengisu Şahin

152120191093 Yiğit Efe Çoşgun

152120201049 Muhammet Eren Söme

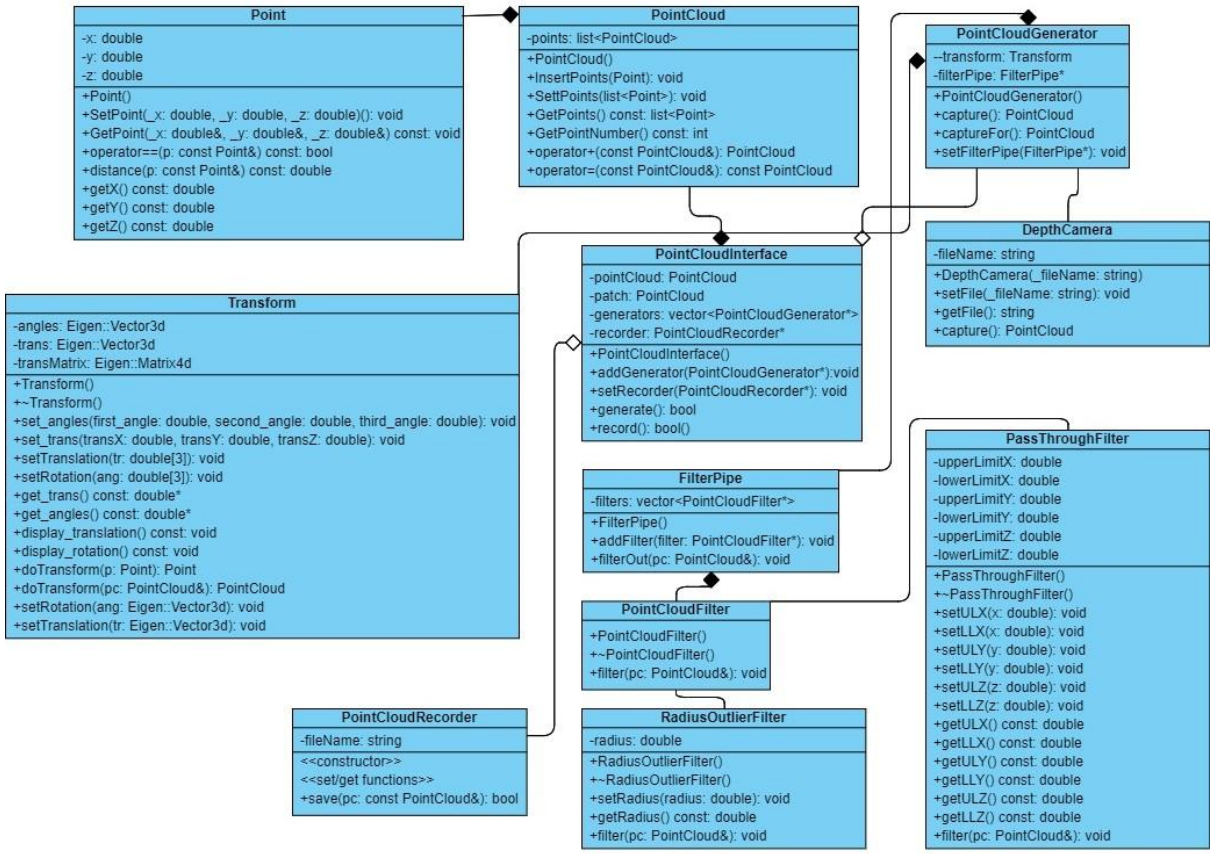
Ocak 2022

1. GİRİŞ

Derinlik kameraları, görüş açısı içindeki objelerin renkli piksel değerleriyle birlikte derinlik bilgilerini de sunar. Kameradan elde edilen üç boyutlu derinlik bilgisi üç boyutlu noktalarla ifade edilir. Bu noktaların oluşturduğu kümeye nokta bulutu (Point Cloud) adı verilir.
Her bir nokta, bulunduğu nokta bulutuna tanımlı koordinat sistemine göre tanımlıdır. Bu sebeple iki nokta bulutunu birleştirmek için her iki nokta bulutunu ortak bir koordinat sistemine göre tanımlı olacak şekilde dönüştürmek gerekir.
Bu proje kapsamında, iki farklı derinlik kamerasından elde edilen nokta bulutlarının taban koordinat sistemine dönüştürülerek tek bir nokta bulutunda birleştirilmesi ve oluşturulan nokta bulutunun filtrelenmesi amaçlanmaktadır.

2. TASARIM

UML Sınıf Diyagramı



Sınıflar

Point: Üç boyutlu nokta nesnesi oluşturmak içindir.
Üyeleri:

- **x**: Noktanın x eksenindeki değeri tutmak içindir. Double tipindedir. Dışarıdan direkt erişimi kısıtlamak için private tanımlanmıştır.
- **y**: Noktanın y eksenindeki değeri tutmak içindir. Double tipindedir. Dışarıdan direkt erişimi kısıtlamak için private tanımlanmıştır.
- **z**: Noktanın z eksenindeki değeri tutmak içindir. Double tipindedir. Dışarıdan direkt erişimi kısıtlamak için private tanımlanmıştır.

Metotları:

- **Point()**: Point sınıfın kurucu fonksiyonudur. Noktanın x, y, z değerlerine başlangıç olarak 0 değerini atar.
- **void SetPoint(double, double, double):** Noktanın x, y, z değerlerine aldığı üç parametreyi sırasıyla atar.
- **void GetPoint(double&, double&, double&) const:** Noktanın x, y, z değerlerini, parametre olarak gönderilen üç değişkene sırasıyla atar.
- **bool operator==(const Point&) const:** İki noktanın aynı olup olmadığını karşılaştırır. Aynı ise true, değilse false döndürür.
- **double distance(const Point&) const:** İki nokta arasındaki uzaklığı hesaplar ve hesaplanan değeri double tipinde döndürür.
- **double getX() const:** Noktanın x değerini döndürür.
- **double getY() const:** Noktanın y değerini döndürür.
- **double getZ() const:** Noktanın z değerini döndürür.

PointCloud: Point nesnelerini bir listede tutarak nokta bulutu (Point Cloud) oluşturmak içindir. PointCloud'ları toplama ve iki PointCloud'ı eşitleme özellikleri vardır.
Üyeleri:

- **points**: Nokta bulutundaki noktaları barındıran bir liste oluşturmak için Point tipinde bir pointer.
- **pointNumber**: Nokta bulutunu tutan listenin boyutunu belirlemek için int tipinde bir değişken.

Metotları:

- **PointCloud(int = 0)**: PointCloud sınıfın kurucu fonksiyonudur. Parametre olarak int tipinde bir değer alır ve bu değerle points listesini oluşturur. Bir atama yapılmazsa default olarak 0 değerini alır.
- **void SetPoints(int, Point):** Nokta bulutuna yeni bir nokta eklemek içindir. Noktanın listede bulunduğu dizin için int tipinde bir parametre ve noktayı temsilen Point tipinde bir parametre alır.
- **Point GetPoints(int) const:** Nokta bulutunu tutan listeden bir noktayı döndürür. Noktanın listedeki indeksini belirten int tipinde bir parametreyle kullanılır.

- **void SetPointNumber(int):** Nokta bulutunu tutan points listesinin boyutunu aldığı parametreyle değiştirir.
- **int GetPointNumber() const:** Nokta bulutunu tutan liste boyutunu döndürür.
- **PointCloud operator+(const PointCloud&):** İki nokta bulutunu birleştirir. Nokta bulutlarının boyutlarını toplar ve toplam boyutunda yeni bir PointCloud oluşturup bu PointCloud'a her iki nokta bulutunun sahip olduğu noktaları ekler.
- **const PointCloud &operator=(const PointCloud&):** PointCloud'u başka bir PointCloud'a kopyalar.
- **void DeletePoints():** Points'in bellekte tuttuğu alanı serbest bırakır.

PointCloudInterface: PointCloudGenerators nesnelerini bir vectorde tutar ve her biri icin captureFor metodunu uyguladıktan sonra dönen nokta bulutunu pointCloud üyesinde birleştirir. Yapılan değişikliklerden sonra oluşan pointCloud nesnesini PointCloudRecorder araciligiyla yeni bir dosyaya kaydeder.

Üyeleri:

- **pointCloud: PointCloud :** Toplam nokta bulutu.
- **patch: PointCloud :** pointCloud'a eklenecek nokta bulutu.
- **generators: vector :** nesnelerini tutmak için vector.
- **recorder: PointCloudRecorder* :** PointCloudRecorder nesnesi.

Metotları:

- **addGenerator(PointCloudGenerator*):void :**Generators üyesine yeni bir PointCloudGenerator nesnesi ekler.
- **setRecorder(PointCloudRecorder*): void** :Recorder üyesine bir PointCloudRecorder nesnesi atar.
- **generate(): bool:** Generators üyesinde bulunan her bir nesne için captureFor metodunu uygular ve dönen nokta bulutunu pointCloud üyesine ekler.
- **record(): bool :**pointCloud üyesinde tanimlanan nokta bulutunu recorder uyesi üzerinden yeni bir dosyaya kaydeder.

PointCloudGenerator: Altına DephtCamera sınıfını alan bir abstract classdır.

Üyeleri:

- **Transform transform:** private erişimli bir Transform class nesnesidir
- **filterPipe FilterPipe:**private erişimli bir Filterpipe class nesnesidir
- **PointCloudGenerator():**PointCloudGenerator sınıfı için constructor.
- **~PointCloudGenerator():** PointCloudGenerator sınıfı için destructor.

Metotları:

- **virtual PointCloud capture():** DephtCamera sınıfındaki capture fonksiyonunu altına alan bir virtual metottur.
- **virtual PointCloud captureFor():** DephtCamera sınıfındaki captureFor fonksiyonunu altına alan bir virtual metottur.
- **void setFilterPipe(FilterPipe*):** private üye olan filterPipe değişkeninin kapsülleme yöntemi ile erişime açar.

Dephtcamera: Txt dosyalarından alınan x,y ve z koordinatlarını çeşitli filtrelerden geçirerek buluta gönderimini sağlar.

Üyeleri:

- **fileName:** Nesne olşturulduğunda girilen txt dosyasının adıdır. O txt dosyasının içinden nokta alımı yapılır.

Metotları:

- **DephtCamera(string):**DephtCamera sınıfı constructor fonksiyondur . Txt dosyasının adını alır.
- **void setFile(string):** private üye olan fileName değişkeninin kapsülleme yöntemi ile erişime açar.
- **string getFile():**private üye olan fileName değişkenini döndürür.
- **PointCloud capture():**fileName içine alınan txt dosyasını okur txt dosyasının içindeki koordinat noktalarını Point classı yardımıyla PointCloud nesnesine atar ve PointCloud nesnesini döndürür.
- **PointCloud captureFor():** öncelikle capture fonksiyonunun işlevini yerine getirdikten sonra RadiusOutlierFilter sonra PassThroughFilter filtrelerinden geçirdikten sonra Transform sınıfındaki doTransform metoduyla dönüşümü gerçekleştirir dönüşümden çıkan PointCloud nesnesini döndürür.

Transform: Dönüşüm matrisini oluşturarak , doTransform fonksiyonu ile alınan nokta ya da nokta bulutunu dönüştürürür ve döndürür.

Üyeleri:

- **angles: Eigen::Vector3d:**Rotasyon açılarını tutar.
- **trans: Eigen::Vector3d :**İki ayrı koordinat sisteminin orijinleri arasındaki fark değerlerini tutar.
- **transMatrix: Eigen::Matrix4d:**Rotasyon matrixi ve trans matrixi kullanılarak oluşturulan transMatrix ini tutar.

Metotları:

- **Transform():** Transform sınıfının yapıcı fonksiyonudur.
- **~Transform():**Transform sınıfının yıkıcı fonksiyonudur.
- **void setAngles(double, double, double)** :Açı değerlerini alarak transform nesnesinde set etmemizi sağlar **.**
- **void setTrans(double,double,double):** İki ayrı koordinat sistemi orijinlerinin noktaları arasındaki uzaklığı trans matrixine atar.
- **void setTranslation(double tr[3]):** Trans matrixini ve üç boyutlu rotation matrixini kullanarak translation matrixini oluşturur.
- **void setTranslation(tr: Eigen::Vector3d):**Trans vektörünü ve 3 boyutlu rotation matrixini kullanarak translation matrixini oluşturur.
- **void setRotation(double ang[3]):** Angles matrixinde tutulan açı değerlerini kullanarak rotation matrixini oluşturur.
- **void setRotation(ang: Eigen::Vector3d):** Angles vektöründe tutulan açı değerlerini kullanarak rotation matrixini oluşturur.
- **double* get_trans() const:** Trans matrixini döndürür.
- **double* get_angles() const:** Angles matrixini döndürür.
- **void display_translation() const:** Translation matrixini konsol ekranına yazdırır.
- **void display_rotation() const:** Rotasyon matrixini konsol ekranına yazdırır.
- **Point doTransform(Point p):** Alınan nokta nesnesini translation matrixini kullanılarak dönüştürülmesini sağlar ve dönüştürülmüş noktayı döndürür.
- **PointCloud doTransform(PointCloud &pc):**Alınan nokta bulutu nesnesinin her bir noktasının translation matrixi kullanılarak dönüştürülmesini sağlar ve dönüştürülmüş nokta bulutunu döndürür.
- **list<Point> doTransform(list<Point> lp):** Alınan nokta list indeki noktaları tek tek dönüştürür. Her dönüştürülmüş noktayı yeni list e ekler ve yeni listi döndürür.

PointCloudFilter: Filtrelerin altinda toplandigi Abstract bir siniftir. PassThroughFilter ve RadiusOutlierFilter sınıfları bu sınıftan miras alır.

Üyeleri:

- **PointCloudFilter():** PointCloudFilter sınıfının yapıcı fonksiyonudur.
- **~PointCloudFilter():** PointCloudFilter sınıfının yıkıcı fonksiyonudur.

- **virtual void filter(PointCloud&):** Pure virtual filter fonksiyonudur. Miras bırakılan alt sınıflarda doldurulması gerekir.

FilterPipe: Birden fazla filtreden geçmesi gereken nokta bulutunun filtreleme işlemini yapacak sınıftır. Bu nesneye, farklı tip filtreler ya da farklı parametrelere sahip aynı tip filtreler addFilter fonksiyonu ile eklenecektir. Daha sonra filterOut fonksiyonuna verilen nokta bulutu, ekleme sırasında göre birer birer filtrelerden geçirilip, sonuç nokta bulutunu döndürecektir.

Üyeleri:

- **vector<PointCloudFilter*> filters:** Filtrelerin tutulduğu vector.

Metotları:

- **FilterPipe():** FilterPipe sınıfının yapıcı fonksiyonudur.
- **~FilterPipe():** FilterPipe sınıfının yıkıcı fonksiyonudur.
- **void addFilter(PointCloudFilter*):** Bu fonksiyon farkli tip filtreler ya da farklı parametrelere sahip ayni tip filtreleri filters vectorune ekler.
- **void filterOut(PointCloud&):** Bu fonksiyon nokta bulutunu ekleme sirasina gore filtrelerden geçirip sonuç nokta bulutunu döndürür.

RadiusOutlierFilter: Bu sınıf nokta bulutunda filtreleme yapar. Filter fonksiyonu ile nokta bulutunu alır ve filtrelenmiş hâlini döndürür. 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.

Üyeleri :

- **double radius :** Yarıçap değerini tutan değişkendir.

Metotları :

- **RadiusOutlierFilter() :** RadiusOutlierFilter sınıfının yapıcı fonksiyonudur.
- **~RadiusOutlierFilter() :** RadiusOutlierFilter sınıfının yıkıcı fonksiyonudur.
- **void setRadius(double) :** Fonksiyon double tipinde bir parametre alır ve bu değeri objenin radius değişkenine atar.
- **double getRadius() const :** Objenin radius değerini geri döndüren fonksiyondur.
- **void filter(PointCloud&) :** Filtrelemenin gerçekleştiği fonksiyondur. İçine parametre olarak referans tipinde PointCloud objesi alır. Bu PointCloud objesini filtreler ve filtrelenmiş halini kaydeder.

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

Üyeleri :

- **double upperLimitX :** X noktası için üst değer limitini belirler.
- **double lowerLimitX :** X noktası için alt değer limitini belirler.

- **double upperLimitY :** Y noktası için üst değer limitini belirler.
- **double lowerLimitY :** Y koordinatı için alt değer limitini belirler.
- **double upperLimitZ :** Z koordinatı için üst değer limitini belirler.
- **double lowerLimitZ :** Z koordinatı için alt değer limitini belirler.

Metotları :

- **PassThroughFilter() :** PassThroughFilter sınıfının yapıcı fonksiyonudur.
- **~PassThroughFilter() :** PassThroughFilter sınıfının yıkıcı fonksiyonudur.
- **void setULX(double) :** Double tipinde parametre alır ve upperLimitX değerini bu parametreye eşitler.
- **void setLLX(double) :**Double tipinde parametre alır ve lowerLimitX değerini bu parametreye eşitler.
- **void setULY(double) :**Double tipinde parametre alır ve upperLimitY değerini bu parametreye eşitler.
- **void setLLY(double) :**Double tipinde parametre alır ve lowerLimitY değerini bu parametreye eşitler.
- **void setULZ(double) :**Double tipinde parametre alır ve upperLimitZ değerini bu parametreye eşitler.
- **void setLLZ(double) :**Double tipinde parametre alır ve lowerLimitZ değerini bu parametreye eşitler.
- **double getULX() const :** upperLimitX değerini geri döndürür.
- **double getLLX() const :** lowerLimitX değerini geri döndürür.
- **double getULY() const :** upperLimitY değerini geri döndürür.
- **double getLLY() const :** lowerLimitY değerini geri döndürür.
- **double getULZ() const :** upperLimitZ değerini geri döndürür.
- **double getLLZ() const :** lowerLimitZ değerini geri döndürür.
- **void filter(PointCloud& pc) :** Filtrelemenin gerçekleştiği fonksiyondur. İçine parametre olarak referans tipinde PointCloud objesi alır. Bu PointCloud objesini filtreler ve filtrelenmiş halini kaydeder.

PointCloudRecorder: Nokta bulutlarının dosyaya kaydedilmesini sağlar.

Üyeleri :

- **string filename:** Dosya adını tutar.

Metotları :

- **PointCloudRecorder():** PointCloudRecorder sınıfının yapıcı fonksiyonudur.
- **void setFileName(string):** Dosya adını atar.
- **string getFileName():** Dosya adını döndürür.
- **bool save(const PointCloud&):** fileName ismi ile verilen dosya açılır ve parametre olarak verilen nokta bulutundaki noktaların dosyaya kaydedilmesini sağlar.

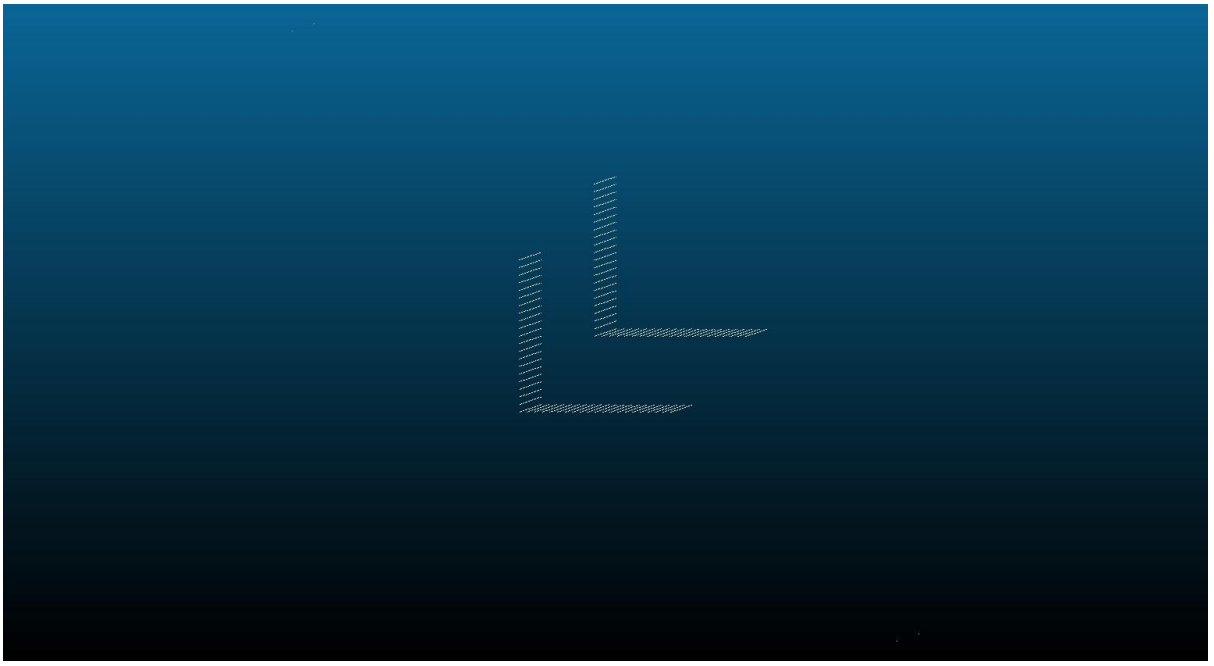
Görev Atamaları

Grup Üyesi	Görevler
Emre Satılmış	Point, PointCloud ve PointCloudInterface sınıfları.
Fatih Yünsel	DepthCamera PointCloudGenerator sınıfı.
Bengisu Şahin	Transform sınıfı.
Yiğit Efe Coşgun	PointCloudRecorder sınıfı.
Muhammet Eren Söme	RadiusOutlierFilter PassThroughFilter FilterPipe sınıfları.

3. SONUÇLAR

Proje C++ dilinde nesne tabanlı programlama ilkelerince çeşitli sınıflar oluşturularak yapılmıştır. Bu sınıfların gerçeklenmesi grup üyelerine eşit bir şekilde dağıtılmıştır. Bu sayede takım çalışması kolaylaştırılmıştır. Yazılan kodların paylaşılması ve diğer grup üyeleri tarafından kontrol edilebilmesi için sürüm kontrol sistemi olarak Bitbucket kullanılmıştır.
Repository Linki : https://bitbucket.org/erennsome/oop_1_hw/src/main/

Input noktalarının CloudCompare içindeki görüntüsü:



Input noktalarının programdan geçirildikten sonra çıkan output noktalarının CloudCompare içindeki görüntüsü: