ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

NESNE TABANLI PROGRAMLAMA I

PROJE RAPORU -2

Proje Başlığı

“: 3B Izgara Tabanlı Haritalama (3D Grid-Based Mapping)”

Projeyi hazırlayanlar:

152120201098 Emre Kart
152120201039 Hakan Yavaş
152120201054 Alperen Güneş
151820181058 Mine Akgül

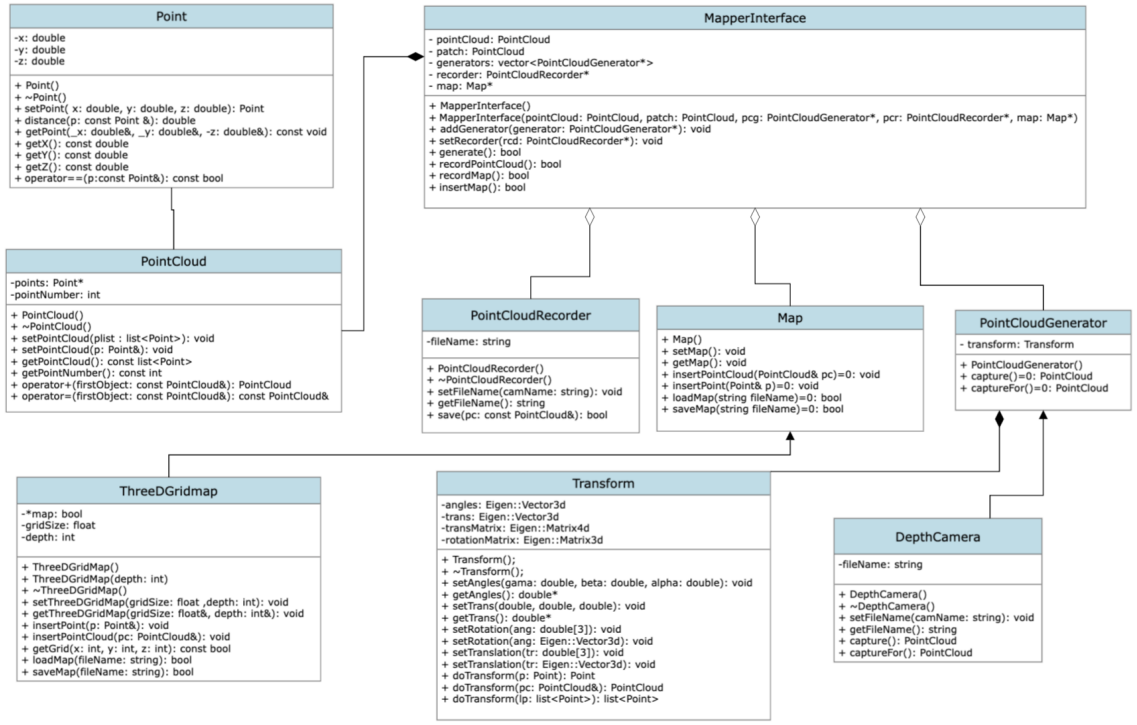
OCAK 2023

1.GİRİŞ

Verilen 4 tane derinlik kamerasından alınan koordinat verileri ve renkli piksel değerleriyle birlikte okutup kameralardan aldığımız 3 boyutlu derinlik bilgileri ile 3 boyutlu çıktı elde etmemiz gerekmektedir. Fakat bu noktalar farklı koordinat düzlemlerinde her bir kameradan alınan verilerin koordinat düzlemleri kameraların yerleri farklı olduğu için koordinat düzlemi de farklı bu nedenle bu nokta bulutlarını tek bir base koordinat sistemine taşıyıp bu koordinat düzleminde 3 boyutlu çıktıyı almamız lazım bunun için bu nokta bulutlarını dönüşüm matrisi ile tek bir koordinat düzlemi üzerine toplarız ve son durumda elde ettiğimiz dosyadaki verileri CloudCompare programında test ederiz.

2.TASARIM

UML CLASS DİYAGRAMI



Sınıflar

Point: bir noktanın koordinat düzlemindeki x, y ve z noktalarını atamak içindir.

Üyeleri:

- double x : Noktanın koordinat düzleminde x ekseni değerini tutmak içindir. private olarak tanımlanmıştır dışardan erişilemez.
- double y : Noktanın koordinat düzleminde y ekseni değerini tutmak içindir. private olarak tanımlanmıştır dışardan erişilemez.
- double z : Noktanın koordinat düzleminde z ekseni değerini tutmak içindir. private olarak tanımlanmıştır dışardan erişilemez.

Metotlar:

+ Point() : Point sınıfının constructor fonksiyonu, başlangıçta noktanın koordinatlarını (x,y,z) değerlerini (0,0,0) atar.
+ ~Point() : Point sınıfının destructor fonksiyonu.
+ Point setPoint(double x, double y, double z) : Aldığı x,y,z değerlerini noktanın x,y,z değerlerine atar.
+ void getPoint(double& _x, double& _y, double& _z) const : noktaların x y z değerlerini görmemizi sağlar.
+ double distance(const Point &); iki nokta arasındaki uzaklığı verir.
+ double getX() const : noktanın x değerini döndürür.
+ double getY() const : noktanın y değerini döndürür.
+ double getZ() const : noktanın z değerini döndürür.
+ bool operator==(const Point&) const : iki noktanın eşit olup olmadığını denetleyen bir operatördür.

PointCloud: Point sınıfından nesneleri alarak onları bir listede tutarak nokta bulutu oluşturur.

Üyeleri:

- points: list<pointCloud> point noktalarını içinde bulunduran nokta bulutudur. Vektör tabanlı bir listede point değerleri tutulur.

Metotlar:

+ PointCloud() : PointCloud sınıfı için constructor
+ ~PointCloud() : Point Cloud sınıfı için destructor.
+ void setPointCloud(list<Point>) : pointclouda bir point ekler.
+ void setpointCloud(Point&p): pointclouda bir point ekler.
+ list<Point> getPointCloud() **const**: point listesini döndürür.
+ int getPointNumber() const : point listesinin boyutunu döndürür
+ PointCloud operator+(const PointCloud &firstObject) : PointCloudları birleştirir
+ const PointCloud &operator=(const PointCloud &firstObject) : bir PointCloudu diğerine eşitler

DepthCamera: Adı verilen txt dosyasını okur ve içinden x,y,z koordinatlarını alarak pointclouda gönderir bu noktaları pointcloudrecorderda kayıt ederiz.

Üyeleri:

- string fileName : girilen txt dosyasının adıdır, o txt dosyasından verileri alırız.

Metotlar:

+ depth camera
+ DepthCamera() : DepthCamera sınıfı için constructor
+ ~DepthCamera() : DepthCamera sınıfı için destructor
+ void setFileName(string camName) : Bu fonksiyon, dosya adini fileName degiskenine set eder.
+ string getFileName() : Bu fonksiyon fileName değişkenini return eder.
+ virtual PointCloud capture() : bu fonksiyon dosyayı açar okur ve her bir noktayı x y z şeklinde pointclouda set eder
+ virtual PointCloud captureFor() : capture ile başlangıçta aynı şeyleri yapar sonrasında nokta bulutu, üyesi olan transform nesnesi ile dönüştürüldükten sonra döndürülecektir.

PointCloudRecorder: Bu sınıfı bir nokta bulutunun verilerini bir dosyaya çıktı olarak verir.

Üyeleri:

- string fileName : Bu fonksiyon kaydedeceğimiz dosyaya vereceğimiz ismi atayabileceğimiz değişkendir.

Metotlar:

+ PointCloudRecorder() : bu fonksiyon PointCloudRecorder sınıfının constructoru
+ ~PointCloudRecorder() : bu fonksiyon PointCloudRecorder sınıfının destructoru
+ void setFileName(string) : bu fonksiyon kaydedilecek dosyaya isim vermemizi sağlar
+ string getFileName() : bu fonksiyon kaydettiğimiz dosyaya verdiğimiz ismi döndürür
+ bu fonksiyon nokta bulutlarını dosyaya kaydeder
+ bool save(const PointCloud &pc);

Transform: Transform sınıfı bize verilen koordinatları base koordinat sisteminde transformasyon matrisini uyulayıp base kordinat sistemindeki yeni noktaları bulmamızı sağlar

Üyeleri

- double angles[3] : açıları tuttuğumuz array
- double trans[3] : translation matrix için iki orijin arasındaki farkı tuttuğumuz array double transMatrix[4][4] : translation matriksimiz.
- double rotationmatrix[3][3] : rotation matriksimiz

Metotlar:

+ Transform() : Transform sınıfının constructor fonksiyonu
+ ~Transform() : Transform sınıfının destructor fonksiyonu
+ void setAngles(double, double, double) : Bu fonksiyon alinan aci değerlerini set eder.
+ double* getAngles() : Bu fonksiyon alınan açı değerlerini gördüğümüz fonksiyon.

+ void setTrans(double, double, double) : Bu fonksiyon iki koordinat düzleminin orijinleri arasındaki uzaklıkları trans arraye değişken olarak atar.
+ double *getTrans() : Burada iki koordinat düzlemi arasında uzaklık değerlerini veren x y z değerlerini döndürür.
+ void setRotation(double ang[3]) : Bu fonksiyon set edilen angle değerlerini kullanarak rotation matrix i doldurur
+ void setRotation(double rotationmatrix[3][3] : Bu fonksiyon bize verilen rotasyon matrisini rotationmatrixe set eder .
+ void setTranslation() : bu fonksiyon 4x4 transform matrisini oluşturur bunun için rotasyonmatrix ve trans nesnesinin x y z sini kullanır.
+ Point doTransform(Point p) : bu fonksiyonda 4x4 transmatrix ile 4x1 BP matriki çarpılır sonuçlar oluşturulan yenipointe set edilir ve yenipoint döndürülür.
+ PointCloud doTransform(PointCloud &pc) : bu fonksiyon nokta bulutunu dönüştürür, dönüştürülmüş nokta bulutunu döndürür.
+ void displayTranslation() const : translation matrisi görmemizi sağlar.
+ void displayRotation() const; } : rotation matrisi görmemizi sağlar.

ThreeDGridmap: Bu sınıf, 3B ızgara tabanlı harita oluşturmaktadır. Burada, Şekil 4'de görüldüğü gibi verine nokta bulutu ızgara tabanlı bir haritaya dönüştürülecektir. Şekil 5'de görüldüğü gibi haritada ortam eşit büyüklükteki küpler ile temsil edilmektedir. Bu, küpler map[][][] ile temsil edilmektedir.

Üyeleri:

- bool *map : depth e göre oluşturduğumuz 3 boyutlu harita true nokta dolu false boş
- float gridSize : mapin içindeki dolu birimlerin boyutları
- int depth : Depth x,y ve z ekseni doğrultularındaki küp sayısını belirtmektedir.

Metotlar:

+ ThreeDGridMap(): Bu fonksiyon ThreeDGripMap için constructor fonksiyonu
+ ThreeDGridMap(int depth): Bu fonksiyon ThreeDGripMap için constructor fonksiyonu
+ ~ThreeDGridMap(): Bu fonksiyon ThreeDGripMap için destructor fonksiyonu
+ void setThreeDGridMap(float gridSize, int depth): Depth'te uygun bir 3dgrid amp set ettik.
+ void getThreeDGridMap(float &gridSize, int &depth): Bu fonksiyonda gridSize ve depth alınır.
+ void insertPoint(Point &p): Bu fonksiyon noktalarımızı true yapar
+ void insertPointCloud(PointCloud &pc): Bu fonksiyon nokta bulutlarını true yapar
+ bool getGrid(int x, int y, int z) const: getGrid fonksiyonu ilgili grid için değer döndürmektedir.
+ bool loadMap(string fileName);
+ bool saveMap(string fileName);

PointCloudGenerator:

Üyeleri:

- Transform transform();

Metotlar:

+ PointCloudGenerator(){}
+ virtual PointCloud capture()=0;
+ virtual PointCloud captureFor()=0;

Map:

Metotlar:

+ Map() : yapıcı fonksiyon
+ void setMap(); : Haritayı ayarlamaya yönelik bir işlem yapar.
+ void getMap(); : mevcut haritayı döndürür.
+ virtual void insertPointCloud(PointCloud& pc)=0; :Verilen PointCloud (pc) nesnesini haritaya ekler. pc: haritaya eklenecek PointCloud (noktalar bulutu) nesnesi.
+ virtual void insertPoint(Point& p)=0: Verilen Point (p) nesnesini haritaya ekler. p: haritaya eklenecek Point (nokta) nesnesi.
+ virtual bool loadMap(string fileName)=0; : Belirtilen dosya adıyla bir harita yükler ve bu haritanın yüklendiğini doğrular. fileName: yüklenecek haritanın dosya adı.
+ virtual bool saveMap(string fileName)=0; : Mevcut haritayı belirtilen dosya adıyla kaydeder ve bu haritanın kaydedildiğini doğrular. fileName: kaydedilecek haritanın dosya adı.

MapperInterface:

Üyeleri:

- PointCloud pointCloud; :
- PointCloud patch;
- vector<PointCloudGenerator*> generators;
- PointCloudRecorder* recorder;
- Map* map;

Metotlar:

+ MapperInterface() : *constructor*
+ MapperInterface(PointCloud pointCloud, PointCloud patch, PointCloudGenerator* : pcg, PointCloudRecorder* pcr, Map* map) : constructor
+ void addGenerator(PointCloudGenerator* generator); : *PointCloudGenerator nesnesi ekler*

+ virtual void setRecorder(PointCloudRecorder* rcd); : *PointCloudRecorder nesnesi ayarlar*
+ virtual bool generate(); : *nokta bulutunu oluşturur*
+ virtual bool recordPointCloud(); *Nokta bulutunu kaydeder*
+ virtual bool recordMap(); *haritaya kaydeder*
+ virtual bool insertMap(); *haritayı yükler*

Görev Atamaları

GRUP ÜYELERİ	Görevler
Emre Kart	**transform güncelledi (eigen)**
Hakan Yavaş	**pointCloud'a liste ekledi ve bütün kodu liste göre güncelledi, MapperInterface sınıfının görevlerini tamamladı.**
Alperen Güneş	**rapor ve dokümantasyon, PointCloudGenerator class'ını tamamladı**
Mine Akgül	**3dgridmap**
Aslı Esen	**Katılım sağlamadı**

3.SONUÇLAR

Proje sonucunda 4 farklı derinlik kamerasından alınan koordinatların dönüşüm matrisi yardımıyla tek bir base koordinat düzlemine aktarılıp o noktalarla 3 boyutlu nokta bulutu çıktısını almayı öğrendik. Birinci aşamada yaptığımızı daha gelişmiş bir sisteme göre yaptık bu aşamada aggregation composition association ilişkilerine göre yeni sınıflar ekleyerek ve birbirlkeri arasındaki bağlantıları değşştirerek güncelledik. Abstract sınıflar kullandık. Görev dağılımını yukarıda belirtildiği gibi sınıfları dağıtarak yaptık BitBucket kullanarak grup ödevlerinin daha verimli yapıldığını öğrendik bu sayede takım çalışmasını kolaylaştırdık birbirimize hata yaptığımız yerlerde daha rahat yardımcı olabildik.

BitBucket linkleri:1.aşama https://bitbucket.org/hakanyavas/oop/src/main/ (çakışma problemi nedeniyle başka bir repoya taşıdık işlemlerimizi)
1.aşama devamı https://bitbucket.org/hakanyavas/oop_recovered/src/master/ (yeni repo)
2.aşama : https://bitbucket.org/hakanyavas/oop_project_step2/src/master/ (yeni repo)

Program Çıktıları: