NESNELERİN ÜÇ BOYUTLU MODELLENMESİ İÇİN KİNECT TABANLI BİR UYGULAMA

Erdal ÖZBAY

Yüksek Lisans Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Danışmanı: Yrd. Doç. Dr. Ahmet ÇINAR

MAYIS- 2013

T.C.

FIRAT ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

NESNELERİN ÜÇ BOYUTLU MODELLENMESİ İÇİN KİNECT TABANLI BİR UYGULAMA

YÜKSEK LİSANS TEZİ

ERDAL ÖZBAY

(101129101)

Tezin Enstitüye Verildiği Tarih : 27.05.2013

Tezin Savunulduğu Tarih : 22.05.2013

Tez Danışmanı : Yrd. Doç. Dr. Ahmet ÇINAR (F.Ü.)

Diğer Jüri Üyeleri : Yrd. Doç. Dr. Ahmet ÇINAR (Danışman)

Yrd. Doç. Dr. Galip AYDIN

Doç. Dr. Arif GÜLTEN

MAYIS- 2013

ÖNSÖZ

Bu tezde üç boyutlu gerçek dünya nesnelerinin yeniden yapılandırılması konusu ile ilgili algoritmalar geliştirilmiştir. Yaptığım bu tez çalışmasının, konu ile ilgilenen lisans, yüksek lisans ve doktora öğrencilerinin yanı sıra bu konuda çalışmak isteyenlere bir rehber olacağını umuyorum.

Yüksek Lisans eğitimim boyunca hem akademisyenlik mesleğinde hem de özel hayattaki yaklaşımlarını örnek aldığım, mesleki engin bilgi ve deneyimlerini cömertçe benimle paylaşmaktan kaçınmayan, her konuda sonsuz desteğini yanımda hissettiğim saygıdeğer hocam Yrd. Doç. Dr. Ahmet ÇINAR'a ve bu süreç içerisinde her konuda sabırla bana destek veren değerli aileme çok teşekkür ederim.

Erdal ÖZBAY
ELAZIĞ-2013

İÇİNDEKİLER

Sayfa No

ÖZET

Günümüzde insan gözünün gördüğü şekliyle, üç boyutlu gerçek dünya nesnelerini, iki boyutlu bir görüntü yapısı ile içeriğini yeniden düzenleyip, yeniden yapılandırmak için çözüm üretmek son derece zordur. İnsan gözünün görsellik yapısı özelliği itibariyle iki boyuttan, (2B) üç boyuta (3B) dönüşüm gerçekleştirme gibi bir zorunluluğu yoktur, bu özellik insan için görsel dünyayı anlamak adına olağanüstü bir yetenektir. Bilgisayar görmesi, sanal gerçeklik, nesne tanınması, arttırılmış gerçeklik, yeniden yapılandırma görsel dünyanın mantıksal yaklaşımını anlamak için insanların bu tür yeteneklerini taklit etmek istemektedir. Fakat gerçek dünya nesnelerini 3B olarak bilgisayar ortamında yeniden yapılandırılması oldukça zahmetlidir.

Bu tez çalışmasının arka planında, temel olarak fiziksel dünya nesneleri ile sanal uzay arasında, çeşitli metotlar kullanarak yeni bir etkileşimin sağlanması yatmaktadır. Bu açıdan amaç, kapalı bir sahnenin, detaylı olarak yeniden yapılandırılması için, gerçek dünya nesnelerinin, hareketli olarak 360 derecelik tarama üretme kapasitesine sahip bir standart 3B kamera ile oluşturulacak olan sanal alanlar içerisinde, nesnenin 3B modelinin görüntülenmesini sağlamaktır. Bu hedefin uygulanmasında öncelikle, fiziksel nesnelerin 360 derecelik sağlam bir yapıda oluşturulması için, nesnenin tam derinlik verilerinin elde edilmesi aşamasında, birden fazla tarama bütünü hakkındaki mevcut yöntemlerin incelenmesi ile ilgili araştırma yapılarak işe başlandı.

Bu tez çalışmasında Microsoft Kinect ve açık kaynak kodlu nokta bulutu kütüphanesi kullanılarak, yakalanan gerçek dünya nesnelerinin nokta bulutu ile yeniden yapılandırılması sırasında en iyi sonuçların elde edilmesi için bu noktaların işlenmesi üzerine odaklanır.

Sonuç olarak bu tez çalışmasında, düzlemsel yüzeyler üzerinde sabit olarak yerleşik durumdaki gerçek dünya nesnelerinden, hareketli olarak birden fazla alınmış olan nokta bulutu kümesi (nbk) derinlik taramaları üzerine, nesneye ve arka planlarına uygulanacak olan çeşitli filtreleme yöntemleri incelenmiş, bu verilerin üçgenleme yöntemiyle birleştirildikten sonra MeshLab ve OpenGL sanal görüntüleme ortamlarında görselleştirilmiştir.

Anahtar Kelimeler: Yeniden yapılandırma, 3B, Nokta bulutu, Kinect kamera, Modelleme

SUMMARY

A Kinect-Based Application For Modelling Three-Dimensional Objects

It is very difficult to generate solutions to reconstruction the three-dimensional real world objects in the form human eye sees by reorganizing their content with the aid of two-dimensional image structure. There no necessity for the human eye to perform a conversion from 2-dimension (2D) to 3-dimension (3D) in respect of its visual quality; this characteristic is an extraordinary ability for the human to recognize the visual world. In order to understand the logical approach of the visual world, such abilities of human are imitated for computer vision, virtual reality, object recognition, augmented reality, restructuring. However, it is fairly troublesome to reconstruction the real world objects in the computer medium in the form of 3D.

Basically, realization of a new interaction between the physical world objects and virtual space by utilizing various methods lies behind the background of this thesis study. From this viewpoint, purpose is to provide visualization of 3D model of the real world object in virtual spaces to be constituted by means of a standard 3D camera that has the capacity to generate a scanning of 360 degrees animatedly in order to reconstruction a closed stage in detail. Studies started by making investigation about the review of existing methods on multiple entire scanning in the stage of obtaining overall depth inputs of the object in order to shape the physical objects in a solid structure of 360 degrees.

In order to obtain the best results during restructuring of world objects with point cloud, we focused on processing of these points in this study by utilizing Microsoft Kinect and open-source code point cloud library.

Consequently, various filtering methods to be applied to the object and its backgrounds were reviewed from real world resident objects on planar surfaces over to depth scans of point cloud library (pcl) taken more than once in animated form; they are visualized in MeshLab and OpenGL visualization platforms after these inputs are combined by triangulation method.

Key Words: Reconstruction, 3D, Point cloud, Kinect camera, Modelling

ŞEKİLLER LİSTESİ

Sayfa No

KISALTMALAR

3B : Üç Boyutlu.
2B : İki Boyutlu.
RGB : Kırmızı, Yeşil, Mavi bileşenleri.
NBK : Nokta Bulutu Kümesi.
SDK : Yazılım Geliştirme Kiti.
OpenGL : Açık Grafik Kütüphanesi
TOF : Uçuş Zamanı
RGB-D : Kırmızı, Yeşil, Mavi Derinlik Bileşenleri.
ICPEn : Yakın Nokta Tekrarlama.
GPU : Grafik İşlem Birimi.
CMOS : Tamamlamalı Metal-oksit Transistör
IR : Kızılötesi.
BSD : Brekel Yazılım Geliştirme.
ROS : Robot İşletim Sistemi.
OpenMP : Açık Çoklu İşletim.
OpenNI : Açık Doğal Etkileşim.
SURF : Sağlam Özellikler Hızlandırma.
RANSAC : Rastgele Örneklem Onayı.
SVD : Tekil Değer Ayrıştırması.
VTK : Görselleştirme Araç kutusu
PSDK : Platform Yazılım Geliştirme Kiti.
DC : Doğru Akım
GPL : Genel Kamu Lisansı.

1. GİRİŞ

İki boyutlu bir görüntü yapısı ile üç boyutlu gerçek dünya nesnelerinin, insan gözünün gördüğü şekliyle, üç boyutlu olarak yeniden yapılandırılması ve bu içeriği yeniden düzenlemek için çözümler bulunması oldukça zahmetli işlemler gerektirir. İnsan gözünün görme özelliği, doğadaki görüş açısı içerisinde bulunan cisimleri iki boyuttan üç boyuta dönüştürmeyi gerçekleştirmek zorunda değildir. Bu özellik görsel dünyayı anlamak için insanlık adına olağanüstü bir yetenektir. Bilgisayar görmesi, sanal gerçeklik, nesne tanıma, artırılmış gerçeklik, 3B yeniden yapılandırma gibi bilgisayarlı grafik uygulamaları mantıksal bir yaklaşım ile görsel dünyayı anlamak için insanların bu tür yeteneklerini taklit etmek istemektedir. Ancak, bilgisayar ortamında gerçek dünya nesnelerinin 3B yeniden yapılandırılması hala çok hantaldır.

Gerçek dünyadaki nesnelerin 3B modellerini oluşturmak birçok alanda önemli bir yer tutar. Ancak, 3B modellemenin, zaman alıcı, masraflı ve zahmetli bir süreç içermesinden dolayı onun yaygın olarak kullanılması hızlı bir artış göstermemektedir. Örneğin, genel bir arkeolojik kazı sırasında sadece en çok önemli olduğu düşünülen nesne taranır ve bulunur.

Özellikle son yıllarda bilgisayarlı animasyon, bilgisayar oyunları, sanal ve arttırılmış gerçeklik, bilgisayar insan etkileşimi gibi birçok bilgisayarlı grafik uygulamaları alanında fiziksel gerçek dünya nesnelerinin 3B modellerini kullanmaya başlamıştır. Fiziksel gerçek dünya nesnelerini oluşturmak için kullanılan çeşitli 3B tarama teknikleri mevcuttur. Gerçek fiziksel nesnelerin modelini çıkarmak için kullanılan bu tekniklerin en yaygın olanları yapısal ışın, lazer tarama yöntemi [1], RGB (RedGreen Blue)yöntemi [2,3], stereografik yöntemler [4], time-of-flightyöntemi [5] ve multi-view yöntemleridir [6]. Bu yöntemler 3B model oluşturmak için farklı teknikleri kullanmamıza olanak sağlasa da bu yöntemler için kullanılan cihazların temini oldukça masraflı ve bunların kullanımları için de ileri düzeyde bilgi sahibi olmak gereklidir. Dahası sahne yakalanma anında modeli oluşturulacak olan nesne hareketli ise bu durumda farklı görüşler arasında seyrek dokular ve karmaşık tıkanıklıkların oluşması gibi problemler meydana gelebilir [7].

Tarama ve derinlik hesaplama yeteneklerine sahip cihazlar arasında son yıllarda bu alanda kullanılmaya başlanan yeni bir tür derinlik kamerası özelliği taşıyan Microsoft Kinect kamera kullanılmaya başlanmıştır. Derinlik özelliği taşıyan bu tür kameralar arasında hızla yerini alan Kinect algılayıcı kamera yeni bir tür olarak toplumda büyük bir

ilgi çekmiştir. Kinect algılayıcı kamera ile geleneksel 3B tarayıcılar karşılaştırıldığında, video oranında görüntü verilerini ve derinlik yakalamada, ışık ve doku durumuna karşı çok toleranslı olan şartlarda dahi az dikkatle var edilebilmektedir. Kinect'in bu tür özelliklerinin yanı sıra onun, normal bir video kamera olarak da kullanılabilmesi, düşük maliyeti, kompakt olması ve genel kullanıcılar tarafından kolaylıkla elde edilebilir olması birer avantajdır.

Derinlik kameraları arasında Microsoft Kinect algılayıcı kameranın, geleneksel 3B tarama cihazlarına nazaran günlük kullanıcılar için daha kolay ve daha ucuz elde edilebilmesi onun en önemli tercih edilme sebeplerinden birisidir. Ancak, belli bir mesafenin dışında Kinect kamera tarafından yakalanan derinlik verisi çok düşük kalitededir.

1.1. Tez Çalışmasının Amacı

Bu tez çalışmasında yöntemler incelenirken görüldü ki bazı araştırmacılar, 3B modeli çıkarılacak olan sahnelerin verileri elde edilirken 3B tarayıcılar gibi Kinect algılayıcı kamerayı kullanmayı denemektedirler. Bunlara örnek bir çalışma olarak, kapalı ortamlarda yoğun 3B haritalar oluşturmak için piksel başına derinlik bilgileriyle birlikte RGB görüntüleri kullanılmıştır [8]. Ancak, asıl sorun algılayıcı kamera olarak kullanılacak Kinect'in 3B tarama için nispeten düşük X / Y çözünürlüğüne ve derinlik doğruluğuna sahip olmasıdır. Bu sorunu gidermek için, derinliği süper çözünürlük ile veri kalitesini artırmak için bir yöntem tarif edilmiştir [9]. Kinect ve yaygın bir grafik donanımı kullanılarak, karmaşık ve keyfi kapalı sahnelerin doğru ve gerçek zamanlı haritalanması için bir sistem sunulmaktadır [10]. Ancak, bu işte Kinect çoğu kez sadece katı cisimleri taramak için kullanılabilmektedir.

Kinect algılayıcı kameralar, yeni bir kavram olmasına rağmen derinlik algılayıcı kamera olarak, herkes tarafından erişilebilir hale gelmiş ve derinlik kameraları arasında yerini almıştır. Derinlik algılama, kalite, düşük maliyet ve gerçek zamanlı uygulanabilir olması açısından Kinect, alanında algılayıcıların kullanımı açısından meraklıları ve araştırmacılar arasında popüler hale gelmiştir. Biz nokta bulutu kütüphanesi NBK (Point Cloud Library) kullanarak kapalı bir ortamda bir 3B model oluşturmak için bu Kinect algılayıcı kamera yeteneklerinden faydalanmaktayız. Ayrıca Kinect ile elde edilen, birden fazla derinlik bilgisi içeren bir resim çerçeve kamera pozisyonu ve onun rotasyonu ile birleştirilirse daha yoğun bir 3B harita elde edilmek için de kullanılabilir. Bu harita daha ayrıntılı yüzey elde

etmek için işlenebilir. Bu işlemde çok sayıda nokta bulutları oluşur. 3B yeniden yapılandırma için birçok yaklaşım geliştirilmiştir [11]. Bu yaklaşımların çoğu mesafe tarayıcı [12] [13], stereo kameralar [14] veya monoküler kameralar [15] gibi çeşitli algılayıcılar içerir. Bu çalışmanın uygulanması sırasında, önceden var olan teknoloji ve yazılım araçları kullanılmaktadır. Yazılım kütüphaneleri, çerçeveler ve bu çalışmada kullanılan uygulamaların tümü açık kaynak topluluğu tarafından yönetilir. Kinect algılayıcı kamera Microsoft'un Xbox360 için verilen ticari bir ürünüdür ve Kinect'in kendi yazılım geliştirme SDK'sı (Software Development Kit) vardır. Bu tez çalışmasının yazılım aşamasında Kinect'in açık kaynak kodlu uygulamaları kullanılmıştır [16].

Bu tezin amacı gerçek bir dünya nesnesinin nokta bulutu ile sanal uzay ortamı içerisinde 3B olarak yeniden yapılandırılmasını sağlamaktır. Nokta bulutu içerisinde çok boyutlu koordinat bilgisi tutan bir veri yapısıdır. Nokta bulutları sıklıkla üç-boyutlu verileri temsil edecek şekilde X, Y ve Z, koordinatları hakkında veri tutması için kullanılır.

Bu tezde, bu nokta bulutu verilerinin OpenGL (Open Graphics Library) platformu içerisinde bir çokgene nasıl dönüştürülebilir ifadesi tarif edilmektedir. Bunun uygulanması için gereken temel ihtiyaç kişiselleştirilmiş sanal alanı oluşturmaktır. Bu tezin araştırma hedefi, tamamen kişiselleştirilmiş bir sanal alan içerisine manipüle ve bir yapı taşı olarak kullanılabilir herhangi bir nesnenin kolay bir şekilde yerleştirilebilmesidir.

Bu tez yeni kullanıcı deneyimleri oluşturmak ve sanal uzay ortamında 3B kullanıcı ara yüzleri incelememizi gerektiren daha büyük bir projenin bir parçasıdır. Bu işin belirli bir bölümü sanal alanlarda içerik oluşturma olanaklarını destekler. Yakalanan nesneler daha sonra farklı cihazlar ve sanal alanlar arasında transfer edilebilir. Bu şekilde, 3B modelleme başlığı altında yatan teknolojisi hakkında deneyimi veya herhangi bir bilgisi olmayan kişilerce sanal bir ortamda tamamen kişiselleştirilmiş alanlar yaratmak kolaylaştırılabilir.

Bilgisayarla görme ve bilgisayar grafiklerinde, gerçek nesnelerin görünümünü ve şeklini yakalama işlemleri 3B yeniden yapılandırma olarak adlandırılmaktadır. Bu işlem, aktif ya da pasif yöntemler ile gerçekleştirilebilir. Bir model zamanla biçim değiştirecek yapıda olduğu zamanlarda katısal-olmayan ya da zamansal-uzay yeniden yapılandırma hakkında konuşmak gerekir.

Aktif olarak adlandırılan yöntemlerde, yeniden yapılandırılacak olan nesneye mekanik veya radyometrik olarak müdahale edilebilir. Mekanik olarak müdahale edilebilecek basit bir örnekte bir pikap üzerine koyulan bir gerçek dünya nesnesinin dönmesi sırasında tüm hatlarının mesafesini ölçmek için derinlik göstergesi kullanılabilir. Radyometrik

yöntemlerde ise yöntemin uygulanacağı nesneye doğru yayılan ışın ve parlaklığın ardından yansıyan kısmının ölçülmesi ilkesi vardır. 3B tarama sırasında daha fazla detay gösterecek olan ışın kaynaklarına örnek olarak mikrodalga fırın veya ultrason için ışık kaynakları, renkli görünür ışık, TOF (time-of-flight) lazerler gösterilmektedir.

3B yeniden yapılandırmadaki pasif yöntemlerde ise yapılandırılacak gerçek dünya nesnesine dokunulmaz, 3B yapısının anlaşılması işleminde sadece bir algılayıcı kullanılarak nesnenin yüzeyine yayılan parlaklığı ölçmek için yansıyan ışınlar kullanılır. Genellikle bu algılayıcı görünür ışığa karşı duyarlı olan ve belirli yöntemlerle bir dizi sayısal görüntü (iki veya daha fazla) veya videoyu işleyebilen bir kameradır. Böylece bu yöntemle 3B model oluşturmada görüntü tabanlı yeniden yapılandırmadan bahsedilebilir.

1.2. Tezin İçeriği

Bu tez çalışmasının temelini oluşturan modellemenin gerçekleştirilmesi aşamalarındaki ilk adım, nesnelerin modellerinin taranmasında kullanılacak yöntemlerin belirlenmesidir. Bunun için farklı tarama yöntemleri incelenmiş, bize en uygun olan Kinect algılayıcı kameranın kullanıldığı yapısal ışık yöntemi tercih edilmiştir. İkinci adım geliştirilen yöntemdeki tasarım ve modelleme verileri üzerine uygulanacak sayısal yöntemlerin incelenmesidir. Üçüncü kısımda yapısal ışık yöntemiyle oluşturulan kaba üç boyutlu modellere uygulanan birkaç filtreleme tekniği incelenmiştir. Tezin dördüncü bölümünde yer alan uygulama kısmında, modellemenin gerçekleştirilmesinde kullanılan yazılımdan, uygulamamızın ön gereksinimlerinden ve programın işleyişinden bahsedilmektedir.

2. NESNE TARAMA YÖNTEMLERİ

Nesnelerin ve bunların içinde bulunduğu sahnelerin üç boyutlu yapısına ait verilerinin elde edilmesi amacıyla kullanılacak olan yöntemlerin belirlenmesi safhasıdır. Bu bağlamda bize göre tercih edilebilecek birçok yöntem, maliyeti, kullanılabilirliği ve geliştirilebilirliği açısından değerlendirilmiştir. Bu yöntemlerin incelenmesi neticesinde, modelleme için kullanılacak olan nesnelerin verilerinin toplanmasında bir yöntem belirlenmesi hedeflenmiştir.

2.1. Lazer Tarama Yöntemleri

Gerçek dünyadan üç boyutlu geometrik içeriklerin elde edilmesi, bilgisayar grafikleri alanındaki birçok sayıda uygulama için vazgeçilmez bir görevdir. Ne var ki, gerçek zamanlı olarak, dünya sahneleri içerisinden geometrik şekillerin mesafe bilgilerini sağlayabilen, kaliteli, yüksek çözünürlüklü, düşük fiyatlı sistemler bulmak zordur. Yapılandırılmış ışık veya lazer taramaya dayalı cihazlar çok yüksek kalite ile insan vücudunu yakalayabilir. Ancak, bu cihazlar pahalıdır ve sık sık çalışması için özel bir bilgi içeren bir dizi işlem yapmayı gerektirir. Örneğin, tüm vücudu renkli olarak 3B elde edebilen Cyberware tarayıcısının fiyatı yaklaşık 240,000$'dır [17].

Yeni geliştirilen bir mesafe ölçüm donanımı olan, derinliği içeriğini 3B olarak hesaplayabilen kamera teknolojisi yeni bir çığır açar. Şu anda derinlemesine kamera teknolojileri alanında istihdam edilmek üzere olan iki ana yaklaşım vardır. Bunlardan ilki TOF hafif bir darbe aktarımları arasındaki gecikme süresinin ölçülmesi prensibine dayanmaktadır [18]. Bu etkin aydınlatma darbesinin sağlanması için gerekli olan özel çipin yüksek maliyeti yüzünden, fiyat yaklaşık 8.000$'ları bulmaktadır. İkinci yaklaşım ise esasen ışık kodlama ilkesine dayanmaktadır, sahne ve esas belirleyici derinlik bilgisi üzerine bilinen bir kızılötesi ışının yansıtılmasıdır. Böyle sadece standart kızılötesi cihaz ihtiyacı karşılayacak şekildeki CMOS kameralar, maliyet konusunda TOF kameralara göre çok daha düşüktür. Bunların en popüler olanlarından biri Microsoft Kinect algılayıcı kameralardır [19], sadece 150$ civarında bir fiyata temin edilebilir.

Derinlik kamerası kavramsal olarak yeni olmasa da, Kinect herkes için erişilebilir bu tür algılayıcılar yapmıştır. Derinlik algılama kalitesi, cihazın düşük maliyetli ve gerçek zamanlı doğası göz önüne alındığında, hem araştırmacılar ve hem de meraklıları için algılayıcı anında popüler hale gelmiştir. Kinect algılayıcı kamera bir fiziksel sahnenin

ayrık aralıklı ölçümlerini içeren gerçek zamanlı derinlik haritaları oluşturmak için yapılandırılmış bir ışık tekniği kullanır [20]. Bu veriler ayrık 3B noktalar kümesi Point Cloud (nokta bulutu) olarak elde edilebilir. Kinect derinlik verisi özellikle piyasada mevcut diğer derinlik kameraları ile karşılaştırıldığında, kabul edilebilir olsa da, hala gürültülü olabilmektedir. Derinlik ölçümlerinde sıklıkla dalgalanmalar ile karşılaşılır ve derinlik haritalarının, ölçüleri belirlenemeyen çok sayıda delik içerdiği görülmektedir.

Şekil 2.1. Derinlik bilgisi alınan sahnenin içerdiği delikler

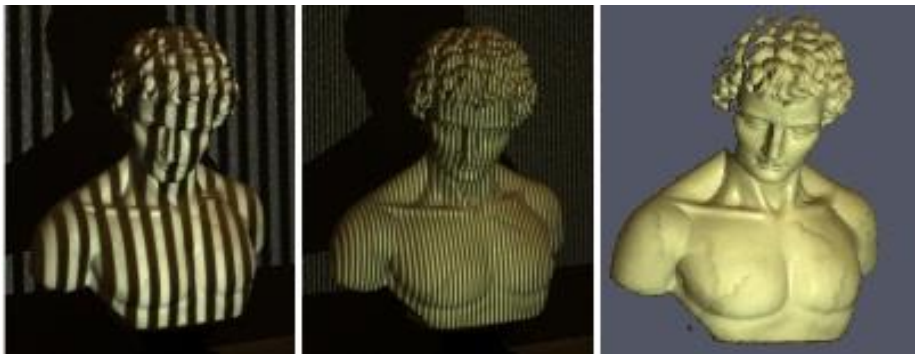
Lazer tarama teknikleri son yıllarda 3B modelleme, bilimsel araştırma, inceleme ve uygulama alanlarında gelişme göstererek, oldukça popüler bir yöntem olarak kullanılmaya başlanmıştır [21]. Bu yöntemin popülerlik kazanmasında yatan temel neden, bu tarama tekniğinin geleneksel tekniklerden daha hızlı veri toplayarak sayısallaştırma işlemlerini kısaltmasıdır. Lazer ile tarama yapmanın en önemli avantajı, yapılacak olan işlemin hızlı ve dokunma olmadan gerçekleştirilmesidir. Lazer ışınlarının yüksek çözünürlük ve inceliği sayesinde, taranan nesnenin her bir yüzeyinin hassas bir şekilde detaylı olarak verilerin alınması sağlanabilir. Bu sayede mühendislik yapıları, otomotiv sektörü, tarihi binalar gibi alanlardaki objelerin hızlı ve etkin bir biçimde ölçülmesi sağlanmaktadır. Bunun dışında insan gözünün algılama sınırları dışına çıkan bazı durumların incelenmesi gerektiğinde kullanılan lazer ışığı sayesinde bu unsurlar belirginleştirilebilir. En önemlisi bir lazer ışık kaynağı ve sayısal kamera kullanılarak kaydedilmiş olan görüntülerden çok kısa bir süre içerisinde 3B modelleme yapılabilmektedir [22].

Şekil 2.2. Lazer tarama tekniği kullanılarak oluşturulmuş 3B modelleme [23].

2.2. Yapısal Işık Yöntemi

Yapılandırılmış ışık yöntemi bir 3B tarayıcıdan yansıtılan ışık düzenleri ve kamera sistemi kullanarak bir nesnenin üç boyutlu şeklini ölçmek için kullanılan bir yöntemdir. Günümüz teknolojisinde bu iki özelliği bünyesinde barındıran nitelikli kameralar RGB-D (Red, Green, Blue-Depth) kamera olarak adlandırılmaktadır. Bunlara bir örnek olarak, kullanıcıların tüketici seviyesinde temin edebileceği bir ürün olan Microsoft Kinect algılayıcı kameradır. Bu gibi RGB-D kameralar özellikleri itibarı ile hareketli ve temassız sınıfı altında kategorize edilebilir. Bu yöntemde temel ilke üç-boyutlu şekilli bir yüzey üzerine ışık dar bir bant boyunca gönderilir. Yansıtma görünümü bir yansıma yüzey üzerinde başka açılardan bozuk görünebilen bir aydınlatma çizgi üretir ve yüzey şekli (açık bölüm) tam bir geometrik yeniden yapılanma için de kullanılabilir. Bu yöntem, aynı zamanda örneklerin çok sayıda edinimine izin verdiği için, bir seferde daha hızlı ve daha çok yönlü birçok çizgiden oluşan desen yansıtılmasına olanak sağlar. Ancak farklı açılardan bakıldığında, geometrik görüntü, nesnenin yüzey şekline bağlı olarak bozuk görünebilir.

Bu yöntemin amacı, bir veya daha fazla dijital kamera ve tek bir yansıtıcı kullanarak bir 3B tarayıcı inşa etmektir. Önceki uygulamalarda kullanılan "masaüstü tarayıcı" ucuz olsa da, sınırlı bir pratik yararı vardır. Tarama süreci çubuk kullanarak elle birkaç ayarlama gerektirir ve statik nesneler oluşturma aşamasında sahne boyunca gölgeli düzlemleri süpürmek için gerekli belirli bir zamana ihtiyaç duyar. Ayrıca, yansıtılan görüntü dizisi içeren çeşitli yapısal ışık aydınlatma dizisi, yansıtıcı satır veya sütunlarına tekabül eden kamera piksellerinin etkili çözümü için de kullanılabilir.



Şekil 2.3. Yapılandırılmış Işık Yöntemi kullanılarak oluşturulmuş 3B modelleme [24].

2.3. RGB Yöntemi

Tarama yöntemleri arasında RGB temelli yöntemler temassız pasif altında kategorize edilebilir. Bu yöntem kullanılarak bir nesnenin şeklini belirlenmesi görünür ışığın varlığına bağlıdır. Bu kategoride çeşitli yöntemler öne sürülmüştür.

Vouzounaras ve arkadaşları ufuk noktası algılama [4] kullanarak tek bir RGB görüntüden 3B geometri türetmek için yeni bir yöntem önermektedirler. Onların yöntemi, ortogonal olarak bilinen geometrik yapılar üzerine dayanır ve yakalanan görüntülerdeki duvarların varlığını gerektirir. Bunlarla blok şeklinde bina ve bina içleri gibi basit geometri ile nesneler yeniden yapılandırabilmektedir. Fakat onların yöntemi, küresel niteliklere sahip nesneleri inşa edememektedir. [16].

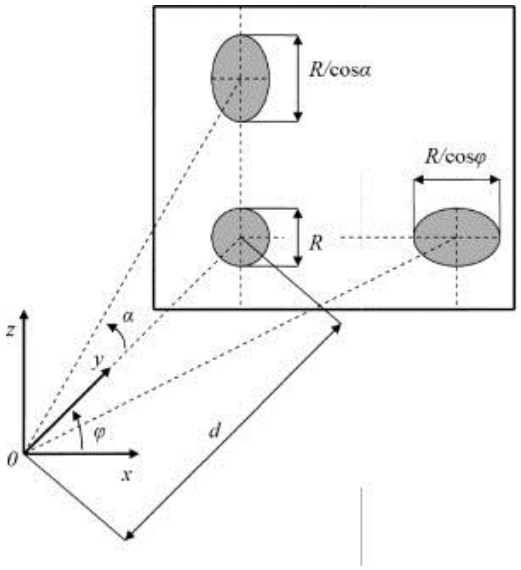
RGB temelli yöntemlerin araştırılması sırasında bazı kısıtlamaların ortaya çıktığı görülmektedir. Bu gibi teknolojiler her zaman derinlik tespiti için ek bir hesaplama adımı gerektirir. Bu da sık sık böyle bir pikap tarzı zemin veya birden fazla kamera gibi ek fiziksel yardımcıları gerektirir. Bu gibi yöntemlerden bazıları da sık sık gerçek nesne geometrisine uymayan sonuçlara yol açar.

2.4. Time-of-Flight Yöntemi

Uçuş zamanı (TOF) olarak adlandırılan bu yöntem, akustik bir ortam aracılığıyla elektromanyetik bir ışın yayılımı ile onun seyahat süresinin hesaplanması, ayrıca diğer dalga için gereken süreyi ölçmek çeşitli yöntemler uygulanması şeklinde açıklanır. Bu ölçüm, bir ortam aracılığıyla belli bir zaman standardı kullanarak hızı veya yol uzunluğunu ölçmek için bir yöntem olarak sunulmaktadır. Böylece hareket eden nesne doğrudan veya dolaylı olarak algılanabilir.

TOF genel anlamda bir kaynaktan yayılan tek darbelik lazer veya kızılötesi ışınlarının gidiş dönüş sürelerini hesaplayarak, kaynağın nesneye olan uzaklığıyla ilgili bilgi verilebilen temassız etkin bir yöntemdir.

Lazer tabanlı yöntemler mevcut derinlik tarama yöntemlerinin en iyi doğruluk sağlayan en eski ve en iyi bilinen yöntemlerdir. Dezavantajı ise, genel olarak fiyatının yüksek seviyelerde olması ve bütün bir sahneyi çekmek için makul bir çözümün bulunmamasıdır.



Şekil 2.4. (TOF) Yöntemi çalışma mantığı

Tarama alt sistemi (dönen esas mekanizma) tarafından tanımlanan lazer ışınının açısal koordinatları φ ve α, nano saniyeler civarında lazer darbelerinin seyahat süresi hesaplanır (d mesafesi).

2.5. Stereografik Yöntemler

Stereografik yöntemlerle RGB temelli yöntemler benzer şekilde temassız pasif altında kategorize edilebilir iken, bu iki yöntem arasında yapılacak bir ayrım vardır. Stereografik kameralar genellikle iki adet birbirine yakın konumlandırılmış ön kalibre RGB kamera içerecek şekilde tanımlanmıştır. Stereografik kameralar şimdilik sadece tüketici seviyesi mobil cihazlarda kullanılabilir halde olmasına rağmen, stereo görünümü bir derinlik haritasının nasıl oluşturulması gerektiği konusundaki araştırma alanında büyük bir ilgi görmeye başlamaktadır.

Çoğu stereoskopik yöntemlerle izleyicinin sağ ve sol gözü için ayrı ayrı iki ofset görüntüler sunulmaktadır. Bu iki-boyutlu her bir görüntü daha sonra 3B derinlik algısı

vermek için beyinde birleştirilir. Bu teknik ile gözlemcinin baş ve göz hareketleri ile görüntülenen üç boyutlu nesneler hakkında bilgilerin arttırılabilmesine olanak sağlanır, ayrıca tam üç boyutlu bir resim görüntülemek için 3B ekranlara ayrılır.

Huang ve arkadaşları [4] stereografik kamera kullanarak bir 3B görüntü yakalayan ve shape-from-shading teoremi kullanan bir yöntem önermektedirler. Zheng ve arkadaşları [25], tüm perspektifleri yakalamak için bir pikap tarzı bir çözüm kullanılır. Bunların uygulanması sırasında küresel yüzeylere sahip nesneler ile ilgili bazı sorunlar meydana gelir. Ayrıca, arka plan nesnenin elde etmek için birkaç görüntü işleme yöntemleri kullanılmıştır, ama bu sürecin ayrıntılarını açıklanmamıştır. Uygulamada, bu yöntem herhangi bir kavisli çizgilere veya küresel yüzeylere sahip nesneler dışındaki ilkel geometrik şekiller ile çalışır.

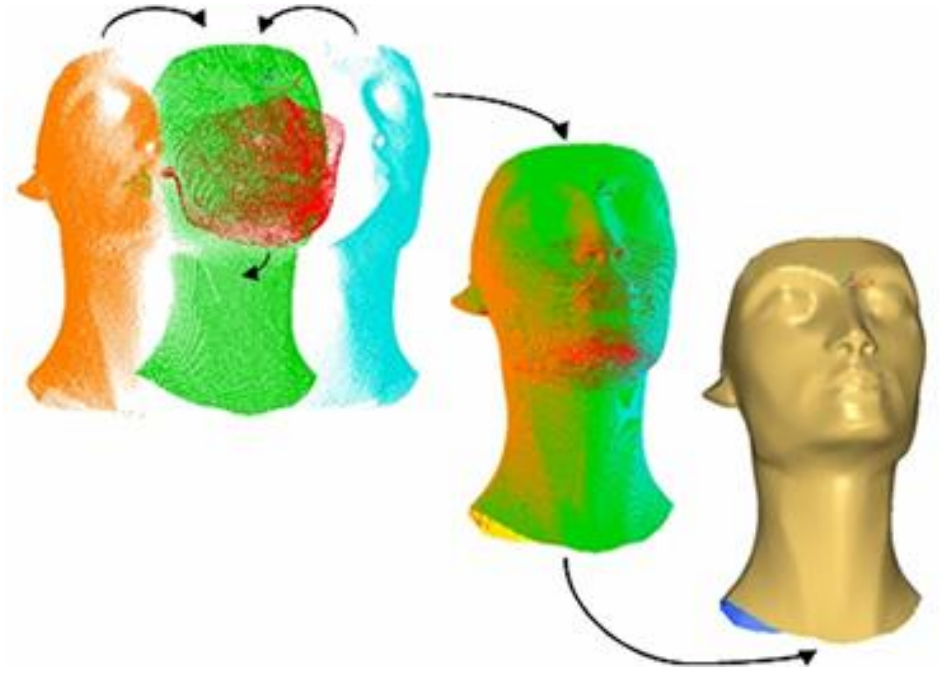
2.6. Sonuç

Bu bölümde, fiziksel nesne modellerinin oluşturulmasının bir ayağı olan, veri elde etme safhasında kullanılacak yöntemin belirlenmesi için yapılan araştırmalar sonucunda bir yöntem belirlenmiştir. Kullanacağımız yöntem, tez çalışmamızda incelediğimiz nesne tarama yöntemleri arasından, uygulamamızda kullandığımız donanımsal algılayıcı Kinect kamera cihazının da desteklendiği, kısmen time-of-flight yöntemini de içeren yapısal ışık yöntemidir. Bu yöntem ile kamera üzerinde bulunan RGB kamera ile birlikte yer alan kızılötesi algılayıcı sayesinde, ışınların nesne yüzeylerine ulaşma süreleri ile nesnelerin sahnelere olan uzaklıkları birleştirilerek sanal ortamdaki konumları belirlenmektedir.

3. GELİŞTİRİLEN YÖNTEMLER

Bu tezin uygulanması sırasında, önceden var olan çoklu araçlar, teknolojiler ve donanım kullanıldı. Uygulamada kullanılan bütün yazılım kütüphaneleri, görüntüler ve uygulamalar açık kaynak topluluğu (özgür yazılım topluluğu) tarafından yönetilmektedir. Microsoft Kinect algılayıcı, ticari bir ürün ve resmi bir yazılım geliştirme kitine sahiptir. Kendisine ait kapalı kaynak kodlu yazılıma sahip olmasının dışında özel bir açık kaynak kodlu SDK'sı da mevcuttur ve bu tezin yazılım uygulamasında bu yazılım kullanıldı. Kullanılan kütüphane ve teknolojilerin bazıları bu tezin (örneğin NBK -Nokta Bulutu Kütüphanesi gibi) uygulama tasarımı sırasında yapılan seçimlerdir. Uygulama yöntemleri nokta bulutu kaydı, algılayıcı kamera yerleştirme yöntemleri ve sayısal yöntemler başlıkları altında incelenmiştir.

Fiziksel bir nesnenin tam anlamıyla yeniden oluşturulması, çok açıklıklı görüntülerin bu nesnenin etrafında farklı açılardan çekilmiş olmasını gerektirir. Bunun sebebi kamera hangi pozisyonda olursa olsun nesnenin bir ya da daha fazla yüzünün her zaman kapalı kalacak olmasıdır. Bu da tek bir nesne çekimleri arasındaki dönüşümü izleme yöntemini bir ihtiyaç haline getirmektedir [26]. Dönüşümleri izleme veya hesaplamak için çeşitli yöntemler ortaya konulmuştur.

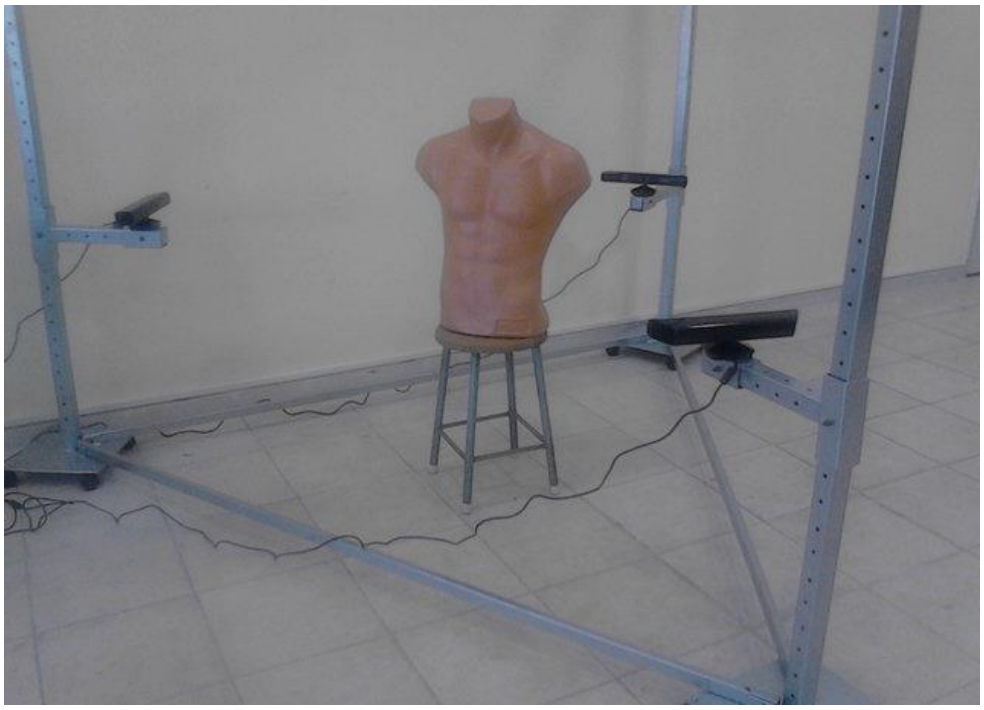


Şekil 3.1. Nokta Bulutu Kaydı (Point Cloud Registration) [27].

3.1. Algılayıcı Yerleştirme Yöntemleri

Yeniden yapılandırılacak olan üç boyutlu gerçek dünya nesnelerinin bulundukları sahnelerin verilerinin düzenli bir şekilde oluşturulması aşamasında, algılayıcı kamera ve donanımların pozisyonlarının en uygun bir şekilde seçilmesi gerekmektedir. Öncelikle uygun bir açı veya gerekli pozisyonların oluşturulması için stratejiler geliştirilmelidir. Uygulanacak olan yönteme göre sıralı kamera görüntülerinden alınan veriler üzerinde işleme veya hazırlanmış dönel bir zemin üzerine yerleştirilmiş nesneden alınan hareketli görüntülerin verileri üzerinde bir dizi işlem gerçekleştirilebilir.

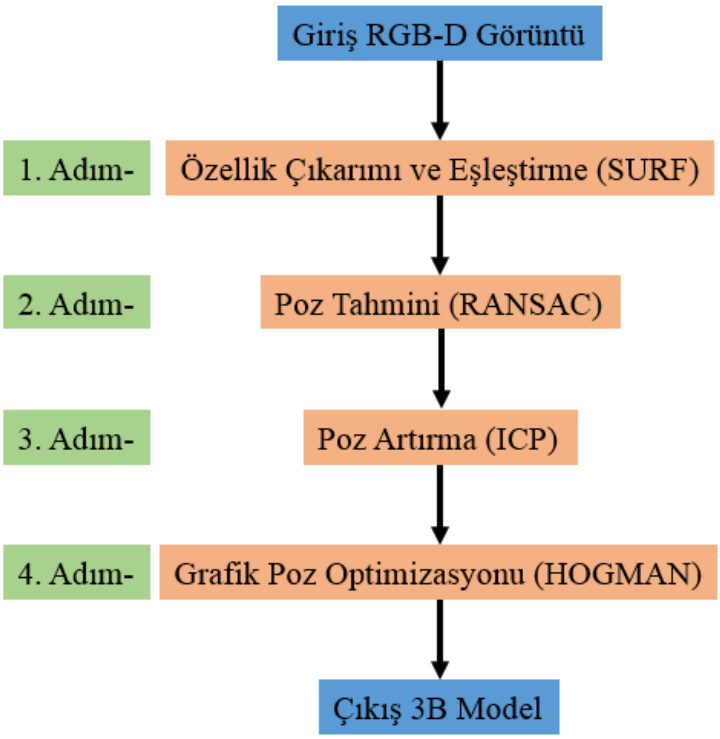
Üzerine yeniden yapılandırılacak olan gerçek dünya nesnesinin konulacağı döner tabanlı bir çözüm üretilebilir, tek tek görüntülerin dönüşümünü izleyerek ve çekilen her bir görüntüyle birlikte tablonun döndürme değerlerini kaydederek kullanılabilir [2, 5, 30].Bu yaklaşımın dezavantajı ise nesneyi döndürmek için harici bir cihaza ihtiyaç duymasıdır. Birden fazla algılayıcı kamera mevcut olduğu zaman, ön ölçümlemeli kamera yöntemi kullanılabilir. Kameralar bilinen pozisyonlarda yerleştirilir ve bunlar arasında dönüşüm gerçek çekim işleminden önce çözülür [29]. Kameralar yeniden yapılandırılacak olan nesnenin etrafına yerleştirileceği ve birbirlerine göre pozisyonları sabit kalmak zorunda olduğu için bu yöntem taşınabilir amaçlar için uygun değildir.



Şekil 3.2. Nesne tarama için algılayıcı konstrüksiyonu

3.2. Sayısal Yöntemler

Bu yöntemler genellikle kapalı ortamlarda yoğun 3B harita oluşturma yeteneklerini göstermek için kullanılmaktadır. Yeniden yapılandırma için çoklu Kinect derinlik bilgilerinin, doğru kamera pozisyonlarının ve görüntü dönüşlerinin birleştirilmesiyle gerçekleştirilir. Etkili küçük çalışma alanlarının 3B haritalarının üretilmesinde kullanılan bu yöntemlerin ana fikrini, ardışık çerçeveler arasındaki mekânsal bilgilerin birleştirilmesi oluşturmaktadır. ICP, (En yakın nokta tekrarlama) 3B şekil hizalama hesaplamasında kullanılan bir algoritmadır. ICP, görüntülerdeki üst üste olan 3B noktaları karşılaştırarak her 3B görüntü için kamera dönüşümlerini hesaplamak için kullanılabilir. Bu yöntemi kullanarak gerçek zamandaki bütün görüntülerde ICP tahminini kullanmanın sadece GPU (Graphics processing unit) uygulamaları sayesinde mümkün olduğu öne sürülmektedir [30].



Şekil 3.3. Kinect ile yakalanan görüntülere uygulanan yaklaşımın dört adımı

Özellik çıkarımı ve eşleştirme, belirli bir alan içerisindeki herhangi bir nesneye ait mekânsal bilgilerinin, kamera pozisyonuna göre üç boyutlu olarak sanal ortamdaki koordinat sistemine göre konumunun belirlenmesidir. Birinci adım (SURF) algoritması

farklı açı ve mesafelerden özellik noktaları ile ilgili ilişkilendirme yapar. Kamera açı hesaplanması sırasındaki farklı tanımlamalar neticesinde yanlış nokta bulutu eşleştirilmesi oluşabilmektedir. Tıkanıklık, örtüşmeme gibi yanlış eşleştirmelerinde ikinci adım (RANSAC) algoritmasına başvurulur. Algoritma, görüntü noktaları ile bir alt görüntü noktaları arasındaki ilişkilendirmeler için kullanılır. Kamera pozu tahmin etmek için üç boyutlu noktalar arasında bir dizi denklemler kümesi oluşturur. Bu denklemler en az görüntüleme hatası için içsel parametreler arasındaki karesel mesafelerin toplamıdır. Bu algoritma aynı hizadaki iki boyutlu veri setlerini takip ederek, olabildiğince büyük düzlemler çıkararak model olması istenen asıl nesneden ayırma işlemidir.

Aynı nesneye ait alınmış, birden fazla nokta bulutu kümesi verisi birleştirilmek istendiğinde, nesnenin bir kesitinin benzer noktalarına ait konumlarında, mesafe farkı meydana gelmiş ise, bu iki nokta arasındaki fark en aza indirilmelidir. Bunun için ise üçüncü adımdaki dönüşüm işlemini gerçekleştiren (ICP) algoritması uygulanmaktadır. Bu algoritma ile benzer bu iki nokta bulutu çiftine komşu olan ortak en yakın bir nokta belirlenir. Dördüncü adımda verilerin görselleştirilmesi için uygun yapıya dönüştürülmesi sağlanmaktadır. Görselleştirilecek olan sanal ortamın formatına uygun dönüşümler gerçekleştirilmektedir.

3.3. Sonuç

Modellenecek olan nesnenin yerleştirileceği düzeneğin ayarlanması, modelleme için kullanılan aygıtın seçimi ve düzeneğe yerleştirilmesi, modeli çıkarılacak nesnenin belirlenip düzeneğe yerleştirilmesi aşamalarından sonra nokta bulutu verileri üzerine uygulanan algoritmalar sayesinde yalnızca istenen nesneye ait yapılandırmanın bilgisayar ortamında görselleştirilmesi sağlanmıştır. Böylece iç sahnelerdeki 360 derecelik fiziksel nesnelerin birden fazla alınmış taramalarının eşleştirilerek bütün bir modeli oluşturulmaktadır. Yine bu yöntemler ile ilgili konuya bir sonraki kısımda ayrıntılı olarak yer verilmiştir.

4. KULLANILAN TEKNOLOJİLER

Bu tez çalışmasında kullanılan teknolojiler ile ilgili öncelikli olarak fiziksel nesnelerin bulunduğu sahnelerin taranmasında kullanılan algılayıcı kamera donanımının fiziksel özellikleri ele alınmıştır. Sonraki aşamada NBK'nin (Nokta bulutu kümesi) elde edilmesini sağlayan yazılım kütüphaneleri ile ilgili detaylı bilgilere yer verilmektedir. Son kısım olan nokta bulutu verilerinin elde edilmesi sonrasında ise, bu kaba veriler üzerine uygulanan birkaç filtreleme işlemi ile ilgili açıklamalar bulunmaktadır.

4.1. Microsoft Kinect Kamera

Kinect algılayıcı kameraları, derinlik bilgisi, renk verileri ve iskelet takip verilerini içermektedir. Kinect SDK kullanıcı ara yüzü ile bu verilere erişebilen ve verileri işleyebilen uygulamalar geliştirilebilmektedir. Alınan veriler sayesinde hareket algılama, ses algılama, yüz tanıma ve iskelet takip gibi pek çok uygulamalar geliştirilmesine imkan sunan Kinect algılayıcı içerisinde çalışan dört adet donanım bileşeni mevcuttur:

• **Renkli VGA Video Kamera:** Bu video kamera; kırmızı, yeşil ve mavi olmak üzere üç renk bileşenini algılayarak yüz tanıma ve diğer algılama özelliklerine destek vermektedir. RGB Görüntü algılayıcısı olarak adlandırılan bu algılayıcı, istenilen çözünürlükte bir fotoğraf makinesi gibi çalışarak uygulamalarda renkli görüntü kullanabilmemize olanak sağlar.

• **Derinlik Algılayıcı:** Aydınlatma koşullarından bağımsız olarak üç boyutlu alanı görmek için bir kızılötesi yansıtıcısı ve tek renkli CMOS (Complimentary metal-oxide semiconductor) ile birlikte çalışır.

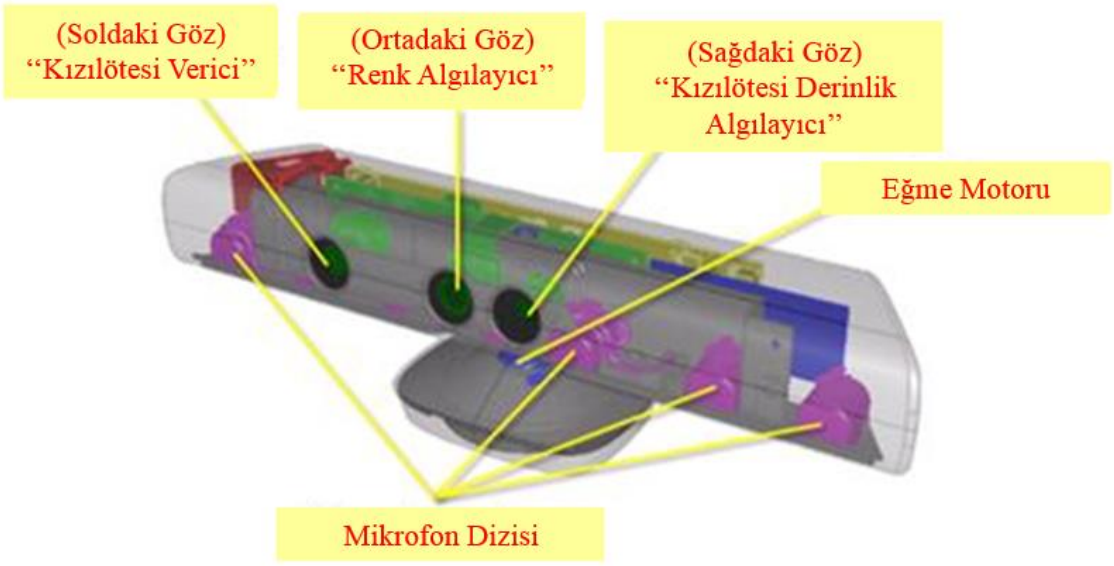
• **Birden çok mikrofon:** Odadaki gürültüden, kullanıcı seslerini izole edebilen dört mikrofon dizisinden oluşmaktadır.

• **Motorlu eğme özelliği:** Algılayıcının otomatik olarak aşağı-yukarı yönde hareketini sağlamaktadır.

Kinect, yüksek performanslı 3B bilgisayarlı görme teknolojileri üzerine uzmanlaşmış ve Microsoft tarafından lisanslı bir şirket olan Prime Sense tarafından geliştirilen RGB-D kameradır. Kinect insan organları ve jestleri okuyabilen bir oyun kontrolörü olarak XBOX-360 platformu üzerinde kullanılmaktadır. Kinect, derinlik haritaları oluşturmak için geleneksel uçuş süresi yöntemleri (lazer tarama gibi) yerine yapılandırılmış ışık yaklaşımını kullanmaktadır. Bu aygıt projeye bir ışık modeli vermek için IR (Infrared)

verici ve derinliği çözümlemek için öngörülen modellerdeki bozulmaları tespit etmek amacıyla IR algılayıcı kullanmaktadır [31].

RGB ve Kinect derinlik akışlarının birleştirilmesi sonucu her bir piksel için renk değerine ilave olarak, bilgi derinlik kullanılabilir içeren bir 640x480 çözünürlükte RGB-D görüntü (yani nokta bulutu) 'dür. İdeal koşullarda, derinlik akışı 3mm çözünürlüğe sahiptir. RGB ve derinlik kameralarının ikisi de 30 Hz frekansla çalışır [28]. Kinect algılayıcı RGB ve IR kamera ve IR vericiden oluşmaktadır.



Şekil 4.1. Kinect algılayıcı kamera donanım bileşenleri

Bu tez çalışmamızda, 3B ölçüm cihazı olarak Kinect algılayıcı kamerayı kullanmaktayız. Hataları incelemek ve derinlemesine deneysel ölçümleri karşılaştırmak için Kinect algılayıcıdan elde edilmiş olan verilerin ölçülebilir değerleri analiz edilebilmektedir. Doğru bir 3B geometrik model elde edilmesi için, derinlik ölçümünün ve Kinect'in doğru ölçümlemesi sağlanmalıdır. Daha sonra 3B Kinect kamera renkli görüntü ölçümlerinin Kinect ölçümleme özelliğinden yararlanarak, ortak bir koordinat sistemi haline dönüştürmek gerekmektedir.

Kinect algılayıcı kamera aslında Xbox360 için üretilen bir ektir. Bu standart RGB kamera, derinlik kamera ve bir mikrofondan oluşur. Kinect algılayıcı kameralarının fiyatlarının nispeten ucuz olması, araştırma amaçları için kullanılabilir olmasını tetiklemektedir. Kinect'in açık kaynak sürücüsünün bulunması sebebiyle Xbox360 dışında da kullanılabilmektedir. Kinect, IR Projektör bize her pikselin görüntülerinin ayrıntılı

bilgilere erişmek için izin verir. Kinect ve sürücüleri sistemden ham verilere erişim için hesaplamalar gerçekleştirir.

Kinect algılayıcı kameradan alınmış derinlik ve RGB görüntüleri aşağıdaki şekilde gösterilmektedir. RGB kamera saniyede 30 çerçeve değişmektedir ve her kare 640x480 piksel çözünürlükte görüntü yakalar. Derinlik ölçümleri parametreleri kamera ölçümleme yöntemleri kullanılarak hesaplanabilir. Kinect algılayıcı kamera fiziksel bir sahnenin ayrık ölçüm aralığı içeren bir derinlik haritası oluşturmak için ışık tekniğini kullanır [20]. Bu veriler ayrık 3B nokta bulutu kümesi olarak ayarlanabilir. Kinect ile alınmış derinlik verisi gürültü, dalgalanma ve çok sayıda 'delik' içerebilir [30].

Kinect algılayıcı kameranın ham derinlik mesafesi maksimum 2^{11}'dir. Ölçüm sırasında metrik derinliği ham derinliğe dönüştürmek gereklidir. IR kamera açısı ile her pikselin derinliği pozisyonunu, aşağıdaki formülü kullanarak metrik derinlik ölçüleri sayesinde ham derinlik mesafesini hesaplamaktadır:

$$X_{ir} = \frac{f_{x\,ir}}{(x - c_{xir})dm}, \quad Y_{ir} = \frac{f_{x\,ir}}{(y - c_{yir})dm}, \quad Z_{ir} = dm \tag{4.1}$$

$$\begin{pmatrix} X_{rgb} \\ Y_{rgb} \\ Z_{rgb} \end{pmatrix} = \begin{pmatrix} X_{ir} \\ Y_{ir} \\ Z_{ir} \end{pmatrix} R + T \tag{4.2}$$

$$X_{rgb} = \frac{X_{rgb}\ f_{xrgb}}{Z_{rgb}} + c_{xrgb} \tag{4.3}$$

$$Y_{rgb} = \frac{Y_{rgb}\ f_{yrgb}}{Z_{rgb}} + c_{yrgb}$$

X ve y derinlik görüntüsünün piksel pozisyonları, f_{xir}, f_{yir} odak uzaklığı, c_{xir}, c_{yir} IR algılayıcının kamera temel nokta pozisyonu ve dm metre derinliğidir, f_{ir} ve c_{ir} ölçümleme tarafından tahmin edilmektedir. Renkli görüntü ve derinlik resim arasındaki haritalamanın RGB ve IR kameralarının döndürülüp R ve çevrilmesi T şeklinde ifade edilebilir [11].

Şekil 4.2. Yakalanmış Kinect derinlik görüntüsü ve RGB görüntüsü

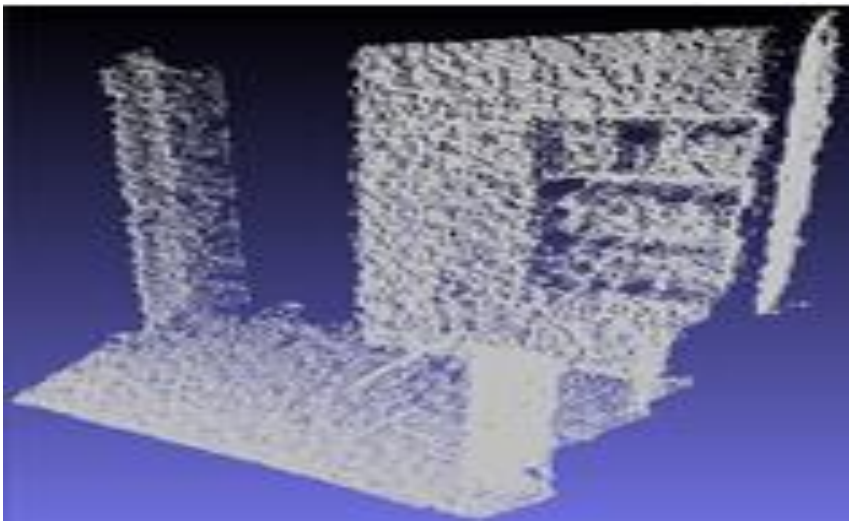
Bazı araştırmacılar 3B tarayıcılar gibi Kinect algılayıcı kameraları kullanmayı denediler. Örneğin, kapalı ortamların yoğun 3B haritaları oluşturmak için piksel başına derinlik bilgileriyle birlikte RGB görüntüleri kullanılmıştır. Ancak, asıl sorun bu Kinect'in 3B tarama için nispeten düşük X / Y derinlik çözünürlüğüne sahip olmasıdır. Bu sorunu gidermek için, [32]'de derinliği süper çözünürlük ile veri kalitesini artırmak için bir yöntem tarif edilmektedir. Bir Kinect ve emtia grafik donanımı kullanarak [33], karmaşık ve keyfi kapalı sahnelerin doğru ve gerçek zamanlı haritalanması için bir sistem sunuldu. Ancak bu yöntemde Kinect algılayıcılar sadece katı cisimlerin taranmasında kullanılabilir.

4.2. NBK (Nokta Bulutu Kümesi)

Bir nokta bulutu topluluğu 3B uzayda farklı açılardan ve ışık koşullarından nokta bulutu görüntüleyebileceğiniz, hatta kendi yüzeylerine renkler ve dokular uygulayabileceğiniz noktalar kümesidir. Bir nokta bulutu yeniden yapılandırma için tam bir 3B model oluşturmayı sağlayan bir kütüphanedir.

NBK n-boyutlu nokta bulutu ve 3B geometri işlemi için yapılmış bir BSD, (Brekel Software Development), lisanslı yazılım kitaplığıdır [34]. NBK, ROS (Robot Operating System) ile tamamen uyumlu olacak şekilde inşa edilebilir, robotik, bilgisayarlı görme ve 3B geometri işlemleri gibi geniş bir kullanım alanına sahiptir. NBK, filtreleme, öznitelik tahmini, yüzeyi yeniden oluşturma, kayıt, model ayarlama ve ayrıştırma için sanat devleti algoritması sağlar. Bunlara ek olarak NBK, çok çekirdekli işlemler için birçok algoritmasının OpenMP, (Open Multi Processing) [35] uygulamasını içermektedir. NBK, robotik ve algı araştırmacılarından oluşan uluslararası bir topluluk tarafından desteklenmiş ve geliştirilmiştir. Bu tez çalışmasında 3B nesne yeniden oluşturulması için, filtreleme,

bölümlendirme ve kayıt gibi çeşitli algoritmaların veriler üzerine uygulanmasına olanak sağlayan gerekli kapsamlı algoritmalarından dolayı NBK seçilmiştir. Bu görevler için MeshLab sunucuları [36] gibi başka kütüphaneler olsa da bunların hiçbiri gerekli işleme yöntemlerinin hepsini kapsamamaktadır.



Şekil 4.3. Kinect ile yakalanan bir iç sahnenin NBK verisi

4.3. Veri Elde Etme

Derinlik kameraları genellikle RGB kamera ile birlikte, düzenli bir şekilde kızılötesi ışık yayılımı içeren yapısal ışık yöntemini kullanmaktadır. Bu özellikleri bünyesinde barındıran cihazlar genellikle RGB-D kamera olarak adlandırılmaktadır. Ayrıca Microsoft Kinect algılayıcı kamera bunlardan biridir. Biz de çalışmalarımızda bireysel geçek dünya nesnelerinin ve iç mekanların yakalamasında Microsoft Kinect kamera kullanarak bir 3B sahnenin modelini oluşturmak için bazı yöntemler üzerinde çalıştık.

Fiziksel bir nesnenin tam yapılandırılması için her zaman farklı açılardan çekilen birden fazla kareye ihtiyaç vardır. Her ne olursa olsun bir kamera pozisyonu için, yakalanan nesnenin bir veya daha fazla yüzeyi tıkalı kalır. Bu nedenle, bazı yöntemler nesne arasında yakalanan dönüşümleri takip etmek için gereklidir [26]. Bu amaçla, bazı çalışmalar dönüşümleri izleme ve hesaplama ile ilgili çeşitli yöntemler üzerinde yoğunlaşmış bulunmaktadır.

Gerçek dünya nesnelerinin yeniden oluşturulmasında kullanılan tüm veriler Microsoft Kinect derinlik kamerasından elde edilir. OpenNI (Open Natural Interaction) araç kiti RGB'yi kullanan Kinect derinliği akışları her biri RGB değer ve koordinat birimleri olan

x,y,z ile yaklaşık 300000 bireysel noktadan oluşan nokta sunumu (örneğin nokta bulutu) yapmak için birleştirilebilir. Bu nokta sunumu NBK görüntüleyici program kullanılarak görselleştirilebilir.

Kinect algılayıcı kameralar, yeniden yapılandırılacak olan gerçek dünya nesneleri için yeterli netlikte derinlik çözünürlüğü sağlamasına rağmen optik gürültüleri, parlak yüzeyleri, görüntülenemeyen alanları ve titreyen dokuları algılamada sorunlar yaratmaktadır [37].

Şekil 4.4. Kinect ile elde edilen ham nokta bulutu verileri [16]**.**

Nesnenin görünmeyen yerlerine doğru görülen renk hataları derinlik ve Kinect algılayıcılarının sağladığı RGB görüntüleri arasındaki küçük değişiklikle açıklanabilir. Bu değişim her Kinect aygıtının kendine özgüdür ve ayrı bir ölçümleme işlemi uygulanarak çözülebilmektedir [38].

Şu an renk algılamasındaki meydana gelmiş olan bir hatayı telafi edecek herhangi bir yöntem uygulaması mevcut değildir. Yeniden yapılandırılan nesnenin kalitesini geliştirmek için böyle bir telafi yöntemi uygulaması olması gerekmektedir.

Şekil 4.5. Kinect ile elde edilen renk edinimi hatası [16]**.**

4.4. Filtreleme

Gerçek dünya nesnelerinin yeniden yapılandırılması sırasında yakalanmış sahne verileri üzerine uygulanan örnekleme, istenmeyen düzlemlerin ayrıştırılması, farklı açılardan alınan sahnelerin birleştirilmesindeki dönüşüm hesaplaması, veri ağırlık ortalaması ile veri azaltma gibi birçok algoritma yakalanmış görüntüler üzerinde yapılması düşünülen çeşitli algoritmaların uygulanması fikrinin gündeme gelmesine neden olmuştur.

Görüntü çekme ve çoklu çekilen görüntüleri tek bir nesne haline getirme sürecinin tamamında veri filtrelemesi gerekli olmasının iki nedeni vardır. Bu nedenleri şöyle açıklayabiliriz.

- Ayrıştırma: Çekilen nesnenin geri kalan verilerden ayrılması gerekir.
- Performans: Çekilen nokta bulutlarının gerçek zaman performanslarını gerçekleştirmek için alt örnekleme yapılması gerekmektedir.

Filtreleme kullanıcı için çekilecek nesnenin gerçek zamandaki görselleştirilmesinde kısıtlayıcı bir faktördür.

Kinect algılayıcıdan yeni bir nokta bulutu temin edildikten sonra filtreleme işleminin tüm adımları buluta uygulanır. Filtreleme işlemi devam ederken, bütün yeni bulut girdileri görmezden gelinir. Filtreleme işlemi sona erdikten sonra çıkan bulut kullanıcı için görselleştirilir ve sistem Kinect algılayıcısından gelecek yeni girdi bulutu bekler. Görüntünün bu nesneyi görselleştirme limiti sadece kameranın incelenen nesneye doğrultulmasındaki hizalama sürecini etkiler.

4.4.1. Nokta Bulutu Derinlik Filtrelemesi

Uzaklık filtrelemede, karar verme mekanizması sadece noktanın kameradan uzaklığına göre işler. Bu filtreleme yönteminde, verilen aralığın dışındaki noktalar sonuçlanan noktadan çıkarılır. Bu tür filtreleme teknolojisini uygulamanın yaygın bir yolu geçiş (Passthrough) yöntemi kullanmaktır.

Yoğunluk filtreleme nokta bulutu filtreleme adımının soyut bir adıdır ve çeşitli algoritmalarla yapılabilir. Bu adımda, nokta bulutunun yoğunluğu, noktalar arasındaki uzaklığa bağlı olarak azaltılır. Bu genellikle tanınmakta olan nesne ile nokta boyutunu önemli ölçüde azaltır.

Kinect derinlik algılayıcı kameranın mesafe aralığı en az 800mm ve an fazla 4000 mm'dir. Bunun yerine Windows donanım olarak kullanılacak olan Kinect algılayıcının standart ölçü aralığı 3000 mm ile 500mm civarında bir dizi mesafeden sağlar. Microsoft

Kinect algılayıcı kamerayı Windows SDK için kullanmaktaysanız onun desteklediği bir sürümü kullanılmalıdır.

Kinect algılayıcı 0.5m ve 15m arasındaki derinliklerden veri alabilmesine rağmen, uzaklık arttıkça derinlik tahmini de büyük oranda artmaktadır. Bazı uygulama çalışmalarında düzlemsel bir yüzey üzerinde bulunan küçük nesneleri çekmek üzerine belirlendiği için, gürültülü arka plan görmezden gelindi.

Derinlik verisini filtrelemek için basit bir direkt filtre (Passthrough) kullanılır. Bazı uygulamalarda, direkt filtre buluttaki bütün noktaların z koordinatındaki derinliklerini karşılaştırır ve 1,5 metre üzerindeki tüm noktaları düşürür. Aşağıdaki şekil derinlik filtrelemesi sonrası orijinal girdi verisini göstermektedir.

Geçiş (Passthrough) filtrelemesinde, önceden belirlenmiş değerler aralığındaki değerlere uyan bütün noktalar verilerinde hiçbir değişiklik olmaksızın geçiş yaparlar. Geçiş (Passthrough) filtrelemesi ana filtreleme yöntemlerinden biridir, hızlıdır ve kullanımı da kolaydır. Bu yöntem NBK'de şablon sınıfı olarak uygulanır bu nedenle de tüm önceden belirlenmiş nokta türlerini destekler.

Voksel ızgara filtrelemede, orijinal nokta bulutu belirlenen yaprak boyutundaki küçük kutulara bölünür. Filtreleme miktarı yaprak boyutuyla kontrol edilebilir. Her kutunun içindeki değerlerden ortak değerler hesaplanır ve bu değerlerden yeni bir nokta bulutu üretilir.

```
// Filtreleme Nesneleri Oluştur
  pcl::PassThrough<pcl::PointXYZ> pass;
  pass.setInputCloud (cloud);
  pass.setFilterFieldName ("z");
  pass.setFilterLimits (0.0, 1.0);
// pass.set Eksi değerleri filtreleme(true);
  pass.filter (*cloud_filtered);

  std::cerr <<"Cloud after filtering: "<< std::endl;
for (size_t i =0; i < cloud_filtered->points.size (); ++i)
    std::cerr <<"    "<< cloud_filtered->points[i].x <<" "
<< cloud_filtered->points[i].y <<" "
<< cloud_filtered->points[i].z << std::endl;
```

Şekil 4.6. Derinlik filtrelemesi işleminin kod parçacığı [39].

Yukarıda Şekil 4.6'daki kod parçacığında gösterilen algoritma ile z koordinatının 0 ile 1 aralığındaki değerleri dışındaki veriler, küme içerisinden çıkarılarak bir filtreleme gerçekleştirilmiştir.

```
Filtreleme öncesi Nokta Bulutu verileri:
    0.352222 -0.151883 -0.106395
   -0.397406 -0.473106 0.292602
   -0.731898 0.667105 0.441304
   -0.734766 0.854581 -0.0361733
   -0.4607 -0.277468 -0.916762
Filtreleme sonrası Nokta Bulutu verileri:
   -0.397406 -0.473106 0.292602
   -0.731898  0.667105 0.441304
```

Şekil 4.7. Filtreleme öncesi ve sonrası veri çıktıları [39].

Yukarıdaki veri çıktılarından anlaşılacağı gibi z koordinat değeri 0 ve 1 arasında olmayan tüm düğümler kümeden çıkarılmıştır. Eksi değere sahip üç z koordinatının çıkarıldığı görülmektedir.

Şekil 4.8. Derinlik filtrelemesi işlemi sonrasında giriş verisi [16].

4.4.2. Alt Örnekleme

Yeniden yapılandırılacak nesnenin birden fazla alınmış olan verilerinin birleştirme işlemleri sonrasında boyutunda meydana gelen artışların önüne geçmek amacıyla verilerin ortalama ağırlık merkezleri alınarak azaltım meydana getirilmektedir.

Girdi verisinin derinlik filtrelemesinin ardından sonraki adımın sayısal gerekliliklerini kolaylaştırmak amacıyla verilerin alt örneklemesi yapılır. Tek tip örnekleme filtreleme işleminin bu adımında kullanılır. Tek tip örneklemede, girdi verisi üzerinde 3B voksel ağı (yani 3B bulutlar kümesi) oluşturulur. Bu aşamadan sonra, voksel alan ortalaması sunan

tek bir nokta oluşturmak için bu noktaların kütle merkezleri kullanılarak bütün noktaların sunduğu her bir voksele yaklaştırılır [40].

```
// Filtreleme nesnesi oluştur
  pcl::VoxelGrid<sensor_msgs::PointCloud2> sor;
  sor.setInputCloud (cloud);
  sor.setLeafSize (0.01f, 0.01f, 0.01f);
  sor.filter (*cloud_filtered);

  std::cerr <<"PointCloud after filtering: "<< cloud_filtered->width * cloud_filtered-
>height
<<" data points ("<< pcl::getFieldsList (*cloud_filtered) <<").";

  pcl::PCDWriter writer;
  writer.write ("table_scene_lms400_downsampled.pcd", *cloud_filtered,
        Eigen::Vector4f::Zero (), Eigen::Quaternionf::Identity (), false);
```

```
Filtreleme öncesi PointCloud: 460400 veri noktası(x y z).
Filtreleme sonrası PointCloud: 41049 veri noktası(x y z).
```

Şekil 4.9. Alt Örnekleme (Down Sampling) filtrelemesi işleminin kod parçacığı ve veri çıktısı [41].

4.4.3. Düzlem Ayrıştırma

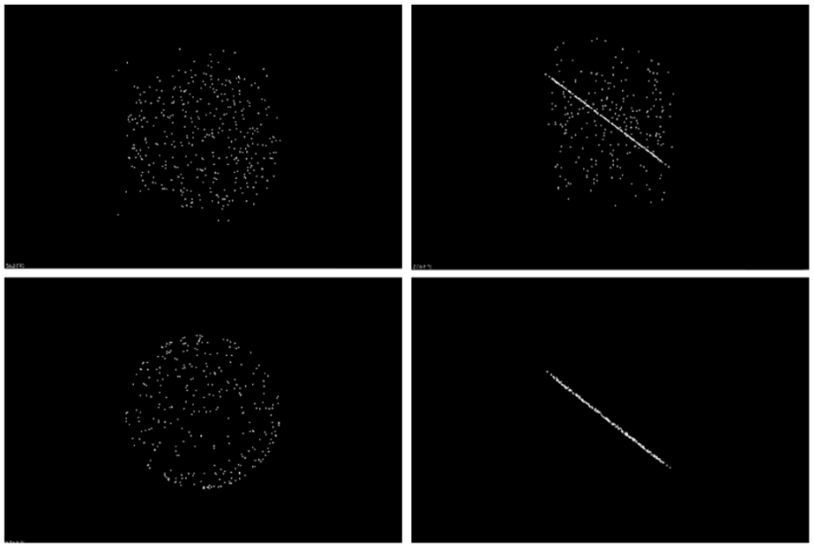
Yeniden yapılandırılacak nesneleri yakalama işleminde verilerin düzgün bir şekilde temin edilmesi amacıyla algılayıcı kameranın pozisyonu ve nesnenin konumunun iyi ayarlanması gerekmektedir. Her ne kadar yakalanacak olan nesnenin verileri sağlam olsa da bu bulunduğu ortam ve yüzeylerde beraberinde işleme tabi olurlar. Bu nedenle asıl işlemi gerçekleştirmek istediğimiz nesneyi onun dışında kalan verilerden arındırmak gerekmektedir.

Bu çalışmaların uygulanması sırasında yakalanan nesnenin yere, masaya ya da başka bir düzlemsel yüzeye yerleştirilmesini gerektirdiği için yeniden yapılandırılacak olan nesneyi temsil eden noktalar yüzeyden ayrı olmalıdır. Bu tarz uygulamalarda yakalanan nesneleri zemin, masa ya da diğer düzlemsel yüzey üzerinde konumlandırılmış olduğunu varsayar, nesneyi temsil noktalar ile bağlı bulundukları yüzeylerden ayrılması gerekir. Fiziksel nesneyi düzlemsel bir yüzeye koymak, çalışmalar sırasında nesnelerin çekilme sürecini kolaylaştırmak için alınan yöntemlerden birisidir. 3B veri setleriyle uyumlu bir RANSAC (Random Sample Consensus) algoritması geniş düzlem bileşenleri tespitinde ve geriye kalan verileri filtrelemede kullanılır. RANSAC örnek verideki içini anlamak için kullanılan

bir algoritmadır [13]. Uygulamalarda kullanılan RANSAC algoritması sahneler içerisindeki düzlemsel modeller üzerinde çalışırken yapılandırılacak olan nesne dışında kalan teferruat verilerinin çıkarılması anlamında kullanılır. Aşağıda bulunan resimde görüldüğü üzere, RANSAC algoritması ile bazı düzlemsel veri bileşenlerinin ayrıştırılmış hali gösterilmektedir.

```
// Oluşturulan RANSAC için uygun model hesaplama
  pcl::SampleConsensusModelSphere<pcl::PointXYZ>::Ptr
    model_s(new pcl::SampleConsensusModelSphere<pcl::PointXYZ> (cloud));
  pcl::SampleConsensusModelPlane<pcl::PointXYZ>::Ptr
    model_p (new pcl::SampleConsensusModelPlane<pcl::PointXYZ> (cloud));
if(pcl::console::find_argument (argc, argv, "-f") >=0)
  {
    pcl::RandomSampleConsensus<pcl::PointXYZ> ransac (model_p);
    ransac.setDistanceThreshold (.01);
    ransac.computeModel();
    ransac.getInliers(inliers);
  }
elseif (pcl::console::find_argument (argc, argv, "-sf") >=0 )
  {
    pcl::RandomSampleConsensus<pcl::PointXYZ> ransac (model_s);
    ransac.setDistanceThreshold (.01);
    ransac.computeModel();
    ransac.getInliers(inliers);
  }
```

Şekil 4.10. RANSAC filtrelemesi işleminin kod parçacığı ve veri çıktısı [42].



Şekil 4.11. Düzlem çıkarma (RANSAC) örnek çıktısı [42].

Şekil 4.12. Düzlem çıkarma (RANSAC) sonrası giriş verileri [16]**.**

Bazı çalışmalar sırasında, yapılandırılacak nesnelerin alınmış verilerinde görüntüsü çekilecek olan nesnenin bulunduğu yerin en geniş düzlemsel bileşeni olduğunu ve diğer düzlemsel yüzeylerin dokunulmamış kaldığını farz etmektedir. Örneğin bir insan modelinin yakalanması sırasında kişinin elinde tuttuğu bir cismi onun vücudunun bir parçasıymış gibi görünmesi. Bu da, bir duvar ya da başka geniş bir düzlemsel yüzeye yakın bulunan nesneyi çekmeye çalışırken sorunlara yol açabilmektedir.

4.4.4. Kümeleme

Yakalanan sahne içerisindeki düzlemlerin ayrıştırılması işlemlerinde aynı düzlem üzerinde bulunup ta verilerinin nesne verilerine dahil olmasının istenmediği durumlarda oluşabilir. Bu yüzden, yapılandırılacak nesne ile istenmeyen verilerin aynı düzlem üzerinde bulunduklarından, düzlemsel ayrıştırma işleminin ardından girdi verisinde hala istenmeyen nesneler bulunabilir. Arka plan nesnelerini filtrelemek için kalan verilerin nesne kümeleri oluşturdukları varsayılır. Her küme, kümedeki diğer noktaların 8cm'lik yarıçapında bulunan noktaların toplamıdır.

Kümeleme işleminin gerçek uygulamasında, NBK kütüphanesinin Öklid küme ayıklama algoritması kullanılır. Bu algoritma ile nokta bulutu veri kümeleri arasında bir seçilim yapılır. Yeniden yapılandırılacak olan nesnenin bulunduğu nokta bulutu kümesi yalnız bırakılacak şekilde kümlere arası eleme yapılır. Yalnızca yeniden yapılandırılacak olan nesnenin nokta bulutu kümesinin verileri kalacak şekilde işlem tamamlanır. Bu algoritma gerçekleştirilerek NBK ayrıştırma ve kümeleri işleme için kapsamlı bir çalışma yerine getirilmiş olur.

Veri girişleri tamamlandıktan sonra nokta bulutu arasından yeniden yapılandırılacak nesne için en geniş nokta bulutu seçilerek varsayılan bulut kümesi olarak tanımlanır ve

geriye kalan diğer kümeler göz ardı edilerek devreden çıkarılır. Daha sonra kalan küme küresel modellerle birleştirilmek üzere görselleştirilir. Bu işlemler sonunda elde edilen yeniden yapılandırılacak salt nesne nokta bulutu veri kümesi küresel modelle birleştirmeye karar verilebilir. Böylece kayıt işlemi başlar.

Şekil 4.13. Filtreleme sonrası giriş verileri [16].

4.4.5. Kayıt

Yeniden yapılandırılacak gerçek dünya nesnelerinin çeşitli filtrelerden geçirilip sadece nesneye ait nokta bulutu verileri elde edilmesinin ardından, kümelenen nesneyi küresel modelle birleştirmeye karar verdikten sonra kayıt işlemi başlar. Kayıt işleminin ilk yinelemesi sırasında, ilk etapta birleştirme işlemi gerçekleşmediğinden küresel model hala boştur ve bu nedenle görüntüsü çekilen küme olduğu gibi küresel modele doğru hareket eder.

Eğer ki farklı açılardan yakalanmış birden fazla farklı kümelerin küresel modelle birleştirilmesine karar verilirse, gerçek kayıt işlemi başlamış olur. Yakalanmış olan farklı diğer küme öncekiyle olduğu gibi birleştirilemez çünkü bu iki kümeyi birbirinden ayıran bir dönüşüm söz konusudur. Çünkü nesneden, aynı noktaya ait farklı birer açıdan alınmış nokta bulutu kümesinin pozisyonları üst üste gelmez. İşte bu noktada kayıt işlemi ile nesnenin bu dönüşümü çözmeyi ve çekilen kümelerin önceden çekilmiş kümelerle uyumlu hale getirilmesi amaçlanır. Bu kayıt işlemi tamamlanma süreci başlangıç uyumu ve dönüşüm tahmini adımları olarak iki kısma ayrılmaktadır.

4.4.5.1. Başlangıç Uyumu

Başlangıç uyumu birden fazla yakalanmış olan nesnenin ardışık nokta bulutu verilerinin birleştirilmesi sırasında farklı iki ayrı bulut kümesinin olabildiğince birbirine yakın

olmasını amaçlar. Bunun için yakalama işlemlerinde nesnenin birden fazla alınmış verilerinin hangi yöne doğru ve hangi başlangıç pozisyonuna göre alındığı önem taşır. Başlangıç uyumu kayıt işleminin sadece ilk iki yinelemesinden sonra uygulanır bunun nedeni de i_0 sırasında önceki modelin bulunmaması ve i_1 sırasında da önceki dönüşümün tahmin edilmemesidir. Bunun nedeni ise ilk iki adımdan sonraki adımın dönüşüm farkının hesaplanarak diğer adımlara uygulanabilmesidir.

Başlangıç uyumu adımının amacı önceki kümenin küresel dönüşümünü uygulayarak yakalanmış kümenin gerçek uyumuna daha yakın bir hale getirmektir. Kümelerin daha yakın bir hale getirilmesi, gerçek dönüşüm tahmininin daha hızlı ve güvenilir olmasına yardımcı olur.

Bu yaklaşımın tek dezavantajı ise kullanıcının nesneyi, istediği açıdan ve rastgele pozisyonlardan çekmesine izin vermemesi ve bunun yerine önceden belirlenmiş yönde nesnenin etrafında kamerayı hareket ettirerek ardışık kareler çekmek zorunda olmasıdır.

4.4.5.2. Dönüşüm Tahmini

Elde edilen nokta bulutu kümelerinden bir önceki küme dönüşümü mevcut kümeye uygulandıktan sonra, bu iki kümenin birleştirilmesiyle oluşturulmuş yeni kümeyle küresel modelin uyumunun nihai dönüşümünün tahmin edilmesi gerekir. Bu dönüşüm tahmini için, ICP (En Yakın Nokta Tekrarlama) algoritması kullanılır. Bu tekrar iki çiğ tarama noktaları arasındaki mesafeyi en aza indirmek için gerekli dönüşümü (çevirme, döndürme) revize etmektedir.

Kavramsal olarak basit ve yaygın bir şekilde gerçek zamanlı olarak kullanılan bu ICP algoritması aslında birleştirilecek olan iki nokta bulutu arasındaki farkı azaltmakta kullanılan bir algoritmadır. Algoritmanın adımları aşağıdaki gibi özetlenebilir:

1. Her iki kümedeki nokta çiftlerinin, en yakın komşularının aranarak ilişkilendirilmesi.

2. En az maliyet fonksiyonunun kullanılarak dönüşümün tahmin edilmesi.

3. Dönüşümün gerçekleştirileceği beklenen noktaların dönüştürülmesi.

4. Ortalama kalite sağlanıncaya kadar işlemlerin tekrarlanması ve sonlandırılması.

En yakın komşu araması ICP yinelemesi sırasında nokta çiftlerini bulmakta kullanılır. Arama, q ∈ M olduğunda bir dizi nokta arasından d boyutlu metrik uzay M içinde sorgu noktası için en yakın çifti bularak çalışır. Metrik içinde temsil edilen d; x, y ve z koordinatlarından ve nokta eğiminden (nokta normali) oluşmaktadır.

NBK'nin ICP uygulaması, birleştirme yapıldığı anda karşılaşan iki farklı nokta dizileri arasındaki dönüştürücü ve devir matrisinde en küçük kareler çözümünü bulmak için tekil değer ayrıştırması SVD (Singular Value Decomposition) algoritmasını kullanır.

Kullanışlı olduğu için birçok alanda başvurulan bir yöntem olan tekil değer ayrışımı özelliklere sahiptir. Örneğin, lineer denklemler, matris dönüşümleri, matris koşulu sayı hesaplama, vektör sistemi dikleştirme ve ortogonal tümleyen hesaplama gibi sistemleri çözmek için kullanılabilir.

Bir mxn şeklindeki A matrisinin tekil değer ayrışımı A = UWV^t, olarak temsilidir. Buradaki U bir ortogonal mxn matrisini, V yine ortogonal nxm matrisini, W ise köşegen elemanları azalan şekilde negatif olmayan sayılardan oluşan, bütün köşegen elemanların sıfır şeklindeki matrisini temsil eder.

Dönüşüm tahmin edildikten sonra, hedef nokta bulutu tahmini dönüşümü kullanılarak dönüştürülür. Bu noktadaki, sonlandırma ölçütleri değerlendirilir. Bir veya daha fazla ölçüt yerine getirilmiş ise, nokta bulutu birleşmiş olarak kabul edilir ve algoritma sonlandırılır. Eğer hiçbir ölçüt yerine getirilmemişse nokta çiftlerini yeniden ilişkilendirerek yeni bir yenileme işlemi başlar. NBK'nin ICP uygulamasının birçok sonlandırma ölçütü vardır. Bu ölçütler aşağıda belirtilmiştir;

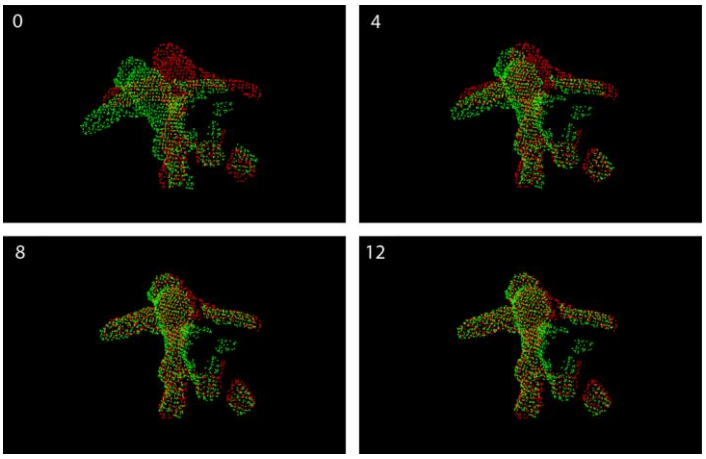
1. Önceden belirlenmiş bir yenileme sayısı mevcut ise o sayıya ulaşılması.

2. Önceki dönüşüm ile mevcut tahmini dönüşüm arasındaki fark önceden belirlenen değerden az olması.

3. Öklid kare hatalarının toplamı önceden belirlenmiş alt sınırdan az olmasıdır.

Bu amaçla genellikle uygulamalarda yineleme için başta belirli bir sayı belirlenir. Örneğin 100 yineleme. Aynı şekilde Öklid kare hataları için de yine belirli bir değer ile karşılaştırma yapılarak işlemler sonlandırılmaktadır. Örneğin 10^{-3} m.

Aşağıdaki şekilde yeniden yapılandırılacak olan nesnenin nokta bulutu verilerine filtreler uygulanmasının ardından, ICP algoritması kullanarak birleştirilecek olan iki farklı kümenin yineleme sırasında iki nokta bulutunun uyumunu ve sonunda 12. yinelemede de birleşmeye ulaşması gösterilmektedir.



Şekil 4.14. Filtreleme sonrası iki nokta bulutunun birleştirilmesi [16].

4.4.6. Küresel Model Artırımı

Yeniden yapılandırılmak istenen nesne için yakalanan nokta bulutu kümesine uygulanmış filtre işlemlerinden sonra oluşturulmuş geçerli küme ile küresel modeli uyumlu hale getiren dönüşümden sonra, küme küresel model ile birleştirilir. Küme ve küresel modelin birleştirilmesindeki uyum işleminde hala küçük sorunlar yaşadığı için sadece kümenin bütün noktalarını küresel modele dahil etmek, bazı karşılayıcı noktaların küresel modelde çoğalmasına neden olur. Bu fazladan oluşturulmuş gereksiz noktalar için çeşitli yöntemler geliştirilmiştir.

Gereksiz oluşacak bu çoğalmadan kaçınmak için iki kümenin birleşmesinin ardından ek filtreleme adımı uygulanabilir. Bu adım sırasında, kümenin tekrar alt örneklemesi yapılır. Bu alt örnekleme, kümenin ilk alt örneklemesinde kullanılan tek tip örnekleme yöntemi kullanılarak yapılır. Bu yöntemi kullanarak, çoğalan bütün noktalar tek bir yeni nokta oluşturmak için bir araya getirilir ve küresel model iki nokta bulutu toplamını daha doğru bir şekilde sunar. Yeni küme küresel modelle birleştirildikten ve filtreleme işlemi bittikten sonra yeni küresel model kullanıcı için görselleştirilir.

4.5. Sonuç

Fiziksel nesnelerin bütün bir modelinin oluşturulması için, kendisinin de içerisinde bulunduğu sahnenin, olduğu haliyle elde edilen verilerinin, modelin kendisi dışındaki fazlalıklarından arındırılması sağlanmıştır. Modelin en iyi sonuca ulaşmasını sağlamak için, birden fazla alınan verilerin birleştirilmesi sırasında kullanılan, model artırımı ve nokta yakınlaştırma gibi filtrelemeler sağlam bir yapıda modellemeye ulaşmak için büyük katkı sağlar.

5. UYGULAMA

Kinect algılayıcı kamera ile genellikle renkli görüntü tanımlanmasının yanı sıra, onunla her bir pikselin derinlik değerine de ulaşılmaktadır. Bu özellik ile Kinect, üç boyutlu nesnelerin taranması gibi senaryolarda da kullanım alanı bulmaktadır. Biz bu çalışmamızda öncelikle Kinect ile derinlik bilgisinin gri tonlamaya çevrilerek görselleştirilmesi üzerinde durduk.

Kinect algılayıcı kamera ile derinlik görüntüsü almanın mantığı, temelde çalışmamızda yine elde ettiğimiz RGB görüntüsü almak ile aynı şekilde işler. Kinect'in sol tarafında bulunan lazer yansıtıcı, 47° dikey, 57° yatay alanda tarama yapar. Derinlik algılayıcı ise, her bir piksel için ışığın yansıma süresini hesaplayarak nesne ile arasındaki uzaklığı belirler. Bu bilgi 640x480 piksele kadar, saniyede 30 defa uygulamamıza raporlanır.

Şekil 5.1. Uygulamada yakalanan sahnenin derinlik ve RGB görüntüsü

Kinect algılayıcı kameranın üzerinde üç adet bulunan gözlerden soldaki göz, algılayıcı kameranın baktığı yöne doğru sürekli olarak lazer taraması yapar. Sağdaki göz ise lazerlerin fiziksel sahneler içerisindeki nesnelere çarpıp geri dönme hızını hesaplayarak 640 x 480 nokta için mesafe bilgisi verir. Ortadaki göz 1280 x 960 çözünürlüğünde bir RGB fotoğraf makinesidir (kamera değil). Çözünürlüğe bağlı olarak saniyede 12 ile 30 arasında fotoğraf çekerek uygulamamıza iletir.

Kinect algılayıcı kameranın alt kısmındaki ızgaralı bölümde yere bakan 4 adet mikrofon bulunmaktadır. Bu mikrofonlar en iyi ses kalitesini yakalamak ve sesin geldiği açıyı belirlemek için kameranın altına mesafeli olarak dizilmiş ve cihaza geniş şeklini vermiştir.

Ayrıca algılayıcıya dikey hareket yeteneği kazandıran mekanizma, basit bir DC (Direct Current) motordan oluşmaktadır. Kullanılan SDK yazılım aracılığıyla algılayıcı kameranın açısı +/- 27 derece hareket ettirebilmektedir. Kinect'in içerisindeki ivme algılayıcı ile yerçekiminin uyguladığı kuvveti ölçerek algılayıcının anlık duruş açısını belirlenebilir. Bu sayede eğimli yüzeylerde bile Kinect'in istenilen açıya bakması sağlanmaktadır.

5.1. Uygulama Ön Gereksinimleri

Kinect algılayıcı kamera ile proje geliştirebilmek ve uygulama çalıştırmak için aşağıda yazılı olan minimum düzeyde donanım gereksinimi sağlanmalıdır:

- Çift çekirdekli 2.66 Ghz işlemci
- En az 2 GB RAM
- Windows 7 ya da 8 (32 ya da 64 Bit) ya da Windows Embedded Standard özelliklerinde bir bilgisayara sahip olmanız gerekmektedir.

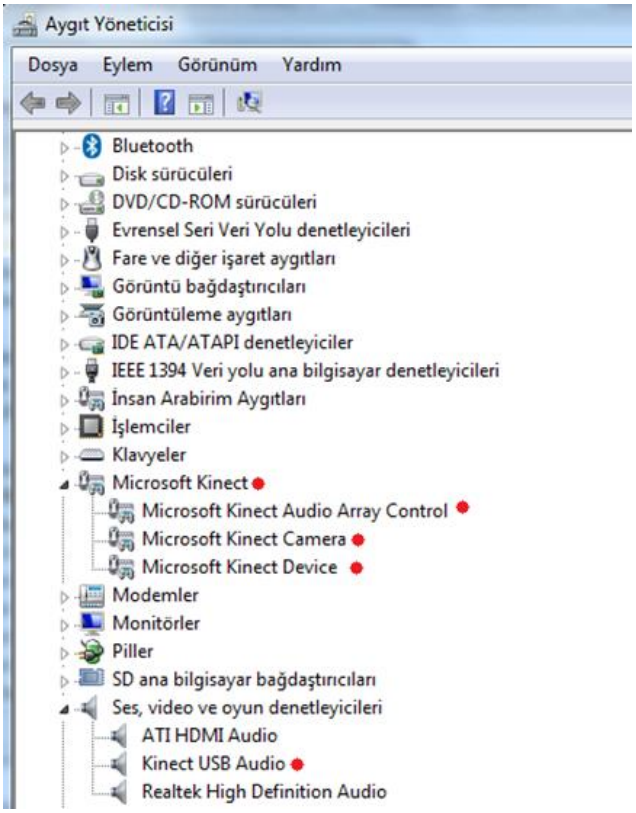
Kinect kamera uygulamaları, belirtilenden çok daha düşük işlemci hızlarında çalışabilmektedir. Fakat bu uygulamaların yoğun interaktif içeriğe sahip olacağını düşünürsek, belirtilen özelliklerin altına inmemek gerekmektedir.

Model oluşturulması sırasında görüntü yakalama aşamasında algılayıcı kameraların daha düzenli bir yapıda veri almasını sağlayacak olan metal yapı düzeneği aşağıda gösterilmektedir.

Şekil 5.2. Modellenecek nesne ve kamera düzeneğinin yerleştirilmesi

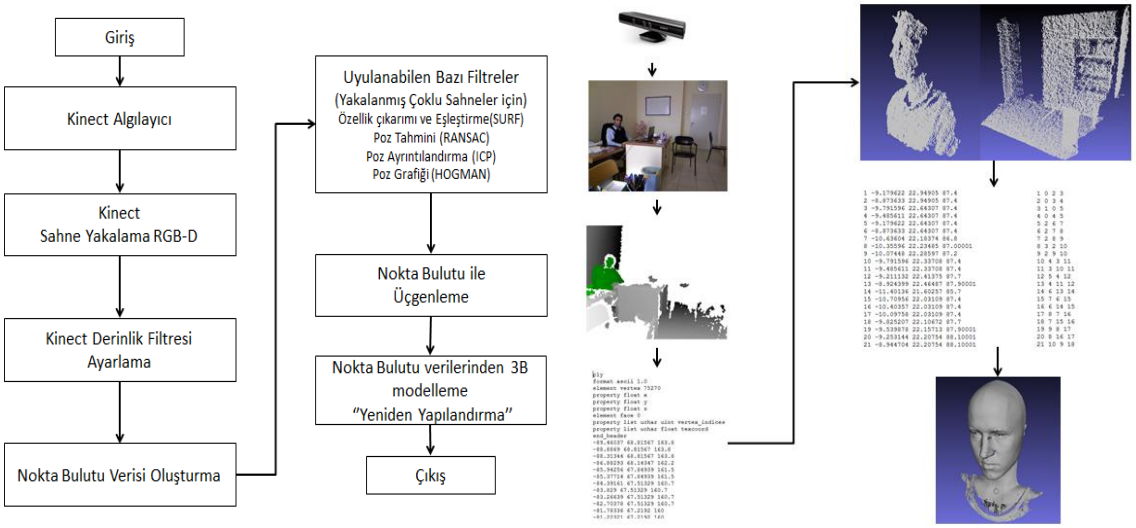
5.2. Kurulum

Öncelikle Kinect algılayıcı kameramızın kişisel bilgisayarımızda sorunsuz bir şekilde çalışmasını sağlamak için Microsoft Kinect SDK'yı mevcut olan resmi sitesinden indirip bilgisayara kurmak gerekmektedir [43]. Biz geliştirdiğimiz programımızda Kinect SDK-v1.0-beta2-x86 sürümünü kullandık. Bu kurulum ile Kinect sürücüsü, uygulama geliştirirken kullanılacak dll dosyası ve birkaç örnek uygulama kaydetmektedir. Kurulumdan sonra Kinect kamerayı bilgisayarınıza bağladığınızda sürücüleri yüklenecek ve Windows işletim sisteminizdeki aygıt yöneticisinde yerini alacaktır. Böylece düzgün bir şekilde takılan cihaz sorunsuz bir şekilde çalışacaktır. Çalışmamız VisualC# ile bir WPF uygulaması olduğu için WPF uygulamamızın referansları arasına Microsoft.Kinect.dll" bileşenlerini eklememiz gerekmektedir. Bunun için de bu SDK'nın bazı metotları çalıştırması için Coding4Fun Kinect Toolkit yine sitesinden indirip kişisel bilgisayarımıza yüklememiz gerekir.



Şekil 5.3. Kinect SDK yükleme doğrulaması (Aygıt Yöneticisi)

5.3. Programın İşleyişi



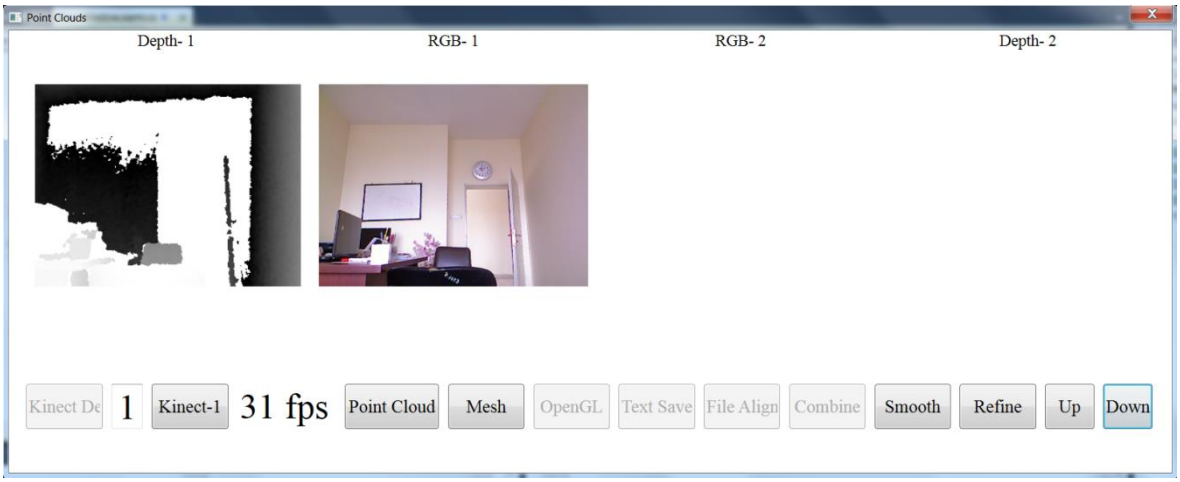
Şekil 5.4. Uygulama akış diyagramı

Uygulamamızda derinlik verisini kullanmak için izleyeceğimiz yol aşağıdaki gibidir:

- Algılayıcı kamera sayısı belirlenir ve Kinect başlatılır.
- İstenilen çözünürlükte derinlik akışı ve renkli görüntü başlatılır.
- Derinlik bilgisine erişeceğimiz olay işleyicisi (event handler) oluşturulur.
- Kinect tarafından saniyede 30 defa derinlik bilgisini raporlayan olay (event) tetiklenir.
- Event argümanları aracılığıyla derinlik bilgisine ulaşılır.
- NBK nokta bulutu kütüphanesiyle derinlik bilgisine ait veriler elde edilir.
- Nokta bulutu verilerinin üçgenleme ile birleştirilmesi sağlanır.

-Birleştirme işlemleri sağlanan verilerin MeshLab ve OpenGL sanal ortamlarında gösterilmesi sağlanır.

Uygulama geliştirmek için Visual Studio 2010 Express sürümü tercih edildi. Fakat çalıştırmak için Visual Studio 2010'un herhangi bir sürümüne sahip olmanız yeterlidir. Bununla birlikte .Net_4 versiyonunu ve ayrıca nokta bulutu kitaplığı kullanmak için NBK ve OpenNI yazılım geliştirme kitlerininin sürümü olan PCL-1.6.0-AllInOne-msvc2010-win32 ve Win32-1.3.2.3-Dev'leri sitelerinden indirip yüklemek gerekmektedir [44].



Şekil 5.5. Kinect uygulamanın grafiksel ara birimi

Kinect uygulama çalıştırıldıktan sonra gelen ekran üzerindeki Kinect Detect adındaki butona basıldıktan sonra program içerisine dahil edilebilen kişisel bilgisayara bağlı olan Kinect cihazların sayısını gösterir. Uygulama için daha sonra çoklu Kinect kullanılabilecek bir uygulama yapılabilir düşüncesiyle birden fazla cihazın bağlanabilmesi düşünülmektedir.

Daha sonra bilgisayara bağlı bulunan algılayıcı kamera Kinect-1 adı verilen butona tıklanarak bu cihaz üzerinden Depth-1 ve RGB-1 başlıkları altındaki alanlarda yakalanan sahnelerin derinlik ve renkli kamera görüntüleri aktarılmaktadır.

Ardından ana ekran üzerindeki Point Cloud isimli butona tıklanmasıyla birlikte NBK kütüphanesi kullanılarak derinlik ve RGB görüntü ekranlarından elde edilecek olan sahnenin nokta bulutu verileri elde edilmektedir.

Programımız üzerinde bulunan Mesh isimli butona tıklanarak bir nesne veya nesnenin de içerisinde bulunduğu tüm sahnenin, nokta bulutu koordinatlarının yanı sıra bu koordinatların hangi düğüm noktalarıyla birleştirileceğini belirten üçgenleme verileri elde edilir. Artık veri dosyamızın içerisinde nesneye ait hem koordinat hem de bu koordinatların hangi düğümler ile birleştirileceğini belirten köşe noktaları bulunur.

Programdaki TextSave, FileAlign ve Combine butonları aynı nesneye ait birden fazla verilerin işlenmesi ve birleştirilmesi sırasında kullanılan butonlardır.

Nokta bulutu kütüphanesi (NBK), nokta bulutu işlemi için açık bir kaynak kodlu kütüphanedir. Berkeley Software Distribution (BSD) lisanslıdır. Ayrıca ticari ve araştırma kullanımı da ücretsizdir. NBK çok platformlu bir kütüphane olmayı hedeflemiş ve bu hedef gerçekleştirilmiştir.

NBK kendi görselleştirme kütüphanesini kurmak için VTK (Visualization Toolkit) ile birleştirilmiştir. Bu kütüphane sayesinde, nokta bulutu işleminden alınan sonuçları hızlı bir şekilde test etmek ve ön örnek oluşturmak mümkün oldu. Görselleştirme n-boyutlu nokta bulutu yapıları ile çalışır. Bu tezde görselleştirme kütüphanesi, yeniden yapılandırılan veri ve yüzey yeniden yapılandırmasıyla oluşturulan yüzeylerin nokta verisi, görselleştirmek için kullanıldı.

NBK nokta bulutu işlemi için birçok modern algoritma sağlar. NBK akıllardaki performans ve verimlilikle yazılmış tam şablonlu ve modern bir C++ kütüphanesi olacak şekilde tasarlanmıştır. NBK'nde bulunan algoritmalar, ilk yöntemdeki sonuçların bir sonraki işleme geçebildiği şekilde beraber çalışmak üzere tasarlanmıştır. Veri etrafa güçlendirmeye dayalı işaretçiler kullanarak yayılır ve bu yüzden de sistemin ana hafızasında veriler hali hazırda bulurken başka bir veri nüshasına gerek yoktur [34].

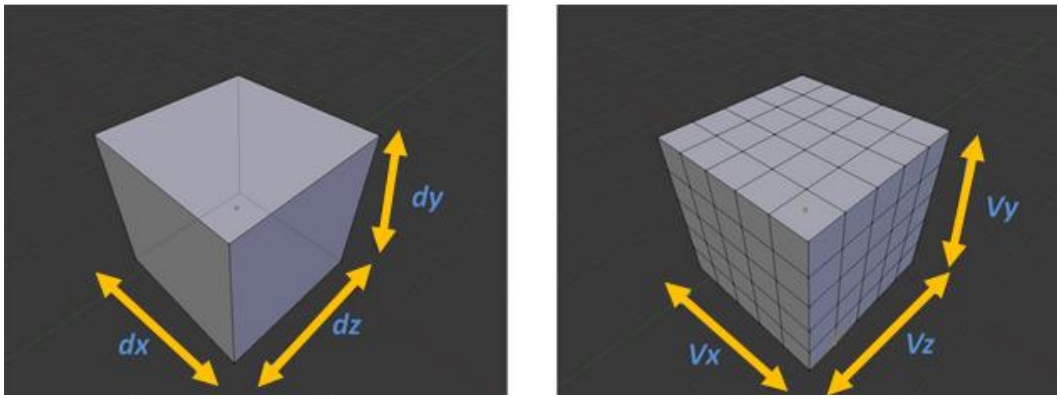
Veri toplama için kullanılan NBK, Prime Sense PSDK, Microsoft Kinect gibi görüntüleme aletlerinin daha halka açık kullanılmasına imkân veren OpenNI (Açık Doğal Etkileşim) doğal etkileşim cihazlarının standartlaştırılması için ortaya çıkan kuruluştur. NBK aynı zamanda içe veya dışa aktarma uygulamalarıyla çeşitli çoklu veri formatını içe ya da dışarı aktarabilir. Bu tasarım, veri yakalama işini en son veri işlemi için kullanılan bilgisayardan farklı bir bilgisayarla yapmayı da mümkün kılar.

Kullanılan nokta bulutu işlemi yöntemleri uygulaması modülerdir (birimsel) ve bu nedenle yöntemler kolaylıkla değiştirilebilir. Bu bölümde uygulamamızdaki farklı işlem algoritmaları kısaca anlatılmaktadır.

5.3.1. Nokta Bulutu

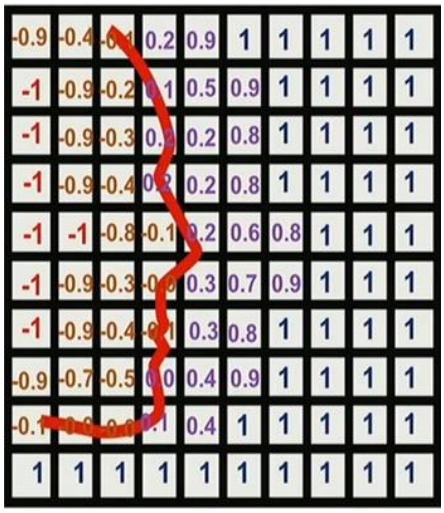
Nokta bulutu çok boyutlu veriyi içinde tutan bir veri yapısına sahiptir. Nokta bulutları genelde üç-boyutlu verileri göstermek için kullanılır ve bunlar x, y ve z koordinatlarıdır. Özellikle z koordinatı sadece derinliğe göre kameranın nesne yüzeyine olan uzaklığını kaydeder. Bunun dışında kalan diğer koordinat uzunlukları Kinect algılayıcı kameranın karakteristik özelliği olarak kameradan 0,8 metre ile 4 metre arasında kalan kısımların x, y koordinatlarıdır. Bu aralıktan daha yakın ve daha uzak olan bütün noktalar algılayıcı kameranın hesaplama yeteneği dışında kalan bir mesafededir. Kinect algılayıcının asıl yakalanması istenilen nesneye göre minimum uzaklığı 0,8 ile 4 metreye ayarlanmalıdır. Bu yüzden de yalnızca 0,8 ile 4 metreye kadar olan uzaklık noktaları korunmuş olur. Bunların dışında kalan uzaklıklardaki veriler için delikli yapılarda sonuçlar doğurabilmektedir.

Nokta verisi daha çok bazı nesnelerin dış yüzeyini göstermek için kullanılır. Geometrik koordinatların yanı sıra nokta bulutu her nokta, normal yön veya her ikisi için renk verileri içerebilir. Veri noktaya eklendiğinde, bulut boyutları artar. Üç boyutlu buluta renk verisi eklendiğinde, bu bulut artık dört boyutlu olur.



Şekil 5.6. Sahnenin voksel yapısı, dx=dy=dz=3m, Vx=Vy=Vz=640x480x480 voksel

NBK kütüphanesi nokta bulutu verilerini oluştururken 3B fiziksel bir sahne içerisindeki alandaki her bir küp piksel değerinin nesne yüzeyi ile kamera arasında bulunan mesafesine göre hesaplayarak koordinat noktasının ölçülmesi mantığıyla çalışır. 2 boyutlu (x,y) piksel değerlerinin üç boyutlu (x,y,z) şeklindeki yapısına voksel adı verilmektedir. Her bir voksel bulunduğu koordinat noktasına göre içerisinde üç boyutlu değer taşır. Vokseller bilinen piksel özelliklerinin yanında fazladan derinlik verisi içerir. Bunun için de vokseller üç boyutlu kübik bir yapı şeklini almaktadır. Küp halinde eşit büyüklükteki vokseller bir dizi şeklinde bölünmüştür. Voksel bütün bir küp içerisinde (sahne) ızgara şeklinde bölünmüş fiziksel alanlara ayrılan belirli sayıdaki küplerin her birine verilen addır. Her bir vokselin boyutları birbirine eşittir. Sahneleri oluşturan varsayılan bütün bir küpün metre cinsinden boyutu eksen başına üç metredir. Varsayılan voksel boyutu eksen başına 512'dir. Veri çekme sırasında, ızgarada önden arkaya geçilirken değerler her voksel için kontrol edilir. Aşağıdaki resimde, her bir vokselin aldığı değerlerin -1 ile 1 arasında değişen farklı ölçülerde olduğu görülmektedir. Bu değer her zaman sabit aralıkta olmak zorunda değildir. Ancak z koordinatı dışında, koordinatların verilen ölçü aralıklarında olduğu görülmektedir. Örneğin bu insan yüzünün x ve y koordinatları 1'den büyük veya -1'den küçük bir değer alamaz.



Şekil 5.7. Bir nesne yüzeyinin voksel değerleri

Uygulamada her bir voksel değeri (x,y,z) koordinatları şeklinde barındıran bir .ply dosyası şeklinde MeshLab sanal ortam içerisinde çalışacak şekilde C sürücüsü içerisine kaydedilmektedir. Bu dosya içeriği aşağıda gösterilmektedir.

```
ply
formatascii 1.0          { ascii, format sürüm numarası }
commentmadebyanonymous  { yorum, anahtar cümle }
commentthis file is a cube
element vertex 2        { dosyanın içerdiği tanımlanmış vertex sayısı }
propertyfloat x         { içerdiğivertexfloat "x" koordinatı }
propertyfloat y         { içerdiğivertexfloat "y" koordinatı}
propertyfloat z         { içerdiğivertexfloat "z" koordinatı}
element face 3           {(Mesh ile birlikte) içerdiği birleşecek vertex element sayısı }
end_header{ başlık sonu }
 0.8412  0.6155  1.4642
 0.7233  0.6323  1.5436
 0 0 0 { vertex listesi başlangıcı }
 0 0 1
 0 1 2
```

```
ply
format ascii 1.0
element vertex 1725
property float32 x
property float32 y
property float32 z
element face 3450
property list uint8 int32 vertex_indices
end_header
0.0474561 0.00671402 0.00774976
0.0476492 0.00711655 0.00771341
0.0468128 0.00536737 0.00784432
0.046762 0.00525942 0.00784425
0.0466932 0.00511251 0.00784099
0.0465647 0.00478846 0.00761509
0.0465471 0.00474429 0.00758437
0.0464256 0.00439643 0.00718348
0.0463905 0.00429643 0.00706995
0.0462817 0.00384311 0.00607561
0.0462155 0.00357221 0.00549328
0.0461564 0.00316135 0.00421748
0.0460851 0.00268506 0.00276459
0.0460872 0.00248846 0.00186518
```

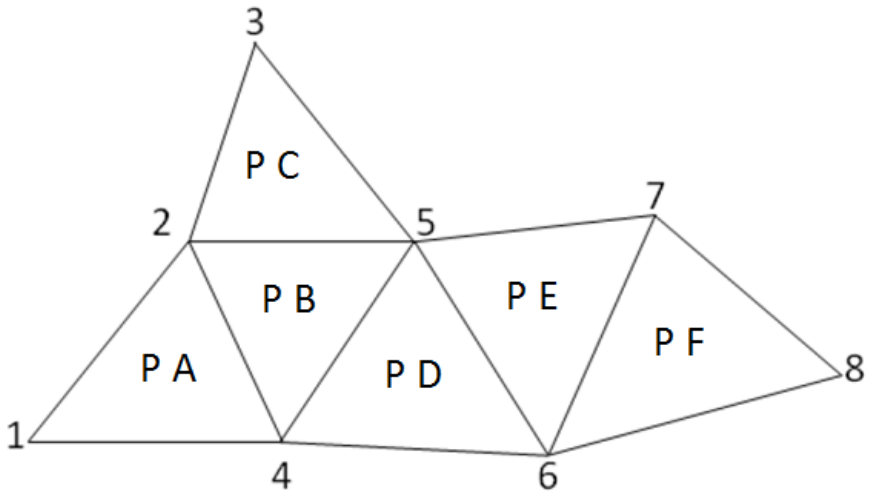
Şekil 5.8. Bir. ply dosyası içeriği

5.3.2. Çözüm Ağı (Mesh)

Çokgen kafes, donanımla çizildiklerinde, 3B modelleri depolamak için oldukça yaygın kullanılan bir veri formatıdır. 3B modelleri oluşturan unsurlar genellikle köşeler, kenarlar, yüzeyler ve çokgenlerdir. Her köşenin kendi yeri vardır ve iki köşe bir kenar oluşturur. Üç kenar bir kapalı yüzey oluşturur ve bu nedenle de yüzler kapalı kanarlar dizisidir.

Çokgenler bir kenarlar dizisinden meydana gelir ve eğer sistem çok kenarlı yüzleri destekliyorsa çokgenler ve yüzler arasında bir fark yoktur. Genellikle oluşturma sistemleri çok kenarlı yüzleri desteklemezler ve bu da kullanıcıya çok kenarlı çokgenleri seçmekten başka seçenek bırakmaz.

Çokgen bir kafeste, her çokgen üç köşeden oluşur ve her köşe çoklu çokgenlerin bir parçası olabilir. Bir nokta birden fazla kenara ait köşeyi temsil edebilir. Aşağıdaki şekilde üç köşeden oluşan çokgen kafesin her yüzünü görebilirsiniz. Örneğin, PA yüzü 1, 2 ve 4 köşesinden oluşmuştur. Aynı zamanda köşelerin çoğunun çoklu yüzlerin bir parçası olduğunu da görebiliriz. Örneğin; 5 köşesi PD, PB ve PC yüzlerinin bir parçasıdır [45].

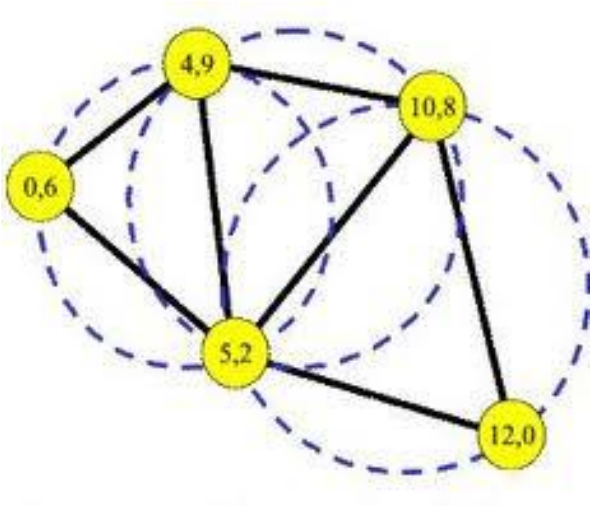


X	Y	Z
1- 0.0474561,	0.00671402,	0.00774976
2- 0.0476492,	0.00711655,	0.00771341
3- 0.0468128,	0.00536737,	0.00784432
4- 0.046762,	0.00525942,	0.00784425
5- 0.0466932,	0.00511251,	0.00784099
6- 0.0465647,	0.00478846,	0.00761509
7- 0.0465471,	0.00474429,	0.00758437
8- 0.0464256,	0.00439643,	0.00718348

Yüzey			
PA-	1	2	4
PB-	2	4	5
PC-	2	3	5
PD-	4	5	6
PE-	5	6	7
PF-	6	7	8

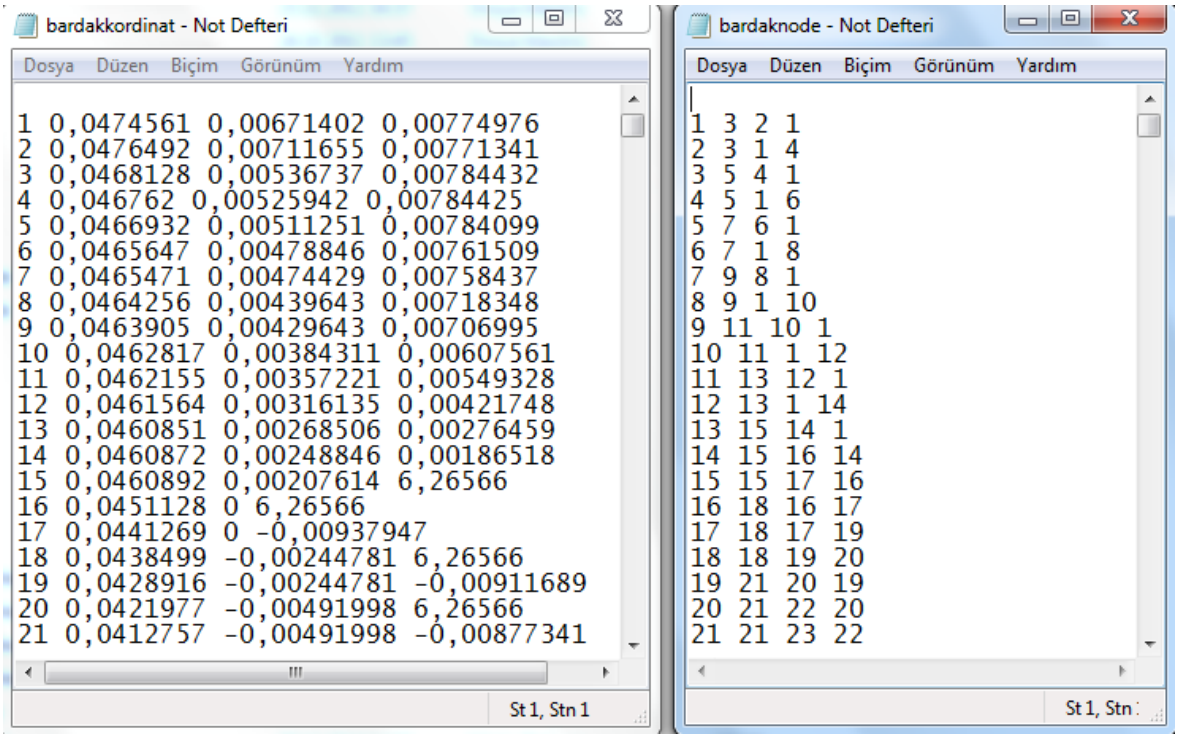
Şekil 5.9. Veri noktalarının üçgenlenmesi

Yukarıdaki şekilde görüldüğü gibi bir yüzeye ait köşe noktaları x,y,z koordinat noktaları ile birleştirilerek çokgenler şeklinde birbirleriyle ilişkilendirilmektedir. Nokta bulutu verilerinin birleştirilmesi aşamasında delaunay üçgenleme yöntemi kullanılmaktadır.



Şekil 5.10. Delaunay üçgenleme

Delaunay üçgenleme yönteminden bahsedecek olursak, öncelikle temel prensip bir üçgenin çevrel çemberi içerisinde başka bir dayanak noktası bulunamamasıdır. Yani üçgeni oluşturacak üç köşeden daha fazla noktanın aynı çember üzerinde yer alması bu kuralın istisnasıdır. Algoritmalarda bu özel durum dikkate alınmalıdır. Bazı belirsizlik durumlarının ortadan kalkması için algoritmaya bir seçim ölçütü konulur. Çevrel çember ölçütü genellikle artan üçgenleme yöntemlerinde kullanılan ölçüttür. Oluşturulacak olan her bir üçgenin 3. noktası belirlenirken bu üç noktadan geçen çevrel çemberin içerisinde bir başka dayanak noktası olup olmadığı kontrol edilir. Çember içerisinde bir başka noktanın olduğu belirlenirse, belirlenen bu nokta yeni oluşacak üçgenin 3. köşesi olmaya aday olur. Daha sonra ise bu aday üçgenin çevrel çemberi içerisinde bir başka nokta olup olmadığı kontrol edilir. Bu işlem çevrel çember içerisinde nokta kalmayana kadar devam eder ve üçgen oluşturulur.



Şekil 5.11. Bir bardak nesnesine ait koordinat ve üçgenleme düğüm noktaları

Geliştirilen programın ana ekranı üzerinde bulunan TxtSave butonuna tıklandığında daha önceden C sürücüsüne kayıtlı nokta bulutu verilerini içeren text dosyasındaki koordinat değerlerini ve düğüm noktalarının değerlerini sıralı bir indis düzenine koyarak verileri OpenGL platformu üzerinde düzgün çalışmasını sağlayacak şekle sokmaktadır.

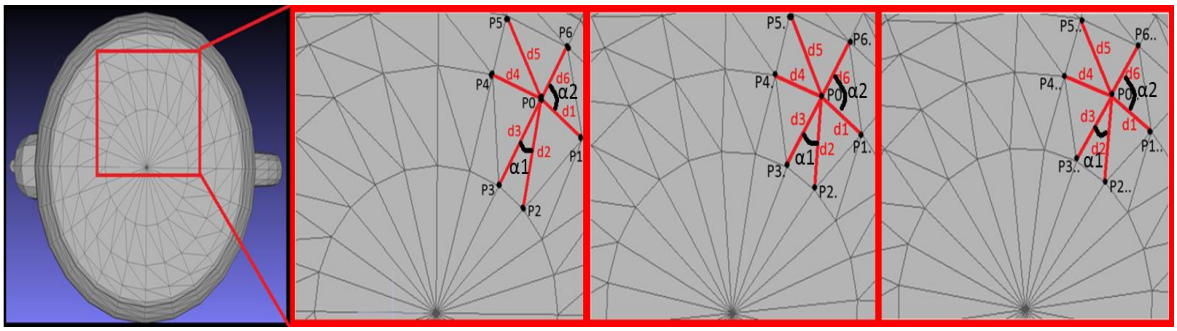
Bundan başka program üzerinde var olan Up ve Down butonları algılayıcı kameranın yere göre açısını ayarlamamıza olanak sağlamaktadır.

Programımızın temel çalışma mantığı C sürücüsüne kayıtlı nokta bulutu koordinatlarının ve bu noktaların birleştirilmesini sağlayacak olan üçgenleme düğüm noktalarının txt dosyasından okunarak OpenGL sanal görselleştirme ortamında ekrana aktarılmasıdır.

5.3.3. Çözüm Ağı Basitleştirme (Yumuşatma)

Üç boyutlu olarak tarama gerçekleştiren hemen her donanımsal cihazlardan elde edilen modelleme sonuçları, her ne kadar düzgün bir yapıya sahip olsa da, modellenen nesnelerin geometrik yapısının karmaşıklığı nedeniyle çoğu zaman olduğu gibi kullanılmaları imkansızdır. Bu nedenle tüm modeller, üzerinde grafik işlemler gerçekleştirmek için, mümkün olabildiğince belirli bir düzene sokulması gerekmektedir. 3-boyutlu modeller oluşturmak ve bu modeller üzerinde bazı işlemler gerçekleştirmek için yaygın olarak kullanılan yöntemlerden biri de poligon temelli ağ yapısıdır (poligonal meshes) [46]. Bu yöntem, nesne modellerinin nokta bulutlarının uzay konumları üzerinde değişiklik yapılarak koordinatlarının daha düzenli bir yapı haline gelmesine olanak sağlar. Poligon yapısı içerisinde, çok sayıda nokta ve bu noktaları birleştiren kenarlar, yüzeyler ve köşeler bulunmaktadır. Bunlar arasındaki ilişkiler incelenerek farklı boyut ve ölçülere sahip olanlar ile bunların komşulukları ile olan durumları göz önüne alınarak değişik türden referanslar uygulanabilir. Hali hazırda var olan yöntemlerin üzerine yenilerini ekleyerek değişik algoritmaların uygulanması sağlanmaktadır. Bu nedenle basitleştirme algoritmaları yöntemlerin gerçekleştirilmesinde önemli bir kısmı oluşturur ve pek çok değişik algoritma geliştirilmiştir [48,49,50,51,52]. Bu çalışma kapsamında mevcut Gaussian yöntemi üzerine farklı bir yöntem uygulanmış sonuçları incelenmiştir.

Modellemeyi oluşturan temel elemanlardan çok sayıdaki nokta, düğüm ve dolayısıyla da poligon içeren büyük yapıdaki poligonsal modellerin basitleştirilmesi ve yumuşatılması bilgisayar grafikleri için önemli bir yer tutar. Ağ basitleştirme işlemlerine olan gereksinimler arasındaki öncelikli neden geometrik nesnelerin karmaşık yapıda olmasıdır. Ayrıca bu yapıların, grafik işleme düzenine sokulmadan önce tarama gerçekleştiren donanımlardan elde edilen haliyle bir ön işleme tabi tutulması gerekmektedir. Bu gereksinimden yola çıkılarak öncelikle önerilen farklı basitleştirme algoritmalarında meydana gelen değişiklikler incelenmiş ve değerlendirmeye tabi tutulmuştur.



Şekil 5.12. Algoritma uygulanan, 2256 Poligondan oluşan çaydanlık modelinden alınmış örgünün bir kesiti

Yukarıda bir basitleştirme algoritmasının birkaç kez uygulanarak aynı kesit alanına sahip olan bir modelin verilerinde meydana gelen değişiklikler gösterilmektedir. Bu çaydanlık modelinin üst yüzeyinden alınan bir kesitine uygulanan algoritma sonucunda, orijinal görüntü ile ilk ve ikinci aşamaları sonrasında alakalı düğüm noktalarının koordinatları ve düğümlerin birbirine olan uzaklıkları arasındaki değişimler incelenmiştir. Bu değişimlerden yola çıkılarak görüldü ki ilgili düğümler arasında birbirine Öklid uzaklığı olarak en uzak olan noktalar algoritma uygulandıktan sonra birbirine daha da yakınlaşmaktadır. Aynı şekilde yakın olanlar da uzaklaşmaktadır. Bu neticeden yola çıkılarak, Gaussian yöntemini temel alan farklı bir algoritma geliştirilmiş ve sonuçları incelenmiştir [52]. Geliştirilen sistemde modelin bir poligon düğümünün bağlı olduğu tüm düğümlere olan Öklid uzaklıkları hesaplanarak en uzak ve en yakın olan düğümler katsayı olarak diğer bağlı düğümlere göre öncelikli duruma geçmektedir. Algoritmaya göre farklı katsayılarla çarpılan bu öncelikli düğümler, diğer kalan düğümlerle birlikte hesaplanarak düğümlerin yeni koordinat değerleri oluşturulmaktadır.

$$Pi_x = \frac{\sum_{i=0}^{n} x_i + (x_{ek} + x_{eb})}{n+2},\ Pi_y = \frac{\sum_{i=0}^{n} y_i + (y_{ek} + y_{eb})}{n+2},\ Pi_z = \frac{\sum_{i=0}^{n} z_i + (z_{ek} + z_{eb})}{n+2} \qquad (5.1)$$

n: P_i ile ilişkili düğümlerin sayıları toplamı.

x_{ek}, y_{ek}, z_{ek} : İlişkili düğümler arasında Öklid uzaklığı en küçük düğüm koordinatı.

x_{eb}, y_{eb}, z_{eb} : İlişkili düğümler arasında Öklid uzaklığı en büyük düğüm koordinatı.

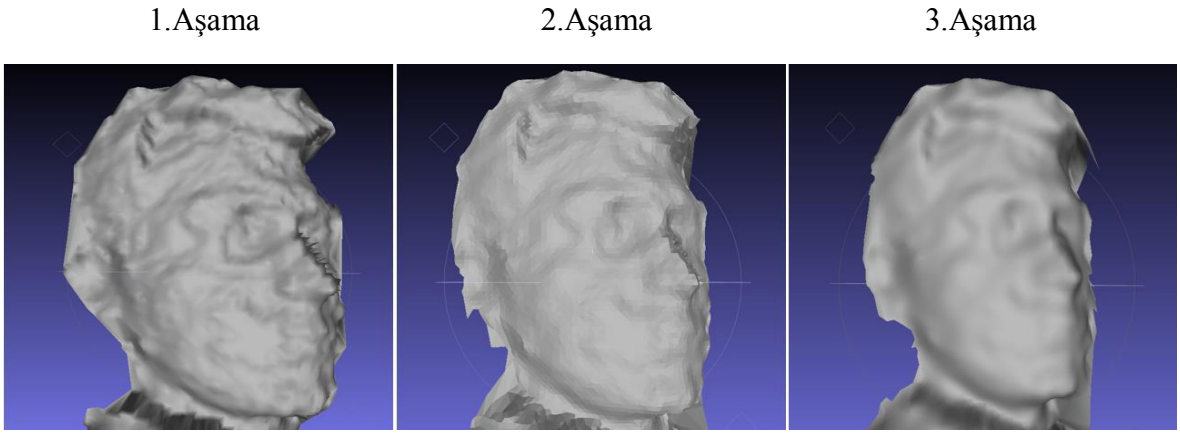
P_i $(_{x,y,z})$: Her bir düğümün bulunan koordinat değeri.

Aşağıda birden fazla basitleştirme algoritması uygulanmış çaydanlık modelinin, bir kesitinin koordinatlarında ve birbiriyle ilişkili düğümlerinin aralarındaki Öklid uzaklıklarında meydana gelen değişimleri gösterilmektedir.

Noktaların Koordinat Değişimleri	**Noktaların Koordinat Değişimleri**	**Noktaların Koordinat Değişimleri**
P0: 1,38137 0 2,45469	**P0:** 1,359634 -0,000603 2,448438	**P0:** 1,337057 0,020553 2,450224
P1: 1,4 0 2,4	**P1:** 1,363122 0,0741844 2,421876	**P1:** 1,333855 0,114904 2,432501
P2: 1,35074 -0,375926 2,4	**P2:** 1,335226 -0,291651 2,421876	**P2:** 1,317554 -0,242121 2,432501
P3: 1,33276 -0,370922 2,45469	**P3:** 1,313011 -0,361698 2,448438	**P3:** 1,296903 -0,333603 2,450224
P4: 1,38426 0 2,4875	**P4:** 1,36091 0,000834 2,481251	**P4:** 1,338637 0,001381 2,475001
P5: 1,33555 0,371699 2,4875	**P5:** 1,314185 0,362255 2,481251	**P5:** 1,286589 0,37278 2,450224
P6: 1,35376 0,376765 2,49844	**P6:** 1,33028 0,367708 2,492188	**P6:** 1,292975 0,354719 2,475001
Noktalar Arası Öklid Uzaklık Değişimleri	**Noktalar Arası Öklid Uzaklık Değişimleri**	**Noktalar Arası Öklid Uzaklık Değişimleri**
d1: 0,608564450021565	**d1:** 0,703925827264045	**d1:** 0,704900314929089
d2: 1,01094078081888	**d2:** 0,926660479802087	**d2:** 0,886171936604949
d3: 0,9107746246766	**d3:** 0,903789286264646	**d3:** 0,891905800368812
d4: 0,484658672431632	**d4:** 0,50476612608753	**d4:** 0,579329762936723
d5: 0,9107746246766	**d5:** 0,903430586657506	**d5:** 0,904509708493443
d6: 1,00242835097346	**d6:** 0,998352724962898	**d6:** 0,97233450506312
dort: 0,821356	**dort:** 0,823487	**dort:** 0,823192
[dort-d1]: 0,212791	**[dort-d1]:** 0,119561	**[dort-d1]:** 0,118291
[dort-d2]: 0,189584	**[dort-d2]:** 0,103173	**[dort-d2]:** 0,062979
[dort-d3]: 0,089418	**[dort-d3]:** 0,080302	**[dort-d3]:** 0,068713
[dort-d4]: 0,336697	**[dort-d4]:** 0,318720	**[dort-d4]:** 0,243862
[dort-d5]: 0,089418	**[dort-d5]:** 0,079943	**[dort-d5]:** 0,081317
[dort-d6]: 0,181072	**[dort-d6]:** 0,174865	**[dort-d6]:** 0,149142
Doğrular Arası Açı Değişimleri	**Doğrular Arası Açı Değişimleri**	**Doğrular Arası Açı Değişimleri**
α1 (P2-P0-P3): 30°	**α1 (P2-P0-P3):** 36°	**α1 (P2-P0-P3):** 39°
α2 (P1-P0-P6): 84°	**α2 (P1-P0-P6):** 80°	**α2 (P1-P0-P6):** 78°
[**α1- α2] :** 54°	[**α1- α2] :** 44°	[**α1- α2] :** 39°
Min α : 7°	**Min α :** 9°	**Min α :** 10°
Max α : 118°	**Max α :** 114°	**Max α :** 112°

Yumuşatma işlemi gerçekleştirirken, yeni değerlerin oluşturulması sırasında (x,y,z) noktalarının yeni koordinatları hesaplanmaktadır "(1)". Yukarıda çaydanlık modelinde görüldüğü üzere P0 düğümüne en uzak kenarlar algoritmaların gerçekleştirilmesi sonrasında yakınlaşmıştır. Noktaların koordinat değişimlerinde de gösterildiği gibi P0 noktasının P2 ve P3'e olan Öklid uzaklıkları sırasıyla d2 ve d3 her algoritmada biraz daha yakınlaşmıştır. İlk algoritma uygulandıktan sonra d2 Öklid uzaklığı 1,0109 değerinden 0,926'ya ve ikinci algoritmada 0,886'ya ulaşmıştır. Aynı şekilde en yakın nokta olan P4'ün Öklid uzaklığını ifade eden d4 değeri 0,484 değerinden ilk algoritma sonrası 0,504'e ve ikinci algoritmada 0,579'a ulaşmıştır. Özetle algoritma mantığına göre birbiriyle ilişkili olan düğümler arasında en uzak olan noktalar yakınlaşmış, en yakın noktalar da birbirinden uzaklaşmıştır. Benzer şekilde ilişkili noktalar arasındaki doğruları birleştiren açılar olabildiğince her algoritma uygulaması sonrası birbirlerinin değerine yakınlaştırılmıştır.

Örneğin orijinal örgü üzerindeki 30°'lik en dar açıya sahip P2-P0-P3 noktaları arasındaki doğruların açısı 1. aşama sonrası 36°'ye 2. aşama sonrasında 39°'ye ulaşmıştır. En geniş açıya sahip P1-P0-P6 noktaları arasındaki doğruların açısı 84° iken 1. aşama sonrasında 80°'ye 2. aşama sonrasında 78°'ye ulaşmıştır. Doğrular arasındaki dar açılar genişlemiş geniş açılar daralarak birbirlerine yakınlaşmıştır. Böylece eşkenara daha yakın bir yapı elde edilmiştir.



Şekil 5.13. 16483 Poligondan oluşan, Algoritma sonrası İnsan modeli görünümü

Yukarıdaki şekillerde 3B algılayıcı kameradan alınan verilerle oluşturulmuş olan model üzerinde gerçekleştirilen algoritma sonrası modelin her bir aşamada daha da pürüzsüz bir yapıya sahip olduğu gösterilmektedir.

Algoritma

Enbüyük → 0

Enküçük → 100

for i : 0 to : poligon sayısı **do**

Poligon[i] ile ilişkili tüm düğümleri bul; **end for**

for i1 : 0 to : poligon sayısı **do**

Poligon[i1] ile ortak olan aynı düğümleri aradan çıkar; **end for**

for i2 : 0 to : poligon sayısı

for j2 : 0 to : i2 **do**

Uzaklık → 0

Poligon[i2]'ye bağlı tüm düğümlerin Öklid Uzaklıklarını hesapla;

If (Uzaklık > Enbüyük) **then**

P[U_Eb] Bu düğümü tut; **end if**

If (Uzaklık < Enküçük) **then**

P[U_Ek] Bu düğümü tut; **end if**

end for

end for

for i3 : 0 to : poligon sayısı

for j3 : 0 to : i3 **do**

Bu düğümün üç koordinatını hafızadan çek;

Her (x,y,z) koordinatları için;

p[i3].düğüm = (p[j3] + P[U_Eb] + P[U_Ek]) / p[i3]+2;

Değeri hesaplanan düğümün eski koordinatını sil;

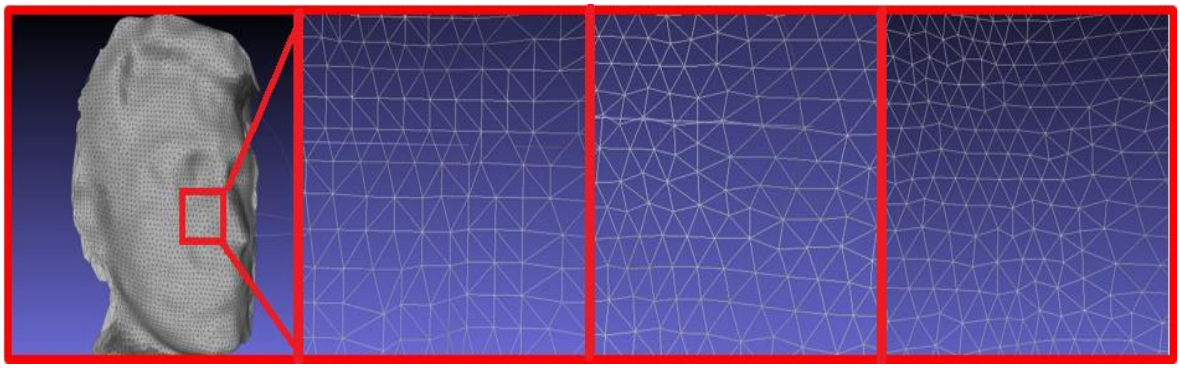
Yeni koordinatı yerine yaz;

Kaydet;

end for

end for

Yukarıda gösterildiği üzere farklı modellerden oluşturulmuş olan veriler üzerine uygulanan algoritmanın karmaşıklığının gerçekleştirme sürelerine olan etkisi incelenmiştir. Karmaşıklık O(n^2) olmasına karşın poligon sayısının fazlalığı işlem süresini artırmaktadır. Algoritmaların uygulanma süreleri, nesnelerin farklı sayıda poligon sayısına bağlı olarak değişmektedir ve aynı zamanda Gaussian yöntemine göre farklı hesaplama tekniği kullanılmasına bağlı olarak gerçekleştirme süresini de etkilemektedir.

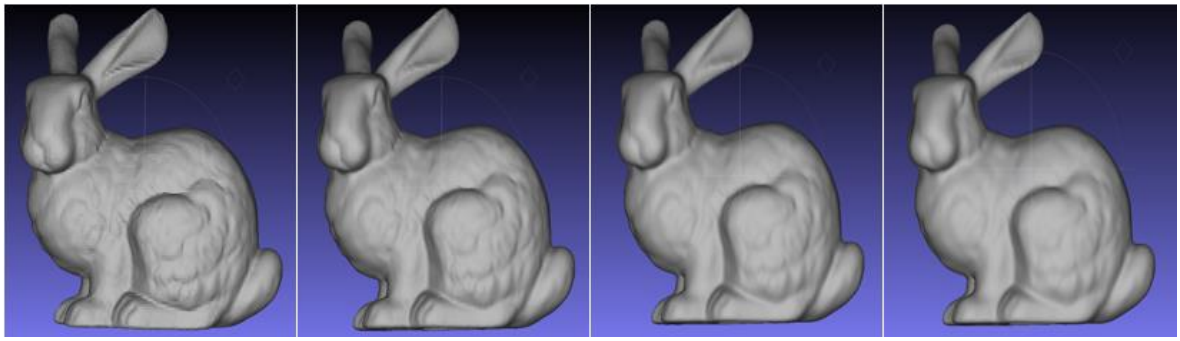


Şekil 5.14. 8254 köşe, 16483 poligona sahip algoritma uygulanan insan modelinden alınan örgünün bir kesiti.

Böylece çalışmamızın bu aşamasında 3B algılayıcı kamera kullanarak nesnelerin 3-boyutlu modellerinin oluşturulması ve ardından oluşturulan bu modeller üzerine ağ basitleştirme tekniği uygulayarak 3-boyutlu nesne modelleme ve görüntüleme sistemi gerçekleştirilmiştir.

Yukarıda gösterildiği üzere modellenen insan görüntüsünün bir kesitine uygulanan algoritmalar sonrasında yeni oluşan modellerin poligon örgülerinde meydana gelen değişimler yer almaktadır. Yeni modellerin poligonlarındaki düğüm değerlerinin her yeni aşamada yerini daha eşkenar yapıya bıraktığı gözlenmektedir. Görüldüğü üzere bu modellerin yüz ve kıvrım noktalarındaki yapısı bozulmalara karşı, olduğu halinden daha dayanıklı bir hale getirilmiştir.

Orijinal 1.Aşama 2.Aşama 3.Aşama

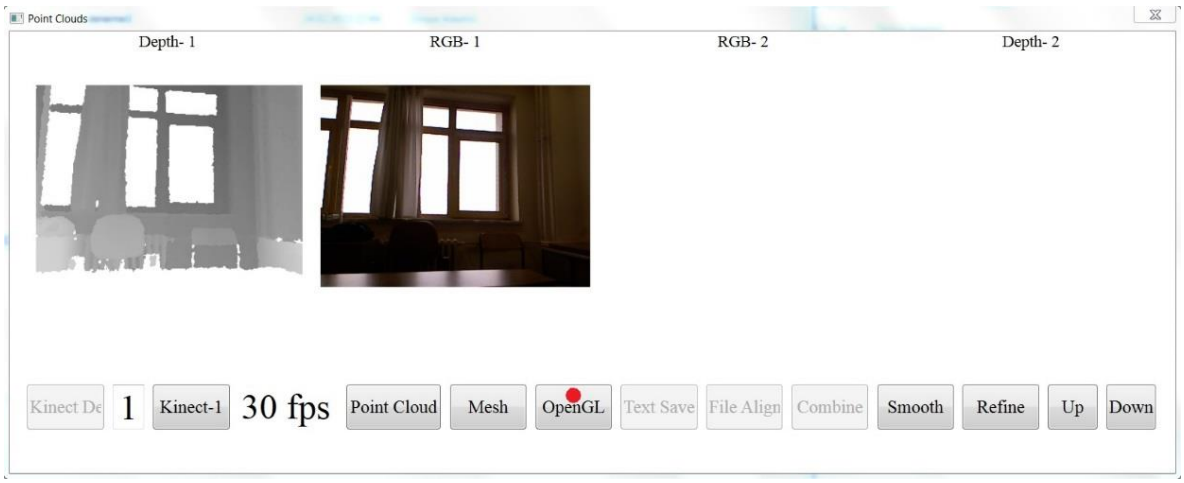


Şekil 5.15. 35947 köşe, 69451 poligona sahip orijinal (sol-1) ve algoritma uygulanan (2-3-4) tavşan modeli

Orijinal 1.Aşama 2.Aşama 3.Aşama

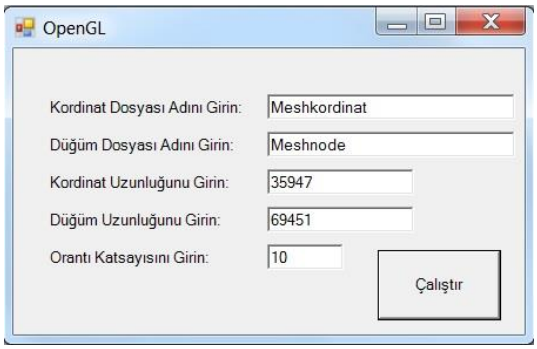
Şekil 5.16. 25003 köşe, 49999 poligona sahip orijinal (sol-1) ve algoritma uygulanan (2-3-4) buddha modeli

Bir nesneye ait alınan veriler ile modelin yeniden yapılandırılması işleminin gerçekleştirilmesinin ardından, modele yumuşatma işlemi uygulanmadan önce ve uygulandıktan sonra yeniden yapılandırılan nesne modelinin OpenGL ortamında gösteriminin yapılabilmesi sağlanmıştır.



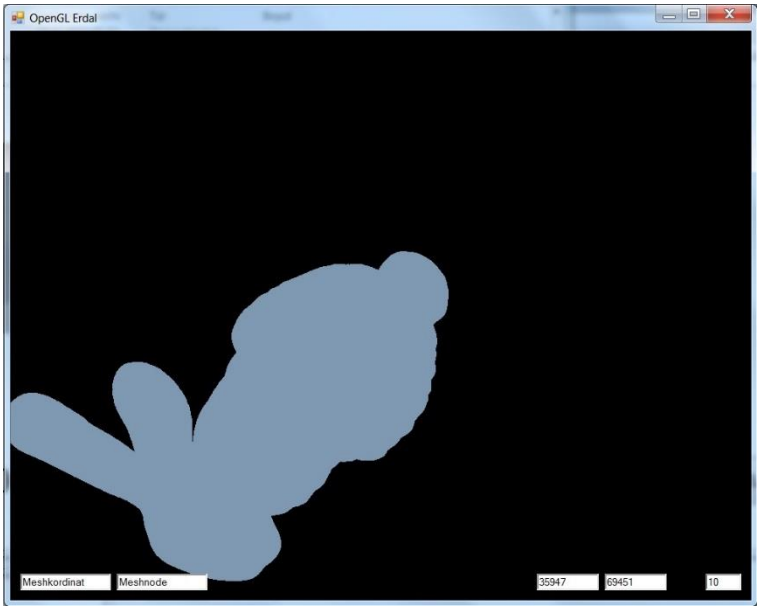
Şekil 5.17. Program üzerindeki OpenGL erişim butonu

Bunun için program üzerinde bulunan OpenGL butonuna tıklanarak, öncelikle C sürücüsü içerisinde oluşturulmuş olan modelleme dosyasına ait bilgileri OpenGL ortamına aktarmak için bir ekran gelir. Bu ekran üzerinde gösterimi yapılacak olan nesneye ait koordinat, düğüm bilgilerinin yanı sıra gösterim büyüklüğünü ayarlamak için, bir orantı katsayısı bilgileri bulunmaktadır.



Şekil 5.18. Gösterimi yapılacak modelin bilgilerinin OpenGL ortamına aktarım ekranı

Yukarıda gösterimi yapılacak tavşan modeline ait koordinat ve düğüm bilgileri tutulmaktadır. Modelin koordinat değerlerinin büyüklüğüne göre orantı katsayısı 10 olarak ayarlanmıştır. Bu ayarlanabilir ortam sayesinde program, modellemesi gerçekleştirilmiş olan tüm nesneler için kullanılabilir bir yapıdadır. Çalıştır butonuna tıklandıktan sonra bu veriler OpenGL ortamına aktarılmaktadır.



Şekil 5.19. Yapılandırılmış nesne modelinin OpenGL ortamında gösterimi

Böylece herhangi bir gerçek dünya nesnesinin yeniden yapılandırılması, bu model üzerine yumuşatma işleminin uygulanması son olarak ta modelin bilgisayar ortamında gösterilmesi işlemleri gerçekleştirilmiş oldu.

5.4. Değerlendirme

Gerçekleştirilen bu tez çalışması ile uzak ortamındaki fiziksel nesnelerin konumlarına ait olan koordinat bilgileri nesneye ve algılayıcıya herhangi bir temas uygulanmadan, aynı açıdan alınmış iki farklı görüntü birleştirilerek, sanal ortama aktarılmaktadır. Bu iki farklı görüntü birleştirilirken, iki parça halindeki bir nesnenin iki yarısının iç ve dış açılarından alınmış görüntülerinin üzerinden işlem yapılması bakımından, diğer modelleme çalışmalarına göre farklı bir yöntemle gerçekleştirilmiştir. Bu yöntemle, 360°'lik fiziksel bir nesne modeli oluşturmak için nokta bulutlarına herhangi bir dönüşüm işlemi uygulanmasına gerek kalmaması sebebiyle, var olan diğer 3B darbeli ve temaslı tarama yöntemlerine göre daha hızlı ve maliyetsiz bir uygulama gerçekleştirilmiştir. Ancak halen bir üç bükey yapıya sahip nesnenin modellemesi sırasında görüntü açılarının kapalı durumda olması dolayısıyla bu tür nesneler için bu yöntem dezavantajlıdır.

Elde edilen kaba modelleme üzerine uygulanan yumuşatma algoritması nokta bulutları üzerine uygulanan kendi geliştirdiğimiz bir hesaplama yöntemini kullanmaktadır. Nokta bulutlarının daha pürüzsüz ve sağlam bir yapı haline getirilmesinde, Gaussian yumuşatma yöntemini temel alan ve bu yöntem üzerine eklenen hesaplama adımlarından oluşmaktadır. Bu açıdan uygulanan yöntem Gaussian yöntemine göre nesnenin ön yüzeyindeki bir yumuşatmadan daha kararlı bir yapı sergilemektedir. Her iki yöntem sonuçları incelendiğinde, uyguladığımız yöntemdeki fazladan işlem adımları noktaların eşkenarlığı konusunda daha iyi sonuçlar çıkarmıştır. Bu açıdan daha sağlam bir yapı oluşturulmasının yanında, fazladan işlem yapılması işlem sürenin uzaması bakımından bir olumsuzluk içermektedir. Bunun da paralel bir hesaplama yöntemiyle giderilmesi düşünülmektedir.

5.5. Sonuç

Bu tez çalışmasında gerçekleştirilen yazılımı kullanarak kapalı bir iç sahnede yer alan herhangi bir nesnenin üç boyutlu modelini çıkarma işlemi hayata geçirilmiştir. Program aynı nesneye ait birden fazla alınan nokta bulutu kümelerinin birleştirilmesi imkanı vermektedir. Nokta bulutu verilerinin daha kararlı bir yapıya dönüşmesini sağlayacak bir yumuşatma algoritması, elde edilen model üzerine uygulanabilmektedir. Son olarak kararlı ve sağlam bir yapıya sahip üç boyutlu fiziksel nesnelerin sanal ortamda (OpenGL) görselleştirilmesi sağlanmıştır. Böylece gerçekleştirdiğimiz tek bir uygulama üzerinden nesnelerin modellenmesi, yumuşatılması ve görselleştirilmesi işlemleri tasarlanmıştır.

6. SONUÇ

Bu tezdeki amacımız fiziksel nesnelerin yeniden yapılandırılması konusunda, çeşitli yöntemler araştırılıp incelendikten sonra, gerçek dünya nesnelerinin 360 derecelik 3B nokta bulutu sunumunu yapabilen işlemler zincirini hayata geçirmekti. Bu anlamda bizler, yeniden yapılandırmak istediğimiz nesneyi diğer verilerden ayırabilen ve fiziksel nesneyi sunan çoklu görüntüleri tek bir nokta bulutu sunumuna birleştirebilen işlemler zincirinin uygulanmasını inceledik. Tezde hayata geçirilen uygulama, gerçek fiziksel nesnelerin ve de bunların içinde bulunduğu bütün sahnenin görüntüsünün yakalanması aşamasında nokta bulutu verilerini oluşturup, elde edilen bu noktaları üçgenleme metoduyla birbirleriyle ilişkilendirip, elde edilen yapılandırma üzerine farklı bir algoritma uygulayarak, yumuşatma işlemini gerçekleştirmeyi sağlamaktadır. Ancak hala daha filtreleme ve birden fazla yakalanmış veriler ile işlem yapma kısmında bazı geliştirmeler olması gerektiği düşünülmektedir.

Nokta bulutu sunumları için MeshLab sanal ortamıyla görselleştirilecek şekilde düzenlenen veriler bunun yanı sıra OpenGL sanal dünya görüntüsüyle birleştirmek için de çaba sarf edildi. Bu düzgün çalışan veri yolu gerçek dünya nesnelerinin sanal dünyaya katılması kavramına bir kanıt sağlar, fakat son kullanıcı uygulaması için hala daha geliştirilmesi gerekmektedir.

Bu çalışmamızın uygulaması, mümkün olduğunca hızlı bir şekilde, Microsoft Kinect algılayıcı kamera kullanılarak, elde edilen işlenmemiş bulutları kullanmak amacıyla geliştirildi. Nesne yakalama için en uygun durum, nesnenin Kinect kamerasından yaklaşık bir buçuk metre uzağında bulunduğu durum olduğu öngörülmektedir.

Bu tezin uygulanması sırasında birçok sorun ve zorluk baş gösterdi. Özellikle de nokta bulutu kaydı için uygulanacak pratik bir yöntem oluşturulmasının zor olduğunu anlamış oldum. Beklenen sonuçları verebilen başarılı yöntemi bulmadan önce nokta bulutu kaydı ve onların filtrelenmesi için birçok yöntem incelendi ve değerlendirildi. Sonunda, bu konu üzerine çalışmak bana bilmediğim konular üzerinde daha iyi çalışmayı ve bilgiyi uygulamaya koyabilmeyi öğretti. Bunun yanında Microsoft Kinect gibi mevcut tüketici sınıfı donanımlarının, büyük ölçüde doğru bir nokta bulutu elde etme olanağı sağladığı için endüstriyel kalite malzemelerinin, nesne taraması için artık bir zorunluluk olmaktan çıktığını göstermiş oldu.

KAYNAKLAR

[1] **Brett A., Brian C., and Zoran P.,** 2003. Thespace of human body shapes: Reconstruction and parameterization from rangescans. ACM Transactions on Graphics, 22(3):587–594.

[2] **Vouzounaras, G., Perez-Moneo Agapito, J. D., Daras, P., & Strintzis, M. G.,** et al. 2010. 3D reconstruction of indoor and outdoor building scenes from a single image. Proceedings of the 2010 ACM workshop on Surreal media and virtual cloning New York, NY, USA: ACM.

[3] **Zheng, H., Yuan, J., & Gu, R.,** 2011. A novel method for 3D reconstruction on uncalibrated images. Proceedings of the Third International Conference on Internet Multimedia Computing and Service New York, NY, USA: ACM.

[4] **Huang, Y. Y., & Chen, M. Y.,** 2011. 3D object model recovery from 2D image sutilizing corner detection. System Scienceand Engineering (ICSSE), 2011 International Conference on June.

[5] **Pheatt C., Ballester J., Wilhelmi D.,** 2005. Low-costthree-dimensional scanning using range imaging. J.Comput. SmallColl. 20(4):13-19 on April.

[6] **Edilson de A., Carsten S., Christian T., Naveed A., Hans-Peter S., and Sebastian T.,** 2008. Performance capture from sparsemulti-view video.In ACM SIGGRAPH 2008 papers, SIGGRAPH'08, pages98:1–98:10, New York, NY, USA, ACM.

[7] **Young M. K., Theobalt C., Diebel J., Kosecka J., Miscusik B., and Thrun S.,** 2009. Multi-view image and tof sensor fusion for dense 3d reconstruction. In IEEE 12th International Conference on Computer Vision Workshops (ICCV Work shops), pages 1542–1549.

[8] **Peter H., Michael K., Evan H., Xiaofeng R., and Dieter F.,** 2010. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In International Symposium on Experimental Robotics (ISER).

[9] **Yan Cuiand Didier S.,** 2011. 3d shape scanning with a kinect. In ACMSIGGRAPH 2011 Posters, pages57:1–57:1, New York, NY, USA, ACM.

[10] **Richard A. N., Shahram I., Otmar H., David M., David K., Andrew J. D., Pushmeet K., Jamie S., Steve H., and Andrew F.,** 2011. Kinectfusion: Real-time dense surface mapping and tracking. In Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on, pages 127–136, on Oct.

[11] **Marek S.,** Scene Reconstruction From Kinect Motion, Doctoral Degree Programme (2), FIT BUT.

[12] **Thrun, S., Burgard W., Fox D.,** 2000. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In Proc. of the IEEE International Conference on Robotics Automation (ICRA).

[13] **Triebel R., Burgard W.,** 2005. Improving simultane ous mapping and localization in 3d using global constraints. In Proc. of the National Conference on Artificial Intelligence (AAAI).

[14] **Konolige K., Agrawal M., Frame SLAM,** 2008. Frombundle adjustment toreal-time visual mapping. IEEE Transactions on Robotics, 25 (5).

[15] **Lemaire T., Berger C., Jung, I.-K., Lacroix S.,** 2007. Vision-Based SLAM: Stereo and Monocular Approaches. International Journal of Computer Vision, 74:343364.

[16] **Miika S.,** 2012. 3D Content Capturing and Reconstruction Using Microsoft Kinect Depth Camera.

[17] Cyberware. http://www.cyberware.com/pricing/domesticPriceList.html, 3 Aralık 2012

[18] **Andreas K., Erhardt B., Reinhard K., and Rasmus L.,** 2009. Timeof-flight sensors in computer graphics. In M. Paulyand G. Greiner, editors, Eurographics-State of the Art Reports, pages 119–134. Eurographics.

[19] Microsoft kinect. http://www.xbox.com/kinect, 2010. 3 Aralık 2012

[20] **Freedman B., Shpunt A., Machline M., and Arieli Y.,** 2008. Depth mapping using projected patterns. Patent Application, WO 2008/120217 A2. 10.

[21] **Xie J., Huang S., Duan Z., Shi Y., Wen S.,** 2005. Correction of the image distortion for laser galvanometric scanning system SCIENCE DIRECT Optics&Laser Technology Vol.37, pp. 305-311.

[22] **Kader Ç., Mehmet Y., İsmet K.,** 2005. Dijital görüntü işleme ile Lazer tarama tekniği kullanarak ürün kalitesinin belirlenmesi. Emo – III. Otomasyon Sempozyumu.

[23] Online:http://www.rob.cs.tu-bs.de/en/research/projects/3dscanner/ 7 Aralık 2012

[24] Online: http://web.media.mit.edu/~dlanman/news.html 9 Aralık 2012

[25] Nokia to buy Trolltech, will become a patron of KDE. [Online]. Available at: http://arstechnica.com/open-source/news/2008/01/nokia-buys-trolltech-will-become-a-patron-of-kde.ars. 2 Aralık 2012

[26] **Bernardini F., Rushmeier H.,** 2002. Computer Graphics forum The 3D Model Acquisition Pipeline Volume 21, Number 2.

[27] [Online]:http://mathnathan.com/2012/03/registering-two-point-clouds/ 5 Aralık 2012

[28] **Tölgyessy, M., & Hubinský, P.,** 2010. The Kinect Sensor in Robotics Education. In Proceedings of 2nd International Conference on Robotics in Education Nov,pp.1–4.

[29] **Jiajun Z.,** et al. 2007. Fast omnidirectional 3D scene acquisition with an array of stereo cameras. 3-D Digital Imaging and Modeling, 2007. 3DIM'07. Sixth International Conference on. IEEE.

[30] **Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., ... & Fitzgibbon, A.,** 2011. Kinect Fusion: real-time 3D reconstruction and interactionusing a moving depth camera. Proceedings of the 24th annual ACM symposium on User interface software and technology New York, NY, USA: ACM.

[31] Kinect: The company behind the techexplains how it works. [Online]. Available at: http://www.joystiq.com/2010/06/19/kinect-how-it-works-from-the-company-behind-the-tech/. 13 Ocak 2013.

[32] **Yan Cuiand Didier S.,** 2011. 3d shape scanning with a kinect. In ACMSIGGRAPH 2011 Posters, pages57:1–57:1, New York, NY, USA, ACM.

[33] **Richard A. N., Shahram I., Otmar H., David M., David K., Andrew J. D., Pushmeet K., Jamie S., Steve H., and Andrew F.,** 2011. Kinect fusion: Real-timedense surface mapping and tracking. In Mixed andAugmentedReality(ISMAR), 2011 10th IEEE International Symposium on, pages 127–136, Oct.

[34] **Rusu, R. B., & Cousins, S.,** 2011. 3D is here: Point Cloud Library (PCL). Robotics and Automation (ICRA), 2011 IEEE International Conference.

[35] About the OpenMP ARB and the OpenMP.org. [Online]. Available at: http://openmp.org/wp/about-openmp/. 9 Aralık 2012

[36] Mesh Lab. [Online]. Available at: http://meshlab.sourceforge.net/. 11 Aralık 2012

[37] **Matyunin, S., Vatolin, D., Berdnikov, Y., & Smirnov, M.,** 2011. Temporal filtering for depth maps generated by Kinect depth camera. 3DTV Conference: The True Vision - Capture, Transmissionand Display of 3D Video (3DTV-CON), on May.

[38] **Smisek, J., Jancosek, M., & Pajdla, T.,** 2011. 3D with Kinect. Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on; Nov.

[39] PCL API Documentation [Online]. Available at: http://pointclouds.org/documentation/tutorials/iterative_closest_point.php. 12 Aralık 2012

[40] PCL API Documentation. [Online]. Available at: http://docs.pointclouds.org/1.5.1/. 1 Aralık 2012

[41] PCL API Documentation. [Online]. Available at: http://pointclouds.org/documentation/tutorials/voxel_grid.php. 4 Aralık 2012

[42] PCL API Documentation. [Online]. Available at: http://pointclouds.org/documentation/tutorials/random_sample_consensus.php. 6 Aralık 2012

[43] Kinect Windows SDK[Online]. Available at: www.kinectforwindows.com. 10 Aralık 2012

[44] PCL Prebuilt binaries for Windows. [Online] http://pointclouds.org/downloads/windows.html. 11 Aralık 2012

[45] **Botsch M., Steinberg S., Bischoff S., Kobbelt L.,** 2002. OpenMesh - a generican defficient polygon mesh data structure.

[46] **Ulucay O., and Sarp E.,** 2004. Resolution adjustable 3-dimensional object modeling. Signal Processing and Communications Applications Conference, 2004. Proceedings of the IEEE 12th. IEEE.

[47] **Gu´eziec A.,** 1999. Locally Toleranced Polygonal Surface Simplification. IEEE Transactions on Visualization and Computer Graphics, 5(2), pp. 178–199, on April-June.

[48] **Lindstrom P., Turk G.,** 2002. Fast and memory efficient polygonal simplification. IEEE Visualization ’98, pages 279–286, October 1998. ISBN 0-8186-9176-X.Botsch M, Steinberg S, Bischoff S, Kobbelt L. OpenMesh - a generican defficient polygon mesh data structure.

[49] **Hoppe H.,** 1996. Progressive meshes. Proceedings of SIGGRAPH 96, pages 99–108, ISBN 0-201- 94800-1. Held in New Orleans, Louisiana, on August.

[50] **Rémi R., & Jarek R.,** 1996. Full-range approximation of triangulated polyhedra. Computer Graphics Forum 15(3), Aug. 1996. Proc. Eurographics ’96 on pages 67-76.

[51] **Schroeder W. J., Zarge J. A., and Lorensen W. E.,** 1992. Decimation of triangle meshes. Computer Graphics (SIGGRAPH’92 Proc.), 26(2):65–70, on July.

[52] **Kovaļovs M., Sisojevs A., Glazs A.,** 2012. A Surface Smoothing Method for a 3D Model of a Medical Object. IFMBE Proceedings. Vol.38: International Symposium on Biomedical Engineering and Medical Physics, Latvia, Rīga, 10.-12. pp74-77. On October.

ÖZGEÇMİŞ

1986 doğumlu olan Erdal ÖZBAY ilk, orta ve lise öğrenimini Elazığ'da tamamladı. 2005 yılında kazandığı Girne Amerikan Üniversitesi Bilgisayar Mühendisliği Bölümünden 2010 yılında mezun oldu. Aynı yıl Fırat Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği (Yazılım) Anabilim Dalında Yüksek Lisans yapma hakkı kazanmıştır. 2011 yılında Karabük Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Anabilim Dalında Araştırma Görevlisi olarak görev yapmıştır. 2012 yılının Haziran ayından itibaren Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Anabilim Dalında Araştırma Görevlisi olarak görev yapmaya devam etmektedir.