T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

GÖRÜNTÜ İŞLEME TEMELLİ 3 BOYUTLU NESNE HARİTALAMA UYGULAMASI

YÜKSEK LİSANS TEZİ

AZİZ DURSUN GÖKTEPE

DENİZLİ, KASIM - 2019

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

GÖRÜNTÜ İŞLEME TEMELLİ 3 BOYUTLU NESNE HARİTALAMA UYGULAMASI

YÜKSEK LİSANS TEZİ

AZİZ DURSUN GÖKTEPE

DENİZLİ, KASIM - 2019

KABUL VE ONAY SAYFASI

Aziz Dursun GÖKTEPE tarafından hazırlanan "**Görüntü İşleme Temelli 3 Boyutlu Nesne Haritalama Uygulaması**" adlı tez çalışmasının savunma sınavı 29.11.2019 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği / oy çokluğu ile Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Dr. Öğr. Ü. Meriç ÇETİN	.....
Üye Prof. Dr. Sezai TOKAT	.....
Üye Doç. Dr. Emre ÇOMAK	.....

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 18/12/2019 tarih ve ...50/13.... sayılı kararıyla onaylanmıştır.

.....

Prof. Dr. Uğur YÜCEL

Fen Bilimleri Enstitüsü Müdürü

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

Aziz Dursun GÖKTEPE

ÖZET

GÖRÜNTÜ İŞLEME TEMELLİ 3 BOYUTLU NESNE HARİTALAMA UYGULAMASI

YÜKSEK LİSANS TEZİ

AZİZ DURSUN GÖKTEPE

PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI:DR. ÖĞR. Ü. MERİÇ ÇETİN)

DENİZLİ, KASIM - 2019

Globalleşen dünya toplumlar için farklı ihtiyaçları ortaya çıkarmaktadır. Gelişen teknolojik uygulamalarla globalleşen dünyanın yarattığı zorlukların üstesinden gelinebilir. Stres kaynağı olarak düşünülen durumların etkisi çeşitli eğlence platformlarında azaltılabilir. Bu tarz uygulamaların en güzeli, insanların eğlenme ya da gezme amacıyla bulundukları ortamların veya bu ortamlarda ilgi çekici gelen nesnelerin başkaları ile paylaşılmasına yönelik geliştirilen uygulamalardır. Bu çalışma ile insanların gittikleri yerlerin veya gördükleri nesnelerin üç-boyutlu haritalarını oluşturabilmeleri, nesne tabanlı oluşturulan bu haritaları sonrasında da bulundukları yerden ayrılmadan yeniden inceleyebilmeleri amaçlanmıştır. Bu çalışma için gerekli olan görüntüler fotoğraf stüdyosu, açık hava ortamı, kapalı alanlar ve kafeler gibi halka açık alanlarda elde edilmiştir. Bu çalışma ile insanların daha rahat okuyup anlayabilecekleri üç-boyutlu haritalar oluşturulmuştur. Bu haritaların oluşturulması sırasında bazı kütüphaneler ve diller kullanılmıştır. Bu çalışmanın üç-boyutlu modelleme sonuçlarına ve uygulama ara yüzlerine çalışmanın sonuçlar bölümünde yer verilmiştir. Hazırlanan ara yüzler ile kullanıcı dostu bir sistem hazırlanmıştır. Kullanıcıların hem yeni haritalar oluşturabilmeleri hem de var olan veya oluşturulan bu haritaları görüntüleyebilmeleri sağlanmıştır. Kullanıcı dostu basit bir ara yüze sahip olması ve maliyetinin düşük olması nedeniyle karmaşık ve pahalı sistemlere göre daha kolay kullanılabilmekte ve ulaşılabilmektedir.

ANAHTAR KELİMELER: Yapay Zeka Teknikleri, Görüntü İşleme, Nesne Haritalama

ABSTRACT

IMAGE PROCESSING BASED 3D OBJECT MAPPING APPLICATION
MSC THESİS
AZIZ DURSUN GÖKTEPE
PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING
(SUPERVISOR:ASSIST. PROF. DR. MERİÇ ÇETİN))

DENİZLİ, NOVEMBER 2019

The globalizing world reveals different needs for societies. With the developing technological applications, the challenges of the globalizing world can be overcome. The impact of situations considered as a source of stress can be reduced on various entertainment platforms. The best of such applications are the applications developed for sharing the environments where people are present for the purpose of having fun or traveling or interesting objects in these environments with others. With this study, it is aimed that people can create three-dimensional maps of the places they visit or the objects they see, and then they can re-examine these object-based maps without leaving their places.The images required for this study were obtained in public areas, photo studios, indoor areas. With this study, thee-dimensional maps have been created for people to understand more easily. Some libraries and software languages were used during the creation of the maps. Three-dimensional modelling results and application interfaces of this study are given in the results section of the study. A user-friendly system has been prepared with the prepared interfaces. Users can create new maps as well as view existing or created maps. Because of its simple user-friendly interface and low cost, it can be used and accessed more easily then complex and expensive systems.

KEYWORDS: Artificial Intelligence, Image Processing, Object Mapping

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

ÖNSÖZ

Globalleşmekte ve gelişmekte olan dünya üzerinde yaşamakta olan insanların bu globalleşme ve gelişime ayak uydurabilmeleri önemlidir. Teknolojinin de gelişimi ile globalleşen bu dünya da insanların imkanları olsun veya olmasın, bu yeni düzene ayak uydurabilmeleri için teknolojinin yardımı gerekmektedir. Bu çalışma ile insanların globalleşen bu dünya üzerinde gezmeye ve görmeye ihtiyaç duydukları yerleri, bulundukları yerden ayrılmadan gezip görmelerine imkan sağlanması amacı üzerine çalışılmış ve bunun sağlanması için 3-Boyutlu haritalar oluşturulmuştur. Çalışmanın en önemli avantajı kullanıcıların hem haritalar oluşturabilmesi hem de bu haritaları gösterebilmesidir.

Bu çalışma süresince desteğini her daim hissettiğim ve bilgi ve tecrübeleriyle yol göstericim olan danışmanım Dr. Öğr. Üyesi Meriç ÇETİN'e, bu çalışma için gerekli olan görüntüleri elde etmemde bana yardımcı olan Ali ERDOĞMUŞ ve Hasan ARSLAN'a ders ve tecrübeleriyle destek olan ve kendimi geliştirmemde yardımcı olan bütün hocalarıma ve beni her daim destekleyen aileme teşekkür etmeyi bir borç olarak bilirim.

1. GİRİŞ

1.1 Tezin Amacı

Globalleşmekte ve gelişmekte olan dünya her gün farklı ihtiyaçları ve zorlukları ortaya çıkarmaktadır. Global dünyanın yaratmış olduğu çalışma ve yaşamını devam ettirme zorunluluğunun yaratmış olduğu stres bu zorluklardan birisidir ve insanların bu stresten dolayı eğlenme, gezme gibi ihtiyaçları ön plana çıkmaktadır. Burada var olan sorun insanların vakitlerinin sınırlı olması ve uzak bölgelere bu zaman probleminden dolayı gezip görmeye gidememeleridir. Bu çalışma ile insanların gittikleri yerlerin üç-boyutlu haritalarını oluşturabilmelerini ve bu haritaları daha sonra bulundukları yerden ayrılmadan yeniden görebilmelerini veya oralara gidebilecek durumda olmayan insanlara oluşturdukları bu haritaları gösterebilmelerini sağlamak amaçlanmıştır. Bu çalışma içerisinde farklı kameralar vasıtasıyla elde edilen görüntülerden otomatik olarak nesne haritalama için görüntü işlemeye dayalı bir yazılım geliştirilmesi planlanmıştır. Böylelikle doğru ve ayrıntılı bir şekilde nesnelerin üç-boyutlu haritalamaları mümkün olabilecek ve elde edilen sonuçlar bir ara yüz gösterilebilecektir. Daha sonrası için ise sanal gerçeklik uygulamaları veya benzeri uygulamalara entegrasyonu sağlanabilecektir.

1.2 Tezin Önemi

Bu çalışma ile üç-boyutlu nesne haritalama yapılırken farklı çeşit ve türlerde kamera görüntüleri aracılığı ile diğer bölümlerde açıklanmış olan farklı metotlar, kütüphaneler veya araçlar ile nesnelerin şekilleri, durumları veya görünümleri gibi özelliklerinin üç-boyutlu olarak modellenerek haritalanması planlanmıştır. Böylelikle bir bölgeye veya nesne ait bir görüntü incelendiğinde o bölge veya nesneye ait özellikler belirlenebilecektir. Ortaya çıkan bu üç-boyutlu nesne haritaları ile arttırılmış gerçeklik veya sanal gerçeklik araçları ile oluşturulan uygulamalar ile olan

etkileşim artacak, hazırlanan ara yüzler ile insanların bölge veya nesne ile olan etkileşimleri sanal olarak sağlanabilecektir.

1.3 Tezin Organizasyonu

Tezin ilk bölümünde tezin hazırlanma amacı, önemi, düzeni ve ilerleyişi hakkında bilgi verilmektedir. İkinci bölümde, problemin tanımına ve literatür çalışmalarına yer verilmektedir. Bu kapsamda, konu hakkında literatürde daha önce yapılan çalışmalardan ve elde edilen sonuçlarından bahsedilmektedir. Üçüncü bölüm yöntem bölümünü oluşturmaktadır. Bu bölümde tez içerisinde kullanılan kütüphaneler ve api'ler anlatılarak teze nasıl uygulandığından ve uygulanırken hangi metotlardan ve yöntemlerden yararlanıldığından bahsedilmektedir. Dördüncü bölümde, tezde geliştirilmiş olan uygulamanın sonuçları hakkında bilgi verilmiştir. Bu sonuçlar örneklendirilerek resim ve tablolarla desteklenmektedir. Beşinci bölümde tezin sonuçlarından bahsedilmektedir. Tezin sonunda elde edilen verilerden ve sonuçlardan bahsedilerek bunların gerçek hayatta uygulanabilirliği hakkında genel bir değerlendirme yapılmıştır.

2. PROBLEM TANIMI VE LİTERATÜR ARAŞTIRMASI

Gelişen teknoloji ile birlikte kameralardan elde edilmekte olan verilerin çözünürlüklerinin artmasıyla beraber nesnelerden elde edilen pikseller de artmıştır ve bu verileri kullanan kişilere daha fazla detay sunulmaktadır. Kameralardan elde edilen bu görüntüler sayesinde üç-boyutlu (3B) haritalama işlemi yapılabilmektedir. Elde edilen bu üç-boyutlu (3B) haritalar, sanal gerçeklik, arttırılmış gerçeklik, oyunlar veya navigasyon/yön bulma uygulamaları gibi farklı bir çok alanda kullanılabilmektedirler.

Ayrıca, nesne haritalama verileri uzaktan algılama verileri ile birleştirilirse karar ve destek üzerine kurulu düzenler, şehrin düzen planının hazırlanması, trafik bölge düzeninin hazırlanması, acil-durum yönetim sistemleri ile tarımsal alanlarda çok çeşitli uygulamalar geliştirilebilir. dan gelmekte olan isteklerle beraber devamlı bir şekilde artma meydana gelmektedir. Bölge arazileri bu istekler için var olan önemli bileşenlerdir.

2.1 Nesne Haritalama

Matsuyama ve diğ. (2004), yaptıkları çalışmada üç-boyutlu video için, gerçek dünyadaki dinamik görsel olayları kaydeden, zamana göre değişen 3B nesne ve yüksek kalitede bir yüzeysel dokuya sahip olan gerçek bir film olduğunu belirtmişlerdir. Son yıllarda yapılan birçok araştırmada, 3B videolar için gerçek zamanlı ve tamamen üç-boyutlu şekillerin yeniden yapılandırma sistemlerinin geliştirilmiş olduğu ve bu sistemlerin tümünün, insan vücudunun eylemlerini yakalamaya odaklandığı belirtilmektedir (Matsuyama ve diğ., 2004).

Matsuyama ve diğ. (2004) göre, üç boyutlu video dünyasını geliştirmek ve onu günlük hayatta kullanmak için bazı teknik sorunların çözülmesi gerekir. Bu teknik sorunlar şu şekilde sıralanabilir :

- Hesaplama Hızı (Computation Speed)
- Yüksek Doğruluk (High Fidelity)

- Geniş-alan Gözlem (Wide-area observation)
- Veri Sıkıştırma (Data Compression)
- Düzenleme ve Görselleştirme (Editing and Visualization)

Niem ve Broszio (1995) yaptıkları çalışmada gerçek nesnelerin otomatik bir şekilde üç-boyutlu olarak modellenebilmesi için çoklu kamera görüntüleri ile uygulanan bir doku eşleşme algoritması oluşturmuşlardır. Elde edilen modeller film, reklam, araç sürme veya uçuş simülatörleri vb. alanlarda uygulama alanı bulmuştur. Bu uygulamalar için düşük maliyetli ve basit ekipmanlı olarak gerçekçi modeller oluşturulması gerekmektedir. Gerçekçi görüntüler elde edebilmek için ise çoklu kamera görüntüleri ile çalışılmalıdır. Niem ve Broszio'nun yaptığı çalışmada çoklu görüntüleri elde edebilmek için sabitlenmiş bir kamera ile dönen bir yüzey üzerine yerleştirilmiş bir nesne kullanılmıştır. Bu görüntüler ile nesnenin şeklini otomatik olarak tahmin edecek teknikler oluşturulmuştur.

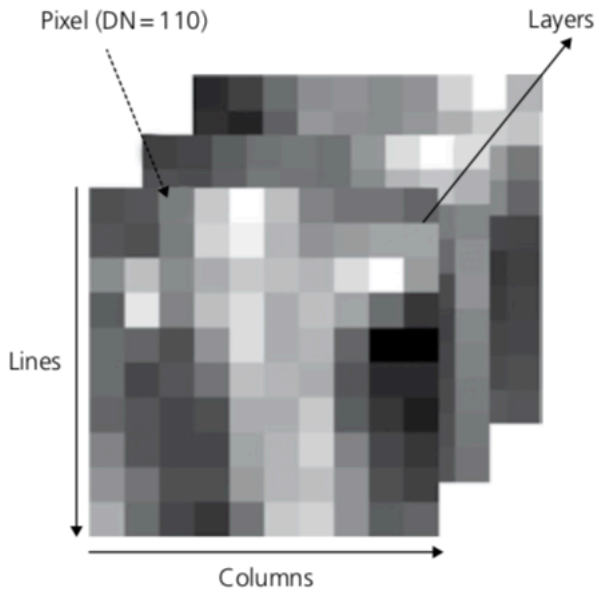
Foley ve diğ.(1990)'ye göre bilgisayar grafikleri, bilgisayarların tanıtılmasından sonra ortaya çıkan plotterlar ve katot ışın tüpü (CRT) ekranlarının verileriyle başlamıştır. Bilgisayar grafikleri nesnelerin model ve görüntülerinin oluşturulmasını, depolanmasını ve manipülasyonunu içerecek şekilde büyümektedir. Bu model ve görüntüler, çeşitli ve geniş bir alandan gelmekte ve fiziksel, matematik, mühendislik, mimarlık vb. içerikler barındırmaktadır. Bilgisayarların grafikleri büyük ölçüde etkileşimli olduğu için kullanıcıların, nesnelerin ve görüntüleme cihazlarının içeriği, yapısı ve uygulaması kontrol edilir. Bilgisayar grafikleri, gerçekçi veya hayali nesnelerin bilgisayar tabanlı modellerinden resimli olarak sentezlenmesi ile ilgili olarak elde edilebilir. Buna karşılık olarak ise görüntü işleme işleme tabanlı uygulamalarda sahnelerin analizi veya modellerin resimlerden elde edilen iki-boyutlu veya üç-boyutlu nesneler olarak yeniden yapılandırılması söz konusudur (Foley ve diğ., 1990).

Huber ve diğ.(2004), yaptıkları çalışmada serbest biçimli (free-form) üç boyutlu nesne tanıma için bilgisayar görüntüsü için bir çok başarılı yaklaşım olduğundan ama buna karşılık olarak daha önce görülmemiş olan bir nesnenin genel bir nesne sınıfına atanması gereken 3B nesne sınıflandırılmasının bir sorun olduğundan bahsetmişlerdir ve çalışmalarında bu sınıflandırma problemine karşı parça tabanlı (parts-based) bir yaklaşım ele aldıklarını söylemişlerdir. Mevcut sınıflandırma çalışmalarında

nesnenin tam bir yüzey modeline ihtiyaç duyulur ancak Huber ve diğ.(2004)'nin çalışmalarında parça temelli yaklaşım sayesinde kısmi bir modelde sınıflandırma yapılabilmektedir. Nesne sınıflandırmasının temel avantajı; önceden görülemeyen nesnelerin tanınmasını sağlamasıdır. 3B tanıma algoritmaları genellikle tanınan nesnelerin bilinen nesne modellerinin veri tabanından çekildiğini varsaymaktadır. Aynı çalışmada nesne sınıflandırmasının, çok sayıda nesneyle büyük nesneler sınıfındaki benzerlikleri kullanarak ve bir sınıftaki tüm nesneler tarafından paylaşılan ortak yapıdan yararlanmayı sağlayan doğal bir yol olduğu belirtilmiştir (Huber ve diğ., 2004).

2.2 Görüntü İşleme

Görüntü; sahne, resim, fotoğraf veya nesnelerin iki-boyutlu (2B) olarak temsil edilmesidir. Görüntü renkler ve tonlarla sunulmaktadır. Dijital görüntü ise iki-boyutlu sayı dizisidir. Bu dizide her bir sayının bulunduğu hücrelere ise piksel denmektedir. Her bir pikselin parlaklığını temsil eden sayıya ise dijital sayı (DN) adı verilmektedir. Şekil 2.1'de görüldüğü gibi, dijital bir görüntü 2B bir dizi olarak satır ve sütunlardaki verilerden oluşmaktadır. Bir pikselin konumu DN'sinin satır ve sütununa atanmaktadır. Dijital görüntüler veri dizileri oldukları için matematiksel işlemler kolayca bu dizi verileri ile gerçekleştirilebilmektedir. Bu matematiksel işlemlere ise dijital görüntü işleme denmektedir (Liu ve Mason, 2016).



Şekil 2.1 Dijital görüntü ve öğeleri (Liu ve Mason, 2016).

Gonzalez ve Woods (2008)'e göre ise, görüntü iki-boyutlu bir fonksiyon olarak, f(x,y), tanımlanmaktadır. Buradaki x ve y uzaysal (spatial) koordinatlardır ve f fonksiyonunun herhangi bir koordinattaki genliği görüntünün o koordinattaki yoğunluğu veya gri seviyesi olarak adlandırılmaktadır. Bu durumda x, y ve f'nin yoğunluk değerleri tamamen sınırlı veya ayrık değerler olduğunda bu görüntü, dijital görüntü olarak adlandırılmaktadır. Dijital görüntü işleme, dijital görüntülerin dijital bir bilgisayar tarafından işlenmesini ifade etmektedir. Dijital görüntüyü oluşturan değerler; görüntü öğeleri, resim öğeleri veya pikseller olarak adlandırılmaktadır. Piksel ise dijital görüntüyü oluşturan değerler veya öğeler için en çok kullanılan isimdir (Gonzalez ve Woods, 2008).

Görme insan duyuları içerisinde en ileri olandır, bu nedenle görüntülerin insan algısında tek önemli rolü oynaması şaşırtıcı değildir. Elektromanyetik (EM) spektrumun görsel bandı ile sınırlı olan görüntüleme makineleri, gama radyo dalgaları kadar geniş bir yelpazede EM spektrumunun hemen hemen tamamını kapsar. Bunlara örnek olarak ultrason, elektron mikroskopisi ve bilgisayarlar tarafından üretilen görüntüler bulunur. Böylece dijital görüntü işleme geniş ve çeşitli alanlarda uygulamaları kapsar (Gonzalez ve Woods, 2008).

2.2.1 Görüntü İşleme Temelli Bölütleme

Gonzalez ve Woods (2008) göre bölütleme, bir görüntüyü kurucu bölgelere veya nesnelere ayırma işlemi olarak tanımlanmıştır. Alt bölümlerin taşıdığı detay seviyesi, çözülmekte olan soruna bağlıdır. Başka bir deyişle, bir uygulamada ilgilenilen nesneler veya bölgeler tespit edildiği zaman bölütlemenin durması gerekir. Gonzalez'in kitabında, elektronik düzeneklerin otomatik düzenlenmesi ile ilgili olarak, eksik bileşenler veya kopuk bağlantı yolları gibi spesifik anomalilerin varlığının veya yokluğunun belirlenmesi amacı ile görüntülerin analiz edilmesi gerektiğini belirtilmiştir. Bu unsurları tanımlamak için gereken ayrıntı düzeyini geçerek bölütlendirmeyi taşımanın anlamı yoktur. Önemsiz veya gereksiz resimlerin bölütlendirilmesinin ise görüntü işlemede en zor görevlerden biri olduğu bilinmektedir. Bölütlemenin doğruluğunu, bilgisayarlı analiz prosedürlerinin nihai başarısı veya başarısızlığı belirlemektedir. Bu nedenlerden dolayı, doğru bölütlendirme olasılığını arttırmak için büyük bir özen gösterilmesi gerekmektedir (Gonzalez ve Woods, 2008).

2.2.2 Görüntü İşleme Temelli Sınıflandırma

Sayısal görüntü üzerine sınıflandırma işleminde görüntü, piksellerine ait olan bazı özelliklere göre belirli gruplara ayrılırlar. Sayısal görüntüyü oluşturan en temel ve küçük eleman şeklinde tanımlanan piksellerin, belirli bir anlama gelen ifadelere göre kategoriler şeklinde ayrılmasına başka bir bakışla görüntünün sınıflandırılması konusuna odaklanan çalışmalar uzun süredir ilgi odağı olmuştur. Çünkü sınıflandırma sonuçları ekolojik ve sosyoekonomik alanda yapılan çalışmalar için temel oluşturmaktadır (Lu ve Weng, 2007) (Zhang ve diğ., 1998). Görüntü sınıflandırma metotları genel olarak görüntülere ait şekilsel ve dokusal nitelikleri kullanırlar (Zhang ve diğ., 2016). Sınıflandırma için incelenen objeler şekil ve doku bazında birbirine yakınlık anlamında değişiklik gösterebilmektedirler. Sınıflandırma teknikleri piksel veya nesne tabanlı olarak elde edilebilmektedirler. Bu tür verilerde nesne ebatları ve piksel sayıları arasında meydana gelmekte olan ilişki piksellere bağlı sınıflandırma tekniklerinde etken şeklinde kullanılamıyor olması sebebiyle nesne-odaklı sınıflandırma tekniklerine olan alaka çoğalmıştır (Blaschke, 2010).

Nesne-odaklı sınıflandırma teknikleri bölütlere ayrılma ve durumsal olarak meydana gelen bu sürecin birleşmiş halidir. Burada ilk etap görüntünün bölütlenmesidir (Castilla ve Hay, 2008). Bölütleme işlemi, nesnelerin spektral, geometriksel, dokusal ve bunlara benzer diğer niteliklerinin sınıflandırılması işlemine bağlı olarak düzenlenmekte olan verinin homojen bir şekilde piksel bölütlerine ayrılmasıdır (Veljanovski ve diğ., 2011). Görüntü bölütleme işlemine uygun girdileri seçimi önemli bir işlemdir. Ayrıca bölütlemenin başarılı olabilmesi için tekniğin kullanım şekline bağlı olarak kalitesi yüksek bir veri kullanımı en önemli etkenlerden biridir (Yu ve diğ., 2006; Lu ve Weng, 2007; Mallinis ve diğ., 2008; Moran, 2010). Örneğin, yüksek-yer küresel-çözünürlük ("VHSR") uyduları aracılığıyla elde edilmiş olan verilerin içerisindeki nesnelerin ebatları ve biçimleri, güncel uygulamalar için olanak tanıyan nesne-tabanlı sınıflandırma teknikleri ile daha çok ayrılabilmektedir. Bu yazılımlar içerisinde var olan sınıflandırma algoritmaları tarım alanları gibi parsel kenarları ve dikim alanlarının belirlenebilmesinde kullanılabilmektedir. Nesne-tabanlı sınıflandırma ile yapılan farklı çalışmalar ile tarım arazileri için sıra (Bobillet ve diğ., 2003), dikim tarzları (Ersan, 2013) ve yaprak yoğunlukları tespit edilebilmiş ve (Bobillet ve diğ., 2003), her bir parselin sınırsal olarak bölünmesine olanak sağlanmıştır. Farklı bir örnek olarak, Wassenaar ve diğ.,(2000) çalışmalarında "Quickbird-2" isimli uydu verilerini kullanarak pikseller ve nesne-odaklı sınıflandırma teknikleriyle "yağ gülü üretimi" yapılmakta olan parsellerin belirlenmesini gerçekleştirmiştir.

Literatürde yer alan farklı çalışmalarda sayısız otomatik/yarı-otomatik olarak yol tespit çalışmaları ile ilgili yaklaşımlar ileri sürülmüştür. Bu çalışmaların arasında; görüntü işleme, sinyal işleme temelli yöntemler, bilgiye dayalı metotlar, denetimli öğrenme, bölütleme teknikleri yer almaktadır. Yapılan bazı çalışmalar sırasında "Canny" filtreleme yöntemi ve "Hough" dönüşüm yöntemi tarzında bazal görüntü işleme teknikleri kullanılmaktadırlar (Zhao ve diğ., 2002; Wang ve diğ., 2005; Vandana ve diğ., 2002; Zhang ve diğ., 1999). Zhao ve diğ. (2002) çalışmalarında ticari amaç ile kullanılmakta olan bir uzaktan algılama yazılımı ile olası yol alanları elde edilmiş, daha sonra ise o yolların sahip oldukları kenarların piksellerini elde edebilmek için görüntüler üzerinde Canny filtreleme yöntemi kullanılmıştır. Kenar pikselleri üzerindeki çizgiler ile keskin bölünme noktaları üzerinden ayrıştırılmış ve benzer doğrultu üzerinde olan kenarlar birbirlerine eklenmişlerdir. Canny filtreleme

yöntemine ilave olacak şekilde Hough dönüşüm yöntemi de yol tespiti amacıyla kullanılmıştır (Wang ve diğ., 2005). Başka bir yarı-otomatik yol tespiti çalışmasında kenar bulunumunun ardından kullanıcı aracılığıyla elde edilen başlangıç noktalarına üzerinde ortalama değişim, taraf ve ebat bilgileri aracılığıyla iz takibi yöntemi ile yol tespiti uygulanmıştır (Vandana ve diğ., 2002). Morfolojik temelli bir diğer çalışma olan Zhang ve diğ. (1999) çalışmasında gri seviye sınıflandırması, objelerin ebat ve şekilleri ile ilgili testleri ve morfolojik amaçlı işlemler sıralı olarak yapılmıştır. Ancak belirli objeler ile ulaşımı engellenmiş yolların tespitinin başarısız bir şekilde sonuçlanmış olması özellikle belirtilmiştir.

Géraud ve Mouret (2004)'in bölütleme kullanılan çalışmalarında, "Water-Shed" isimli algoritma kullanılarak, bölütleme işlemi ile elde edilen verilerde meydana gelen sınırlara yönelik bitişik eğri grafiği (curve adjacency graph) hazırlanmış olup bitim işlemi için "Markov Rastgele Alan modelini" (MRF) elde edilen grafikler aracılığı dahilinde yol düzeni oluşturmak amacı ile kullanılmıştır. Sirmacek ve Unsalan (2010) çalışmalarında pankromatik veriler ile Canny kenar çıkarım yöntemi ile uzaysal ağırlıklandırma (voting) yöntemi uygulanmıştır. Sonrasında ağırlıklandırılmış verilerde bölütleme yapılarak yolların başlangıç pikselleri elde edilmiştir. Bu pikseller ile yapılan işlemlerde ilk olarak izleme (tracking) algoritması tekrar ağırlıklandırılmış matriste kullanılarak yol düzen haritası elde edilmiştir. Bilgiye bağlı yaklaşımlar yine yol düzeni çıkarım işlemi için uygulanmıştır. Örnek olarak Lee ve diğ. (2000) çalışmalarında bölütleme işlemi ile elde edilen sonuç ile gri düzey analizi, bölütlerin ebatlarının oranları ile alakalı veriler kullanılarak yol bölütleri oluşturulmuştur.

Gözetim ile öğretim teknikleri (Amini ve diğ., 2002), frekans bazlı yaklaşımlar (Hu ve diğ., 2007), k-ortalama sınıflandırma (Amini ve diğ., 2002), bulanık c-ortalama sınıflandırma (Liu ve diğ., 2003), genetik algoritmalar, Markov Rastgele Alanlar (MRF) metodu (Géraud ve Mouret, 2004) gibi teknikler bölütlenen görüntülerin eşleştirilmesi amacıyla kullanılan yöntemlerdendir.

2.3 Yapay Zeka

Yapay zeka (YZ), makinelerin akıllı bir şekilde düşünüp davranmalarını sağlamanın bir yoludur. Bu makineler içermekte oldukları kodlar tarafından kontrol edilmektedirler. Bu nedenler YZ'nin bu makineleri kontrol eden yazılım programları ile de ilgilidir. Aynı zamanda makinelerin dünyayı anlamaları ve bu durumlara göre insanların verdiği tepkiler gibi tepkiler vermelerine yardımcı olabilecek teorileri ve metodolojileri bulma bilimi olarak da adlandırılabilir (Joshi, 2017).

Nilsson (1998)'a göre yapay zeka genel olarak tanımlanırsa, eserlerdeki akılla davranışlar ile ilgilidir. Akıllı davranışlar sırasıyla, karmaşık ortamlarda algı, muhakeme, öğrenme iletişim kurma ve hareket etmeyi içermektedir. Yapay zekanın uzun vadeli hedeflerinden birisi, insanların yapabildiği, insanların yapabildikleri veya insanların yapabildiklerinden daha iyilerini yapabilmektir. Yapay zekanın bir başka amacı ise makinelerde veya insanlarda meydana gelen bu davranışları anlamaktır.

Poole ve diğ. (1998) yapay zeka ile ilgili yaptığı tanıma göre, yapay zeka hesaplamalı zeka olarak adlandırılan bir alan için belirlemiş bir addır. Hesaplamalı zeka ise tasarım ajanlarının/değişkenlerinin (agent) incelenmesi olarak tanımlanmaktadır. Ajanlar ortamda bulunan solucanlar, köpekler, termostatlar, uçaklar, insanlar veya organizasyonlar gibi birtakım oldular veya nesneler olabilmektedir.

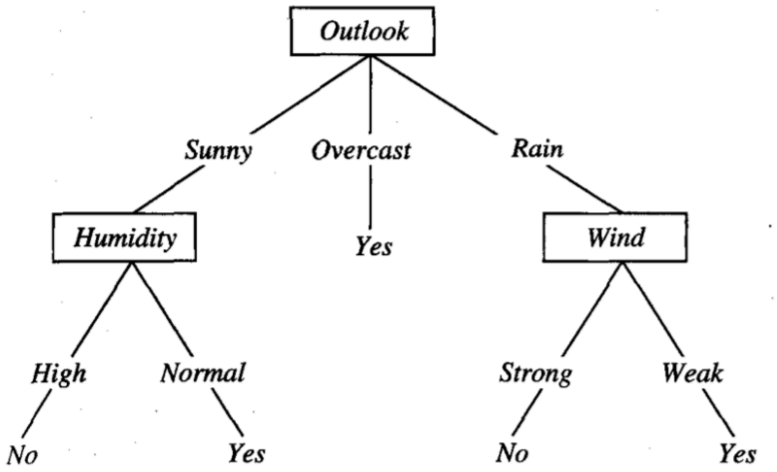
Yapay zekanın bir çok alt dalı vardır. Bu alt dallardan bazıları Mitchell (1997)'a göre aşağıdaki şekilde listelenmiştir.

- Karar Ağaçları ile Öğrenme
- Yapay Sinir Ağları
- Genetik Algoritmalar

Bu alanlar aşağıda genel ve özet şeklinde tanımlanmıştır.

2.3.1 Karar Ağaçları ile Öğrenme

Karar ağaçları ile öğrenme, öğrenilmekte olan bir fonksiyonun bir karar ağacı ile temsil edilmekte olduğu ayrık değerli hedef fonksiyonlarına yaklaşmak için kullanılmakta olan bir yöntemdir. Öğrenilen ağaçlar aynı zamanda okunabilirliği geliştirmek ve kolaylaştırmak için eğer-o zaman (if-then) kural kümeleri ile yeniden temsil edilebilmektedirler. Karar ağaçları, örnekleri ağacın kökten bazı yaprak düğümlere doğru sıralayarak sınıflandırılmasını sağlayan bir sınıflandırma yapmaktadır. Ağaçta bulunan her bir düğüm, örneğin bazı özelliklerinin testini belirtir ve bu düğümden inmekte olan her bir dal, bu özellik için olası değerlerden birine karşılık gelmektedir. Şekil 2.2 karar ağaçları için örnek olarak gösterilebilir. Bu şekilde tenis oynamak için havanın uygun olup olmadığını karar ağacı ile bulmak için oluşturulmuştur (Mitchell, 1997).

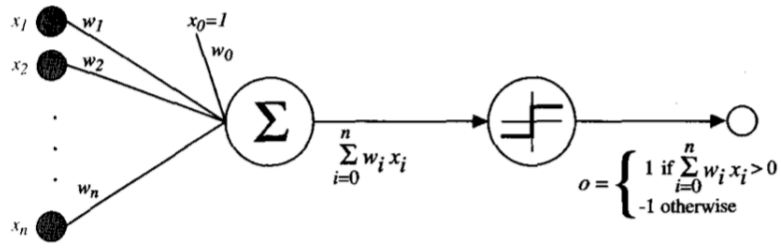


Şekil 2.2 Tenis oynama konsepti için bir karar ağacı (Mitchell, 1997).

2.3.2 Yapay Sinir Ağları

Yapay sinir ağları (YSA) ile eğitim yöntemleri, gerçek değerli, ayrık değerli ve vektörel değerli hedef fonksiyonlarına yaklaşmak için bir yöntem sunmaktadır. Bir YSA sistemi, Şekil 2.3'te gösterildiği gibi "perceptron" olarak adlandırılan bir birime dayanmaktadır. Perceptron gerçek değerlere sahip girdilerin bir vektörünü almaktadır ve bu girdilerin doğrusal kombinasyonunu hesaplamaktadır, ardından

sonuç eşik değerinden eşit veya büyükse 1 değerini vermektedir değil ise -1 değerini vermektedir (Mitchell,1997).



Şekil 2.3 Perceptron (Mitchell, 1997).

2.3.3 Genetik Algoritmalar

Genetik algoritmaların (GA) ele aldığı sorun, en iyi hipotezi tanımlamak için aday hipotezler uzayında araştırma yapmaktır. GA'da "en iyi hipotez", hipotez uygunluğu (fitness) olarak adlandırılmakta olan, var olan sorun için önceden tanımlanmış çözümü optimize etmek olarak tanımlanmaktadır. Örneğin, eğer öğrenme görevi girdi ve çıktısının eğitim örnekleri verilen bilinmeyen bir fonksiyona yaklaşma problemi ise, bu eğitim verisi üzerindeki hipotezin doğruluğu uygunluk (fitness) olarak tanımlanabilmektedir. Eğer görev, satranç oynamak için bir strateji öğrenmekse uygunluk (fitness), diğer kişilere karşı oynarken, bireyin kazanmakta olduğu oyun sayıları olabilmektedir (Mitchell, 1997).

3. MATERYAL VE YÖNTEM

Bu tez hazırlanırken kullanılmış olan materyaller ve yöntemler bu bölümde anlatılmıştır. Bu kapsamda ilk olarak, tez hazırlanırken kullanılan kütüphanelerden, api'lerden, ide'lerden ve dillerden bahsedilmiştir. Sonrasında ise kullanılan materyaller ile oluşturulmuş olan uygulamanın, hazırlanışından ve yöntemlerden bahsedilmiştir.

3.1 Kullanılan Materyaller

Bu alt bölümde tez hazırlanırken kullanılan materyallerden bahsedilmiştir.

3.1.1 Python

Python öğrenmesi kolay ve güçlü bir programlama dili olarak tanımlanmaktadır. Verimli yüksek-seviyeli veri yapılarına ve nesne-tabanlı programlamaya basit ve oldukça etkili bir yaklaşım sunmaktadır. Python'un zarif sözdizimi (syntax) ve dinamik yazımı, yorumlanabilir doğası ile birçok büyük platformda ve birçok alanda komut (script) dosyası ve hızlı uygulama geliştirme için uygun ve ideal bir dildir (Python, 2018[a]).

Python yorumlayıcısı, C veya C++ ile (veya C ile çağrılabilen diğer diller) uygulanabilen yeni fonksiyon ve veri tipleri ile kolay ve basit bir şekilde genişletilebilmektedir. Python özelleştirilebilir uygulamalar için de bir uzantı dili olarak kullanılabilmektedir (Python, 2018[a]).

Python sözdizimine örnek olarak aşağıda Şekil 3.1 gösterilmektedir. Bu şekilde örnek olarak Fibonacci serisinin bulunmasının Python ile hazırlanmış kod kümesi gösterilmektedir.

```
# Fibonacci numbers module

def fib(n):    # write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()

def fib2(n):   # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result
```

Şekil 3.1 Python sözdizimi örneği (Python, 2018[b]).

3.1.2 OpenDroneMap

OpenDroneMap hava araçları ile elde edilen görüntülerin işlenmesi için kullanılan bir açık kaynaklı araç seti olarak tanımlanmaktadır. Standart dronlar basit bas ve çek kameralar kullandıkları için bu dronlardan alınan görüntüler standart bas ve çek kameralardan yani metrik olmayan görüntülerden alınan fotoğraflara benzemektedir. OpenDroneMap, bu basit görüntüleri diğer coğrafi veri kümeleriyle birlikte kullanılabilmeye olanak sağlayan üç-boyutlu (3B) coğrafi verilere dönüştürür. OpenDroneMap kısaca, ham bir şekilde elde edilen insansız hava aracı (İHA) görüntülerini diğer faydalı ürünlerde işlemek için kullanılan bir araç zinciridir. Bu ürünlere örnek verilecek olursa; Nokta Bulutları (Point Clouds), Dijital Yüzey Modelleri (Digital Surface Models), Dokulu Dijital Yüzey Modelleri (Textured Digital Surface Models), Sınıflandırılmış Nokta Bulutları (Classified Point Clouds), Dijital Yükseklik Modelleri (Digital Elevation Models) vb (Opendronemap, 2018).

3.1.3 OpenCV

OpenCV, açık kaynaklı bir bilgisayar görüşü ve makine öğrenimi yazılım kütüphanesidir. OpenCV, bilgisayar görüşü uygulamaları için ortak bir altyapı sağlamak ve makine algı kullanımını hızlandırmak için inşa edilmiştir. OpenCV, BSD lisansı ile kodların kullanımını ve değiştirilebilmesini kolaylaştırmıştır. Kütüphane hem klasik hem de modern bilgisayar görüşü ve makine öğrenmesi

algoritmalarını kapsayan 2500'den fazla optimize edilmiş algoritmaya sahiptir. Bu algoritmalar ile yüz algılama, yüz tanıma, nesne algılama, nesne tanıma, videolar üzerinden insan hareketlerini sınıflandırma, kamera hareketlerini izleme, hareketli nesneleri izleme, nesnelerin üç-boyutlu (3B) modellerini çıkarma, stereo kameralar kullanarak 3B nokta bulutları (point clouds) üretme gibi işlemler yapılabilmektedir (OpenCV, 2000).

3.1.4 Docker

Bir konteyner, bir kodu ve tüm bağımlılıklarını paketleyen standart bir yazılım birimi olarak tanımlanabilir, böylece uygulama bir bilgisayar ortamından başka bir bilgisayar ortamına hızlı ve güvenli bir biçimde çalışabilmektedir. Docker bir konteyner görüntüsüdür ve bir uygulamayı çalıştırmak için gereken her şeyi içeren hafif, bağımsız ve çalıştırılabilir bir yazılım paketidir. Konteyner görüntüleri çalışma zamanı (runtime) sırasında konteyner haline gelmektedirler, Docker konteynerleri durumunda ise görüntüler Docker Engine'de çalışırlarken konteyner haline gelmektedirler. Konteynerler yazılımları ortamdan izole etmektedirler ve bu sayede farklılıklara rağmen düzgün bir şekilde çalışmaktadırlar örneğin, geliştirme ve görüntüleme arasındaki fark gibi düşünülebilmektedir (Docker, 2013).

3.1.5 PyWaveFront

PyWaveFront, Wavefront 3B nesneleri içeren dosyaları (“.obj”, “.mtl” gibi uzantılı dosyalar) okumak için kullanılmaktadır. Bu nesneler içerisinde yer alan yapay olarak oluşturulmuş olan vertex verilerini görselleştirme (rendering) için hazırlar (Pypi, 2019).

Nesneleri oluşturabilmek için (isteğe bağlı olarak) basit bir görselleştirme modülünü de bulunmaktadır (Pypi, 2019).

PyWaveFront’un en sık kullanılan özellikleri aşağıda belirtilmiştir:

- Konumlar

- Doku (texture) koordinatları
- Vertex renkleri
- Materyal ayrıştırma

PyWaveFront'un genel olarak kullanım örneği aşağıda Şekil 3.2 üzerinde gösterilmiştir (Pypi, 2019).

```
import pywavefront
scene = pywavefront.Wavefront('something.obj', strict=True, encoding="iso-8859-1", parse=False
scene.parse()  # Explicit call to parse() needed when parse=False

# Iterate vertex data collected in each material
for name, material in scene.materials.items():
    # Contains the vertex format (string) such as "T2F_N3F_V3F"
    # T2F, C3F, N3F and V3F may appear in this string
    material.vertex_format
    # Contains the vertex list of floats in the format described above
    material.vertices
    # Material properties
    material.diffuse
    material.ambient
    material.texture
    # ..
```

Şekil 3.2 PyWaveFront Kullanım Örneği (Pypi, 2019).

3.1.6 Pyglet

Pyglet, Windows,MAC OS x ve Linux'ta oyunlar ve görsel bakımdan zengin içerikli uygulamalar geliştirebilmek için kolay bir kullanım sağlayan güçlü bir Python kütüphanesidir. Pencereleme, kullanıcı arayüzü olay işleme, donanımsal kontrol cihazları, OpenGl grafik, resim ve video yükleme, ses ve müzik çalma gibi işlemleri desteklemektedir (Pyglet, 2018[a]).

Pyglet'in harici bağımlılıklar (dependencies) ya da kurulum gereksinimleri yoktur. Birçok uygulama ve oyun gereksinimi için, dağıtım ve kurulumu basitleştiren Python dışında başka bir şeye ihtiyaç duymamaktadır (Pyglet, 2018[a]).

Pyglet, çoklu platformlardan ve çoklu monitör masaüstlerinden faydalanılmasına olanak sağlayan gerçek platform pencereleri sağlamaktadır. Çoklu monitör kurulumlarının tam olarak farkındadır, uygulamaların veya oyunların nasıl görüntüleneceğinin kontrol edilebilmesini sağlamaktadır. Tam ekran ya da pencereli

olarak hangi monitörde görüntülendiklerini kontrol edilebilen birden fazla kayan pencere ya da tek pencereler oluşturulabilmektedir (Pyglet, 2018[a]).

Pyglet'in genel kullanım şekilleri aşağıda Şekil 3.3 ve Şekil 3.4 üzerinde gösterilmiştir (Pyglet, 2018[b]).

```
from pyglet import gl
# Create template config
config = gl.Config()
config.stencil_size = 8
config.aux_buffers = 4
# Create a window using this config
win = window.Window(config=config)
```

Şekil 3.3 Pyglet Genel Kullanım Örneği 1 (Pyglet, 2018[b]).

```
display = pyglet.canvas.get_display()
screens = display.get_screens()
windows = []
for screen in screens:
    windows.append(window.Window(fullscreen=True, screen=screen))
```

Şekil 3.4 Pyglet Genel Kullanım Örneği 2 (Pyglet, 2018[b]).

3.1.7 GTK

GTK, grafiksel kullanıcı arayüzleri oluşturmak için kullanılan çoklu platform desteği olan bir araçtır. GTK, eksiksiz bir widget seti sunmaktadır ve küçük, bir kerelik araçlardan geniş kapsamlı uygulamalara kadar olan birçok proje için uygundur (GTK, 2007).

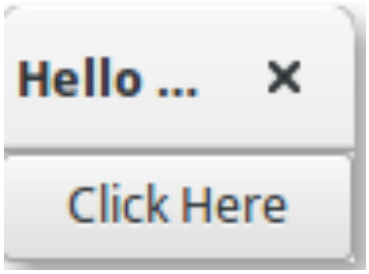
GTK, çapraz-platform olarak çalışabilmektedir ve bu sayede uygulama geliştirmeyi hızlandırıp ve kolaylaştırmaktadır (GTK, 2007).

GTK, C dilinde yazılmıştır, fakat sadece C/C++ dilini değil geniş bir dil yelpazesini destekleyebilmek için sıfırdan tasarlanmıştır. GTK'yı Perl ve Python gibi dillerle kullanmak, (özellikle kullanıcı arayüzü oluşturmak için) hızlı ve kolay bir uygulama geliştirmek için etkili olmaktadır (GTK, 2007).

Aşağıda Şekil 3.5 ile GTK kod örneği ve Şekil 3.6 ile Şekil 3.5'deki kodların sonuçları gösterilmiştir (PythonGTKTutorial, 2007).

```
1  import gi
2  gi.require_version('Gtk', '3.0')
3  from gi.repository import Gtk
4
5  class MyWindow(Gtk.Window):
6
7      def __init__(self):
8          Gtk.Window.__init__(self, title="Hello World")
9
10         self.button = Gtk.Button(label="Click Here")
11         self.button.connect("clicked", self.on_button_clicked)
12         self.add(self.button)
13
14     def on_button_clicked(self, widget):
15         print("Hello World")
16
17 win = MyWindow()
18 win.connect("destroy", Gtk.main_quit)
19 win.show_all()
20 Gtk.main()
```

Şekil 3.**5** GTK Kod Örneği (PythonGTKTutorial, 2007).



Şekil 3.6 Şekil 3.5'de yer alan kodların ekran çıktısı (PythonGTKTutorial, 2007).

3.2 Yöntem ve Uygulama

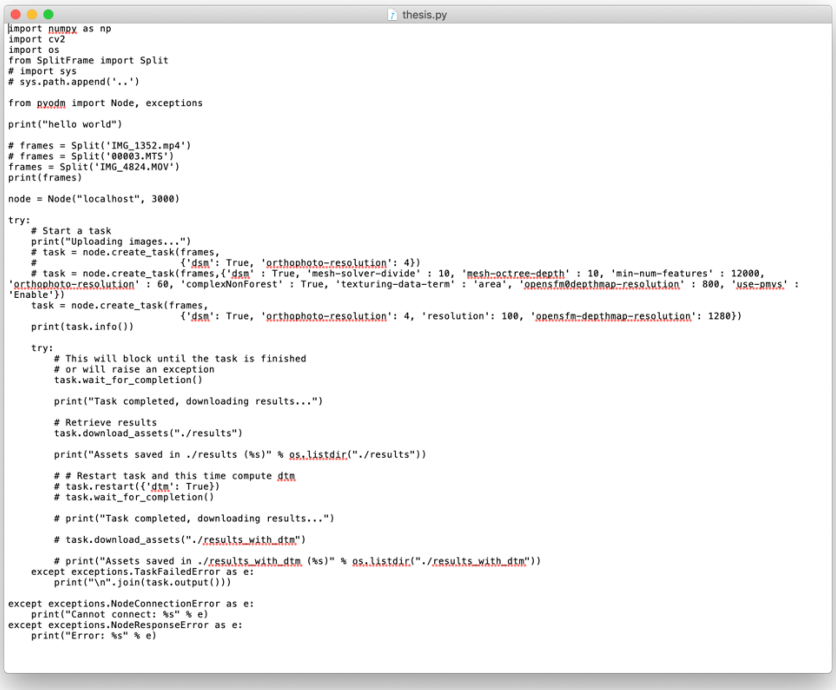
Bu tez çalışmasında, geliştirilen uygulama için farklı cihazlarla elde edilen çeşitli görüntüler ve videolar kullanılmıştır. Elde edilen videoların OpenDroneMap ile işlenebilmesi için videoların öncelikle resim veya frame haline dönüştürülmesi gerekmektedir. Bu işlem için Şekil 3.7'de görüldüğü üzere Python dili kullanılarak bir metot oluşturulmuştur. Bu metot içerisinde OpenCV ile video dosyası alınıp videonun uzunluğuna göre uygun değerler ile resimlere bölünmüştür. Bölme işlemi için kullanılan en sık değer olarak videonun kendi FPS (Frame Per Second) değeri olmuştur. Bu sayede video kaç saniye ise o kadar resim elde edilmiştir.

SplitFrame.py

```
import numpy as np
import cv2
import os
def Split(FileName):
        cap = cv2.VideoCapture(FileName)
        lengthOfCap = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
        width = cap.get(3)
        height = cap.get(4)
        fps    = int(cap.get(cv2.CAP_PROP_FPS))
        # fps = fps * 3
        fps = fps / 3
        print(str(width) + '-' + str(height))

        try:
                if not os.path.exists('Frames'):
                        os.makedirs('Frames')
        except OSERROR:
                print('Error: Creating directory of Frames')

        currentFrame = 0
        currentName = 1
        frames = []
        while (True):
                ret, frame = cap.read()



                currentFrame += 1
                if currentFrame == lengthOfCap:
                        break
```

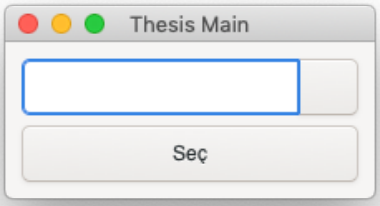
Şekil 3.7 SplitFrame Metodu.

Elde edilen resimler daha sonra OpenDroneMap aracılığı ile Şekil 3.8'deki gibi işlenmiştir. OpenDroneMap API'sinin çalışabilmesi için bir konteyner gerekmektedir. Gerekli olan konteyner ise Docker ile elde edilmiştir ve kullanılmıştır. İşlenme sırasında OpenDroneMap tarafından her bir resim tek tek ve birbirleriyle karşılıklı olarak incelenerek ortak noktalar ve ortak pikseller bulunmuştur. Bu ortak noktalar ve pikseller daha sonra yine OpenDroneMap aracılığı ile birleştirilerek üç-boyutlu (3B) model elde edilmiştir. 3B model hem .ply hem de .obj formatında olmak üzere oluşturulmuştur. Bu modellerden .ply ile poligonal bir model elde edilmiştir, .obj ile ise 3B grafiklerde kullanılan bir model elde edilmiştir.



Şekil 3.8 Resimlerin OpenDroneMap ile İşlenmesi.

Elde edilen bu modellerin görüntülenmesi için yine Python dili kullanılmıştır. Görüntüleme işlemi için OpenDroneMap aracılığı ile elde edilen .obj uzantılı vertex tabanlı objeler kullanılmıştır. Elde edilen bu objelerin görüntülenebilmesini sağlamak için öncelikle Şekil 3.9'da gösterilen seçim ekranı oluşturulmuştur. Bu seçim ekranı oluşturulurken gerekli olan kullanıcı arayüzü nesnelerinin oluşturulabilmesi için Bölüm 3.1.7'de bahsedilen GTK aracı kullanılmıştır. Bu arayüz içerisinde yer alan seçim alanı ile daha önce OpenDroneMap ile elde edilen .obj uzantılı objeler gösterilmiştir. Bu seçim alanı ile istenilen obje seçilmektedir. Arka planda ise bu seçilen objenin görüntülenebilmesi için gerekli olan dosya yolu elde edilmektedir. Elde edilen bu dosya yolu, arayüz içerisinde yer alan buton ile görüntülemenin yapılacak olduğu bir diğer Python kod dosyasını çalıştırmaktadır. Dosya yolunun elde edilmesini sağlayan kodlar aşağıda Şekil 3.10 ve Şekil 3.11 ile gösterilmiştir.

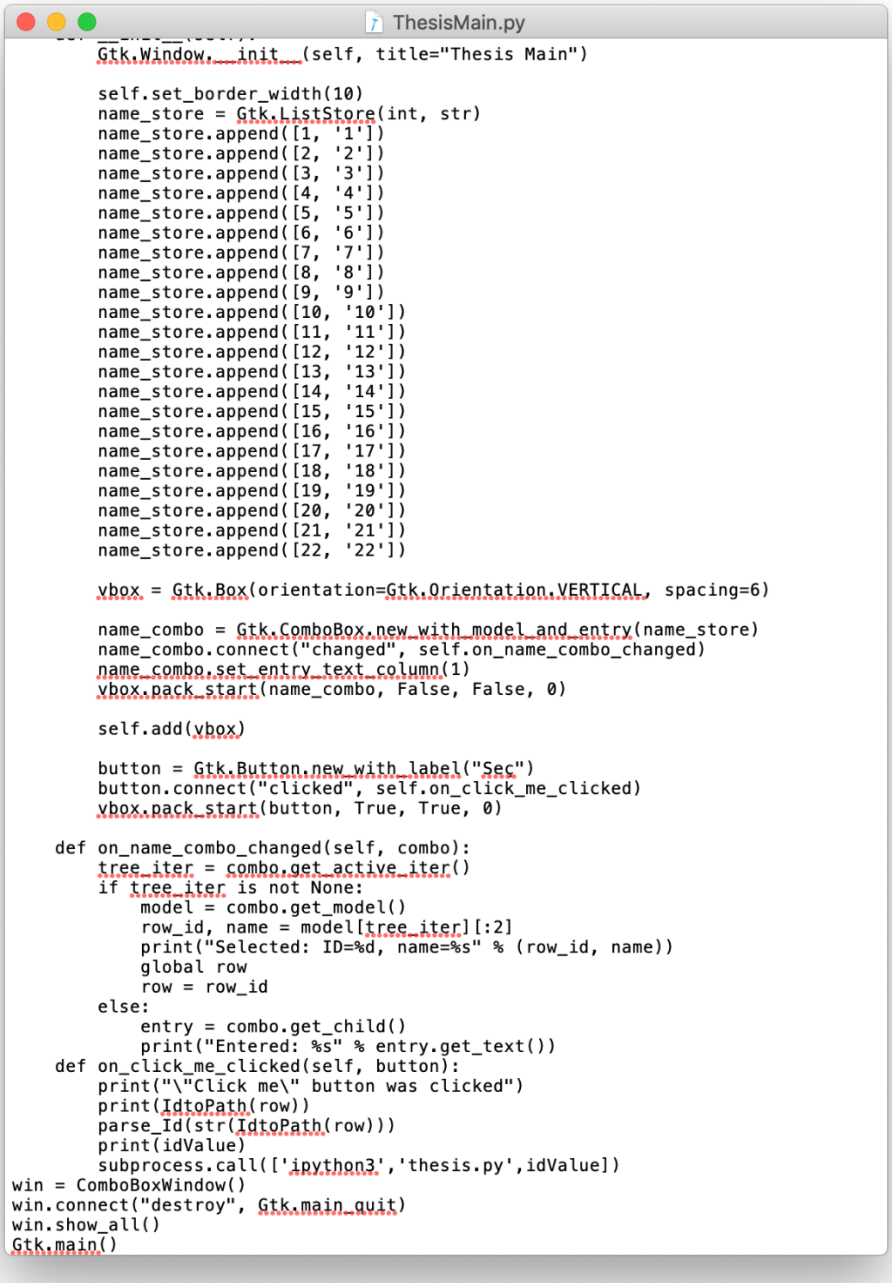


Şekil 3.9 Dosya Seçim Arayüzü.

ThesisMain.py

```
import gi
gi.require_version('Gtk', '3.0')
from gi.repository import Gtk
from enum import Enum

import subprocess

idValue = ""
row = ""
def parse_Id(value):
        lenVal = len(value)
        global idValue
        counter = 0
        returnVal = ""
        while counter<lenVal:
            if(value[counter]=='.'):
                break
            else:
                counter += 1

        counter += 1

        while counter<lenVal:
            returnVal += value[counter]
            counter += 1
        idValue = returnVal

class IdtoPath(Enum):
    results1 = 1
    results2 = 2
    results3 = 3
    results4 = 4
    results5 = 5
    results6 = 6
    results7 = 7
    results8 = 8
    results9 = 9
    results10 = 10
    results11 = 11
    results12 = 12
    results13 = 13
    results14 = 14
    results15 = 15
    results16 = 16
    results17 = 17
    results18 = 18
    results19 = 19
    results20 = 20
    results21 = 21
    results22 = 22

class ComboBoxWindow(Gtk.Window):

    def __init__(self):
        Gtk.Window.__init__(self, title="Thesis Main")

        self.set_border_width(10)
        name_store = Gtk.ListStore(int, str)
        name_store.append([1, '1'])
        name_store.append([2, '2'])
```

Şekil 3.10 Dosya Seçim Arayüzü Kodları 1.



Şekil 3.11 Dosya Seçim Ekranı Kodları 2.

Görüntüleme için OpenDroneMap ile elde edilen .obj uzantılı vertex tabanlı objelerin ve bu objelerin GTK ile oluşturulmuş olan arayüz ile elde edilen dosya yollarının görüntülenebilmesi için gerekli olan işlemler python kodları ile oluşturulmuştur. Bu işlem için öncelikle .obj dosyalarının okunması gerekmektedir. Bu okuma işlemi Bölüm 3.1.5'te anlatılan PyWaveFront Kütüphanesi ile sağlanmıştır. PyWaveFront kütüphanesi ile obje dosyadan okunmuş ve

görüntülenmesi için gerekli olan konum ve vertex bilgileri ile doku (texture) verileri elde edilmiştir. Elde edilen bu verileri kullanarak görüntüleme işleminin sağlanması için Bölüm 3.1.6'da anlatılan Pyglet kütüphanesi kullanılmıştır. Pyglet kütüphanesi ile öncelikle bir pencere oluşturulmuştur ve bu pencere Şekil 3.12 içerisinde yer alan "on_resize" metodu ile boyutlanmıştır. Objenin arayüz içerisinde hareket edebilmesini sağlamak için gerekli global değişkenler tanımlanmıştır. Bu global değişiklikler kullanılarak arayüze devamlı olarak görüntünün çizilmesi Şekil 3.12'de yer alan "on_draw" metodu ile Pyglet aracılığı ile sağlanmıştır. Şekil 3.12'de yer alan "on_draw" metodu ile objenin görüntülenmesini sağlayacak olan sahne ışığı gibi veriler oluşturulmuştur ve yine Şekil 3.12'de yer alan "draw_box" metodu ile görüntüleme için gereken açı ve yön bilgileri global değişkenleri kullanarak belirtilmiş ve Pyglet'in ekrana çizim metodu olan "visualization.draw" metodu ile elde edilen veriler aracılığı ile görüntüleme sağlanmıştır.

Elde edilen görüntü üzerinde hareketin sağlanması için klavye üzerinden giriş gerekmektedir. Bu girişler:

- Q : Saat Yönünün Tersine Dönme (Z ekseni)
- E : Saat Yönünde Dönme (Z ekseni)
- W : İleri Hareket
- S : Geri Hareket
- A : Sol Yönde Hareket
- D : Sağ Yönde Hareket
- R : Yukarı Yönde Hareket
- F : Aşağı Yönde Hareket
- Sol Ok Tuşu : Sola Dönme (X ekseni)
- Sağ Ok Tuşu : Sağa Dönme (X Ekseni)
- Yukarı Ok Tuşu : Yukarı Dönme (Y ekseni)
- Aşağı Ok Tuşu : Aşağı Dönme (Y Ekseni)
- Z : Dönme Hareketlerinin Hızını Arttırma
- X : Dönme Hareketlerinin Hızını Azaltma
- C : Dönme Hareketlerinin Hızını Sıfırlama

Bu girişlerin kontrolleri yine Şekil 3.14'de yer almaktadır. Bu işlem sırasında klavyeden basılan tuşun hangisi olduğu tespit edilmekte ve basılan tuşa göre gerekli olan işlemler global değişkenler aracılığı Şekil 3.13'de yer alan kodlar aracılığı ile sağlanmaktadır.

thesis.py

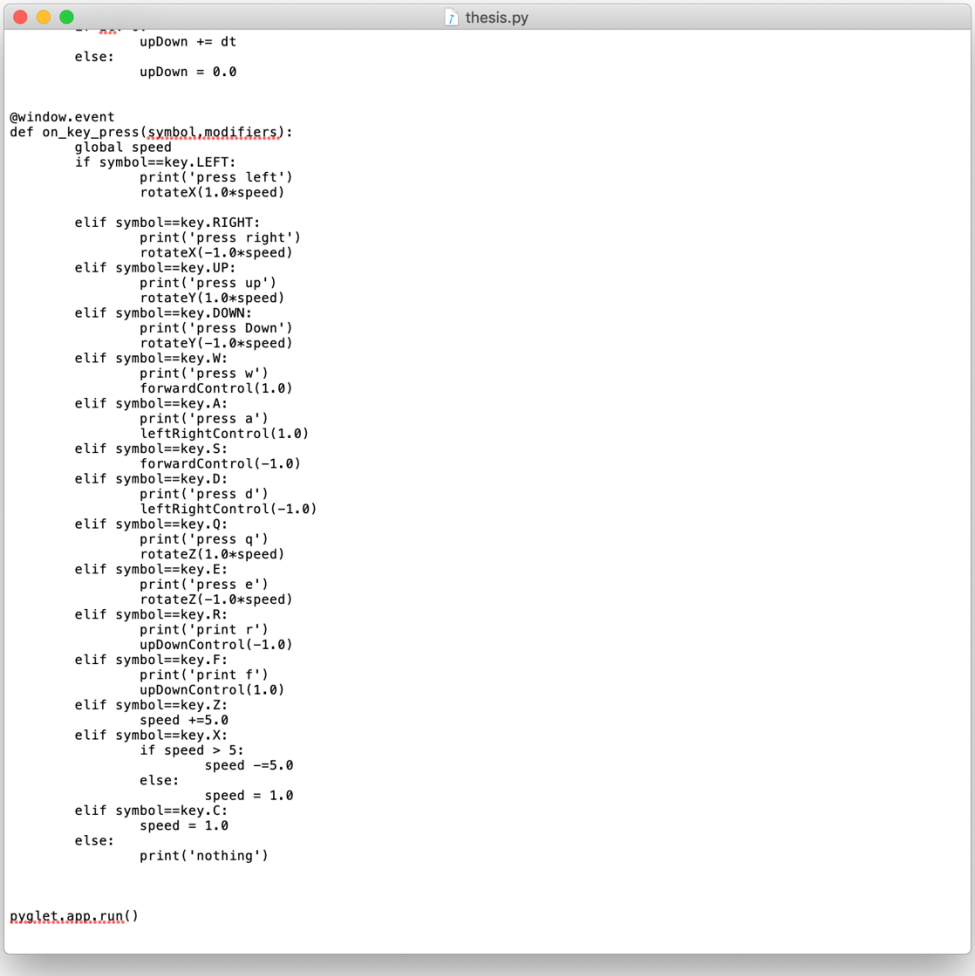
```
import ctypes
import os
from pyglet.gl import *
from pywavefront import visualization, Wavefront

import sys

fileId = sys.argv[1]

print(fileId)

window = pyglet.window.Window(width=2560, height=1600, resizable=True)

root_path = os.path.dirname(__file__)

box1 = Wavefront(os.path.join(root_path, fileId+'/odm_texturing/odm_textured_model.obj'))

key = pyglet.window.key

speed = 1.0
rotationAxisX = 0.0
rotationAxisY = 0.0
rotationAxisZ = 0.0
forward = 0.0
leftRight = 0.0
upDown = 0.0

lightfv = ctypes.c_float * 4

@window.event
def on_resize(width, height):
    viewport_width, viewport_height = window.get_framebuffer_size()
    glViewport(0, 0, viewport_width, viewport_height)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(45., float(width)/height, 1., 100.)
    glMatrixMode(GL_MODELVIEW)
    return True

@window.event
def on_draw():
    window.clear()
    glLoadIdentity()

    glLightfv(GL_LIGHT0, GL_POSITION, lightfv(-1.0, 1.0, 1.0, 0.0))

    draw_box(box1)


def draw_box(box):
    glLoadIdentity()
    glTranslated(leftRight,upDown,forward)
    glRotatef(rotationAxisX, 0.0, 1.0, 0.0)
    glRotatef(rotationAxisY, 1.0, 0.0, 0.0)
    glRotatef(rotationAxisZ, 0.0, 0.0, 1.0)


    visualization.draw(box)
```

Şekil 3.12 Görüntüleme Kodları 1.

thesis.py

```
    glRotatef(rotationAxisX, 0.0, 1.0, 0.0)
    glRotatef(rotationAxisY, 1.0, 0.0, 0.0)
    glRotatef(rotationAxisZ, 0.0, 0.0, 1.0)


    visualization.draw(box)


def rotateX(dt):

        global rotationAxisX
        if dt!=0:
                rotationAxisX += dt
        else:
                rotationAxisX = 0.0

def rotateY(dt):

        global rotationAxisY
        if dt!=0:
                rotationAxisY += dt
        else:
                rotationAxisY = 0.0
def rotateZ(dt):
        global rotationAxisZ
        if dt!=0:
                rotationAxisZ += dt
        else:
                rotationAxisZ = 0.0

def forwardControl(dt):
        global forward
        if dt!=0:
                forward += dt
        else:
                forward = 0.0

def leftRightControl(dt):
        global leftRight
        if dt!=0:
                leftRight += dt
        else:
                leftRight = 0.0

def upDownControl(dt):
        global upDown
        if dt!=0:
                upDown += dt
        else:
                upDown = 0.0


@window.event
def on_key_press(symbol,modifiers):
        global speed
        if symbol==key.LEFT:
                print('press left')
                rotateX(1.0*speed)

        elif symbol==key.RIGHT:
                print('press right')
                rotateY(-1.0*speed)
```

Şekil 3.13 Görüntüleme Kodları 2.



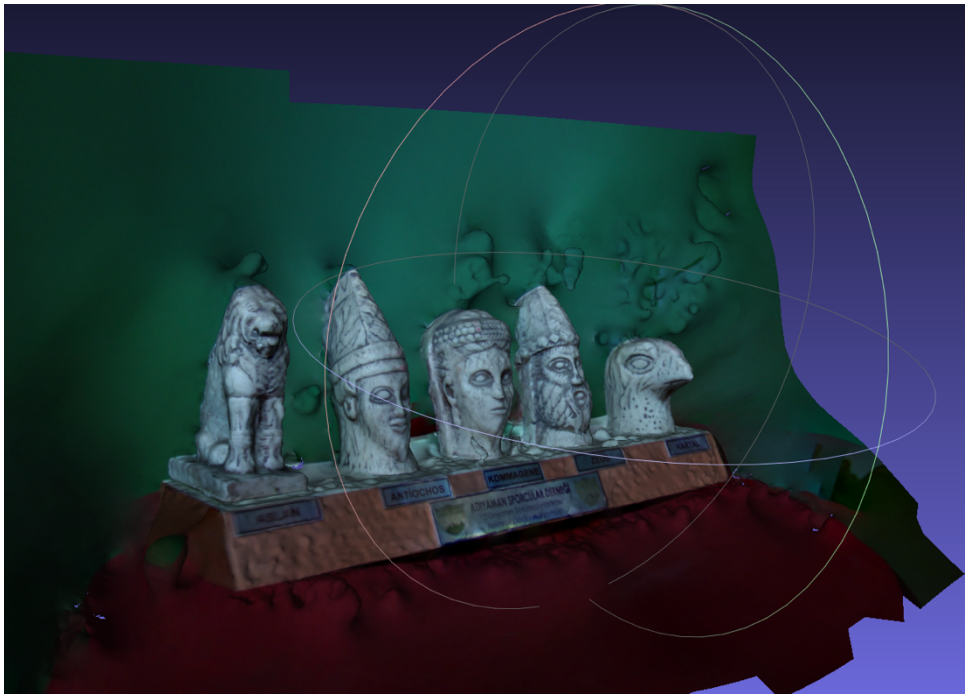
Şekil 3.14 Görüntüleme Kodları 3.

4. UYGULAMA SONUÇLARI

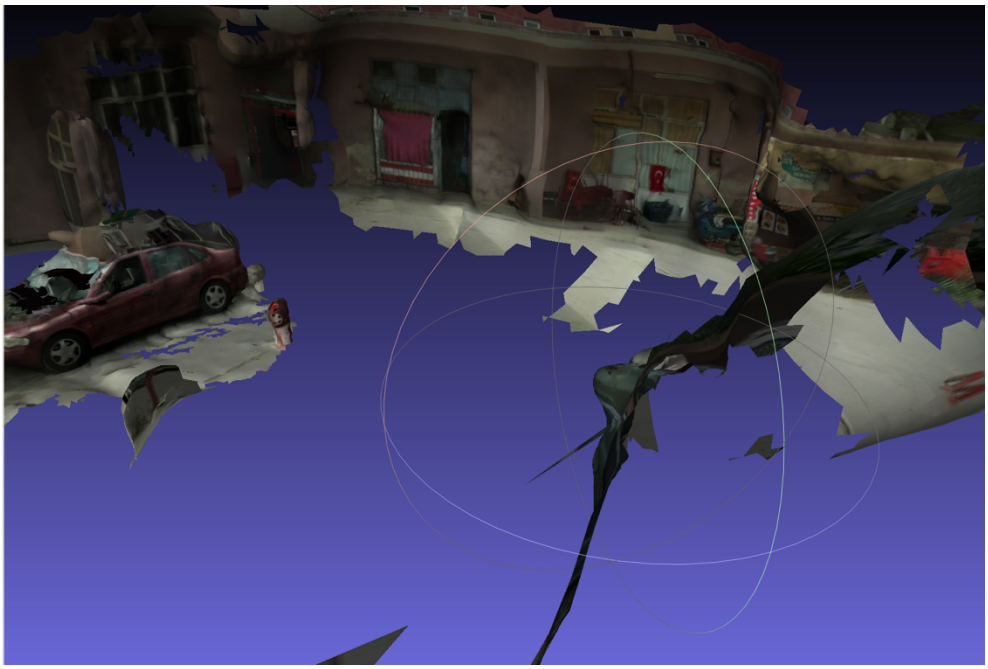
Bu bölümde, hazırlanan uygulama ile elde edilmek istenen nesnenin üç boyutlu haritası için kullanılacak üç-boyutlu modeller elde edilmiştir. Bu işlemler sırasında docker üzerinde çalışan OpenDroneMap kullanılmış ve bu kullanımı sağlamak için dil kolaylığı ve okunabilirliği açısından Python platformu tercih edilmiştir. OpenDroneMap ile elde edilen modeller, eğer konum verileri içeriyorlar ise yardımcı programlar aracılığı ile Google Maps gibi online haritalar üzerine yerleştirilebilmektedir. Elde edilen modellerin sonuçlarının test edilebilmesi için modeller MeshLab isimli bir görüntüleme uygulaması üzerinde gösterilmiştir. Test için elde edilen bazı modeller aşağıda Şekil 4.1 ile Şekil 4.6 arasında resimlendirilmiştir.

Şekil 4.1 MeshLab model örnek görüntüsü-1.

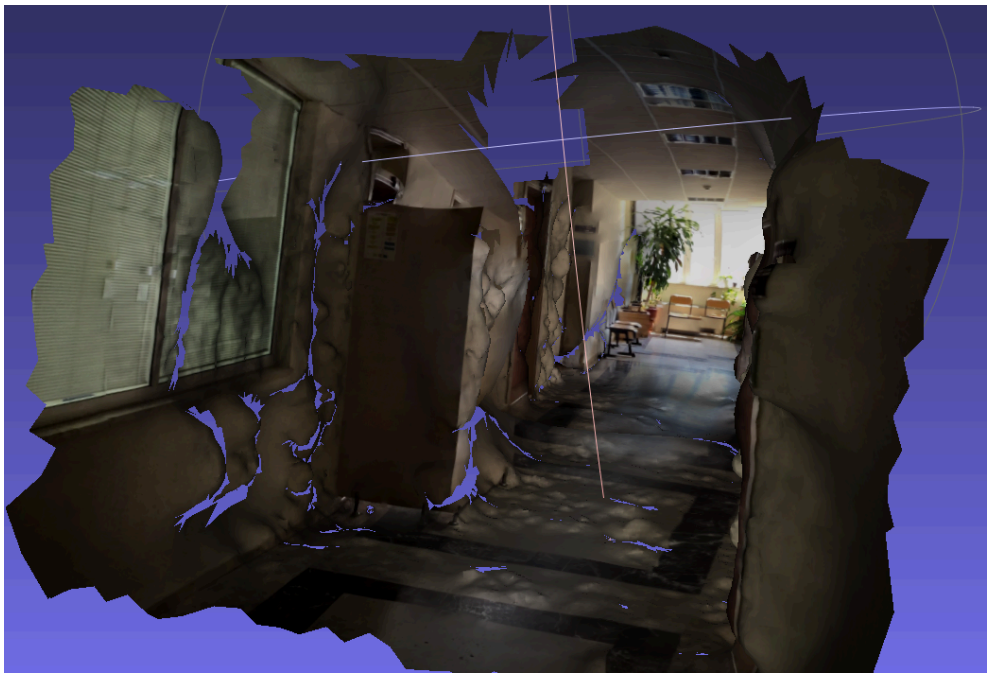
Şekil 4.2 MeshLab model örnek görüntüsü-2.



Şekil 4.3 MeshLab model örnek görüntüsü-3.



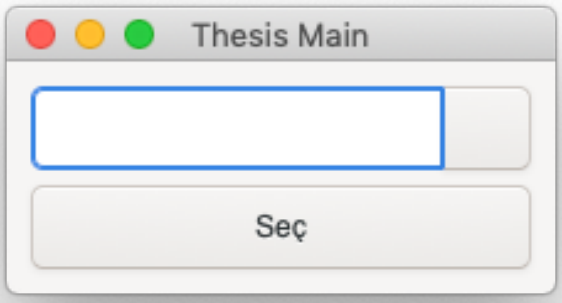
Şekil 4.4 MeshLab model örnek görüntüsü-4.



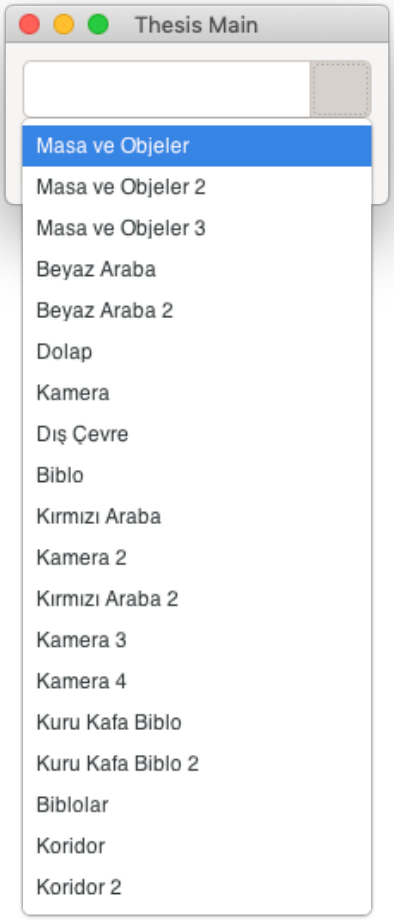
Şekil 4.5 MeshLab model örnek görüntüsü-5.

Şekil 4.6 MeshLab model örnek görüntüsü-6.

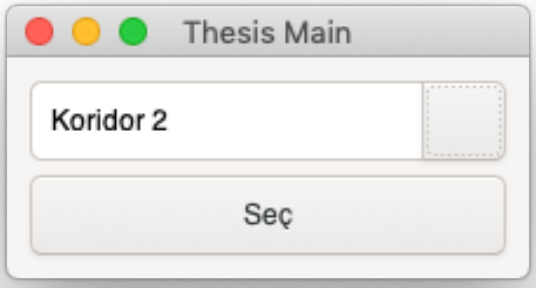
Elde edilen modeller test edildikten sonra haritanın görüntülenebilmesi için gerekli arayüzler elde edilmiştir. Bu arayüzlerden bir tanesi ile elde edilen modeller arasından görüntülenmek istenen modelin seçilmesi sağlanmıştır. Bu işlem için GTK kullanılmıştır. Daha sonra bu arayüz ile elde edilen veriler ile haritanın görüntüleneceği arayüz hazırlanmıştır. Bu arayüz içerisinde harita üzerinde klavye hareket etme işlemleri sağlanmıştır. Bu arayüz oluşturulurken GTK ile elde edilen veriden gelen modele ulaşmak için PyWaveFront kullanılmıştır. Ulaşılan modelin görüntülenmesi için ise Pyglet kullanılmıştır. İşlemlerin sonucunda 3-Boyutlu olarak elde edilen ve üzerinde klavye tuşları dolaşılabilen bir harita elde edilmiştir. Model seçim ve görüntüleme arayüzleri ve harita örnekleri aşağıda Şekil 4.7 ve Şekil 4.18 arasında gösterilmiştir.



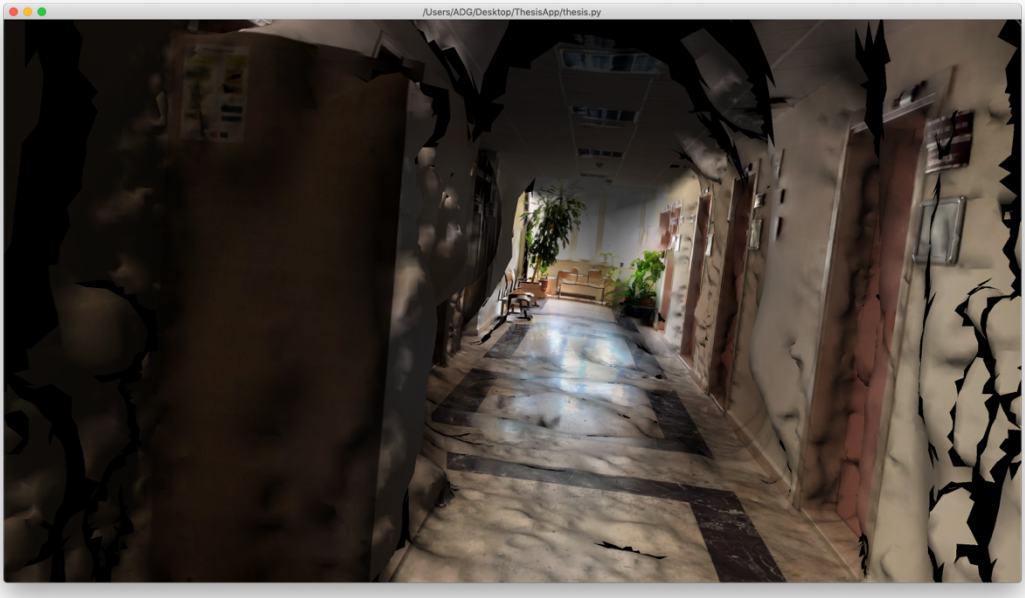
Şekil 4.7 Seçim Ekranı 1.



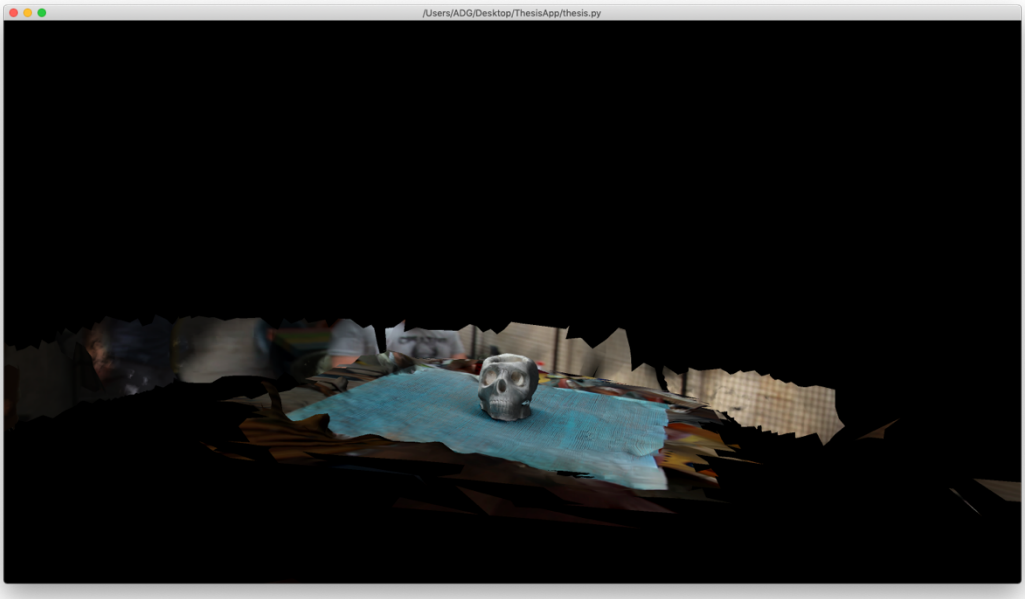
Şekil 4.8 Seçim Ekranı 2.



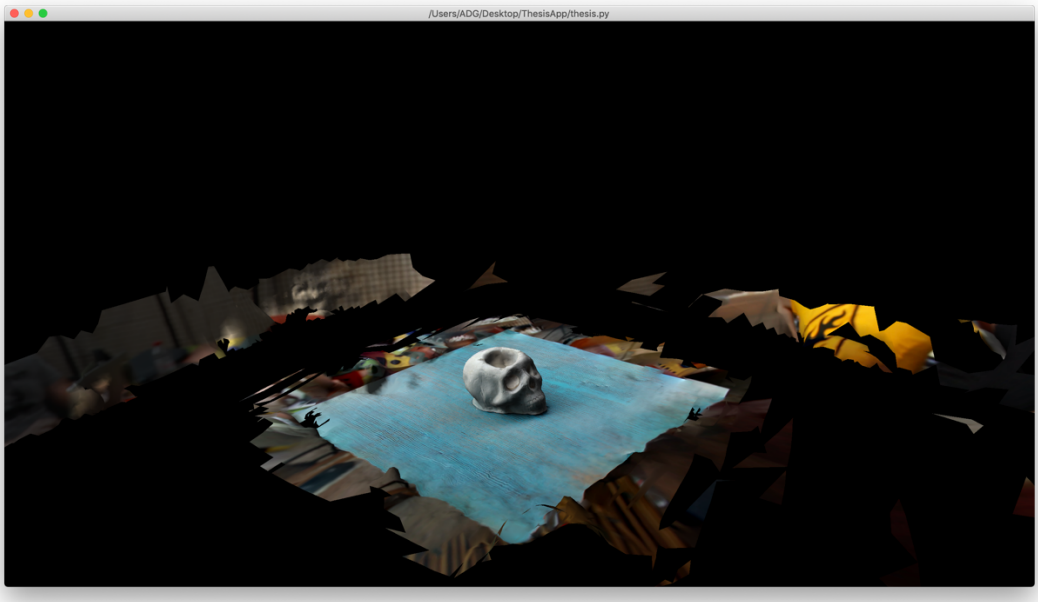
Şekil 4.9 Seçim Ekranı 3.



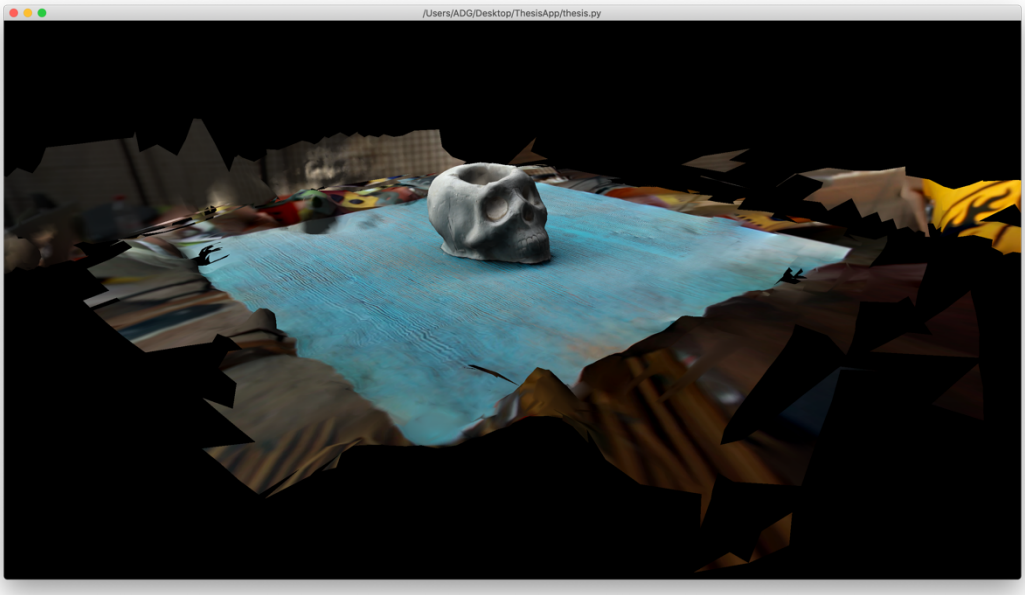
Şekil 4.10 Görüntüleme Ekranı 1.



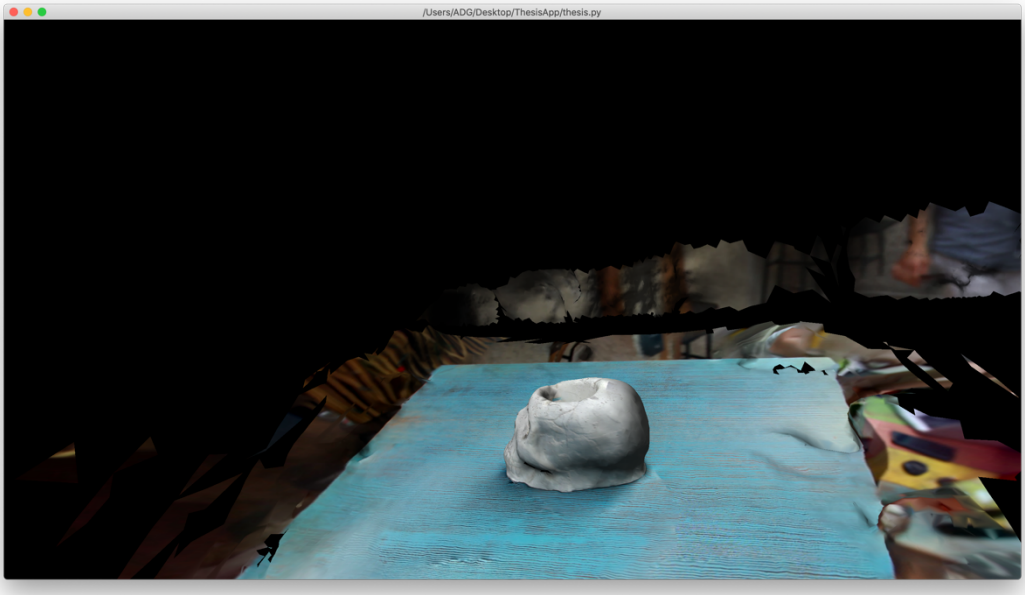
Şekil 4.11 Görüntüleme Ekranı 2.



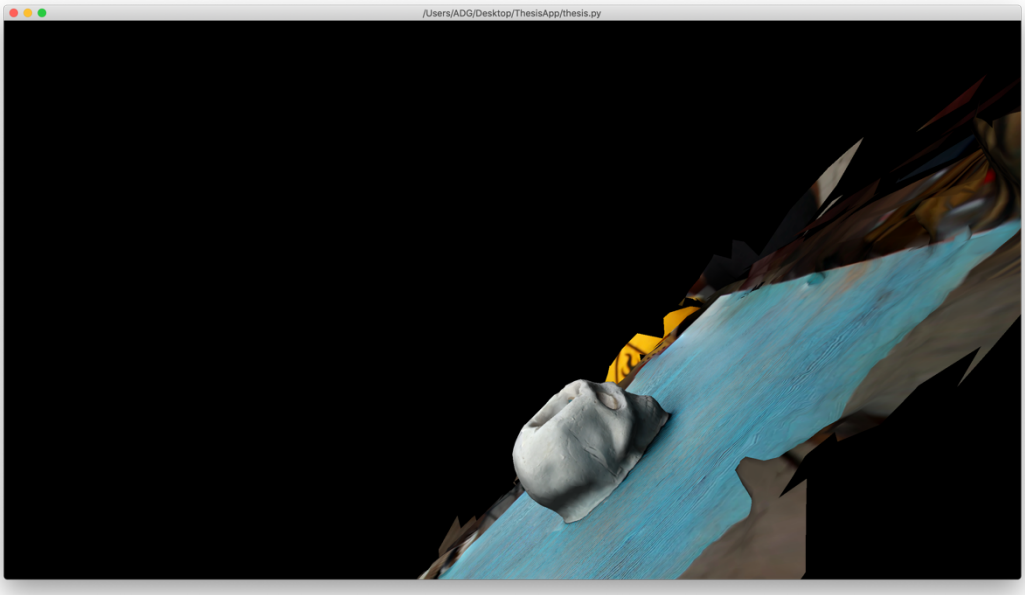
Şekil 4.12 Görüntüleme Ekranı 3.



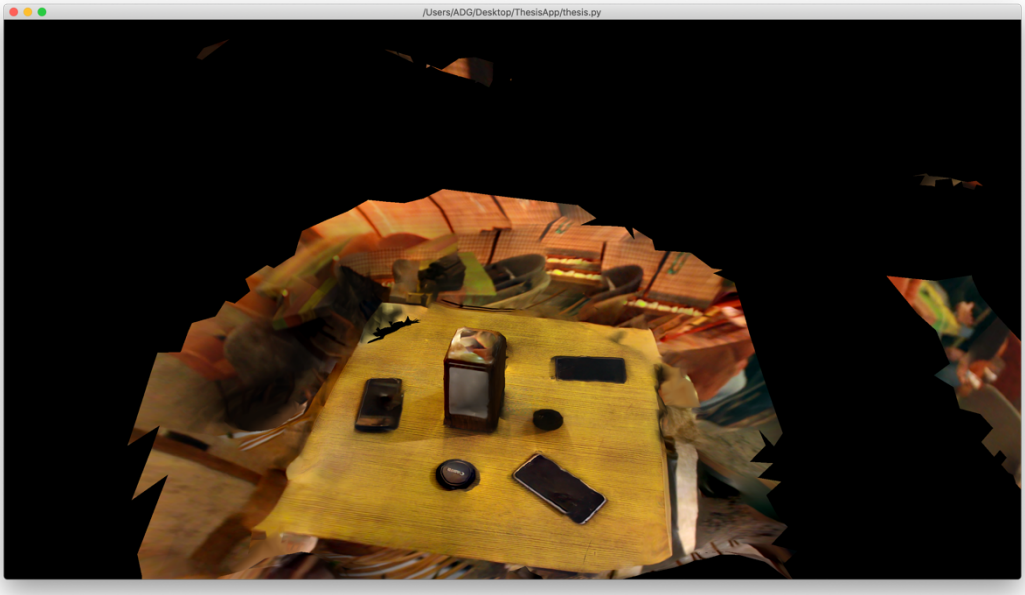
Şekil 4.13 Görüntüleme Ekranı 4.



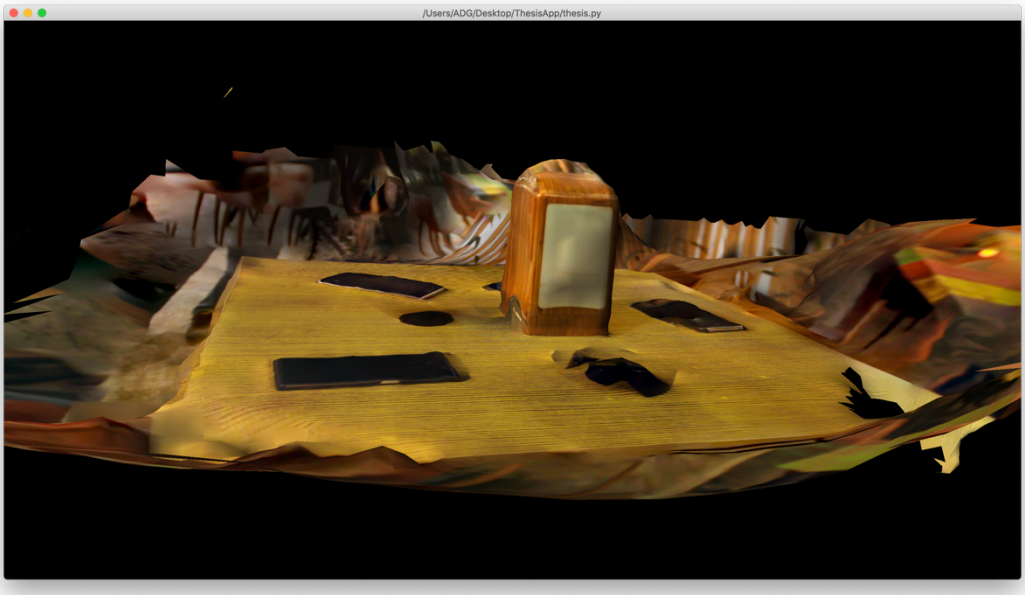
Şekil 4.14 Görüntüleme Ekranı 5.



Şekil 4.15 Görüntüleme Ekranı 6.



Şekil 4.16 Görüntüleme Ekranı 7.



Şekil 4.17 Görüntüleme Ekranı 8.

Şekil 4.18 Görüntüleme Ekranı 9.

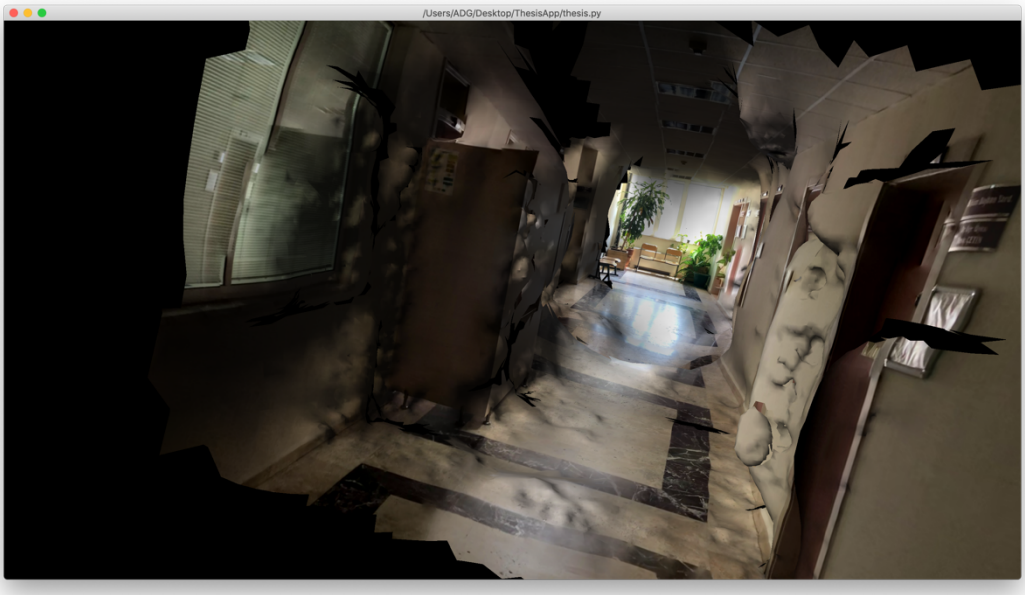
Bu çalışmada daha önce SplitFrame metodu kullanılarak elde edilen resimlerin (frame) sayısının FPS (Frame Per Second) değerine göre belirlendiği anlatılmıştır. Çalışma sırasında bu FPS değeri üzerinde oynamalar yaparak ve farklı değerler kullanılarak modellerin elde edilme süreleri ve elde edilen modellerin detayları arasındaki farklar belirlenmiştir. Bu farklar Tablo 4.1 de gösterilmiştir.

Farklı FPS değerlerine göre elde edilen haritalar ise Şekil 4.19 ve Şekil 4.23 arasında gösterilmiştir.

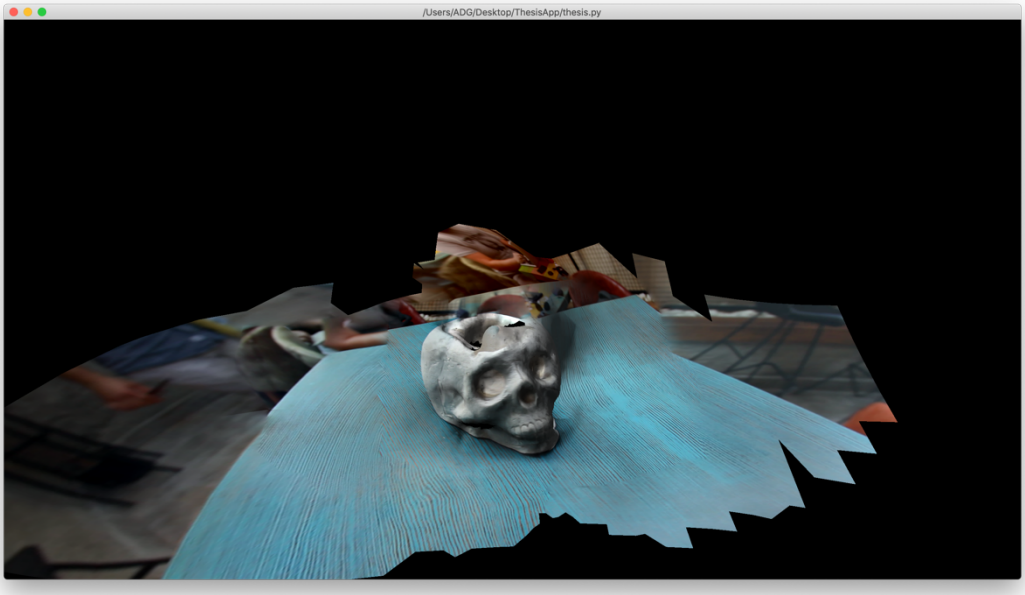
Tablo 4.1 2 Videonun Farklı Resim Sayıları ile Elde Edilen Süreler.

Harita ismi	Resim (Frame) Sayısı	Resim Bölme Süresi	Haritalama Süresi	Video Süresi
A1	39	7.489 sn	27:21.457 dk	20 sn
A2	9	7.317 sn	3:13.332 dk	20 sn
B1	9	4.901 sn	2:49.281 dk	19 sn
B2	19	4.558 sn	11:28.884 dk	19 sn
B3	39	7.727 sn	66:22.775 dk	19 sn

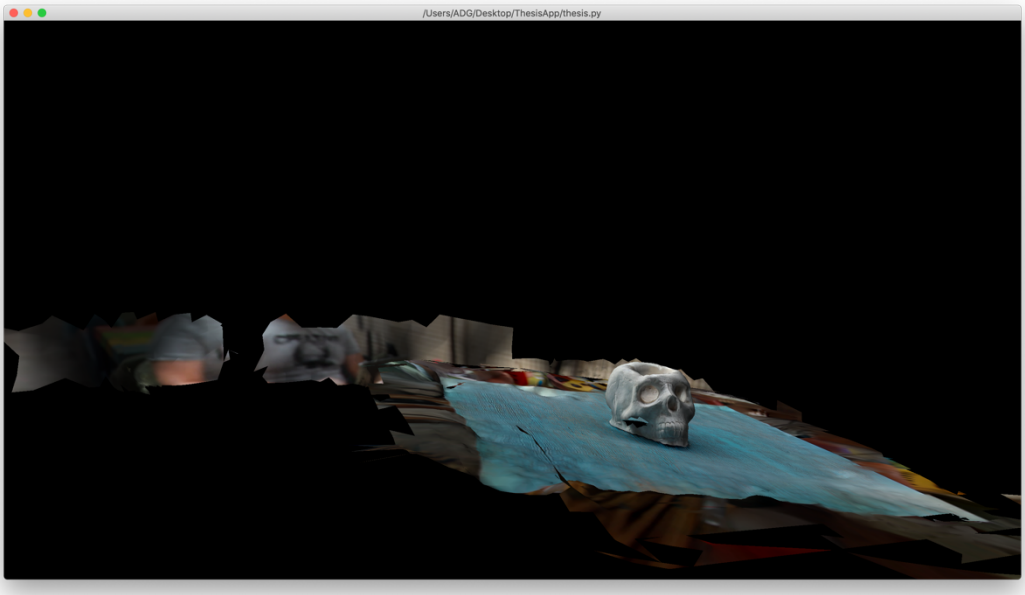
Şekil 4.19 A1 Görüntüsü.



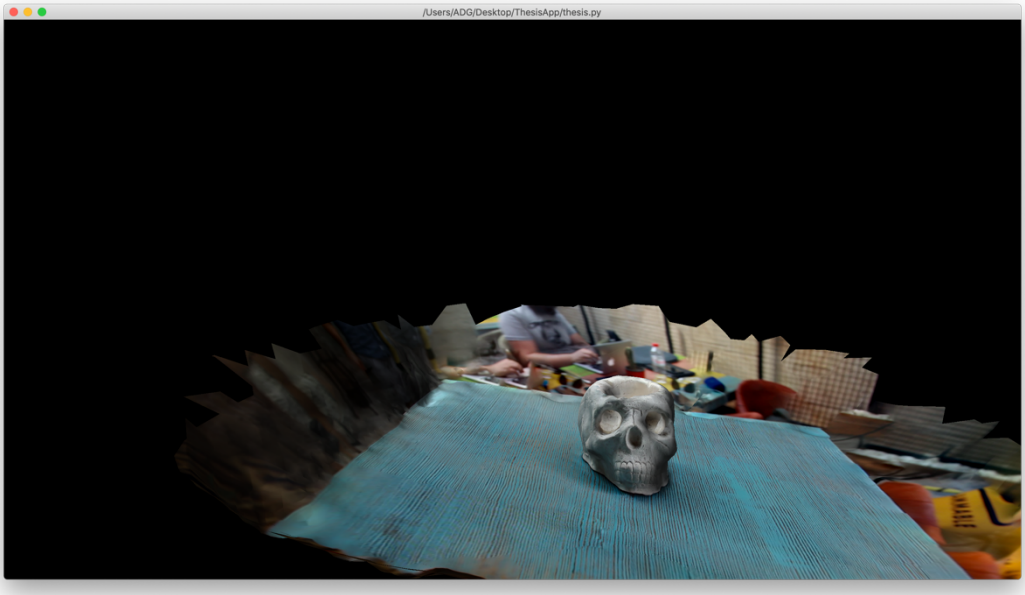
Şekil 4.20 A2 Görüntüsü.



Şekil 4.21 B1 Görüntüsü.



Şekil 4.22 B2 Görüntüsü.



Şekil 4.23 B3 Görüntüsü.

5. SONUÇ VE TARTIŞMA

Globalleşmekte ve gelişmekte olan dünya üzerinde yaşamakta olan insanların bu globalleşme ve gelişime ayak uydurabilmeleri önemlidir. Teknolojinin de gelişimi ile globalleşen bu dünyada insanların imkanları olsun veya olmasın, bu yeni düzene ayak uydurabilmeleri için teknolojinin yardımı gerekmektedir. Bu çalışma ile insanların globalleşen bu dünya üzerinde gezmeye ve görmeye ihtiyaç duydukları yerleri, bulundukları yerden ayrılmadan gezip görmelerine imkan sağlanması amacı üzerine çalışılmış ve bunun sağlanması için 3-Boyutlu haritalar oluşturulmuştur.

Yapılan çalışma ile insanların görmek istedikleri yerleri 3-Boyutlu olarak görmeleri veya imkanı olmayan insanlara gösterebilmek için istedikleri yerlerin 3-Boyutlu haritalarını oluşturabilmeleri sağlanmıştır.

İki aşamadan oluşan bu projenin ilk aşaması 3-Boyutlu haritalamayı bazı kütüphane ve araçlar yardımı ile sağlamakta ve diğer aşamada ise oluşturulan bu haritaların 3-Boyutlu olarak görüntülenebilmesi başarılmıştır.

İlk aşama sırasında haritalama sağlanırken kullanılan farklı resim sayıları ile sonuçlar elde edilirken resim sayısı arttıkça elde edilen harita üzerindeki detayların arttığı ama sonuçlar elde edilirken geçen sürenin ise arttığı görülmüştür. Sonuç olarak ne kadar çok resim kullanılırsa o kadar iyi sonuç alınmasına karşın işlem süresinin arttığı belirlenmiştir.

Bu alanda var olan birçok karmaşık proje bulunmaktadır. Hazırlanan bu çalışma ile olabildiğince basit ve kullanıcı dostu bir sistem hazırlanmaya çalışılmıştır. Birçok ücretli ve pahalı uygulamaların aksine bu çalışmanın minimum maliyetle çalışması sağlanmıştır.

Hazırlanan bu proje açık kaynaklı olmasının da etkisiyle sanal gerçeklik uygulamaları gibi uygulamalara kolayca entegre edilebilecektir.

6. KAYNAKLAR

Amini, J., Lucas, C., Saradjian, M. R., Azizi, A., Sadeghian, S. "Fuzzy Logic System For Road Identification Using Ikonos Images". *Photogrammetric Record*, (2002).

Blaschke T., "Object Based Image Analysis for Remote Sensing", *International Society for Photogrammetry and Remote Sensing Journal of Photogrammetry and Remote Sensing*, *vol.65(1), pp. 2-16,* (2010)

Bobillet W., Da Costa J. P., Germain C., Lavialle O. and Grenier G., "Row Detection in High Resolution Remote Sensing Images of Vine Fields". *In: Papers from the 4th European Conference on Precision Agriculture*, *15-19 June 2003, Berlin, Germany, pp. 81-87,* (2003)

Castilla G. and Hay G. J., Image Objects and Geographic Objects. In Blaschke T., Lang S. and Geoffrey J. H. (Eds.), "Object-based Image Analysis", *Berlin, Heidelberg, pp. 91–110,* (2008)

Docker, "What is Container [online]" (22 Aralık 2018), https://www.docker.com /resources/what-container/, 2013

Ersan R. "Gül tarım alanlarının yüksek çözünürlüklü uydu verileri ile belirlenebilirliği". Yüksek lisans tezi, *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü* (Basılmamış), Isparta, (2013)

Foley J. D., van Dam A., Feiner S. K. and Hughes J. F.*, Computer Graphics: Principles and Practise Second Edition*, Addison-Wesley Publishing Company, (1990)

Géraud, T. and Mouret, J.B., "Fast road network extraction in satellite images using mathematical morphology and Markov random fields". *EURASIP Journal of Applied Signal Processing*, vol.16, pp. 2503-2514., (2004)

Gonzalez R. C. and Woods R. E., *Digital Image Processing Third Edition*, Pearson International Edition, (2008)

GTK, "What is GTK, and how can I use it? [online]", (3 Eylül 2019), https://www.gtk.org/, 2007

Hu, J., Razdan, A., Femiani, j., Wonka, P., and Cui M., "Fourier Shape Descriptors of Pixel Footprints for Road Extraction from Satellite Images". *ICIP*, (2007)

Huber D., Kapuria A., Donamukkala R. and Hebert M., "Part-Based 3D Object Classification", *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, (2004)

Joshi P., *Artificial Intelligence with Python*, Packt Publishing, (2017)

Lee, H. Y., Park, W., and Lee, H. K., "Automatic Road Extraction from 1M-Resolution Satellite Images." *ISPRS Journal of Photogrammetry and Remote Sensing, vol. 55,* (2000).

Liu, H., Li, J. and Chapman, M. A. "Automated Road Extraction from Satellite Imagery Using Hybrid Genetic Algorithms and Cluster Analysis." *Journal of Environmental Informatics,* vol.1(2), pp.40-47., (2003)

Liu J. G. and Mason P. J., *Image Processing and GIS for Remote Sensing*, Wiley Blackwell, (2016)

Lu D. and Weng Q. "A survey of Image Classification Methods and Techniques for Improving Classification Performance". *International Journal of Remote Sensing* vol.28(5): pp.823-870, (2007).

Mallinis G., Koutsias N., Tsakiri-Strati M. and Karteris M., "Object-based Classification Using Quickbird Imagery for Delineating Forest Vegetation Polygons in a Mediterranean Test Site". *ISPRS Journal of Photogrammetry & Remote Sensing* vol.63, pp. 237-250, (2008)

Matsuyama T., Wu X., Takai T. and Wada T., "Real-Time Dynamic 3-D Object Shape Reconstruction and High-Fidelity Texture Mapping for 3-D

Video", *IEEE Transactions On Circuits and Systems For Video Technology.,* vol. 14, no. 3, pp. 357-369, (2004)

Mitchell T. M., *Machine Learning*, McGraw-Hill Science/Engineering/Math, (1997)

Moran E. F., "Land cover classification in a complex urban-rural landscape with quickbird imagery. Photogrammetric Engineering and Remote Sensing" vol. 76(10), pp. 1159-1168, (2010)

Niem W. and Broszio H., "Mapping Texture From Multiple Camera Views Onto 3D-Object Models For Computer Animation", *Proceedings of the International Workshop on Streoscopic and Three Dimensional Imaging*., pp. 99-105, (1995)

Nilsson N. J., *Artificial Intelligence New Synthesis*, Morgan Kaufman Publishing, (1998)

OpenCv, "OpenCV About [online]", (3 Ocak 2018), https://opencv.org/about/, 2000

Opendronemap, "What is it? [online]", (13 Kasım 2019), https://opendronemap.github.io, 2018

Poole D., Mackworth A. and Goebel R., *Computational Intellingence A Logical Approach*, Oxford University Press, (1998)

Pyglet, "Pyglet About [online]", (18 Ekim 2019) , http://pyglet.org/, 2018[a]

Pyglet, "Pyglet Getting Started [online]", (18 Ekim 2019), https://pyglet.readthedocs.io/en/stable/modules/window.html, 2018[b]

Pypi, "PyWaveFront Project Description [online]", (16 Ekim 2019), https://pypi.org/project/PyWavefront/, 2019

Python, "The Python Tutorial [online]", (18 Kasım 2018), https://docs.python.org/3/tutorial/index.html, 2018[a]

Python, “The Python Tutorial Modules [online]”, (18 Kasım 2018), https://docs.python.org/3/tutorial/modules.html, 2018[b]

PythonGTKTutorial, “Python GTK Getting Started [online]”, (3 Eylül 2019), https://python-gtk-3-tutorial.readthedocs.io/en/latest/introduction.html, 2007

Sirmacek, B. and Unsalan C., "Road Network Extraction Using Edge Detection and Spatial Voting," *Pattern Recognition (ICPR), 2010 20th International Conference* pp.3113-3116, 23-26., (2010)

Vaihingen B. and S. Labeling S., “Use of the Stair Vision Library within the ISPRS Use of the Stair Vision Library within the ISPRS 2D,”, (2016).

Vandana S., ChandraKanth R. and Ramachandran R., "Semi-Automatic Road Extraction Algorithm For High Resolution Images Using Path Following Approach*," ICVGIP02*, pp. 201-207., (2002)

Veljanovski T., Kanjir U. and Ostir K., “Object-based Image Analyses of Remote Sensing Data”. *Geodetski Vestnik* vol.55(4), 665-688, (2011)

Wang L., Qin Q., Du S., Chen D., and Tao J., "Road Extraction From Remote Sensing Image Based on Multiresolution Analysis," *in International Symposium on Remote Sensing of Environment Saint Petersburg, Russian Federation*, (2005).

Wassenaar T., Robbez-Masson J. M., Andrieux P. and Baret F., “Vineyard Spatial Structure Analysis by Perfield Aerial Photograph Processing”, *International Archives of Photogrammetry and Remote Sensing* vol. 33(Part B7), pp. 1692-1699, (2000)

Yu Q., Gong P., Clinton N., Biging G., Kelly M. and Schirokauer D., “Object-based Detailed Vegetation Classification with Airborne High Spatial Resolution Remote Sensing Imagery”. *Photogrammetric Engineering and Remote Sensing* vol.72, pp. 799-811, (2006)

Zhang C., Shunji M., and Emmanuel B., “Road Network Detection by Mathematical Morphology”, ISPRS Workshop "3D Geospatial Data

Production: Meeting Application Requirements", Paris, France, 185-200, (1999)

Zhang H. J., Vailaya A. And Jain A., “On image classification: City images vs. landscapes,” Pattern Recognit., vol. 31, no. 12, pp. 1921– 1935, (1998).

Zhang Y., Wang W., Yang N., Wang F., Cao T., and Eklund P., “A review of road extraction from remote sensing images,” J. Traffic Transp. Eng. (English Ed., vol. 3, no. 3, pp. 271–282, (2016).

Zhao, H., Kumagai, J., Nakagawa, M. and Shibasaki, R., "Semi-automatic Road Extraction from High- resolution Satellite Image," Proc. Photogrammetric Computer Vision ISPRS Commission III, Symposium, September 9-13, 2002, Graz, Austria, pp. 406-411., (2002)

7. ÖZGEÇMİŞ

Adı Soyadı	:	Aziz Dursun GÖKTEPE
Doğum Yeri ve Tarihi	:	Sarayköy/Denizli 14.08.1993
Lisans Üniversite	:	Pamukkale Üniversitesi
Elektronik posta	:	azizdursungoktepe@gmail.com
İletişim Adresi	:	15 Mayıs Mahallesi 762. Sokak No:46/26 Pamukkale/DENİZLİ