TRAKYA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

LİSANS BİTİRME PROJESİ DÖNEM RAPORU
PROJE I

PROJE ADI:
DİLSİZ ÇEVİRMENİ

HAZIRLAYANLAR:
1201602020 Ozan Can SARI
1201602039 Buse UYSAL
1201602649 Melisa VASİJA
1201602617 Anal KARAALI

DANIŞMAN:
Dr. Öğr. Üyesi Rembiye KANDEMİR

Edirne, Ocak 2023

İÇİNDEKİLER

KISALTMA LİSTESİ

TİD : Türk İşaret Dili

IDA : İşaret Dili Algılama

M. Ö. : Milattan Önce

ABD : Amerika Birleşik Devletleri

WPF : Windows Presentation Foundation

NYP : Nesne Yönelimli Programlama

DDİ : Doğal Dil İşleme

AI : Artificial İntelligence

ML : Machine Learning

IDE : Integrated Development Environment

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Bu proje işitme engelli bireyler ile daha kolay bir şekilde iletişim kurabilmek, engelli bireyin işaret dili bilmeyen birisine ihtiyaçlarını rahatlıkla anlatabilmesi ve işaret dili öğrenilmesinde kolaylık sağlanması adına hazırlanmıştır.

Bu projenin planlanmasında, araştırılmasında ve yürütülmesinde bizlere ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübesiyle bizlere yol gösteren sayın hocamız Dr. Öğr. Üyesi Rembiye KANDEMİR'e sonsuz teşekkürlerimizi sunarız. Kendisine sadece projemize sağladığı yönlendirme ve destek için değil, aynı zamanda bize öğrenciler olarak ilham verdiği için içtenlikle teşekkür ederiz.

Projede birlikte emek harcayan dört arkadaş olarak, her bir arkadaşımıza içten saygılarımızı iletiyoruz. Gelecekte, bilgisayar mühendisi olarak edindiğimiz bilgi ve becerileri her zaman toplum için faydalı projelerde kullanacağımıza söz veriyoruz.

ÖZET

Günlük yaşantımızın karmaşıklığında farkına varılmayan birçok zorlukla karşılaşıyoruz. Bu zorluklardan biri, işitme engelli bireylerin iletişim zorluğuyla baş etmeleridir. Projemiz, bu iletişim zorluklarını en aza indirmeyi amaçlayarak geliştirilmiştir.

Proje, esasen işitme engelli bireylerin işaret dili bilmeyen kişilerle daha kolay iletişim kurmalarını hedeflemektedir. Kullanıcılar, kamera karşısına geçerek işaret diliyle iletişim kuracak ve bu işaretleri Türkçe'ye çevirerek diğer kişilerin algılamasını sağlayacaktır. Bu sayede, işitme engelli bireyler, işaret dili bilmeyen kişilerle daha etkili bir şekilde iletişim kurma imkânına sahip olacaklardır.

Projemiz, teknolojinin gücünü kullanarak engelli bireylerin toplumsal katılımını artırmayı ve iletişimde karşılaştıkları zorlukları en aza indirmeyi amaçlamaktadır. Bu süreçte, kullanıcıların günlük yaşamlarında daha etkin ve bağımsız olmalarını sağlamak adına projemizin önemli bir adım olduğuna inanıyoruz.

ABSTRACK

In the complexity of our daily lives, we encounter many challenges that often go unnoticed. One of these challenges is dealing with communication difficulties faced by individuals with hearing impairments. Our project has been developed with the aim of minimizing these communication challenges.

The project essentially aims to facilitate communication for individuals with hearing impairments with those who do not know sign language. Users will stand in front of a camera, communicate using sign language, and the system will translate these signs into Turkish to allow others to perceive the message. This way, individuals with hearing impairments will have a more effective means of communicating with those who do not know sign language.

By harnessing the power of technology, our project seeks to increase the social participation of disabled individuals and minimize the communication challenges they face. Throughout this process, we believe that our project is a significant step in enabling users to be more effective and independent in their daily lives.

BÖLÜM 1

GİRİŞ

İnsan kulağı belirli sesleri duyma yeteneğine sahiptir. Fakat bazı biyolojik nedenlerden dolayı duyma yetisi azalabilir veya yok olabilir. Bu duruma *"işitme kaybı"* denir. Bu durumu yaşayan işitme engelli bireyler işaret dili sayesinde iletişim kurarlar.

Türk İşaret Dili (TİD) dünyadaki en eski işaret dillerinden biri olarak kabul edilir (Zeshan, 2003). TİD Türkiye'de ve Kuzey Kıbrıs Türk Cumhuriyeti'nde kullanılmaktadır. Her işaret dili gibi TİD de Türkçe'nin gramer yapısından farklı olarak kendine has bir gramer yapısı mevcuttur. Türkçe'de eş seslilik mevcutken, TİD'de eş işaretlilik mevcuttur.

1.1 Projenin Amacı ve Hedefi

İşitme engelli bireyler ile daha kolay iletişim kurulabilmesi için işaret dili ve görüntü işleme tekniğinden yararlanılarak yeni bir sistem tasarlanması amaçlanmıştır. Bu proje sayesinde engelli bireylerin ihtiyaç ve arzularını, yaşadıkları iletişim problemini en aza indirerek bildirmeleri hedeflenmektedir.

Unutulmamalıdır ki her insan bir engelli adayıdır. Bu sebeple TİD'in öğrenilmesinde kolaylık sağlanması ve TİD bilen kişi sayısının artması da hedeflenmektedir.

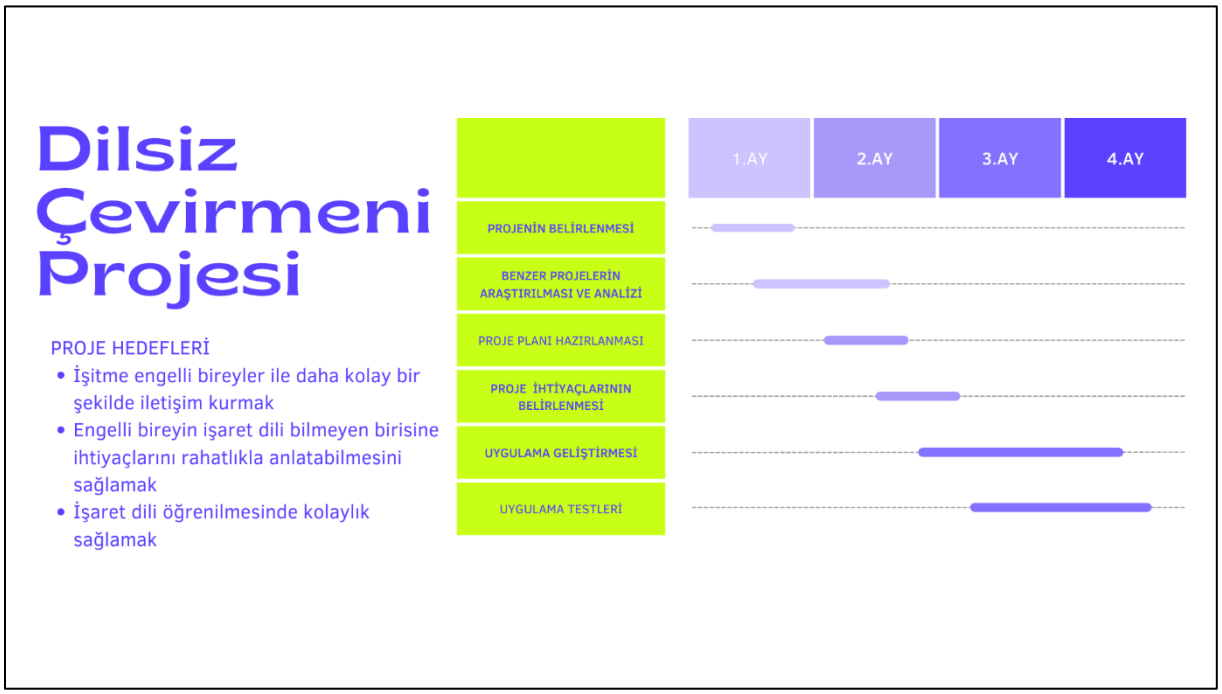
Şekil 1.1

1.2 Gereksinim Analizi

Projenin ihtiyaç duyduğu ana modüller analiz edilmiş (software requirements specification), proje amaçları ve hedefleri detaylandırılmıştır.

1.2.1 Zaman Fizibilitesi

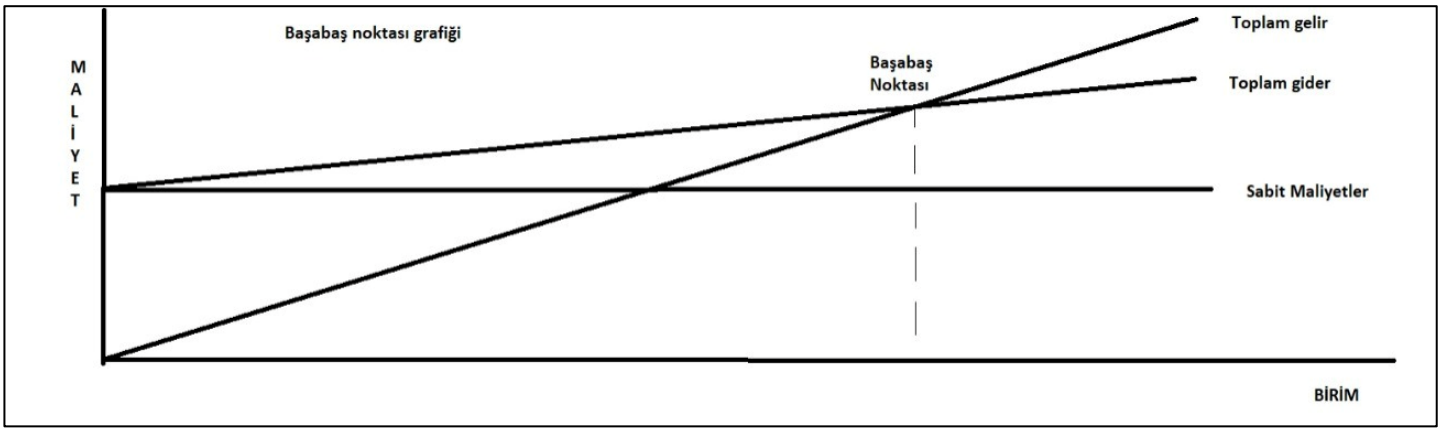
Projenin zaman fizibilitesi hazırlanmıştır. Planlanan gantt şeması Çizelge 1.1'de gösterilmektedir.



Çizelge 1.1 Projenin Zaman Fizibilitesi

1.2.2 Ekonomik Fizibilite

Proje için gerekli maliyetler hesaplanmıştır. Bu doğrultuda ekonomik fizibilite hazırlanmıştır. Planlanan gantt şeması Çizelge 1.2'de gösterilmektedir.



Çizelge 1.2 Projenin Ekonomik Fizibilitesi

1.2.3 Teknik Fizibilite

Proje Tanımı:

Proje Adı: Dilsiz Çevirmen
Proje Amacı: İşitme engelli bireylerin ve işitme düzenekleri kullanıcılarının konuşmalarını anlamalarına yardımcı olmak için bir yazılım geliştirmek.

Teknik Gereksinimler:
İşaret Dili Tanıma: İşaret dili anlama ve çözümleme yetenekleri.
Çeviri Motoru: Çeşitli diller arasında anlam çıkarabilen bir çeviri motoru.
Kullanıcı Arayüzü: Kullanıcıların kolayca etkileşimde bulunabileceği bir arayüz.

Yazılım Geliştirme Süreci:
Agile veya Scrum gibi esnek yazılım geliştirme metodolojileri kullanılabilir.
İteratif geliştirme ile kullanıcı geri bildirimleri dikkate alınabilir.

Teknolojik Altyapı:
Bulut Bilişim: Hesaplama gücü ve depolama sağlamak için bulut tabanlı altyapılar kullanılabilir.
Yüksek Performanslı Sunucular: Hızlı ve etkili işlemler için güçlü sunucular.

Veri Güvenliği ve Gizlilik:
Kullanıcı verilerinin güvenliği için endüstri standardı şifreleme yöntemleri kullanılmalıdır.
GDPR ve diğer gizlilik düzenlemelerine uygunluk sağlanmalıdır.

Uyum ve Entegrasyon:
Mobil uygulama ve web tarayıcıları üzerinden erişilebilirlik sağlamak için platform bağımsız bir tasarım.
İşitme cihazları ve diğer teknolojik araçlarla uyumluluk.

Test ve Doğrulama:
Kapsamlı test süreçleriyle yazılımın doğruluğu ve güvenilirliği sağlanmalıdır.
Beta testleri ve kullanıcı geri bildirimleri ile sürekli iyileştirmeler.

Güncelleme ve Bakım:
Yazılımın güncel ve güvenli kalması için periyodik bakım ve güncelleme yapılmalıdır

1.3 İşaret Dilinin Tarihsel Gelişimi

İşaret diline dair en eski yazılı kayıt M.Ö. 5. yüzyıla dayanmaktadır. 19. yüzyıla kadar işaret dili parmak alfabesi ile sınırlıydı. İlk parmak alfabesini geliştiren kişinin Pedro Ponce'de Leόn (1520-1584) olduğu söylenir.

Türk İşaret Dili'nin (TİD) kökeni kesin olarak bilinmese de varsayımsal olarak kökü Osmanlı saraylarında bulunan sağırlara dayandırılmaktadır. Eğitim konusunda ise ilk olarak II. Abdülhamit döneminde yapılmıştır. Bu dönemde sağırlar için çeşitli yerlerde sağır okulları açılmıştır.

Türkiye Cumhuriyeti'nin ilk yıllarında açılan okullar oralist metotla çalışmaktaydı, fakat 2000'li yıllarda bu metodun terk edilmesiyle, Türkiye'de bulunan okullar işaret dilinde eğitime geçti.

İşaret dillerinin genellikle bulunduğu bölgelerdeki konuşma dilleriyle arasında hiçbir dilbilimsel bağ bulunmaz. İşaret dili ve konuşma dili arasındaki ilişki karmaşık ve konuşma dilinden çok bulunduğu ülkeye göre değişim gösteren bir şeydir.

Zaman zaman, sağır insanların yaygınlığının yeterli olduğu yerlerde, yerel topluluğun tamamı tarafından bir işaret dili kullanılmaya başlanır ve bu *"köy işaret dili"* ya da *"ortak işaret topluluğu"* oluşmasına yol açar. En çok bilinen örnekler: Martha's Vineyard İşaret Dili (ABD), Mardin İşaret Dili (Türkiye), Orta Toroslar İşaret Dili (Türkiye).

İşaret dili çevirisi bazen konuşma içeren televizyon programları için sağlanmaktadır. Paddy Ladd 1980'lerde Britanya televizyonunda sağırlar için bir program başlattı ve kendisi işaret dilini televizyona taşıyan ve sağır çocukların işaret dilinde eğitim görmesini sağlayan kişi olarak bilinir. [1]

1.4 Projenin Benzerleri

İşaret dili tanıma ile ilgili projeler mevcuttur. Bu projelerde de görüntü işleme yöntemi kullanılmaktadır.

1.4.1 Üç Boyutlu Sanal Model ile Türk İşaret Dili Simülasyonu

2018 yılında Karabük Üniversitesi'nde Mehmet Fatih KARACA tarafından doktora tezi olarak hazırlanan bu proje TİD'in detaylı araştırılmasıyla birlikte jest ve mimik kullanarak oluşturulan kapsamlı bir öğrenme aracı olarak tasarlanmıştır. Simülasyon tasarlanırken C# programlama dilinden ve Unity oyun motorundan yararlanılmıştır. [2]

1.4.2 Ses ve Metin Olarak Girilen İşaret Dili Hareketlerinin Robot Kol Tarafından Gerçekleştirilmesi

Bekir AKSOY, Zakaria GHAZAL, Ramazan ŞENOL, Mevlüt ERSOY tarafından hazırlanan çalışmada ses yoluyla ve klavyeden giriş olarak algılanan ifadelerin robot kol ile işaret diline çevrilmesi amaçlanmıştır. 3B yazıcı ile tasarlanan robot kol, Python programlama dili, Google Speech, Pyserial ve Serial kütüphanesinden yararlanılmıştır. [3]

1.4.3 Türk İşaret Dili Kaynak Sitesi

TİD Sözlüğü, Boğaziçi Üniversitesi, Bilgisayar Mühendisliği, Algısal Zekâ Laboratuvarı'nda yürütülen "İşaret Dili Eğitmeni" projesi kapsamında ortaya çıkmıştır. Birçok projede kaynak olarak kullanılmaktadır. Veritabanı 8 farklı işaret sınıfından oluşur: Nötr, Kafa LR, Baş Yukarı, Kafa F, Üzüntü, Kafa UD, Mutluluk, Mutlu UD. Bu sınıflandırmalar mimik hareketlerine göre olan bir sınıflandırmadır. Videolarda 6 kadın, 5 erkek bulunmaktadır. 11 farklı konuyu içermektedir. Her konu 8 dersin her biri için 5 tekrar gerçekleştirir. Veritabanında toplam 440 video mevcuttur. Her video yaklaşık 1-2 saniye sürer. Videolar ışık ayarı yaplmış odalarda çekilmiştir. Video dosyaları *"[konu adı]_[#classId]_[#repetitionNo].avi"* olarak adlandırılır. İşaretler: Nilay ARSLAN, Tamyürek ÖZPARLAK'a; kod tasarım: İsmail ARI'ya; kamera-kurgu: Pınar SANTEMİZ'e aittir. [4]

1.4.4 İşaretçe

İşaretçe görüntülü bir TİD sözlüğüdür. Bu alandaki kaynak eksikliğinden dolayı yaşadıkları sıkıntıyı çözmek amacıyla Başak ve Serdar ULUDAĞ'ın hazırlamış oldukları bir platformdur. [5]

1.4.5 IDA (İşaret Dili Algılama) Projesi

Ertuğrul Kuşva tarafından hazırlanan proje kameradan algılanan hareketleri yazıya çevirmektedir. Python programlama dili ile yazılan projede MediaPipe, OpenCv ve Keras kütüphanelerinden yararlanılmıştır. [6]

1.4.6 İşaret

Recep Kocur tarafından yapılan bir uygulamadır. Google Play ve App Store'da mevcuttur. Kelimelerin TİD'deki karşığını olan videoları göstermektedir. [7]

BÖLÜM 2

SİSTEM BİLEŞENLERİ

2.1 Projede Kullanılacak Teknolojiler

İşaret dili çevirmeni projesinde kullanılacak teknolojiler aşağıdaki gibi sıralanabilir:

Donanım ve İşletim Sistemi
Proje, Windows 10 işletim sistemi üzerinde çalışan bir dizüstü bilgisayarda geliştirilmiştir. Projenin üzerinde çalışacağı donanım sistemi, bilgisayarın işlemci gücü, RAM miktarı ve ekran çözünürlüğü gibi faktörler göz önünde bulundurularak seçilmiştir.

Programlama Dili
Projenin ana geliştirme dili olarak Python kullanılmıştır. Python, güçlü ve esnek bir dil olması nedeniyle işaret dili çevirmeni gibi karmaşık bir proje için uygun bir seçimdir. Python, aşağıdaki özelliklere sahiptir:

Projede, OpenCV görüntü işleme kütüphanesi kullanılmıştır. OpenCV, görüntü işleme için kullanılan açık kaynaklı bir kütüphanedir.

Projede, TensorFlow makine öğrenimi kütüphanesi kullanılmıştır. TensorFlow, makine öğrenimi için kullanılan açık kaynaklı bir kütüphanedir.

Projede, MySQL veritabanı kullanılmıştır. MySQL, açık kaynaklı bir ilişkisel veritabanı yönetim sistemidir.

2.2 Metotlar ve Yöntemler

Proje tasarlanırken birçok metot ve yöntemden yararlanılmıştır.

2.2.1 Donanım Sistemi ve İşletim Sistemi

Proje, genel olarak kişisel bilgisayarları hedeflerken, farklı işletim sistemlerini de destekleme potansiyeline sahiptir. Öncelikli olarak Windows işletim sistemi düşünülmüş olsa da, projenin ölçeği ve kullanıcı kitlesi göz önüne alındığında, diğer popüler işletim sistemleri olan macOS ve Linux'a da destek verilebilir.

2.2.2 Kullanılabilecek Programlama Dilleri

2.2.2.1 C#

C#, projemiz için ideal bir seçenek olmasını çeşitli avantajlarına dayandırıyoruz. İlk olarak, C#, özellikle Windows tabanlı uygulamalar geliştirmek için tasarlanmış güçlü bir dil olarak bilinir.

C# dilinin, Windows Presentation Foundation (WPF) gibi araçlarla entegrasyonu, etkileyici ve kullanıcı dostu grafiksel kullanıcı arayüzleri oluşturmayı kolaylaştırır. Bu, projemizin kullanıcılar için daha erişilebilir ve anlaşılır olmasına olanak tanır.

Nesne yönelimli programlama (NYP) özellikleri, C# dilinin kodu modüler ve yönetilebilir kılmasına katkı sağlar. Bu da projenin büyüklüğü ve karmaşıklığı arttıkça geliştirme sürecini daha etkili ve sürdürülebilir hale getirir.

Geniş kütüphane ve topluluk desteğine sahip bir dil olduğu için, projenin geliştirilmesi hızlanır ve karşılaşılan potansiyel sorunlara daha hızlı çözümler üretebiliriz. Ayrıca, Visual Studio gibi gelişmiş entegre geliştirme ortamları, C# dilini kullanarak projenin daha verimli bir şekilde yönetilmesine olanak tanır. [8]

2.2.2.2 Java

Java, projemizin ihtiyaçlarına uygun bir programlama dili olarak seçildiğinde çeşitli avantajlar sunabilir. Java, "write once, run anywhere" prensibiyle bilinir. Bu da Java ile yazılmış bir programın farklı platformlarda (Windows, macOS, Linux) çalışabilir olduğu anlamına gelir. Bu özellik, projenin farklı bilgisayar sistemlerinde kullanılabilirliğini artırabilir.

Nesne yönelimli programlamanın (NYP) temel prensiplerini destekler. NYP, kodun modüler ve yönetilebilir olmasına olanak tanır. Bu özellik, projenin büyüklüğü arttıkça daha iyi organize edilmesine yardımcı olabilir. [9]

2.2.2.3 Python

Python'un açık kaynaklı doğası ve büyük bir geliştirici topluluğuna sahip olması, projenin sürekli geliştirilmesi ve güncellenmesi için önemli bir avantaj sağlar. Geliştirici topluluğunun büyüklüğü, projemizle ilgili olası sorunlara hızlı çözümler bulmamıza ve yenilikleri takip etmemize yardımcı olabilir.

Python'un platform bağımsızlığı da projemizin farklı ortamlarda sorunsuz çalışabilmesine olanak tanır. Bu da projenin daha geniş bir kullanıcı kitlesine ulaşmasını sağlar. [10]

2.2.3 Seçilen Programlama Dili Python'dır

Projemizin geliştirilmesinde Python programlama dilini tercih etmemiz, çeşitli avantajları ve projemizin özel gereksinimleri doğrultusunda değerlendirmelerimize dayanmaktadır.Python, proje ekibimizin ihtiyaçlarına yönelik geniş bir kütüphane ve modül yelpazesi sunan bir programlama dilidir. Bu kütüphaneler, projemizin karmaşıklığını ve özel ihtiyaçlarını karşılamak adına güçlü araçlar sağlar.

Özellikle, işitme engelli bireylerle etkili iletişim kurabilmek için kullanıcı arayüzünden doğal dil işleme (NLP) özelliklerine, veri analizi ve görselleştirmeye kadar birçok alanda geniş bir destek sunar.

Python'un açık kaynak olması ve geniş bir geliştirici topluluğuna sahip olması, projemizin sürekli geliştirilmesi ve güncellenmesi için önemli bir avantajdır. Ayrıca, Python'un platform bağımsız olması, projenizin farklı işletim sistemlerinde sorunsuz bir şekilde çalışabilmesine olanak tanır.

Proje sürecinde hızlı prototipleme yapma imkânı, Python'un dinamik tip sistemine ve esnek sözdizimine dayanır. Bu özellik, projenin farklı aşamalarında hızlıca prototip oluşturarak tasarımın ve işlevselliğin daha etkili bir şekilde değerlendirilmesine katkı sağlar.

Yapay zekâ (AI) ve makine öğrenimi (ML) alanlarında Python'un popülerliği ve bu konuda sunduğu kapsamlı kütüphaneler (TensorFlow, PyTorch) da projemizin yapay zekâ entegrasyonu gerektirmesi durumunda büyük bir avantaj sağlar.

Sonuç olarak, Python'un sunduğu geniş kapsamlı özellikler, kütüphaneler ve topluluk desteği, projemizin başarıyla uygulanması ve işitme engelli bireylerle etkili iletişim kurulması için uygun bir seçenek olduğunu göstermektedir. [10]

2.2.4 Tümleşik Geliştirme Ortamları

2.2.4.1 PyCharm:

Projemizin ana tümleşik geliştirme ortamı olarak PyCharm'ı seçtik. PyCharm, özellikle Python geliştirmeye odaklanmış güçlü bir IDE'dir. Hızlı hata ayıklama yetenekleri, entegre test araçları ve Python diline özel zengin özellikleri ile projemizin Python programlama dilindeki geliştirme sürecini optimize etmek için tercih edilmiştir.

2.2.4.2 Visual Studio Code:

Visual Studio Code, projemizde alternatif bir tümleşik geliştirme ortamı olarak değerlendirilmiştir. Hafif yapısı, geniş eklenti desteği ve çok platformlu uyumluluğu ile öne çıkar. Ancak, PyCharm'ın Python odaklı özellikleri ve araçlarına odaklanmış olmamız nedeniyle projenin ana IDE'si olarak tercih etmedik.

2.2.4.3 Eclipse:

Eclipse, genel amaçlı bir IDE olup çeşitli programlama dillerini destekler. Projemizde ek olarak değerlendirilen bir seçenek olarak karşımıza çıkmaktadır. Geniş eklenti desteği ve esnek proje yönetimi araçları, projenin özelleştirilebilirliğini artırarak değişen gereksinimlere hızlı bir şekilde adapte olmasını sağlar. Ancak, Python odaklı geliştirmemiz nedeniyle ana IDE olarak seçilmedi.

Bu tümleşik geliştirme ortamları, projemizin ihtiyaçlarına uygun olarak değerlendirilmiş ve seçilmiştir.

2.2.5 Geliştirme Sürecini Kolaylaştıracak Kütüphaneler

2.2.5.1 OpenCV:

OpenCV (Open Source Computer Vision Library), görüntü işleme uygulamaları geliştirmek için kullanılan güçlü bir açık kaynak kütüphanedir. İlk olarak 1999 yılında Intel tarafından başlatılan bu proje, günümüzde sürekli olarak geliştirilerek geniş bir kullanıcı tabanına sahiptir. OpenCV, C++ ve Python gibi popüler programlama dillerini destekler, bu sayede geliştiricilere esneklik sağlar. Kütüphane, geniş bir yelpazede görüntü işleme işlevselliği sunar, örneğin nesne tanıma, yüz tanıma, hareket tespiti, kamera kalibrasyonu gibi birçok temel ve gelişmiş görevi gerçekleştirebilir.

OpenCV'nin başlıca avantajları arasında genişletilebilirlik, hızlı yürütme, platform bağımsızlık ve ücretsiz olarak kullanılabilir olması bulunur. Kapsamlı belgeleri ve topluluğu, geliştiricilere kütüphaneyi etkili bir şekilde kullanmaları konusunda yardımcı olur. OpenCV'nin özellikle işaret dili tanıma projelerinde kullanılması, görüntü işleme algoritmaları sayesinde el hareketlerini algılamak ve anlamak için güçlü bir temel sağlar. [11]

2.2.5.2 Tensorflow:

TensorFlow, açık kaynaklı bir makine öğrenimi ve derin öğrenme kütüphanesidir ve Google tarafından geliştirilmektedir. Derin öğrenme modelleri oluşturmak, eğitmek ve dağıtmak için kullanılan güçlü bir araçtır. TensorFlow, özellikle büyük veri setleri üzerinde karmaşık öğrenme süreçlerini gerçekleştirmek için tasarlanmıştır. Kütüphane, makine öğrenimi projeleri için geniş bir yelpazede kullanılabilecek modüler araçlar ve kaynakları içerir.

TensorFlow'un başlıca özelliklerinden biri, kullanıcılarına derin öğrenme modellerini hızlı ve etkili bir şekilde oluşturma yeteneğidir. Hem araştırmacılar hem de endüstri profesyonelleri tarafından yaygın olarak kullanılan bu kütüphane, görüntü tanıma, doğal dil işleme, ses analizi gibi çeşitli uygulamalarda başarılı sonuçlar elde etmek için uygun algoritmalar sunar.

TensorFlow, aynı zamanda TensorFlow Lite gibi platformlar arası entegrasyon araçlarıyla mobil ve yerleşik sistemlerde kullanımı da destekler. Bu, özellikle taşınabilir cihazlarda veya gömülü sistemlerde uygulama geliştirme süreçlerini kolaylaştırır. TensorBoard gibi araçlar, model eğitimi ve performansını görselleştirmek için kullanıcı dostu bir arabirim sunar.

İşaret dili tanıma projeleri için TensorFlow, önceden eğitilmiş derin öğrenme modellerini kullanma, transfer öğrenme uygulama ve özelleştirilmiş modeller oluşturma konusunda geliştiricilere esneklik sağlar. [12]

2.2.5.3 Imutlis

Imutils, görüntü işleme projeleri için kullanılan bir yardımcı kütüphanedir. Bu kütüphane, görüntü işleme görevlerini hızlandırmak ve basitleştirmek amacıyla geliştirilmiştir. İşaret dili tanıma projeleri gibi uygulamalarda özellikle kullanışlıdır.

Imutils kütüphanesinin sağladığı bazı temel özellikler şunlardır:

Görüntü İşleme İşlevleri: Imutils, temel görüntü işleme görevlerini gerçekleştirmek için çeşitli işlevler içerir. Bu işlevler arasında yeniden boyutlandırma, döndürme, kaydırma gibi temel işlemler bulunur. Bu özellikler, görüntüler üzerindeki çeşitli dönüşümleri kolayca gerçekleştirmeyi sağlar.

OpenCV Entegrasyonu: Imutils, OpenCV kütüphanesi ile uyumlu olarak çalışır. OpenCV ile gelen bazı işlevselliği daha kullanıcı dostu ve kısa bir şekilde kullanma imkânı sunar. Bu, OpenCV'yi kullanırken kodun daha okunabilir ve sade olmasını sağlar.

Video İşleme: Video işleme projelerinde kullanılan bazı özel işlevler içerir. Örneğin, video dosyalarından veya kamera akışlarından görüntü okuma ve işleme yeteneklerini içerir.

Genişletilebilirlik: Imutils, geniş bir geliştirici topluluğu tarafından kullanılmakta ve desteklenmektedir. Bu da projelerinizi hızlı bir şekilde geliştirir. [13]

2.2.5.4 Keras

Keras, yüksek seviyeli bir derin öğrenme API'sidir ve özellikle TensorFlow ile uyumludur. Keras, derin öğrenme modelleri oluşturmak, eğitmek ve değerlendirmek için kullanılan kullanıcı dostu bir arayüz sunar. İşaret dili tanıma projeleri gibi çeşitli görevlerde kullanılmak üzere tasarlanmıştır.

Keras'ın temel özellikleri şunlardır:

Modüler ve Esnek Tasarım: Keras, modüler bir yapıya sahiptir ve kullanıcıların kolayca özel derin öğrenme modelleri oluşturmalarını sağlar. Katmanlar, aktivasyon fonksiyonları ve optimizasyon algoritmaları gibi bileşenlerin kolayca birleştirilebilmesi, projelerin gereksinimlerine uygun modellerin tasarlanmasını kolaylaştırır.

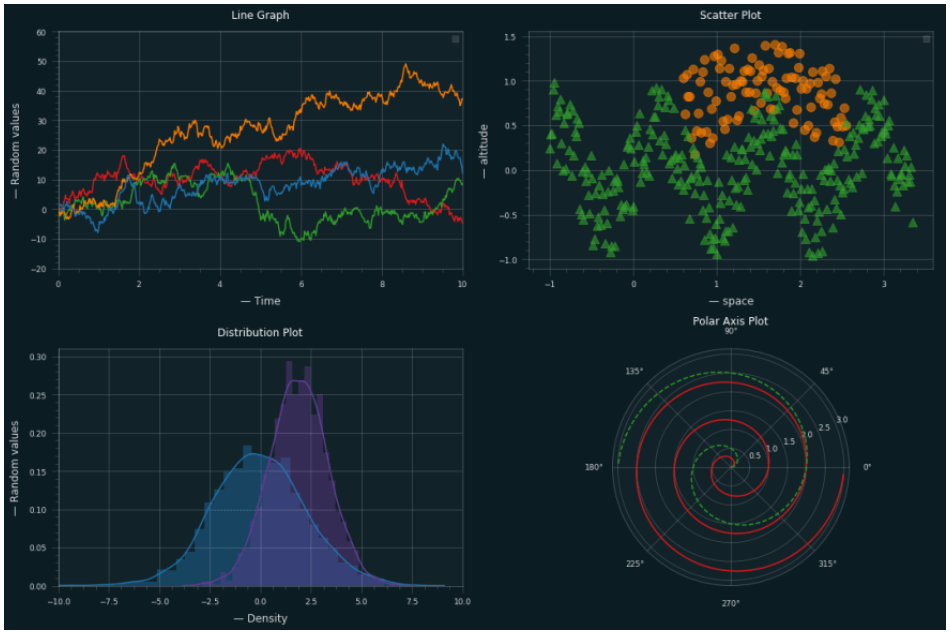
Önceden Eğitilmiş Modeller: Keras, popüler derin öğrenme modellerinin önceden eğitilmiş versiyonlarını içerir. Bu, kullanıcıların transfer öğrenme yöntemini kullanarak kendi projelerine hızlı bir başlangıç yapmalarını sağlar.

Çeşitli Aktivasyon Fonksiyonları: Keras, çeşitli aktivasyon fonksiyonlarını destekler. Sigmoid, ReLU, tanh gibi yaygın olarak kullanılan fonksiyonlar, kullanıcılara modelin öğrenme sürecini etkileme ve özelleştirme imkânı sağlar.

GPU Desteği: Keras, GPU tabanlı hesaplamaları destekler ve büyük veri setleri üzerinde eğitim süreçlerini hızlandırır. Bu, derin öğrenme modellerinin daha büyük ve karmaşık veri setleri üzerinde etkili bir şekilde çalışabilmesini sağlar. [14]

2.2.4.6 Matplotlib

Matplotlib, veri görselleştirmesi için kullanılan güçlü bir Python kütüphanesidir. Bu kütüphane, çeşitli grafik türlerini oluşturmak, özelleştirmek ve görselleştirmek için geniş bir araç seti sunar. İşitme engelli bireylerle iletişimi geliştirmek ve işaret dili projelerinde sonuçları anlamak için önemli bir araçtır. Matplotlib grafikleri Şekil 2.1'de gösterilmektedir. Matplotlib kütüphanesinin temel özellikleri şunlardır:



Şekil 2.1 MATPLOTLİB Grafikleri

Çeşitli Grafik Türleri: Kütüphane, çizgi grafikleri, sütun grafikleri, dağılım grafikleri, pasta grafikleri, histogramlar ve daha birçok grafik türünü destekler. Bu, çeşitli veri türlerini etkili bir şekilde görselleştirmek için esneklik sağlar.

Özelleştirme Yetenekleri: Grafiklerin renkleri, etiketleri, eksenleri ve diğer birçok özelliği üzerinde detaylı özelleştirmeler yapma imkânı sunar. Bu, projenin özel gereksinimlerine uygun görselleştirmeler oluşturmayı kolaylaştırır.

Interaktif Grafikler: Jupyter Notebook gibi ortamlarda interaktif grafikler oluşturma yeteneği sağlar. Bu, veriyi dinamik bir şekilde keşfetmeyi ve anlamayı kolaylaştırır.

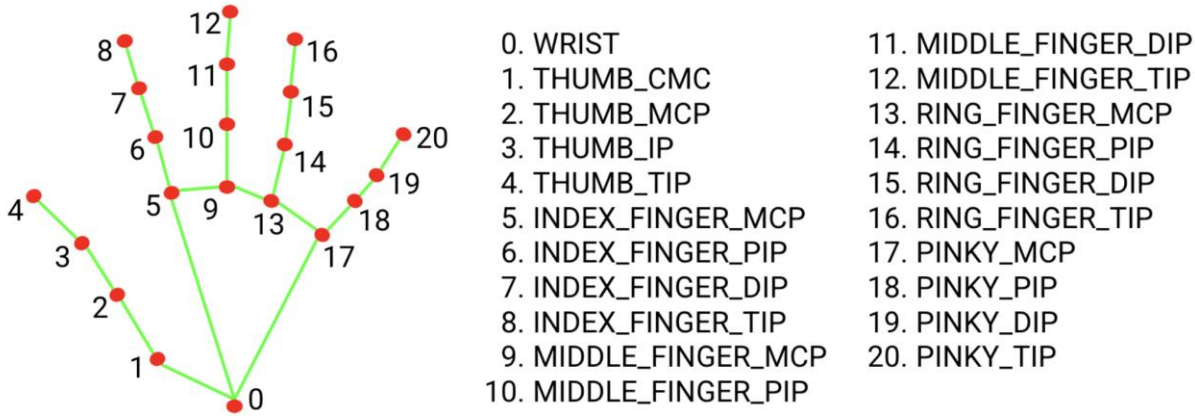
Birden Çok Eksen Desteği: Birden çok eksen içeren karmaşık grafikler oluşturabilir. Bu, farklı veri setlerini aynı grafik üzerinde karşılaştırmak veya ilişkilendirmek için kullanışlıdır.

Yüksek Kaliteli Çıktılar: Grafikler, yayın kalitesinde çıktılar üretmek için tasarlanmıştır. Bu, projenin sonuçlarını etkili bir şekilde sunmak için önemlidir.

Renk ve Kontrast Kontrolü: Matplotlib, renk skalaları ve paletleri üzerinde tam kontrol sağlar. Bu özellik, renk körü bireyler veya düşük görme yeteneğine sahip kullanıcılar için uygun kontrast ve renk seçimini mümkün kılar. İşitme engelli bireylerin görsel olarak bilgi almasını destekler. [15]

2.2.4.7 Mediapipe

MediaPipe, Google tarafından geliştirilen bir açık kaynaklı işaret işleme kütüphanesidir. Bu kütüphane, çeşitli görsel işleme görevlerini gerçekleştirmek ve uygulamalara entegre etmek için kullanılır. MediaPipe, özellikle el, yüz, vücut ve nesne izleme gibi görsel algılama görevlerini kolaylaştırmak için tasarlanmıştır. İşitme engelli bireylerle iletişimi geliştirmek ve işaret dili projeleri gibi uygulamalarda kullanılabilecek bir dizi özelliğe sahiptir. MediaPipe'ın el koordinatları Şekil 2.2'de, vücut koordinatları Şekil 2.3'te gösterilmektedir. MediaPipe'ın bazı temel özellikleri şunlardır:



Şekil 2.2 Mediapipe'ın algıladığı elde koordinatlarını döndürdüğü noktalar

Hands (Eller) Modülü: MediaPipe, ellerin izlenmesi ve analizi için özel bir "Hands" modülü içerir. Bu modül, el hareketlerini algılamak ve izlemek için kullanılır. İşitme engelli bireylerle etkileşimli bir sistemde, işaret dili hareketlerini tanımak için el izleme bu modül aracılığıyla gerçekleştirilebilir.

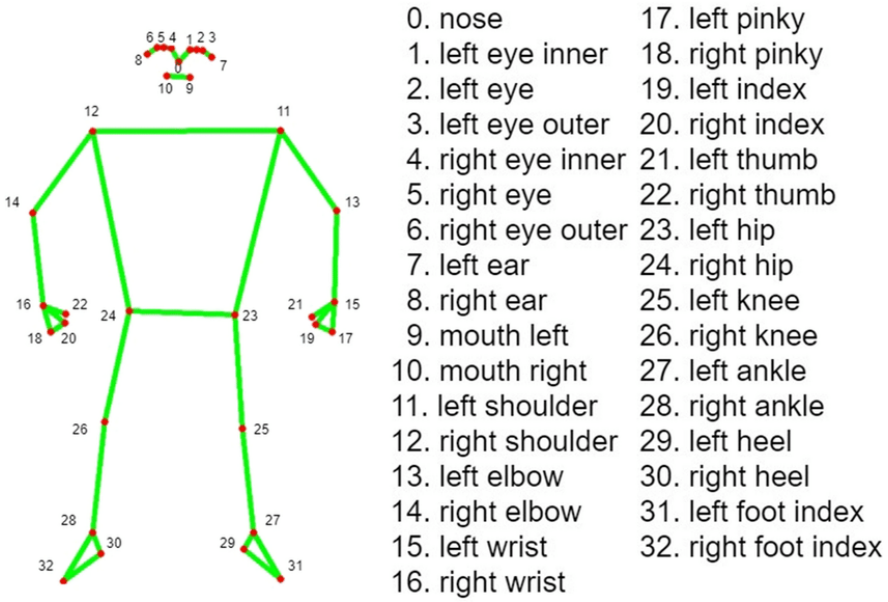
"Hands landmark" noktaları, elin belirli bölgelerindeki önemli noktaları temsil eder. Her bir landmark, eldeki belirli bir bölgeyi ifade eder. Bu noktalar, elin bileşenlerini, pozisyonunu ve hareketlerini anlamak için kullanılır. Bu noktalar genellikle 3D uzayda belirli koordinatlarla temsil edilir.

Örneğin, "wrist" (bilek) landmark'i elin bilek bölgesini temsil eder ve genellikle "0" numarası ile ifade edilir. "thumb" (baş parmak) bölgesindeki landmark'lar "1" ila "4" numaraları arasında yer alır, "index" (işaret parmağı) bölgesindeki landmark'lar "5" ila "8" numaraları arasında yer alır.

Bu "landmark" noktaları, elin pozisyonunu ve hareketlerini izlemek, analiz etmek veya tanımak için kullanılır. Özellikle işaret dili tanıma projeleri gibi uygulamalarda, bu noktalar, belirli el hareketlerini veya işaretleri tanımlamak için önemli bir rol oynar.

Holistic Modülü: "Holistic" modülü, vücut, el ve yüz izleme işlemlerini birleştiren geniş kapsamlı bir modüldür. İşitme engelli bireylerle iletişimde, yüz ifadelerini, el hareketlerini ve vücut pozisyonlarını takip ederek çeşitli işaret dili ifadelerini anlamak için kullanılabilir.

Birden Çok Platform Desteği: MediaPipe, birden çok platformda (Windows, macOS, Linux, Android, iOS) kullanılabilir. Bu özellik, projenizi farklı cihazlarda uygulamak veya taşımak istediğinizde esneklik sağlar.



Şekil 2.3 Mediapipe'ın algıladığı vücutta koordinatlarını döndürdüğü noktalar

Yüz ve El İzleme: MediaPipe, yüz ve el izleme işlemlerini kolaylaştıran özel modüllere sahiptir. Bu modüller, yüz ifadelerini ve el hareketlerini doğrulamak ve analiz etmek için kullanılabilir. İşitme engelli bireylerle etkileşimli bir sistemde, yüz ifadeleri ve el işaretleri önemli bir rol oynar.

"Face landmark" noktaları, yüz üzerinde belirli bölgeleri temsil eden referans noktalarıdır ve genellikle görsel işleme uygulamalarında yüz izleme veya analizi için kullanılır. Her bir nokta, yüzdeki belirli bir anatomi bölgesini temsil eder ve bu sayede yüzün farklı bölgelerindeki detayları belirleme olanağı sağlar. Örneğin, bu noktalardan biri, burunun ucu, diğerleri sol göz, sağ göz, ağızın sol köşesi, ağızın sağ köşesi ve çene gibi yüzdeki belirli özelliklere karşılık gelir.

Yüzdeki farklı bölgeleri temsil eden bu noktalar, çeşitli analizler ve uygulamalar için temel oluşturur. Örneğin, ağız köşeleri olan 9. ve 10. noktalar arasındaki mesafe, bir kişinin gülümsüyor olup olmadığını belirlemek için kullanılabilir. Aynı şekilde, gözlerin konumu ve burunun ucu gibi diğer noktalar, yüz ifadelerini tanımlamak ve anlamak için kullanılabilir. Bu "face landmark" noktaları, bilgisayarlı görü ve yapay zeka uygulamalarında yüz tabanlı etkileşimleri izlemek, yüz ifadelerini analiz etmek ve yüz animasyonları oluşturmak gibi birçok uygulama için hayati öneme sahiptir.

Yüz izleme teknolojisi, genellikle kamera görüntülerini işleyerek bu "face landmark" noktalarını belirler. Bu, yüz analitiği, duygu analizi ve yüz tabanlı etkileşimlerin geliştirilmesi gibi birçok alanda kullanılabilir. Yüz izleme, bilgisayarlı görü ve yapay zeka alanlarında önemli bir araştırma ve uygulama konusu olmuştur, çünkü yüz ifadelerini anlama, tanıma ve yorumlama yeteneği birçok uygulama için önemlidir. [16]

2.2.4.8 PyTorch

Şekil 2.4 PyTorch

PyTorch, derin öğrenme uygulamaları geliştirmek için kullanılan açık kaynaklı bir makine öğrenimi kütüphanesidir. PyTorch, araştırmacılara ve geliştiricilere esneklik, kontrol ve geniş bir topluluk desteği sunar.
PyTorch'un bazı temel özellikleri:
Dinamik Hesaplama Grafiği: PyTorch, dinamik hesaplama grafiği kullanır. Bu, kullanıcıların çalışma zamanında grafiği oluşturmasına ve değiştirmesine olanak tanır. Özellikle araştırma ve prototip aşamalarında esneklik sağlar.
TorchScript: PyTorch, modellerinizi dışa aktarmak ve entegre etmek için kullanılan bir dil olan TorchScript'i destekler. Bu, PyTorch modellerini çeşitli uygulama ortamlarına (örneğin, mobil cihazlar veya gömülü sistemler) taşımak için önemlidir.
Geniş Model Koleksiyonu: PyTorch, önceden eğitilmiş modelleri içeren bir model koleksiyonuna sahiptir. Bu, kullanıcıların transfer öğrenme uygulamalarında hızlı bir şekilde başlamasına olanak tanır.
TorchHub: TorchHub, kullanıcıların modelleri paylaşmasını ve başkalarının modellerini kullanmasını sağlayan bir platformdur. Bu, topluluk tarafından oluşturulan modellere erişimi kolaylaştırır. [17]

2.2.4.9 Numpy

Numpy, Python programlama dilinde kullanılan güçlü bir sayısal hesaplama kütüphanesidir. Temel olarak çok boyutlu dizilerle çalışmayı sağlayan Numpy, bilimsel ve matematiksel hesaplamalar için optimize edilmiş bir dizi işlev ve araç sunar. İşitme engelli bireylerle etkileşimli sistemler veya işaret dili projeleri gibi uygulamalarda, Numpy kütüphanesi, veri manipülasyonu, analizi ve işlemenin yanı sıra sayısal hesaplamalar için güçlü bir araç olarak öne çıkar.

Numpy kütüphanesinin temel özellikleri şunlardır:

Çok Boyutlu Diziler (Arrays): Numpy, homojen veri tipleriyle çalışan çok boyutlu dizileri destekler. Bu, büyük veri setlerinin hızlı ve etkili bir şekilde işlenmesine olanak tanır.

Hızlı Matematiksel İşlemler: Numpy, vektör ve matris işlemleri gibi matematiksel operasyonları hızlı bir şekilde gerçekleştirebilen özel uygulamalara sahiptir. Bu, işitme engelli bireylerle etkileşimli sistemlerde matematiksel operasyonların hızlı ve doğru bir şekilde yapılmasını sağlar.

Rastgele Sayı Üretimi: Numpy, rastgele sayı üretme işlevleri sunar. Bu, işaret dili projelerinde kullanılacak örnek veri setlerini oluşturmak veya test senaryolarını çeşitlendirmek için faydalıdır. [18]

BÖLÜM 3

3.1 Mantıksal Tasarım

Veri setlerindeki fotoğrafların net, düzenli, ayırt edilebilir vb. olması makine öğrenmesi için oldukça önemlidir. Modelin doğruluk oranının yüksek olması için birçok yöntem denenmiştir.

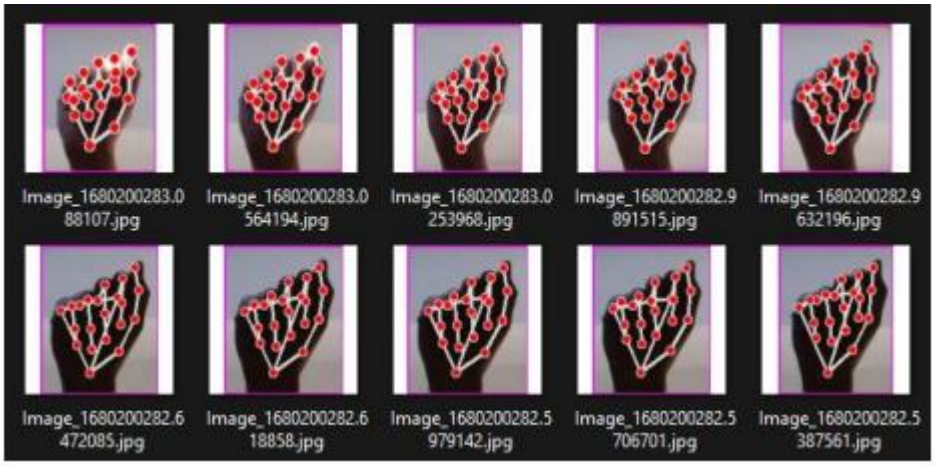
3.1.1 Denenen Veri Setleri

3.1.1.1 Kaggle Harf Veri Seti

Kaggle bünyesinde makine öğrenmesi, veri analizi, yapay zekâ, istatistik, veri görselleştirme, matematik bilimleri gibi birçok veri seti barındıran global çaplı bir web sitedir. İçerisinde kullanıcıların kendi paylaştığı veri kümeleri, bu verilerin analizini paylaştığı kaynak kodlar gibi içeriklere sahip. Kaggle içerisinde birçok veri seti ile karşılaşılmıştır. Bulduğumuz veri setlerinde genellikle harf başına düşen fotoğraf sayısı oldukça fazladır. Bunlardan birine örnek verecek olursak harf başına düşen 4000 adet fotoğraf bulunmaktadır. Bazı fotoğraf örnekleri Şekil 3.1'de ve Şekil 3.2'de gösterilmektedir.

Şekil 3.1 Kaggle Veri Seti

Fotoğrafların görüntü kalitesinin güzel olmaması, fotoğrafların karanlık olması, yapılan el işaretinin ayırt edici olmaması gibi sebeplerden dolayı MediaPipe kütüphanesi kullanılarak veri setine landmarklar eklenmiştir. Yapılan geliştirme sonucunda veri setinin daha doğru bir model oluşturması amaçlanmıştır.



Şekil 3.2 Landmark eklenmiş Kaggle Veri Seti

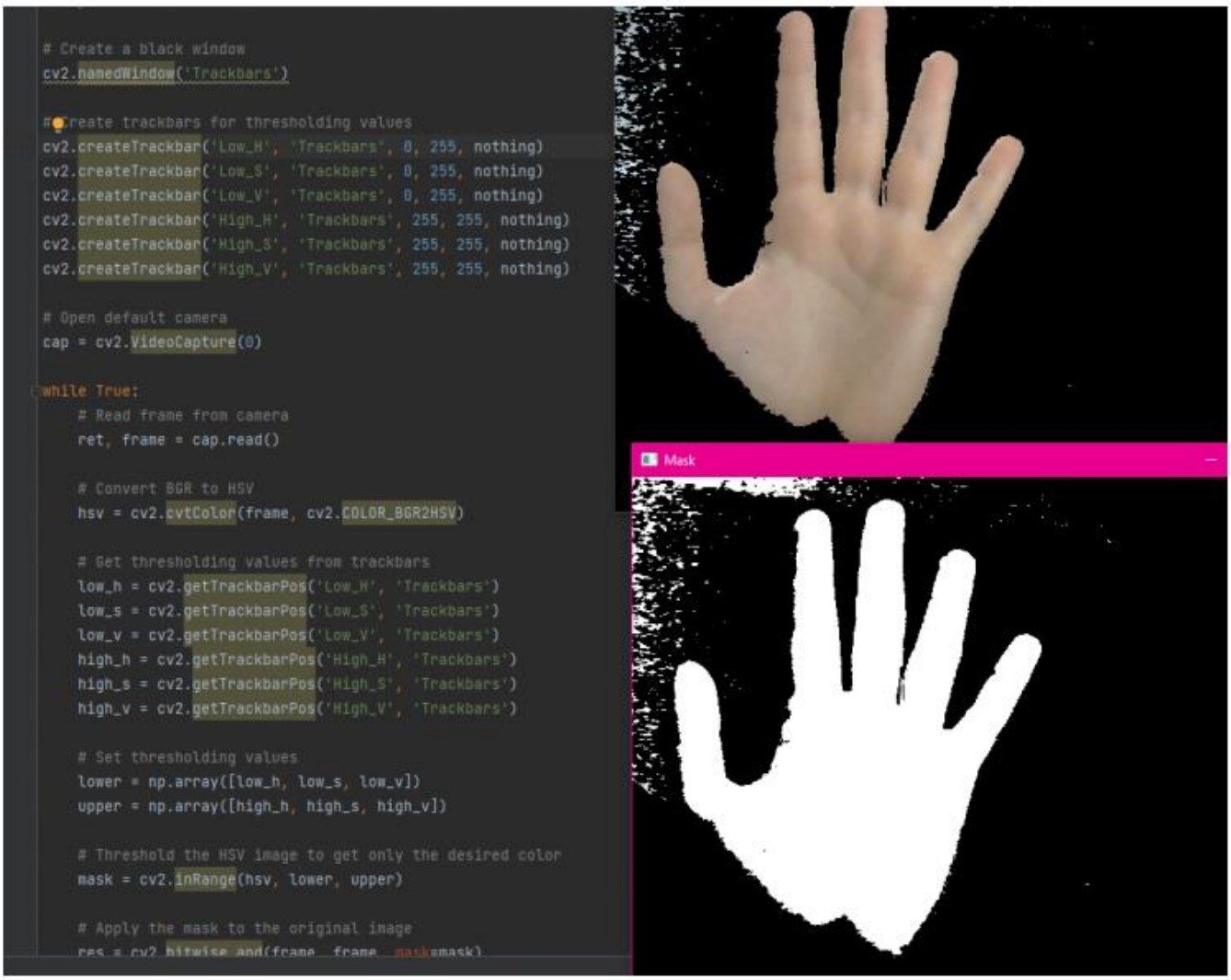
Yapılan tüm geliştirmelerin ardından landmark eklenmiş veri seti kullanılmıştır. Makine öğrenmesi, web tabanlı Teachable Machine kullanılarak gerçekleştirilmiştir. Oluşan model ile çeşitli testler yapılmıştır. Yapılan testlerin sonucunda doğruluk oranının oldukça düşük olduğu tespit edilmiştir. Bu yüzden farklı yöntemlere başvurulmuştur. Şekil 3.3'te Mediapipe kod örneği gösterilmektedir.

```
else:
    imgCrop = res[y - offset:y + h + offset, x - offset:x + w + offset]
    aspectRatio = h / w
    if aspectRatio > 1:
        k = imgSize / h
        wCal = math.ceil(k * w)
        imgResize = cv2.resize(imgCrop, (wCal, imgSize))
        wGap = math.ceil((300 - wCal) / 2)
        imgWhite[:, wGap:wCal + wGap] = imgResize
        f = io.StringIO()
        with redirect_stdout(f):
            prediction, index = classifier.getPrediction(imgWhite)
        # print(prediction,index)
        # print(labels[index])
        harf_ekle(labels[index])
        tahmin_et()

    else:
        k = imgSize / w
        hCal = math.ceil(k * h)
        imgResize = cv2.resize(imgCrop, (imgSize, hCal))
        hGap = math.ceil((300 - hCal) / 2)
        imgWhite[hGap:hCal + hGap, :] = imgResize
        f = io.StringIO()
        with redirect_stdout(f):
            prediction, index = classifier.getPrediction(imgWhite)
        # print(prediction,index)
        # print(labels[index])
        harf_ekle(labels[index])
        tahmin_et()

    cv2.imshow("ImageCrop", imgCrop)
    cv2.imshow("ImageWhite", imgWhite)
```

Şekil 3.3 Mediapipe'tan yararlanarak eli "z" indeksi fark etmeksizin 300x300'e yeniden şekillendiren ve eğitilen modele tahminde bulunan kod

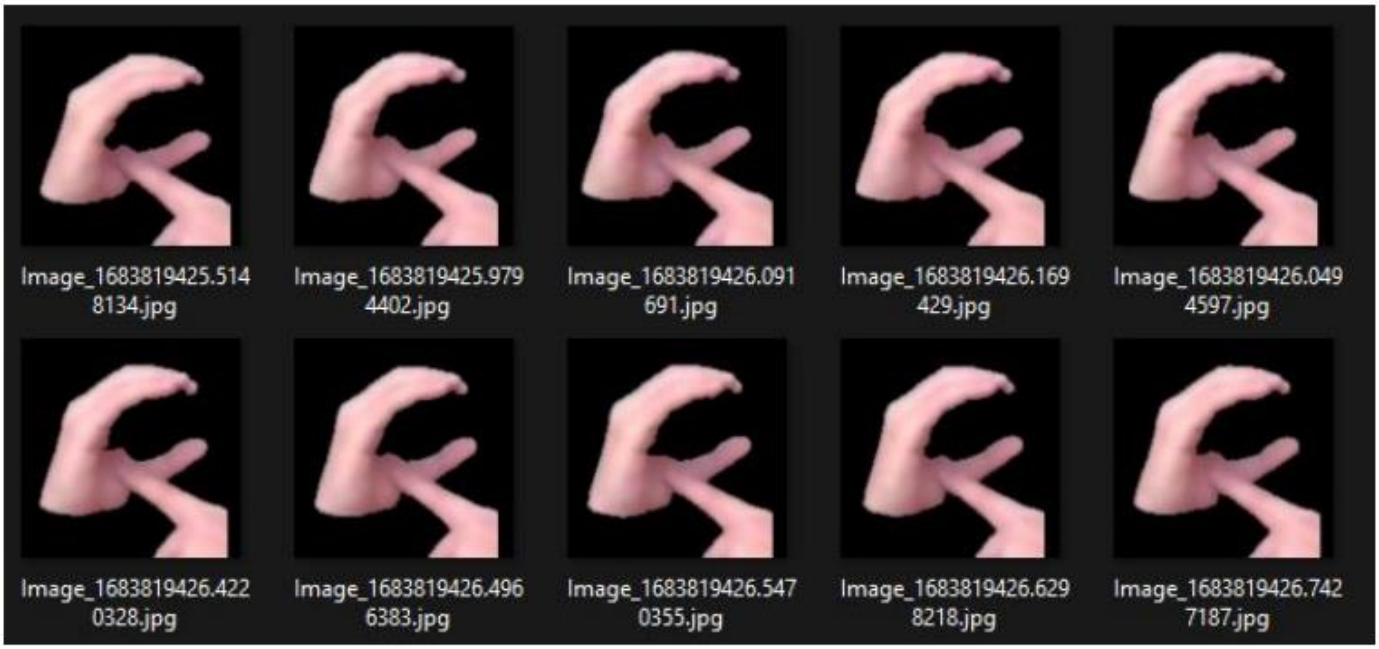
3.1.1.2 Maskeleme Yöntemiyle Oluşturulan Veri Seti

Modelin doğruluk oranının daha iyi olması için veri setinin tekrar oluşturulmasına karar verilmiştir. Veri seti fotoğrafları çekilirken renk ve maskeleme ayarlarını değiştirilmesini sağlayan algoritma eklenmiştir. Bu sayede eli odak noktası haline getirerek maskeleme yapılmıştır. Elin arka planı siyah gözükecek şekilde fotoğraflar çekilmiştir. Bu kod ve ekran çıktısı Şekil 3.4'te gösterilmektedir. El işaretlerinin renk ve arka plana bağlı olmadan makine öğrenmesi hedeflenmiştir.




```
# Create a black window
cv2.namedWindow('Trackbars')

# Create trackbars for thresholding values
cv2.createTrackbar('Low_H', 'Trackbars', 0, 255, nothing)
cv2.createTrackbar('Low_S', 'Trackbars', 0, 255, nothing)
cv2.createTrackbar('Low_V', 'Trackbars', 0, 255, nothing)
cv2.createTrackbar('High_H', 'Trackbars', 255, 255, nothing)
cv2.createTrackbar('High_S', 'Trackbars', 255, 255, nothing)
cv2.createTrackbar('High_V', 'Trackbars', 255, 255, nothing)

# Open default camera
cap = cv2.VideoCapture(0)

while True:
    # Read frame from camera
    ret, frame = cap.read()

    # Convert BGR to HSV
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Get thresholding values from trackbars
    low_h = cv2.getTrackbarPos('Low_H', 'Trackbars')
    low_s = cv2.getTrackbarPos('Low_S', 'Trackbars')
    low_v = cv2.getTrackbarPos('Low_V', 'Trackbars')
    high_h = cv2.getTrackbarPos('High_H', 'Trackbars')
    high_s = cv2.getTrackbarPos('High_S', 'Trackbars')
    high_v = cv2.getTrackbarPos('High_V', 'Trackbars')

    # Set thresholding values
    lower = np.array([low_h, low_s, low_v])
    upper = np.array([high_h, high_s, high_v])

    # Threshold the HSV image to get only the desired color
    mask = cv2.inRange(hsv, lower, upper)

    # Apply the mask to the original image
    res = cv2.bitwise_and(frame, frame, mask=mask)
```

Şekil 3.4 Maskeleme yöntemiyle fotoğrafların çekilmesini sağlayan algoritma

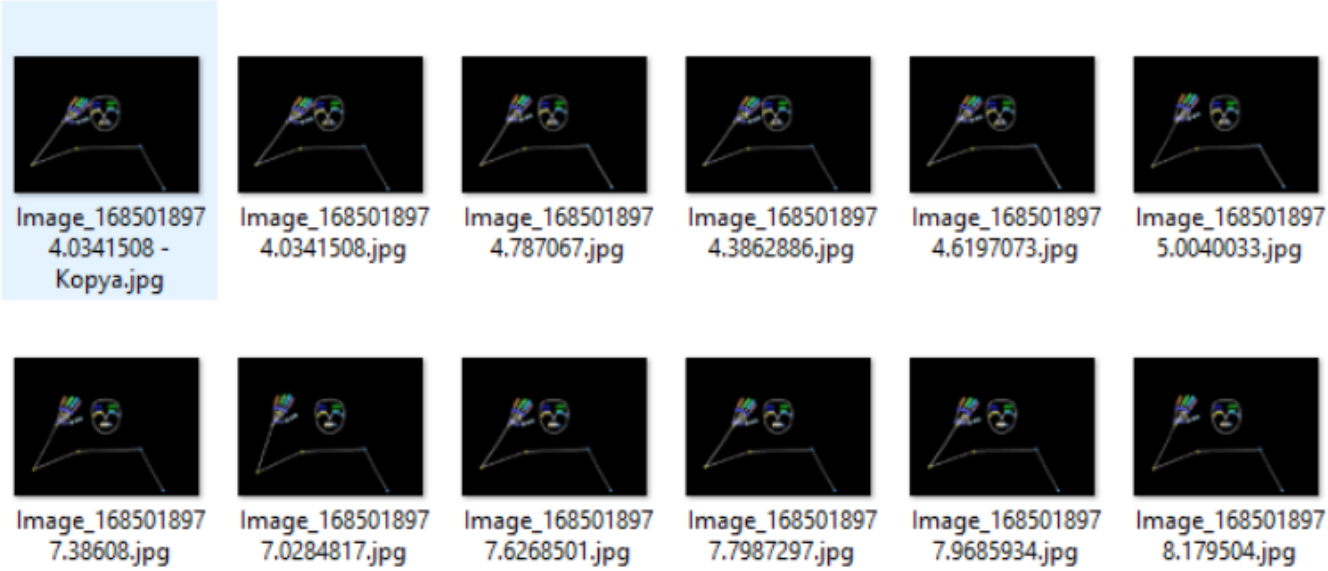


Şekil 3.5 Maskeleme yöntemiyle oluşturulan veri seti

Maskeleme yöntemiyle oluşturulan veri seti kullanılarak Teachable Machine üzerinden eğitim modeli oluşturulmuştur. Veri seti Şekil 3.5 'te gösterilmektedir. Oluşturulan model koda entegre edildikten sonra testler yapılmıştır. Yapılan testler sonucunda uygulamanın yaptığı tahminlerin çok fazla yanılması sonucunda modelin doğruluk oranının yeterli olmadığına karar verilmiştir.

3.1.1.3 Vücut Hatlarından Yararlanılarak Oluşturulan Veri Seti

Vücut hatlarından yararlanmak ve arka plan, ışık vb. dış etkenlerden etkilenmemesi için MediaPipe kütüphanesi kullanılmıştır. Siyah bir arka plan üzerine eklenen landmarkların gözükeceği şekilde veri seti oluşturulmuştur. Veri seti Şekil 3.6'da gösterilmektedir. Sadece vücut hatları ve ellerin gözüktüğü landmarkların daha net ve ayırt edilebilir gözüktüğünden doğruluk oranı arttırılması hedeflenmiştir.



Şekil 3.6 Vücut hatlarından yararlanarak oluşturulan veri seti

Oluşturulan veri seti ile web tabanlı Teachable Machine üzerinden eğitim modeli oluşturulmuştur. Modeli kendi oluşturduğumuz koda entegre ettikten sonra çeşitli testler yapılmıştır. Yapılan testlerin sonucunda tahminlerin doğruluk oranının oldukça düşük olduğu tespit edilmişti

BÖLÜM 4

GELİŞTİRME

İlk olarak gerekli olan kütüphaneler indirildi. Python versiyonunda hata alındı. Bunun için gerekli versiyon güncellendi (Python 3.8.10). Daha sonra görüntü işleme ile ilgili denemeler yapılarak kütüphanelerin çalışması denendi ve proje ile ilgili fikir oluşturuldu.

Bu denemelerde OpenCv(), MediaPipe() kütüphanelerinden yararlanıldı. Kamera açtırarak elin algılanması ve elin durumuna göre (açık-kapalı, olumlu-olumsuz) ekrana çıktı oluşturan kodlar yazıldı. [19]

Elin açık kapalı durumda olduğunu tespit eden kod Şekil 4.1'de gösterilmektedir.

acık_kapalı.py > ...

Click here to ask Blackbox to help you code faster

```python
# el açık yada kapalı iken ekrana yazdırır
import cv2
import mediapipe as mp

mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles
mp_hands = mp.solutions.hands

# For static images:
IMAGE_FILES = []
with mp_hands.Hands(
    static_image_mode=True,
    max_num_hands=2,
    min_detection_confidence=0.5) as hands:
  for idx, file in enumerate(IMAGE_FILES):
    image = cv2.flip(cv2.imread(file), 1)
    results = hands.process(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

    print('Handedness:', results.multi_handedness)
    if not results.multi_hand_landmarks:
      continue
    image_height, image_width, _ = image.shape
    annotated_image = image.copy()
    for hand_landmarks in results.multi_hand_landmarks:
      print('hand_landmarks:', hand_landmarks)
      print(
          f'Index finger tip coordinates: (',
          f'{hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_TIP].x * image_width}, '
          f'{hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_TIP].y * image_height})'
      )
```

Şekil 4.1.a Elin açık kapalı durumda olduğunu tespit eden kod

```
      mp_drawing.draw_landmarks(
          annotated_image,
          hand_landmarks,
          mp_hands.HAND_CONNECTIONS,
          mp_drawing_styles.get_default_hand_landmarks_style(),
          mp_drawing_styles.get_default_hand_connections_style())
    cv2.imwrite(
        '/tmp/annotated_image' + str(idx) + '.png', cv2.flip(annotated_image, 1))

# For webcam input:
cap = cv2.VideoCapture(0)
with mp_hands.Hands(
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as hands:
  while cap.isOpened():
    success, image = cap.read()
    if not success:
      print("Ignoring empty camera frame.")
      continue

    image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
    image.flags.writeable = False
    results = hands.process(image)
    image.flags.writeable = True
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

Şekil 4.1.b Elin açık kapalı durumda olduğunu tespit eden kod

```
    if results.multi_hand_landmarks:
      for hand_landmarks in results.multi_hand_landmarks:
        ## el açık ve kapalı iken ekrana yazdırma
        x, y = hand_landmarks.landmark[9].x, hand_landmarks.landmark[9].y
        x1, y1 = hand_landmarks.landmark[12].x, hand_landmarks.landmark[12].y
        font = cv2.FONT_HERSHEY_PLAIN
        if y1 > y:
            cv2.putText(image, "KAPALI", (10, 50), font, 4, (0, 0, 0), 3)
        else:
            cv2.putText(image, "ACIK", (10, 50), font, 4, (0, 0, 0), 3)
        mp_drawing.draw_landmarks(
            image,
            hand_landmarks,
            mp_hands.HAND_CONNECTIONS,
            mp_drawing_styles.get_default_hand_landmarks_style(),
            mp_drawing_styles.get_default_hand_connections_style())
    cv2.imshow('MediaPipe Hands', image)
    if cv2.waitKey(5) & 0xFF == 27:
      break
cap.release()
```

Şekil 4.1.c Elin açık kapalı durumda olduğunu tespit eden kod

Elin olumlu olumsuz hareket yaptığını tespit eden kod Şekil 4.2'de gösterilmektedir.

olumlu_olumsuz.py > ...

Click here to ask Blackbox to help you code faster

```python
# el açık yada kapalı iken ekrana yazdırır
import cv2
import mediapipe as mp
mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles
mp_hands = mp.solutions.hands

# For static images:
IMAGE_FILES = []
with mp_hands.Hands(
    static_image_mode=True,
    max_num_hands=2,
    min_detection_confidence=0.5) as hands:
  for idx, file in enumerate(IMAGE_FILES):
    image = cv2.flip(cv2.imread(file), 1)
    results = hands.process(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

    print('Handedness:', results.multi_handedness)
    if not results.multi_hand_landmarks:
      continue
    image_height, image_width, _ = image.shape
    annotated_image = image.copy()
    for hand_landmarks in results.multi_hand_landmarks:
      print('hand_landmarks:', hand_landmarks)
      print(
          f'Index finger tip coordinates: (',
          f'{hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_TIP].x * image_width}, '
          f'{hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_TIP].y * image_height})'
      )
```

Şekil 4.2.a Elin olumsuz hareket yaptığını tespit eden kod

```
30            mp_drawing.draw_landmarks(
31                annotated_image,
32                hand_landmarks,
33                mp_hands.HAND_CONNECTIONS,
34                mp_drawing_styles.get_default_hand_landmarks_style(),
35                mp_drawing_styles.get_default_hand_connections_style())
36        cv2.imwrite(
37            '/tmp/annotated_image' + str(idx) + '.png', cv2.flip(annotated_image, 1))
38
39    # For webcam input:
40    cap = cv2.VideoCapture(0)
41    with mp_hands.Hands(
42        min_detection_confidence=0.5,
43        min_tracking_confidence=0.5) as hands:
44      while cap.isOpened():
45        success, image = cap.read()
46        if not success:
47          print("Ignoring empty camera frame.")
48          continue
49
50        image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
51        image.flags.writeable = False
52        results = hands.process(image)
53
54        image.flags.writeable = True
55        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
56        if results.multi_hand_landmarks:
57          for hand_landmarks in results.multi_hand_landmarks:
58
```

Şekil 4.2.b Elin olumsuz hareket yaptığını tespit eden kod

```
58
59          ## el açık ve kapalı iken ekrana yazdırma
60          x, y = hand_landmarks.landmark[0].x, hand_landmarks.landmark[0].y
61          x1, y1 = hand_landmarks.landmark[4].x, hand_landmarks.landmark[4].y
62
63          font = cv2.FONT_HERSHEY_PLAIN
64
65          if y1 > y:
66              cv2.putText(image, "OLUMSUZ", (10, 50), font, 4, (0, 0, 0), 3)
67          else:
68              cv2.putText(image, "OLUMLU", (10, 50), font, 4, (0, 0, 0), 3)
69
70          mp_drawing.draw_landmarks(
71              image,
72              hand_landmarks,
73              mp_hands.HAND_CONNECTIONS,
74              mp_drawing_styles.get_default_hand_landmarks_style(),
75              mp_drawing_styles.get_default_hand_connections_style())
76      cv2.imshow('MediaPipe Hands', image)
77      if cv2.waitKey(5) & 0xFF == 27:
78        break
79  cap.release()
```

Şekil 4.2.c Elin olumsuz hareket yaptığını tespit eden kod

BÖLÜM 5

SONUÇ

İşitme engelli bireylerin yaşadıkları zorluklar bilinmekle birlikte daha iyi anlaşılmıştır. İşaret dili sadece engelli bireyler tarafında değil tüm toplum tarafından bilinmesi gereken bir dildir. Çünkü iletişim tek taraflı değildir.

Bu projeye, yaşanılan bu iletişim sorununu en aza indirmek amacıyla başlanılmıştır. Araştırmalarımız sonucu edindiğimiz bilgiler doğrultusunda veri seti araştırmalarımız başlamıştır. Bulunan veriler projemizin kapsamı doğrultusunda elenmiştir.

Bu zaman içerisinde kod yazabileceğimiz bilgiye ulaşıldı. Projede kullanmamız gereken görüntü işleme yöntemleri ile ilgili küçük projeler yapıldı. Bunun için Python dilinden; OpenCv() ve MediaPipe() kütüphanelerinden yararlanıldı.

Böylece projemizi geliştirmek için yeterli donanıma sahip olduk.

KAYNAKLAR

[1] https://tr.wikipedia.org/wiki/%C4%B0%C5%9Faret_dili

[2] Karaca, M. F. (2018). Üç Boyutlu Sanal Model ile Türk İşaret Dili Simülasyonu, Doktora Tezi.

[3] Aksoy, B., Ghazal, Z., Şenol, R., Ersoy, M. (2020). Ses ve Metin Olarak Girilen İşaret Dili Hareketlerinin Robot Kol Tarafından Gerçekleştirilmesi. Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 8(1), 220-232.

[4] https://www.cmp.boun.edu.tr/tid/

[5] https://isaretce.com

[6]https://tr.linkedin.com/posts/ertugrulkusva_mediapipe-opencv-keras-activity-6836987838622396417-s1sj?trk=public_profile_like_view

[7]https://www.isaret.com

[8] https://learn.microsoft.com/en-us/dotnet/csharp/

[9] https://tr.wikipedia.org/wiki/Java

[10] https://www.python.org/

[11] https://tr.wikipedia.org/wiki/OpenCV

[12] https://en.wikipedia.org/wiki/TensorFlow

[13] https://pypi.org/project/imutils/

[14] https://pypi.org/project/keras/

[15] https://en.wikipedia.org/wiki/Matplotlib

[16] https://developers.google.com/mediapipe

[17] https://pypi.org/project/torch/

[18] https://pypi.org/project/numpy/

[19] https://github.com/bulentsezen/el-izleme/tree/main

ÖZGEÇMİŞ

Ad Soyad : Ozan Can SARI

Doğum Yeri :Merkez/Diyarbakır

Doğum Yılı :22.01.2001

Lise :2016-2020

Staj Yaptığı Yerler: NTT DATA Business Solutions Turkey

E-posta : 7ozzy35@gmail.com

Ad Soyad : Buse UYSAL

Doğum Yeri : Ezine/Çanakkale

Doğum Yılı : 20.09.2001

Lise : 2016-2020

Staj Yaptığı Yerler: Egemen Yazılım ve Otomasyon San. ve Tic. Ltd. Şti.

E-posta : buseuysal3158@gmail.com

Ad Soyad : Melisa VASİJA

Doğum Yeri : Prizren/Kosova

Doğum Yılı : 21.09.2001

Lise : 2017-2020

Staj Yaptığı Yerler: Best Vision Balkan

E-posta : vasijamelisa@gmail.com

Ad Soyad : Anal KARAALI

Doğum Yeri :Merkez/Tekirdağ

Doğum Yılı :28.05.2001

Lise :2015-2019

E-posta : anil.karaali59@gmail.com