T.C.
AKDENİZ ÜNİVERSİTESİ

Elektrik-Elektronik Mühendisliği Bölümü

BİTİRME ÇALIŞMASI

Proje İsmi

PCR NUMUNE ROBOTU

Öğrenci İsmi

MAHMUT ŞATIR

Danışman
Prof. Dr. HÜSEYİN GÖKSU

Haziran, 2021

Bu çalışma …. / …. / 2021 tarihinde aşağıdaki jüri tarafından Elektrik-Elektronik Mühendisliği Bölümü'nde Lisans Bitirme Projesi olarak kabul edilmiştir.

Bitirme Projesi Jürisi

Danışman Adı	Prof. Dr. Hüseyin GÖKSU	
Üniversite	Akdeniz Üniversitesi	
Fakülte	Mühendislik Fakültesi	

Jüri Üyesi		
Üniversite	Akdeniz Üniversitesi	
Fakülte	Mühendislik Fakültesi	

Jüri Üyesi		
Üniversite	Akdeniz Üniversitesi	
Fakülte	Mühendislik Fakültesi	

ÖZET

Hareketli nesne takibi, uluslararası savunma alanında kullanılan, bilinen bir yöntemdir. Günümüzde bir çok hava, deniz, kara araçlarında kullanılarak gerek hedef takibi gerekse sınır güvenliği gibi bir çok alanda kullanılmıştır. Ülkeler bunun sayesinde sınırlarının bütünlüğünü koruyabiliyor ve sistemin avantajlarından yararlanabiliyor. Gelişen teknoloji ile birlikte modern cihazların sağlık alanında kullanımı artmıştır. Ve birçok tıbbi teknik bu yöntem ile geliştirilmiştir. Bu teknikler hekimlere zaman, maliyet, tedavi ve teşhis konusunda büyük kolaylık sağlamaktadır. Herkesin bildiği gibi günümüzde covid-19 sebebi ile hekimler ve diğer bütün sağlık çalışanları bu konuda zor günler geçirmekte. Gözlemlerime dayanarak ise öncelikle bir covid-19 testinin yapılışında bir doktorun en çok zorlandığı kısmın numune alırken yaşadığı yakınlık ve risk anıdır. Ve robot tarafından gerçekleşecek bir test ve numuneyi o teste girmesi için alacak ve cihaza konumlandırabilecek bir robot büyük bir amaca hizmet edebilecektir.

Görüntü işleme günümüzde birçok alanda bize hizmetler sunmaktadır. Bize sunduğu bu hizmetler doğrusunda farklı alanlarda kullanılan işlevselliğiyle bize birşeyler katmakta ve birçok algoritma kullanıp karar verebileceğimiz konularda anlık olarak sensörlere ihtiyaç duymadan görüntüyü tanıyarak bir karar mekanizması ile bize kolaylık imkanını sunar. Bu proje de bize bu imkanlar doğrultusunda attığımız adımlar ile bir kamera ve mekanik dört eksenli bir hareket kabiliyetine sahip bir kol ve onun hareket desteğini sağlayan motorlar ile insan burnuna otonom olarak girip alması gereken örneği (sürüntü) alıp çıkması ile hasta ile çalışan sağlık personeli arasında ki teması kesmeye yönelik çalışma olmaktadır.

TEŞEKKÜR

Çalışmam da yardımlarından dolayı, bilgi ve birikimlerini benimle paylaşan Erkut Savunma Sanayi bünyesinde çalışmakta olan Elektrik Elektronik Mühendisi Hasan Can'a her zaman bana güvenen, yanımda olan aileme, çalışma süresince büyük ya da küçük yardımı geçen herkese ve çalışmamın her aşamasında yardımlarını ve imkanlarını benden esirgemeyen danışman hocam Sayın Prof. Dr. Hüseyin GÖKSU' ya teşekkürlerimi sunarım.

İÇİNDEKİLER

GİRİŞ

Bilgisayarın yazıcısı yada mutfak robotunuz gerçekten birer robot mudur? Bir makineye robot diyebilmek için, en önemli koşullardan biri algılamadır. Bir robot az veya çok dış dünyadan bir algılama yapabilmelidir. Bu algılamalar sensörler sayesinde olur. Isı, ışık, şekil, dokunma gibi olabilir. Daha sonra bu bilgileri otonom olarak yorumlamak, algıya ne gibi tepkide bulunacağına karar vermelidir. Son olarak ta robot verdiği kararı uygulamaya koyabilmelidir. Özetlersek robot 3 ana kısımdan oluşur. Buna göre bir robotta; çevre hakkında gerçek zamanlı bilgi edinmek için kullanılan sensörler, karar vermeyi sağlayan mikro işlemci verilen kararların uygulanmasını sağlayan eyleyiciler ve hareket sistemleri bulunur .,

'PCR Numune Robotu' adlı projemizin amacı dışarıdan alınan görüntüler ile hareket halinde olan bir insan yüzü algılanırsa bu yüzü kare içine alarak konumunu ve hareket yönünü takip etmektir. Takip sonrası ise yüzün merkez konumu burun olacak şekilde insan yüzünü ortalama için aldığı merkeze uzaklık derecelerine göre Mcu ya bilgi gönderilip sürekli merkez konumda kalması sağlanılmıştır. Sonrasında ise butonlar yardımı ile ileri geri hareket sağlanıp çubuk (örnek alma çubuğu) burnun içinde ilerlemesi sürülmesi sağlanılmak istenmiştir. Algoritmalar öncelikle Python programla dili tercih edilerek open cv kütüphanesi ile görüntü işlenilip Arduino uno ile haberleşmesi sağlanıp fiziksel hareket sağlanılması amaçlanmıştır.

KAYNAK TARAMASI

Karabük Üniversitesi Mühendislik Fakültesi Mekatronik Mühendisliği Cankat BİLEK' in bitirme tezi yüz takip sistemi adlı çalışmasından yüz algılama hakkında bilgi toplanmıştır.

Kocaeli Üniversitesi Mühendislik Fakültesi Enes Baytürk Lisans Tezi Derin Öğrenme Uygulaması adlı çalışmasından derin öğrenme hakkında bilgi toplanmıştır.

Sakarya Üniversitesi Mühendislik Fakültesi Burak YALIM , Yunus Emre SİNEKOĞLU , Alp Tuğrul KURT teknoloji fakültesi Eem tasarımı raporu adlı çalışmadan eksen robot kol tasarımı hakkında bilgi toplanmıştır.

Yildiz Teknik Üniversitesi Fen Bilimleri Enstitüsü Makina Mühendisi Volkan İZGİ Alti Eksenli Endüstriyel Robot Tasarımı adlı yüksek lisans tezi çalışmasından robot kol mekaniği hakkında bilgi toplanmıştır.

1 TEMEL BILGILER

Temel olarak bir robotun aşağıdaki özelliklerinin olması gerekir:

İşlem Yapma Yetisi: Bir işlemi fiziksel yada farazi olarak yerine getirebilmelidir, yoksa robot olmaz sadece bir madde olur. İşlemin Sonucunu Belirleme Yetisi: İşlemi yaptıktan sonra mutlak olarak işlemin sonucunu belirlemelidir ki işlem tam olarak yapılmış olsun.

Karar Verme Yetisi: İşlem sonucuna göre yada dış etmenlere göre mutlaka bir yargı kurabilmelidir.

Bu yapıları bünyesinde barındıran bir sisteme genel olarak ROBOT adını verebiliriz. Fakat asıl robot kavramı bu yapıların çok daha ilerisine giderek doğada en karmaşık olan insanoğlunun yetilerini taklit etmek amacıyla yapılan makinelerdir. Robot kavramı da onlar üzerine kurulmuş olmasına rağmen tanım genel olarak takdire dayanan yapıları da içermektedir (Endüstriyel robotlar ve uygulama alanları, b.t).

Robotik teknolojilerin güncel alt basamaklarından biri de robot kollardır. Günümüzde hemen hemen her farbrikanın ürtüm bandından, robot kolları yerini almış ve gelecek zamanlar da daha fazla kullanım olanağı bulunacağı tartışılmaz bir gerçektir. Işte bu çalışmaların ve anaizlerin doğrultusunda bu tez çalışmasında robot kolları teorik ve mekanik olarak incelenmiş matematiksel çözümlemeleri hesaplanmoştır. Teorik çalışma deneysel ve uygulamasal olarak geliştirilip prototip olarak test edilmiştir. Dört eksenli robot kol teorik ve uygulama bölümlerinde en uygun robot kol olarak belirlenmiştir. Gerek hızı herek ise hassasiyeti noktasında uygulamalar için yeterli verimi sağlamaktadır.

1.1 Literatür Çalışması

Robotlar ve mekatronik alanında birçok çalışma yapılmıştır. Çalışmalar genellikle robot kolu üzerine yoğunlaşılmıştır. Tez konusu robot kolu ile ilgili olmasından dolayı benzeri bazı çalışmalara aşağıda değinilmiştir.

Daehie, Steven A. Velinsky ve KazuoYamazaki, uygulamalarında otobanların yapım ve bakım onarım işlerinde kullanılan bir mobil robotu ve bu robotun kontrol sistemini açıklamaktadırlar. Bu robotta Servo sistem kontrolü kullanılarak optimum kontrol gerçekleştirilmiştir (Daehie ve Steven, 1997).

İzmir Dokuz Eylül Üniversitesi'nde bir çalışmada internet üzerinden erişilebilecek mikrodenetleyici tabanlı bir elektronik kartın tasarlanması ve gerçeklenmesi yapılmıştır. Uygulama olarak, internet üzerinden robot kolu kontrolü başarıyla gerçekleştirilmiştir. (Yarım, 2004).

Yayınlanan bir makalede değişik nesneleri tanımaya yönelik görüntü işleme sistemi ile bu nesneleri görüntü destekli ayırmak için kullanılan robot manipülatörü ile ilgili çalışmalar sunulmuştur. DC motorların üç boyutlu uzayda verilen bir yörüngeyi takip edebilmesi için C++ ile özel bir mafsal kontrol algoritması yazılmıştır. Görüntü tanımayla bağlantılı olarak robot kolun senkron çalışması ve değişik yolları takip edebilme yeteneği, 24 tanımlı nesne

için test edilmiştir. Sonuç olarak, 5 ile 10 mm arasında bir kesinlik değeri ile yörünge takip edilebilmiş ve %95'lik bir nesne tanıma sonucuna ulaşılmıştır (Ayberk, 2001).
Başka bir yüksek lisans projesinde; yap ilan simülasyon çalışmaları sonucunda alt ı serbestlik dereceli PUMA 560 robotunun önceden hesaplanmış dinamik parametreler altında; PD kontrol algoritması kullanılarak, hesaplanmış moment yöntemi metodu ile yörünge kontrolü yapılmıştır. Zamana bağlı olarak eklemlerin konum ve hız eğrileri elde edilmiştir (Bostan, 2004)

Bir doktora tezinde ise üç eklemli bir robotik manipülatörün, görmeye dayalı kontrolü YSA kullanılarak yapılmıştır. Simülasyon programı kullanılmıştır (Köker, 2002)

1.2 Çalışma Takvimi

EKİM	**PROJE PLANLAMASI**
KASIM	**LİTERATÜR TARAMASI**
ARALIK	**GEREKLİ YAZILIM DİLİ BELİRLENMESİ**
OCAK	**PROTOTİP İÇİN GEREKLİ MEKANİK PLANLAMASI**
ŞUBAT	**YAZILIM GELİŞTİRİLMESİ**
MART	**MEKANİK PROTOTİP OLUŞTURULMASI**
NİSAN	**YAZILIM İLE MEKANİK BİRLEŞTİRİLMESİ**
MAYIS	**PROTOTİP DENEMELERİ**
HAZİRAN	**ROBOTUN ÇALIŞTIRILMASI VE TEZ YAZIMI**

Tablo 1 Çalışma Çizelgesi

2 TEORİK ALTYAPI

2.1 Genel Bilgiler

Robotik, Makine Mühendisliği , Bilgisayar Mühendisliiği ve Elektrik Elektronik Mühendisliği alanlarının ortak çalışma alanıdır. Robotlar da ; döner, silindirik, prizmatik , küresel, helisel ve düzlemsel eklemlerden biri kullanılır. Ve bu eklem türüne göre sınıflandırılırlar. Döner ve prizmatik eklemler robotikte en çok kullanılan eklem türleridir. Bir robotik, mekanik bölümler, kontrol birimleri ve hareketlendirici birimlerinden oluşmaktadır.
1: Mekanik Kısmı: Robotun iskelet kısmını oluşturur.

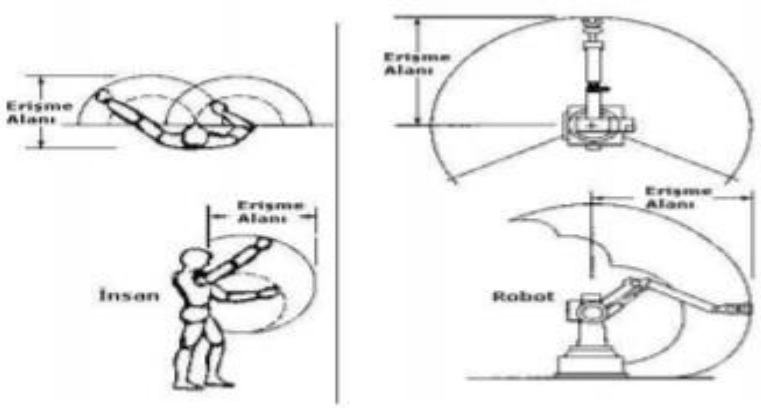
2:Tutaç: (EndEffector) : Bu eleman gerçek işi yapan kısımdır, robotun en uç noktasıdır ve uygulamada aktif olarak yer alır.

3: Motorlar: Eklemleri ve tutacı hareket ettirmek için kullanılır, en çok kullanılanları servo ve hidrolik motorlardır.

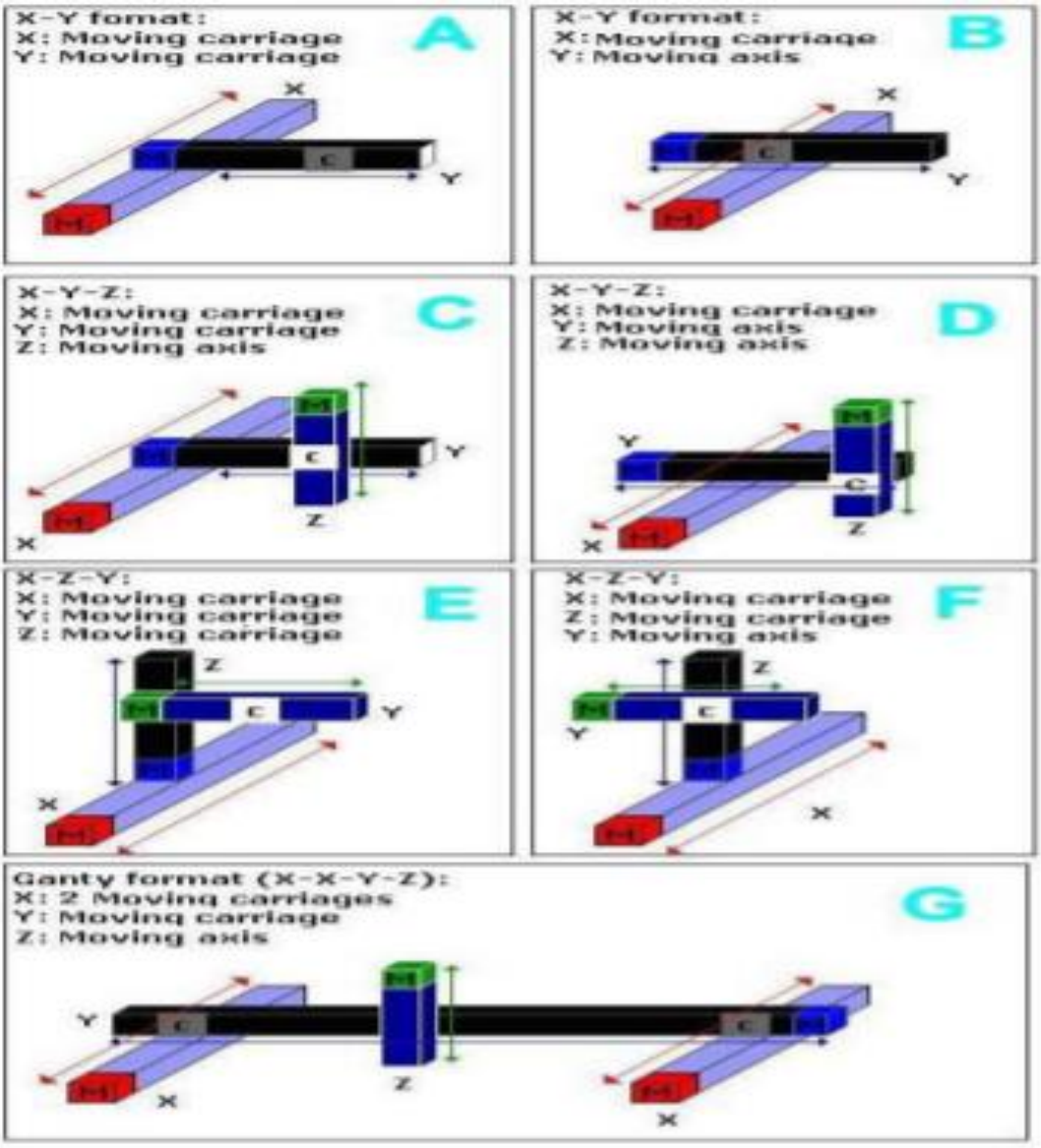
4: Kontroller: Girişi işleyip robotun yapması gereken görevini gerçekleştirir.

5: Sensörler: Kontrollere bağlıdır, robotun görevini yapması için robota geri dönüş ve giriş verisi sağlarlar. Her zaman gerekli değillerdir.

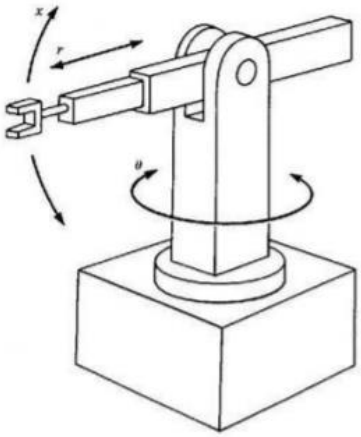
Manipülatörün kendi ekseni veya serbestlik derecesi diye tanımlanan değişik hareketleri vardır. Eğer bir manipülatör kendi ekseni etrafında dönüyorsa, bu robota “tek eksenli robot” denir. Eğer manipülatör yukarı ve aşağı doğru hareket ediyorsa, bu robota “çift eksenli robot” denir. Kendi ekseni etrafında dönen ve yukarı aşağı hareket eden manipülatör, yatay eksende ileri-geri hareket de edebilir. Bu robota “üç eksenli robot’’ denir. Endüstriyel robotlar en az üç eksene sahiptirler. Bu hareketler, kendi ekseni etrafında dönmesi, yukarı-aşağı ve ileri-geri hareket edebilmesidir.



Şekil 2.1 İnsan kolu ile mafsallı robotun çalışma alanları arasındaki benzerlik



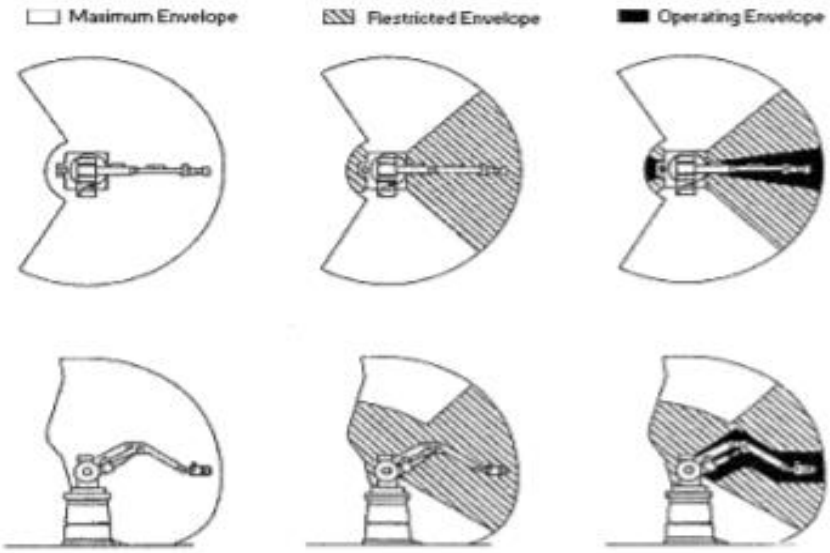
Şekil 2.2 Kartezyen Eksenler



2.2 Mekanik Bilgiler

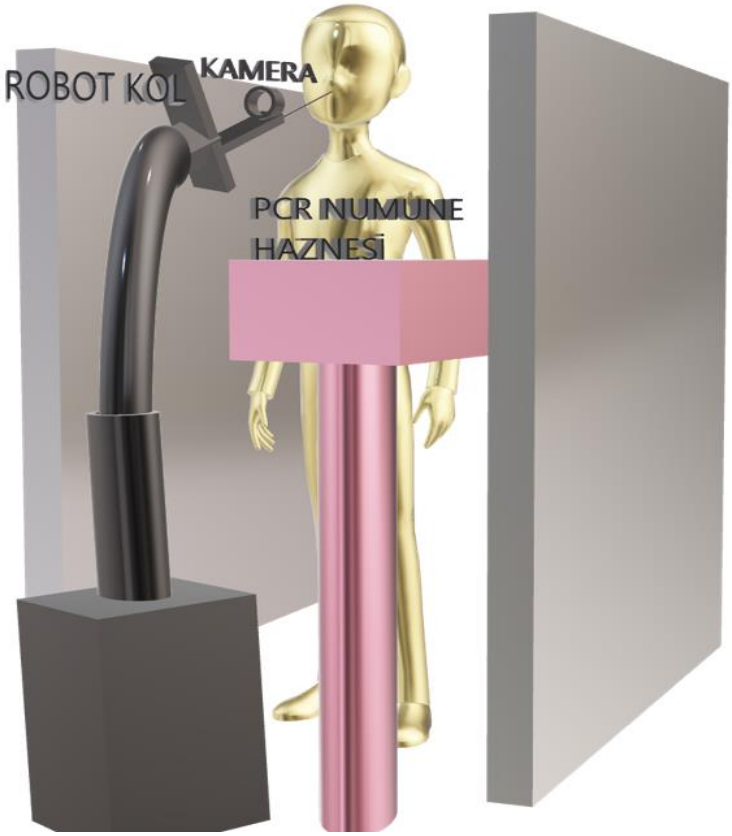
Bir robot bir iş yaparken kolu dairesel hareketli bağlamlarla oluşturuyorsa, bu tip robotlara Döner koordinat sistemli robotlar denir. Robot kolunun bağlantıları gövde üzerine, etrafında dönecek şekilde monte edilmiştir ve dayanak noktaları birbirine benzeyen iki ayrı bölümü taşır. Dönen parçalar yatay ve dikey monte edilebilir. Bu robotlar projede tercih edilmiştir.

Şekil 2.2.3 Robot Kol



Şekil 2.3 Robot Kolun Çalışma Alanı

360° dönme sağlanamaz ancak bu kayıplar minimuma indirilebilir. Şekil 2. 3'de döner koordinatlarda çalışma hacmi görülmektedir. Bu tip robotlarda robot kolun çalışması zor gözlenir. Çalışma hacmindeki noktalara farklı yörüngelerle ulaşılabilir. Buna göre sistem parametrelerinin en uygun olduğu yol seçilmelidir. Döner koordinatlı robotlarda kontrol işlemi karmaşıktır, dolayısıyla kontrol donanımının da bu karmaşıklığı karşılayabilecek kapasitede olması gerekir. Ayrıca bu tip robotlarda mafsallarda sızdırmazlı kolayca sağlanabilmektedir.

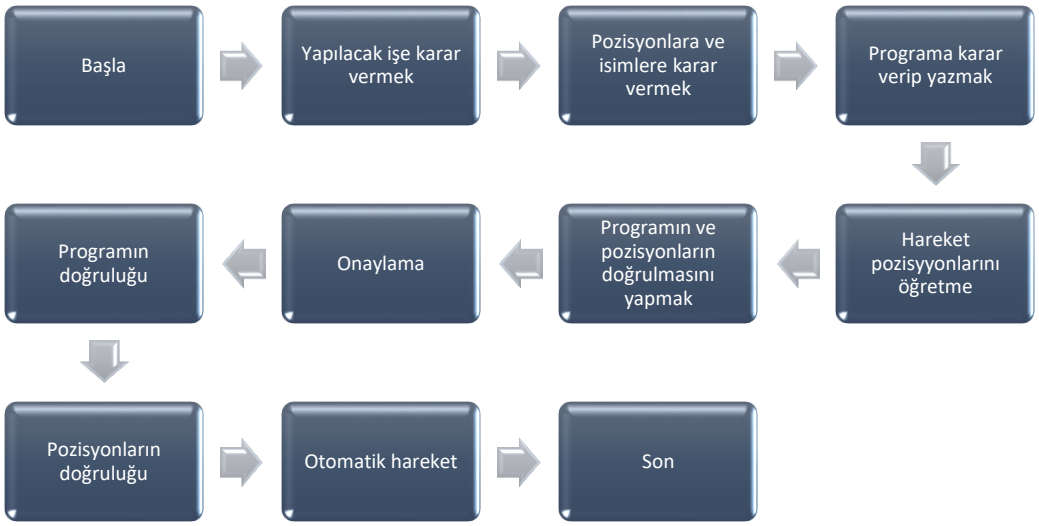


Şekil 2.3.2 Proje Animasyonu

2.3 Robot Programlama

Robotların programlanmasında iki farklı yöntem kullanılmaktadır. Bu yöntemler ; öğretim yöntemi ve bir ara yüz program ile programlanmaktadır.

2.3.1 Öğretim Yöntemi



Şekil 2.4 Öğretim Kutusu

Görüntü işleme isim (Almanca Bildbearbeitung) ölçülmüş veya kaydedilmiş olan elektronik (dijital) görüntü verilerini, elektronik ortamda (bilgisayar ve yazılımlar yardımı ile) amaca uygun şekilde değiştirmeye yönelik yapılan bilgisayar çalışması. Görüntü işleme, verilerin, yakalanıp ölçme ve değerlendirme işleminden sonra, başka bir aygıtta okunabilir bir biçime dönüştürülmesi ya da bir elektronik ortamdan başka bir elektronik ortama aktarmasına yönelik bir çalışma olan "Sinyal işlemeden" farklı bir işlemdir. Görüntü işleme, daha çok, kaydedilmiş olan, mevcut görüntüleri işlemek, yani mevcut resim ve grafikleri, değiştirmek, yabancılaştırmak ya da iyileştirmek için kullanılır.

Görüntü iyileştirme ve görüntü onarmadan farklı olarak görüntü bölütleme, görüntü analizi ile ilgili bir problem olup görüntü işlemenin gösterim ve tanılama aşamalarına görüntüyü hazırlama işlemidir. Bu anlamda görüntü bölütleme, bir görüntüyü her biri içerisinde farklı özelliklerin tutulduğu anlamlı bölgelere ayırmak olarak tarif edilebilir. Bu özellikler örneğin, görüntü içerisindeki benzer parlaklıklar olabilir ve bu parlaklıklar ilgili görüntünün farklı bölgelerindeki nesneleri temsil edebilir. Görüntü içerisinde aynı parlaklıklara sahip nesne parçacıklarının belirlenmesi, sınıflandırma ve tanılama amacı için kullanılabilir. Unutulmamalıdır ki, tüm görüntülere uygulanabilecek genel (universal) bir bölütleme yöntemi yoktur ve hiçbir bölütleme yöntemi mükemmel değildir. Başka bir deyişle, görüntü iyileştirme ve onarma problemlerinde olduğu gibi görüntü bölütleme için tasarlanan yöntemler ve bu yöntemlerin başarımları, görüntüden görüntüye ve uygulamaya dayalı olarak değişiklik arz eder. Genel olarak gri-ton görüntüler için bölütleme algoritmaları, gri seviye değerlerinin iki temel özelliğinden birine dayalı olarak tasarlanırlar. Bu özellikler, görüntü içerisindeki gri seviye değerlerindeki süreksizlik (discontinuity) ve benzerlik (similarity) ile ilgilidir.

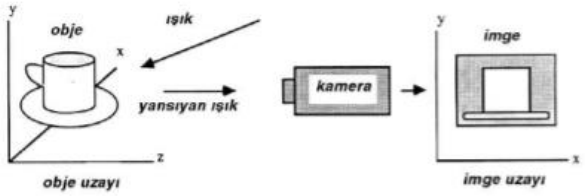
3 GÖRÜNTÜ İŞLEME

3.1. İmge Kavramı

İmge (image), çeşitli yollarda elde edilen bilgilerin görüntüsel olarak saklanmasına ve göstermesine olanak sağlayan resimlerdir. Her türlü iki boyutlu bilgi imge olarak ele alınabilir.

3.2 İmgenin Oluşturulması

İmgeler, 3-Boyutlu gerçek nesne uzayından sadece 2-Boyutun kullanımı ile oluşturulur. Kamera benzeri cihazların ışığa duyarlı 2-B yüzeyine nesneden yansıyan ışık kullanılarak imge uzayına geçilir.



Şekil 3.1 İmge oluşumu

3.3 Sayısal İmge Kavramı

İmge Modeli: İmge iki boyutlu ışık-yoğunluk fonksiyonu olarak adlandırılabilir ve f(x,y) ile gösterilir. (x,y) koordinatları belirtmekte, f(x,y) fonksiyonunun değeri ise (x,y) noktasındaki pikselin ışıklık (intesity) değerini ifade etmektedir.

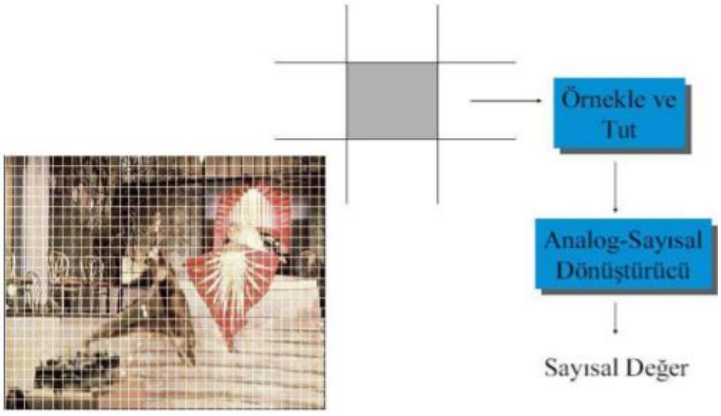
Şekil 3.1 İmge Modeli

3.4 Sayısal İmge :

Sayısala Dönüştürme İmgelerin sayısal imge işleme teknikleri ile işlenebilmesi analog biçimden sayısala çevrilmesi gerekmektedir. Bilindiği gibi analog bir işaretin sayısala çevrilmesi iki aşamada gerçekleşir.

►Örnekleme : İşaretten belirli zaman anlarında örnekler alınması anlamına gelir. İki boyutlu imgelerde uzumsal düzlemin sayısallaştırılmasına karşılık gelir.

►Nicemleme(Kuantalama) : Genlik seviyelerinin sadece belirli değerleri alması işlemidir. Genlik değerinin sayısallaştırılmasına karşılık gelir.



Şekil 3.3

3.5 OpenCV

OpenCV (Open Source Computer Vision) açık kaynak kodlu görüntü işleme kütüphanesidir. 1999 yılında İntel tarafından geliştirilmeye başlanmış daha sonra Itseez, Willow, Nvidia, AMD, Google gibi şirket ve toplulukların desteği ile gelişim süreci devam etmektedir. İlk sürüm olan OpenCV alfa 2000 yılında piyasaya çıkmıştır. İlk etapta C programlama dili ile geliştirilmeye başlanmış ve daha sonra birçok algoritması C++ dili ile geliştirilmiştir. Open source yani açık kaynak kodlu bir kütüphanedir ve BSD lisansı ile altında geliştirilmektedir. BSD lisansına sahip olması bu kütüphaneyi istediğiniz projede ücretsiz olarak kullanabileceğiniz anlamına gelmektedir. OpenCV platform bağımsız bir kütüphanedir, bu sayede Windows, Linux, FreeBSD, Android, Mac OS ve iOS platformlarında çalışabilmektedir. C++, C, Python, Java, Matlab, EmguCV kütüphanesi aracılığıyla da Visual Basic.Net, C# ve Visual C++ dilleri ile topluluklar tarafından geliştirilen farklı wrapperlar aracılığıyla Perl ve Ruby programlama dilleri ile kolaylıkla OpenCV uygulamaları geliştirilebilir.

OpenCV kütüphanesi içerisinde görüntü işlemeye (image processing) ve makine öğrenmesine (machine learning) yönelik 2500'den fazla algoritma bulunmaktadır. Bu algoritmalar ile yüz tanıma, nesneleri ayırt etme, insan hareketlerini tespit edebilme, nesne sınıflandırma, plaka tanıma, üç boyutlu görüntü üzerinde işlem yapabilme, görüntü karşılaştırma, optik karakter tanımlama OCR (Optical Character Recognition) gibi işlemler rahatlıkla yapılabilmektedir.

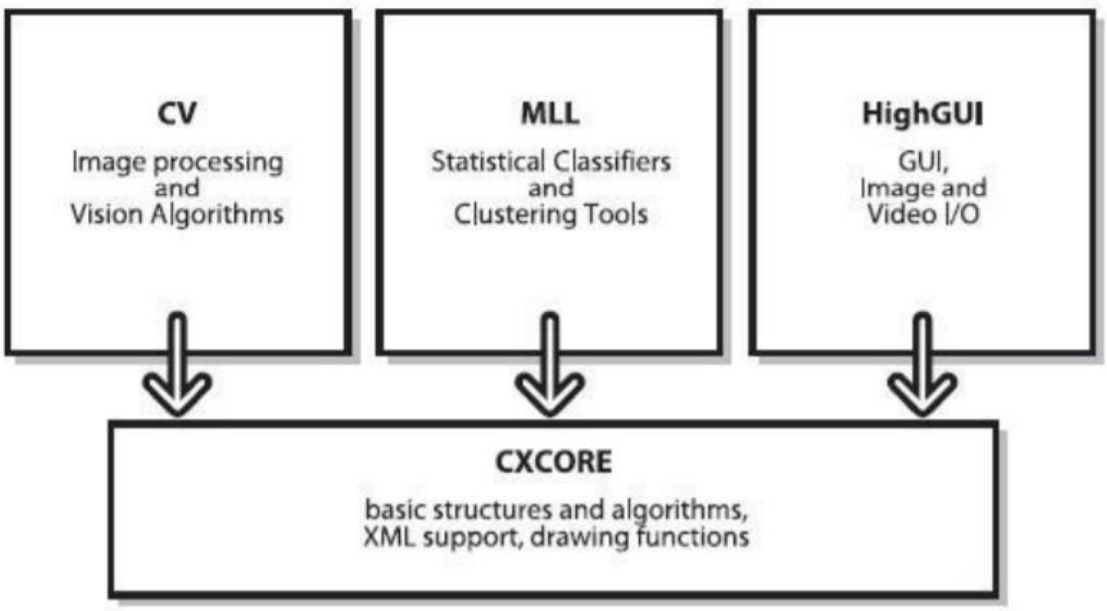
Core: OpenCV'nin temel fonksiyonları ve matris, point, size gibi veri yapılarını bulundurur. Ayrıca görüntü üzerine çizim yapabilmek için kullanılabilecek metotları ve XML işlemleri için gerekli bileşenleri barındırır.

HighGui: Resim görüntüleme, pencereleri yönetme ve grafiksel kullanıcı arabirimleri için gerekli olabilecek metotları barındırır. 3.0 öncesi sürümlerde dosya sistemi üzerinden resim dosyası okuma ve yazma işlemlerini yerine getiren metotları barındırmaktaydı.

Imgproc: Filtreleme operatörleri, kenar bulma, nesne belirleme, renk uzayı yönetimi, renk yönetimi ve eşikleme gibi neredeyse tüm fonksiyonları içine alan bir pakettir. 3 ve sonra sürümlerde bazı fonksiyonlar değişmiş olsada 2 ve 3 sürümünde de bir çok fonksiyon aynıdır.

Imgcodecs: Dosya sistemi üzerinden resim ve video okuma/yazma işlemlerini yerine getiren metotları barındırmaktadır.

Videoio: Kameralara ve video cihazlarına erişmek ve görüntü almak ve görüntü yazmak için gerekli metotları barındırır. OpenCV 3 sürümü öncesinde bu paketteki birçok metot video paketi içerisindeydi.



Şekil 3.4 OpenCV Bileşenleri

4 PROGRAMLAMA

4.1 Programlama da işlem basamakları

- Derin öğrenme yapay sinir ağlarının karmaşık hali olarak düşünülebilir
- YSA insan da ki öğrenmeden esinlenerek harekete geçilen bir yöntemdir
- YSA da insan daki nöron şeklinde tanımlanan yapılar birbirleri ile bağlantılı olarak modellenir.
- Öğrenme hafızaya alma ve veriler arası kıyaslama ilişki kurma adımları kapasite olarak düşünülür
- Yüz tanıma sisteminin kurulması ; yüz tarama için öncelikle yüzün referans noktaları alınmalıdır. Yüz tanımayı ikiye ayırabiliriz. Resimler üzerinden yapılan ve hareketli görüntü üzerinden yapılan yüz tanıma tekniği. Proje de bu iki yönteme de yer verildi.
- Uygulama olarak öncelikle yazılım dili tercih edildi. Python kullanılarak işlemler gerçekleştirildi**.**
- Python bir çok alanda sağladığı işlem kolaylığı gibi yüz tanıma algılama sistemlerinde de işlem kolaylığı açısından birçok kısayol sağlıyor.
- Python bize sadece yüz tanıma işlemlerinde yeterli olmayacaktır. Sonrası için bir OpenCV kütüphanesi seçildi. Tercih edilme sebebi ise OpenCV kütüphanesi ile birlikte haarcascade hazır filtreler yardımı ile bu işlemin kolaylıkla yapılabiliyor olmasıdır.
- OpenCV kütüphanesi kullanımı ile öncelikle yüz kayıtları alındı.
- Alınan yüz kayıtları istenilen ölçütler ile bir görüntü elde edildi.
- Elde edilen görüntü özelliklerinde görüntünün boyutları ve renk seçeneği belirlendi. Gray scale bir resim işlemlerde karmaşıklığı ve 2 boyutlu bir matris kullanılarak yüz ölçülerinden alacağımız pixel değerlerinde düzenleme görüntüyü düzenleme konusunda daha tutarlı sonuçlar vereceğinden tercih edildi.
- OpenCV nin bize sağladığı önemli kısımlardan biri ise Highgui ; görüntü ve hareketli görüntü veya video okuma yazma işlemlerinde yardımcı fonksiyonu barındırmasıdır.
- Python kullanmak için arayüz olarak VisualStudio tercih edildi.
- Veriseti kısmında gösterileceği üzere bilgisayar kamerası üzerinden kamera açıldı.
- Kamera üzerinden alınan görüntüler değişkene atandı.
- Atanan değikenler üzerinde olan görüntüler öncelikle boyutlandırıldı.
- Boyutlandırılan görüntüler üzerinde renk değişimi yapıldı ve grayscale bir görüntü elde edildi.
- Elde edilen grayscale resim veriseti olarak hazırlandı ve sonra eğitim için kullanılmak üzere bir klasöre kaydedildi.
- Eğitim için Numpy kütüphanesi tercih edildi. Numpy kütüphanesi tez de belirtildiği üzere çok boyutlu dizi veya matrislerde çalışmaya yardımcı matematiksel işlemler yapabileceğimiz bir kütüphanedir. Görüntü işlemede bizim kullanacağımız görüntünün matris değerlerinin tutulmasında ve onlar üzerinde işlem yapılabilmesinde yardımcı oldu.
- Sonra PIL kütüphanesi kullanılarak grafikler kaydedildi ve filtreleme yapıldı.
- Yapılan filtreleme sonucunda eğitim gerçekleştirildi.
- Yüz takibi işlemi için öncelik adımlardan sonra şimdi ise anlık görüntü alımı ve alınan görüntü üzerinde filtreleme yapılması ve ayrıca tanıma kullanılarak tutarlılık sağlandı.
- Robot kol otonom hareketi için bir pencere ve yüz etrafında dikdörtgen bir çerçeve oluşturuldu .

- Merkez konumları ikisi içinde belirlendi belirlenen merkez konumlarının birbirine olan uzaklıkları arasında ki vektörel uzaklık alınarak karar mekanizmasında sağ-sol-aşağı-yukarı olarak tanımlandı ve buna göre Arduino ile seri iletişim sağlanıp arduinoya bu bilgiler karakter harfler olarak gönderildi.
- Arduinoda seri porttan aldığı bu bilgileri kullanılarak istenilen yönde hareket gerçekleştirildi.

4.2 Programlama Dili

Projenin yapımı aşamasında belirtildiği üzere Python programlama dili kullanılmıştır. Ve programlama yapılırken arayüz olarak Visual Studio Code tercih edilmiştir. Görüntü işleme python üzerinden gerçekleştirilip Arduino ile seri haberleştirilerek motor kontrolleri ve sensor kontrolleri Arduıno IDE üzerinden kodlanmıştır. Bu bölümde kullanılan kodlar işlevleri ve fonksiyon basamakları ve kullanım amaçları anlatılacaktır.

4.2.1 Python Tercih Edilen Kütüphaneler

```
import cv2
import numpy as np
from PIL import Image
import os
import serial
import time
```

OpenCV kütüphanesi

import cv2 yazarak kodumuza dahil ettiğimiz kütüphanedir.
Tüm görüntü işleme işlemlerinde kullanılan ve bu alanın öncüsü bir kütüphanedir.

Numpy kütüphanesi

NumPy (Numerical Python) bilimsel hesaplamaları hızlı bir şekilde yapmamızı sağlayan bir matematik kütüphanesidir. Numpy'ın temelini numpy dizileri oluşturur. Numpy dizileri python listelerine benzer fakat hız ve işlevsellik açısından python listelerinden daha kullanışlıdır.Ayrıca python listelerinden farklı olarak Numpy dizileri homojen yapıda olmalıdır yani dizi içindeki tüm elemanlar aynı veri tipinden olmalıdır.

PIL Kütüphanesi

Python Imaging Library (kısaca **PIL**), Python programlama dili için geliştirilen, açık kaynak kodlu grafik işleme kütüphanesidir. Bu **kütüphane**, içinde barındırdığı hazır fonksiyonlar sayesinde programcıya üstün bir grafik işleme imkânı sunar.

Os Modülü

Os modülü Python'da hazır olarak gelen , dosya ve dizinlerde kolaylıkla işlemler yapmamızı sağlayan bir modüldür. İşletim sistemlerinin çalışma mantıklarının birbirinden farklı olduğu göz önünde bulundurulduğunda os modülü bizlere farklı işletim sistemleri ile tutarlı bir şekilde iletişim kurabilmemizi sağlayacak pek çok fonksiyon sunar.

Serial Kütüphanesi

Arduino ve Python seri haberleşmesini sağlamak için kullanılan kütüphanedir.

Time Fonksiyonu

Time fonksiyonu zamanın başlangıcından içinde bulunduğumuz ana kadar geçen süreyi hesaplamakta kullanılmaktadır. başlangıç olarak 1 Ocak 1970 yılı genellikle baz alınmaktadır. Kullanımı aşağıdaki gibidir.

4.2.2 Veri Seti Oluşturma

```
import cv2

kamera = cv2.VideoCapture(0)  # kameranın aktif hale getirilmesi
kamera.set(3, 640) # video genişliğini belirle
kamera.set(4, 480) # video yüksekliğini belirle
face_detector = cv2.CascadeClassifier('Cascades/haarcascade_frontalface_default.xml')
# Her farklı kişi için farklı bir yüz tamsayısı ata
# face_id = input('\n enter user id end press <return> ==>  ')
MAXFOTOSAY = 50 # Her bir yüz için kullanılacak görüntü sayısı
face_id = 1
print("\n [INFO] ekrana bak ve görüntü alımı için bekle ...")          # ekrana görüntü alımı için bakılması gerektiğini yazdırıldı

say = 0  // ilk fotoğrafın dögüye girmeden sayacın atanması

while(True):
    ret, img = kamera.read() // kamerayı aç ve görüntü al
    # img = cv2.flip(img, -1)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # aldığımız RGB görüntüyü gray scale bir görüntü haline opencv kaynağının ölçütleri doğrultusunda dönüştür
    yuzler = face_detector.detectMultiScale(gray, 1.3, 5)
    for (x,y,w,h) in yuzler:
        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)  # görüntünün büyüklüğü kenar uzunlukları belirlenen kısım
        say += 1
        # Yakalanan imajı veriseti klasörüne kaydet
        cv2.imwrite("veri/" + str(face_id) + '.' + str(say) + ".jpg", gray[y:y+h,x:x+w])  # resmi kaydetmek icin kullanılan komut
        cv2.imshow('imaj', img) # imaj adı altında bizim görüntümüzü gösterildi
        print("Kayıt no: ",say) # çekilen fotoğraf sayısını sırayla ekrana yazdırıldı
```

```
    k = cv2.waitKey(100) & 0xff # klavyeye bağlanma fonksiyonu olarak kullanılır bu da klavyeden atanan tuş ile program durdurma pencerenin kapanması için kullanım sağlar
    if k == 27: # burada 27 ms içinde tuşa basılıp basılmadığını kontrol edildi
        break
    elif say >= MAXFOTOSAY:
        break
print("\n [INFO] Program kapatılıyor.")
kamera.release() # kameray kapatıldı
cv2.destroyAllWindows() # işlemler kapatıldı
```

Veriseti oluşturma ile önizleme evresinde, yüz için kişi yüzleri çekilerek elde edilen görüntü gibi görsel kaynaklar belirli işlemden geçerek bir verisetine dönüştürülmüştür.

Şekil 4.1 Veri adlı klasörde toplanması sağlanmıştır.

- `face_detector = cv2.CascadeClassifier('Cascades/haarcascade_frontalface_default.xml')`
- **Haar Cascade** Sınıflandırıcısı, **Opencv** de çok işlevsel bir özelliktir. Sınıflandırıcı önce belirli örneklerle eğitilir ve sonrasında o nesne algılanır.
- Burada bu cascade kullanılarak yüz ölçülerinin kaydedilmesi ortalama bir yüzün algılanması için alacağı ölçü değerlerinin kaynak kodunun cv2 kütüphanesi ile kullanımı.

Veri Çoğullama

Elde edilen görüntüleri daha hızlı ve sisteme uygun bir şekilde kullanmak için yeniden boyutlandırma işlemi yapılmış ve 640x480 olarak boyutlandırılmıştır.

Veri Setinin Normalize Edilmesi

Etiketlenen görüntülerin uzunluk(dH) ve genişliği(dW), DPY nesnesinin görüntünün tam olarak hangi konumunda olduğu bilgisini veren noktaları ve görüntü üzerinde kaç adet DPY olduğu bilgisi text dosyası içerisine kayıt edilmektedir. Tüm görüntüler üzerinde bu işlem uygulanarak etiketli veri seti oluşturulmuştur. Oluşturulan veri setini, sistemde kullanılan YOLO [23] mimarisine uygun hale getirmek için normalizasyon işlemi yapılmaktadır. Normalizasyon işlemi ile koordinatları 0 ve 1 aralığına indirgenmiştir. Bu işlem ile görüntü içerisindeki etiketli. LabelTool Veri Etiketleme Programı[54] 23 DPY'nin, merkez noktası koordinatları(X,Y), yüksekliği(H) ve genişliği(W) bilgileri elde edilmektedir.

$$X = \frac{X_1 + X_0}{2} x \frac{1}{dW}$$

$$Y = \frac{Y_1 + Y_0}{2} x \frac{1}{dH}$$

$$W = (X_1 - X_0) x \frac{1}{dW}$$

$$H = (Y_1 - Y_0) x \frac{1}{dH}$$

X : Etiketli verinin orta noktasına ait X koordinatı.
Y : Etiketli verinin orta noktasına ait Y koordinatı.
W : Etiketli verinin genişliği
H : Etiketli verinin yüksekliği

Şekil 4.2 Normalizasyon işleminin formulü

4.2.3 Veriseti Eğitimi

```
import cv2
import numpy as np
from PIL import Image
import os

path = 'veriseti' # kullanılacak verinin alınacağı kaynak için bir yol değişkeni atandı
recognizer = cv2.face.LBPHFaceRecognizer_create() # veri kümemizi atadık bir sonraki satırda bunu nereden alacağımızı belirtildi
detector = cv2.CascadeClassifier("Cascades/haarcascade_frontalface_default.xml") # filtreleme işlemi için kullanılacak cascade I seçtik
def getImagesAndLabels(path):  # path klasöründen dosyaları alındı
    imagePaths = [os.path.join(path,f) for f in os.listdir(path)] # path klasöründeki dosyaları listeledik
    ornekler=[] # listedeki her görüntünün etiketini tutuldu
    ids = []
    for imagePath in imagePaths:
        PIL_img = Image.open(imagePath).convert('L')    # gri
        img_numpy = np.array(PIL_img,'uint8')
        id = int(os.path.split(imagePath)[-1].split(".")[0])
        # print("id= ",id)
        yuzler = detector.detectMultiScale(img_numpy)
        for (x,y,w,h) in yuzler:
            ornekler.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)
### son 9 satırda  imagePath listesinde ki her bir eleman kullanılarak ilgili resim okunup griye dönüştürüldü ve Pılimg değişkenine atandı. Sonra bu görüntüler 8 bitlik pozitif satılardan oluşan bir numpy arrayi haline dönüştürüldü. Sonra ise dosyadan alınan girdi ismi tam sayı olarak ıd değişkenine tanımlandı. Img  numpy görüntüyü numpy arrayinde bulunan yüz ölcülerini bulup faces listesine atandı.
    return ornekler,ids
print ("\n [INFO] yuzler eğitiliyor.")
yuzler,ids = getImagesAndLabels(path)
recognizer.train(yuzler, np.array(ids)) # burada ise yüzlerimizi eğitildi. Sonra ise bunu kaydediliyor.
```

```
recognizer.write('egitim/egitim.yml')
print(f"\n [INFO] {len(np.unique(ids))} yüz eğitildi.  sonlandırılıyor.")
```

Burada görüntüler eğitildi ve eğitim adlı klasöre kaydedildi . Bu algoritmalar öncelikle bir eğitim gerektirir. Eğitim yüz resimlerinin algoritmaya verilerek bu görüntülerden bir vektör oluşturur bu vektör eğitimde kullanılan yüzlerin belirgin özelliklerinden oluşur. Bu sayede elde edilen vektör veri tabanındaki tüm yüzlerden yararlanarak oluşturulmuş olur. Daha sonra karşılaştırma için gönderilecek yüzler bu vektör ile karşılaştırılır. Her eğitim sonrası, bu vektör değişir ve eğitim için kullanılan veri ne kadar fazla ise başarı oranı da bir o kadar artar. Eğitim için en az iki adet görüntüye ihtiyaç vardır ve eğitim için kullanılacak her dosyanın genişliği ve yüksekliği aynı boyutta olmalıdır. Eğitim için örnek veritabanlarını indirip kullanacağınız gibi kendinizde çekmiş olduğunuz resimleri kullanabilirsiniz. Kullanılacak görsellerin formatları genellikle jpg, jpeg, png, pmg vb. dir. Att veritabanını indirirseniz, içerisindeki görüntü formatları pmg dir. OpenCV de eğitim için FaceRecognizer sınıfı altında train() metodu mevcuttur. Bu metot parametre olarak eğitim için kullanılacak mat vektörlerini yani yüzleri ve bu yüzler için kullanılacak etiketleri (label) alır. Etiket kavramını daha önce açıklamıştık fakat tekrardan değinmek gerekirse, eğitilen her yüz görüntüsünün eşleşme sonucunda adlandırılması için verdiğimiz id numaralarıdır. Bu etiketler görüntü dosyasının isimlerinde kullanılır.

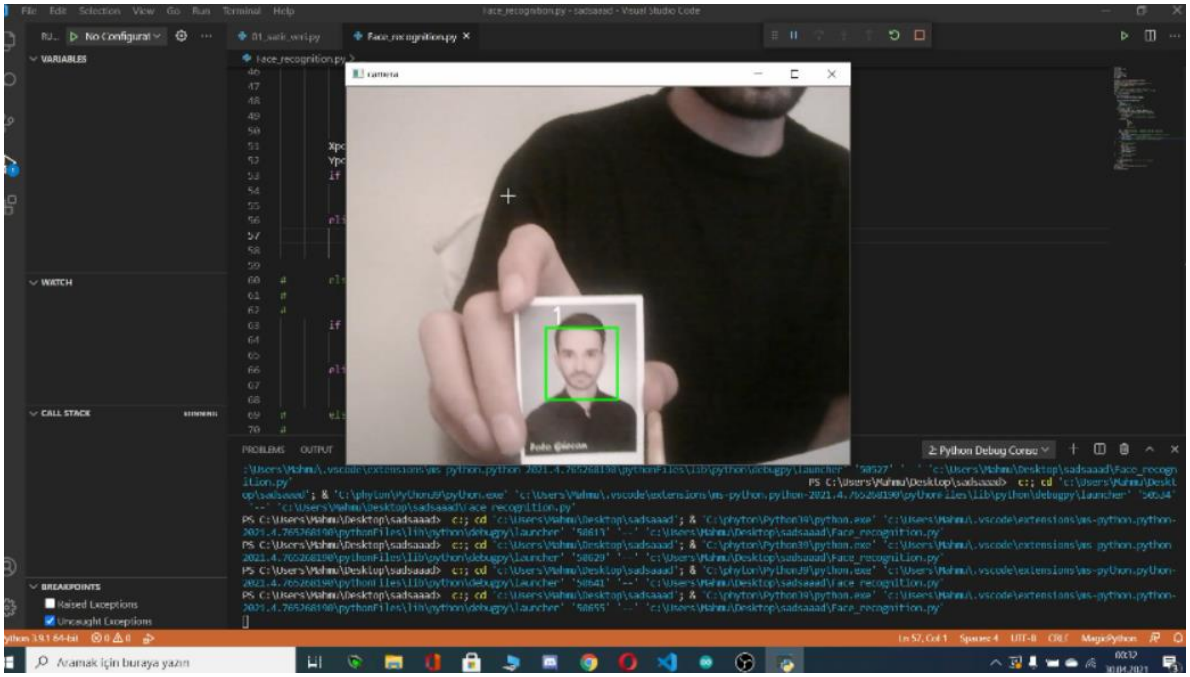
4.2.4 Yüz Algılama

```
import cv2
import numpy as np
import os               # kütüphanelerimiz eklendi.
import serial
import time
ard= serial.Serial() # Arduino ile seri haberleşme sağlanılması için gereken komut satırı
ard.port = "COM3"  # haberleşilecek olan Arduino nun bağlanılacağı pc portu seçildi
ard.baudrate = 9600 # Arduino baudrate'i  seçildi
ard.open()  # Arduino dosyaları açıldı
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('egitim/egitim.yml')  # eğitim klasörü kaynak olarak okundu
cascadePath = "Cascades/haarcascade_frontalface_default.xml" # filtre kaynağı seçildi.
faceCascade = cv2.CascadeClassifier(cascadePath); # filtreleme gerçekleştirildi
font = cv2.FONT_HERSHEY_SIMPLEX # yazı fontu seçildi
id = 0
names = ['None', 'Kisi Algilandi', 'hastax ', 'hastay', 'Z', 'W']
cam = cv2.VideoCapture(1) # kullanılmak istenen kamera tanımlandı
while True:
    ret, img =cam.read() # kameradan görüntü alındı
    img = cv2.flip(img, -1) # görüntü img adına atandı
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY) # gray olarak dönüştürülüp kaydedildi

    faces = faceCascade.detectMultiScale(
```

```
        gray,
        scaleFactor = 1.2,
        minNeighbors = 5,
       minSize = (80, 80),#(int(minW), int(minH)
       )
    #cv2.circle(img,(320,240), 90, (255,0,0), 2)
    for(x,y,w,h) in faces:
        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2) # pencere boyutlandırıldı
        id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

        if (confidence < 100):
            id = names[id]
            confidence = "  {0}%".format(round(100 - confidence))

        cv2.putText(
                    img,
                    str(id),
                    (x+5,y-5),
                    font,
                    1,
                    (255,255,255),
                    2
                   )
# aşağı daki komut satırlarında Merkez konuma uzaklıklar seçildi ve buna göre Arduino ya bilgi gönderildi.
        Xpos = x+(w/2)#X KORDİNATININ  MERKEZ NOKTASI
        Ypos = y+(h/2)# YKORDİNATININ  MERKEZ NOKTASI
        if Xpos >= 380:
            ard.write('L'.encode())#YÜZ SOL DA İSE
            time.sleep(0.01)
        elif Xpos <= 260:
            ard.write('R'.encode())
            time.sleep(0.01)

#       else:
#           ard.write('S'.encode())
#           #time.sleep(0.01)
        if Ypos > 300:
            ard.write('D'.encode())
            time.sleep(0.01)
        elif Ypos < 180:
            ard.write('U'.encode())
            time.sleep(0.01)
#       else:
#           ard.write('S'.encode()) # Arduino nun seri portuna yukarıda encode olarak yazılan değer iletildi.
#           time.sleep(0.01)
        break
```

```
    cv2.imshow('camera',img)
    k = cv2.waitKey(10) & 0xff # Press 'ESC'
    if k == 27:
        break

cam.release()
cv2.destroyAllWindows()
```

Bu bölümde eğitim klasöründen alınan veri veya kameradan alınan veri arasında ve kameradali görüntünün sadece kendi içinde eşleme yapılır.iki seçenek olarak eşleme kullanılmasının sebebi önceden bir yüz tanıtıldığı zaman program daha verimli kordinat işleyebilmekte. Öncelik kütüphaneler eklenmekte Arduino ile seri haberleşme sağlanmakta sonrasında ise recognizer ile eğitim klasöründen alınan veri okunmakta sonra ise cascade ile bu klasör arasında eşleme yapılmakta sonrasında ise kameradan görüntü alınmakta kameradan anlık alınan görüntü cascade ve eğitim klasörü ile kıyaslanmakta veri uyuşması noktasında yani yüz algılanma noktasında algılanan yüzün etrafında tam uç noktalara gelecek şekilde bir alan belirlenmekte . Belirlenen alan kullanılarak center (Merkez) noktası tayin edilir. Edilen Merkez nokta kullanılarak ana pencerinin Merkez konumu ile bir veri olarak alınıp X ve Y kordinatları olmak üzere kordinatların birbirine uzaklığı belirlenir. Belirlenen uzaklık X ve Y ekseninde hesaplanarak hesaplanan bilgi doğrultusunda kartezyen sistemde – veya + ; x, y nin mesafesi harf olarak tayin edilir. Merkez konumlar eşlenene dek sürekli - + kısımda bilgi analiz edilir ve harf olarak kaydedilir. Kaydedilen veriler seri haberleşme sağlanan arduinoya gönderilir. Arduino ise bu verileri alarak servo konumlarını Merkez konumların eşlenmesi için istenen yönde hareketini sağlar. Böylelikle orta konumda burna odak sağlanır.



Şekil 4.1 Yüz Algılama Testi

4.3 Arduino IDE

Arduino IDE, arduino kitleri için geliştirdiği; komutların yazılmasına, derleme işleminin yapılmasına ve son olarakta derlenen kodları doğrudan (Bilgisayarın USB portuna bağlı olan) Arduino kite yüklenmesine olanak sağlayan yazılım geliştirme platformudur. Aşağıda Arduino IDE programının ara yüzü görülmektedir. Programın doğru şekilde çalışması ve python dan veri alınması alınan verinin karar verilerek servoların yönünü belirleme işlemelerinin yapılması için yazılan arduino kodu aşağıda ki gibidir.

```
#include <Servo.h>
#define RIGHT  12
#define LEFT  13
        Servo servoX;
        Servo servoY;
        Servo servoZ;
        Servo servoT;
int k=5;
int x=90 ;
int y=115 ;
int z=110 ;
int t=90 ;
const int led=4;
const int button=3;
int stateb=0;
char input = ' ';
void setup() {
          Serial.begin(9600);
 // Input/output
 pinMode(led,OUTPUT);
 pinMode(button,INPUT_PULLUP);
 servoX.attach(8);
 servoY.attach(9);
 servoZ.attach(10);
 servoT.attach(11);
```

```
 servoX.write(x);
 servoY.write(y);
  servoZ.write(z);
 servoT.write(t);
pinMode(LEFT,INPUT_PULLUP); // pin 12 buton için atandı
 pinMode(RIGHT,INPUT_PULLUP);// pin 2 buton için atandı
 delay(1000);
}
void loop() {
 stateb=digitalRead(button);
  //************
          if(digitalRead(RIGHT) == LOW){
   if (z > 0 && z <= 180) {
     z = z - k;
      if(z < 0){
       z = 0;
      }else{
     servoZ.write(z); // move the servo to desired angle
      }
   }
 delay(100); // waits for the servo to get there
 }// while
//

  if  (digitalRead(LEFT) == LOW){
   if (z >= 0 && z <= 180) {
     z = z + k ;
     if(z >180){
       z =180;
      }else{
```

```
    servoZ.write(z); // move the servo to desired angle
     }
   }
 delay(100); // waits for the servo to get there
 }// ****************************
if(stateb==LOW)// if button ON
{
digitalWrite(led,LOW);
if(Serial.available()){ //seri arabellekte herhangi bir veri olup olmadığını kontrol eder
 input = Serial.read(); //verileri bir değişkene okur
 // Send results to Serial Monitor
 Serial.print("Distance = ");
 if(input == 'D'){
  y -= 1;
  servoY.write(y);
 }
 else if(input == 'd')
 {
  y -= 2;
  servoY.write(y);
 }
 else if(input == 'U'){
  y += 1;
  servoY.write(y);    }
 else if(input == 'u'){
  y += 2;
  servoY.write(y);
 }
/* else{
  servoY.write(y);
```

```
 } */
 if(input == 'L'){
 x -= 1;
 servoX.write(x);
 }
 else if(input == 'l'){
 x -= 2;
 servoX.write(x);
 }
 else if(input == 'R'){
 x += 1;
 servoX.write(x);
 }
 else if(input == 'r'){
 x += 2;
 servoX.write(x);
 }
//  else
//  {
//  servoX.write(x);
//  }
 input = ' ';          //clear
 }
 }// end if
  else if(stateb==HIGH)
  {
     for(x;x<=180;x+=1){
servoX.write(x);  //*** x ekseni for
delay(50); }
  for(z;z<=140;z+=1){ //** z eksenı for
```

```
servoZ.write(z);
delay(50);                          }
for(y;y>=70;y-=1){ //******* y ekseni for
servoY.write(y);
delay(50);                          }
 for(t;t>=6;t-=1){  //** t eksenı for
servoT.write(t);
delay(50);                          }
 delay (150);
     for(x;x>=90;x-=1){                  // dönüs
servoX.write(x);  //*** x ekseni for
delay(50); }
   for(z;z>=110;z-=1){ //** z eksenı for
servoZ.write(z);
delay(50);                          }
for(y;y<=130;y+=1){ //******* y ekseni for
servoY.write(y);
delay(60);                          }
 delay (1500);                      }
  else
  {
    digitalWrite(led,HIGH);
    servoT.write(90);                   }}
```

5 PROTOTIP YAPILIŞI

5.1 Materyaller

Arduino Uno

Arduino ailesi içinde yapılacak projeye, kullanıma ve isteğe göre çeşitli Arduino kartları vardır. Bunlar; Arduino Uno, Arduino Mega 2560, Arduino Pro, Arduino Leonardo, Arduino Fio, Arduino Mega ADK, Arduino Nano vs. gibi isimlendirilmiş performansları, özellikleri, işlemcileri ve kabiliyetleri farklı kartlardır. Arduino Uno R3 kartı üzerinde 8 bitlik Atmega328 işlemcisi, 14 digital giriş- çıkış (Input- Output) pini, bunlardan 6 tanesi PWM çıkışı olarak kullanılabilmektedir, ayrıca 6 analog giriş pini bulunmaktadır. Ayrıca 16 Mhz kristal osilatör, USB bağlantısı, regüle edilmiş 5V, ICSP başlığı ve reset butonu bulunmaktadır. Çalışması için gerekli gücü USB bağlantısı ile PC'den veya 7-12 Volt'luk DA güç kaynağından sağlar. Besleme gerilimi için alt ve üst sınırlar 6-20 V olarak belirlenmiştir. Giriş çıkış pini başına akım 40 mA dir. Kart üzerinde regüle edilmiş 3.3 V çıkışı da bulunmaktadır ve 3.3 V için çıkış akımı 50 mA'dir. FLASH Hafıza 32KB, SRAM 2 KB, EEPROM 1KB olarak hafıza büyüklükleri de belirtilmiştir. Ayrıca; harici bir güç kaynağı kullanılacağı zaman Arduino kartı üzerindeki VIN pininden giriş yapılabilir.

Servo Motor:

Motorlar,dairesel olarak hareket ederek kuvvet üreten endüstriyel cihazlardır. Motorlar genel olarak DC Motor, AC Motor,Servo Motor ve Step Motor olmak üzere dört farklı türde sınıflandırılır. Tüm bu motor çeşitleri yapı olarak birbirlerine benzeseler de, özellik ve çalışma mantıklarına göre bir birlerinden farklıdırlar. Servo ve step motorları gelişmiş motorlar olarak gösterebiliriz. Günümüzde kontrol sistemleri oldukça yaygın olarak kullanılmaktadır. Hemen hemen her alanda ve uygulamada bir kontrol sistemi mevcuttur. Kontrol sistemlerinin; robot sektörü, otomativ sektörü, uçak sektörü, radarlar gibi çok geniş uygulama alanları vardır. Servo motorlara kontrol motorları da denilmektedir. Özellikle kontrol sistemlerinde çıkış hareketlerini kontrol edici olarak kullanılmak üzere tasarlanılıp üretilirler. Bir servo sistem veya servomekanizma geri beslemeli bir kontrol sistemi olup, sistemin çıkışı mekanik konum,ivme veya hız olabilir. Servo sistem ile konum (veya hız) kontrol sistemleri aynıdır. Servo sistemler günümüzde modern endüstride çok sık olarak kullanılmaktadır. Robot teknolojisinde en çok kullanılan motor çeşididir. Bu sistemler mekanik olabileceği gibi elektronik, hidrolik-pnömatik veya başka alanlarda da kullanılabilmektedir. Servo motorlar; çıkış, mekaniksel konum, hız veya ivme gibi parametrelerin kontrol edildiği, özetle hareket kontrolü yapılan bir düzenektir. Servo motor içerisinde herhangi bir motor AC, DC veya step motor bulunmaktadır. Ayrıca sürücü ve kontrol devresini de içerisinde barındırmaktadır.

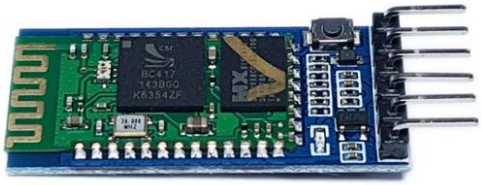
Dört Eksen Robot Kol

Burada mekanik kısım olarak örnek bir tasarım alınmıştır alınan tasarım istenilen açılarda hareket kabiliyetine sahiptir. Robot kolun parçaları 3D print ile ölçülü çizilmiş olup 3D baskı yapılarak ilk aşama için yeterli olacağı düşünülmüştür

Şekil 5.1 Robot Kol

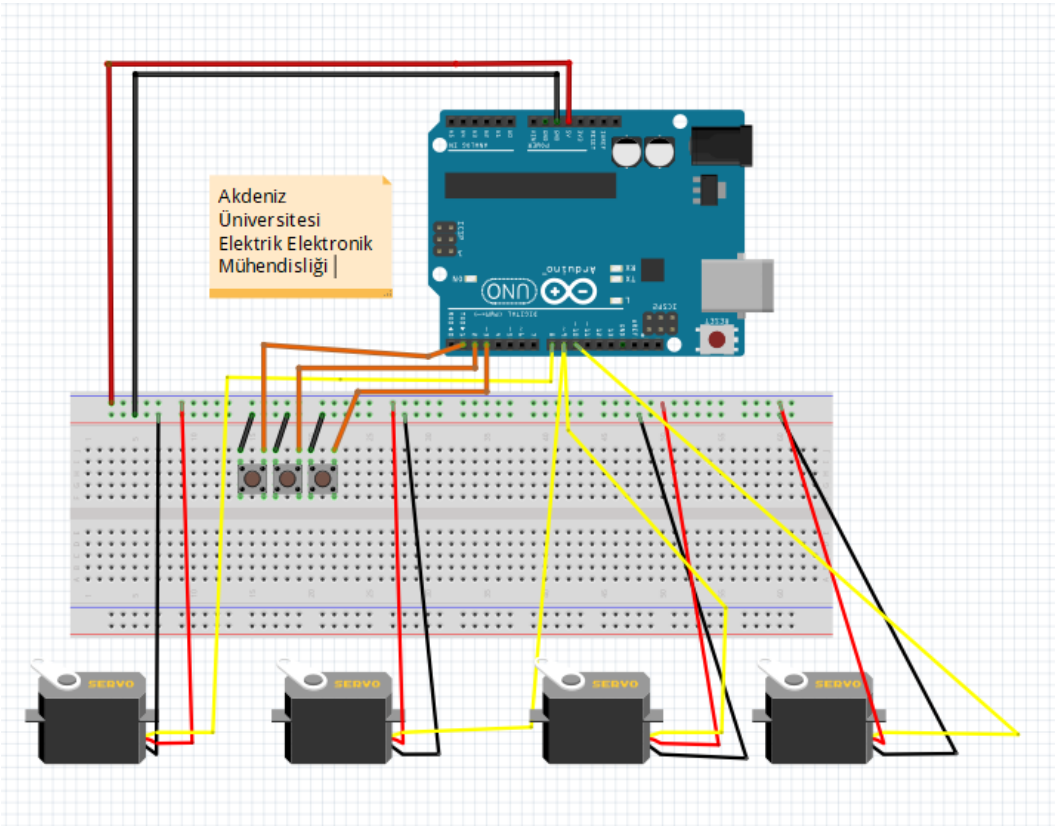
Hc05 Bluetooth Modülü

HC05 Bluetooth Modülü, **Bluetooth** SSP(Serial Port Standart) kullanımı ve kablosuz seri haberleşme uygulamaları için yapılmıştır. ... Bu kart **bluetooth** 2.0'ı destekleyen, 2.4GHz frekansında haberleşme yapılmasına sağlar. Açık alanda yaklaşık 10 metre büyüklüğünde bir haberleşme mesafesine sahiptir.



Şekil 5.2 Bluetooth Modulü

5.2 Devre Şeması



Şekil 5.3 Devre Şeması

5.3 Kullanılan Arayüz

Visual Studio Code

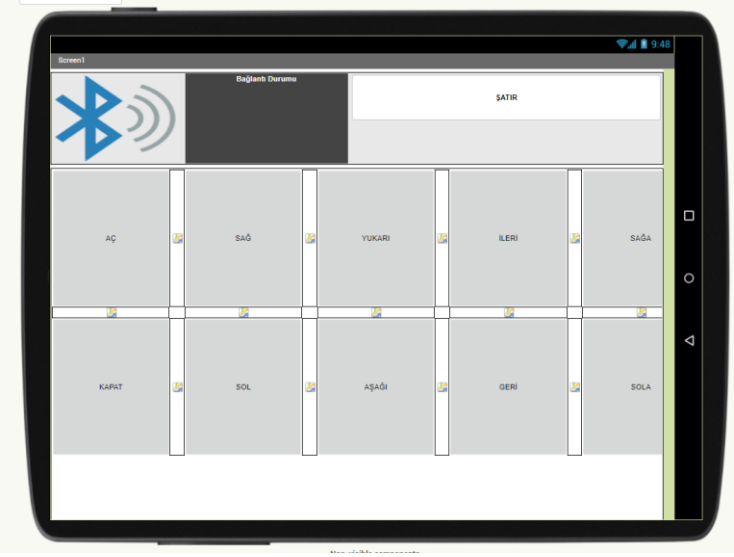
Visual Studio Code hızlı, hafif olmanın yanında Microsoft, Linux ve Mac işletim sistemlerinde çalışabilen bir araçtır. Kurulum yapıldığında basit bir metin editörü görüntüsü ile karşımıza çıkan ürün, eklentiler sayesinde Node JS, Ruby, Python, C/C++, C#, Javascript gibi bir çok programlama dilini desteklemektedir. Yani klasik Visual Studio ürünleri gibi her şeyi bünyesinde barındırmak yerine çekirdek bir yapı ile başlatılıp lazım olan parçaların eklentiler halinde ürüne eklenmesi sağlanarak mantıklı bir hareket yapılmıştır. Eklentilerin indirilebildiği bir market ortamı oluşturulmuştur.

Visual Studio Code, hata ayıklama özellikleriyle, gelişmiş web ve bulut uygulamaları üstünde kodları düzenlemeye, yeniden tanımlamaya ve optimize etmeye yarar. Visual Studio Code tamamen ücretsiz olup, dilediğiniz gibi kullanabilir, kodlarını inceleyebilir ve kendi ihtiyaçlarınıza göre değişim yapabilirsiniz. Uygulama, çoklu platform desteğine sahip olduğu için Linux, Mac OS X ve Windows üzerinde çalışır ve programcılar için yaklaşık 30 programlama dili desteği sunar.

Şekil 5.4 Prototip Görüntüsü

MIT App Inventor

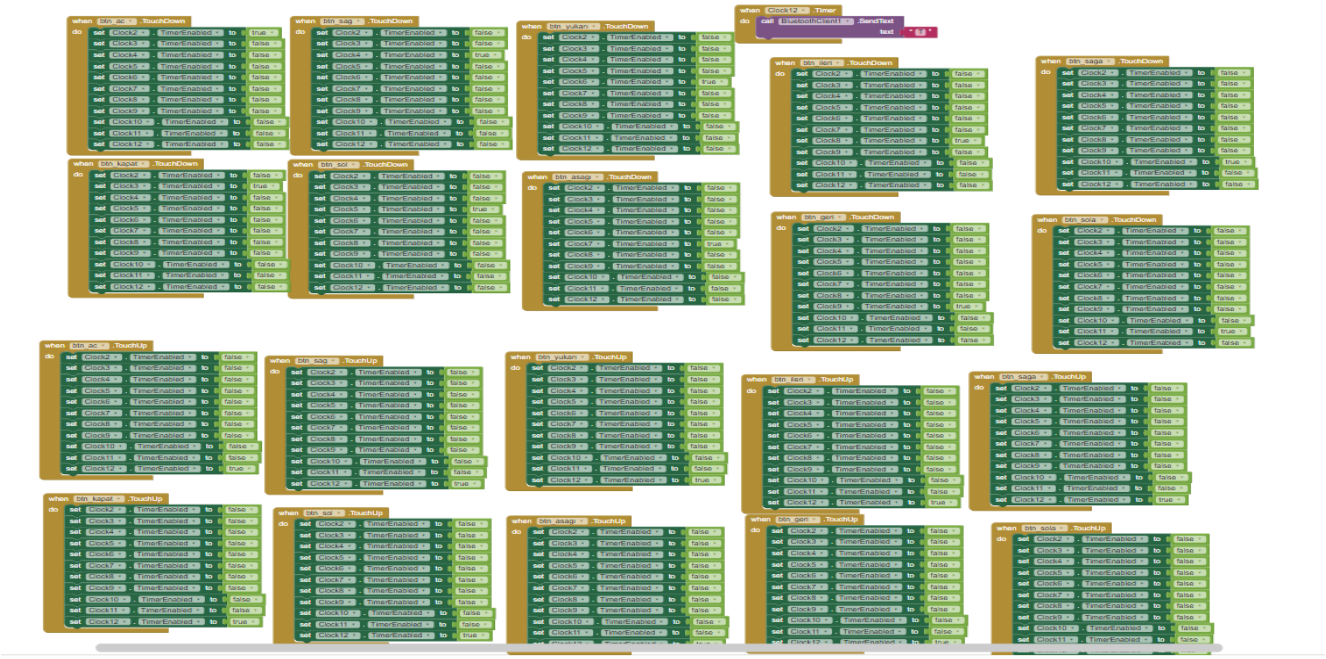
MIT App Inventor, Google tarafından ortaya çıkarılan ve sonrasında Massachusetts Institute of Technology (MIT) tarafından geliştirilen, özgür bir uygulama geliştirme aracıdır. Yeni başlayanların Android işletim sistemi için blok kodlama yöntemiyle uygulama geliştirmesine olanak sağlar. Özellikle yapboz gibi olan yapısı ve sürükle-bırak mekanizması sayesinde kolaylıkla uygulama yapılabilir



Şekil 5.5 Android Uygulama Ekranı

Android uygulamasında robot kolun burun içine girip çıkarken kullanıcı kontrolünde uzaktan yönetilebilmesi için android uygulaması yazıldı. Var olan butonlar arduinoya tanımlanarak kullanılması öngörüldü.

Öncelikle Bluetooth uygulamasının planı tasarlandı. Ve işlem türüne göre bloklar oluşturuldu. Sonra ise 12 adet clock tanımlandı. Tanımlanan clocklar ile sürekli olarak aktiflik ve inaktiflik durumu kontrol edildi. Sürekli sağlanan bu kontrol de aşağı da ki şekilde görüldüğü gibi her bir clock bir butonu temsil etmekte ve bu butonlara basıldığında eğer sadece o buton aktif olduğu durumda veri olarak çıkışa bir char değerinde harf gönderildi. Gönderilen harf ise Bluetooth üzerinde hc05 modülü vasıtası ile arduinoya iletildi. Arduino da alınan girdi ile servoların +;- yönde çalışması sağlandı.



Şekil 5.7 Bluetooth uygulaması işlem blokları

SONUÇ

Robotik sistemler her alanda kullanılmakta ve insan hayatına büyük ölçüde gelişme katmaktadır. Sağlık sisteminde ki robotik alanda gelişmeler ise önem arzetmektedir. Bu gelişmeler ile daha fazla insana daha kısa sürede hizmetler verilebilmektedir.

Tez kapsamında derin öğrenme ile görüntü işlemenin bize kattığı kolaylıklardan fayda alınarak hazırlanan proje de anlatıldığı üzere bir kameradan alınan veride ki renklerin pixel değerlerinin kullanılıp filtrelenerek elde edilen bir yüz şeması ile mekanik hareket kabiliyetine sabit bir robot kol için işlevsel karar verebilen bir beyin oluşturulup davranış ile bir insan hareketi gerçekleştirildi.

Alınan veriler ve gönderilen veriler arasında ki hatalar deneme ve test yolları ile prototip aşamasında ayıklama geçirildi. Sonuç olarak ortaya çıkan robot kol insan burnuna öncelik otonom olarak ilerleme sağladı. Bu ilerlemenin katetdiği yol ve uzaktan kontrol edebilmesi sonucunda insanın numune alınacak kişiye teması olmadan veya istenildiği kadar uzakta kalarak müdahele edilip numune alındı. Numune alınma işlemi PYTHON ve ARDUINO IDE nin sunduğu kütüphaneler kapmasında tutarlı bir hareket gerçekleştirilmesini sağladı. Ancak projede yaklaşık %70'e tekabül eden doğruluk oranına ulaşıldı. Doğruluk oranının düşük olmasının birçok nedeni vardır. Görüntü kalitesinin düşük olması , yüz bulma algoritmasının ve görüntü işleme tekniklerinin yetersiz kalması ayrıca mekanikte oluşan sapmalar bu doğruluk oranının düşme nedenlerindendir. Literatür araştırmalarında bu derecelerde robotun %100 doğruluk oranına ulaşmasını mümkün olduğunu gösterebilen projeler görüldü. Bu yönde geliştirmeler için çalışmaya devam edilmektedir. Bu proje ev ortamında tam teşekküllü olmayan koşullarda test edilmiştir. Hata paylarının azaltılması, geliştirmeler vb işlemler sonucunda amacına uygun hizmet vermesi sağlanılabilir.

KAYNAKLAR

Anonymous. 2021. Web Sitesi: www.docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html

Anonymous. 2021. Web Sitesi: www.eyalarubas.com/face-detection-andrecognition.html

MESUT PİŞKİN 2021. OPENCV İLE GÖRÜNTÜ İŞLEME web sitesi: mesutpiskin.com/blog

Burak YALIM, Yunus Emre SİNEKOĞLU,Alp Tuğrul KURT. Aralık 2016 Görüntü İşleme Tabanli 4+1 Eksenrobot Kol Tasarimi Web sitesi : https://eem.sakarya.edu.tr/sites/eem.sakarya.edu.tr/file/11-_Burak_Yalim-1_-128.pdf

Asaf VAROL, Betül CEBE 2011. Algorithms Of Face Recognition web sitesi : http://web.firat.edu.tr/icits2011/papers/27856.pdf

Muhammed TANRIVERDİ. 2017. Yüz Bulma Ve Tanima Tabanli Otomatik Sinif Yoklama Yönetim Sistemi. Web sitesi: https://dspace.ankara.edu.tr/xmlui/bitstream/handle/20.500.12575/33768/Muhammed_Tanriverdi.pdf?sequence=1&isAllowed=y

SONGÜL TORANOĞLU. 2021. Derin Öğrenme Web sitesi: http://kergun.baun.edu.tr/20172018Guz/YZ_Sunumlar/Derin_Ogrenme_Songul_Toranoglu.pdf

ÖZGEÇMİŞ

Mahmut Şatır, Malatya'nın Yeşilyurt ilçesinde 25.04.1996 yılında doğdu. İlköğretim eğitimini Sümer İlköğretim Okulunda, lise eğitimini Kernek Anadolu Lisesi'nde tamamladı. 2015 yılında Lisans Eğitimi için Akdeniz Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği bölümünü kazandı. Eğitimine devam ediyor.