Teknoloji Fakültesi

T.C

MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

RASPBERRY Pİ İLE YÜZ TANIMA FONKSİYONLU GÜVENLİK SİSTEMİ

Furkan Fikret ŞAFAK

DANIŞMAN

Prof. Dr. Şafak SAĞLAM

İSTANBUL,2021

MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

ELEKTRİK VE ELEKTRONİK

MÜHENDİSLİĞİ

Marmara Üniversitesi Teknoloji Fakültesi Elektrik ve Elektronik Mühendisliği Lisans Öğrencisi Furkan Fikret Şafak'ın "Raspberry Pi İle Yüz Tanıma Fonksiyonlu Güvenlik Sistemi" başlıklı tez çalışması ……/……/…….. tarihinde savunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Prof. Dr. Şafak SAĞLAM (Danışman)

Marmara Üniversitesi .....(İMZA) .....

Prof.Dr. Bülent ORAL (Üye)

Marmara Üniversitesi .....(İMZA) .....

Prof..Dr. Ümit Kemalettin TERZİ (Üye)

Marmara Üniversitesi .....(İMZA) .....

ÖNSÖZ

Pandemi sürecinde bu tezin hazırlanması için gösterdiği her türlü destek ve yardımdan dolayı danışman hocam olan Sayın Prof. Dr. Şafak SAĞLAM ‘ a teşekkür ederim.

Bu çalışma boyunca bana maddi manevi destek olan aileme ve Elektrik Elektronik Mühendisliği Kulübü (EEMK) ‘ne teşekkür ederim.

İçindekiler

Şekil Listesi

KISALTMALAR

RPİ: Raspberry Pi

GPIO: General Purpose Input Output

LBPH: Local Binary Pattern Histogram

HDMI: High Definition Multimedia Interface

LED: Light Emitting Diode

OpenCV: Open Source Computer Vision Library

ÖZET

Gelişen teknolojiler ile hayatımızın güvenliği de tehlikeye girmeye başladı. İlk başlarda 4-5 haneli şifreler kullanılmakta iken devamında farklı şifre şekilleri ortaya çıkmıştır. Yeni nesil güvenlik sistemlerinde ise parmak izi tanıma ve yüz tanıma sistemleri hayatımıza girmiştir. Yapılan tez çalışmasında yüz tanımanın çalışma mantığı, kullanılan yüz tanıma fonksiyonları, sistemin kodları açıklanmıştır. Tez çalışmasının sonucunda Raspberry pi ile yüz tanıma fonksiyonlu güvenlik sistemi tasarlanmıştır.

Sistemin tasarımı sağlanırken LBPH yüz tanıma algoritması ve kodlama dili olarak Python kullanılmıştır. Yapılan sistem önceden belirlenmiş veri setlerini öğrenerek anlık kameradan gelen veya daha öncesinde sisteme yüklenmiş olan görsel ile karşılaştırıp sonuca ulaşmaktadır.

Anahtar Kelimeler: Yüz Tanıma, Raspberry Pi, Güvenlik Sistemleri, LBPH, Haarcascade, OpenCV

1.GİRİŞ

Yaşamakta olduğumuz dünyada güvenlik her zaman önemli olmuştur. Bu sebeple insanlar kendilerini ve mallarını korumak için çeşitli güvenlik önlemleri geliştirmişlerdir. Gelişen teknolojiler sebebiyle alınan güvenlik önlemleri yeterli olmamaya başladığında insanlar parmak izi güvenlik kilidi, yüz tanıma güvenlik sistemi, ses tanımalı güvenlik sistemleri geliştirilmiştir.

1.1 Güvenlik Sistemleri

1.1.1 Parmak İzi Güvenlik Sistemi

Parmak izi tanıma, insanlar için eşsiz olan parmak izinin veri tabanı ile eşleşmesini doğrulamak için otomatik yöntem anlamına gelir. Parmak izleri, kişileri tanımlamak ve kimliklerini doğrulamak için kullanılan birçok biyometri formlarından biridir. Parmak izlerinin eşleşmesi için analizi, genellikle baskı deseninin birkaç özelliğinin karşılaştırılmasını gerektirir. Eşleşen algoritmalar; önceden depolanmış parmak izi şablonlarını, kimlik doğrulama amacıyla aday parmak izlerine karşı karşılaştırmak için kullanılır.

Şekil 1 Parmak İzi Güvenlik Sistemi

1.1.2 Yüz Tanıma Güvenlik Sistemi

Biyometri kişilerin fiziksel ve davranışsal özelliklerini göz önüne alarak yürütülen kimlik belirleme çalışmaları olarak tanımlanır. Bu kişi özelliklerine örnek olarak, retina ve iris görüntüleri, yüz yapı şekilleri, saç sakal yapısı, göz çevresindeki çöküntüler ve kaşlar verilebilir. Daha genel anlamda, biyometri bilişim teknolojilerinin bir parçası olup, insanların fiziksel özelliklerini inceleyip karşılaştırarak kişiyi tanıma amaçlı ipuçları çıkaran bir bilim dalıdır. Bizim üzerinde çalıştığımız projede kullanılan biyometri; yüz özelliklerine bağlı insan tanıma, diğer bir deyiş ile yüz bulma ve yüz tanımadır. Yüz tanımadaki üç temel unsur şu şekildedir:

- **Kişi tanılama:** Veri tabanında bulunan bir kişinin kim olduğunu anlamak.

- **Kişi doğrulama:** Veri tabanı ile görüntüdeki kişinin yüzünün doğru olup olmadığını anlamak.

- **Kişi onaylama:** Kişinin önceden kayıtlı kişi olup olmadığını, kayıtlı ise kişinin doğruluğunun onaylanmasını sağlamak.

Yüz tanıma sisteminin çalışma mantığı daha önce öğrenilen yüzlerin karşılaştırılması ile gerçekleşir. Yüz tanıma sistemlerinin temelleri 1960'lı yılların başlarında Woody Bledsoe isimli bilgisayar bilimci tarafından atılmıştır. O zamanlarda yapılan yüz tanıma işlemi çok uzun sürerken günümüzde anlık olarak veriler tanınıp kişiyle eşleştirilebilmektedir. Modern yüz tanıma sistemlerine gelecek olursak; kullanılan algoritmalar geliştiği gibi, kameralar ve verileri işleyen işlemciler de gelişmiştir. Dolayısıyla verilerin algılanıp tanınması da kolaylaşmıştır.

Yüz tanıma sisteminde, kişilerin yüzleri bir kamera ile çekilir ve analiz edilir. Sistem; gözler, burun, ağız ve çene kenarlarındaki mesafeler de dâhil olmak üzere bütün yüz yapısını ölçer. Bu ölçümler bir veri tabanında saklanır ve kullanıcı kamera önüne geldiği zaman yapılacak karşılaştırmalar için kullanılır. Her yüz farklı karakteristik özelliklere sahip olduğu için karşılaştırmalarda hata payı, en aza indirilmiş olunur. İnsan yüzü yaklaşık 80 düğüm noktasına sahiptir. Yüz tanıma teknolojisiyle gözler arasındaki mesafe, burun genişliği, göz çukurlarının derinliği, elmacık kemiklerinin şekli, çene hattının uzunlukları vs. ölçülür. Bilgisayar yazılımları bu 80 düğüm yerine 15-20 noktaya gereksinim duyar. Uzmanlar dudaklar ve şakaklar arasındaki altın üçgene odaklanmıştır. Kilo alınsa, yaşlanmaya bağlı yüz hatları değişse, saç ve sakal uzasa ya da gözlük takılsa bile bu alan hep aynı kalır.

Şekil 2 Yüz tanıma güvenlik sistemi

1.1.3 Diğer Güvenlik Sistemleri

Kullanılan diğer güvenlik yöntemleri en basite indirgenecek olursa, 4-5 haneli parolaların kullanılması, RFID kartlar ile giriş yapma, ses ile güvenlik sistemi gibi örnekler verilebilir. Çeşitli çalışmalarda farklı güvenlik önlemleri gerekebileceği için birden fazla güvenlik sistemi mevcuttur.

1.2 Proje Amacı ve Gerekliliği

Projenin amacı elimizde olan imkânlar ile video, fotoğraf ve kamera üzerinden yüz tanıma güvenlik sistemi tasarlayarak güvenlik sağlamaktır.

1.2.1 Güvenlik Sistemleri ve Yüz Tanımalı Güvenlik Sistemleri Gerekliliği

İnsanlar evini, parasını, arabasını, verilerini vs. korumak için bir güvenlik sistemine ihtiyaç duymaktadır. Bu sistemler de zamanla gelişme kaydetmiştir ve güvenlik seviyesi artmıştır. Önceki yıllarda yaşanan güvenlik kırılma olayları günümüzde daha az seviyelerdedir. Güvenlik sistemlerinin tarihçesine baktığımızda basit güvenlik önlemlerinin dışında parmak izi okuyucusu, yüz tanıma sistemleri, biyometrik analiz yapan güvenlik sistemleri gelişmiş ve doğruluk oranı arttırılmıştır. Her insanın yüzü kendine özgü olduğundan dolayı yüz tanıma sistemleri de popüler hale gelmiştir. Telefonlarda, bilgisayarlarda, şehir güvenlik sistemleri ve bunun gibi yerlerde yüz tanımalı güvenlik sistemleri kullanılmaktadır.

1.2.2 Yüz Tanıma Sisteminin Avantajları

Yüz tanıma sisteminin sıklıkla kullanılabileceği yerlerde yüz tanıma sistemi hijyen, zaman tasarrufu, doğruluk oranı gibi yararları vardır. Ayrıca RFID kart kullanan şirketlerde başkasının yerine kart basma, anahtarı ya da şifresi olan bir kasanın başkası tarafından açılabilmesi gibi problemlerin önüne geçebilmesi en büyük avantajlarından biridir. Bir diğer avantajı ise hızlı tepki verip zaman kazandırmasıdır. Yeni teknoloji ile hayatımıza giren telefonlar saniyeler içerisinde yüzümüzü tanıyıp kilidi açabilecek seviyedelerdir.

1.3 Literatür Taraması

Yapılan literatür taramasında yüz tanıma sistemlerinin çeşitli amaçlarla kullanıldığı görülmüştür. İstanbul Aydın Üniversitesi öğretim görevlisi Zihni Kaya hocamız yüz tanıma sistemini sınavlar ile birleştirerek başkasının yerine sınava girme durumunu önlemeyi amaçlayan bir makale yayınlamıştır. [1] Ayşe ELDEM hocamızın Görüntü İşleme Teknikleriyle Yüz Algılama Sistemi Geliştirme [2] makalesinde sistemin nasıl çalıştığı hakkında bilgi alınmıştır. Taşınabilir bir sistem tasarımı istediğimizden dolayı Raspberry Pi gibi güçlü ve gelişmiş bir kart kullanımı tercih edilmiştir. Bu sistem tasarımı yapılırken Python dili kullanılmıştır. Yüz tanıma ve görüntü işleme yapabilmek için OpenCV ve Frontal_Face_Haarcascade kullanılmıştır.

2.Raspberry Pi ile Yüz Tanıma Fonksiyonlu Güvenlik Sistemi Bileşenleri

2.1 Yüz Tanıma Algoritmaları

Proje yapılırken Phyton yazılım dili ve OpenCV kullanılmıştır. Yüz tanıma işlemi için OpenCV' de Eigenfaces, Fisherfaces ve LBPH olmak üzere üç adet algoritma mevcuttur. Bu algoritmalar veri tabanında ki görüntülerle o anki tanımlanan yüz arasındaki korelasyona bakarak tahmin işlevini gerçekleştirmektedir. Bu algoritmaların karşılaştırma amaçlı kullanacağı yüz resimleri manuel olarak oluşturabilir ve xml formatında yazdırılabilmektedir. Ama bu işlem, binlerce veri söz konusu olduğunda zahmetli olacağı için yazılıma yaptırılmıştır. Yazılımın kişilerin yüzlerini alıp arka planda belirli bir veri tabanında depolamasına ve xml formatında bu resimleri organize etmesine "eğitim" denilmektedir. Yani yazılımın yüzlere özel kendini eğitmesi söz konusu olacaktır.

2.1.1 Eigenfaces

Veri tabanında ki tüm resimlerin ortalamasını alarak ortaya çıkan görüntünün ortalamadan farkını alır. Böylece her görüntü aslında bir diferansiyel görüntü gibi değerlendirilir.

2.1.2 Fisherfaces

Eigenfaces'e nazaran daha da gelişmiş bir yüz tanıma algoritmasıdır. Bir kişiye ait birden fazla fotoğrafı bir sınıfın üyeleri olarak kabul edip sınıflar arası tanıma işlemi yapmaya dayanır.

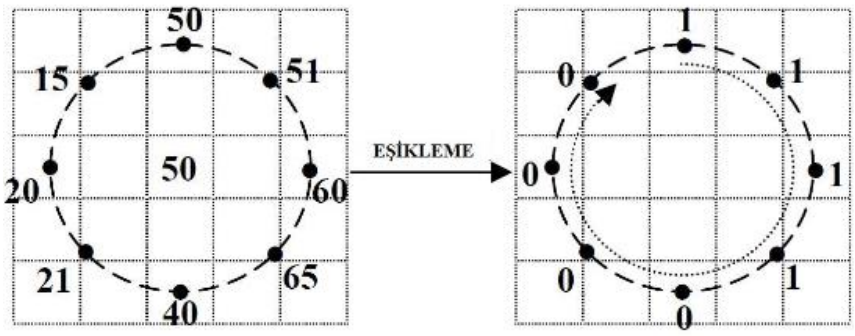
2.1.3 LBPH

Bir noktanın etrafındaki piksellerin yoğunluk değerlerine göre oluşan ikili(binary) örüntülerden yaratılır ve bu örüntüler üzerinde işlemler gerçekleştirerek nesneyi aramaktadır. [3]

Proje gerçekleştirirken LBPH algoritması kullanılmıştır.

2.2 LBPH Algoritması

Local Binary Pattern (LBP) algoritması yerel ikili örüntüleri kullanarak yüz tanıma işlemi yapmaktadır. Bu operatör görüntünün 3×3 pencerelerin ortasındaki piksel değeri eşik seviyesi olarak seçilerek birbirlerine göre karşılaştırılması sonucu ikili değerlerin atanması temeline dayanmaktadır. Oluşturulan ikili sayı dizisine LBP kodu denir ve bu kod ile görüntüdeki farklı tipteki özellikleri belirlemek mümkün hale gelir. Örneğin; kenarlar, köşeler, aydınlık veya karanlık bölgeler, çizgi bölgeleri gibi. Fakat 3×3'lük operatörler baskın özellikleri yakalayabilmek için etkin olmayabilir. Dolayısıyla probleme göre operatörün uzunluğu belirlenmeli veya adaptif olarak atanmalıdır. Bunun yerine dairesel olarak da LBP tanımlamaları yapılmaktadır. Burada tanımlanması gereken iki parametre bulunmaktadır. Bunlar örnek nokta sayısı P ve simetrik dairesel komşuluğun yarıçapı ölçüsü R olarak belirlenir [4]



Şekil 3 LBP Algoritması

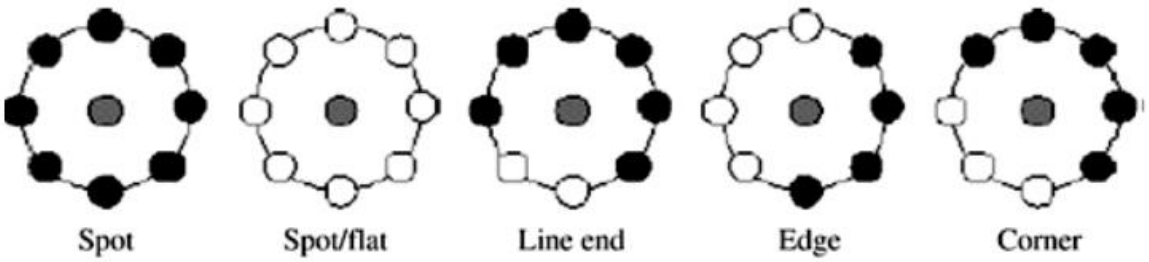
LBP operatörü LBPP, R olarak oluşturulan P tane komşu piksel setinden uygun 2P farklı çıkış değeri tanımlanır. Ayrıca LBP sonucu elde edilen ikili LBP kodunda eğer iki ya da daha az bir geçişi varsa buna uniform LBP denir. Örneğin; 00000000, 00111000, 11100001 uniform LBP kodudur. LBP ile etiketlenen görüntünün yeni histogramı aşağıdaki gibi ifade edilir.

$$LBP_{P,R}(x_c, y_c) = \sum_{P=0}^{P-1} s(i_P, i_C)2^P$$

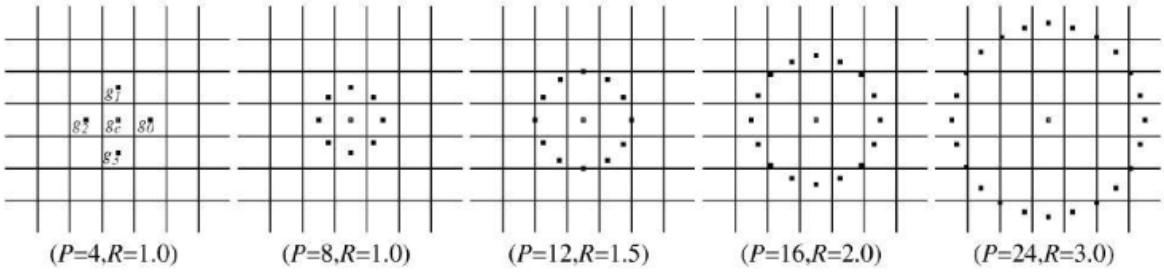
Burada verilen (Xc ve Yc) pikselin yerini İp ve İc merkez pikselin gri seviyesini ifade eder. s fonksiyonu aşağıdaki gibi tanımlanmaktadır.[4]

$$s(x) = \begin{cases} 1, eğer\ x \geq 0 \\ 0, eğer\ x < 0 \end{cases}$$

Ayrıca LBP histogramı, görüntünün içindeki yerel mikro örüntülerin dağılımı ile ilgili (örneğin kenarlar, aydınlık bölgeler vs. gibi) bilgiyi de içermektedir. Böylece görüntünün karakteristiği istatistiksel olarak tanımlanabilir.[4]



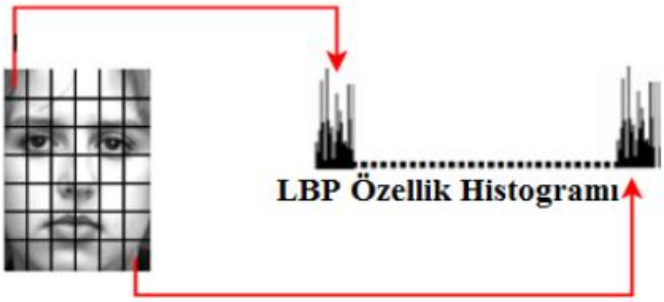
Şekil 4 LBP örüntülerinin görüntü içindeki anlamları [4]



Şekil 5 P örnek sayısı ve R simetrik dairesel yarıçap çeşitleri [3]

LBP histogramı lokasyon bilgisi olmaksızın tüm görüntü üzerinden mikro örüntüleri hesaplamaktadır. LBP histogramının çıkardığı R0, R1, R2,...,Rm gibi küçük bölgelere görüntüsü şekil bilgisi eşit olarak dağılmaktadır. Her alt bölge için çıkarılan LBP özellikleri yeni konum bilgisini aşağıdaki gibi tanımaktadır. [4]

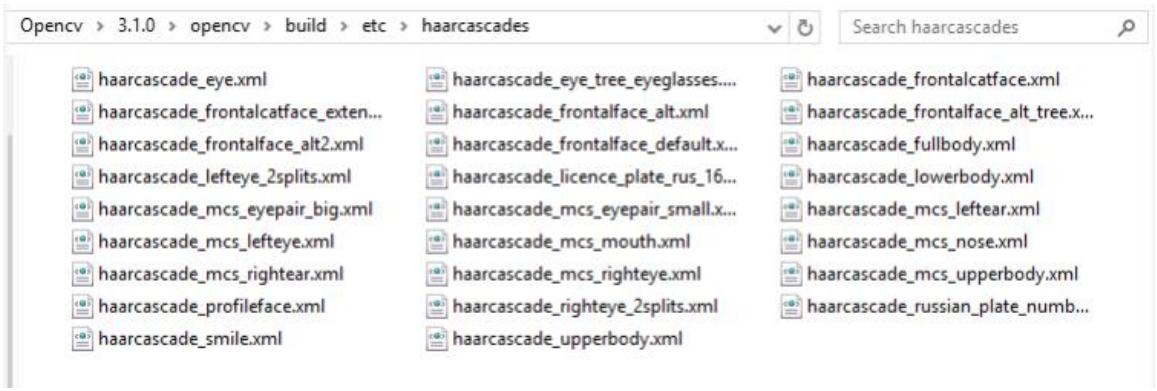
$$H_{i,j} = \sum_{x,y} \mathrm{I}(f_i(x,y) = i), \mathrm{I}\left((x,y)\epsilon R_j\right), \qquad i = 0,\dots,n-1, \;\; j = 0,\dots,m-1$$



Şekil 6 LBP ile histogramın elde edilmesi [3]

2.3 HaarCascade Sınıflandırıcısı

Haarcascade sınıflandırıcısı, OpenCV ve görüntü işlemede sık sık kullanılan bir eşleştirme yöntemidir. OpenCV'nin github sayfasında çeşitli olaylar için Haarcascade mevcuttur. Bunlardan bazılarında yüz tanıma, göz tanıma, ağız tanıma, araç plakası tanıma gibi haarcascadeler mevcuttur. Projede kullanılan Haarcascade ise haarcascade_frontalface_default.xml yani ön yüzü tanıyan sınıflandırıcıdır. Eğer farklı amaçlar için haarcascade kullanılması gerekirse sıfırdan sınıflandırıcı eğitilebilir. Bu sınıflandırma yönteminde negatif ve pozitif fotoğraflar kullanılmaktadır.[6]



Şekil 7 Haarcascade Listesi

2.4 Devre Bileşenleri

2.4.1 Kontrol Kartı

Raspberry Pi, üzerinde bulunan portlar ve içerisine yüklenebilen işletim sistemiyle kontrol kartları arasında kendisini ön plana çıkartan bir karttır. Hızlı işlem yapması ve diğer kontrol kartlarına göre fiyatının uygun olması projede kullanılması için önemli nedenlerden birisidir. Raspberry Pi, kontrol kartı üzerine takılan LCD ekran veya uzak masaüstü programı olan VNC viewer ile kolayca kontrol edilebilmektedir. Piyasada bulunan görüntü işleme yapılabilen kartlardan bazıları şunlardır:

- Asus Tinker
- LePotato
- Rock64
- Nvidia Jetson
- Intel Galileo Gen 2

İnternette Raspberry Pi ile alakalı daha çok makale ve bilgi olması da proje için bu kartın seçilme sebebini arttırmıştır.

2.4.1.1 Raspberry Pi Model 3B+

Projede Raspberry Pi Model 3B+ ve RaspiCam kullanılmıştır. Raspberry Pi içerisinde CPU, GPU, depolama birimi, ram, HDMI portu, Ethernet girişi, USB portu vb. bulunduran bir geliştirme kartıdır. Raspberry Pi kurulumu yapılırken RasbianOS kullanılmıştır.

►Broadcom BCM2835 (700MHz, ARM1176JZF-S tabanlı) İşlemci,

►Broadcom VideoCore IV (OpenGL ES 2.0, 1080p destekli) Grafik İşlemci,

►Model A'da 256MB, Model B'de 512MB Ram,

►USB 2.0 (Model B'de 2 tane, Model A'da 1 tane bulunmaktadır),

►HDMI yuvası,

►SD Kart Okuyucu,

►3.5mm ses jakı,

►RCA Video Çıkışı,

►CSI Bağlantısı

►10/100 Ethernet (Model B'de bulunmaktadır),

►İşletim sistemi: Debian GNU/Linux, Fedora, Arch Linux ve türevleri,

►Düşük Seviye Çevre Birimleri: 8 adet GPIO, UART, I²C bus, SPI bus'la birlikte iki Chip Select, +3.3 V, +5 V, ground

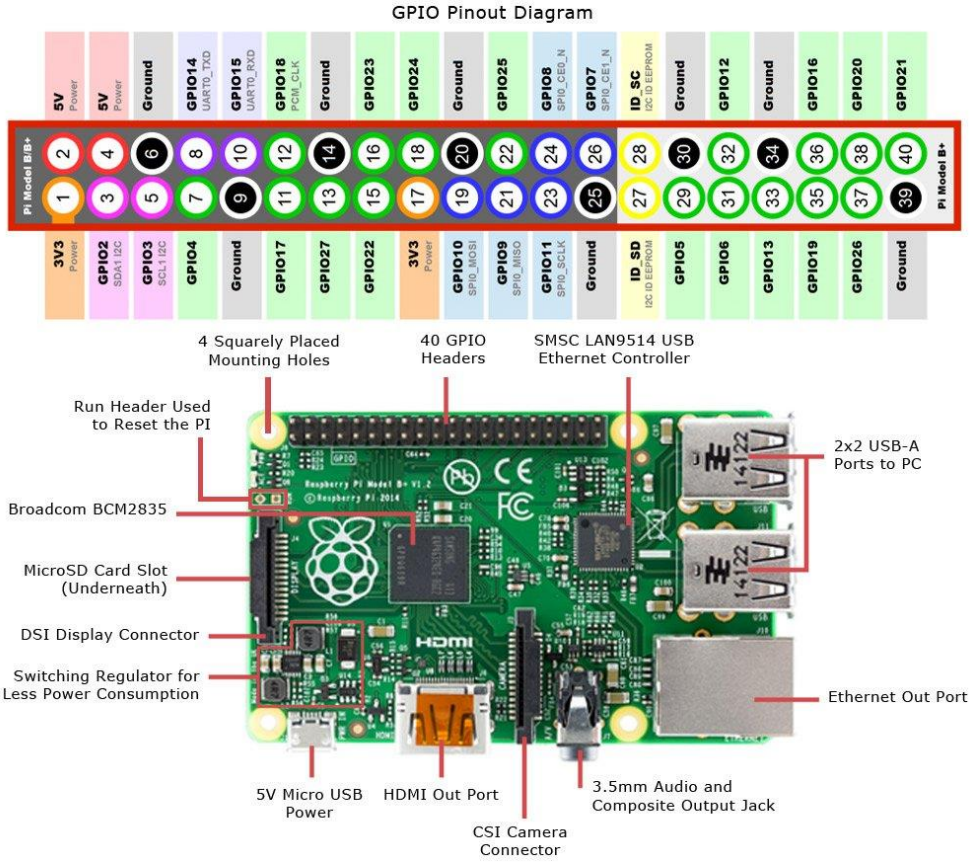
►45 gram ağırlığında,

►Model A 1.5W, Model B ise 3.5W güç tüketmektedir,

►Çalışma gerilimi: +5V DC

Teknik özelliklerine sahiptir. [7]

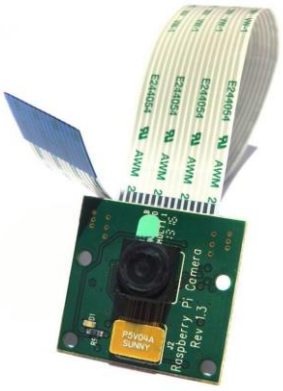
Şekil 8 Raspberry Pi



Şekil 9 Raspberry Pi Model 3B+ Pin Diyagramı

Raspberry Pi üzerinde 40 adet pin bulundurmaktadır. Pinlerin görevleri üzerlerinde yazmaktadır. Kontrol kartı 5V 2.5A adaptör ile rahatça çalışmaktadır. Akım gücü düşük olan adaptörlerde istenilen performans alınamamaktadır. Üzerinde dâhili olarak bulunan Wİ-Fİ ve Ethernet portu internet erişimi sağlamak için büyük bir avantaj sağlamaktadır. Kart üzerinde bulunan CSI kamera portu Raspberry Pi için özel üretilmiş kameranın bağlantısı için gereklidir. Projede yüz tanıma sağlayabilmek için CSI portunu kullandığımız Raspi Cam kullanılmıştır.

2.4.2 RaspiCam



Şekil 10 Raspi Cam

- 8 mega piksel sabit odak noktalı
- 1080p, 720p60 ve VGA90 destekli
- Sony IMX219PQ CMOS görüntü algılayıcı
- 15-pin şerit kablo

Özelliklerine sahip olan kamera netlik ve kalite bakımından beklenilen performansı sağlamaktadır. Projede kamera performansını arttırmak için kamera çevresine isteğe bağlı açılabilen NeoPixel adreslenebilir halka LED koyulmuştur. LED ile yapılan yüz algılama işlemlerinde yüz tanıma oranı %10 seviyesinde artmaktadır. Kamera içerisinde bulunan otomatik poz ayarlama, beyaz dengesi ayarlama, band filtresi, siyah dengesi ayarlama almış olduğumuz görüntünün kalitesini arttırmaktadır. -20 derece ile +60 derece sıcaklık arasında çalışabilen kamera, tasarlanan sistem içerisinde oda sıcaklığında çalışmaktadır. Bu sebeple soğutma gereksinimi olmamıştır.

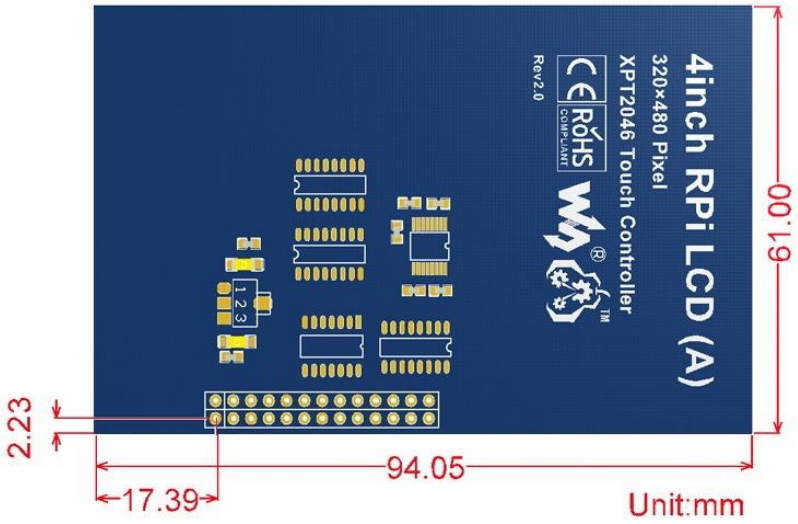
2.4.3 WaveShare 4 İnç RPİ LCD ekran

Şekil 11 WaveShare LCD Arka

Şekil 12 WaveShare LCD Ön

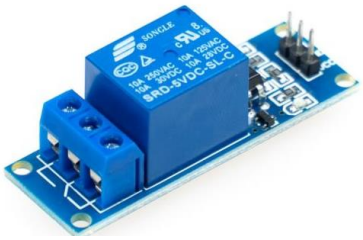
- 5V ile çalışır. Harici güç kaynağına ihtiyaç duymaz. Gücünü Raspberry Pi üzerinden alır.
- Ekran Oranı: 8:5
- Çözünürlük: 800x480
- Ekran Ebat: 4"
- Tüm Raspberry Pi Modelleri ile Uyumlu
- LED Arka ışık
- SPI Arayüz
- Resistif Dokunmatik Panel
- XPT2046 Dokunmatik Kontrolcüsü [8]

Projede, bu ekran yüz tanıma işlemi yapılırken kişinin kendini görebilmesi için tercih edilmiştir. Harici bir güç kaynağı istememesi ve resistif dokunmatik olması projede kullanılmasını sağlamıştır.



Şekil 13 WaveShare LCD Boyutları

2.4.4 Röle Kartı



Şekil 14 Role Kartı

Projede kullanılan röle kartı 5V ile tetiklenmektedir.30VDC veya 220VAC gerilimde 10 ampere kadar akımı anahtarlayabilmektedir. Raspberry Pi üzerindeki GPIO pinlerinden gelen sinyal ile tetiklenen röle selenoid kilidi aktifleştirmektedir.

2.4.5 Selenoid Kilit

Şekil 15 Selenoid Kilit

12 Volt ile çalışan selenoid kilit röleye bağlanarak kontrol edilmektedir. Tasarımda daha kolay kullanılabilmesi için bağlantı pinleri USB giriş olarak değiştirilmiştir.

2.4.6 NeoPixel Adreslenebilir LED ve ATtiny85

Yüz tanıma işlemi yapılacağı sırada doğruluğu arttırmak için ışık gerekebiliyor bu sebeple tasarım yapılırken kameranın çevresine halka LED eklenmiştir ve bu LED ATtiny85 kartı ile kontrol edilmektedir.

Şekil 17 NeoPixel Halka LED

Şekil 16 ATtiny85 Kontrol Kartı

2.5 Kullanılan Programlar

Proje yapım aşamasında 4 farklı program kullanılmıştır.

- PyCharm
- SolidWorks
- VNC Viewer
- Arduino IDE

2.5.1 PyCharm

Proje ilk başta Windows üzerinde yapılmıştır. Testler bittikten sonra Raspberry Pi üzerinde çalıştırılmıştır. Bu yazılan kodların testinin yapılması ve geliştirilmesinin yapılması da Windows üzerinde çalışan PyCharm Python derleyicisi ile yapılmıştır.

2.5.2 SolidWorks

Birleştirilen devrelerin kullanıcının kolayca kullanabileceği şekilde olması için bir kutu içerisinde olması gerekmektedir. Bu sebeple SolidWorks üzerinde elimizde bulunan parçaların ölçülerine göre bir kutu tasarımı yapılmıştır.

2.5.3 VNC Viewer

VNC viewer, Raspberry Pi ile bilgisayar arasında bağlantı kurmamıza yarayan uzak masaüstü yazılımıdır. Projeye uzaktan bağlanıp işlem yapmak için sık sık kullanılan bir programdır.

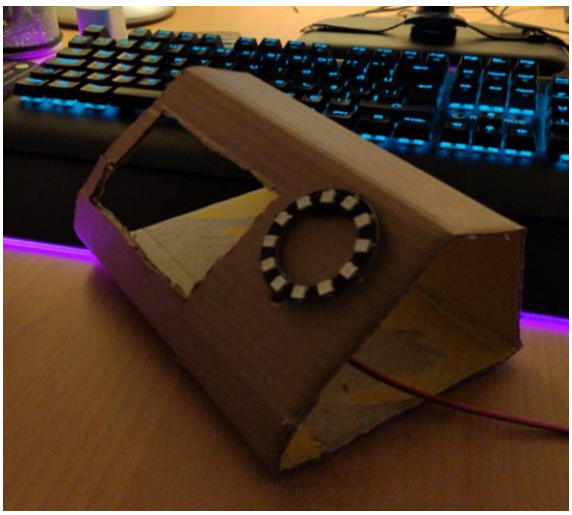
2.5.4 Arduino IDE

NeoPixel ve ATtiny85 kartlarının kodlaması ve buton ile kontrol edilmesini sağlamak için gerekli kodlama Arduino IDE üzerinde yapılmıştır.

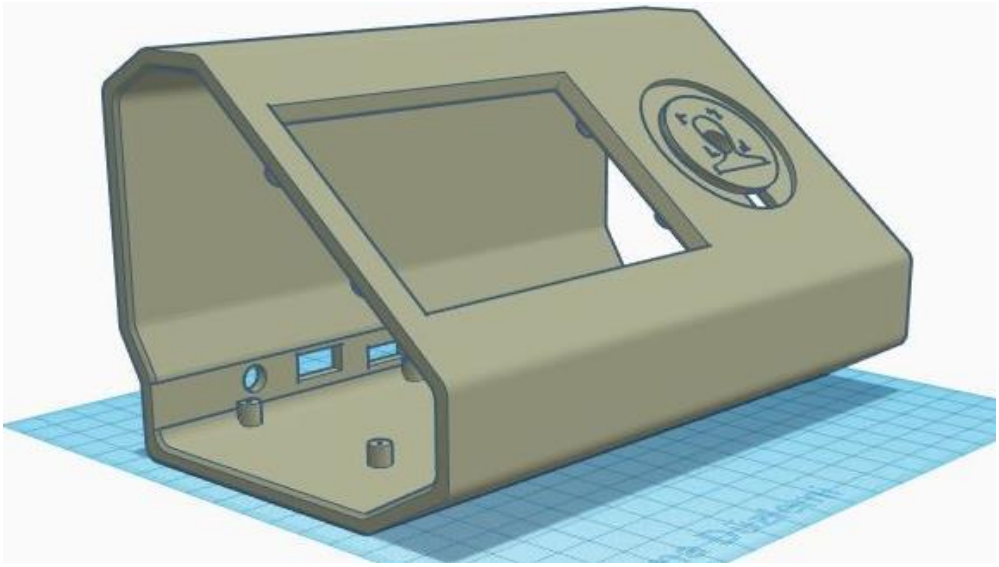
3. Raspberry Pi ile Yüz Tanıma Fonksiyonlu Güvenlik Sistemi

3.1 Dış Kutunun Tasarımı

Dış kutunun tasarımı yapılırken önce boyutlarını ölçebilmek amacı ile kartondan prototip hazırlanarak ölçümleri yapılmıştır. Yapılan ölçümlere göre SolidWorks'de 3D çizimleri yapılmıştır. Ardından 3D yazıcı ile çıktısı alınmıştır. Kutu tasarımı yapılırken kullanıcının zorlanmaması ve pratiklik hedeflenmiştir.

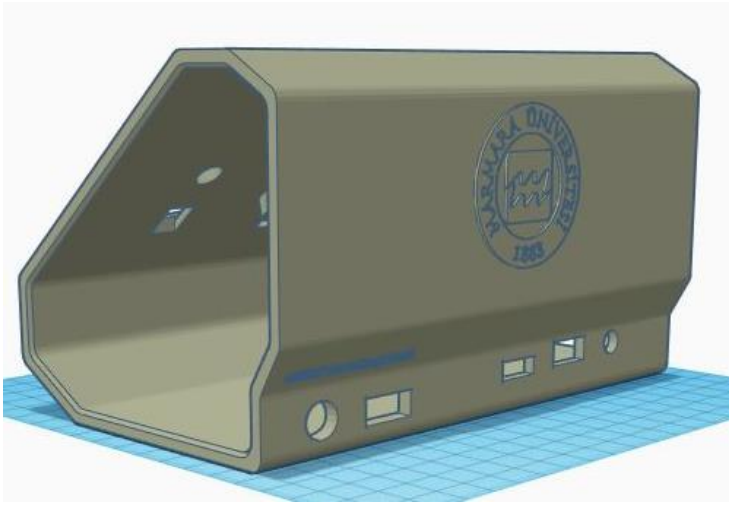


Şekil 18 İlk Prototip



Şekil 19 3D Çizim Önden Görünüş

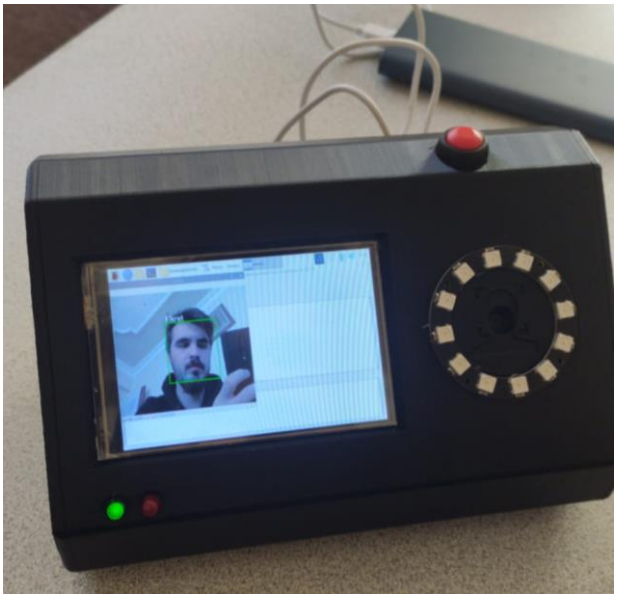
Ön tarafta bulunan büyük boşluğa LCD ekran yanındaki halka kısma ise LED yerleştirilmiştir. Kamera ise arkadan bağlanmıştır ve delik olan kısımdan görüntü almaktadır.



Şekil 20 3D Çizim Arkadan Görünüş

Kutunun arka kısmında yer alan 12Volt dc gerilim giriş yeri ve sağında selenoid kilit için tasarlanan USB girişi bulunmaktadır. İsteğe bağlı olarak yüz tanıma ile farklı cihazlar da kontrol edilebilir. En sağ kısımda ise Raspberry Pi güç girişi, HDMI çıkışı, 3.5mm ses çıkışı bulunmaktadır.

Şekil 21 3D Baskı



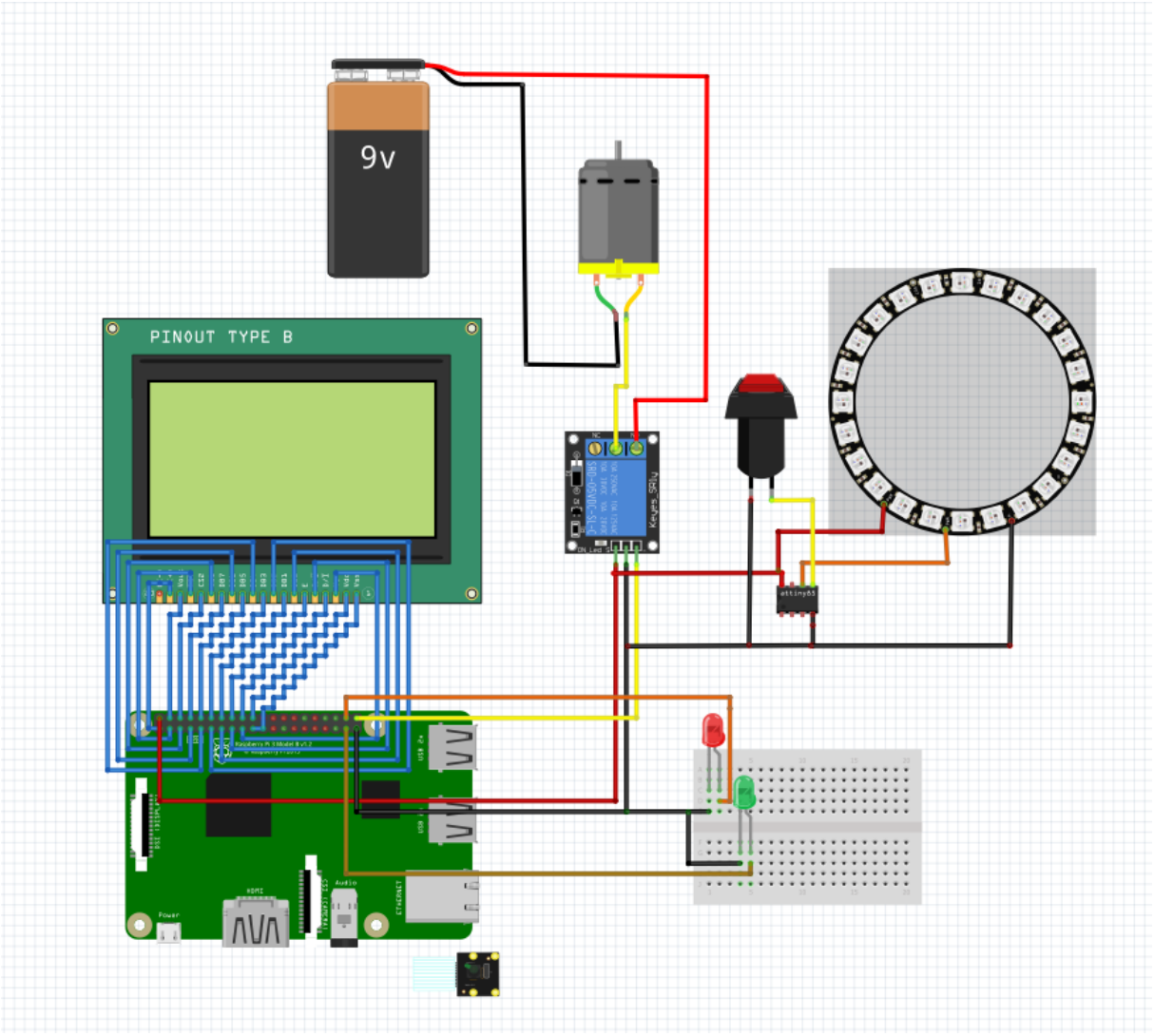
Şekil 22 Tasarımın Bitmiş Hali Önden Görünüş

Şekil 23 Tasarımın Bitmiş Hali Arkadan Görünüş

Şekil 24 Tasarımın Bitmiş Hali Yandan Görünüş

3.2 Elektriksel Bağlantı

Elektriksel bağlantı olarak LCD ekran, GPIO pinlerinden 26 tanesini kullanmaktadır. Bu bağlantılar LCD üretiminde tam denk gelecek şekilde üretilmiştir fakat kutu tasarımında Raspberry Pi aşağıda kalması gerektiği için jumper kablo yardımı ile 45°lik bir açıyla yerleştirdik.



Şekil 25 Devre Şeması

Devre şemasını göstermek için Fritzing üzerinde bir çalışma yapılmıştır. Raspberry Pi üzerindeki 40.Pin yani GPIO21 röleyi kontrol etmektedir. Yüz tanıma işleminin başarılı olup olmadığını anlayabilmemiz için iki tane LED bulunmaktadır. Kırmızı LED sinyali 38.Pin üzerinden alırken yeşil LED 37.Pin üzerinden almaktadır. Tasarımda da görülebileceği üzere ön tarafın sağında bulunan halka LED'i ATtiny85 ile kontrol edilmektedir. P0 pinine butondan gelen sinyal halka LED'in açılıp kapanmasını sağlamaktadır.

Şekil 26 Arka Giriş Portları

Şekil 26 da görüldüğü gibi en sol kısımda 12 Volt giriş jakı bulunmaktadır. Bu jak röleye bağlanmaktadır. Hemen sağında bulunan USB girişi ise selenoid kilidin bağlantısından sorumludur. Yukarıda görülen buton ise ön tarafta bulunan halka LED lerin açılıp kapanmasından sorumludur.

3.3 Kodlama

Kodlar Python dilinde yazılmıştır.

3.2.1 Tanımlamalar

Kodlama yapılırken birden fazla uygulama yapılmıştır. Bunlar sırasıyla:

- Face_recognition.py
- Data_training.py
- photo_recognition.py
- New_user.py
- Video_recognition.py

Bu uygulamaları yazarken OpenCV, numpy, pickle, RPİ, GPIO kütüphaneleri kullanılmıştır. Öncelikle Raspberry Pi üzerine Python 3 yüklemesi yapılmıştır. Ardından gerekli kütüphaneler yüklenmiştir.

```
sudo apt-get install python3 python3-setuptools python3-dev -y
```

Komutu ile python3 yüklemesi yapıldı.

```
sudo apt-get install build-essential cmake cmake-curses-gui pkg-config
```

Gerekli kütüphane dosyalarının yüklenmesi için gereken komutlar [10]

```
sudo apt-get install \
  libjpeg-dev \
  libtiff5-dev \
  libjasper-dev \
  libpng12-dev \
  libavcodec-dev \
  libavformat-dev \
  libswscale-dev \
  libeigen3-dev \
  libxvidcore-dev \
  libx264-dev \
  libgtk2.0-dev
```

```
wget https://github.com/opencv/opencv/archive/3.2.0.zip -O opencv_source.zip

wget https://github.com/opencv/opencv_contrib/archive/3.2.0.zip -O opencv_contrib.zip
```

Raspberry Pi'a github üzerinde bulunan repoları indirmemizi sağlayan komut

```
unzip opencv_source.zip
unzip opencv_contrib.zip
```

İndirilen repolar zip dosyasından çıkartılır.

```
cd /home/pi/usbmem/opencv-3.2.0
mkdir build
cd build
```

OpenCV dosyalarına girip performans artışı sağlamak için Cmake ayarlarının yapılması gerekmektedir.

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
        -D CMAKE_INSTALL_PREFIX=/usr/local \
        -D BUILD_WITH_DEBUG_INFO=OFF \
        -D BUILD_DOCS=OFF \
        -D BUILD_EXAMPLES=OFF \
        -D BUILD_TESTS=OFF \
        -D BUILD_opencv_ts=OFF \
        -D BUILD_PERF_TESTS=OFF \
        -D INSTALL_C_EXAMPLES=ON \
        -D INSTALL_PYTHON_EXAMPLES=ON \
        -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib-3.2.0/modules \
        -D ENABLE_NEON=ON \
        -D WITH_LIBV4L=ON \
```

Komutları ile OpenCV kurulumu sağlanmaktadır ve gereken ayarlar yapılmış olmaktadır. [13]

OpenCV Kütüphanesi: Tüm görüntü işleme işlemlerinde kullanılan ve bu alanın öncüsü bir kütüphanedir. CUDA, Intel, ARM gibi şirketlerin beraber çalışmaları sonucunda ortaya çıkmış bir kütüphanedir. Projede yüz algılama ve tanıma işlemleri bu kütüphane sayesinde yapılmaktadır.

Pickle Kütüphanesi: Pickle, bir modül olarak Python nesnelerinin süreçler arasında kaydedilmesini sağlar. Pickle modülü, boole'lar, dizeler ve bayt dizileri, listeler, sözlükler, işlevler gibi veri türlerini depolayabilmektedir.

Numpy Kütüphanesi: NumPy (Numerical Python) bilimsel hesaplamaları hızlı bir şekilde yapmamızı sağlayan bir matematik kütüphanesidir.

RPİ.GPIO Kütüphanesi: Raspberry Pi üzerinde bulunan GPIO pinlerinin kullanımı için gereken kütüphanedir.

3.2.2 Fonksiyonlar

Yazılan uygulamalar birbirleri ile entegre şekilde çalışmaktadırlar. İlk başta kullanıcının New_User.py ile örnek fotoğrafları alınarak veri seti oluşturulur. Ardından Data_training.py ile kaydedilen görüntüler trainer.yml dosyasına öğretilir. Face_recognition.py uygulaması ise kameradan gelen görüntü ile trainer.yml dosyasında olan veri ile karşılaştırır. En sonunda doğru sonucu ekrana yansıtmaktadır. Photo_recognition.py ile sisteme tanıtılan fotoğraftaki yüzün eğitilen veriler ile karşılaştırılıp çıktısının alınması sağlanır. [14]

New_User.py

Gerekli olan kütüphaneler dâhil ediliyor.

```
import cv2
import numpy as np
import os
import sys
```

Cv2.VideoCapture(0) ile kamera üzerinden görüntü alışverişi sağlanıyor. Camera.set ile kamera çözünürlüğü ayarlanmaktadır.

```
camera = cv2.VideoCapture(0)
camera.set(3,640)
camera.set(4,480)
```

```
faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
```

Ön yüz tanıma haarcascade dosyası tanımlanmaktadır.

```
name = input("İsim giriniz =  ")
dirName = "./images/" + name
print(dirName)
if not os.path.exists(dirName):
    os.makedirs(dirName)
    print("Klasör oluştruldu")
else:
    print("İsim önceden kullanılmış")
    sys.exit()
count = 1
```

Kullanıcı ismi sorulmakta ve bu isimde bir klasör açılmaktadır. Eğer aynı isimde bir klasör varsa program tekrardan aynı klasörün içine fotoğrafları kaydetmemekte ve programdan çıkış yapmaktadır.

```
while True:
        if count > 30:
        break
    ret, im =camera.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray, scaleFactor = 1.5, minNeighbors = 5)
    for (x, y, w, h) in faces:
        roiGray = gray[y:y+h, x:x+w]
        fileName = dirName + "/" + name + str(count) + ".jpg"
        cv2.imwrite(fileName, roiGray)
        cv2.imshow("face", roiGray)
        cv2.rectangle(im, (x, y), (x+w, y+h), (0, 255, 0), 2)
        count += 1
```

30 tane fotoğraf çekilmekte ve program görevini tamamlamaktadır.

Burada yapılan işlem kameradan okunan görüntü im değişkenine atanıyor. Görüntü OpenCV kütüphanesinde bulunan cvtColor komutu ile görüntüyü griye çevrilmektedir. Fotoğrafın griye çevrilmesinin sebebi RGB olan renk düzleminde bakmamız gereken pixel sayısı artmasıdır. Bu durum da çalışma performansını kötü yönde etkileyecektir. Yapılan griye çevirme işleminden sonra faceCascade.detectMultiScale fonksiyonu ile yüz algılama işleminin kalitesi ve tutarlılığı ayarlanmaktadır. Devamında yapılan işlemler algılanan yüzün fotoğrafının kaydedilmesi ve isimlendirmesini sağlamaktadır. [11][12]

Şekil 27 Yüz verileri

Yapılan veri oluşturma işleminden sonra alınan verilerin makineye öğretilmesi gerekmektedir. Bu işlem de Data_training.py ile sağlanmaktadır.

Data_training.py

Bir önceki uygulamada kullanılan kütüphaneler bu programda da import edilmiştir.[13]

```
faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
recognizer = cv2.face.LBPHFaceRecognizer_create()
```

OpenCV paketinde bulunan LBPH (YEREL İKİLİ PATTERNS HISTOGRAMS) yüz tanıma algoritması kullanılmıştır. Algoritmanın çalışabilmesi için yüz tanıma sınıfı oluşturulmaktadır.

```
baseDir = os.path.dirname(os.path.abspath(__file__))
imageDir = os.path.join(baseDir, "images")
```

New_User.py programı ile kaydedilen fotoğrafların dizinden çekilmesi işlemi yapılmıştır.

```
currentId = 1
labelIds = {}
for root, dirs, files in os.walk(imageDir):
    print(root, dirs, files)
    for file in files:
        print(file)
        if file.endswith("png") or file.endswith("jpg"):
            path = os.path.join(root, file)
            label = os.path.basename(root)
            print(label)
```

imageDir olarak tanımlanan veri dizisi içerisinde dolaşarak fotoğraflar aranmaktadır. Eğer dosya uzantısı png veya jpg ise numaralandırma yapılmaktadır.

```
    if not label in labelIds:
                labelIds[label] = currentId
                print(labelIds)
                currentId += 1
```

Eğer fotoğrafta etiket yoksa ilk başta belirlenen currentId yani 1 değişkeni labelIds[Lablel] dizisine atanmakta ardından başka bir kullanıcı kaydı olabileceği için currentId bir arttırılmaktadır.

```
id_ = labelIds[label]
            pilImage = Image.open(path).convert("L")
            imageArray = np.array(pilImage, "uint8")
            faces = faceCascade.detectMultiScale(imageArray, scaleFactor=1.1, minNeighbors=5)
            for (x, y, w, h) in faces:
                roi = imageArray[y:y+h, x:x+w]
                xTrain.append(roi)
                yLabels.append(id_)
```

Doğru görüntülerle işlem yapıldığından emin olmak için yüz algılama işlemi tekrardan yapılmaktadır.

```
with open("labels", "wb") as f:
    pickle.dump(labelIds, f)
    f.close()
recognizer.train(xTrain, np.array(yLabels))
recognizer.save("trainer.yml")
print(labelIds)
```

Dizin isimleri ve etiketleri pickle komutu sayesinde saklanmaktadır. Verileri eğitip kayıt işlemini tamamladıktan sonra program tamamlanmaktadır.

Yapılan bu iki işlem ardından makinenin öğrendiği veriler kullanıma hazırdır. Bu verileri kullanabilmek için Face_recognition.py programı kullanılmaktadır.

Face_recognation.py

Yapılan yüz tanıma işleminin görünen kısmı bu programda sağlanmaktadır. Bir önceki programlarda olduğu gibi aynı kütüphaneler bu programda da kullanılmıştır.

```
id = 0

names = ['None', 'Fikret', 'Hakan', 'A', 'B', 'C']
```

Etiketlenen verilerin karşılığını bulması için names dizisi içerisine kişilerin id numaraları ile eşleşecek isimler sırayla yazılmıştır. Data_training.py programında currentId 1 den başladığı için names dizisinin ilk elemanı none olarak alınmıştır.

```
relay_pin = 38

led_pin = 40

led2_pin = 37
```

Röleyi tetikleyecek olan pin numarası ve yüz tanımanın doğru yanlışlığına göre yanacak olan LED pinleri belirlenmiştir. Belirlenen pin numaraları kart üzerindeki sayıya göre belirlenmiştir. Bu belirleme işlemi GPIO. setmode(GPIO. BOARD) ile yapılmaktadır.

```
GPIO.setup(relay_pin, GPIO.OUT)
GPIO.setup(led_pin, GPIO.OUT)
GPIO.setup(led2_pin, GPIO.OUT)
GPIO.output(relay_pin, True)
GPIO.output(led_pin, True)
GPIO.output(led2_pin, True)
```

GPIO pinlerinin ilk durumları ve giriş çıkış durumları belirlenmiştir.

```
with open('labels', 'rb') as f:
    dicti = pickle.load(f)
    f.close()
```

Hafızaya alınan etiket bilgileri bu programda okunmaktadır.

```
camera = cv2.VideoCapture(0)
camera.set(3,640)
camera.set(4,480)
minW = 0.1*camera.get(3)
minH = 0.1*camera.get(4)
```

Kamera çözünürlük ayarları yapılmıştır.[14]

```
path = os.path.dirname(os.path.abspath(__file__))
faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read("trainer.yml")
```

Eğitilen yüz verisi recognizer.read(“trainer.yml”) fonksiyonu ile okunmuştur. LBPH yüz tanıma sınıflandırması oluşturulmuştur.[15][16]

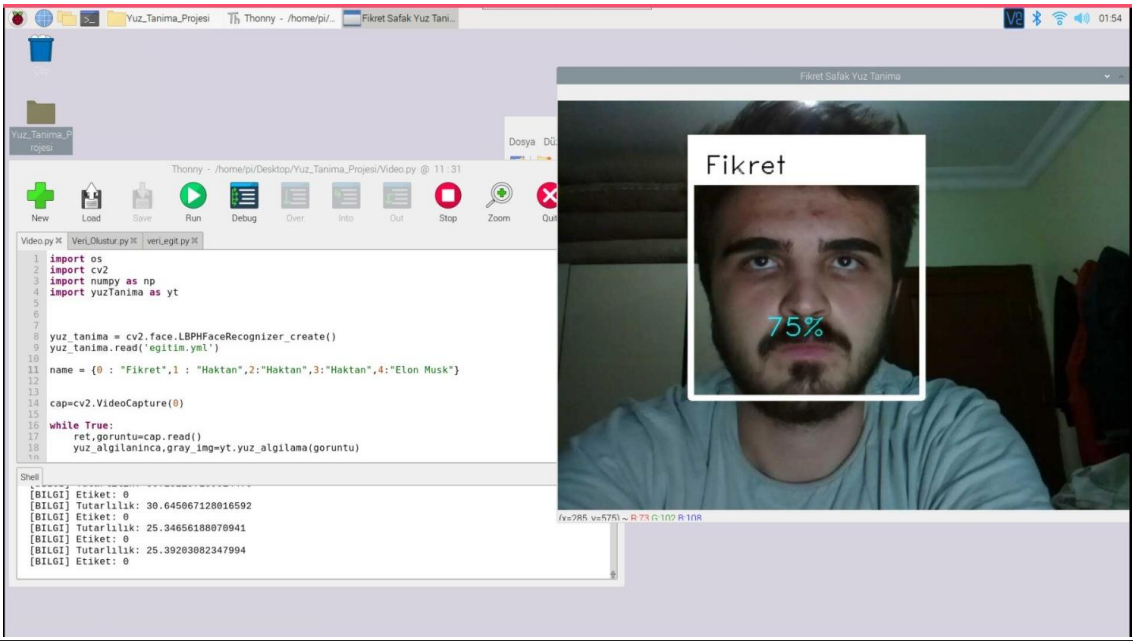
```
while True:
    ret, im =camera.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100),flags=cv2.CASCADE_SCALE_IMAGE)
    for (x, y, w, h) in faces:
        cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 2)
        id, confidence = recognizer.predict(gray[y:y + h, x:x + w])
```

New_user.py de yapılan yüz algılama işlemi tekrardan yapılmaktadır. Kameradan gelen görüntü griye çevrildikten sonra yüz algılanmaktadır. Eğer yüz algılanmışsa yüzü kare içine alıp hata oranını yazma komutu devreye alınmıştır.[16]

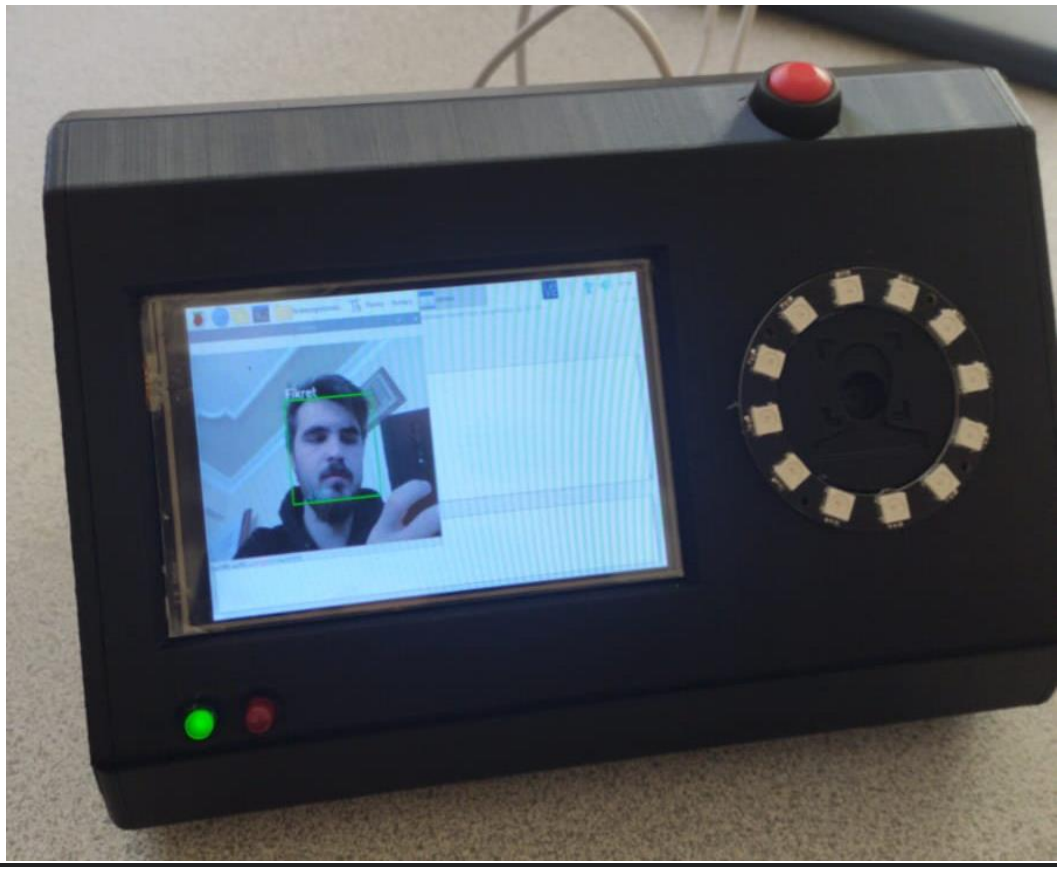
```
if (confidence < 60):
            GPIO.output(relay_pin,False)
            GPIO.output(led_pin,False)
            GPIO.output(led2_pin,True)
            id = names[id]
            confidence = "  {0}%".format(round(100 - confidence))
        else:
            GPIO.output(relay_pin,True)
            GPIO.output(led_pin,True)
            GPIO.output(led2_pin,False)
            id = "Bilinmiyor"
            confidence = "  {0}%".format(round(100 - confidence))
```

Eğer hata oranı 60 dan düşük ise röle aktifleşmekte ve yeşil LED yanmaktadır. Names dizisinde bulunan isimler ile id numarası eşleştikten sonra görüntü üzerinde gösterilecektir.

Eğer hata oranı fazla ise yani eğitilmemiş bir kişi sisteme bakıyorsa röle açılmayacak, kırmızı LED yanacak ve gözüken isim kısmında “Bilinmiyor” yazacaktır.



Şekil 28 Raspberry Pi üzerinde test



Şekil 29 Başarılı Yüz tanıma işlemi

Photo_recognition.py

Diğer programlardan farklı olarak görüntü kamera üzerinden değil bir dosya üzerinden alınmaktadır. Yüz tanıma işlemi aşamaları aynı şekilde bu programda da sağlanmaktadır.

```
im =cv2.imread('test-fotolari/1.jpg')
```

Sadece kaynak kamera üzerinden değil fotoğraf üzerinden de alınmıştır.

Video_recognition.py

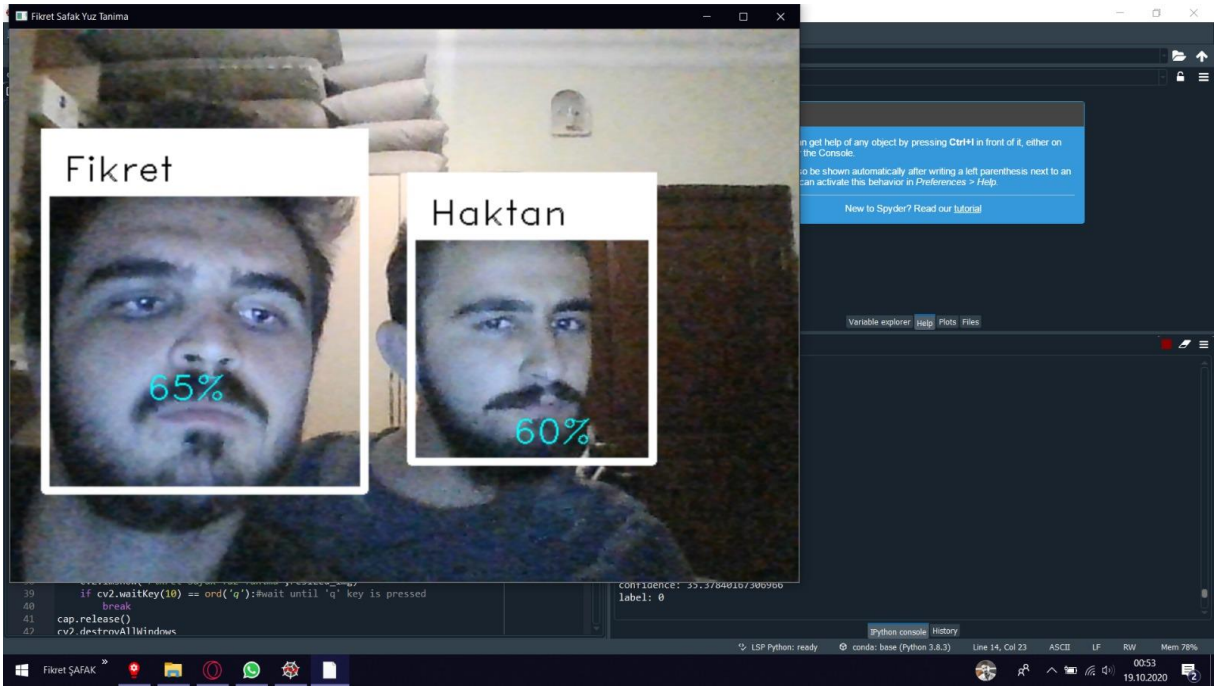
Fotoğraftan tanıma uygulamasında olduğu gibi kaynak dosyası değiştirilmektedir.

```
okunanVideo = cv2.VideoCapture('test-fotolari/video.avi')
```

Okunacak videonun önce yakalanması gerekmektedir. OpenCV kütüphanesinde bulunan VideoCapture fonksiyonu hem kamera üzerinden görüntü alırken hem de dışarıdan veri alırken kullanabilecek kullanışlı bir fonksiyondur.

```
Ret,im = okunanVideo .read()
```

Yakalanan video görüntüsü.read() ile işlemek için kullanılacaktır. Çıkış diğer uygulamalar gibi olacaktır. Videoda yüz algılanmakta ve tanınmaktaysa röle aktifleşmektedir. Ardından yeşil LED yanmaktadır. Programların birden fazla yüz tanıma testleri yapıldığında başarılı bir sonuç alınmaktadır.



Şekil 30 Birden Fazla Yüz Tanıma Testi

4. SONUÇ

Lisans bitirme projesinde tüm insanların genel sorunlarından birisi olan güvenlik sistemlerine bir çözüm sunulmuştur. Raspberry Pi Model 3B+ ile daha öncesinden tanıtılan kişinin tanınması işlemi gerçekleştirilmiştir. Tasarım açısından taşınabilirlik açısından PLA madde kullanılmıştır Solid Works de tasarlanan kutu 3D Printer ile bastırılmıştır. Tanıma işleminin başarılı olması durumunda yeşil LED yanmaktadır ve röle aktifleşmektedir. Kutuya doğrudan bağlanarak kontrol edilebilen selenoid kilit çalışmaktadır. Taşınabilir güvenlik sistemi yapılarak güvenlik arttırılmaya çalışılmıştır.

KAYNAKÇA

1. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjphNP8sp7uAhWLxoUKHV9dCZMQFjACegQIBxAC&url=http%3A%2F%2Fwww.jret.org%2FFileUpload%2Fks281142%2FFile%2F10a.zihni_kaya.pdf&usg=AOvVaw2JSFyPVaRpLSyFaFzs658R
2. https://dergipark.org.tr/tr/download/article-file/391087
3. https://www.gencayyildiz.com/blog/emgucv-multiple-face-recognitioncoklu-yuz-tanima/
4. https://docplayer.biz.tr/99501687-Local-binary-pattern-yontemi-ile-yuz-ifadelerinin-taninmasi.html
5. https://docs.opencv.org/3.4/df/d25/classcv_1_1face_1_1LBPHFaceRecognizer.html
6. https://medium.com/@yunusemrealkaz/python-opencv-yüz-tanıma-haarcascade-53cdf4c64c0e
7. https://www.robotistan.com/raspberry-pi-3-model-b-plus
8. https://www.robotistan.com/4-raspberry-pi-dokunmatik-ips-lcd-ekran-birincil-ekran
9. https://dergipark.org.tr/tr/download/article-file/891005
10. http://gomuluyazilim.com/bilgisayarli-gorme-computer-vision-raspberry-pi-opencv-kurulumu/
11. https://medium.com/@hakkitoklu/python-ile-yüz-tanıma-uygulaması-8dc4e4bd8fcf
12. https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/
13. https://towardsdatascience.com/how-do-you-train-a-face-detection-model-a60330f15fd5
14. https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348
15. https://acikerisim.siirt.edu.tr/xmlui/handle/20.500.12604/2711
16. https://python.gurmezin.com/python-ve-opencv-ile-yuz-tanima/

ÖZGEÇMİŞ

Ad Soyad: Furkan Fikret Şafak

Doğum Yeri: Erzincan /Merkez

Doğum Tarihi: 25.01.1999

Lise: Erzincan Milliyet Anadolu Öğretmen Lisesi (2013-2017)