



ELEKTRİK-ELEKTRONİK FAKÜLTESİ

ÇOK DİSİPLİNLİ TASARIM PROJESİ

SONUÇ RAPORU

Proje Başlığı: Göz Takibi Kullanılarak Araç Kontrolü			
Proje Sonuç Raporu Teslim Tarihi:			
No	Proje Ekibi Ad, Soyadı	Bölüm (ELM/EHM/BLM/KOM)	İmza
1	Emre SAKA	EHM	
2	Muhammed Turgut ÖZDEMİR ALP	EHM	
3	Umutcan SEVDİ	BLM	
4	Burak ÖZCAN	KOM	
5	Mahmut Onur DEMİRCİ	KOM	
6			
	Doç. Dr. Erdin GÖKALP	ELM	

Proje Sonuç Raporu Teslim Tarihi

ÖNSÖZ



ÖZET

Proje kapsamında gerçekleştirilen çalışmaların amacı, kapsamı, kullanılan yöntem(ler), varılan sonuç(lar) ve yönetim düzeni açık ve öz olarak belirtilmelidir. Özet 450 kelime veya bir sayfa ile sınırlanmalıdır.

Proje Özeti

Bu projede felç geçirmiş bir kişinin, dışarıdan bir yardıma ihtiyaç duymadan, kendi kendine bir yerden bir yere gitmesi amaçlanmaktadır. Omurilik felci geçirmiş bir kişide kas kaybı ve duyu fonksiyonlarının kaybı söz konusudur. Bu hastaların kullanabileceği göz hareketlerinin takibi ile hareket eden bir araba projesi yapılmıştır.

Öncelikli olarak bu projenin hayatı geçirilmesi için gerçek bir tekerlekli sandalye yerine robot bir araba ile gerçeklenmiştir. Projede göz takibi yapabilmek için bir kameralan faydalanyılmıştır.

Bu göz hareketlerinin ifade ettiği komutlar görüntü işleme ile anlaşılmıştır. Burada OpenCV kullanılmıştır. Bu işlemler ise Python programlama dili kullanılarak yapılmıştır. Algılanan bu komutları arabaya aktarmak için ise Wifi üzerinden UDP protokolü ile haberleşmesi sağlanmıştır. Verilerin iletiminde socket.py isimli kütüphane kullanılmıştır. Raspberry pi üzerinde ise gelen verileri okuyup anlamlandıran ve araç kontrol işlemesine iletten C ile yazılmış bir program bulunmaktadır. Elde ettiği verileri gömülü sistemi kontrol eden programa iletmektedir. Gömülü sistem ile UDP server programları iki ayrı threadde çalışmaktadır. Bu sayede araç kesintisiz hareket edebilmektedir. Gömülü sistem C programlama dilinde yazılmıştır ve araç kontrolü motor sürücü devresine gönderilen veriler ile sağlanmaktadır.

Araç 2 tekerleklidir ve öndeği bilye ile dengesini sağlamaktadır. Araç hareketi hız değerlerinin değiştirilmesi ile gerçekleştirilmektedir.

Anahtar Kelimeler: Python, OpenCV, Dlib, C, Linux, Socket Programming, Wi-Fi, H Bridge Motor Driver

1. GİRİŞ

Projenin amacı, kapsamı, kullanılan yöntem(ler), variılan sonuç(lar) detaylandırılarak yazılır. Tasarımın niteliğine göre, ekonomi, çevre sorunları, sürdürülebilirlik, üretilebilirlik, etik, sağlık, güvenlik, sosyal ve politik sorunlar gibi öğeler de irdelenmelidir.

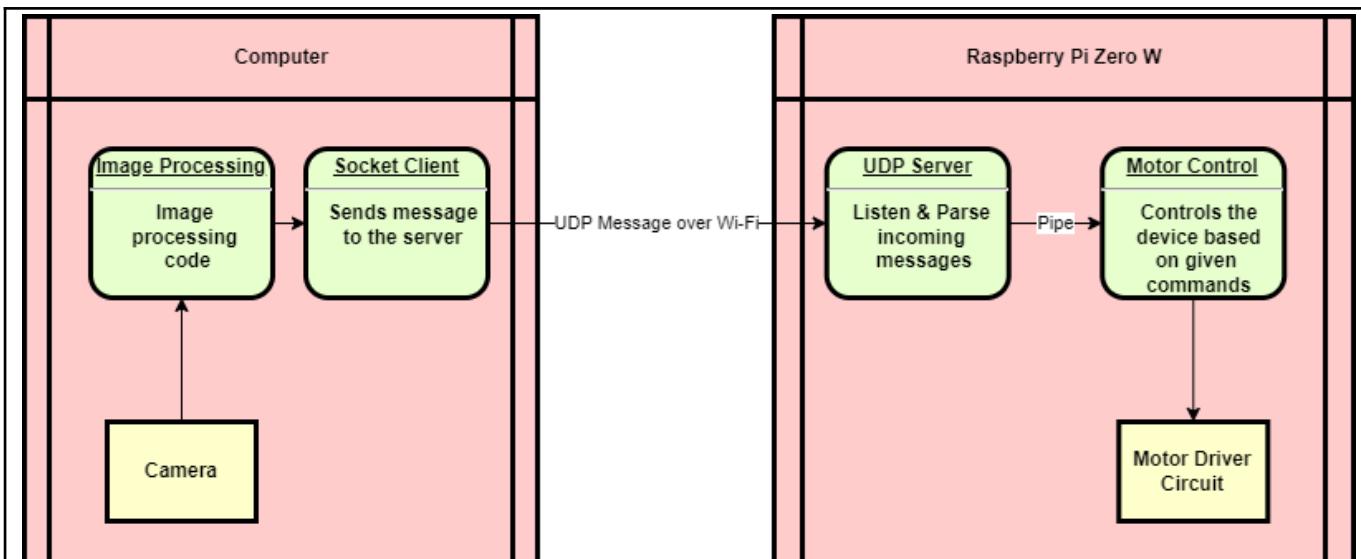
Tasarlanan bu projede sistem, omuriliği felci olan birisinin ya da boynunun alt kısmının komple bir şekilde işlevini yerine getirememesi durumlarında kullanılan elektrikli sandalyeler gibi özgürce dolaşmalarını sağlayacak prototip bir araba ile gerçekleşir. Bu proje için önceden gerçekleştirilmiş bir göz takip sistemine ihtiyaç vardır. Bu sistemde göz hareketlerini algılama ve takibi için kamera kullanılır. Kamera gözün hareketlerini algılandıktan sonra gözün konumlarını tanımlayarak hareket komutlarına çevirir. Bu hareket konumları 2 eksenli olarak düşey ve yatay şekildedir. Bu eksenlere göre hareket konumları yönlendirilir. Hareket konumları arabaya iletilir ve iletildikten sonra hareketi algılayan araba, verileri işleyerek otonom bir şekilde motorları doğru yönde çevirerek hareket eder. Bu sistem ve proje sayesinde hastaların hareket etmesi sağlanır ve hayatın içerisinde rol almalarını hedeflenmiştir.

Göz hareketi sistemi uygulandıktan sonra, araba; kullanılan sistem tarafından algılanan hareket türlerini kullanarak kontrol edilir. Bu sistemde görüntü işleme algoritmaları kullanılacaktır. Motorlarla haberleşmenin ve arabanın fiziksel hareketini, donanımsal ve yazılımsal olarak sağlanacaktır. Bu amaç doğrultusunda insan gözünün hareketleriyle algılanan veriler, kablosuz olarak bilgisayara aktarılır ve Wifi yardımıyla da mekanizmeye aktarılır. Mekanizmada bunu fiziksel harekete dönüştürür. Bu sayede de istediğimiz şekilde hareket kontrolü sağlanmış olur.

2. GEREÇ, YÖNTEM VE YÖNETİM DÜZENİ

Projede uygulanan yöntem ve araştırma teknikleri açıklanır. Kullanılan yöntem ve tekniklerin proje yönetim planında öngörülen amaç ve hedeflere ulaşmaya elverişli olup olmadığı açıklanır. Değişiklikler varsa açıklanır. Yönetim planında belirtilen iş paketleri ile görev dağılımının gerçekleşme düzeyi açıkça belirtilir.

Programımız 2 ayrı cihazda çalışmaktadır. Görüntü işleme işlemi kullanıcı bilgisayarında, araç kontrolü ve haberleşme ise Raspberry Pi üzerinde gerçekleşmektedir.



Projede ilk önce alınan yüz görüntüsünün griye çevrilme işi yapılmıştır. Daha sonrasında griye çevrilen görüntüde sadece göz frame'i ele alınmıştır. Bu sayede sadece göz kısmına odaklanılmış olundu. Daha sonrasında threshold yöntemi kullanılarak göz kısmındaki göz bebeği tam olarak tespit edilmiştir. Sonrasında göz bebeğinin hareketine göre kontür (koordinat) değerleri bulunmuştur. Kontür değerleri bulunduktan sonra göz bebeğini kapsayacak şekilde kare şeklinde bir şekil oluşturulmuştur.

```

for face in faces:
    x, y = face.left(), face.top()
    x1, y1 = face.right(), face.bottom()
    cv2.rectangle(frame, (x, y), (x1, y1), (0, 255, 0), 2)

    landmarks = predictor(gray, face)

    # Detect blinking
    left_eye_ratio = get_blinking_ratio([36, 37, 38, 39, 40, 41], landmarks)
    right_eye_ratio = get_blinking_ratio([42, 43, 44, 45, 46, 47], landmarks)
    blinking_ratio = (left_eye_ratio + right_eye_ratio) / 2

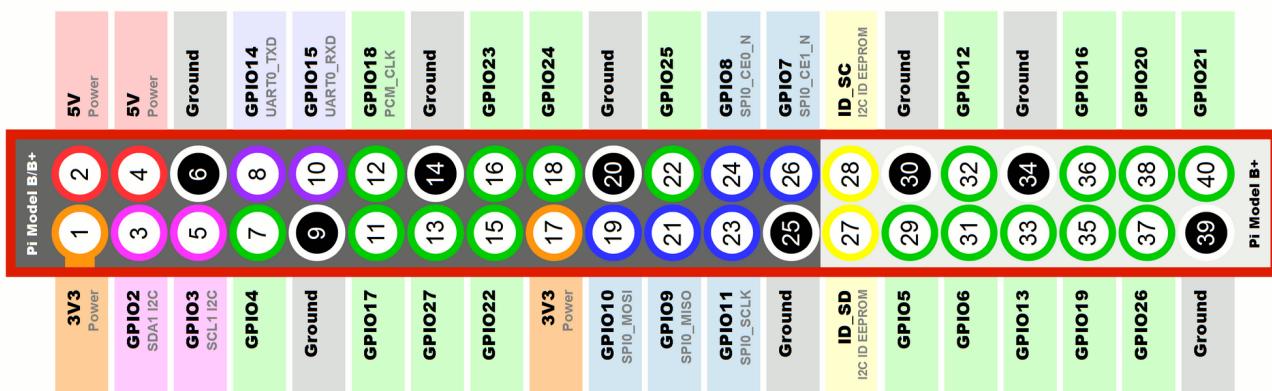
    # Gaze detection
    gaze_ratio_left_eye = get_gaze_ratio([36, 37, 38, 39, 40, 41], landmarks)
    gaze_ratio_right_eye = get_gaze_ratio([42, 43, 44, 45, 46, 47], landmarks)
    gaze_ratio = (gaze_ratio_right_eye + gaze_ratio_left_eye) / 2
    direction = get_direction(left_eye_ratio, right_eye_ratio, gaze_ratio_left_eye)
    frame_directions.append(direction)
    if len(frame_directions) >= 10:
        send(get_max_direction())

def get_direction(left_eye, right_eye, gaze_left) -> DIRECTION:
    direction: DIRECTION = DIRECTION.NONE
    if left_eye > 4.5 and right_eye > 4.5:
        if left_eye < 6.5 and right_eye < 6.5:
            direction = DIRECTION.DOWN
    elif gaze_left <= 1.2:
        direction = DIRECTION.RIGHT
    elif gaze_left > 4:
        direction = DIRECTION.LEFT
    else:
        direction = DIRECTION.UP
    return direction

```

Elde edilen görüntü ile ilgili bilgiler doğrudan gönderilmez. 10 adet frame için yön hesaplandıktan sonra aralarında en çok olan yön seçilir.

Görüntü işlemenin ardından elde edilen veriler Socket.py kütüphanesi kullanarak UDP protokolü ile Wifi üzerinden Raspberry Pi'ya aktarılmaktadır. Rasberry Pi ağ üzerinde "pi" adlı hostname ile kendini tanıtmaktadır. Dolayısıyla ip adresi değişse de cihaz tarafından tanınabilecektir.



www.raspberrypi-spy.co.uk

Raspberry Pi Zero üzerindeki bulunan GPIO pinleri üzerinden motor sürücü devresine jumperlarla bağlanmıştır. Raspberry pi üzerindeki C programlama dilinde yazılmış Wiring Pi adlı kütüphane ile motor sürücü devresinin hareketi gerçekleştirilecektir.

Raspberry Pi'nın gelen UDP mesajlarını okuyabilmesi için bir UDP serverin tasarlanması gereklidir. Bunun için Rasbian işletim sistemindeki UNIX soket kütüphaneleri kullanılmıştır. C dilinde yazdığımız bu program düzenli olarak gelen mesajları dinleyip structlara dönüştürerek araç kontrol kısmına iletmektedir. Aracın dinleme yaparken harekete devam edebilmesi için başlangıçta multi process olarak yazılmıştır. Ardından performansı artırmak için multi thread'e çevrilerek refaktör edilmiştir. Bu sayede gelen mesajlara araç kontrol sistemi doğrudan ulaşabilmektedir.

Aracın tekerleri şaseye sabitlenmiştir. Bu sebeple yön vermek için hız değerleriyle oynamaktadır. Gelen mesajdaki yön bilgisine göre ters yönde tekerleğin hızı artırılır ve istenilen yöne döner.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <strings.h>
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>

enum DIRECTION {
    DIRECTION_NONE = '0',
    DIRECTION_LEFT,
    DIRECTION_DOWN,
    DIRECTION_UP,
    DIRECTION_RIGHT,
    DIRECTION_END,
};

/*
 * configuration related information about the UDP
 * listener server
 */
typedef struct SV_SERVER_CONFIG {
```

```

int port;
int max_line;
logger *log;
} sv_conf;

/***
* sv_listen - creates a UDP server
* @args sv_conf pointer arguments of server in server configuration
*/
void *sv_listen(void *args);

/***
* sv_handle - handles incoming request and sends received sv_motion to
* write_pipe
* @sockfd socket to handle
* @cliaddr client address
* @sv_args arguments of the sv_listen thread.
*/
void sv_handle(int sockfd, struct sockaddr_in cliaddr, const sv_conf *sv_args);
/***
* configuration related information about the driver
*/
typedef struct DV_DRIVE_CONFIG {
logger* log;
} dv_conf;

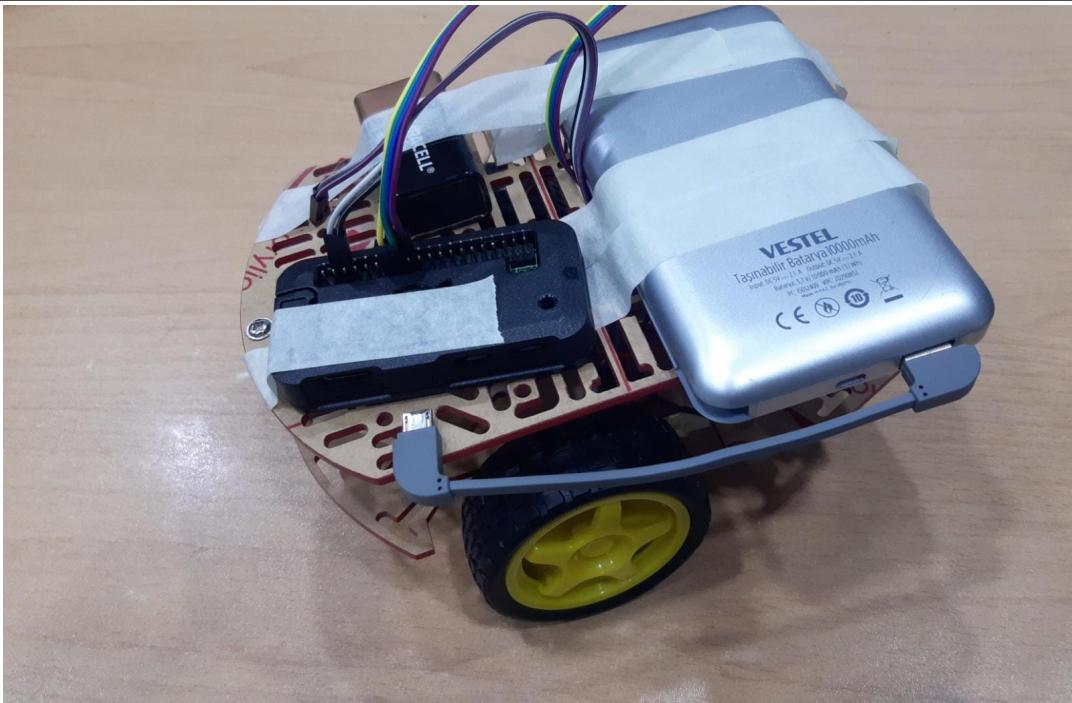
/***
* dv_drive - listens the direction and controls the device
* @args dv_drive configuration arguments in dv_conf
*/
void *dv_drive(void *args);

```

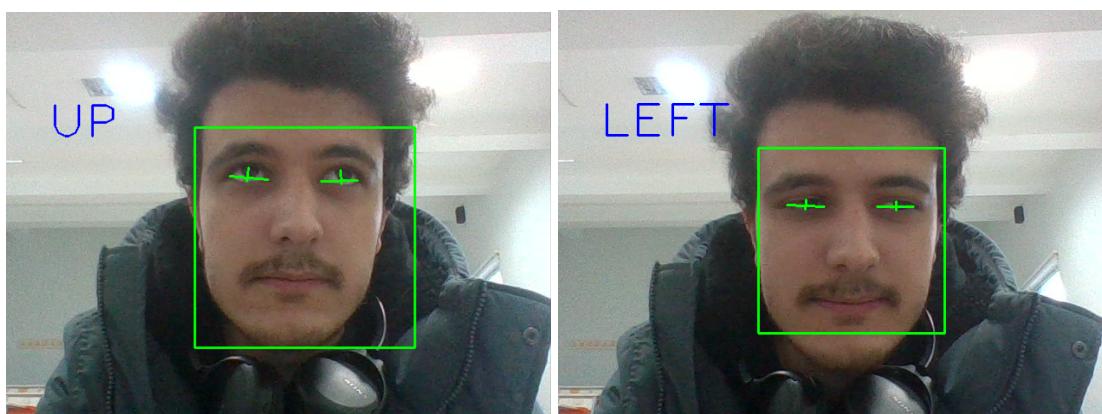
3. SONUÇLAR

Projede hedeflenen çıktılarla ulaşıldığı tablo, şekil ve grafikler kullanılarak açıklanır.

Yapılan projede kablosuz şekilde bilgisayarla haberleşebilen ve bilgisayar kamerasından alınan göz komutları vasıtasiyla hareketi kontrol edebilen robot arabanın tasarımını ve fiziksel olarak gerçekleştirilmesi yapılmıştır. Aracın son hali şekildeki gibidir:



Kameradan gelen görüntü işlenmektedir. Bu görüntü işleme, makine öğrenmesi modeli ile göz bebeği konumu tespit edilerek yapılmıştır. Gözün etrafındaki farklı noktalara olan uzaklıklar hesaplanarak kişinin bakmış olduğu yön tespit edilmektedir. Bunun sonucunda; sağ, sol, yukarı, aşağı ve gözün kapalı olduğu durum olacak şekilde 5 farklı komut algılanmaktadır:



Robot araba için istenilen işlevleri güvenli ve verimli bir şekilde gerçekleştirmeyi sağlayan mekanik bir tasarım tasarlanmıştır. Yapılan çalışmalar test edilmiştir. Yapılan testler sonucunda aracın istenilen şekilde çalıştığı gözlemlenmiştir.