T.C.
İSTANBUL GELİŞİM ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY SİNİR AĞI KONTROLLÜ OTONOM RC ARAÇ UYGULAMASI

ERDEM ŞANLI

YÜKSEK LİSANS TEZİ
MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN
DR. ÖĞR. ÜYESİ ÜMİT ALKAN

İSTANBUL, 2018

Erdem ŞANLI tarafından hazırlanan "YAPAY SİNİR AĞI KONTROLLÜ OTONOM RC ARAÇ UYGULAMASI" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ / OY ÇOKLUĞU ile İstanbul Gelişim Üniversitesi Mekatronik Mühendisliği Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Dr. Öğr. Üyesi Ümit ALKAN

Bilgisayar Mühendisliği, İstanbul Gelişim Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum ……

Başkan: Prof. Dr. ALİ OKATAN

Bilgisayar Mühendisliği, İstanbul Gelişim Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum ……...

Üye: Prof. Dr. Ekrem YANMAZ

Elektrik-Elektronik Mühendisliği, İstanbul Esenyurt Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum ……...

Tez Savunma Tarihi: ...../……/……

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……

Prof. Dr. Nuri KURUOĞLU

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

İstanbul Gelişim Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu, bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Erdem ŞANLI
(…./…./20…)

YAPAY SİNİR AĞI KONTROLLÜ OTONOM RC ARAÇ UYGULAMASI

(Yüksek Lisans Tezi)

Erdem ŞANLI

GELİŞİM ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Mayıs 2018

ÖZET

Bu tez çalışmasında, RC (Radio Control) otonom bir araç üzerine yerleştirilen kamera modülü ile Yapay Sinir Ağları algoritmaları kullanılarak, aracın şeritler arasında kalabilmesi amaçlanmıştır. Bir mikrobilgisayar olan RPİ ile 120x160 çözünürlüklü görüntü verileri elde edilerek, Python programlama dili üzerinde, TensorFlow görüntü işleme teknikleri kullanılmış ve uygulama yapılmıştır. Tezin birinci kısmında uygulamanın amacı, yöntemi, literatür taraması ve geçmiş bilimsel çalışmalar ele alınmıştır. Tezin ikinci kısmında Yapay Sinir Ağları hakkında detaylı bilgi verilmiştir. Tezin üçüncü kısmında, uygulama aşamasında kullanılan elektronik komponentler hakkında teknik bilgiler verilerek bağlantı şekilleri ele alınmıştır. Tezin dördüncü kısmında gerekli yazılım kurulumları, otonom aracın eğitimi, uygulaması ve simülasyon üzerindeki çalışmalar incelenmiştir. Son kısmında ise, uygulama sonuçları ele alınmış, gelişime açık öneriler ve karşılaşılan problemlerden bahsedilmiştir.

Anahtar Kelimeler : Yapay sinir ağları, otonom araç, görüntü işleme

Sayfa Adedi : 74

Danışman : Dr. Öğr. Üyesi Ümit ALKAN

ARTIFICIAL NEUTRAL NETWORK CONTROLLED AUTONOMOUS RC VEHICLE APPLICATION

(M. Sc. Thesis)

Erdem ŞANLI

GELİŞİM UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

May 2018

ABSTRACT

In this thesis, by using Artificial Neural Networks algorithms with the camera module placed on an RC autonomous vehicle it is aimed to keep the vehicle between the lanes. By obtaining 120x160 resolution image data with Artificial Neural Networks, a microcomputer, TensorFlow image processing techniques have been used and an experiment has been executed on the Python programming language. In the first part of the thesis, the purpose and methodology of experiment, literature review and past scientific studies are discussed. In the second part, detailed information about Artificial Neural Networks is released. In the third part of the study, the connection forms are discussed by giving technical information about the electronic components used in the application phase. In the fourth part of the thesis, necessary software installations, training of autonomous vehicle, application and studies on simulation are analyzed. In the final part, the results of the execution are handled, proposals being open for improvement and the problems encountered are mentioned.

Key Words : Artificial neural networks, autonomous vehicle, image processing

Page Number : 74

Supervisor : Assist. Prof. Dr. Ümit ALKAN

TEŞEKKÜR

Yüksek lisans tez çalışmam esnasında kıymetli fikir, vakit ve yardımlarını esirgemeyen, sayın danışman hocam Dr. Öğr. Üyesi Ümit ALKAN' a, eğitim ve kariyer hayatım boyunca maddi manevi desteğini hiçbir zaman esirgemeyen değerli büyüğüm, hocam Dr. Öğr. Üyesi Korhan KAYIŞLI 'ya, tez çalışmam esnasında kıymetli fikir ve tecrübelerini benden eksik etmeyen dostum, hoca arkadaşım Arş. Gör. Hakan Yıldız'a saygılarımı ve teşekkürlerimi sunarım.

Hayatım boyunca her türlü desteğiyle yanımda olan, sonsuz mutluluğu hak eden kıymetli annem Leyla ŞANLI, babam Ali ŞANLI ve tüm zorluklarda yanımda olan, çalışma azmiyle bana ilham olan kıymetlim Öğr. Gör. Elif TAŞDEMİR' e sonsuz minnet ve teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
A	Amper
X	Hücrenin Giriş Vektörünü
Y	Hücrenin Net Girişini
Y	Hücrenin Net Girişini
Z	Hücre Çıkışını
W	Hücrenin Ağırlıklar Matrisi
φ (v)	Hücrenin Aktivasyon Fonksiyonunu
Kg	Kilogram
KV	Devir Katsayısı
mAh	Mili amper

Kısaltmalar	**Açıklamalar**
ALVINN	Autonomous Land Vehicle in a Neural Network
CSI	Camera Serial Interface
ESC	Electronic Speed Controller
GPIO	General purpose input/output
GPU	Graphic Processing Unit
I^2C	Inter-Integrated Circuit
LİPO	Lityum Polimer
SSH	Secure Shell
RC	Radio Control
RPİ	Raspberry Pi
TPU	Tensör İşleme Ünitesi
VIAC	Vislab Intercontinental Autonomous Challeng
YSA	Yapay Sinir Ağları

1. GİRİŞ

İnsan beyni, bilindiği gibi eşsiz bir hesaplama cihazıdır. Çok miktarda veriyi idare edebilen beyin, çaba göstermeden çevremizdeki kalıpları tanır ve işler. Tüm bu işlemler sonucunda, bir olay karşısında vereceğimiz tepki ve kararlar, önceki yaşanmış deneyimlerden öğrenilir. Günlük hayatta verilen bu kararlar insan türünün çok hızlı ilerlemesine yardımcı olmuştur. Teknolojinin gelişmesiyle birlikte, bilgisayarlar daha güçlü hale geldiğinden bazı beyin işlevlerini taklit etme fikri ve bu işlemleri makinelerde uygulamak gerçek olmuştur.

Otomotiv sektöründeki gelişmelere bağlı olarak rekabet artmakta, büyük yatırımlar yapılmakta ve önemli yenilikler ortaya çıkmaktadır. Bu sektörde, geçmişte daha çok fiyat rekabeti yaşanmışsa da fiyat tek parametre olmaktan çıkmıştır. Pazar paylarını artırmak isteyen üreticiler, ar-ge harcamalarını her yıl artırarak ve yenilik peşinde koşarak müşterilerini tatmin etmeye çalışmaktadırlar. Otomotiv sektörü diğer sanayi kollarıyla iş birliği içinde olan bir sektördür. Bu nedenle de otomotiv sektörü, üretim ve teknoloji sektörünün en önde gelen unsurlarından biridir.

Problem Durumu / Konunun Tanımı

Cheng (2011) Otonom araçları tanımlarken, belirlenen bir konumdan istenilen konuma insan müdahalesi olmadan ya da kendi karar mekanizmasıyla önceden tanımlanmış bir görevi gerçekleştiren araçlardır, ifadesini kullanmaktadır [1]. Bu düşünce yakın geçmişte ne kadar uzak gelse de günümüzde Yapay Zekâ konusundaki gelişmeler sayesinde yapılabilir hale gelmiştir. Dünya genelinde giderek artan trafik kazalarından dolayı güvenlik ihtiyacının oluşması, sürücülerin hayati öneminin artması, değişen konfor ve verimlilik anlayışı sonucunda otomotiv üreticileri, daha güvenli olduğu düşünülen otonom ve yarı otonom sistemlere ilgi duymaya başlamıştır. Trafik kazalarının genel nedenleri olarak taşıt, yaya ve yol kusurları gibi etkenler gösterilse de kullanıcıdan kaynaklanan hataların, kazalarda çok ciddi bir paya sahip olduğu görülmektedir. Bu hataları en düşük seviyeye indirgemek için daha "akıllı" araçlara ihtiyaç duyulmaktadır. Sürücü kaynaklı sorunlarda, kötü hava şartlarında ve uzun süren yolculuklarda otonom veya yarı otonom sistemlerin daha avantajlı olduğu düşünülmektedir.

İnsan tepkisinin uzun olabildiği anlık olaylarda otonom sistemler kaza önlemede daha önemli bir unsur haline gelebilir. Ayrıca otonom sistemlerde yüksek oranda yakıt tasarrufu sağlandığından, harcanan enerji ve salınan sera gazları önemli ölçüde azalmaktadır. Otonom teknoloji, kendine özgü dört temel yapıdan oluşur: çevre algısı ve modelleme, yerelleştirme ve harita oluşturma, yol planlama ve karar verme, son olarak hareket kontrolü olarak gruplandırılmıştır [1].

Diğer yandan otonom teknolojilerde Yapay zekâ, YSA (Yapay Sinir Ağları), Görüntü İşleme teknikleri, Python, Matlab, Scala, Ruby ve R gibi makine öğrenimi dillerinin uygulanması son yıllarda mekatronik alanının gelişimi ve etkinliği sayesinde oldukça yaygınlaşmıştır [2,3]. Bu otonom aracı geliştirmek için kullanılabilecek çeşitli teknolojiler bulunmaktadır. YSA ve görüntü işleme teknikleri kullanılarak sorumluluk ve kontrol mekanizması otonom araçlara bırakılmıştır. Bu sayede otonom araçlar trafik sıkışıklığı, trafik kazası, gecikmeli yolculuk gibi birçok trafik sorununa çözüm olabilir. Belirlenen bir parkurda belli noktalar arasında kendi kendine gidebilen minyatür RC otonom araba bu tezin önemli bir parçasıdır.

Araştırmanın Amacı

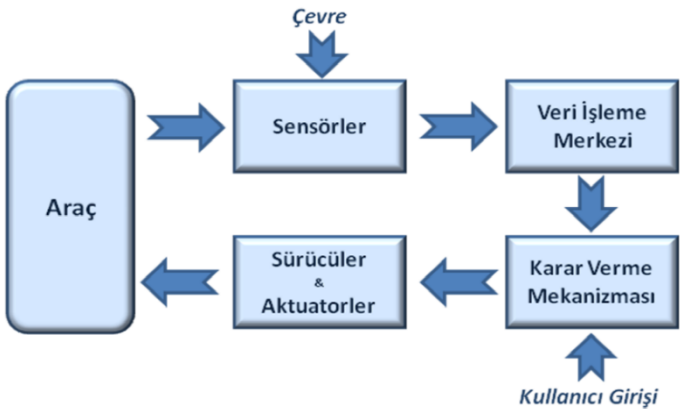
Bu tez, bilgisayar bilimlerinde Makine Öğrenimi olarak adlandırılan öğrenme yöntemine odaklanacaktır. Mitchell (2006) Makine öğrenimini, bilgisayarlar aracılığıyla, programlanabilir diğer cihazların, çalışma alanlarına göre programlanarak öğrenebilme olanağının sağlanmasıdır, ifadesi ile açıklamaktadır [2]. Birden fazla Makine Öğrenme algoritması vardır. Bunlardan biri YSA'dır. Bir YSA sistemi olan görüntü işleme tekniklerini kullanarak otonom bir RC (Radio Control) araç eğitmeye çalışıyoruz. Bu araç beyin nöronları konseptinde, bir kamera veya diğer sensörlerden gelen girdilerle beslenir ve bu girdiler, makinenin yapması gereken bir komut gibi çıktı vermeyi gerçekleştirir. Gündelik hayatta olduğu gibi çıkış komutuna da "yanlış ya da doğru karar" ifadesi verilir. Bu durumda otonom araba kendini ayarlama şansı bulacaktır. Ağ üzerinden ne kadar çok kaliteli veri geçerse algoritma ve görüntü işleme o derece doğru olur. Böylece makine öğrenimi sağlanır.

Bu tezin uygulama aşamasında otomobilin işlem birimi olarak mikrobilgisayar olan RPİ (Raspberry Pi) ve görüntü birimi için Raspberry Pi V2.1 kamera modülü kullanılmıştır. Görüntü işlemede kullanılacak yazılım ise Google'ın Makine İstihbarat Araştırması

organizasyonu içindeki, Google Brain ekibinde çalışan araştırmacılar ve mühendisler tarafından 2015 yılında geliştirilmiştir.
Açık kaynak kodlu Tensör İşleme Ünitesi (TPU) kullanılmış, elde edilen veriler TPU ile işlenerek belirli bir sürüş ortamında bir noktadan diğerine doğru yol alabilmesi için aracın şeritler arasında gidebilmesi ve engellerden kaçması amaçlanmıştır.

Araştırmanın Yöntemi

Bu tezin uygulama aşamasında, ilk olarak eğitilen RC araç üzerindeki Pi kamera modülünden 120x160 çözünürlükte, 10 000 ile 30 000 arasında görüntü verisi elde edilmiştir. RPİ mikrobilgisayarın işlem hızının yeterince yüksek olmaması sebebiyle elde edilen veriler ana bilgisayara aktarılmış ve bu veriler, Python (versiyon 3.6) programlama dilinde Keras ve Tensorflow kütüphanesi kullanılarak görüntü işleme süreci başlatılmıştır.



Şekil 1.1. Otonom Araç Genel Sistemi

İşlenen bu veriler sonucunda çıktısı alınan otopilot analizi RPİ mikrobilgisayarına aktarılarak belirlenen parkurda gözlemlenmiş, gerekli direksiyon açısı, gaz ayarları ve bozuk işlenen görüntü aktarımları düzeltilmiştir. Genel olarak otonom araçların işlem yapısı birbirlerine benzemektedir, Şekil 1.1'de görüldüğü gibi beş temel bloğa ayrılır:

- Sensörler
- Veri işleme merkezi
- Karar verme merkezi
- Sürücüler ve Aktüatörler

Birinci aşama genel çevre ve araçtan veri toplayan sensörlerden oluşmaktadır. Bu sensörler farklı özellikteki bilgileri elektriksel sinyallere çevirir ve bir sonraki birime aktarım sağlanır. Sensörlerden alınan bilgiler çok fazla ve karmaşık bir yapıya sahiptir. Elde edilen bu verilerin, doğrudan karar mekanizması tarafına aktarılıp yorumlanması ve sonrasında bir karar alınmasını beklemek uzun bir süre kaybına sebep olacaktır. Bu tür sorunları çözebilmek için, bu tip otonom sistemlerde karar alma ve yorumlama sonraki birimle paylaşılır. Sonraki birim veri işleme merkezidir. Bu birimde veri işleme ve algılayıcıların doğruluğunu birbirleriyle test edebilmektir.

Karmaşık bir yapıya sahip olan veri topluluğu, veri işleme merkezi tarafından daha sade ve düzenli bir hale getirerek sonraki birim olan karar verme merkezine yollanır. Karar verme mekanizması, kişilerin sisteme eklediği girdileri ve veri işleme merkezinden gelen bilgileri karşılaştırıp değerlendirerek otonom aracın ne yapması gerektiğini ne kadar hızda ve hangi yönde gidebileceğini belirler. Bu işlemler sonucunda bir sistemin veya mekanizmanın hareket etmesini sağlayacak aktüatörleri hareketlendirir. Algılayıcılar, iç yapılarında sunulan algoritmalarıyla gelen referans değerlere göre otonom arabanın fonksiyonlarına (gaz, fren, direksiyon) müdahale ederler. Tüm bu işlemler sonucunda döngü tamamlanmış olur.

Otonom Araçların Gelişimi / Yapılan çalışmalar

Otonom araçlar üzerine ilk çalışmaların, 1939 yılında, “Futurama Dünya Fuarı”nı düzenleyen General Motors şirketi öncülüğünde başladığı kabul edilmektedir. Fuarda sunulan araç, insanların güvenli bir şekilde şehir içerisinde otomatik olarak belli bölgelere taşıyan bir araç olarak tanımlanmıştır [3]. Dikkat çeken ilk çalışmanın ise Japonya’nın Tsukuba bölgesinde bulunan Makine Mühendisliği Laboratuvarı’nda 1977 yılında başladığı söylenebilir. Beyaz çizgiyi takip edebilen ve aynı zamanda 20 km/h hıza ulaşabilen bir araç geliştirmişlerdir. Günümüzde bu hız düşük bir seviye olarak görülse de geliştirildiği yılların şartlarına göre öne çıkan bir çalışmadır [4]. Bundeswehr Üniversitesi’nde, Ernst Dickmanns ve ekip arkadaşları tarafından 1980’li yıllarda geliştirilen Mercedes-Benz robot kamyon araç üzerine kamera ve sensörler eklenerek 95 km/h hıza ulaşan otonom araç üretmişlerdir. İlk denemeler güvenlik nedeniyle trafiğe kapalı yollarda gerçekleşmiştir [3].

1995 yılında 800 milyon dolarlık “European Prometheus Projesi” kapsamında Navlab isimli proje ile Carnegie Mellon Üniversitesi, No Hands Across America aracı %98.2 verim ile 5000 km yol gidebilmiştir.

2002 yılında, ABD Savunma Bakanlığı tarafından “Büyük Mücadele” anlamına gelen Grand Challenge adında bir yarışma başlatılır. ABD vatandaşları haricindekilerin katılamadığı bu yarışmada amaç, otonom bir taşıtın ABD’de bulunan bir çölde belirlenmiş parkurda kendi kendine gidebilmesini sağlayacak bir sistem geliştirmektir. 2004 ve 2005 yıllarında DARPA (Defense Advanced Research Projects Agency) projesi destek fonunun düzenlediği otonom araç yarışlarında kazanan takıma milyon dolarlık teşvik vermiştir. 2010 yılında VIAC (Vislab Intercontinental Autonomous Challeng), VisLab isimli şirket tarafından 13000 km’ lik bir yarışma düzenlenmiş ve yarışmaya katılan 4 otonom elektrikli araç Shanghai şehrinde bulunan Expo’ya uğrayacak şekilde İtalya’dan Çin’e giderek yarışmayı tamamlamışlardır [5]. Bu alanda Google, Tesla, Ford gibi diğer önde gelen firmalar da kendi makine öğrenimini ve özellikle kendi bilgisayar sistemlerini daha fazla geliştirmek için derin öğrenme ve görsel veriler üzerinde çalışmalar başlatmıştır. Google'ın kendi kendine çalışan otomobil departmanı, Waymo isimli otonom araç üzerinde derin öğrenme tekniklerini kullanmış ve özellikle tehlikeli "kenar durumlarda" çözümlemeye odaklanmıştır [6]. Sürüş için yapay görüş teknolojisi sağlayan Tesla ve Mobileye, çoğunlukla derin öğrenme yaklaşımına odaklanmıştır. Tesla’nın Model S serisinde de görüldüğü gibi otonom sistemlerle donatılmış şerit takibi, otonom park seçeneği, otomatik manevra kabiliyeti, gaz ve fren kontrolünün araca bırakılması gibi otopilot sistemleri uygulanmıştır [7].

Pomerleau (1989) Zamanın teknolojisi ile ABD’de ise DARPA’nın destek verdiği kendi kendine gidebilen bir kara aracı olan ALVINN (Autonomous Land Vehicle in a Neural Network), ilk yol sunumunu gerçekleştirdi. Araç üzerine konumlandırılan kamera, lazer radar ve robot kontrol mekanizmaları sayesinde, görüntü aktarımı sağlanmış ve diğer sensörlerden aktarılan veriler kullanılarak, araç saatte 30 kilometre hıza çıkabilmiştir. 1987 yılına gelindiğinde ise GPS ve harita bilgisi olmadan, sadece sensör bazlı arazi (off-road) sürüşünü gerçekleştirdi. Gerçekleştirilen sürüşte, konumunu bilmeyen bir sürücü direksiyon başına geçirilerek, belirlenen parkurda A konumundan B konumuna ilerlemesi şeklinde düşünebiliriz. 600 metrelik yol boyunca yokuşlar, virajlar, kayalar ve hava şartları gibi karmaşık bir güzergâh yapısına sahip olduğu belirtilmiştir [8].

Ulmer (1994) Ernst Dickmanns'ın tasarladığı VaMP ve Vita-2 adı verilen bu araçlar, Paris'te bulunan, 1000 km uzunluğa ve 3 şerite sahip bir otobanda 130 km hıza ulaşarak sürüşü tamamlamayı başarmıştır. Aracın otonom bir sürüş ile kontrol ettiği şartlar şunlardı; şeritte araç yok iken, arka arkaya sıralı bir şekilde ilerlerken ve sürüş esnasında, sağa ve sola manevra yaparak şerit değiştirirken [9].
Dickmanns ve diğerleri (1994) Mercedes-Benz'in, S-sınıfı bir modeli kullanılarak 1600 kilometrelik mesafeye sahip, Münih'ten Kopenhag'a götürüp getirmeyi başardı. Saatte 175 kilometre hıza ulaşan araç, %95'lik bir oranla kontrolü kendisi devralmıştır [10].

Bertozzi ve Broggi (1998) Parma Üniversitesi'nde çalışmakta olan bilim adamı Alberto Broggi'nin sunmuş olduğu Lancia Thema markalı araç, otoban üzerinde ki trafik şeritleri arasında kalabilmek üzere tasarlanmıştır. İtalya'nın kuzeyinde bulunan 2000 kilometre mesafeye sahip bir yolu, ortalama 90 kilometre hız ile 6 günde tamamlamayı başarmıştır. Araç yolun %94'lük kısmını otonom sürüş olarak tamamlamıştır. Aracın öne çıkan en önemli özeliği ise, bu mesafeyi siyah beyaz bir kamera aracılığıyla bitirmiş olmasıdır [11].

Güner (2011) çalışmasında, "Okan Otonom Otomobil Projesi" kapsamında geliştirilmiş otonom araç için hız kontrolörü tasarlanmış, bu yapı hem simülasyon ortamında hem de yol testlerinde uygulanmıştır. Otonom araçta kullanılan hız kontrolünün temel fonksiyonu, elektronik bir sinyal olarak uygulanan referans hızda, yolun eğim gibi diğer bozucu etkenlerini gözetmeksizin aracın ilerlemesini sağlamaktır. MatLaB programı sayesinde değişken referans hız girişleri ile bozucu etkenlerle birlikte simülasyonu sağlanmıştır [12].

Manyika ve diğerleri (2013) Bir otonom araç için, 3D kamera ve diğer sensörler (Lazer, Görüntü Algılama ve Aralığı, LIDAR ve GPS) kullanmıştır. Araca kamera ve algılayıcılardan gelen giriş sinyallerini, yapay zekâ ile verileri depolayabilen ve trafik kurallarına göre aracın nasıl çalışması gerektiğine karar veren yazılımı geliştirerek bu konuda öncü olmuşlardır [13].

Bojarski ve diğerleri (2016) Teknoloji devi NVIDIA şirketinin otonom araçlar için görüntü işleme alanında uzun çalışmalar sonucunda kendi CUDA programlamasından faydalanarak çevre birimleri ve GPU (Graphic Processing Unit) donanım altyapısını kullanmıştır. Uçtan uca öğrenme ile derin bir sinir ağının eğitildiği ve test edildiği bir yaklaşımı açıkladılar. Otonom aracın direksiyon açısını, ekledikleri bir kamera ile ön giriş görüntüsüne dayanarak

başarılı öngörür bir model tanımladılar. Komple bir yazılımın parçası olarak otonom sürüş için verileri kullandıklarında, öğrenen Pilot Net olarak bilinen sistemi yarattılar. Kullanıcıların davranışlarını taklit etmek ve kendi kendine çalışan bir otomobil olarak konumlandırmak için kontrol ünitesi geliştirilmiş ve Pilot Net, oluşturulan direksiyon açılarıyla eşleştirilmiş ve yol imgeleri kullanılarak eğitilmiştir [14].
Kurakin, Goodfellow ve Bengio (2016) Pilot Net sisteminin nesneleri tanımadaki başarısından sonra ek olarak şerit işaretleri, yol kenarlarında bulunan sabit etkenler ve diğer arabalar gibi belirgin özellikleri öğrenme odaklı geliştirmişlerdir. Sistem bu sayede öngörme ve programlamanın zor olacağı daha ince özellikleri öğrenmeyi başarmıştır [15].

Doruk (2016), bir platform üzerindeki şeridi takip eden ve yoldaki işaretlere göre hız ayarlaması yapabilen otonom bir robot araba uygulaması yapmıştır. Bu sistemde Ardunio Uno, Şerit takibi için TCRT5000 kızılötesi algılayıcıları, renk algılama için TCS3200 renk algılama kartı,mesafe ölçümü için HC-SR04 Ultrasonik Algılayıcı ,ve uzaktan bağlantı içinde HC05 Bluetooth Modülü gibi algılayıcılar kullanılmıştır. Arduino'ya uzaktan erişim için Android işletim sistemli bir telefon ve Android'e yazılmış bir program kullanılmıştır. Android kısmı için oluşturulan program, App Inventor uygulamasında oluşturulmuştur. Siyah bir zemin üzerine beyaz renkten oluşan bir pist ve aralarda yeşil ve mavi renklerden oluşan çizgiler çizilmiştir. Robot araba hareket komutunu aldıktan sonra beyaz renklerle sınırlandırılmış pist üzerinde hareket etmekte ve üzerinden geçtiği renkteki şeride göre hızını ayarlamaktadır [16].
William ve Edwin (2016) Çalışmalarında otonom bir araç için YSA algoritmasını kontrol ünitesi olarak kullanmıştır. Tek girişi olan bir kameradan alınan görüntüler, tasarladıkları algoritma sayesinde test edilip değerlendirilmiştir. Bu çalışmalarında birinici nesil bir Pi kamera ve Ahududu Pi 2 Modeli kullanılmıştır. Görüntü işlemede ise ANNA drive kullanılmış PyGame kütüphanesi ile birlikte sadece klavyeden kontrol edilebilmiştir. 25x25, 50x50 ve 100x100 piksel görüntü çözünürlüğüne sahip üç farklı 900 veri kümesi elde edilmiş ve aralarındaki başarı seviyesi ölçülmüş ve sonuç olarak 100x100'lük çözünürlüğe sahip veri kümesinden %78 başarı sağlanmıştır [17]. Bojarski ve diğerleri (2017) Visual Back Prop adlı yeni bir yöntem geliştirerek, elde edilen görüntüler arasından, hangilerinin tahminlere en fazla katkıda bulunduğunu, bir konvansiyonel sinir ağı tarafından görselleştirilerek bulunmasını sağlayacak duruma getirdiler. Bu sayede gereksiz veriler atılarak iş yükünden ve zamandan kazanç sağlamışlardır [18].

2. YAPAY SİNİR AĞLARI

Bu başlık altında Biyolojik Sinir Hücresini, Yapay Sinir Ağını, YSA'nın yapılarına ve öğrenme algoritmalarına göre çalışma prensipleri detaylıca incelenmiştir.

2.1. Biyolojik Sinir Hücresi

McCullock ve Pitts (1943) Biyolojik sinir hücresinden yola çıkarak, sinir sistemi için hesaplamalı bir model ağı yarattı. İki farklı model yaklaşımı gösteren araştırmacılar, birinci modelde beyindeki biyolojik süreç üzerine odaklanmış ve diğerin de sinir ağlarının yapay zekâ uygulamalarına yoğunlaşıp insan beyninin matematiksel hesaplama kabiliyetini örnek almışlardır. Bu bakış açısıyla yola çıkarak elektrik devrelerinden basit yapılı bir sinir ağı tasarlamışlardır [19]. Donald Hebb, kitabında bahsettiği gibi nöronlar arasındaki bağlantıların aynı zamanda insan beyninin öğrenmesi için gerekli olan patlamalara sebep olduğunu belirtmiştir [20].

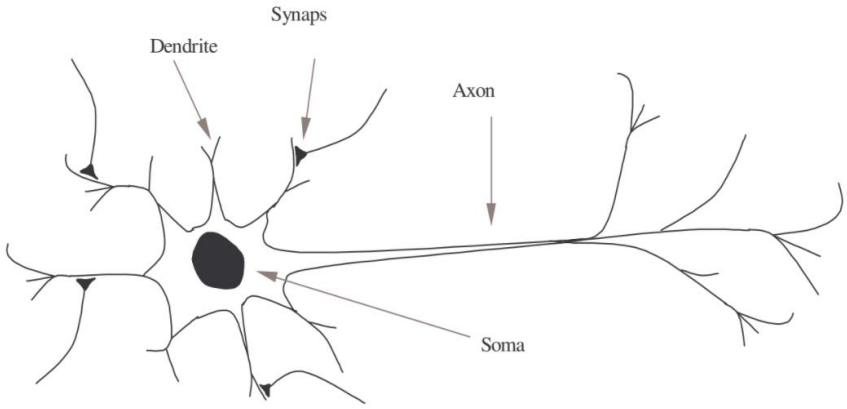
Biyolojik sinir sistemi, bilgileri aktaran, yorumlayan ve aynı zamanda sonuç üreten merkezi sinir sisteminin (beyin) bulunduğu, üç birimden oluşan bir sistem olarak açıklanmaktadır. Sinir sisteminin ilk kısmını oluşturan alıcı sinirler (reseptör), yapı içerisinde veya dış çevredeki uyarıcıları, ikinci kısım olan merkezi sinir sistemine aktarılacak elektriksel sinyallere dönüştürürler. Aktarılacak sinyaller, algılayıcı nöronlar sayesinde merkezi sinir ağına iletimi sağlanır. Son kısım olan tepki sinirleri (efektör) ise merkezi sinir ağından alınan sinyalleri, organizma çıktısı olarak hareketsel bir tepki oluştururlar. Şekil 2.1' de bir sinir sisteminin blok gösterimi verilmiştir.

Şekil 2.1. Biyolojik Sinir Sisteminin Blok Gösterimi

Bir nöron, soma (hücre çekirdeği), dendritler ve aksonlardan oluşur. Bunlar girdi ve çıkış kanallarında bulunan nöronları birbirine bağlar. Elektriksel sinyaller dendritler yoluyla nöronlar arasında iletişimi sağlayarak aktive eder.

Toplanan bu elektriksel sinyal girdilerini etkinleştirmek için nöronlar, sinyalleri akson boyunca iletir ve bu elektriksel sinyalleri bir sonraki aksonlara bağlı nöronlara aktarmış olur [21].

İnsan beyni yaklaşık 86 milyar nörondan oluşmaktadır, her nöron araştırmacılara göre yaklaşık 10 000 diğer nörona bağlanmıştır. Şekil 2.2'de biyolojik sinir hücresinin yapısı gösterilmiştir.



Şekil 2. 2. Biyolojik Sinir Hücresi (Nöron)

Biyolojik sinir hücresi, diğer sinir hücrelerinden gelen elektriksel sinyalleri snapslar ile dendritlerine aktarır. Gelen elektriksel sinyaller snapslar üzerinden zayıflatılır veya güçlendirilir. Sinyaller hücre gövdesine dendritler tarafından iletilirler. Gelen sinyallerin birbirlerini güçlendirme ve zayıflatma etkilerine göre hücre gövdesi aktif hale gelir. Eğer sinyaller yeterince birbirlerini güçlendirir ve bir eşik değerine sahip olabilirse aksona elektriksel sinyal gönderilir ve sinir aktif duruma gelir. Diğer durumda ise aksona elektriksel sinyal gönderilmez ve pasif durumda kalır.

YSA'da, nöronlara perceptronlar denir. Rosenblatt (1958)'de ilk perceptron'u yarattı, algoritma iki katmanlı bir bilgisayar öğrenme ağına dayalı basit ekleme ve çıkarma işlemlerini gerçekleştirmiştir [22]. Werbos (1975), Matematiksel gösterimle Rosenblatt'ın tanımladığı devrede, temel algılayıcıda olmayan, devrenin sinir ağları tarafından işlenemeyen gizli katmanları olan geri yayılım algoritmasını oluşturdu [23].

Bununla birlikte, sinir ağı algoritmasının hesaplama gücü artırılmış ve sinir ağları üzerindeki iyileştirmeler ve çalışmalar sayesinde derin katmanlı ağlara sahip sinir ağları 2000'li yıllardan sonra bilgisayarla paralel bir şekilde ilerleme göstermiştir.

2.2. Yapay Sinir Hücresi

Teknolojik gelişmeler, bilgisayar bilimlerinin son yıllarda hızla ilerlemesini sağlamışlardır. İnsanoğlunun daha önceleri hayal bile edemediği gelişmelerin doğmasına yaratıcı fikirlerin ortaya sunulmasına katkıda bulunmuştur. Bu gelişmelerin en önemlisi ise yapay zekâdır. YSA'da, yapay zekâ biliminin bir alt kolu olarak araştırmacıların yoğun ilgi gösterdikleri bir araştırma alanıdır. YSA'ların karşılaştıkları sorunları farklı örnekler ile öğrenebilme kabiliyeti genelleme yapabilme özellikleri sayesinde çok güçlü ve esnek çözümler üretebilmiş insanın öğrenebilme, anlayabilme, yorumlayabilme ve düşünebilme kabiliyetlerini programlama ile taklit ederek problem ve sorunların çözümünde kullanmaktadırlar.

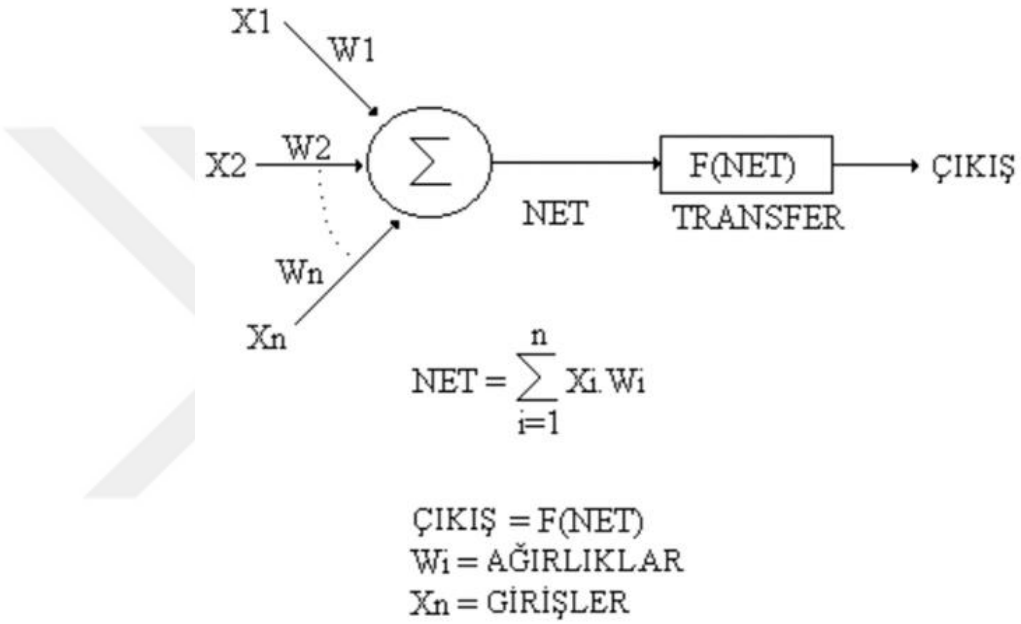
Kohonen(1987) YSA'ları, biyolojik sinir sistemindeki genel yapı gibi, gerçek hayatta cisimlerin birbirleriyle etkileşimini amaçlayan basit elemanların, hiyerarşik düzenlemelerinin paralel ve iç içe bağlantılı bir ağ organizasyonudur, ifadesi ile tanımlamıştır[24]. Başka bir tanıma göre ise YSA, bir insan beyninin temel modelidir. Beyinde, nöron adı verilen hücreler bulunmaktadır. Nöronlar elektrokimyasal veya elektrik sinyalleri ile diğer nöronlara bağlıdır" [25]. Biyolojik Sinir Sistemi ile Yapay Sinir Sistemi arasındaki ilişki aşağıda gösterilmektedir.

Biyolojik Sinir Sistemi		Yapay Sinir Sistemi
• Nöron	→	İşlemci Eleman
• Sinapslar	→	Ağırlıklar
• Dentrit	→	Toplama Fonksiyonu
• Hücre gövdesi	→	Aktivasyon Fonksiyonu
• Aksonlar	→	Yapay Nöron Çıkışı

Yapay sinir sistemleri girdiler (işlemci eleman), ağırlıklar, toplama fonksiyonu, aktivasyon fonksiyonu ve çıktılardan (yapay nöron çıkışı) oluşmaktadır. Girdi katmanı dış ortamlar veya diğer hücreden gelen verileri işlemek için bilgileri alır.

Bu katman problemlere etki eden parametrelerden oluşmaktadır. Toplama fonksiyonu, hücreye gelen bilgilerle bu hücrelerin ağırlıklarının çarpımını toplar ve hücrenin net giriş bilgisini hesaplar. Daha sonra aktivasyon fonksiyonu hücreye gelen net bilgiyi analiz ederek bu girdiye göre karşılık üreteceği çıkış bilgisinin belirlemesini sağlar.

Aktivasyon fonksiyonu tarafından oluşturulan çıktı, dış dünyaya veya bir YSA hücresine ya da kendisine giriş bilgisi olarak iletebilmektedir. Şekil 2.3'de YSA modeli gösterilmektedir.



Şekil 2.3. Yapay sinir ağı modeli

$$v = \sum_{i=0}^{n} W_i\, x_i \quad ya\ da\ \ v = \sum_{i=0}^{n} W_i x_i + b, y = \varphi(v) \tag{2.1}$$

Burada; W- hücrenin ağırlıklar matrisini, x- hücrenin giriş vektörünü, v- hücrenin net girişini, y- hücre çıkışını ve φ (v)-hücrenin aktivasyon fonksiyonu gösterilmektedir. Denklem den, x giriş vektörünün bileşenlerinin dış (geri beslemesiz) girişler olması durumunda hücrenin doğrusal olmayan statik bir işlevi gerçekleştireceği görülmektedir. Tercih edilecek fonksiyonlar probleme göre hız, öğrenme bakımından ise hız ve başarım olarak farklılıklar gösterir. Bu yüzden problemimize göre fonksiyonları doğru seçmemiz başarılı sonuçlar elde etmenizi sağlayacaktır. En yoğun kullanılan toplama fonksiyonların Çizelge 2.1'de gösterilmektedir.

Çizelge 2.1. Yapay Sinir Ağı Toplama Fonksiyonları

Toplam $\text{Net} = \sum_{i=1}^{N} X_i * W_i$	Ağırlık değerleri girdiler ile çarpılır ve bulunan değerler birbirleriyle toplanarak Net girdi hesaplanır.
Çarpım $\text{Net} = \prod_{i=1}^{N} X_i * W_i$	Ağırlık değerleri girdiler ile çarpılır ve daha sonra bulunan değerler birbirleriyle çarpılarak Net Girdi Hesaplanır.
Maksimum $\text{Net} = \text{Max}(X_i * W_i)$	n adet girdi içinden ağırlıklar girdilerle çarpıldıktan sonra içlerinden en büyüğü Net girdi olarak kabul edilir.
Minimum $\text{Net} = \text{Min}(X_i * W_i)$	n adet girdi içinden ağırlıklar girdilerle çarpıldıktan sonra içlerinden en küçüğü Net girdi olarak kabul edilir.
Çoğunluk $\text{Net} = \sum_{i=1}^{N} Sgn(X_i * W_i)$	n adet girdi içinden girdilerle ağırlıklar çarpıldıktan sonra pozitif ile negatif olanların sayısı bulunur. Büyük olan sayı hücrenin net girdisi olarak kabul edilir.
Kumilatif Toplam $\text{Net} = \text{Net(eski)} + \sum_{i=1}^{N} X_i * W_i$	Hücreye gelen bilgiler ağırlıklı olarak toplanır. Daha önce hücreye gelen bilgilere yeni hesaplanan girdi değerleri eklenerek hücrenin net girdisi hesaplanır.

YSA'lar ağırlıklandırılmış şekilde, birbirlerine bağlanmış birçok işlem birimlerinden (nöronlar) meydana gelen matematiksel yapılardır. Bu işlem birimleri arasındaki nöronlardan sinyaller alınarak birleştirilir, dönüştürülür ve matematiksel bir sonuç çıkartır. İşlem birimleri genel yapı itibariyle nöronlara karşılık gelmektedir ve bu nöronlar birbirlerine bağlıdırlar, gerçekleşen bu iletişim sayesinde sinir ağlarını oluşturmaktadır. YSA'lar sinir hücrelerinin birbirleriyle çeşitli şekillerde iletişimiyle oluşmaktadır ve bu yapılar katmanlar olarak kısımlara ayrılır. Bu yapıyı örnek alarak, çeşitli teknolojik alanlarda elektronik devreler ya da bilgisayarların gelişimini sağlayan yazılımlar olarak ifade edilir.

YSA, insan beyninin çalışma prensibinden esinlenerek, bilgiyi gerekli yöntemlerle işleyerek bir öğrenme sürecine dahil eder. İşlenen bilgiler daha sonra toplanarak, hücreler arasında güçlendirilerek aktarılır. YSA'lar toplanan veriler saklayarak, genelleme özelliği ile geliştiren bir işlemci yapısına sahiptir. YSA, gelişmiş birçok alanda aşağıdaki özellikleri ve avantajları sebebiyle etkili olmuş ve uygulama alanı bulmuştur.

Genelleme

YSA performansı çoğunlukla genelleme kabiliyetine bağlıdır. YSA' nın genelleştirilmesi, görünmeyen verileri işlemek yeteneğidir. Ağın genelleme kabiliyeti, çoğunlukla sistem karmaşıklığı ve ağın eğitimi ile belirlenir.

Ağ aşırı eğitimli veya sistem karmaşıklığı (veya serbestlik derecesi) eğitim verisinden nispeten fazla olduğunda zayıf genelleme görülmektedir. Verilere uyabilen daha küçük bir ağ, iyi bir genelleme kabiliyetine sahip
olacaktır.

Doğrusal Olmama

YSA sinir sistemleri, doğrusal olmayan modelleri de içermektedir. Toplanan bazı veriler karmaşık doğrusal olmayan fonksiyonları ve basit doğrusal olmayan hesaplamalar bulunduran bir dizi içerirler. Aslında, nasıl yapılandırıldığına bağlı olarak, YSA'lar doğrusal olmayan bir fonksiyona çevrilebilirler, bu da onları oldukça güçlü bir model sınıfı yapar. Bu özelliğin YSA'lar için ayrılmamış olduğunu, kernel yöntemlerinin de "evrensel tahmin ediciler" olarak kabul edildiği unutulmamalıdır. Birden çok katmana sahip olan sinir ağlarının, diğer fonksiyonlara kıyasla keyfi işlemleri gerçekleştirirken daha etkili olduğu da ortaya çıkmaktadır.

Öğrenme

YSA' ların, öğrenebilme özelliği insanların düzenli yaşamlarında ve faaliyetlerinde ki öğrenebilme kabiliyetleriyle yakından ilgilidir. Bir eylem gerçekleştiririz ve belirli bir görevde daha iyi nasıl ilerleyeceğimizi anlamak için bir eğitmene ihtiyaç duyar ve kendimizi geliştiririz. Benzer şekilde, sinir ağları, girdiye bir cevap olarak neyin üretilmesi gerektiğini tanımlamak için bir eğiticiye ihtiyaç duyar. Gerçek değer ile ağ tarafından çıkarılan değer arasındaki farka bağlı olarak, bir hata değeri hesaplanır ve sistemden geri gönderilir. Ağın her bir katmanı için hata değeri analiz edilir ve bir sonraki giriş için eşik ve ağırlıkları ayarlamak için kullanılır. Bu şekilde ağ, her bir koşulu ve değerleri nasıl analiz edeceğini öğrenmiş olur.

Uyarlanabilirlik

YSA' lar, birçok sistemin öğrenme aşamasında bazı sorunlarla karşılaşmaktadır. Bu tür ortamlarda öğrenme, öğrenme sistemleri için önemli bir zorluğu temsil etmektedir. Bu bağlamda, öğrenme için kullanılan model gerçek zamanda çalışmalı ve sürecin gerekliliklerine bağlı olarak kontrol parametrelerini, hatta yapılarını bile ayarlayarak kendi başına hareket etme ve tepki verme yeteneğine sahip olmalıdır.

YSA’ lar eğitim aşamasından sonra, ağın çoklu katmanlarından geçmek ve tüm girdi örnekleri için hata değerini en aza indirmek amacıyla ağın parametrelerini ve eşik değerini ayarlayabilme kabiliyetine sahiptirler. Yapılan bu işlemler geri bildirim olarak bilinir ve hata değeri minimumda tutulana kadar bir ağ üzerinden sürekli olarak uygulanır. Bu noktada, sinir ağı artık böyle bir eğitim sürecine ihtiyaç duymaz ve ayarlama yapmadan çalıştırılmasına izin verilir. Ağ, daha sonra ayarlanmış ağırlıklar ve eşikleri kılavuz olarak kullanarak uygular.

Hata Toleransı

YSA'nın önemli bir özelliği olan hata toleransı, bir ağın önemli bölümlerinin kaybolması durumunda güvenilirliklerini garanti eder. Bir sistemin giriş çıkış ilişkisinin doğruluğunu ve gerçekliğini sağlar. Hata toleransı, özellikle yüksek derecede değişken veya “başarısız” sistemlerde ve YSA yapımında da önemli bir husustur. Hata toleranslı bir YSA, bileşenlerinden bazıları beklenmedik şekilde hasar görmüş olsa bile, normal olarak veya en azından belirli bir dereceye kadar çalışmak üzere tasarlanmış özel bir YSA sistemi içerir.

YSA’larda donanım arızalarının üstesinden gelmek için farklı stratejiler sunulmaktadır. Katmanlı besleme yönlendirme ağlarının gizli katmanındaki bir veya daha fazla birimin arızalanması özellikle ele alınmaktadır. Farklı tipte yeniden eğitim teknikleri araştırılır ve gerekli sınıflandırma çalışmaları, belirli sınıflandırma görevleri için içsel temsillerle ilişkilendirilir. Ardından, gerçek hata toleransı elde etmek için, yani ağ bir veya daha fazla gizli ünitenin arızasından sonra doğru şekilde çalışmaya devam etmek için pratik bir teknik sunulur.

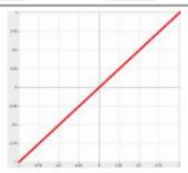
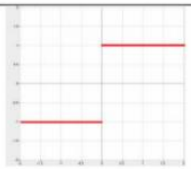
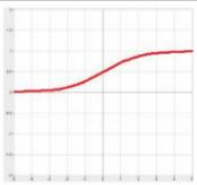
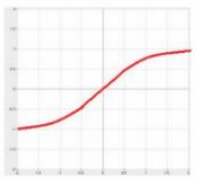
Donanım ve Hız

YSA’ lar için iki farklı hız özelliği vardır, bunlardan bir tanesi öğrenme hızı diğeri ise işlem hızıdır. Öğrenme hızı hesaplama ve güncelleme ile ilgilidir, öğrenme aşamasında bağlantı ağırlıkları bir sistemin giriş ve çıkış katmanlarını ne kadar sürede eşleştirme hızıdır. İşleme hızı, transfer fonksiyonunun hesaplanması olarak ifade edilir. Bu hesaplamaya etki eden, en önemli faktörlerden başında sistem içerisinde kullanılan donanım özellikleridir.

2.2.1. Aktivasyon Fonksiyonları

Aktivasyon fonksiyonları ya da diğer bir ifadeyle transfer fonksiyonları, öğrenme parametresi olarak da tanımlanmaktadır. Bu fonksiyonlar nöronların çıkış bandını beklenen değerler aralığında sınırlamamıza yarar. Bu değerler genellikle [0,1] veya [-1,1] aralığındadır. Bu fonksiyonlar aynı zamanda, bir eşik (kutuplama bias) değeri uygulaması ile aktivasyon fonksiyonu yükseltebilir. YSA'da kullanılan farklı fonksiyonların türevi alınabilir ve devamlılık göstermesi gerekmektedir. YSA'nın kullanım amacına göre tek veya çift girişli aktivasyon fonksiyonları kullanılabilmektedir. Transfer fonksiyonlarının doğrusal veya doğrusal olmayan problemlerde kullanılması, YSA'da karmaşık problemlerin çözümlenmesini sağlamıştır [26]. Problemimizi buna göre modellemeniz değerler üretmemiz, ağırlıklar belirleyip daha sonrasında en uygun fonksiyonlar yardımıyla algoritmamızı oluşturmanız gerekli. En çok tercih edilen aktivasyon fonksiyonları Çizelge 2.2'de gösterilmiştir.

Çizelge 2.2. Yapay Sinir Ağı Aktivasyon Fonksiyonları

Doğrusal (Lineer) Aktivasyon Fonksiyonu		F(NET)=A* NET (A sabit bir sayı)	Doğrusal problemler çözmek amacıyla aktivasyon fonksiyonu doğrusal bir fonksiyon olarak seçilebilir. Toplama fonksiyonundan çıkan sonuç, belli bir katsayı ile çarpılarak hücrenin çıktısı olarak hesaplanır.
Adım (Step) Aktivasyon Fonksiyonu		$F(Net)=\begin{cases}1 \text{ if Net>Eşik Değer}\\0 \text{ if Net<=Eşik Değer}\end{cases}$	Gelen Net girdinin belirlenen bir eşik değerin altında veya üstünde olmasına göre hücrenin çıktısı 1 veya 0 değerini alır.
Sigmoid Aktivasyon Fonksiyonu		$F(Net)=\frac{1}{1+e^{-Net}}$	Sigmoid aktivasyon fonksiyonu sürekli ve türevi alınabilir bir fonksiyondur. Doğrusal olmayışı dolayısıyla yapay sinir ağı uygulamalarında en sık kullanılan fonksiyondur. Bu fonksiyon girdi değerlerinin her biri için 0 ile 1 arasında bir değer üretir.
Tanjant Hiperbolik Aktivasyon Fonksiyonu		$F(Net)=\frac{e^{Net}+e^{-Net}}{e^{Net}-e^{-Net}}$	Tanjant hiperbolik fonksiyonu, sigmoid fonksiyonuna benzer bir fonksiyondur. Sigmoid fonksiyonunda çıkış değerleri 0 ile 1 arasında değişirken hiperbolik tanjant fonksiyonunun çıkış değerleri -1 ile 1 arasında değişmektedir.
Eşik Değer Fonksiyonu		$F(Net)=\begin{cases}0 \text{ if Net<=0}\\Net \text{ if } 0<Net<1\\1 \text{ if Net>= 1}\end{cases}$	Gelen bilgilerin 0 dan küçük-eşit olduğunda 0 çıktısı, 1 den büyük-eşit olduğunda 1 çıktısı, 0 ile 1 arasında olduğunda ise yine kendisini veren çıktılar üretilebilir.
Sinüs Aktivasyon Fonksiyonu		F(Net) = Sin(Net)	Öğrenilmesi düşünülen olayların sinüs fonksiyonuna uygun dağılım gösterdiği durumlarda kullanılır.

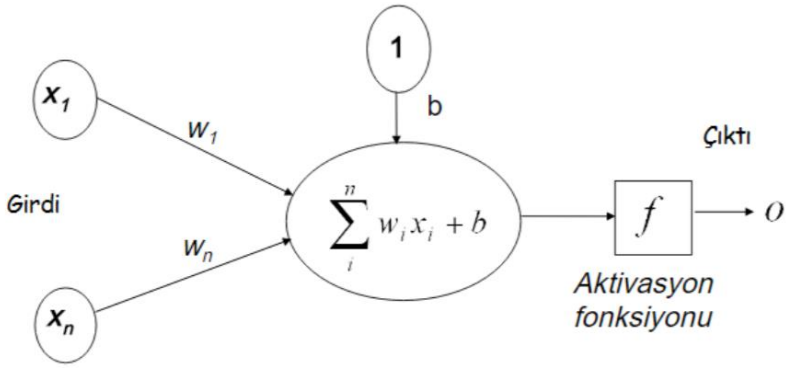
2.2.2. Yapılarına Göre Yapay Sinir Ağları

YSA'lar ağın yapısına göre, ağın içindeki akış yönü veya sinirler arasındaki bağlantıların yönlerine göre birbirinden ayrılırlar. Bu modeller ileri ve geri beslemeli olarak, iki gruba ayrılmaktadır [27]. Öğrenme kuralına göre; Kohonen, Hopfield, Hebb, ve Delta olmak üzere dört grupta incelenir. Ağın öğrenmesi için gerekli olan eğitim verilerinin belirlenmesi de son derece önemlidir. Eğitim verileri en az bilgi ile optimum öğrenmeyi sağlayacak şekilde belirlenmelidir [28].

YSA' lar öğrenme algoritmalarına göre; danışmanlı, danışmansız ve destekleyici olmak üzere üç grupta incelenir. Uygulama alanlarına göre ise; off-line ve on-line, ağırlık matris değerlerine göre; sabit veya değişken, ağırlık matrislerine göre; simetrik ya da asitmetrik ve ağdaki düğümlerin özellikleri, aktivasyon fonksiyonlarının öğrenme zamanına göre dinamik veya statik oluşuna göre de gruplandırılabilir. Genel olarak ileri beslemeli sinir ağlar örüntü tanıma sorunlarında, geri beslemeli sinir ağlar ise optimizasyon sorunlarında tercih edilmektedir.

2.2.2.1. İleri Beslemeli Tek Katmanlı Ağlar

İleri beslemeli ağ mimarisinde girdiler doğrudan çıkış nöronlarına düzenli katmanlar ile bağlıdır. Bu bağlantılar tek yönlü ve sadece bir sonraki katmana bağ bulunmaktadır. YSA'ya gelen bilgiler ilk olarak giriş katmanına, ara katmana ve çıkış katmanından geçerek dış dünyaya iletilir, sadece çıkış katmanı değeri hesaba katıldığı için tek katmanlı adını almıştır [29]. Şekil 2.4' de ileri beslemeli tek katmanlı ağ gösterilmiştir.



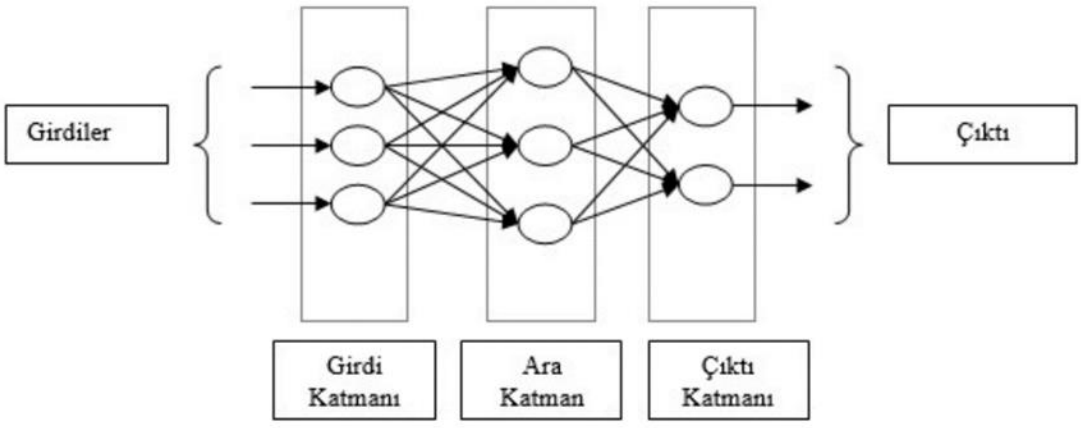
Şekil 2. 4. İleri Beslemeli Tek Katmanlı Yapay Sinir Ağı

Bu ağlar en basit tipte ki sinir ağlarıdır, tek katmanlı bir çıkış düğümünden oluşan tek katmanlı bir perceptron ağıdır; Giriş katmanı dahil, bu yapıda iki katman vardır. Bununla birlikte, giriş katmanın da hiçbir hesaplama yapılmadığından bu katman sayılmaz. Giriş sinyalleri çıkış katmanı üzerine ağırlıklarla aktarılır ve çıkış katmanındaki nöronlar çıkış sinyallerini hesaplar. Girişler, bir seri ağırlıkla doğrudan çıktılarla beslenir. Bu şekilde, en basit besleme iletimi ağı olarak düşünülebilir. Ağırlıkların ve girdilerin ürünlerinin toplamı, her düğümde hesaplanır ve eğer değer bir eşiğin tipik olarak 0'ın üzerindeyse, nöron patlar ve aktive edilen değeri alır, aksi halde devre dışı bırakılan değer olan -1'i alır [22].

Eşik değeri, ikisi arasında olduğu sürece, aktif ve devre dışı bırakılmış durumlar için herhangi bir değer kullanılarak bir perceptron oluşturulabilir. Perceptronlar genellikle delta kuralı olarak adlandırılan basit bir öğrenme algoritması ile eğitilebilir. Hesaplanan çıktı ile örnek çıktı verileri arasındaki hataları hesaplar ve bunu ağırlıklara göre bir ayarlama oluşturmak için kullanır, böylece bir degrade alçalma biçimi uygular.

2.2.2.2. İleri Beslemeli Çok Katmanlı Ağlar

YSA'nın öğrenmesi istenilen, girdi ve çıktı arasındaki ilişkiler doğrusal değil ise daha önce anlatılan modelleri kullanarak öğrenmeyi gerçekleştirmek mümkün olmayacaktır. Bu tür karmaşık yapıların eğitilebilmesi için daha gelişmiş modellere ihtiyaç duyulur. Bu modeller arasında en önemlisi ise çok katmanlı ağlardır. [30]. Şekil 2.5'de ileri beslemeli çok katmanlı ağ yapısı gösterilmiştir.



Şekil 2. 5. İleri Beslemeli Çok Katmanlı Yapay Sinir Ağı

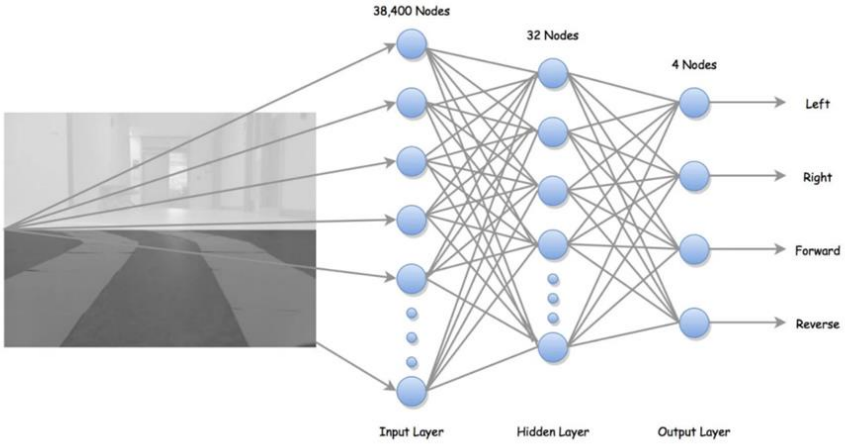
Bu ağlar üç katmandan oluşur. Giriş katmanındaki düğümler dış dünyadan alınır ve çıkış katmanındaki düğümleri kullanıcıya çıktı olarak sunar. Ara katmanlardaki düğümler genellikle gizli katmanlar olarak adlandırılır, önceki katmandan girdi alır ve onu bir sonraki katmana iletir [31].

Çok katmanlı ağlar, en popüler geri yayılım olan çeşitli öğrenme teknikleri kullanır. Burada, çıkış değerleri, önceden tanımlanmış bazı hata fonksiyonlarının değerini hesaplamak için doğru cevapla karşılaştırılır. Çeşitli tekniklerle, hata daha sonra ağ üzerinden geri beslenir. Bu bilgiyi kullanarak, algoritma hata fonksiyonunun değerini azaltmak için her bağlantının ağırlığını biraz küçük miktarlarda ayarlar. Yeterli sayıda eğitim döngüsü için bu işlemi tekrarladıktan sonra, ağ genellikle hesaplama hatalarının küçük olduğu bazı durumlara yaklaşacaktır. Bu durumda, ağın belirli bir hedef işlevi öğrendiği söylenebilir.

Bu ağ sınıfı, genellikle ileriye dönük olarak birbirine bağlı birden fazla hesaplama birimi katmanından oluşur. Tek bir katmandaki her nöron, sonraki tabakanın nöronlarına bağlantılar yöneltmiştir. Birçok uygulamada, bu ağların birimleri bir aktivasyon fonksiyonu olarak bir sigmoid fonksiyonu uygular.

YSA'ların bir avantajı, ağ eğitim gördükten sonra eğitilmiş parametrelerin yüklenmesi gerektiğidir, bu nedenle tahmin çok hızlı olabilir. Giriş görüntüsünün yalnızca alt yarısı, eğitim ve tahmin amaçları için kullanılır. Şekil 2.6' da tez çalışmasında elde edilen görüntü verilerinden tek kare görüntünün ileri beslemeli çok katmanlı YSA örneği gösterilmektedir.

Giriş katmanında 38 400 düğüm ve gizli katmanda 32 düğüm vardır. Gizli katmandaki düğüm sayısı oldukça keyfi olarak seçilmiştir. Orada her bir düğüm direksiyon kontrolü talimatları göstermektedir. Çıkış katmanındaki dört düğüm ifadeleri şunlardır: sol, sağ, ileri ve sırasıyla ters düğümleridir, fakat bu projede ters düğüm noktası kullanılmaz, yine de çıkış katmanına dahil edilir.

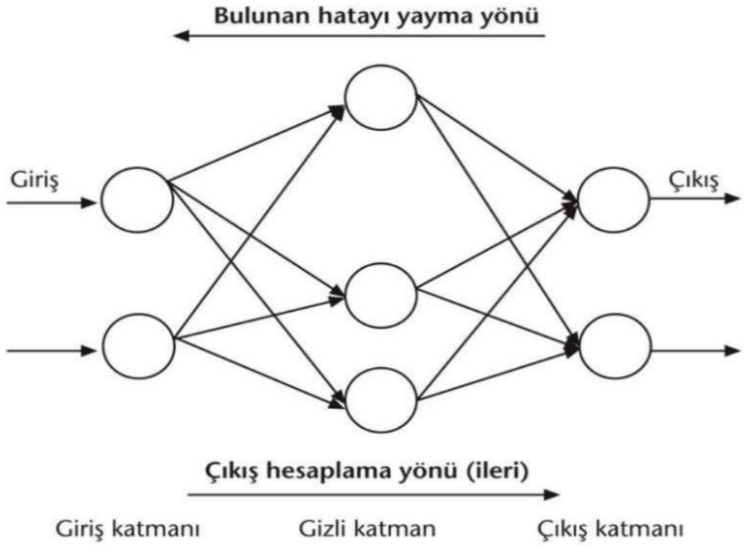


Şekil 2.6. Tek Kare Görüntünün İleri Beslemeli Çok Katmanlı YSA Örneği

2.2.2.3. Geri Beslemeli Ağlar

Geri beslemeli ağlar genel olarak danışmansız öğrenme kuralının gerektirdiği yapılarda kullanılmaktadır. Hopfield ağı, geri beslemeli bir mimariye sahip YSA'dır. Geri besleme, diğer katmanlar arasındaki hücreler arasında olabileceği gibi, tek bir katmandaki hücreler arasında da olabilmektedir

İleri beslemeli YSA yapılarından farklı olarak, geri beslemeli ağlarda, bağlam (context) bölümü bulunmaktadır. Bu tabaka gizli tabakada ki çıktıları güçlendirerek, 29 tekrar ile gizli bölüme girdi olarak aktarmaktadır [32]. Şekil 2. 7'de iki katmanlı ve çıkışlarından giriş katmanına geri beslemeli bir YSA yapısı görülmektedir



Şekil 2. 7. Geri Beslemeli Yapay Sinir Ağı

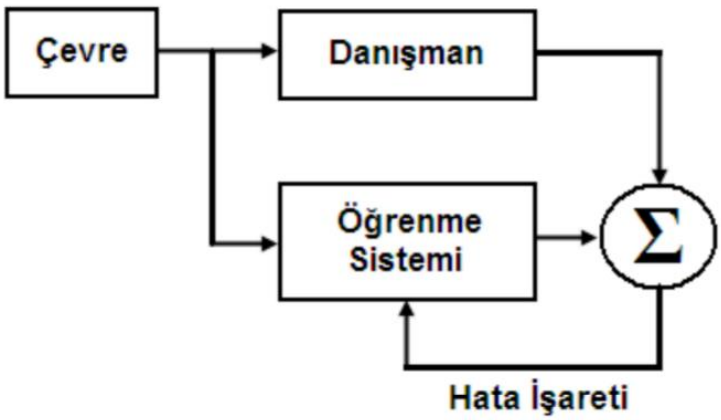
2.2.3. Öğrenme Algoritmalarına Göre Yapay Sinir Ağları

YSA öğrenme algoritmalarına göre, danışmanlı, danışmansız ve destekleyici olmak üzere 3'e ayrılır.

2.2.3.1. Danışmanlı Öğrenme

Danışmanlı öğrenme diğer bir ifadeyle denetimli öğrenmedir. Adında anlaşılacağı üzere, bir öğretmen gözetiminde gerçekleşir. Bu öğrenme süreci bağımlıdır. Denetlenen öğrenme altında YSA eğitimi sırasında, girdi vektörü bir çıkış vektörü üretecek olan ağa sunulur. Bu çıktı vektörü istenen hedef çıktı vektörüyle karşılaştırılır. Gerçek çıkış ile istenen hedef çıkış vektörü arasında bir fark varsa bir hata sinyali üretilir. Bu hata sinyaline dayanarak ağırlıklar, istenen çıktı ile eşleşene kadar ayarlanacaktır [33].

Danışmanlı öğrenim, girdiyi bir girişe ekleyerek, örnek giriş çıkış çiftlerine dayalı olarak eşleştirilmesini öğreten, makine öğrenme görevidir. Danışmanlı öğrenmede bir dizi eğitim örneğinden oluşan verilerinden her bir örnek, bir giriş nesnesinde, tipik olarak bir vektör ve istenen çıkış değerinden oluşan çiftlerdir. Denetlenen bir öğrenme algoritması, eğitim verilerini analiz eder ve yeni örnekleri haritalamak için kullanılabilecek bir çıkarılmış fonksiyon üretir. En uygun senaryo, algoritmanın görünmeyen örnekler için sınıf etiketlerini doğru bir şekilde belirlemesine izin verecektir. Bu öğrenme algoritmasının eğitim aşamasında görülmeyen verilere bir genelleştirilme özelliği eklenmektedir. Şekil 2.8'de danışmanlı öğrenme yapısı gösterilmiştir.

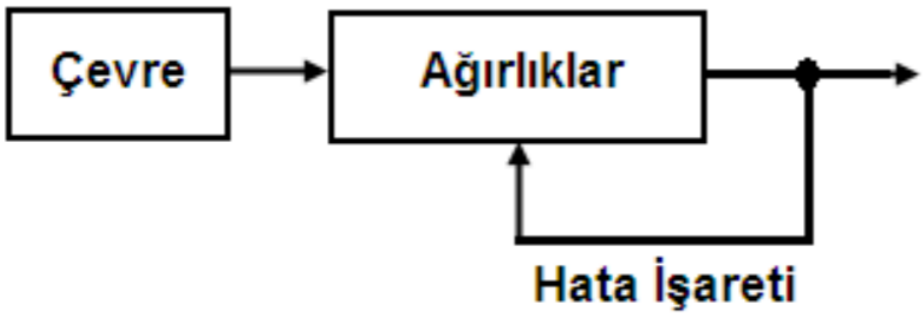


Şekil 2. 8. Danışmanlı Öğrenme Yapısı

2.2.3.2. Danışmansız Öğrenme

Danışmansız öğrenmenin bir diğer adı da denetimsiz öğrenmedir. Bir öğretmen gözetimi olmadan yapılır ve bu öğrenme süreci bağımsızdır. Denetimsiz öğrenim altında YSA eğitimi sırasında, benzer tipteki giriş vektörleri kümeler oluşturmak için birleştirilir. Yeni bir giriş modeli uygulandığında, nöral ağ girdi modelinin ait olduğu sınıfı belirten bir çıkış tepkisi verir. Bu bağlamda, çevreden istenen çıktının ne olması gerektiği ve doğru ya da yanlış olup olmadığı konusunda geri bildirimde bulunulmayacaktır. Bu nedenle, bu öğrenme türünde ağın kendisi kalıpları, giriş verilerinden gelen özellikleri ve çıktı üzerinden girdi verilerinin ilişkisini keşfetmelidir [34].

Denetlenen öğrenme sürecinin aksine hem girdi hem de çıktı verilerinin ağa sunulduğu yerde, denetimsiz öğrenme süreci, ağa sunulan tüm veri kümesi ile karakterize edilir ve daha sonra veriler arasındaki ilişkileri arar. Bu yaklaşım temel olarak sınıflandırma amaçları için kullanılır. Denetlenmeyen öğrenme sadece ağın iç yapısına dayanır. Denetimsiz öğrenme süreci sırasında, nöronlar rekabet edebilir, iş birliği yapabilir veya her ikisi olabilir. Şekil 2.9'da danışmansız öğrenme yapısı gösterilmiştir.



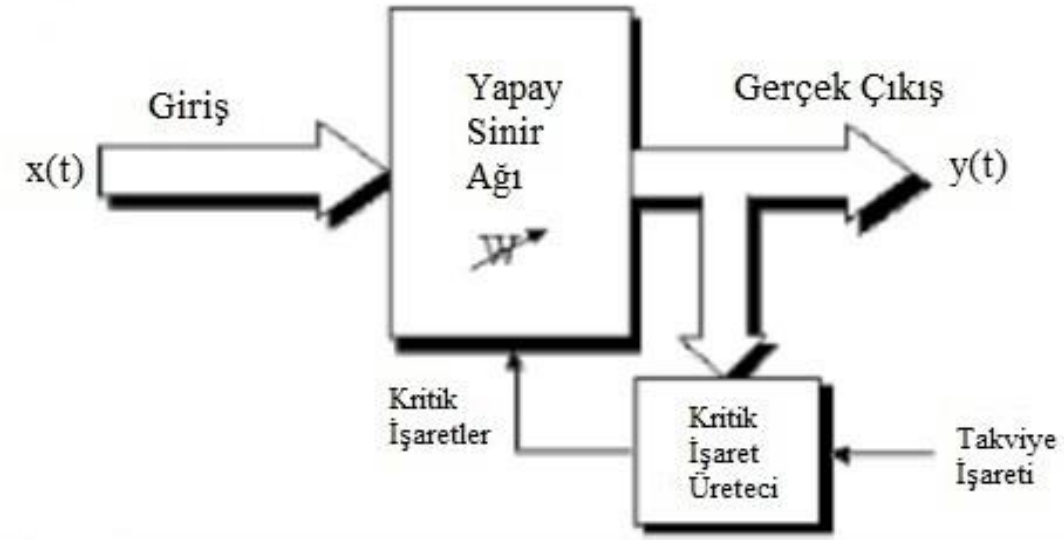
Şekil 2. 9. Danışmansız Öğrenme Yapısı

2.2.3.3. Destekleyici Öğrenme

Bu öğrenmenin diğer adı da rekabetçi öğrenmedir. Bu yaklaşımda ağın her iterasyonu sonunda elde ettiği sonucun iyi veya kötü olup olmadığına dair bir bilgi verilir. Ağ bu bilgilere göre kendini yeniden düzenler.

Rekabetçi öğrenme şekli, denetimsiz öğrenme içinde YSA düğüm girdilerinin, veri alt kümesiyle yanıt hakkı için rekabet ettiği, Hebbian öğrenmenin bir çeşididir. Destekleyici öğrenme, ağdaki her düğümün uzmanlığını artırarak çalışır. Veri içinde kümeleri bulmak için çok uygundur.

Rekabetçi öğrenme ilkesine dayanan modeller ve algoritmalar, vektör nicelleştirme ve kendi kendini organize eden haritaları içermektedir. Şekil 2.10'da destekleyici öğrenme blok diyagramı gösterilmiştir.



Şekil 2.10. Destekleyici Öğrenme Blok Diyagramı

3. RC (Radio Control) OTONOM ARAÇ TASARIMI

Bu bölümde RC otonom araç tasarımı için gerekli olan RC araç şasisi, Elektrik- Elektronik elemanların araç üzerinde yapılmış olan elektronik bağlantıları, iletişim modülü ve tercih edilmiş olan elektronik devre elemanlarının özellikleri ve neden kullanıldıkları açıklanmıştır.

Bu uygulama için kullanılmış olan devre elemanları Mikrobilgisayar (Raspberry Pi3 model B), Kamera (Raspberry Pi Kamera V2.1), PWM sürücü (PCA9685 16 kanal 12bit), iletişim için kullanılan Mobil Modem (Huawei e5775), Motorların hareketini sağlamak amacıyla kullanılmış olan Lityum polimer bataryalar, Motor sürücüsü ve Mikrobilgisayarın optimal seviyede çalışabilmesi için 2 A çıkışa sahip Powerbank ve tankın hareketini sağlamakla görevli olan fırçasız ve servo motorlar ayrı ayrı incelenmiş, bu devre elemanları ile ilgili bilgiler verilmiş ve bağlantı şemaları gösterilmiştir.

3.1. Elektrik-Elektronik Komponentler

Bu başlık altında RC araç şasisi ve üzerinde kullanılmış olan elektronik komponentler işlenmiştir. Bu parçalar Terremoto V2 1/10 elektrikli arazi aracı, motor çeşitleri, batarya, mikrobilgisayar, kamera modülü ve iletişim ekipmanları incelenmiştir.

3.1.1. RC Araç

Projemizin temel yapısını oluşturan RC araç şasi tercihinde en önemli etken bir yarış parkurunda hızlı hareket edebilmesi, zemin ve hava şartlarına dayanaklı yapısı ve güçlü motorlara sahip olması bakımından Terremoto firmasının üretmiş olduğu Terremoto10 V2 1/10 ölçekli fırçasız elektrikli arazi aracı modeli kullanılmıştır. 4WD (4 çeker) olan Terremoto10 modelinin özellikleri şu şekildedir. Özel olarak tasarlanmış şasi, eloksallı alüminyum (özel bir yüzey kaplama), uzun parkurlar için ayarlanabilir süspansiyon, stratejik olarak yerleştirilmiş elektronik ekipmanlar, merkeze yerleştirilmiş LİPO pil ve suya dayanıklı 15 Kg metal dişli direksiyon servo özelliklerine sahiptir. Dört sürüş mesafeli alüminyum yağ dolu süspansiyonlar sayesinde Terremoto-10 V2'nin neredeyse her yere gidebilmesi için yeterince uzun menzilli sönümlemeyi sağlar.

Güçlü 3000 KV (devir katsayısı) fırçasız motor ve suya dayanıklı 60A 2s-3s LİPO pil için hazır ESC (elektronik hız kontrol)'ye sahiptir. Resim 3.1'de RC araç şasi özellikleri gösterilmiştir.

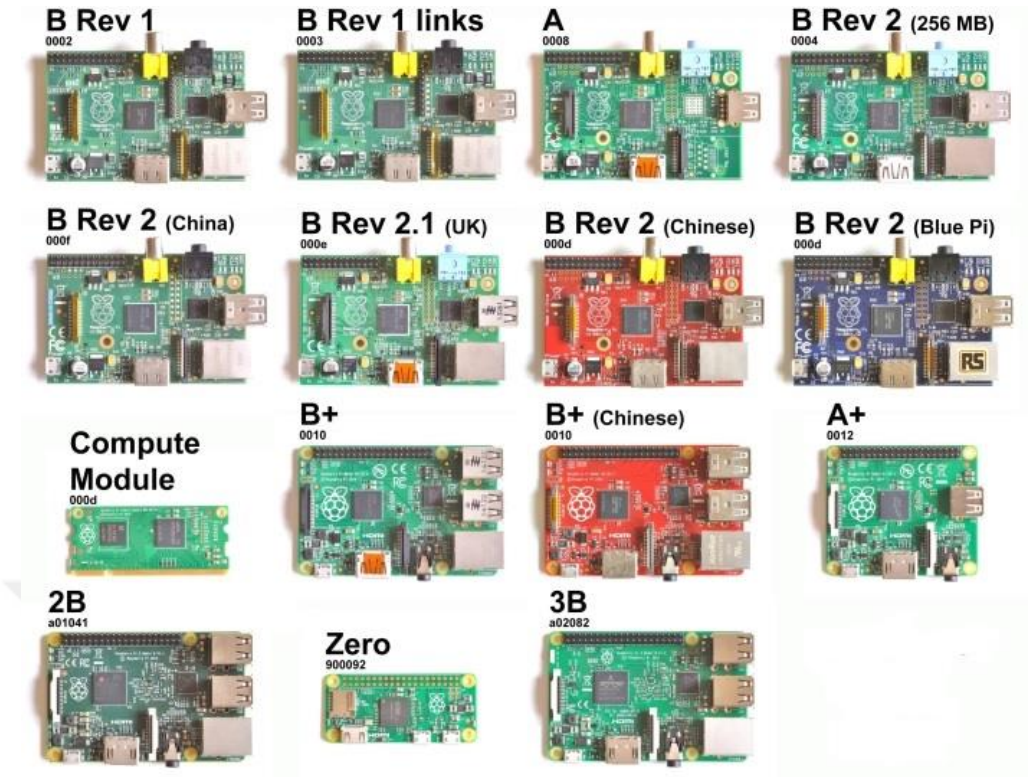
Resim 3.1. Terremoto10 Fırçasız Elektrikli Arazi Aracı

3.1.2. Mikrobilgisayar

RPİ, kâr amacı gütmeyen Raspberry Pi Vakfı tarafından, kendi işletim sistemi olan, kredi kartı boyutlarında tek kart bilgisayar olarak geliştirilmiştir. Sadece bir geliştirme kartı olmanın ötesinde avantajlara sahiptir. Bir monitöre ve klavyeye bağladığınızda Linux yüklü işletim sistemi ile birlikte bir kişisel bilgisayar halini alır. Akıllı TV, oyun konsolları ve birçok elektronik aletin geliştirilmesinde RPİ önemli bir kullanım alanına sahiptir.

RPİ, Raspbian adı verilen Linux tabanlı bir işletim sistemi ile çalışmaktadır. Cihaz bu işletim sistemi haricinde, Pardus ARM, Arch Linux ve Windows 10 IoT Core sistemlerinide desteklemektedir. RPİ aynı zamanda Python, BBC Basic, C ve Perl programlama dilleri ile programlanabilir. Resim 3.2'de günümüze kadar çıkarılan RPİ modelleri gösterilmektedir.

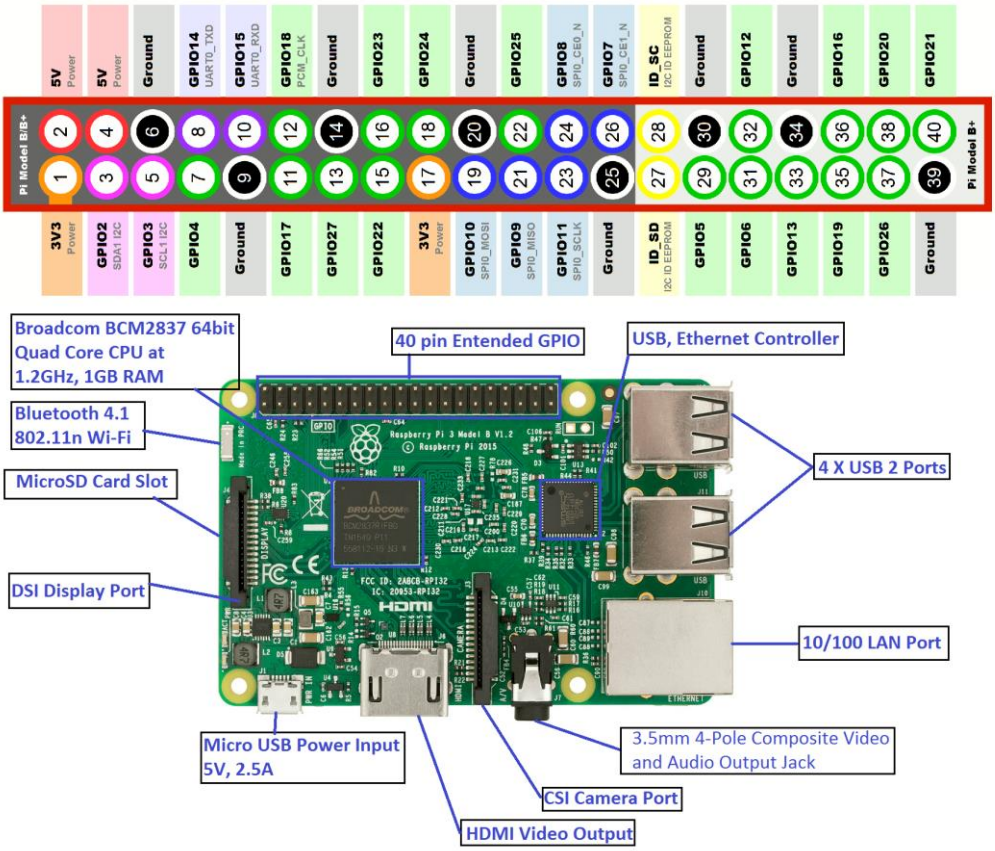
Projemizde kullanılan RPİ, Mart 2016'da satışa sunulmuş olan RPİ 3 model B olanıdır. RPİ mikrobilgisayarı kullanmamızda ki önemli etkenler, küçük boyutlarda, yüksek işlemci hızına sahip olması ve Python programlama dilini desteklemesidir.



Resim 3.2. Örnek RPİ Modelleri

Dahili olarak gelen Wi-Fi ve Bluetooth özellikleri sayesinde araç ile uzaktan bağlantı erişimine olanak sağlamıştır. Sistem üzerinde 1.2 Ghz, 4 çekirdekli 64 bit ARM Cortex-A53 işlemci birimini içerisinde Broadcom BCM2837 mikroişlemcisine bulunmaktadır. Bellek kapasitesi 1 GB ve 2 çekirdekli Videocore IV GPU grafik işlem birimine sahiptir. Üzerinde 4 adet USB 2.0 port, 10/100 Mbit/s destekli ethernet portu ve HDMI çıkışı bulunmaktadır. [35, 36]. Resim 3.3'de RPİ kartı pin girişler ile gösterilmiştir.

RPİ 3 Model B kartı üzerinde 40 tane GPIO (General Purpose Input/Output) pini bulunmaktadır. Bunlardan 2 tanesi 5V, 2 tanesi 3.3V, 8 tanesi GND ve 2 tanesi EEPROM için ayrılmış pinlerdir. Geriye kalan 26 pin Input/Output pinidir. Bu Input/Output pinlerinden 5 tanesi SPI (Serial Peripheral Interface) haberleşmesini, 2 tanesi I^2C (Inter-Integrated Circuit) haberleşmesini ve 2 tanesi de UART (Universal Asynchronous Receiver Transmitter) haberleşmesini desteklemektedir. RPİ üzerinde bulunan GPIO pinlerini programlamada Raspbian işletim sistemiyle birlikte gelen Python derleyicisini kullanmaktayız.



Resim 3.3. RPİ 3 Pinleri

Raspberry Pi 3 Teknik Özellikleri:

- 64-bit quad-core ARMV 8 işlemci
- 1.2 GHz
- 1 GB RAM
- Dahili Wifi-BCM43143
- Bluetooth 4.1 (Bluetooth Low Energy-BLE)
- 40 Adet GPIO
- 4 Adet USB 2
- 4 uçlu Stereo çıkışı ve Composite video çıkışı
- Full HDMI
- RPİ Kamera bağlantısı için CSI kamera portu
- RPİ 7” dokunmatik ekran için DSI ekran portu
- Micro SD soketi
- Güncellenmiş güç katı (2,5 A’ekadar destekleniyor)
- Güç ve aktivasyon ledi.

3.1.3. Kamera Modülü

RC otonom aracın, otopilot eğitimi aşamasında görüntülerin işlenebilmesi, şerit çizgilerini araç üzerinden anlık olarak ana bilgisayara video aktarımı ve fotoğraf kaydı yapabilmesi için RPİ modeli ile uyumlu ve yüksek çözünürlüğe sahip RPİ için on board olarak tasarlanmış, RPİ Kamera V2.1 modülü kullanılmıştır. Yüksek kaliteli 8 mega piksel çözünürlüklü Sony IMX219 görüntü sensörlü ve özel sabit odak lense sahiptir. CCTV güvenlik kamera, hareket algılama ve zaman atlamalı işlemlerde tercih, edilirler. 3280 x 2464 piksel fotoğraf çekim ve 1080p30, 720p60 ve 640x480p90 video çekimi yapabilmektedir. Kamera Modülü, RPİ üzerinde bulunan ve kameralar için özel olarak tasarlanmış CSI özel arabirimi ile doğrudan kullanılabilir. Resim 3.4'de kullanılan kamera modülü gösterilmektedir.

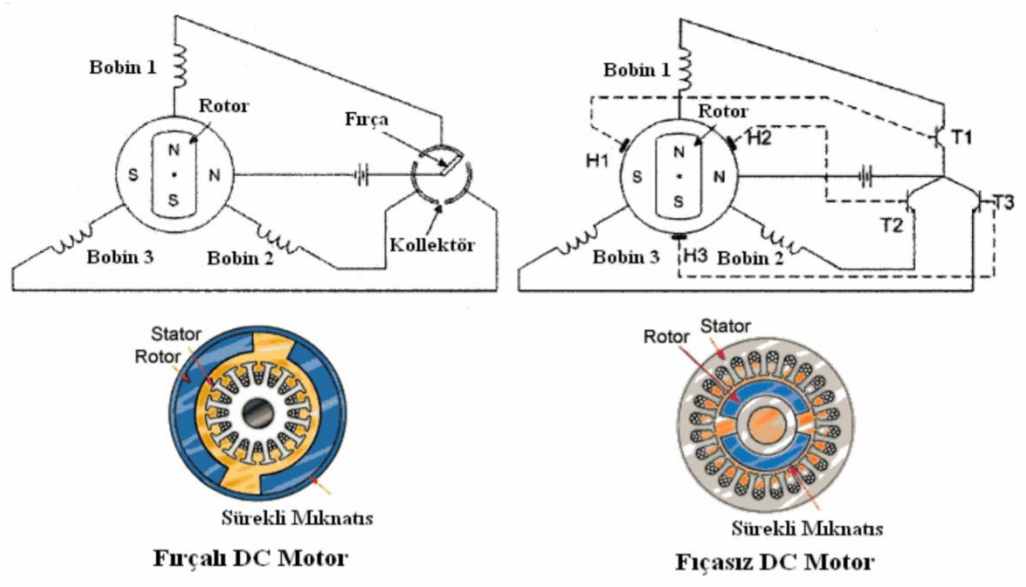
Resim 3.4. RPİ Kamera V2.1 Modülü

Raspberry Pi Kamera V2.1 Teknik Özellikler:

- Sabit Odaklı Lens
- 8 mega piksel doğal çözünürlüklü sensör- 3280 x 2464 piksel fotoğraf çekebilir
- 1080p30, 720p60 ve 640x480p90 video destekliyor.
- Boyut: 5mm x 23mm x 9mm
- Ağırlık: Sadece 3gram
- Raspbe RPİ rry Pi işletim sistemi Raspbian'ın son versiyonu ile uyumlu olarak çalışmaktadır.
- Kısa ribon kablo ile RPİ 'ye doğrudan bağlanabilir

3.1.4. Motorlar

Projemizde kullanılan Terremoto-10 şasi üzerinde 2 adet motorumuz bulunmaktadır. Şasinin arka kısmında konumlandırılmış, tekerliklere hız veren 3000KV'lık fırçasız bir motor ve ön tarafta konumlandırılmış direksiyon kontrolünü sağlayan servo motor bulunmaktadır. Fırçasız motora sahip bir araç tercihimizin sebebi, fırçasız elektrik motorları ile fırçalı motorları birbirinden ayıran en önemli fark, fırçalı motorlar mekanik yapıda, fırçasız motorlar ise elektronik yapıda kontrol edilebilmesidir. Şekil 3.1'de fırçalı ve fırçasız motor iç yapısı gösterilmektedir.



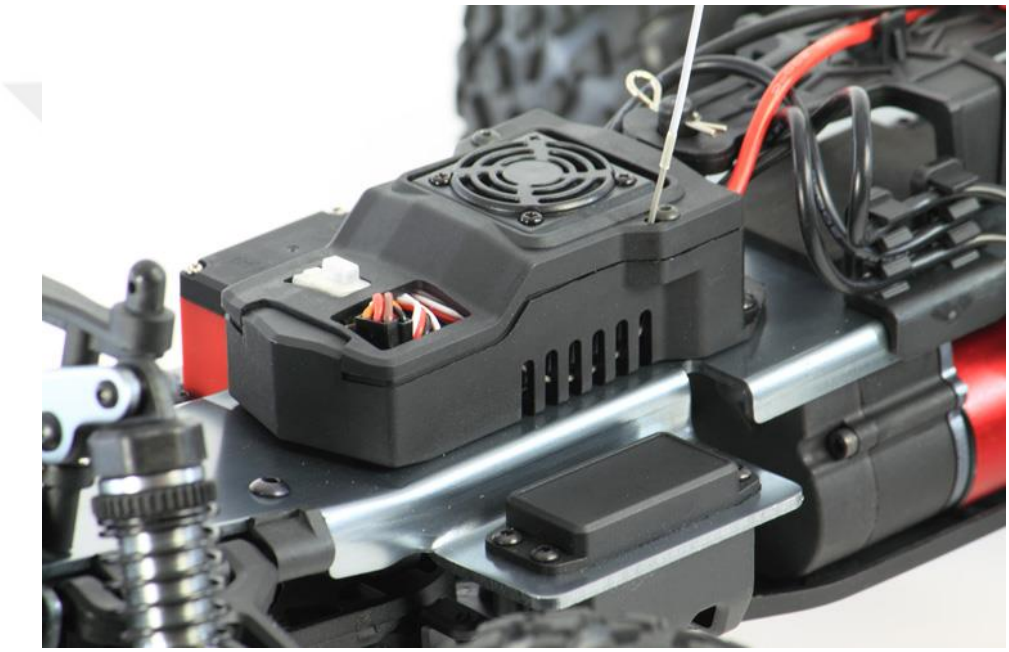
Şekil 3. 1. Fırçalı ve Fırçasız DA Motor İç Yapılarının Karşılaştırılması

Elektromıknatıslar (bobin sargıları), fırçalı motorlarda hareketli iken fırçasız motorlarda sabittir. Fırçasız motorların, fırçalı motorlara göre tercih edilme sebepleri şöyle sıralanabilir;

- Fırçalı motorlara göre daha sessizdirler.
- Fırçalardan kaynaklı kayıplar oluşmadığı için fırçasız motorlara göre daha verimlidir.
- Kullanımdan kaynaklı eskiyecek bir fırça olmadığından daha uzun ömürlüdür.
- Fırçasız motorlara göre daha az radyo frekansı paraziti yayarlar.
- Soğutulmaları daha kolaydır.

- Fırçasız motorlarda, akım, voltaj oranları devir/dakika ile orantılı olarak değerlendirilir, bu sayede istenen tork gücü hesaplanan pervane çapı ile doğru orantılı olarak artar ve azalır.

Fırçasız motorlar kaynağa kesinlikle direkt bağlanmaz! Böyle bir durumla karşılaşıldığında motorların kısa devre yaparak yandığını göreceğiz. Fırçasız elektrik motorları, ESC (Electronic Speed Controller) edilmelidir. Resim 3.5'de araç üzerinde kullanılan 60A kadar akım koruyucu ESC gösterilmektedir.



Resim 3.5. ESC Modülü

ESC sadece elektrikli motor ile çalışan model araçlarda kullanılır. ESC model arabadaki kumanda alıcısına bağlı çalışır. Kumandadan gelen gaz ve fren tepkilerini motora ileterek arabayı hareket ettirir. Gaza veya frene bastığımız zaman motora gidecek olan enerjinin doğru orantılı olarak yansıtılmasını sağlar. ESC sayesinde araç gaza veya frene basış şeklimize göre az veya çok tepki vererek hızlanma, yavaşlama, durma ve geri gitme işlevlerini yerine getirebiliyor.

Aracın ön tekerlikleri arasında konumlandırılmış direksiyon yönünü ve açısını belirleyen bir servo motor kullanılmıştır. Küçük çaplı ve genel olarak içinde kompanzasyon sargısı olan, kuvvetli manyetik alanlı doğru akım motorlarına servo motor olarak ifade edilir. Servo motor imalatı DC motorlar gibi yapılır. 1 devir/dakika hız belgelerinin altında bile çalışan,

moment-hız kontrolü yapabilen yardımcı motorlardır. Robot teknolojisinde en çok kullanılan motor çeşididir. Resim 3. 6'da araç üzerinde kullanılan servo motoru gösterilmiştir.

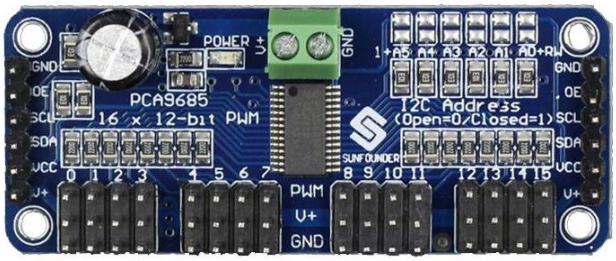
Resim 3.6. Servo Motor Örneği

Bir sistemde servo motor, hız ve pozisyon kontrolünün gerektiği uygulamalarda, geri besleme ile bir karar verme ünitesine gönderilerek sistemin davranışını ayarlar ve son kontrolü yapar. Aracımızda kullanılma amacı da direksiyon kontrolüne hızlı tepkimeler ile yanıt verebilmesidir.

3.1.5. Motor Sürücüsü

RC araç kontrolünü sağlayabilmek için kullanmış olduğumuz RPİ modellerinde PWM (Pulse Width Modulation) giriş pinleri bulunmamaktadır.

RC aracın gaz ve direksiyon kontrollerinin sağlanabilmesi için motorları PWM giriş sürücü kartı ile kontrol etmek gerekmektedir. Tercih ettiğimiz sürücü kartı, I^2C haberleşmesine sahip PCA9685 entegresi bulunan 16 adet PWM çıkışına sahiptir. Böylelikle, RPİ'deki PWM çıkış pininin yetersiz kaldığı projelerimizde bu shield'ı kullanabiliriz. Resim 3.7' de motor sürücü kartı gösterilmiştir.



Resim 3.7. PCA9685 16 Kanal 12 Bit Pwm Sürücü Kartı

I^2C, mikrodenetleyiciler, EEPROM'lar, A / D ve D / A dönüştürücüler, G / Ç arabirimleri ve gömülü sistemlerdeki diğer benzer çevre birimleri gibi düşük hızlı aygıtları bağlamak için kullanılan bir seri protokoldür. Philips tarafından icat edilmiştir. I^2C veri yolu, kullanımı basit olduğundan, birden fazla ana ünitenin bulunabileceği, sadece üst veri yolu hızının tanımlanabileceği ve neredeyse sınırsız sayıda I^2C cihazını bağlamak için iki kabloya ihtiyaç duyulduğu için popülerdir. I^2C, daha yavaş mikro denetleyicileri genel amaçlı I / O pinleriyle bile kullanabilir, çünkü yalnızca bir bayt okuma ve yazma işlevlerine ek olarak doğru Start ve Stop koşullarını üretmeleri germektedir.

Her bir bağımlı aygıtın benzersiz bir adresi vardır. Ana cihazdan ve bağlı cihazdan aktarım seridir ve 8 bitlik paketlere ayrılır. Tüm bu basit gereksinimler, I^2C arabirimini özel I^2C donanım denetleyicisine sahip olmayan ucuz mikrodenetleyicilerle bile gerçekleştirmeyi kolaylaştırmıştır. Komutları göndermek ve almak için sadece 2 adet ücretsiz I / O pini ve birkaç basit I^2C rutinine ihtiyacınız vardır. I^2C sadece iki kablo kullanır: SCL (seri saat) ve SDA (seri veri). Her ikisinin de + Vdd'ye bir dirençle bağlanması gerekir. Farklı voltajlara sahip iki I^2C veri yoluna bağlanmak için kullanılabilecek, I^2C seviye değiştiriciler de bulunmaktadır. Temel I^2C iletişimi 8 bit veya bayt aktarımını kullanıyor. Her bir I^2C bağımlı aygıtının, veri yolu üzerinde benzersiz olması gereken, 7 bitlik bir adresi vardır. Bazı cihazlar I^2C adresini sabitlerken, diğerleri I^2C adresinin alt bitlerini belirleyen birkaç adres hattına sahiptir.

7 bit adres, 7'den 1'e kadar bitleri gösterirken bit 0, cihaza okuma veya yazma sinyalini vermek için kullanılır. Eğer bit 0 (adres baytında) 1 olarak ayarlanırsa ana cihaz slave I^2C cihazından okuyacaktır. Ana cihaz, saat üretmesi (SCL aracılığıyla) ve bireysel I^2C slave cihazlarını adreslediğinden, başka bir adres gerektirmez.

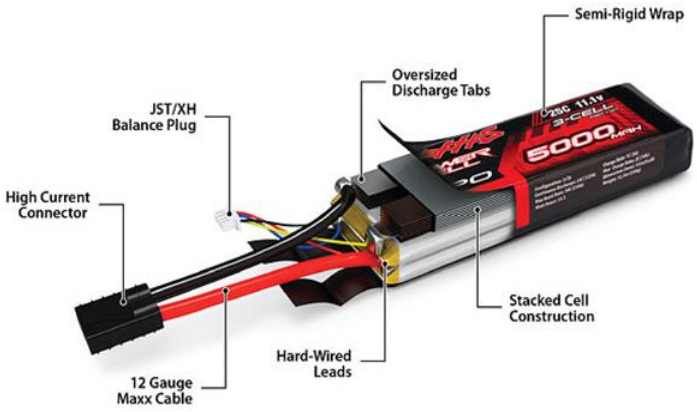
Normal durumda, her iki hat da (SCL ve SDA) yüksektir. İletişim ana cihaz tarafından başlatılır. Çoğu I^2C cihazı tekrarlı başlatma koşulunu destekler. Bu, iletişimin bir durdurma koşulu ile bitmeden önce, ana aygıtın adres baytı ile başlangıç koşulunu tekrarlayabileceği ve moddan yazmayı ve okumayı değiştirebileceği anlamına gelir. I^2C veri yolu birçok entegre devre tarafından kullanılır ve uygulanması basittir. Herhangi bir mikro denetleyici, özel bir I^2C arabirimine sahip olmasa bile I^2C aygıtlarıyla iletişim kurabilir. I^2C özellikleri esnek olabilir, veri yolu yavaş cihazlarla iletişim kurabilir ve ayrıca büyük miktarlarda veri aktarmak için yüksek hız modlarını kullanabilir. Birçok avantajdan dolayı, I^2C veri yolu, entegre devreleri tahtaya bağlamak için en popüler seri arabirimlerden biri olarak kalacaktır.

PCA9685 16 Kanal 12bit Pwm Sürücü Kartı Teknik Özellikler:

- Dahili saat frekansı üretme özelliği sayesinde sürekli güncelleme gerektirmez.
- 6 adet adres değiştirme lehim jumper'ı (62 farklı adres seçimi olanağı)
- 1.6kHz'e kadar ayarlanabilir PWM frekansı
- 12-bit çözünürlük (60Hz güncelleme frekansında ~4µs çözünürlük)
- Push-pull ya da open-drain çıkış seçenekleri
- 3-pin servo uyumlu header bağlantıları, 4'erli grup halinde
- V+ hattı için kapasitör bağlantısı (ihtiyaç duyulduğunda lehimlenebilir)
- 5x20 prototipleme alanı
- 7-bit I2C adres seçimi (0x60-0x80)
- Boyutlar: 2.1" x 2.7" x 0.1" (54mm x 70mm x 3mm)

3.1.6. Bataryalar

Projemizde aracın ihtiyaç duyduğu enerjiyi ve yüksek akımı LİPO pil sayesinde gidermekteyiz. Yapısında Lityum ve Polimer kimyasallarını bulunduran pillere, kısaca LİPO pil denir. LİPO piller hücrelerden oluşur. Her bir hücrenin nominal voltajı 3,7 volttur.
Tek bir hücrenin boş hali 3V dolu hali ise 4,2V olmalıdır. LİPO içerisinde bulunan hücreler S harfi ile ifade edilir. LİPO piller tercih edilirken ihtiyaç duyulan voltaj aralığına göre değerlendirilmeli ve hücre sayısı S harfinden önceki rakam ile kontrol edilmelidir. Bu tür LİPO piller kullanılmadığında, hafıza mantığı olmadığından diğer pillere göre enerji kayıpları daha yavaştır. Resim 3.8'de LİPO pil iç yapısı gösterilmektedir.



Resim 3.8. LİPO pil bileşenleri

Resim 3.9. Tp-link powerbank

LİPO piller kapasiteleri aynı olan diğer pillerle kıyaslandığında, ağırlık ve kullanım zamanı açısından %25-%30 daha fazla verimliliğe sahiptirler. Aynı zamanda bu piller, diğer pillere göre daha fazla anlık amper çıkışına sahiptirler. Bu özellikler göz önüne alındığında fırçasız motorlu kullanımında ihtiyaç duyulan ve tercih edilmesi gereken piller, LİPO pillerdir.

Diğer pillerin deşarj oranı (C değeri) 6 C ile 10 C arasında değişmektedir. Bu oran LİPO pillerde 12 C ve 40 C arasında değişmektedir. NiMH piller (3600 mAh için) 21.6 Amper anlık akım verirken. 3600 mAh 25c LİPO pil 90 A sürekli akım verebilir. Aralarında ki bu yaklaşık 70 amperlik fark LİPO pillerin kullanılması için yeterli bir tercih sebebi yapmaktadır. Genellikle 2S, 3S, 4S ve 6S gibi seri bağlanmış setler halinde yani sırasıyla 2x3.7=7.4 V, 3x3.7=11.1 V, 4x3.7=14.8V, 6x3.7=22.2 V. olarak gruplar halinde bulunur.

Mikrobilgisayar olarak kullandığımız RPİ ‘nin stabil çalışma gerilimi 5V, dalgalanmaların önüne geçebilmek için ise 2A çıkışlı araç üzerinde uzun süre çalışabilecek güç adaptörü için powerbank tercih edilmiştir. Genel olarak taşınabilir şarj cihazı olarak bilinmektedir. USB giriş-çıkışı sayesinde önce şarj edip daha sonra ihtiyaç anında kullanılabilir.

Kullanım alanı prizden uzak olduğumuz yerlerde faydasını görmekteyiz. Kapasitesi büyük pillerdir, mAh değeri ne kadar yüksekse, o kadar yüksek pil ömrü sağlamaktadır. Powerbank’ın mAh değeri yükseldikçe ebatlarıda artmaktadır. RPİ güç adaptörü olarak Tp-link model 6700 mAh ve 2.1 amper çıkış veren powerbank kullanılmıştır. Resim 3.9’da kullanılan powerbank gösterilmektedir.

3.1.7. İletişim

Uygulamamızda RC otonom aracın uzaktan kontrolü ve görüntüleri işlemek için kullanılacak ana bilgisayarla ile RPİ arasındaki haberleşmeyi sağlayacak ekipman olarak mobile modem tercih edilmiştir.

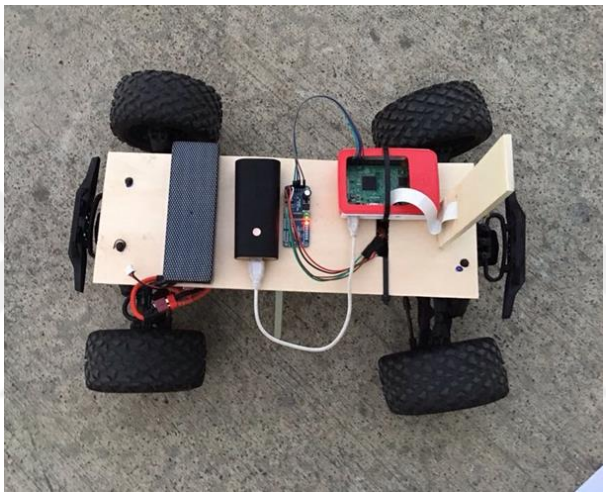
SSH (Secure Shell) bağlantı sayesinde RPİ'ye uzaktan erişim sağlanır. SSH güvenli veri iletimi için kripto grafik ağ protokolüdür. Shell Client ve dosya transferi (FTP) işlemleri için "Secure File Transfer Client" olmak üzere iki uygulamayı içerir. SSH uzaktaki bir makinada komut çalıştırmak için uzaktaki makinada bir kullanıcı oturumu açmayı sağlayan bir uygulamadır. SSH güvenli olmayan bir ağ üzerindeki güvenilir olmayan iki sistemin şifreli dolayısı ile güvenli iletişim kurmalarını sağlar. X11 bağlantıları ve çeşitli TCP/IP bağlantı portları da güvenli kanal üzerinden iletilebilirler. Bu bağlantı sayesinde aracımızın eğitimini yapacağımız yerin konumu fark etmeksizin yanımızda taşıyabileceğimiz, otopilot eğitim esnasında uzaktan kontrolünü sağlayabileceğiz.

RPİ ara yüzüne ulaşarak kod girişi ve ayarları yaparak bir web sunucusu üzerinden anlık görüntüleri izleyerek klavye ya da fare kullanarak aracın hız ve yön kontrolü yapılabilmesini sağlayacaktır. Otopilot eğitimi bittikten sonra alınan görüntü dosyaları bu bağlantı sayesinde ana bilgisayara aktarılacaktır. Resim 3.10'da Huawei e5775 modem gösterilmektedir.

Resim 3.10. Huawei Mobil Modem

3.2. Elektronik Komponentlerin Birleştirilmesi

RC otonom aracın elektronik bileşenleri ve diğer ekipmanlarının konumlandırılması, montajı ve birleştirilmesini bu bölümde inceleyeceğiz. Öncelikle RC araç şasisi üzerinde bulunacak mikrobilgisayar, adaptör, PWM sürücüsü, Pi kamera ve uzun süre çalışabilmesi için yedek LİPO pilin üzerinde bulunacağı malzeme olarak, model uçak teknolojisinde kullanılan balsa levha tercih edilmiştir. Balsa levha bir ahşap türüdür ve diğer ahşaplardan ayrılan en önemli özelliği hafif, dayanıklı ve kolay şekillenebilir olmasıdır. Resim 3.11 ve Resim 3.12'de RC otonom aracın üst ve yan görünüşleri gösterilmektedir.

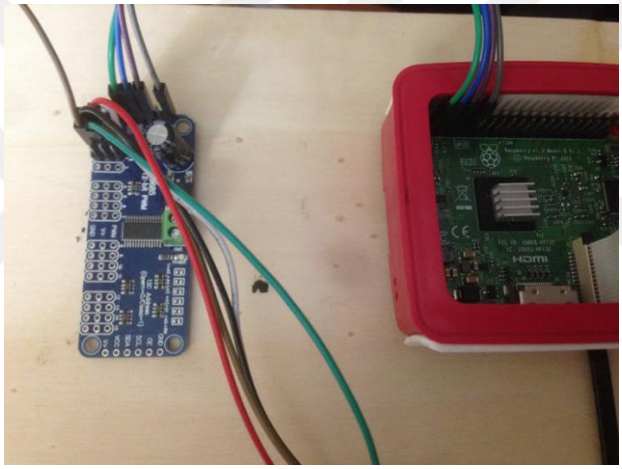


Resim 3.11. RC Otonom Araç Üst Görünüş

RC otonom araç üzerinde bulunan ekipmanlar çift taraflı bant ve klipsler sayesinde, Pi kamera ise açısı ayarlanarak vidalar sayesinde dikey bir konumda yerleştirilmiştir.

Resim 3.12. RC Otonom Araç Yan Görünüş

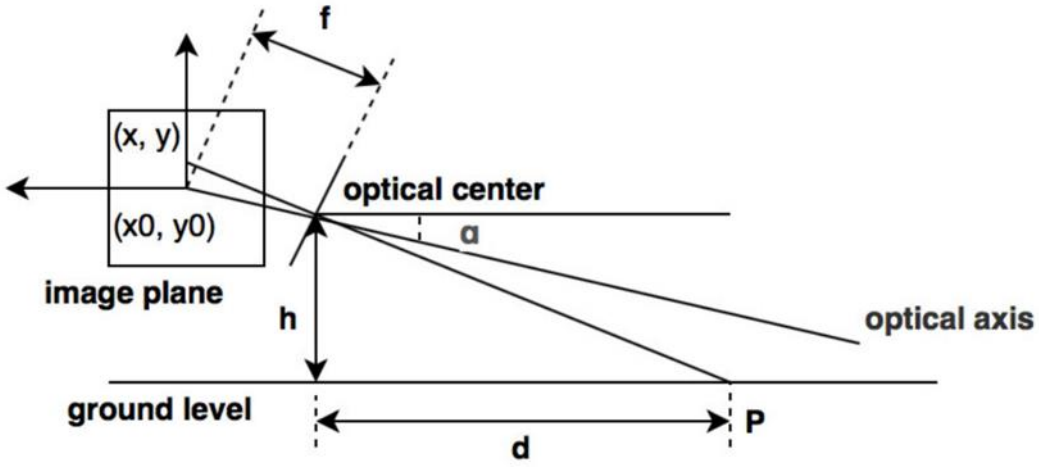
RC otonom aracın tasarımına başlarken ilk önce araç üzerinde sabit bulunan kumanda alıcısına bağlı, fırçasız motorun hız kontrolünü ve direksiyon açısını sağlayan servo motora bağlı pinleri çıkarılmıştır. 1/10 ölçekli aracımızın üst plakası için uygun ebatlarda balsa levha kesilmiş ve şasi üzerine sabitlenecek şekilde bağlantı yerlerinden uygun delikler açılmış, balsa levha şasi üzerine ataç ve benzeri tel vasıtasıyla sıkıştırılmıştır. Ana bilgisayar ile SSH bağlantı protokolünü kullanarak, uzaktan bağlantı ile hız ve konum değişiklerinin yapılabilmesi için PWM sürücü kartı ile RPİ mikrobilgisayar üzerinde bulunan I^2C pinleri dişi-dişi jumperlar ile bağlanması gerekmektedir. Bu işlemde PWM motor sürücüsünün SCL, SDA, Vcc ve GND isimli bacakları RPİ üzerindeki 1,3,5 ve 9 nolu pinlere bağlanarak iletişim sağlanacaktır. Örnek bağlantı Resim 3.13'de gösterilmiştir.



Resim 3.13. RPİ ile PWM Sürücü Bağlantısı

PWM sürücü 3.3 volt ile çalışmaktadır, bu yüzden sürücü üzerindeki Vcc besleme pinini RPİ üzerinde bulunan 3.3 Volt çıkış veren 1 numaralı pine bağlanır. Aynı zamanda GND isimli pini RPİ'nin 9 numaralı kısmına bağlayarak topraklama işlemi gerçekleştirilir. PWM sürücü üzerindeki SCL ve SDA pinleri ise, RPİ üzerindeki I^2C bağlantı pinleri olan 3 ve 5 numaralı pinlere bağlanarak iletişim sağlanmış olur. PWM sürücü kartı üzerinde bulunan kanal 1 ve kanal 2 pinlerinden jumperlar aracılığıyla fırçasız motor ESC' si ve servo motor bağlantısı yapılır. Bu bağlantı tamamlandıktan sonra, ana bilgisayar ve mobil telefon ile araca sinyal yollanılabilir hale gelerek gerekli hız ve yön kontrolleri sağlanmış olacaktır.

Uygulamamızın önemli bir parçası olan Pi kamera modülünün konumlandırılması aşamasında Chu, Ji, Guo, Li ve Wang (2004) tarafından önerilen optimum görüş yöntemini kullanarak bir nesneye olan mesafeyi tespit eden bir geometri modelinden esinlenilmiştir.



Şekil 3.2. Optimum Kamera Açısı Geometri Modeli [37]

Şekil 3.2'de P, hedef nesne üzerinde bir noktadır; d, optik merkezden P noktasına olan mesafedir. Yukarıdaki geometri ilişkisine dayanarak, formül (2), mesafeyi nasıl hesaplayacağımızı gösterir, d Formül (2) 'de; f kameranın odak uzaklığıdır, α kamera eğim açısıdır, h optik merkez yüksekliği, (x0, y0) görüntü düzlemi ve optik eksenin kesişme noktasını ifade eder, (x, y) görüntü düzleminde P noktasının projeksiyonunu ifade eder. O1'i (u0, v0), optik eksen ve görüntü düzleminin kesişme noktasının kamera koordinatı olduğunu varsayalım, aynı zamanda görüntü düzleminde x eksenine ve y eksenine karşılık gelen bir pikselin fiziksel boyutunun dx ve dy olduğunu varsayalım. Sonra;

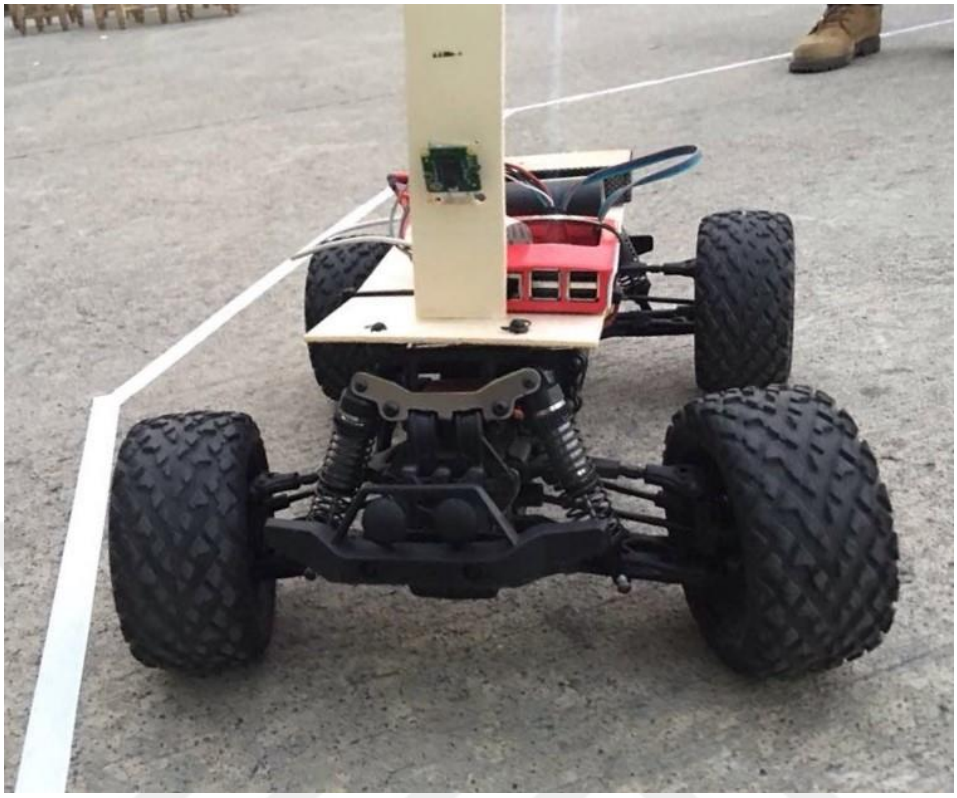
$$d = h/\tan(\alpha + \arctan((y - y_o)/f)) \quad (3.1)$$

$$u = \frac{x}{dx} + u_0 \quad v = \frac{y}{dy} + v_0 \quad (3.2)$$

$x_0 = y_0 = 0$, (3.1) ve (3.2) numaralı eşitlikler yardımıyla uzaklığı gösteren yeni denklem aşağıdaki şekilde yeniden düzenlenebilir:

$$d = h/\tan(\alpha + \arctan((v - v_o)/a_y)), \ \ (a_y = f/dy) \quad (3.3)$$

v, y eksenindeki kamera koordinatlarıdır ve nesne algılama işleminden döndürülebilir. Diğer tüm parametreler, kamera matrisinden alınabilen gerçek parametreleridir [37]. Pi kameranın açısını bizde bu modelden esinlenerek konumlandırdık. Şekil 3.12' de aracın ön görünüşü gösterilmektedir.



Resim 3.14. RC Otonom Aracın Ön Görünüşü

4. RC OTONOM ARACIN YAZILIM TASARIMI

Tez çalışmasının bu bölümünde, otonom aracın manevra ve hareket kabiliyetlerini sağlamak, anlık görüntüleri kayıt altına alarak görüntü veri dosyalarının oluşturulması, bu dosyaların ana bilgisayar üzerinde işlenerek otopilot eğitiminin gerçekleştirilebilmesi için gerekli yazılımlar, kütüphaneler ve kalibrasyon ayarları incelenmiştir. Detaylı kodlar Ekler bölümünde sunulmuştur. Kullandığımız mikrobilgisayar türünün RPİ olmasından ötürü açık kaynak kodlu Python programlama dili kullanılmıştır. Görüntü işleme teknikleri içerisinde kullanılan MatLAB, Visual Studio, Opencv gibi birçok programlama dilleri ve kütüphaneleri mevcuttur. Otopilot eğitimi aşamasında Python programlama dili ile uyumlu çalışan ve Google mühendisleri tarafından geliştirilmiş TensorFlow, Keras, Tornado ve birçok temel kütüphaneler ile birlikte kullanılmıştır. TensorFlow ve Keras kullanılarak bir sinir ağı sistemi beslenir. Sistemi, birkaç kez parkur üzerinde manuel olarak kullanarak ve sonra kendi kendine kullanmasına izin vererek sistemin eğitim aşaması incelenmiştir. Otonom aracımızın farklı yol ve şartlarda eğitilebilmesi için, Unity oyun platformu üzerine inşa edilmiş simülasyon programı üzerinde denemeler yapılmıştır.

4.1. Python Yazılımı ve Kütüphaneler

Python, Guido Van Rossum adlı Hollandalı bir programcı tarafından yazılmış programlama dilidir. Geliştirilmesine 1990 yılında başlanan Python; C ve C++ gibi programlama dillerine kıyaslarsak şöyle sonuçlar elde edebiliriz.

- Daha kolay öğrenilir.
- Program geliştirme sürecini kısaltır daha hızlı yazılabilmesine imkân sağlar.
- C ve C++ programlama dillerinin aksine ayrı bir derleyiciye ihtiyaç duyulmaz.
- Daha okunaklı, daha düzenli kod ve söz dizilimine sahiptir.

Python işlemleri hızlı, sisteminize entegre ve etki bir şekilde yapmanızı sağlayan bir programlama dilidir [38]. Windows, Linux / Unix ve MacOS işletim sistemleri üzerinde çalışmasının yanı sıra Java ne .NET sanal makinelerine de port edilmiş durumdadır. Python açık kaynak kodlu, özgür lisanlı ve ücretsiz bir yazılım olması sebebiyle çok fazla geliştirici tarafından tercih edilmiştir. Nesne yönelimli programlama, fonksiyonel ya da yapısal programlama gibi birden fazla programlama paradigmasını desteklemektedir.

Python dilini öne çıkaran unsurlardan bir diğeri ise bilimsel araştırmalarda tercih edilmesi ve hızlı bir yapısına sahip olmasıdır. Bir başka yönden incelediğimizde RPİ ve Arduino benzeri programlanabilir elektronik kartlar ile uyumlu bir alt yapıya sahip olmasıdır.

Bilimsel araştırmalarda çok yaygın olarak kullanılan Python için, çeşitli bilim dallarında kullanılmak üzere hazırlanmış özel kütüphaneler bulunmaktadır. Python kütüphanesi deyince neyi kastettiğimizi açıklamak gerekirse, önceden yazılmış ve çok kullanılan program parçacıkları daha sonra tekrar tekrar kullanılmak üzere arşivlenerek dosyalarda saklanır. Özellikle herkes tarafından kullanılan dosyalar, bu programlama dilini bilgisayarımıza kurduğunuzda kurulum dosyaları ile birlikte bilgisayarımıza yüklenir. Gerek duyduğumuzda kod penceremizden gerekli yol gösterme tanımlamaları ve çağırma komutları girerek, bu kod parçacıklarını kullanabiliriz, bu sayede işlemlerimiz daha hızlı gerçekleşecektir, bu kütüphaneler profesyonel yazılım ekipleri tarafından hazırlanmaktadır, yoğun şekilde test edilmekte ve yine açık kaynaklı olması sebebiyle üzerinde değişiklikler yapılarak amacımıza uygun hale getirilebilir ve geliştirilebilmektedir. Uygulamamızda kullanılan Python versiyonu ve en çok tercih edilen kütüphaneler hakkında bu bölümde bilgi verilecektir

Anaconda (Python);
Anaconda; Python, R programlama dili, veri bilimi ve bilimsel araştırmalar ile uğraşan insanlar için Anaconda kuruluşu tarafından geliştirilmiş bir ön yükleyicidir. Geliştirilen bu on yükleyici kullanıcıların işlerini büyük ölçüde kolaylaştırmaktadır. Yüklendiği zaman kendisiyle beraber veri işleme ve raporlama dahil olmak üzere bu işlemlerle ilgili 100'den fazla kütüphaneye otomatik olarak erişilebilmektedir. Aynı zamanda istenilen 600'den fazla kütüphane conda yükleyicisi sayesinde rahatlıkla kurulabilmektedir.

Altı milyonu aşkın kullanıcı ile açık kaynak kodlu Anaconda dağıtım, Python veri bilimi ve makine öğrenimini yapmanın en kolay yoludur. Windows, Linux ve MacOS için 250'den fazla popüler veri bilimi kütüphanesi ve sanal ortam yöneticisi içerir. Conda, Scikit-learn, TensorFlow ve SciPy gibi karmaşık veri bilimi ve makine öğrenim ortamlarını kurmayı, çalıştırmayı ve yükseltmeyi hızlı ve kolay hale getirir [39]. Python programlama dilinde ihtiyaç duyduğumuz kütüphaneleri tek tek yüklemek zaman kaybına yol açacağından, Anaconda (Python) önyükleyicisini ana bilgisayarımız da kullanmayı tercih etmekteyiz.

TensorFlow;

TensorFlow, geliştiricilerin derin öğrenim modellerini tasarlamalarını, üretmelerini ve eğitmelerini kolaylaştırmak için Google tarafından 2015 yılında yayınlanan açık kaynaklı bir yazılım kütüphanesidir. TensorFlow, Google geliştiricilerinin kurum içi modelleri oluşturmak için kullandıkları bir dahili kütüphane olarak ortaya çıktı ve açık kaynak sürümüne sahiptir. TensorFlow, geliştiricilerin kullanabileceği çeşitli seçeneklerden sadece biri olsa da düşünceli tasarımı ve kullanım kolaylığı nedeniyle tercih edilmektedir.

TensorFlow, yüksek düzeyde, kullanıcıların veri akışlarının grafiği olarak keyfi hesaplamayı ifade etmelerini sağlayan bir Python kütüphanesidir. Bu grafikteki düğümler matematiksel işlemleri temsil ederken, kenarlar bir düğümden diğerine iletilen verileri temsil eder. TensorFlow'daki veriler çok boyutlu diziler olan tensörler olarak temsil edilir. Hesaplamaya yönelik bu çerçeve birçok farklı alanda değerli olsa da TensorFlow temel olarak uygulamada ve araştırmada derin bir öğrenme için kullanılır. TensorFlow ile geliştirme yapabilmek için Python sürümümüzün 3.5.x veya üzerinde olması gerekiyor. Kurulumu ise pip yükleme komutu ile gerçekleşmektedir. Bu kütüphanelerin GPU üzerinde de çalışabildiği bilinmektedir, eğer NVIDIA markalı bir GPU kullanılıyorsa, GPU versiyonu kurulmalıdır. Yükleme komutu aşağıda gösterilmektedir.

CPU Sürümü için;
pip3 install --upgrade tensorflow

GPU Sürümü;
pip3 install --upgrade tensorflow-gpu

TensorFlow'u doğrudan Linux, MACOS ve Windows üzerine kurulabilmektedir. TensorFlow'un en temel özelliği geniş veri kümelerinde YSA'nın hızlı bir şekilde aktarımını sağlayan çok katmanlı düğüm sistemidir. Bu sayede Google'ın ses ve obje tanıma gibi birçok özelliği güçlenmiş durumdadır. Alternatifleri Caffe, Torch, Keras, Theano, deeplearning4j vb. kütüphanelerdir. Çizelge 4.1'de bu kütüphanelerden bazıları hakkında kısa bilgiler verilmiştir.

Çizelge 4.1. Derin Öğrenme Kütüphaneleri [40]

Kütüphane	Yazıldığı dil	Geliştirici	Öne çıkan özellikleri
Theano [41]-[42]	Python	MILA Lab	- Öğreticileri (tutorial) çok etkili. - Keras, Blocks gibi API sayesinde matematiksel hesaplar kolaylaştırması. - GPU desteği.
Caffe [43]	Python	Berkeley Vision and Learning Center (BVLC)	- Caffe Model Zoo üzerinden indirilebilecek ve hemen kullanılacak önceden eğitimli ağların bulunması. - GPU desteği.
Torch [44]	Lua	Ronan Collobert, Clement Farabet, …	- Algoritmaları oluşturma konusunda maksimum esnekliğe ve hıza sahip olması. - GPU desteği. (CUDA) - Kullanıcı dostu arayüz
Digits [45]	C++	NVIDIA	- Çoklu GPU sistemleri üzerinde sinir ağları tasarımı ve eğitimi, - Gelişmiş görselleştirmelerle performansı gerçek zamanlı olarak izleme - Tamamen etkileşimli
TensorFlow [46]	Python	Google	- Tek bir API ile bir masaüstü, sunucu veya mobil cihazdaki bir veya daha fazla CPU'ya veya GPU'ya dağıtma olanağı.
DeepLearning [47]	Java	Adam Gibson	- JVM tabanlı
KNET [48]	Julia	Deniz Yuret	- Kolay anlaşılır, kısa kodlama yeteneği. - İfade gücü. - GPU Desteği

İncelediğimiz bu kütüphaneler arasında Knet, Theano, Torch, Caffe ve TensorFlow için farklı veri kümeleri ile modeller üzerinden, tek GPU ile çalışan bilgisayarlar için zaman performans karşılaştırması Çizelge 4.2'de verilmiştir [49].

Çizelge 4.2. Kütüphanelerin Çalışma Zamanı Performansının Karşılaştırılması [40]

Model	Veri Kümesi	Knet	Theano	Torch	Caffe	TensorFlow
LinReg	Housing	2.84	1.88	2.66	2.35	5.92
Softmax	MNIST	2.35	1.40	2.88	2.45	5.57
MLP	MNIST	3.68	2.31	4.03	3.69	6.94
LeNet	MNIST	3.59	3.03	1.69	3.54	8.77
CharLM	Hiawatha	2.25	2.42	2.23	1.43	2.86

Çizelge 4.2 incelendiğinde Theano kütüphanesinin diğer kütüphanelere göre daha hızlı, TensorFlow ise daha yavaş çalışan kütüphane olarak görülmektedir. TensorFlow'un diğer kütüphanelere göre daha yakın tarihte ortaya çıkmış olması ve geliştirilme aşamasında olduğu göz önünde bulundurulmalıdır.

Keras;

Keras üst düzey bir sinir ağları API kütüphanesidir, Python ile yazılmış ve TensorFlow , CNTK veya Theano gibi kütüphanelerle birlikte çalışabilmektedir. Hızlı deneylerin yapılabilmesine odaklanarak geliştirilmiştir [50]. Python ile yazılmış, yüksek düzeyde yapay sinir ağı inşa edebilmek için geliştirilmiş bir ara yüzdür. Kolay kullanımlı ve basittir. Yüksek düzeyde genişletilebilirliğe sahiptir. Tüm fikir, katmanlara dayanır ve geriye kalan her şey bu katmanlar arasında inşa edilir. Veriler tensörlerle hazırlanır, ilk katman veri girişini, son katman veri çıkışını sağlar ve oluşturulacak model bu katmanların arasına yerleştirilir. Arka planda Theano veya TensorFlow'u kullanır ama son dönemlerde Microsoft kendi kütüphanesi olan CNTK (Cognitive Toolkit) yapısı ile Keras kütüphanesini tümleştirmek için çalışmalar yürütmektedir.

Theano;

Theano, Python için çok boyutlu serilerle çalışmak için, ancak daha fazla matematiksel yapı ile inşa edilmiş ve ek olarak makine öğrenmesi ve derin öğrenme kütüphanesi dahilinde kullanılmak üzere Google desteği ile Montreal Üniversitesi'ndeki bir grup tarafından geliştirilmeye başlanmıştır. Theano düşük seviyeli işlemlerde tamamen NumPy ile çalışmaktadır. Kütüphane ayrıca hassas verileri işlerken CPU ve GPU optimizasyonu için de imkân sunmaktadır. Verimlilik ve stabilizasyon için ufak ayarlamalar yapıldığında kütüphanenin sunacağı hesaplama sonuçları gözle görülebilir derece keskin sonuçlar verebilmektedir. Kütüphaneyi geliştiren ekibin farklı işlere yönelmesinden dolayı sadece açık kaynak topluluğu tarafından gelişimi devam etmektedir.

Tornado;

Frendfeed ekibinin geliştirdiği ve açık kaynak kodlu olan Tornado; aynı anda binlerce bağlantının sağlanması ve bu sayede gerçek zamanlı web hizmetlerinin ideal hale getirilmesi için tasarlanmıştır. Python için hazırlanmış yüksek performanslı web sunucusu ve asenkron kütüphanesidir. Bu kütüphaneyi kullanma amacımız otonom aracın web sunucusu üzerinden anlık görüntü aktarımını sağlayabilmesi ve otopilot eğitimi aşamasında manevre ve hız kontrolünü bize sunuyor olmasıdır.

Requests;

Bu kütüphane ile web üzerindeki isteklerimizi yöneteceğiz, bu modül ile en temel şekilde http/https protokollerine yönelik request/response işlemlerimizi gerçekleştirebiliriz.

NumPy (Numeric-Python);

Python ile bilimsel bazlı hesaplama yapabilme yeteneği olan bir kütüphanedir. NumPy sayesinde Python dili ile diziler ve matrislerde işlemler kolayca yapılabilmektedir.

Ayrıca kütüphanede özellik olarak N-Boyutlu dizi nesnelerinde işlem yapabilme, kütüphanenin kendisine özgü olarak özelleşmiş fonksiyonları barındırma (Fourier dönüşümleri, lineer cebir işlemleri, düzgün biçimli rastgele sayı dağılımları, düzgün biçimli olmayan sayı dağılımları vb.), C/C++ ve Fortran kodları ile tümleşik çalışabilme gibi yetenekler bulunmaktadır.

PIL (Pillow) - Python Image Library;

Python'da fotoğraf dosyalarıyla oynamak için en çok tercih edilen modüldür. Kullanımı fazlasıyla basit, dokümantasyonu diğerlerine göre daha anlaşılabilir ve topluluk olarak daha büyük bir topluluğa sahiptir.

Docopt;

Python'da komut satırı ara yüzlerini kolayca oluşturmanıza yardımcı olur. Program içerisinde yazdığımız kodlar için kendimize özel yardım mesaj eklememizi sağlar. Bu şekilde, tekrarlanabilir kod çözümlemesini yazmak yerine yalnızca yardım mesajını yazabiliriz.

H5py;

Büyük miktarda sayısal veri depolamamıza ve bu verileri NumPy 'den kolayca değiştirmemize izin verir. Binlerce veri kümesi tek bir dosyada saklanabilir, kategorize edilebilir ve istediğiniz gibi etiketlenebilir.

Flask;

Flask; Python programlama dili ile yazılmış, web uygulamaları yazmak için kullanılan bir mini frameworktür. Yani çekirdeği basit ve geliştirilebilirdir. RC Otonom aracımızı web sunucu üzerinden kullanırken, karşımıza gelen ekrandaki görsellerin hazırlandığı modüldür.

RPIO;

RPİ' de bulunan giriş çıkış pinlerini (GPIO) kontrol edilebilme yeteneğini ekler. Bu kütüphane ağırlıklı olarak motorlar için PWM kontrolü sağlar.

Pandas;

Veri analizi ve veri ön işlemeyi kolaylaştıran açık kaynak kodlu bir kütüphanedir. Dil olarak Python kullanır. Pandas dağıtık işlemeye uygun değildir. Pandas, veri yapıları için çok uygun bir kütüphanedir. En çok kullanılan nesnesi Data Frame'dir. NumPy'daki veri yapısı ile Excel ve SQL gibi ilişkisel veri yapılarını işleyebilir, index oluşturabilir.

PiCamera:

CSI aracılığıyla PiCamera üzerinde kontrol sağlar. Bu kütüphane sayesinde RPİ üzerinde kullandığımız kameranın ayarlarına erişebilme ve değiştirebilme olanağı sağlar.

PyGame:

RPİ kullanıcılarının bağlı olduğu bilgisayarındaki klavyeye erişimi için kullanılmaktadır, otonom aracın parkur üzerinde eğitimi aşamasında, klavye üzerinden manuel kullanımına olanak sağlayacaktır.

Adafruit_pca9685;

Bu modül RPİ üzerinden adafruit pca9685 pwm elektronik kartımızın servo kontrollerini I^2C haberleşme protokolü ile gerçekleştirmemize olanak tanır.

4.2. Yazılım Kurulumu

Tez uygulamamızın bu bölümünde, üçüncü bölümde anlatmış olduğumuz elektronik komponentlerin birleşiminden sonra RPİ 'de kullanacağımız işletim sisteminin yüklenmesi, gerekli ayarlamalar, ana bilgisayar ile RPİ arasında kurulacak SSH bağlantı şekli, ana bilgisayarda kullanacağımız ve görüntü işlemede yardımcı olacak gerekli programların kurulumu anlatılacaktır.

4.2.1. RPİ Kurulum

RPİ, mikrobilgisayar olması sebebi ile kendi alt yapısına ait bir işletim sistemi kurulumuna ihtiyaç duymaktadır. RPİ'nin ilk kurulumunda ön yüklemeli, minimum 4gb kapasiteye sahip Micro SD (Secure Digital) karta ihtiyaç duyulur. RPİ Vakfının resmî sitesi olan "https://www.raspberrypi.org/downloads/raspbian/" sayfasından ihtiyaç duyulan Raspbian İşletimi Sistemi, (Resim 4. 1) ana bilgisayarımıza indirilir.

Raspbian Stretch ve Raspbian Stretch Lite olmak üzere iki farklı dosya karşımıza gelecektir. Lite olan sürümde grafik ekran ara yüzü olmadığı için RPİ 3'ün HDMI çıkışından görüntü alamıyoruz. Bu sürüm ileri seviye kullanıcıların RPİ'yi komut satırından (terminalden) kullanmaları için hazırlanmıştır. Projemizde ekran görüntüsüne ihtiyaç duyulmadığından ve işlemlerin daha hızlı yapılabilmesi için Lite sürümünü tercih edilmiştir.

Resim 4.1. Raspbian İşletim Sistemi Çeşitleri

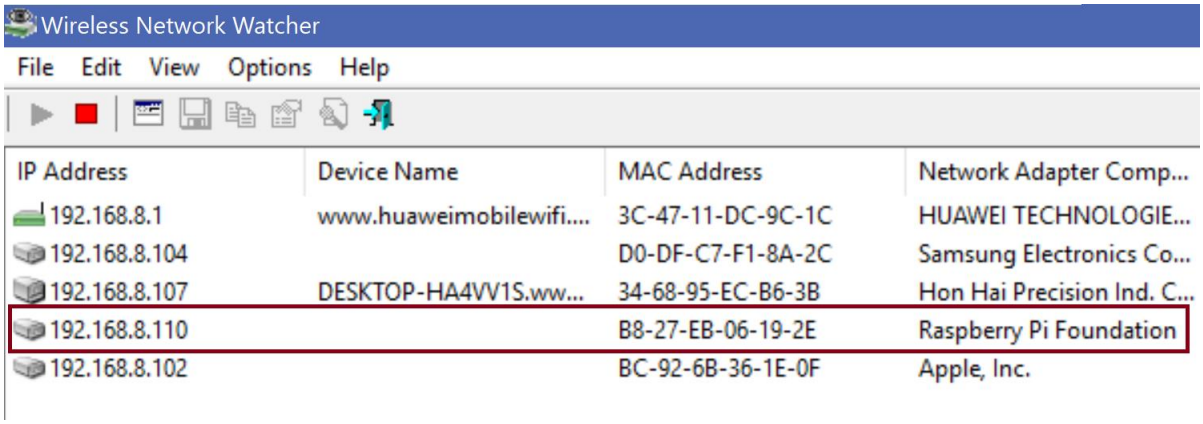
İndirdiğimiz Lite imaj dosyasını .zip formatı içerisinden çıkarıyoruz. SD karta işletim sistemi yazdırmamız için daha önce ana bilgisayarımız indirdiğimiz win32diskImager programını açıyoruz. İmaj dosyamızı belirtilen yerden seçiyoruz. SD kartımızın bilgisayara takılı olduğundan emin olduktan sonra Device kısmında görebiliriz. Ardından Write butonuna tıklayıp yazma işlemini başlatıyoruz. Yazma işlemi yaklaşık 2-3 dk. sürmektedir. Yazma işleminin bitmesini, yeni açılan pencerede "Write Succesful." yazısını görene kadar bekliyoruz.

RPİ'nin ilk açılışta wi-fi' ye giriş yapabilmesi için kullanılacak özel bir dosya oluşturabiliriz. Windows'ta, SD kart takılı halde iken, disk üzerinde iki bölüm olan iki sürücü bulunmaktadır. İlki etiketli önyükleme yapılabilmesini sağlayan, ortak FAT türü ile biçimlendirilmiştir. Bu sürücü ilk açılışta wi-fi'mizi bulup açmamıza yardımcı olacak bazı dosyaları düzenleyeceğimiz yerdir. Windows'ta bir metin editörü olan Not Defteri'ni açarak aşağıda paylaştığımız kodları, wi-fi'mize uygun şekilde yapıştırıp düzenlemeliyiz.

```
country=TR
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
   ssid="<your network name>"
   psk="<your password>"
}
```

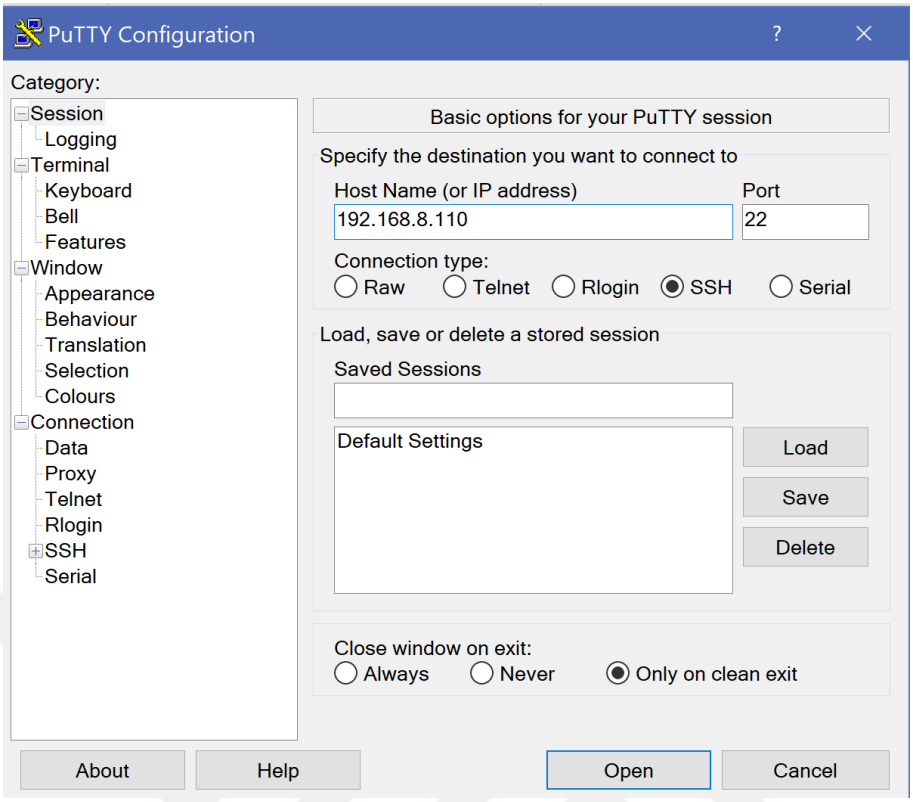
<your network name> Bağlanacağımız ağın kimliği bu kısma yazılmalıdır. Projemizde kullandığımız, Huawei mobil modemin kimliği “HW-4G-MobileWİFİ-9C1C” ifadesi bu alana yazılmıştır. <your password> Bu alan ise modemimize ait şifre ile değiştirilmelidir. Değişiklikler yapıldıktan sonra dosya adı “wpa_supplicant.conf.” adı ile değiştirilerek kaydedilmelidir. Bu dosya, SD kart içerinde “/etc/wpa_supplicant/wpa_supplicant.conf” klasörü içerisine taşınmalıdır.

Yukarıda bahsetmiş olduğumuz talimatları uyguladıktan sonra, RPİ artık wi-fi ağımıza bağlanacak duruma gelmiştir, SD kartımızı ana bilgisayardan çıkartarak, RPİ’mizin mikro SD giriş kısmına takılarak işlemlerimize başlayabiliriz. RPİ’mizin çalışabilmesi için, USB bağlantı kablosu ile powerbank adaptörü bağlanarak çalıştırılmıştır. Artık RPİ’nin IP adresini bulmanız gerekiyor, böylece SSH ile bağlantı sağlanacaktır. Bu bağlantıların yapılabilmesi için Windows üzerinde çeşitli programların çalıştırılması gerekmektedir. İlk olarak IP adresinin bulunabilmesi için ana bilgisayar ile RPİ’nin aynı modem üzerinde bağlantı kurması gerekmektedir. Ana bilgisayar üzerinde “Wireless Network Watcher” programı çalıştırılarak RPİ’nin IP kimliği öğrenilir. Resim 4.2’de aynı modeme bağlı farklı ağların IP kimlikleri gösterilmiştir.



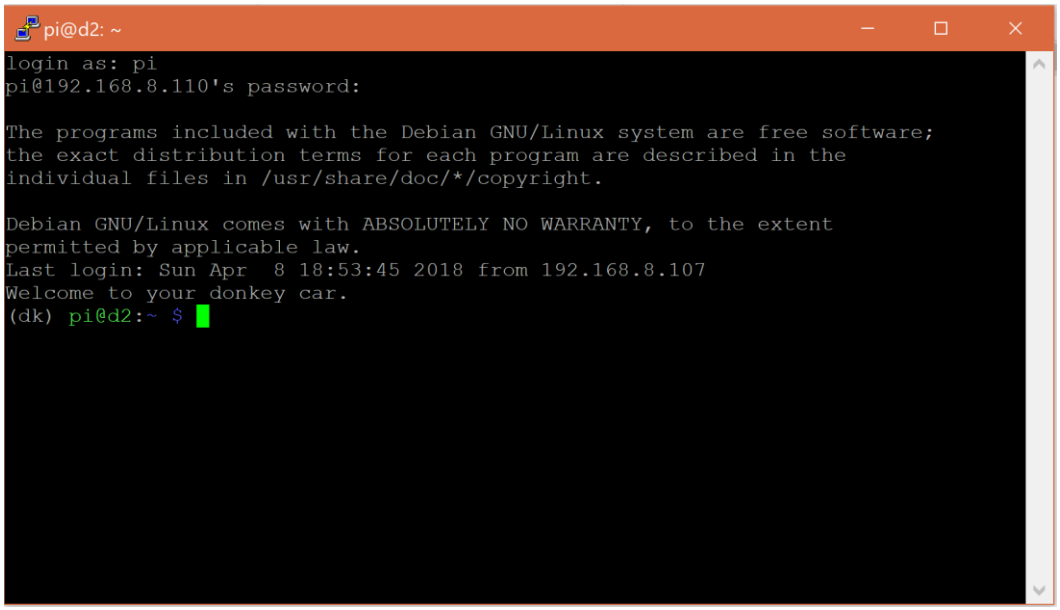
Resim 4.2. Wireless Network Watcher Giriş Sayfası

Program çalıştırıldıktan sonra RPİ’ ye ait IP adresi “192.168.8.110” gösterilmektedir. Bu adres farklı giriş zamanlarında değişiklik gösterecektir. IP adresi öğrenildikten sonra RPİ ile SSH bağlantı sağlanabilmesi için Putty programı tercih edilmiştir.Windows üzerinde Putty programı çalıştırıldıktan sonra Resim 4.3’te görüldüğü gibi açılış ekranı bizi karşılayacaktır.



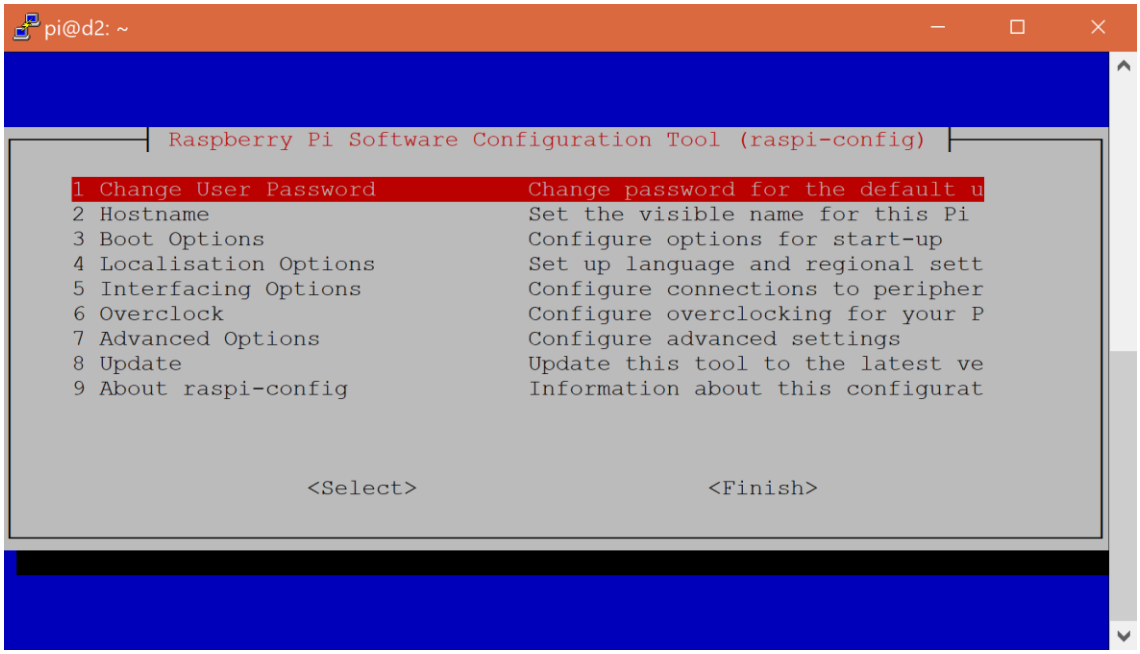
Resim 4.3. Putty Giriş Sayfası

Putty giriş ekranı karşımıza geldikten sonra "Host Name" bölüme RPİ'nin IP adresi girilmeli ve sonrasında bağlantı türü olarakta SSH alanı seçili olmalıdır. "Open" tuşuna tıklayarak, RPİ'nin komut terminaline erişim sağlanacaktır. Şekil 4.4'te giriş komut ekranı görülmektedir.



Resim 4.4. RPİ Komut Ekranı

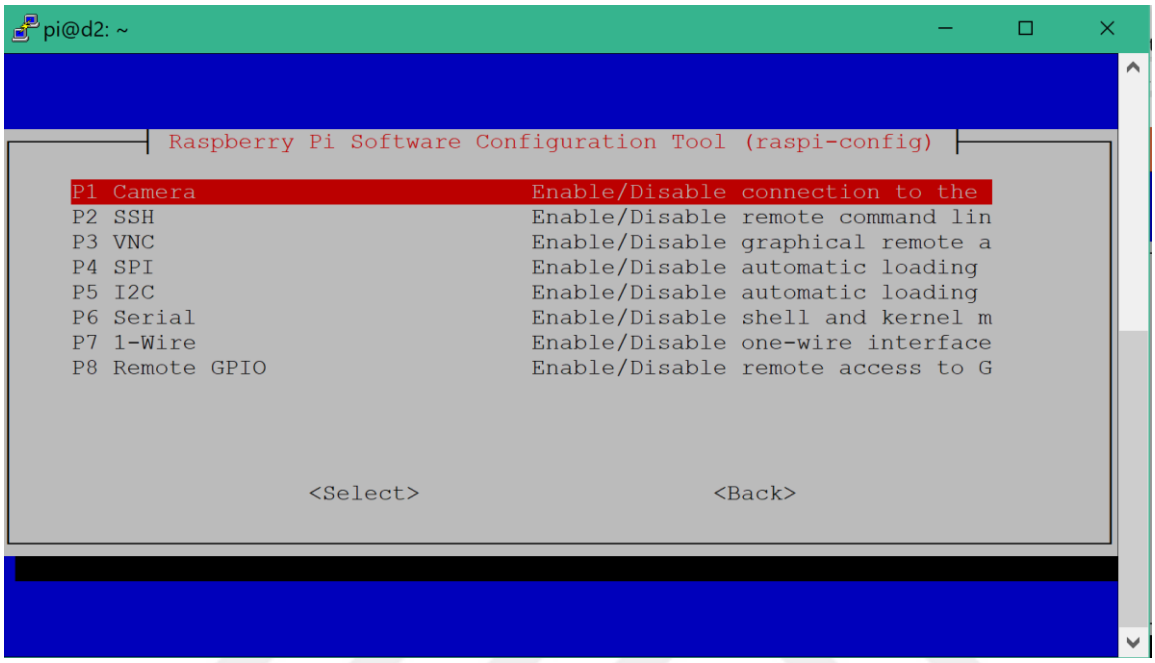
RPİ kişisel bir mini bilgisayar olduğu için, ilk giriş ekranında bizden “login as:” kullanıcı adı, “password:” şifre isteyecektir. Raspbian işletim sistemleri ilk kurulumda hazır “login as: pi” ve “password: raspberry” bilgileriyle gelmektedir. RPİ’ ye erişim sağlandıktan sonra yapılacak ilk iş şifre kısmının değiştirilmesi ve ayarlar kısmına giderek projemizde kullandığımız GPIO giriş çıkış pinlerini, I^2C bağlantı girişini ve Pi kamera seçeneğini aktif etmek olacaktır. Komut satırından “sudo raspi-config” ifadesi girilerek ayarlar giriş ekranına erişim sağlanacaktır. Şekil 4.5’de RPİ ayarlar giriş ekranı gösterilmektedir.



Resim 4. 5. RPİ Ayarlar Giriş Ekranı

- **Change User Password:** RPİ’nin kullanıcı adı ve şifresinin değiştirildiği bölümdür.
- **Hostname:** RPİ’nin ağ taramasında bulunacağı ismin değiştirildiği bölümdür.
- **Boot Options:** Raspbian işletim sistemine sahip RPİ’nin farklı açılış modlarının seçilebildiği bölümdür.
- **Localisation Options:** RPİ’nin dil ve bölge yerleşimi değişikliklerinin ayarlandığı bölümdür.
- **Interfacing Options:** RPİ’ ye dışarıdan bağlanacak çevre donanımlarının aktif veya pasif yapılandırmasını sağlayan ayardır.
- **Overclock:** RPİ’nin çalışma performansını artırmak için kullanılan ayardır.
- **Advanced Options:** Gelişmiş seçeneklerin yapılandırıldığı ayarlardır.
- **Uptade:** RPİ’yi güncelleştirmemizi sağlayan ayarlardır.
- **About raspi-config:** Kullandığımız yapılandırma sistemi hakkında bilgi verir.

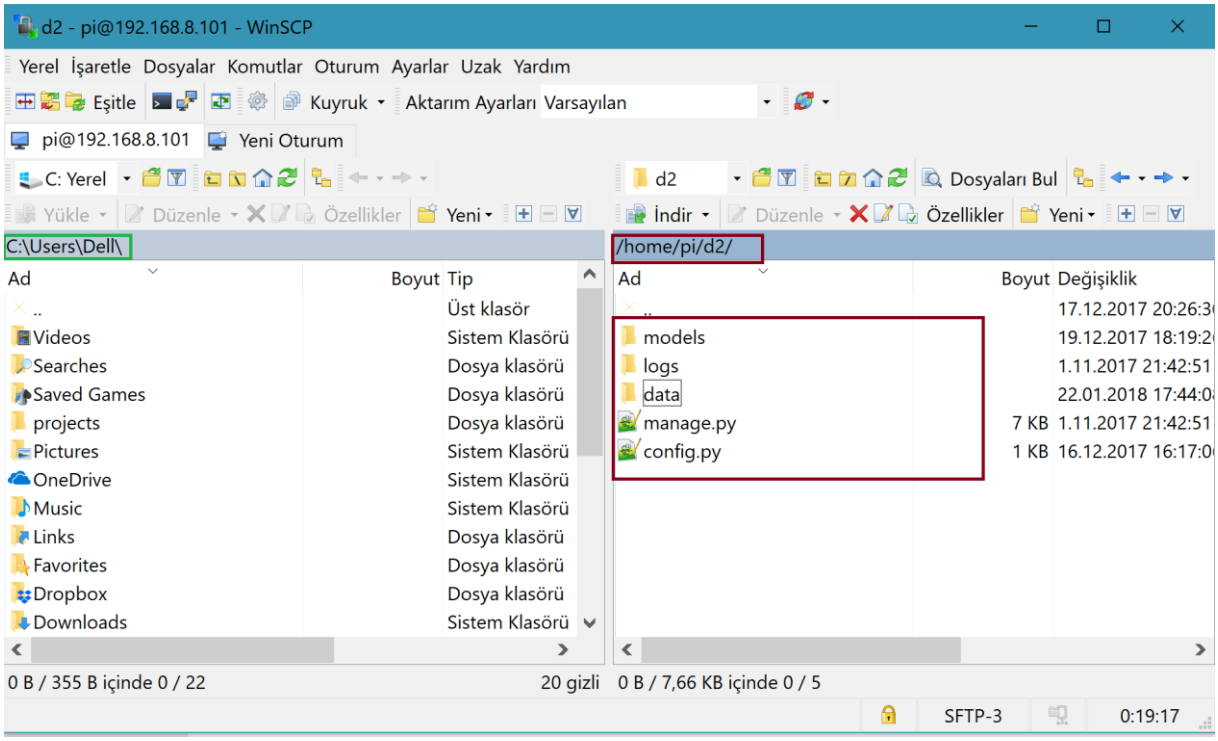
RPİ ayarlar bölümünden “Interfacing Options” kısmına giriş yaparak kullacağımız gerekli donanımları aktif hale getirmemiz gerekmektedir. Resim 4.6’da çevre birim ayarlar ekranı gösterilmiştir.



Resim 4.6. RPİ Çevre Birimler Ayar Ekranı

- **Camera:** RPİ ana kartı üzerinde bulunan CSI konektörüne bağlı kameranın aktif edildiği alandır.
- **SSH:** Farklı sistemler ile ağ üzerinden RPİ’nin iletişim kurabilmesi için gerekli SSH bağlantının aktif edildiği bölümdür.
- **VNC:** Raspian işletim sistemiyle birlikte gelen VNC uzaktan erişim programın aktif edildiği bölümdür.
- **SPI:** SPI bağlantıyla çalışan diğer donanımların, RPİ üzerinde çalışabilmesi için gerekli giriş-çıkış pinlerinin aktif edildiği alandır.
- **I2C:** I^2C bağlantıya sahip cihazlarla RPİ iletişim kurabilmesi için gerekli giriş çıkış pinlerinin aktif edildiği alandır.
- **Serial:** Uzaktan bağlantı sağlayarak komut satırına ulaşabilmek için, GPIO pinleri içerisinde yer alan UART pinlerinin aktif edildiği bölümdür
- **1-Wire:** Donanımsal olarak 1-wire protokülüyle çalışan sensörlerin RPİ ile iletişim kurabilmesi için gerekli bağlantıların aktif edildiği bölümdür.
- **Remote GPIO:** Rpi.GPIO isimli servis sayesinde ağ bağlantısı üzerinden RPİ’nin giriş-çıkış pinlerini kontrol eder.

RPİ üzerinde tüm ayarlamalar yapıldıktan sonra ana bilgisayarda gelişmiş metin editörü programı olan “notepade++” üzerinde yazdığımız kodları, RPİ ana klasörüne daha hızlı bir şekilde aktarılabilmesi için SSH bağlantı ile dosya paylaşımı imkânı sunan “WinSCP” programı kullanılmıştır. Resim 4.7’de WinSCP bağlantı ekranı gösterilmiştir.



Resim 4.7. WinSCP Bağlantı Ekranı

Ana bilgisayar üzerinde oluşturduğumuz “manage.py” ve “config.py” Python uzantılı kod dosyalarımızı WinSCP programı ile RPİ ana klasöründe bulunan /home/pi/ alt başlıkları altında yeni oluşturduğumuz “d2” klasörüne aktarıyoruz. Detaylı kodlar Ekler bölümünde verilmiştir.

Tüm bu işlemler gerçekleştirildikten sonra, RPİ’nin işlemci gücünün düşük olması sebebi ile ana bilgisayar üzerinde RC otonom araç ile elde edeceğimiz görüntü veri dosyalarının işlenebilmesi için Python ortamı yaratılmalıdır. Miniconda Python 3.6 64 bit sürümü https://conda.io/miniconda.html sitesinden, Windows için kurulum dosyası indirilir ve kurulum tamamlanır. Başlat menüsünden Anaconda terminal satırı açılarak Python Anaconda ortamı için aşağıdaki komutlar girilmelidir.

conda env create -f envs\windows.yml

activate donkey

Anaconda terminalini kapattıktan sonra tekrar açtığımızda, Python kitaplığı ile eşleştirmeyi yeniden etkinleştirmek için “activate donkey” komutunu yazmamız germektedir. Artık RC otonom araç kalibrasyon ve otopilot eğitim bölümüne geçebiliriz

4.3. Kalibrasyon ve Otopilot Eğitim

Uygulamamızın bu aşamasında, RPİ ve ana bilgisayar üzerinde gerekli programların kurulumu ve kodların aktarımından sonra direksiyon dönüş açısı ve gaz tepkimeleri için kalibrasyon ayarlamaları, otopilot eğitimi için hazırlanmış parkur hakkında bilgi, manuel otopilot eğitimi, eğitim dosyalarının ana bilgisayara aktarımı ve Python üzerinde Tensorflow ile aktarılan veri dosyalarının görüntü işleme süreci, eğitilen otopilot görüntü dosyasının RPİ ’ye aktarılarak RC otonom aracın parkur üzerinde ki davranışı incelenmiştir.

4.3.1. Direksiyon ve gaz kalibrasyonu

RC otonom aracımızın direksiyon açısı ve gaz tepkimeleri için kullandığımız, PCA9685 PWM 16 kanallı kontrol kartı üzerinde, direksiyonu kontrol eden servo motor ve fırçasız motoru kontrol ESC kablolarının 0 ve 1 kanallarına doğru bağlantı yapıldığının kontrol edilmesi gerekmektedir. RPİ’ ye aktarılmış yapılandırma dosyası olan config.py içerisinden bağlantılar kontrol edilir. Komut satırından config.py dosyasına ulaşabilmek için “nano ~/d2/config.py” kodunun girilmesi gerekmektedir. Yapılandırma dosyasının içeriği Resim 4.8’de gösterilmektedir.

Resim 4. 8. Otonom Araç Yapılandırma Ekranı

Yapılandırma dosyası kontrol edildiğinde, bağlantıların doğru yapıldığının anlaşılması için “THROTTLE_CHANNEL = 0” ve “STEERING_CHANNEL = 1” ifadelerini görmemiz gerekmektedir. Bu ifadeler PCA9685 PWM kontrol kartı üzerinde, gaz kontrolü sağlayan kabloların “0.” kanala, direksiyon kontrolünü sağlayan kabloların ise “1.” kanala bağlantılı olduğunu göstermektedir. Yapılandırma ifadelerinin açıklaması aşağıda verilmiştir.

```
#YOL
DRIVE_LOOP_HZ = 20  # ARACIN SANİYEDEKİ GÖRÜNTÜ KARE HIZ
DÖNGÜSÜ
MAX_LOOPS = 100000  #ARACIN MAKSİMUM DÖNGÜ SAYISI

#KAMERA
CAMERA_RESOLUTION = (120, 160)  # KAMERANIN GÖRÜNTÜ
ÇÖZÜNÜRLÜĞÜ
CAMERA_FRAMERATE = DRIVE_LOOP_HZ  # ARACIN GÖRÜNTÜ KARE HIZI

#DİREKSİYON
STEERING_CHANNEL = 1  # DİREKSİYON KABLOSUNUN BAĞLANTI KANALI
STEERING_LEFT_PWM = 425  #DİREKSİYON SOLA DÖNÜŞ PWM DEĞERİ
STEERING_RIGHT_PWM = 325  # DİREKSİYON SAĞA DÖNÜŞ PWM DEĞERİ

#GAZ
THROTTLE_CHANNEL = 0  # GAZ ESC KABLOSUNUN BAĞLANTI KANALI
THROTTLE_FORWARD_PWM = 500  # GAZ İLERİ SÜRÜŞ PWM DEĞERİ
THROTTLE_STOPPED_PWM = 400  # GAZ DURDURMA PWM DEĞERİ
THROTTLE_REVERSE_PWM = 410  # GAZ GERİ SÜRÜŞ PWM DEĞERİ

#EĞİTİM
BATCH_SIZE = 128
TRAIN_TEST_SPLIT = 0.8
```

```
#KONSOL
USE_JOYSTICK_AS_DEFAULT = False
JOYSTICK_MAX_THROTTLE = 0.25
JOYSTICK_STEERING_SCALE = 1.0
AUTO_RECORD_ON_THROTTLE = True
```

Yapılandırma ayarları içerisinde direksiyon ve gaz PWM değerlerinin doğru girilmesi, otonom aracı eğiteceğimiz parkurun şartlarına göre farklılıklar göstermektedir. Keskin virajlı bir parkurda direksiyon açısı maksimuma yakın değerde girilmeli ve gaz PWM değerleri de düşürülmelidir. Gireceğimiz bu değerler otonom araç için optimum verimlilik sunacaktır.

4.3.2. Otopilot Eğitim

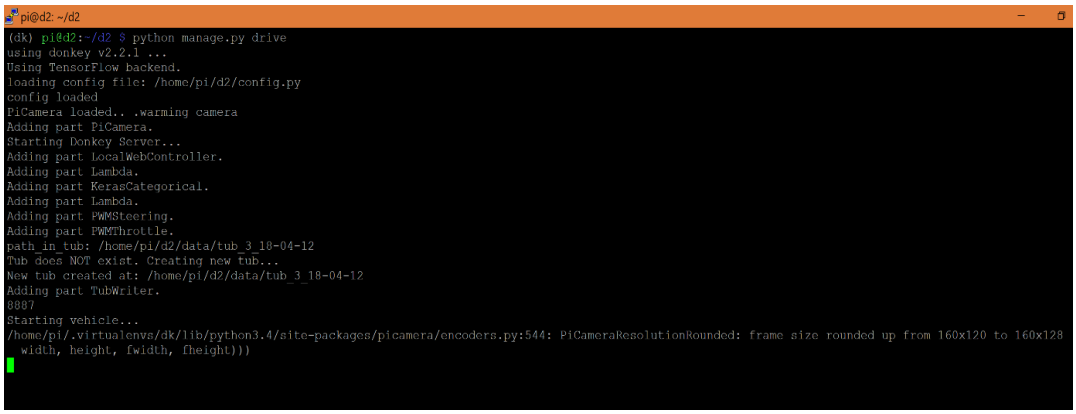
Otonom aracımızın kalibrasyon ayarları yapıldıktan sonra otopilot eğitim aşamasına geçebiliriz. Eğitim aşamasında kullanılan parkurlar, İstanbul Gelişim Üniversitesi Meslek Yüksekokulu bahçesinde ve hava şartlarının olumsuz gitmesi sonucunda Elektrik Laboratuvarında araştırmacı tarafından oluşturulmuştur. Resim 4. 9'da parkura ait üst görünüş gösterilmektedir.

Resim 4.9. Otopilot Eğitim Parkuru Üst Görünüş

Parkur 5 cm genişliğinde beyaz güçlü bir bant ile oluşturulmuştur. Otonom aracın farklı genişliklerdeki tepkilerinin görülebilmesi için en dar şerit aralığı 450 mm en geniş şerit aralığı 600 mm' dir. Kullanılan otonom araç şasisinin uzunluğu 500 mm, genişliği 360 mm, yüksekliği 210 mm ve tekerlek aralığı 316 mm'dir. Parkur üzerinde farklı bir araç sabitlenerek otonom aracın otopilot sürüş anında ki tepkileri de incelenmiştir. Resim 4.10'da kullanılan araç ve parkurun yan görünüşü gösterilmektedir.

Resim 4.10. Otopilot Eğitim Parkuru Yan Görünüş

RPİ ile ana bilgisayar SSH bağlantı prosedürü ile bağlanarak otopilot manuel sürüş eğitimi için Python terminali açılarak kod girişleri yapılır. Otonom aracın çalışabilmesi için gerekli kodların bulundu d2 klasörünün içerisine "cd ~/d2" komutu ile giriş yapılır. Klasör içerisinde ki eğitim kodlarının aktif hale getirilmesi, eğitimin başlayabilmesi ve otopilot modeli için gerekli görüntü dosyası kayıt altına alınmalıdır. Python terminale "python manage.py" kodu girilerek manuel eğitim başlatılır. Resim 4.11'de otonom araç için manuel sürüş kod akışı gösterilmiştir.

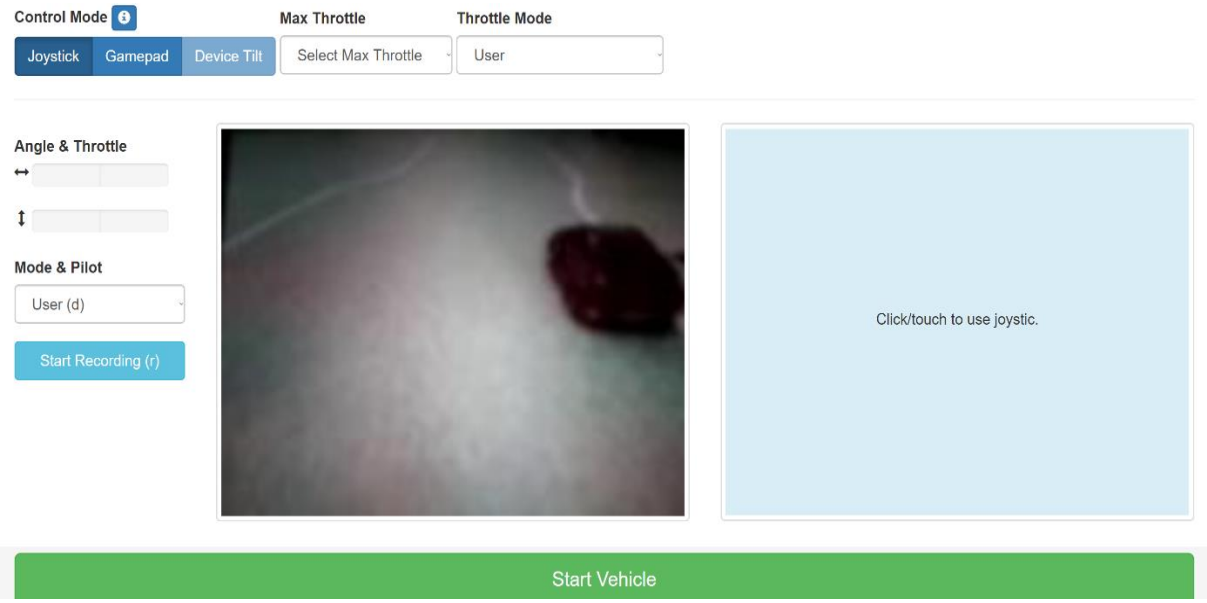


Resim 4.11. Otonom Araç Manuel Sürüş Kod Akışı

Otonom araç manuel sürüş çalışma kodu girildikten sonra sistem ilk önce Tensorflow kütüphanesini çalıştırarak tüm işlemler bu kütüphane altında yapılır. Sistem yapılandırma bilgilerini kontrol ederek, RPİ üzerinde bağlantılı olan Pi kamera modülüne bağlanır, 120x160 çözünürlükte çalışmasını başlatır. Otonom aracın, anlık görüntü aktarımı ve klavye fare gibi çevre birimler sayesinde otonom araç kontrolü sağlanabilmesi için Tornado kütüphanesi, web kontrol sunucusu "<your_cars_ip_address>:8887" adresinde tarayıcı üzerinden bağlantı adresini aktif etmektedir.

Aracımızın ip adresi girilerek web sunucusuna ulaşılabilir, Resim 4.12'de gösterilmiştir. Lambda fonksiyonu çalıştırılır. Manuel sürüş esnasında kayıt altına alınan görüntülerin işlenebilmesi için Keras kütüphanesi aktif hale getirilir. Görüntü veri dosyaları kayıt edilirken, her görüntüye karşılık gelen direksiyon ve gaz değerlerini saklayan .json (JavaScript Object Notation) uzantılı dosya ile birlikte veri dosyası oluşturulur. JSON, veri alış-verişi için geliştirilmiş, xml'e formatına karşılık olarak sunulmuş, javascript tabanlı bir veri değişim formatıdır. JSON'un kullanım amacı yüksek kapasiteli dosyaların veri alış-verişini hızlandırabilmek için daha küçük boyutlara dönüştürerek aktarımını sağlamaktır.

Yapılandırma ayarlarında girilen direksiyon ve gaz PWM değerleri sisteme aktarılır. Eğitim esnasında görüntülerin hangi dosya yoluna aktarılacağı gösterilir, "/home/pi/d2/data/tub_3" belirtilen klasörde saklanacaktır.

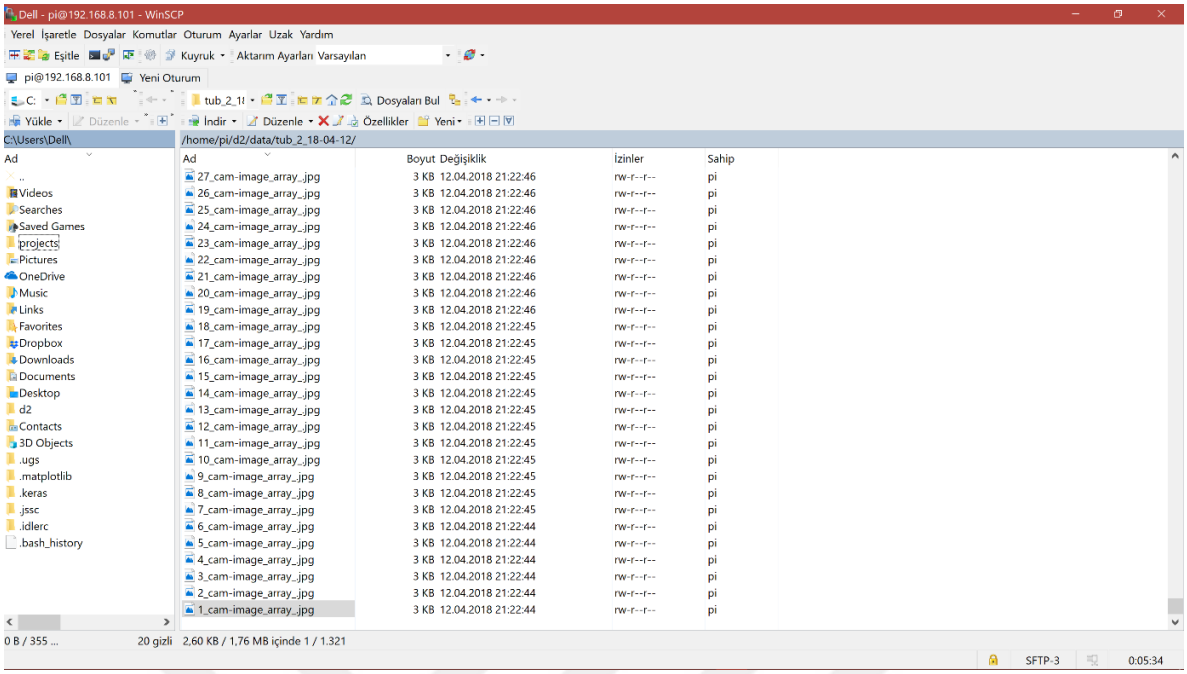


Resim 4.12. Otopilot Web Kontrol Sunucu Ekranı

Web kontrol sunucu ekranına bağlanılarak eğitim başlatılır. Sunucu üzerinde “Control Mode” sekmesi altında otonom aracımızın joystick veya gamepad ile kontrolü için seçenekler bulunmaktadır. Otonom aracımızı joystick ile çalıştırmak istenirse RPİ üzerinde bulunan USB girişleri ile bağlantı sağlanır ve çalışma kodu “python manage.py drive –js” değiştirilir. “Max Throttle” otonom aracın çıkabileceği maksimum hızı, yüzdelik dilimde gösterir. “Throttle Mode”, hız artışının manuel ya da otomatik seçilmesini sağlar. “Angle&Throttle” sekmesi, otonom aracın eğitimi aşamasından manevra ve hız dilimini göstermektedir. “Mode&Pilot” sekmesi ise eğitim aşamasında, “User” modu seçilerek tüm kontrolün kullanıcada olduğunu, eğitilen otonom aracın kendi kendine sürüş sağlaması için ise “Pilot” seçeneği aktif olmalıdır.

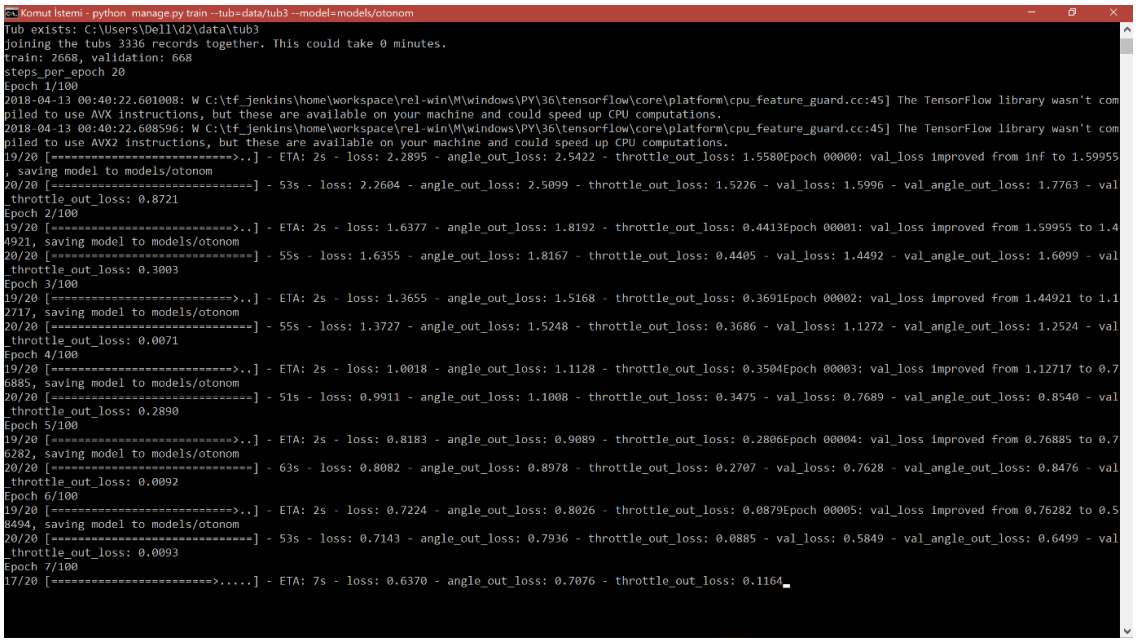
Sunucu ekranın sağ bölgesinde “Click/touch to use joystick” bölgesinde fare imleci getirilerek yukarı, aşağı, sağ ve sol yaparak araç kontrolü sağlanır. Jiroskop özellikli cep telefonlarıyla web sunucuna bağlanıldığında, cep telefonu aşağı, yukarı, sağ ve sol konumlandırılarak otonom araç kontrolü sağlanır. Eğitimlerimizde araç kontrolü klavye ile gerçekleşmiştir. Klavye üzerinde “Space” tuşu arabayı ve kaydı durdur, “r” tuşu kayıt değiştirir, “i” tuşu hız artırma, “k” tuşu hız azaltma, “j” tuşu sola çevir, “l” tuşu sağa çevir olarak kullanılmıştır. “Start Vehicle” sekmesi ile görüntüler kayıt altına alınır ve sunucu ekranının ortasında canlı görüntü aktarımı sağlanır.

Keras kütüphanesi sayesinde, aracın manuel sürüş özelliklerini gösterebildiği bir sinir ağı eğitimi başlatılır. Otonom araç eğitimi aşamasında kaliteli veri elde edilebilmesi için deneyimlerimiz sonucunda dikkat edeceğimiz bazı hususlar şunlardır. Verileri kaydetmeden önce araç birkaç kez pistte sürüş yapılarak deneyim sağlanır. Sürüş kalitesinden emin olunduğunda, yaklaşık 10 tur parkur üzerinde manuel sürüş sergilenir. 10-20 dakikalık süre aralığında yaklaşık 5 000 ile 20 000 görüntü arasında kaliteli veri topladıktan sonra, aracımız Ctrl-C kombinasyonu ile çalıştırılması durdurulur. Topladığımız veriler, en son konumlandırdığımız “/home/pi/d2/data/tub_3” klasörü içerisinde yer almaktadır. Resim 4.13’te görüntülerin aktarıldığı veri dosyası ekranı gösterilmektedir.



Resim 4.13. Görüntü Veri Dosyası Ekranı

RPİ çok güçlü olmadığından, verileri eğitmek için bir PC bilgisayarına aktarmamız gerekiyor. Kayıt altına alınan dosyalar WinSCP programı ile ana bilgisayara aktarılır. Ana bilgisayar üzerinde python terminali açılarak “python manage.py train --tub=data/tub3 --model=models/otonom” komutu girilerek, otonom ismi adı altında eğitim modeli TensorFlow kütüphanesi ile başlatılır, bu süreç yaklaşık 2 ile 4 saat arasında sürmektedir. Resim 4.14’te TensorFlow görüntü işleme akışı gösterilmiştir.



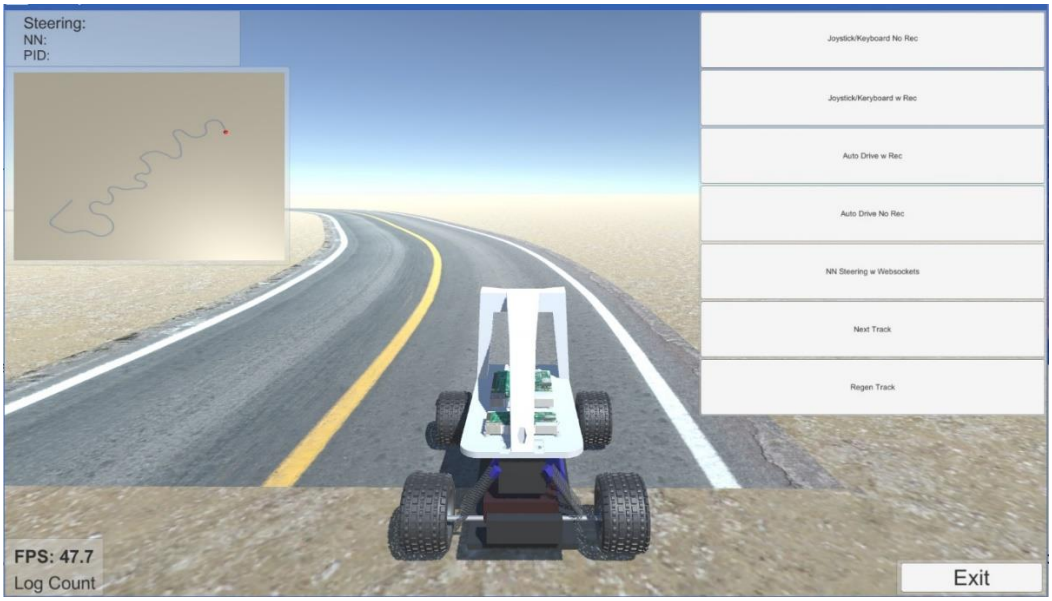
Resim 4.14. Tensorflow Görüntü İşleme Akışı

TensorFlow ile işlenen otopilot eğitim dosyası 3.189 KB boyutlarındadır. Ana bilgisayar üzerinde eğitilmiş otopilot dosyası, RPİ içerisinde "/home/pi/d2/models/" klasörüne aktarılır. Otonom aracın, otopilot sürüşünün incelenebilmesi için araç parkur üzerinde herhangi bir yere konumlandırılmış, RPİ komut satırında "python manage.py drive --model ~/d2/models/otonom" kodu girilerek web sunucusuna bağlanılmıştır. "Mode&Pilot" sekmesi altında Pilot seçeneği aktif edilerek RC otonom aracın parkur üzerindeki davranışları incelenerek kayıt altına alınmış, eksik görülen bölgelerde ki davranışları düzeltilmiştir.

4.4. Simülasyon

RC otonom aracımızın, otopilot eğitiminden sonra farklı genişlik, viraj ve uzunluktaki parkurlarda göstereceği davranışları PC üzerinden inceleyebilmek için, Diyrobocars ekibinin, dünya çapında yapılan otonom araç yarışları için hazırladıkları simülasyon programı kullanılmış ve davranışları incelenmiştir. Bu simülatör, Unity oyun platformu üzerine inşa edilmiş, iç fiziklerini ve grafiklerini kullanır. Otonom aracı kontrol etmek ve eğitimli modelimizi kullanmak için bir Python sürecine bağlanır. Bu simülatör görüntü veri dosyası biçiminde günlük verileri oluşturabilir.

"Log" dosyası altında kaydedilen verilere erişerek görüntü işleme süreci başlatılabilir. Resim 4.15'te simülasyon programı ana ekranı gösterilmiştir.



Resim 4.15. Simülasyon Programı Ana Ekranı

Simülatörün amacı, rastgele oluşturulmuş bir yol ve böylece farklı yol yüzeylerinde kilometrelerce kıvrım oluşturabilmektir. Bir yolda antrenman yapabilir veya tamamen farklı bir yüzeyi test edebilirsiniz. Simülasyon ekranı üzerindeki komutların çalışması şu şekildedir.

Joystick/Keyboard No Rec;

Otonom araç, joystick veya klavye ile sürülür. Bu seçenekte, görüntü kayıt altına almaz.

Joystick/Keyboard w Rec;

Otonom araç, joystick veya klavye ile sürülür. Bu seçenekte, görüntü verileri Log klasöründe saklanır.

Auto Drive No Rec;

Bu seçenek, otonom aracı yönlendirmek için yol bilgisini kullanır. Yönlendirmek için bir PID denetleyicisi kullanır, bu nedenle bazı salınımlar vardır. Bu seçenekte hiçbir veri saklanmaz.

Auto Drive w Rec;

Bu seçenek, "Auto Drive No Rec" seçeneğinden farklı olarak görüntü verilerini kayıt altına alır.

Next Track;

Yeni bir yol sahnesi oluşturur, bu yol yüzeyini ve şerit genişliğini değiştirecektir.

Regen Track;

Mevcut yüzey tipini kullanır, ancak yeni bir rastgele yol oluşturur.

Simülasyon programını çalıştırmak için sırası ile;

- Log dosyasının var olduğu ve boş olduğu kontrol edilmeli.
- Herhangi bir yol sahnesi seçilir.
- Auto Drive w Rec düğmesine basılmalı
- Çeşitli sürüş stilleri elde etmek için Max Speed, Prop ve Diff değerleri değiştirilebilir.

- 10 bin görüntü veri karesi kaydedene kadar 10-15 dakika beklenmeli
- Stop düğmesine basılarak simülasyon durdurulur,
- Exit düğmesine basılarak oyundan çıkılır.
- Log dosyası içerinde oluşturulan veriler için görüntü işleme teknikleri uygulanır.

Simülasyon programı sürüş PID ayarlama seçeneği sunmaktadır,

Max hız;

Bu ayar, PID otomatik sürücü sırasında hedef hızını belirler. Ayrıca, klavye kumandaları ile sürüş sırasında hızı da etkileyecektir (önerilmez).

Prob;

Bu seçenek, PID' nin sapmaya orantılı olarak, yola geri yönlendirmeyi ayarlayan P kısmıdır.

Diff;

Bu seçenek, PID' nin, sapma eğilimine yönelik bir türevi olarak, simülatörün yönlendirmeyi sınırlamaya çalışan D kısmıdır.

5. SONUÇ ve ÖNERİLER

Günümüzde sensor ve izleme teknolojilerinin yeterince gelişmesiyle birlikte, araçlarda akıllı sistem teknolojileri kullanılarak sürücüsüz gidebilen araçlar üzerine önemli çalışmalar yürütülmektedir. Otonom araçlar teknolojik yapıları itibariyle, kontrol mekanizmaları sayesinde herhangi bir kullanıcıya ihtiyaç duymadan, sensörlerden alınan verileri işleyerek trafik akışını, yol durumunu ve çevresel faktörleri kontrol ederek, kendi kendine gidebilen yapıları sayesinde, sürücü hatalı kazaları en aza indirerek bu tür kazaların önüne geçilmeyi mümkün kılmıştır.

Otonom çalışma yapısına sahip araçlar, gelişen sensörler (radar, lidar, GPS, odometri vb.) sayesinde, bilgisayar görüşü gibi teknolojiler ve tekniklerden faydalanılarak çevrede bulunan diğer canlı veya cansız varlıkları algılayabilmektedir. Böylelikle trafik akışında araç ve yaya güvenliği, araç verimliliği ve iyileştirme gibi sayısız avantajlar sunmaktadır. Tam bir otonom arabaya doğru gidilen otonom teknolojisi, araştırma endüstrileri tarafından da desteklenmektedir. Bu alanda karşılaşılan başlıca zorluklardan biri araba ve çevrenin mevcut durumu hakkında algılayıcılar tarafından algılanan farklı belirsizlikler ile başa çıkmaktır.

Bu uygulamada, RC araç şasi sistemi üzerine yerleştirilmiş bir kamera modülünden alınan veriler ile aracın parkur üzerindeki yönelimi ve manevra kabiliyeti, eğitilebilen bir YSA sistemi yardımıyla, otomatik olarak tamamlanmıştır. Sistem, parkur üzerindeki yol, çizgilerini referans alarak karar vermektedir. Eğitim aşamasında parkurdan sırayla 6 000, 10 000 ve 15 000 görüntü verisi elde edilerek, farklı modeller incelenmiştir. Kullanılan görüntüler ile en verimli otonom sürüş modelinin bulunması amaçlanmıştır.

Test edilen, 6 000 görüntü ile %83 doğruluk oranı, 10 000 görüntü ile %88 doğruluk oranı ve son olarak 15 000 görüntü ile %91 doğruluk oranıyla otonom araç ile şeritler arasında düzgün sürüş ve manevra kabiliyeti elde edilmiştir.

Test aşamasında, çok daha fazla görüntü verileriyle modelleme gerçekleşmesi sonucu, bu oranlar daha düşük seviyelere indirgenmiştir. Alınan görüntülerin her biri, farklı özelliklere sahip matrisler ile işlenerek modellere transfer edilmiştir.

Karar vermek için eğitilen YSA sisteminde, tüm görüntüler için oluşturulan farklı matrisler, giriş katmanında düğüm sayısını oluşturmuştur.

Bu matrisler, eğitim aşamasında alınan her bir örnek için karar mekanizması oluşturmuş ve otonom sürüş anında kameradan alınan görüntü ile hedef çıkış katmanı karşılaştırılarak araca yön verilmiştir.

Elde edilen 15 000 üzerindeki görüntü verilerinde, otonom sürüş oranında ki azalmanın sebebi, sistem içerisindeki karar mekanizmasında, farklı seçeneklerin yoğunluğu ile sistemin çalışma hızının birbirini karşılayamamasıdır. Düşük verilerde, yüksek otonom sürüş kabiliyetini artıran özellikler içerisinde en önemli etkenin şerit çizgilerinin kalınlığı, şerit aralarında ki mesafe ve kamera modülünün konumu olarak saptanmıştır. Verimliliği artırabilmek adına iki şerit arasına konumlandırılan, kesikli şerit sayesinde daha optimal bir sürüş izlenmiştir.

Minyatür yapıda incelenen otonom aracımıza farklı sensörler ilave edilerek kullanım amacı genişletilebilir. GPS modülü, mesafe ölçer sensörler ve ihtiyaç doğrultusunda kullanılacak diğer sensörler sayesinde toplu taşıma araçlarında, lojistikte, uzun yol araçlarında ve savunma sanayisi gibi birçok alanda geliştirilmeye açık bir sistemdir. TensorFlow yakın bir tarihte sunulmasına rağmen diğer görüntü işleme kütüphanelerinin daha alt seviyesinde kalsa da açık kaynak kodlu oluşu ve geliştiriciler tarafından rağbet gösterilmesi sonucu ileri ki zamanlarda kullanım alanının genişletilmesini sağlayacaktır.

KAYNAKLAR

1. Cheng, H., (2011). Autonomous Intelligent Vehicles: Theory, Algorithms, and Implementation. *New York: Springer Verlag London Limited, 10-12.*

2. Mitchell, T.M., (2006). Machine Learning Discipline. *Technical report CMU-ML-06-108, Computer Science Faculty,* Carnegie Mellon University.

3. Ozguner, U., Acatman, T. ve Redmil, K., (2011). *Autonomous Ground Vehicles.* USA, Artech House, 3-5

4. İnternet: Highlights of Robot Car History. URL: http://www.idsia.ch/~juergen/robotcars.html, Son Erişim Tarihi: 18.12.2017.

5. Forret, A., Konca, M., (2007). Autonomous Cars and Society. *Department of Social Science and Policy Studies Worcester Polytechnic Institute,* Worcester.

6. İnternet: Waymo Is Crushing The Field In Driverless Cars. URL: https://www.forbes.com/sites/chunkamui/2017/02/08/waymo-is-crushing it/#2d899975aa9f, Son Erişim Tarihi: 12.10.2017.

7. İnternet: How It Works. URL: https://www.mobileye.com/en-us/technology/how-it-works/, Son Erişim Tarihi: 01.11.2017.

8. Pomerleau. D.A., (1989). ALVINN: An Autonomous Land Vehicle in a Neural Network. *In Advances in Neural Information Processing Systems,* Morgan Kaufmann, San Mateo.

9. Ulmer, B., (1994). VITA II - Active collision avoidancein real traffic. *In Intelligent Vehicles Symposium,* Paris.

10. Dickmanns, D., Maurer, M., Behringer, R., Hildebrandt, T., Thomanek, F., Schiehlen, J., and Dickmanns. E.D., (1994). An advanced platform for visual autonomousroad vehicle guidance. *In Mobile Robots IX*, Vamors-P, Boston

11. Bertozzi, M., Broggi. A., (1998). GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection, *IEEE Trans. Image Process.* vol. 7, no. 1, pp. 62–81.

12. Güner, Ş., (2012). *Otonom Bir Otomobil İçin Hiz Kontrolörü Tasarimi Ve Uygulamasi*, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

13. Manyika, J., Chui, M., Bughin, J., Dobbs, R., Bisson, P., and Marrs, A. (2013). Disruptive technologies: Advances that will transform life, business, and the global economy, *McKinsey Global Institute,* Washington, D.C. 78-86

14. Bojarski, M., Testa, D.D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K., (2016). *End to end learning for self-driving cars*. arXiv preprint arXiv:1604.07316.

15. Kurakin, A., Goodfellow, I., and Bengio, S., (2016). *Adversarial examples in the physical world.* arXiv preprint arXiv:1607.02533.

16. Doruk, A., (2016). Şerit Takibi ve Hız Ayarı Yapabilen Otonom Robot Araba. Konferans*: Uluslararası Blgisayar Bilimleri ve Mühendisliği Konferansı*

17. William, B., Edvin, V.O., (2016). *Artificial Neural Network Autonomous Vehicle*, Master's Thesis, Kth Royal Institute of Technology, Stockholm, SWEDEN.

18. Bojarski, M., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., Muller, U., ve Zieba, K., (2016). *Visualbackprop:efficient visualization of cnns,* arXiv:1611.05418.

19. McCulloch, W. S., & Pitts, W., (1943). A logical calculus of the ideas immanent in nervousactivity. *The bulletin of mathematical biophysics*, 5(4), 115-133.

20. Hebb, D.O., (1949). *The organization of behavior: A neuropsychological theory*. Wiley NewYork, 20-21

21. Bell, J., (2015). *Machine Learning: Hands-On for Developers and Technical Professionals.* John Wiley & Sons, Inc. 92

22. Rosenblatt, F., (1958). The perceptron: A probabilistic model for information storage andorganization in the brain. *Psychological review*, 65(6), 386.

23. Werbos, P. J., (1974). *Beyond regression: new tools for prediction and analysis in the behavioral science,* Doctoral Thesis, Harvard University, USA.

24. Kohonen T., (1987), 'State Of The Art In Neural Computing', *IEEE First International Conference on Neural Networks*, 1, 79-90.

25. Awad, M., and Khanna, R., (2015). *Efficient Learning Machines: teories, Concepts,and Applications for Engineers and System Designers*. Apress Media, LLC.

26. Gülbağ, A., (2006). *"Yapay Sinir Ağı ve Bulanık Mantık Tabanlı Algoritmalar ile Uçucu Organik Bileşiklerin Miktarsal Tayini*", Doktora Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya.

27. Özer, K., (2009*). İstanbul Deniz Otobüslerinin Bir Hattında Yolcu Talep Tahmini, ,* Yüksek Lisans, Marmara Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul

28. Çuhadar, M., Kayacan, C., (2005). Yapay Sinir Ağları Kullanılarak Konaklama İşletmelerinde Doluluk Oranı Tahmini: Türkiye'deki Konaklama İşletmeleri Üzerine Bir Deneme, Anatolia, *Turizm Araştırmaları Dergisi, 16 (1): 24-30,*

29. Nabiyev, V., (2012). *Yapay Zeka*. Seçkin Yayınları, Ankara, 776 s.

30. Bilgin, S., (2008) *"Kalp Hızı Değişkenliğinin Dalgacık Dönüşümü ve Yapay Sinir Ağları Kullanılarak Analizi*", Doktara Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya.

31. Rumelhart, D.E., Hinton, G.E., Williams, R.J., (1958). *"Learning Representations by Backpropogation Errors",* Nature, Vol. 323, 533-536.

32. Hopfield, J.J., Tank D.W., (1985). *"Neural Computation of Decisions in Optimization Problems",* Biological Cybernetics, Vol. 52, 141-152.

33. Elmas, Ç., (2007). *Yapay Zeka Uygulamaları,* Seçkin Yayıncılık, Ankara, 88-137.

34. Kosko, B., (1988). "Feedback Stability and Unsupervised Learning", *Proceedings of IEEE, International Conference on Neural Network*, Vol. 1, 141-152.

35. Gareth M., (2012) "The Raspberry Pi single-board computer will revolutionise computer science teaching", *Engineering & Technology 7.3 stems with Different Network Technologies*, pp.26-26.

36. Jain, S., Vaibhav, A., Goyal, L., (2014). "Raspberry Pi based interactive home automation system through E-mail", *In Optimization, Reliabilty, and Information Technology (ICROIT), International Conference on IEEE*, pp. 277-280.

37. J. Chu, L., Ji, L., Guo, B., and Wang, R., (2004). *"Study on method of detectingpreceding vehicle based on monocular camera,"* in Proc. IEEE Intell. Vehicle, pp. 750–755.

38. İnternet: What is Python? Executive Summary. URL: https://www.python.org/doc/essays/blurb/, Son Erişim Tarihi: 10.01.2018.

39. İnternet: The Most Popular Python Data Science Platform. URL: https://www.anaconda.com/what-is-anaconda/, Son Erişim Tarihi: 12.01.2018.

40. Şeker, A., Diri, B., Balık, H.H., (2017).Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme, *Gazi Üniversitesi Mühendislik Bilimleri Dergisi*, Ankara 3(3): 47-64.

41. Theano Development Team, (2017) *"Theano: A {Python} framework for fast computation of mathematical expressions,"* arXiv e-prints, vol. abs/1605.02688.

42. İnternet: "DeepLearning 0.1 documentation", URL: http://deeplearning.net/tutorial/contents, Son Erişim Tarihi:

43. İnternet: "Caffe: An open source convolutional architecture for fast feature embedding," URL: http://caffe.berkeleyvision.org/, Son Erişim Tarihi: 05.02.2018

44. İnternet: "Torch | Scientific computing for LuaJIT.," NIPS Workshop on Machine Learning Open Source Software URL: http://torch.ch/, Son Erişim Tarihi: 06.02.2018

45. İnternet: "NVIDIA DIGITS." URL: https://developer.nvidia.com/digits, Erişim Tarihi: 06.02.2018

46. İnternet: "TensorFlow." URL: https://www.tensorflow.org/, Erişim Tarihi: 07.02.2018

47. İnternet: "Deeplearning4j: Open-source, Distributed Deep Learning for the JVM." URL: https://deeplearning4j.org/, Erişim Tarihi: 10.02.2018

48. İnternet: "Welcome to Knet.jl's documentation!", URL: http://denizyuret.github.io/Knet.jl/latest/., Erişim Tarihi: 10.02.2018

49. İnternet: “Julia ve Knet ile Derin Öğrenmeye Giriş,” URL: http://www.denizyuret.com/2016/09/julia-ve-knet- ile-derin-ogrenmeye-giris.html. Erişim Tarihi: 10.02.2018

50. İnternet: “Keras: The Python Deep Learning library”, URL: https://keras.io . Erişim Tarihi: 01.03.2018

EKLER

EK – 1. RC otonom araç otopilot eğitim ve sürüş kodları.

```
 import os
from docopt import docopt
import donkeycar as dk

#Kütüphaneler
from donkeycar.parts.camera import PiCamera
from donkeycar.parts.transform import Lambda
from donkeycar.parts.keras import KerasCategorical
from donkeycar.parts.actuator import PCA9685, PWMSteering, PWMThrottle
from donkeycar.parts.datastore import TubHandler, TubGroup
from donkeycar.parts.controller import LocalWebController, JoystickController

def drive(cfg, model_path=None, use_joystick=False):

    #Otonom araba başlat
    V = dk.vehicle.Vehicle()
    cam = PiCamera(resolution=cfg.CAMERA_RESOLUTION)
    V.add(cam, outputs=['cam/image_array'], threaded=True)

    if use_joystick or cfg.USE_JOYSTICK_AS_DEFAULT:
        ctr = JoystickController(max_throttle=cfg.JOYSTICK_MAX_THROTTLE,
                                 steering_scale=cfg.JOYSTICK_STEERING_SCALE,
                                 auto_record_on_throttle=cfg.AUTO_RECORD_ON_THROTTLE)
    else:
        #Web sunucusunu aç
        #Direksiyon gaz ve model oluşturma.
        ctr = LocalWebController()
```

```
V.add(ctr,
      inputs=['cam/image_array'],
      outputs=['user/angle', 'user/throttle', 'user/mode', 'recording'],
      threaded=True)

#Pilot modeli çalıştırma
def pilot_condition(mode):
    if mode == 'user':
        return False
    else:
        return True

pilot_condition_part = Lambda(pilot_condition)
V.add(pilot_condition_part, inputs=['user/mode'], outputs=['run_pilot'])

#otopilot çalıştır.
kl = KerasCategorical()
if model_path:
    kl.load(model_path)

V.add(kl, inputs=['cam/image_array'],
      outputs=['pilot/angle', 'pilot/throttle'],
      run_condition='run_pilot')

# Aracın hangi girişleri değiştireceğini seçin.
def drive_mode(mode,
               user_angle, user_throttle,
               pilot_angle, pilot_throttle):
    if mode == 'user':
        return user_angle, user_throttle
```

```
    elif mode == 'local_angle':
        return pilot_angle, user_throttle

    else:
        return pilot_angle, pilot_throttle

drive_mode_part = Lambda(drive_mode)
V.add(drive_mode_part,
      inputs=['user/mode', 'user/angle', 'user/throttle',
              'pilot/angle', 'pilot/throttle'],
      outputs=['angle', 'throttle'])

steering_controller = PCA9685(cfg.STEERING_CHANNEL)
steering = PWMSteering(controller=steering_controller,
                       left_pulse=cfg.STEERING_LEFT_PWM,
                       right_pulse=cfg.STEERING_RIGHT_PWM)

throttle_controller = PCA9685(cfg.THROTTLE_CHANNEL)
throttle = PWMThrottle(controller=throttle_controller,
                       max_pulse=cfg.THROTTLE_FORWARD_PWM,
                       zero_pulse=cfg.THROTTLE_STOPPED_PWM,
                       min_pulse=cfg.THROTTLE_REVERSE_PWM)

V.add(steering, inputs=['angle'])
V.add(throttle, inputs=['throttle'])

#Görüntü verilerini kaydet
inputs=['cam/image_array', 'user/angle', 'user/throttle', 'user/mode']
types=['image_array', 'float', 'float',  'str']
```

```
    th = TubHandler(path=cfg.DATA_PATH)
    tub = th.new_tub_writer(inputs=inputs, types=types)
    V.add(tub, inputs=inputs, run_condition='recording')

    #Aracı 20 saniye çalıştır.
    V.start(rate_hz=cfg.DRIVE_LOOP_HZ,
            max_loop_count=cfg.MAX_LOOPS)

    print("You can now go to <your pi ip address>:8887 to drive your car.")

def train(cfg, tub_names, model_name):
    '''
Yapay sinir ağını eğitmek için belirtilen isimlerde ki görüntü dosyalarını kullanarak
eğitimli modeli adını oluşturarak kayıt altına almak.    '''
    X_keys = ['cam/image_array']
    y_keys = ['user/angle', 'user/throttle']

    def rt(record):
        record['user/angle'] = dk.utils.linear_bin(record['user/angle'])
        return record

    kl = KerasCategorical()
    print('tub_names', tub_names)
    if not tub_names:
        tub_names = os.path.join(cfg.DATA_PATH, '*')
    tubgroup = TubGroup(tub_names)
    train_gen, val_gen = tubgroup.get_train_val_gen(X_keys, y_keys, record_transform=rt,
                                                    batch_size=cfg.BATCH_SIZE,
                                                    train_frac=cfg.TRAIN_TEST_SPLIT)
```

```
    model_path = os.path.expanduser(model_name)

    total_records = len(tubgroup.df)
    total_train = int(total_records * cfg.TRAIN_TEST_SPLIT)
    total_val = total_records - total_train
    print('train: %d, validation: %d' % (total_train, total_val))
    steps_per_epoch = total_train // cfg.BATCH_SIZE
    print('steps_per_epoch', steps_per_epoch)

    kl.train(train_gen,
             val_gen,
             saved_model_path=model_path,
             steps=steps_per_epoch,
             train_split=cfg.TRAIN_TEST_SPLIT)

if __name__ == '__main__':
    args = docopt(__doc__)
    cfg = dk.load_config()

    if args['drive']:
        drive(cfg, model_path = args['--model'], use_joystick=args['--js'])

    elif args['train']:
        tub = args['--tub']
        model = args['--model']
        cache = not args['--no_cache']
        train(cfg, tub, model)
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ŞANLI Erdem
Uyruğu : T.C.
Doğum tarihi ve yeri : 25.06.1985 Ankara
Medeni hali : Bekar
Telefon : 0 506 7900549
Faks : -
e-mail : esanli@gelisim.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Mekatronik Müh.	
Lisans	Elektronik Öğr.	01.07.2011
Lise	Meslek Lisesi	24.06.2002

İş Deneyimi

Yıl	Yer	Görev
2015	İstanbul	Öğretim Görevlisi

Yabancı Dil

İngilizce

Yayınlar

-

Hobiler

Hobi elektroniği, Maker hareketi, Kitap okumak, Spor, Sinema, Müzik

GELİŞİM GELİŞMEKTİR...