T.C.

SELÇUK ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ

Ocak-2019

KONYA

T.C.

SELÇUK ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ

GÖRÜNTÜ İŞLEME TABANLI OTONOM ARAÇ

Feyyaz YAVAŞ

BİTİRME PROJESİ

Ocak-2019

KONYA

BİTİRME PROJESİ KABUL VE ONAYI

..... tarafından hazırlanan "……..” adlı bitirme proje çalışması …/…/… tarihinde aşağıdaki jüri üyeleri tarafından oy birliği/oy çokluğu ile Selçuk Üniversitesi Teknoloji Fakültesi Elektrik Elektronik Mühendisliği bölümünde bitirme projesi olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Danışman Dr. Öğr. Üyesi Mehmet Akif ŞAHMAN	……..
Üye Dr. Öğr. Üyesi Güzin ÖZMEN	……..
Üye Arş. Gör. Yunus Emre ACAR	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Mehmet ÇUNKAŞ
Elektrik Elektronik Mühendisliği
Bölüm Başkanı

PROJE BİLDİRİMİ

Bu projedeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve proje yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Feyyaz YAVAŞ
16.01.2019

ÖZET
GÖRÜNTÜ İŞLEME TABANLI OTONOM ARAÇ

Feyyaz YAVAŞ

Selçuk Üniversitesi Teknoloji Fakültesi
Elektrik Elektronik Mühendisliği Bölümü

Danışman: Dr. Öğr. Üyesi Mehmet Akif ŞAHMAN

2019, 20 Sayfa

Jüri
Dr. Öğr. Üyesi Mehmet Akif ŞAHMAN
Dr. Öğr. Üyesi Güzin ÖZMEN
Arş. Gör. Yunus Emre ACAR

Bu projede derin öğrenme modeli geliştirilerek Raspberry Pi ile otonom araç kontrolü yapılmıştır. Bu projede yapay sinir ağları temel olarak kullanılmış ve görüntü işleme tekniklerinden yararlanılmıştır. Derin öğrenme modelinin eğitilmesi için gerekli veriler toplanmış ve eğitim sonucunda aracın testi gerçekleştirilmiştir.
Anahtar Kelimeler: Derin öğrenme, görüntü işleme, otonom araçlar, raspberry pi, yapay sinir ağları.

ÖNSÖZ

Yapay zekâ alanında makine öğrenmesi, derin öğrenme ve yapay sinir ağlarının kullanımı gün geçtikçe artmaktadır. Neredeyse üretilen her teknolojik alet bu tekniklerle donatılarak piyasada yer almaktadır. Günümüzde büyük veri kavramının ve etiketli veri sayısının artmasıyla derin öğrenme uygulamaları geliştirmek daha kolay hale gelmektedir. Derin öğrenme alanında birçok görüntü sınıflandırma uygulaması yapılabildiği gibi daha farklı problemlerin çözümünde de kullanılabilir. Ancak görüntü sınıflandırma yapabilmek için görüntü işleme tekniklerinden yararlanmak faydalı olacaktır.

Bu projede aracın otonom hareket edebilmesi için bir derin öğrenme modeli geliştirilmiştir. Bu modelin araç üzerinde test edilmesinde ve modelin eğitimi için veri toplarken görüntü işleme tekniklerinden yararlanılmıştır.

Proje çalışmalarım boyunca bana destek olan proje danışmanım Dr. Öğr. Üyesi Mehmet Akif ŞAHMAN' a teşekkür ve saygılarımı sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

ARM:	Acorn RISC Machine
BCE:	Binary Cross Entropy
CE:	Cross Entropy
CSI:	Camera Serial Interface
DC:	Direct Current
GB:	Giga Byte
GHz:	Giga Hertz
MSE:	Mean Squared Entropy
OpenCV:	Open Source Computer Vision
PWM:	Pulse Width Modulation
RAM:	Random Access Memory
ReLU:	Rectified Linear Unit
SGD:	Stochastic Gradient Descent

1. GİRİŞ

Günümüzde görüntü işleme teknikleri ile kanserli hücre tanısı yapmak veya askeri sahalarda kullanılacak sensörler geliştirmek gibi hayati önem taşıyan projeler geliştirilebilmektedir. Görüntü işlemedeki bu ileri teknoloji, başta makine öğrenmesi olmak üzere, derin öğrenme ve yapay sinir ağları gibi yan başlıkların kullanımıyla mümkün olmaktadır.

Derin öğrenme uygulamaları hesaplamaların daha hızlı yapılabildiği işlemciler sayesinde giderek önem kazanmaktadır. Derin öğrenme modelleri ile belli bir problemin çözümü artık daha kolaydır ve başarım oranları çok yüksektir.

Bu tezin amacı, aracın otonom hareket edebilmesi için derin öğrenme modeli geliştirmek ve eğitilen model ile aracın kontrolünü test etmektir.

Bu projede çok katmanlı yapay sinir ağı yerine derin öğrenme modeli kullanılmıştır. Derin öğrenme modeli hem standart sinir ağlarını hem de evrişimli sinir ağlarını kapsar. Evrişimli sinir ağları sayesinde başarım oranı standart sinir ağlarına göre daha fazladır.

2.KAYNAK ARAŞTIRMASI

Bruce ve Otter (2016) tarafından gerçekleştirilen otonom araç projesinde çok katmanlı sinir ağı modeli kullanılarak otonom aracın kontrolü sağlanmıştır. Modelin eğitilmesi için 900 adet etiketli veri kullanılmıştır ve başarım oranı %78' dir. Bu çalışmada eğitim için kullanılan etiketli veri sayısının ve görüntü büyüklüğünün eğitim başarısı üzerindeki etkileri de test edilmiştir.

Wang (2016) tarafından yapılan çalışmada Matlab ortamında çok katmanlı perseptron modeli kullanılarak yapay sinir ağı eğitilmiştir. Matlab ile anlık olarak aracın haberleşmesi sağlanarak araçtan alınan görüntüler yapay sinir ağına uygulanmıştır. Elde edilen çıkış verileri kullanılarak aracın otonom hareketi sağlanmıştır. Bu çalışmada otonom sürüşün yanı sıra trafik levhalarına ve trafik ışıklarına uyulmasını sağlayan bir yapı gerçekleştirilmiştir. Bu işlemleri gerçekleştirebilmek için Alfred Haar adında bir bilim adamı tarafından bulunan Haar sınıflayıcısı kullanılmıştır.

Ungurean (2018) tarafından yapılan çalışmada klasik kontrol sistemleri ve yapay sinir ağları arasındaki farklılıklar ortaya konularak aracın otonom kontrolü yapay sinir ağlarıyla sağlanmıştır. Yapılan bu çalışmada Raspberry Pi ile araç üzerinde bulunan kameradan alınan görüntüler eğitilen yapay sinir ağına uygulanmış ve aracın otonom kontrolü sağlanmıştır. Bu çalışmada farklı parametreler kullanılarak farklı eğitim sonuçları gözlemlenmiştir.

3. MATERYAL VE YÖNTEM

3.1. Görüntü İşleme

Görüntü işleme, gelişmiş bir görüntü elde etmek ya da ondan bazı yararlı bilgiler çıkarmak için kullanılan bir tür sinyal işleme tekniğidir. Bu yöntemin girdisi video kesiti ya da fotoğraf görüntüsüdür. Çıktısı ise görüntünün istenilen ya da dikkat edilmesi gereken bölümüne karşılık gelir.

Otonom araç için kullanılacak görüntü işleme yöntemleri, giriş görüntüsünün gri seviyeli resme dönüştürülmesi ve kenar algılama yönteminden oluşmaktadır. Elde edilen son görüntü veri setinde ve eğitilen derin öğrenme modelinde giriş değerlerini ifade etmektedir.

Bu çalışmada, Raspberry Pi ile kameradan elde edilen görüntülerin gri seviyeli resme dönüşümü ve kenar algılama işlemi OpenCV görüntü işleme kütüphanesi kullanılarak yapılmaktadır.

3.1.1. Gri seviye dönüşümü

Görüntü işleme uygulamalarında genellikle kameradan alınan görüntüler üç kanallı renkli görüntülerdir. Bu görüntüler gri seviyeli görüntülere dönüştürülürken görüntü matrisi üzerinde aynı indisteki değerlerin aritmetik ortalaması alınarak tek kanallı bir matris elde edilir. Oluşturulan bu matris gri seviyeli görüntüdür.

3.1.2. Canny kenar algılama

Canny Kenar Algılama, popüler bir kenar algılama algoritmasıdır. 1986 yılında John F. Canny tarafından geliştirilmiştir. Bu yöntem yararlı yapısal bilgiyi farklı görme nesnelerinden çıkarma ve işlenecek veri miktarını önemli ölçüde azaltma tekniğidir. Çeşitli bilgisayarlı görü sistemlerinde yaygın olarak uygulanmaktadır.

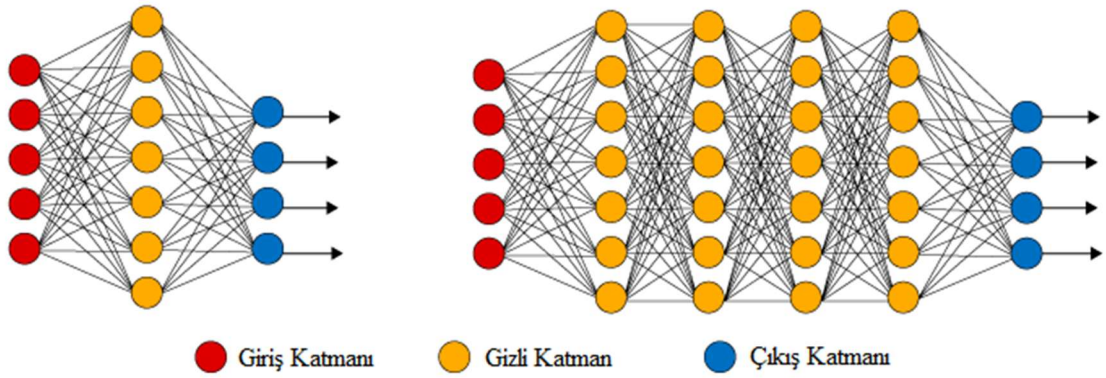
(a) Orijinal Görüntü (b) Gri Seviyeli Görüntü (c) Canny Kenar Algılama

Şekil 3.1. Giriş görüntüsüne uygulanan görüntü işleme yöntemleri

3.2. Derin Öğrenme

Derin öğrenme, bir makine öğrenmesi tekniğidir. Derin öğrenmenin temel prensibi insanlara doğal gelen şeyleri bilgisayarlara öğretmektir. Derin öğrenme, otonom araçların hareketini sağlamalarını, dur işaretini tanımalarını veya bir yayayı bir elektrik direğinden ayırmalarını sağlayan bir teknolojidir.

Derin öğrenmede, bir bilgisayar modeli sınıflandırma görevlerini doğrudan görüntü, metin veya seslerden yapmayı öğrenir. Derin öğrenme modelleri, bazen insan seviyesindeki performansı aşarak, en gelişmiş doğruluğu elde edebilir. Modeller, çok sayıda etiketli veri içerir ve yapay sinir ağı mimarisi kullanılarak eğitilir. Derin öğrenme modelleri birden fazla gizli katmandan oluşur.

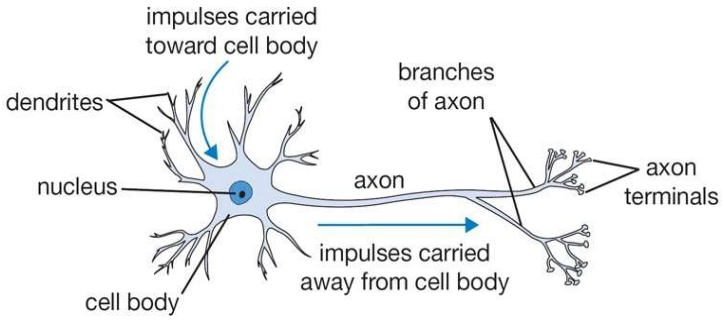


Şekil 3.2. Basit yapay sinir ağı ve derin sinir ağı modeli

3.2.1. Yapay sinir ağları

Yapay sinir ağları, insan beyninin bilgi işleme tekniğinden faydalanarak geliştirilmiş sistemlerdir. Yapay sinir ağı ile basit bir biyolojik sinir sistemi taklit edilir. Bu sinir hücreleri nöronlar içerir ve bu nöronlar çeşitli şekilde birbirlerine bağlanarak yapay sinir ağını oluştururlar.

İnsandaki bir sinir hücresinin(nöron) biyolojik modeli Şekil 3.3.' de ve matematiksel modeli Şekil 3.4.' deki gibidir:



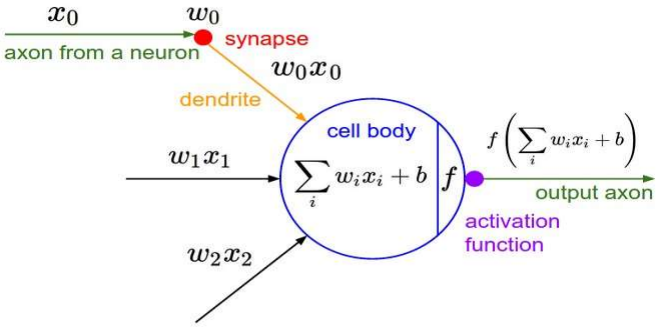
Şekil 3.3. Sinir hücresinin biyolojik modeli

Akson: Çıkış darbelerinin üretildiği elektriksel aktif gövdedir ve gövde üzerinde iletim tek yönlüdür. Sistemin çıkışıdır.

Dentritler: Diğer hücrelerden gelen işaretleri toplayan elektriksel anlamda pasif kollardır. Sistemin girişidir.

Çekirdek: Akson boyunca işaretlerin periyodik olarak yeniden üretilmesini sağlar.

Sinaps: Nöronların aksonlarının diğer dentritlerle olan bağlantısını sağlar.



Şekil 3.4. Sinir hücresinin matematiksel modeli

Yukarıdaki nöronun çıkış değeri denklem 3.1' de gösterildiği gibi hesaplanır. Burada x_i giriş değerini, w_i ağırlık değerini, b bias değerini, σ aktivasyon fonksiyonu ve y nöronun çıkış değerini temsil eder.

$$y = \sigma(w_i x_i + b) \tag{3.1}$$

3.2.1.1. Aktivasyon fonksiyonları

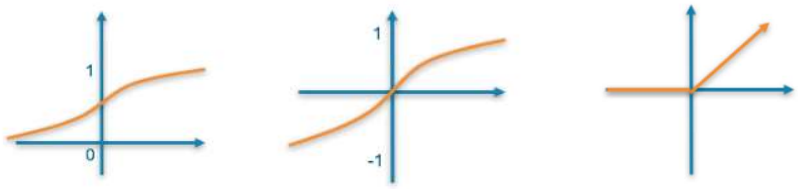
Aktivasyon fonksiyonları, nörona gelen net girdiyi işleyerek nöronun bur girdiye karşılık hangi çıktıyı üreteceğini belirler. Aşağıda en sık kullanılan aktivasyon fonksiyonları formülleriyle birlikte açıklanmıştır.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3.2}$$

$$tanh(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{3.3}$$

$$ReLU(x) = max(0, x) \tag{3.4}$$

Sınıflandırma problemlerinde softmax aktivasyon fonksiyonu kullanılır. Softmax fonksiyonunun çıktısı, kategorik bir olasılık dağılımına eşdeğerdir, sınıfların herhangi birinin doğru olma ihtimalini söyler. Bu projede softmax aktivasyon fonksiyonu ile çıkış katmanında bulunan beş adet nöronun sınıflandırılması yapılmıştır.



(a) Sigmoid (b) Hiperbolik tanjant (c) ReLU

Şekil 3.5. Aktivasyon fonksiyonlarının grafiği

3.2.1.2. Kayıp(Yitim) fonksiyonları

Kayıp fonksiyonları, bir problemin çözümü için tasarlanan sinir ağı modelinin başarımını belirleyen bir fonksiyondur. Çeşitli optimizasyon teknikleriyle hatanın sıfıra yaklaşması sağlanır. Kayıp fonksiyonu sınıflandırma problemi ya da sayısal bir değerin tahmin edilmesi problemine göre belirlenir. Aşağıda en çok kullanılan yitim fonksiyonları açıklanmıştır.

Ortalama karesel hata fonksiyonu, öngörülen değer ile gerçek değer arasındaki ortalama kare farkını bulur.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2 \tag{3.5}$$

İkili çapraz entropi fonksiyonu, iki olasılık dağılımı arasındaki farkı belirler. Çıkış tek nörona sahip ve aktivasyon fonksiyonu sigmoid ise tercih edilir.

$$BCE = -(y\log(y') + (1-y)\log(1-y')) \tag{3.6}$$

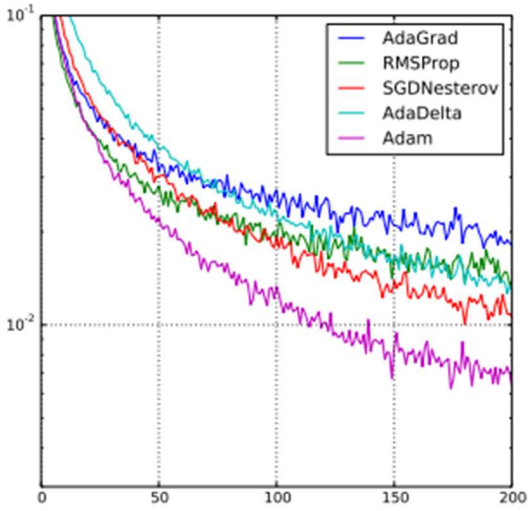
Çapraz entropi fonksiyonu, birden fazla sınıftan oluşan, çıkış katmanında aktivasyon fonksiyonu softmax olan ve tek bir etiket öngören modellerde kullanılır. Bu projede softmax ile sınıflandırma yapıldığı için çapraz entropi fonksiyonu kullanılmıştır. Çapraz entropi fonksiyonuna ait denklem formül 3.7'de verilmiştir. Burada y gerçek değeri, y' öngörülen değeri, M ise sınıf sayısını ifade etmektedir.

$$CE = -\sum_{\mathrm{i}}^{M} y_i \log(y_i') \tag{3.7}$$

3.2.1.3. Optimizasyon algoritmaları

Derin öğrenme uygulamalarında öğrenme işlemi temel olarak bir optimizasyon problemidir. Doğrusal olmaya problemlerin çözümünde optimum değeri bulmak için optimizasyon yöntemleri kullanılmaktadır. Derin öğrenme uygulamalarında genel olarak Stochastic Gradient Descent, Adagrad, Adadelta, Adam, Adamax gibi optimizasyon algoritmaları kullanılmaktadır. Bu algoritmalar arasında başarım ve hız bakımından farklılıklar bulunur.

SGD diğer optimizasyon yöntemlerine göre daha yavaş olup görüntü tanıma problemlerinde kötü sonuçlar verebilmektedir. Adagrad ise görüntü tanıma gibi veriler için daha uygundur. Adagrad' da her parametrenin kendi öğrenme hızı vardır ve algoritmanın özelliklerine bağlı olarak öğrenme oranı giderek azalır. Bu nedenle belli bir noktadan sonra sistem öğrenmeyi bırakır. Adadelta optimizasyon algoritması Adagrad' ın bu sorununu çözerek hızlı düşüşü önler. Şekil 3.6' da optimizasyon algoritmalarının rakam tanıma veri seti üzerindeki performansını gösteren grafik yer almaktadır.



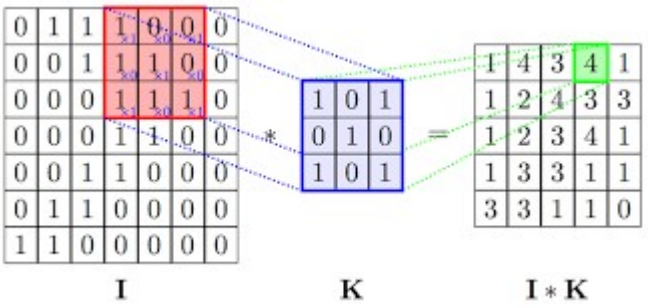
Şekil 3.6. Farklı optimizasyon algoritmalarının performansı

3.2.2. Evrişimli sinir ağları

Evrişimli sinir ağları, temel olarak görüntüleri sınıflandırmak, benzerlikle kümelemek ve sahnelerde nesne tanıma yapmak için kullanılan derin yapay sinir ağlarıdır. Yüzleri, bireyleri, sokak işaretlerini, tümörleri ve görsel verinin diğer birçok yönünü tanımlayabilen algoritmalardır. Evrişimli sinir ağları sıradan sinir ağlarına benzer şekilde öğrenilebilir ağırlıklar ve önyargıları olan nöronlardan oluşur. Evrişimli sinir ağları, evrişim, ortaklama ve tam bağlantı katmanlarından oluşur. Evrişimli sinir ağlarında ilk katman her zaman evrişim katmanıdır ve birden fazla kullanılabilmektedir.

3.2.2.1. Evrişim katmanı

Evrişim katmanı evrişimli sinir ağlarının temel işlemidir. Evrişim işlemi, bir giriş görüntüsünden özellik çıkarmayı sağlar. Evrişim katmanında bir görüntünün kenar algılama, bulanıklaştırma ve keskinleştirme gibi işlemleri farklı filtreler kullanılarak gerçekleştirilebilir. Şekil 3.7’ de evrişim işlemi gösterilmiştir. Burada I giriş matrisini, K 3x3 filtreyi ve I*K konvolüsyon çarpımını ifade etmektedir.

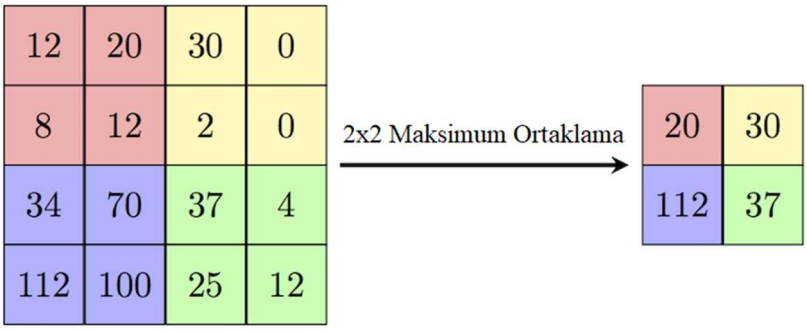


Şekil 3.7. Matrislerde konvolüsyon çarpımı

3.2.2.2. Ortaklama katmanı

Ortaklama, matrislerin boyutunu azaltan ancak önemli bilgileri koruyan alt örnekleme veya alt örnekleme olarak adlandırılır. Ortaklama üç farklı türde olabilir. Bunlar maksimum ortaklama, ortalama ortaklama ve toplam ortaklamadır. Ortaklama katmanında genellikle maksimum ortaklama kullanılır.

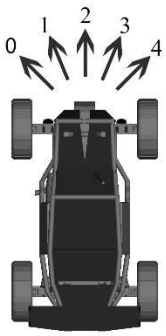
Maksimum ortaklama, özellik haritasındaki en büyük değeri alır. Şekil 3.8’ de maksimum ortaklama örneği verilmiştir.



Şekil 3.8. Maksimum ortaklama işlemi

3.3. Otonom Aracın Yapısı

Bu projede tasarlanan araç otonom(kendi kendine) hareket etme kabiliyetine sahip olup mevcut donanımlar sayesinde kontrolü sağlanmaktadır. Bu aracın otonom hareket etmesini sağlayan temel yapı derin öğrenme tekniğidir. Otonom hareketin sağlanabilmesi için araç üzerinde bulunan kameradan alınan görüntüler işlenerek, eğitilen derin öğrenme modeline uygulanır ve modelin çıktısına göre aracın direksiyon açısı değiştirilir. Şekil 3.9' da aracın temel yapısı mevcuttur. Burada aracın direksiyon açısı beş farklı sınıfta temsil edilmektedir. Her bir sınıf sayesinde araç farklı açı değerlerinde hareket ettirilebilmektedir.



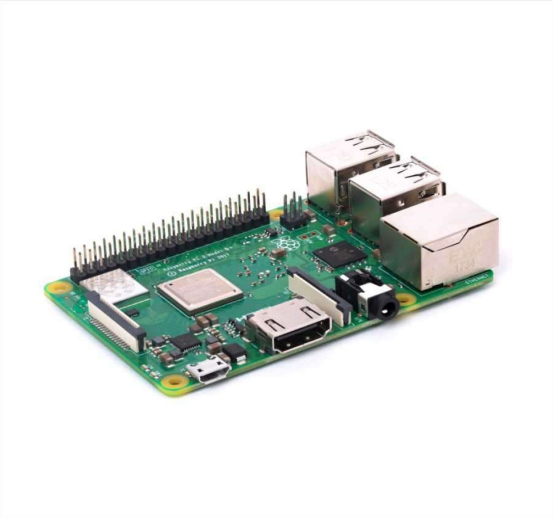
Şekil 3.9. Farklı direksiyon açılarına ait sınıfları gösteren tasarım

3.3.1. Elektronik donanım

3.3.1.1. Raspberry pi 3 model b+

Raspberry Pi Birleşik Krallık' ta Raspberry Pi Vakfı tarafından bilgisayar bilimini öğretmek amacıyla geliştirilmiş kredi kartı büyüklüğünde tek kartlı bir bilgisayardır.

Raspberry Pi genellikle gömülü sistemlerde ve işletim sistemi uygulamalarında kullanılır. Bu projede tercih edilme sebebi görüntü işleme uygulamalarında yüksek performans vermesi ve eğitilen yapay sinir ağı modelinin kullanımına olanak sağlamasıdır.



Şekil 3.10. Raspberry Pi 3 Model B+

Bu projede Raspberry Pi 3 Model B+ kullanılmıştır. Aşağıda bu kartın özellikleri mevcuttur.

- 1.4GHz 64-bit ARM Cortex-A53 işlemci
- 1GB RAM
- Kamera bağlantısı için CSI portu
- 40 adet giriş çıkış pini
- İşletim sistemi ve depolama için kart yuvası

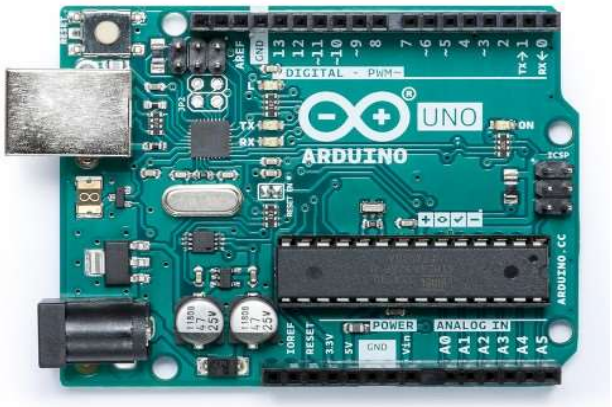
3.3.1.2. Raspberry pi kamera modülü

Raspberry Pi ile uyumlu olan kamera modülü sayesinde araç üzerinden anlık olarak görüntü alınarak kolayca işlem yapılabilmektedir. Raspberry Pi üzerinde kamera için bağlantı portu bulunduğundan bu kamera tercih edilmiştir.

3.3.1.3. Mikrodenetleyici

Bu projede mikrodenetleyici olarak Arduino Uno kartı kullanılmıştır. Bu kartın tercih edilme sebebi mikrodenetleyici programlama devrelerine gereksinim duymadan programlanabilir olmasıdır. Araç üzerindeki DC motor ve servo motor bu

mikrodenetleyici ile kontrol edilmektedir. Arduino ile Raspberry Pi arasında seri haberleşme protokolü kolayca veri alışverişi yapılabilmektedir.



Şekil 3.11. Arduino Uno R3

3.3.1.4. Motor sürücü devresi

Araç üzerinde bulunan DC motorun kontrolünün sağlanması ve devre besleme gerilimlerinin yapılabilmesi için bu sürücü devresi kullanılmaktadır. Mikrodenetleyici tarafından PWM sinyali gönderilerek DC motorun kontrolü yapılmaktadır.

3.3.1.5. Servo motor

Servo motor PWM sinyali ile çalışmaktadır. Bu PWM sinyali bir mikrodenetleyici ya da uzaktan kumandadan sağlanabilmektedir. Servo motorlar hareket etmeleri için bir komut aldıklarında önce istenilen pozisyona hareket ederler, sonrasında ise o pozisyonda kalırlar. Servo motorlar genellikle 0°-180° arasında hareket ettirilebilmektedir. Bunun yanında 360° dönebilen servo motorlar da vardır. Bu projede otonom aracın direksiyon açısını belirlemek için kullanılmaktadır. Bu projede kullanılan minimum ve maksimum açı değerleri 45° ve 135° olarak belirlenmiştir.

Şekil 3.12. Tower Pro MG995 Servo Motor

3.3.2. Programlama dilleri ve kütüphaneler

3.3.2.1. Python

Python, nesne yönelimli, yorumlanabilen, birimsel ve etkileşimli bir programlama dilidir. Girintilere dayalı basit sözdizimi, dilin öğrenilmesini ve akılda kalmasını kolaylaştırır. Bu da ona söz diziminin ayrıntıları ile vakit yitirmeden programlama yapılmaya başlanabilen bir dil olma özelliği kazandırır. Modüler yapısı, sınıf dizgesini ve her türlü veri alanı girişini destekler. Hemen hemen her türlü platformda çalışabilir. Python ile sistem programlama, kullanıcı arabirimi programlama, ağ programlama ve uygulama programlama gibi birçok alanda yazılım geliştirilebilir.

Bu projede birden fazla Python kütüphanesi kullanılmaktadır. Bunlar Raspberry Pi' a ait kütüphaneler ve matris işlemlerinde kullanılan numpy kütüphanesidir.

3.3.2.2. OpenCV

OpenCV, ilk olarak 1999 yılında Intel'in Rusya'daki laboratuvarlarında geliştirilmeye başlanmış açık kaynak kodlu görüntü işleme kütüphanesidir. Bilgisayarlı görü uygulamalarında ortak bir altyapı sağlayarak makine algısının kullanımını hızlandırmak için oluşturulmuştur. Bu projede OpenCV kullanılarak görüntü işleme gerçekleştirilmektedir.

3.3.2.3. Keras

Keras, Python programlama dilinde yazılmış, sinir ağı modelleri oluşturmak için özel olarak kullanılan üst düzey bir kütüphanedir. Keras, fikirlerin hızlı bir şekilde uygulanması için özel olarak geliştirilmiştir. Basit ve oldukça modüler bir ara yüze sahiptir, bu da karmaşık sinir ağı modellerini oluşturmayı kolaylaştırır.

Keras ile eğitilen yapay sinir ağı modelleri, diğer derin öğrenme çerçevelerinden daha geniş bir platform yelpazesinde kolaylıkla kullanılabilir.

3.3.2.4. H5py

Çok sayıda sayısal veriyi saklamaya olanak sağlayan Python paketidir. Binlerce veri seti tek bir dosyada saklanabilir, istenilen şekilde kategorize edilebilir ve etiketlenebilir. Bu kütüphane ile eğitilen yapay sinir ağı modelleri kolayca dosya haline

getirilip tekrar kullanılabilirler. H5py kütüphanesi ile oluşturulan dosya içerisinde sinir ağı modeline ait tüm değerler yer alır.

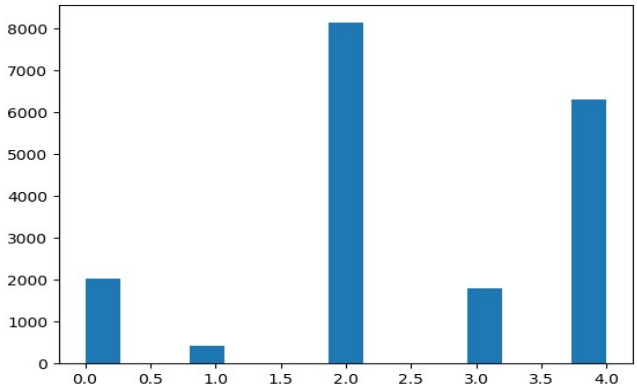
3.4. Veri Seti Oluşturma

Otonom araçta kullanılacak derin öğrenme modelinin eğitimi için gerekli olan eğitim ve test verileri, aracın kullanıcı tarafından hareket ettirilirken anlık olarak kameradan alınan görüntü ve o anki direksiyon açısı(servo açısı) değerlerinin kaydedilmesiyle oluşturulurmuştur. Veri setindeki direksiyon açısı değerleri Raspberry Pi'a Arduino tarafından seri haberleşme yapılarak gönderilmiştir. Elde edilen veri seti ".csv " formatında Raspberry Pi' da kaydedilmiştir. Elde edilen veri setine ait giriş değerlerinden örnekler Şekil 3.13' de verilmiştir.

Şekil 3.13. Veri setindeki girişlere ait örnek görüntüler

Oluşturulan veri setinde toplam 18720 adet veri bulunmaktadır ve bu verileri elde etmek için araç yaklaşık olarak 32 dakika boyunca veri toplamıştır. Veri setinde çıkış değerleri beş sınıfta temsil edilmektedir.

Veri setindeki sınıflara ait grafik Şekil 3.14.' deki gibidir. Burada her bir sınıf farklı açı değerini ifade etmektedir. Araç veri toplarken genellikle ileri ve sağa hareket ettirildiği için 2 ve 4 değerleri diğerlerine göre daha fazladır.



Şekil 3.14. Veri setinin histogram grafiği

3.5. Derin Öğrenme Modelinin Oluşturulması ve Eğitilmesi

Derin öğrenme modelinin oluşturulması ve eğitimi Google Colab üzerinde Python programlama dili kullanılarak gerçekleştirilmiştir. Modelin oluşturulması ve eğitimi için Keras derin öğrenme kütüphanesi kullanılmıştır.

Derin öğrenme modeli oluşturulmadan önce veri seti eğitim ve test verisi olmak üzere iki gruba ayrılmıştır. Eğitim verisi tüm verinin %80'ini, test verisi ise tüm verinin %20' sini kapsayacak şekildedir. Bu işlemlerden sonra veri setinin tümüne normalizasyon uygulanarak 0-255 arasındaki giriş değerleri 0-1 arasına normalize edilmiştir. Normalizasyon işlemi formül 3.8' deki gibidir.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.8}$$

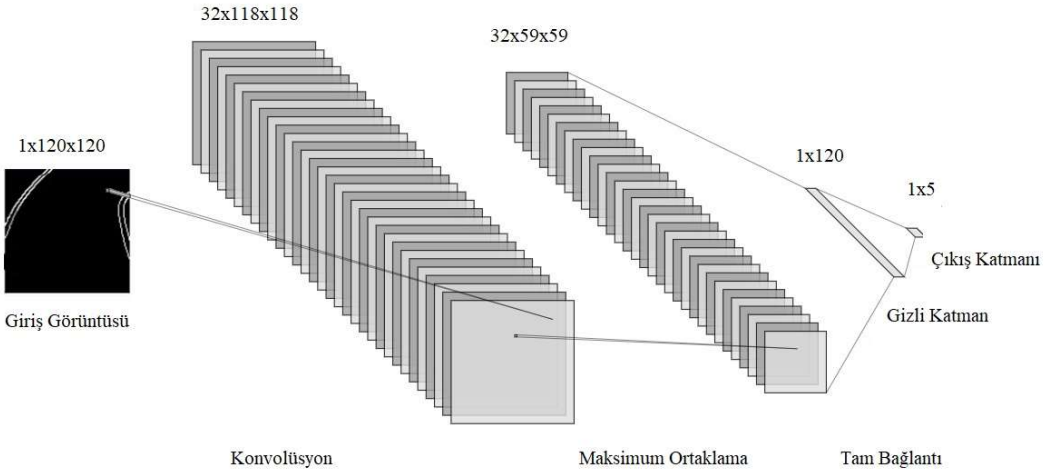
Veri setinde yapılan işlemlerin ardından modelin eğitiminde kullanılacak tüm parametreler tanımlanmıştır. Bu parametreler giriş görüntüsünün boyutu, çıkıştaki sınıf sayısı, eğitimin kaç tur olacağı ve her iterasyonda kaç adet görüntünün eğitileceğini belirten değerlerden oluşmaktadır.

Keras derin öğrenme kütüphanesi yardımıyla eğitilecek model tanımlanarak bu model için gerekli olan evrişim katmanı, ortaklama katmanı ve tam bağlantı katmanları eklenmiştir. Modelin tanımlanması için gerekli kod satırları Şekil 3.15' de verilmiştir.

```
model = Sequential()

# 3x3 boyutunda 32 adet filtreden oluşan ReLU aktivasyonlu CONV katmanı eklendi.
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))

# 2x2 boyutlu çerçeveden oluşan Maxpool katmanı eklendi.
model.add(MaxPooling2D(pool_size=(2, 2)))

# Dropout işlemi
model.add(Dropout(0.25))

# Tam bağlantılı katmanına geçiş için düzleştirme yapıldı.
model.add(Flatten())

# 120 nörondan oluşan ReLU aktivasyonlu gizli katman eklendi.
model.add(Dense(120, activation='relu'))

# Dropout işlemi
model.add(Dropout(0.5))

# Çıkış katmanına sınıf sayısı kadar Softmax aktivasyonlu nöron eklendi.
model.add(Dense(num_classes, activation='softmax'))
```

Şekil 3.15. Modelin tanımlanmasını gösteren kod satırları

Şekil 3.15’ deki kod satırlarıyla oluşturulan model; evrişim katmanı, maksimum ortaklama katmanı, seyreltme katmanı, gizli katman ve çıkış katmanından oluşmaktadır. Evrişim katmanı, 32 adet 3x3’ lük filtreden oluşmaktadır ve aktivasyon fonksiyonu ReLU’ dur. Maksimum ortaklama katmanında ise 2x2’ lik ortaklama işlemi yapılır ve bu katmanda ağırlık parametresi hesaplanmaz. Seyreltme katmanı, hesaplanmış ağırlık parametrelerinin bir kısmının silinmesini sağlar. Bu silme işlemi genellikle 0.25 – 0.5 olarak iki kez yapılmıştır. Gizli katmanda 120 adet nöron bulunmaktadır ve aktivasyon fonksiyonu ReLU olarak seçilmiştir. Çıkış katmanında ise 5 adet nöron olup softmax aktivasyon fonksiyonu kullanılmaktadır. Oluşturulan bu derin öğrenme modelinin görselleştirilmiş hali Şekil 3.16’ da verilmiştir.



Şekil 3.16. Kullanılan derin öğrenme modeli

Model tanımlama işleminin ardından oluşturulan modelin eğitimi için kullanılacak yitim fonksiyonu ve optimizasyon algoritması belirlenerek eğitim ve test işlemi gerçekleştirilmiştir. Oluşturulan modelin eğitimi için gerekli kod satırları Şekil 3.17’de verilmiştir.

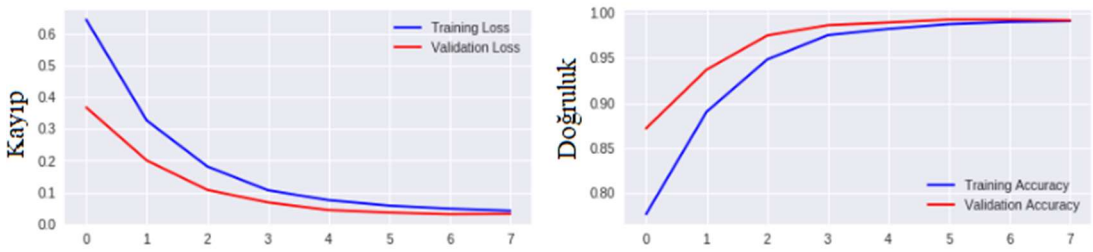
```
# Loss fonksiyonu ve optimizasyon algoritmasını belirleme.
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])
```

```
# Model eğitimi
hist=model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
```

Şekil 3.17. Modelin eğitimi için kullanılan kod satırları

3.5.1. Eğitim sonuçları

Modelin eğitimi sonucunda elde edilen eğitim ve test işlemlerindeki başarım oranı %98 civarındadır. Eğitim ve doğrulama işlemlerindeki doğruluk ve kayıp grafikleri Şekil 3.18' deki gibidir.



Şekil 3.18. Eğitim ve test işlemlerinin sonucunu gösteren grafik

3.5.2. Eğitilen modelin test edilmesi

Eğitilen sinir ağı modelini araçta kullanılabilmek için eğitim sonunda hesaplanan parametreler ".h5" dosya formatında bilgisayara kaydedilir. Kaydedilen bu dosya Python programlama dilinde keras, tensorflow ve h5py kütüphaneleri ile kolayca okunabilmektedir. Bu çalışmada eğitilen sinir ağı "model.h5" isimli dosya olarak kaydedilip Raspberry Pi' a aktarılmıştır. Eğitilen modelin kullanımını gösteren örnek kod satırı Şekil 3.19' da gösterilmiştir.

```
egitilen_model=load_model('model.h5')
giris=goruntu.reshape(1,120,120,1)
cikis=egitilen_model.predict(giris)
cikis_5=egitilen_model.predict_proba(giris)
print(cikis,cikis_5)
```

```
[4] [[1.0681881e-06 8.2427078e-09 1.0940270e-07 3.5433294e-07 9.9999845e-01]]
```

Şekil 3.19. Eğitilen modelin kullanımına ait kod satırları

Şekil 3.19' da model dosyası bulunduğu klasörden okunarak, bu modele çıkışın dört olması öngörülen görüntü değerleri uygulanmıştır. Bunun sonucunda modelin çıkışında beş nörona ait çıkış değerleri oluşur. Bu değerlerden bire en yakın olan sınıf, aracın hangi açıda hareket edeceğini belirten değeri verir. Burada beşinci nöronun çıkışı bire en yakın olduğu için, bu modelin girişe göre çıktısı dört olur. Böylece modelin beklenen değeri verdiği gözlemlenmiştir.

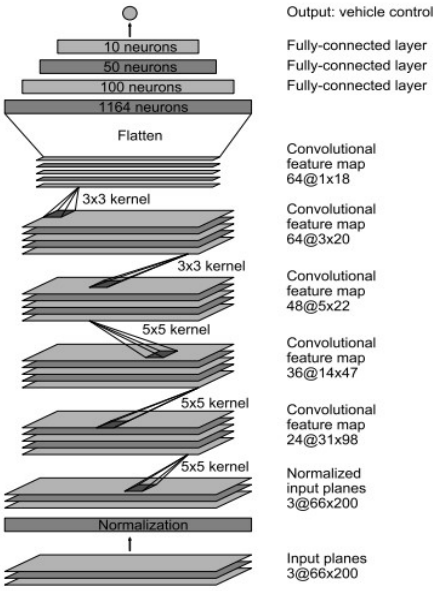
4.ARAŞTIRMA BULGULARI VE TARTIŞMA

Otonom araç kontrolünün geliştirilmesinde, farklı problemlere göre farklı çözümler üretilebilmektedir. Bu yöntemler aracın hareket ettirileceği yere göre değişmektedir. Otonom araçlarda genel olarak bilgisayarlı görü kullanılmaktadır. Bunun yanı sıra farklı sensör ve konum algılayıcılarla yapılan otonom araçlar da bulunmaktadır.

Bilgisayarlı görü kullanılarak geliştirilen sistemlerde, kameradan alınan görüntüler ile şeritlerin dönme açısı hesaplanarak kontrol yapılabilir. Bu yöntemde birçok görüntü işleme algoritması kullanılarak işlem yapılır.

Görüntü işlemeden yararlanılarak gerçekleştirilen sistemlerde ise kameradan alınan görüntüler önce işlenir daha sonra eğitilen sinir ağı modeline uygulanır. Bu sayede öğrenilmiş değerler ile aracın otonom kontrolü sağlanır.

Otonom araçlar için Nvidia tarafından sanal ortamda geliştirilen derin öğrenme modeli Şekil 4.1' de verilmiştir.



Şekil 4.1. Nvidia tarafından geliştirilen derin öğrenme modeli

Bu projede kullanılan yapay sinir ağı modeli, tasarlanan otonom araç için yeterli performansı gösterdi. Ancak kamera, algılaması gereken yolu düzgün algılayamadığında ya da çalışmadığında geliştirilen bu kontrol mekanizması aksaklık gösterebilir. Bunun yanı sıra elde edilen veri seti yeterli olmayabilir ve farklı bir ortama göre veri elde edilmesi gerekebilir.

5. SONUÇLAR VE ÖNERİLER

Eğitilen sinir ağı modeli ile araç iki şerit arasında, belli kıvrımları olan yolda otonom olarak hareket ettirilmiştir. Otonom aracın kontrolünü sağlamak için eğitilen yapay sinir ağı modeli 18760 adet etiketli veri kullanılarak eğitilmiştir. Eğitilen modelin başarı oranı yaklaşık olarak %98' dir. Yapay sinir ağı modelinde etiketli veri sayısı ne kadar artarsa modelin başarım oranı da belli oranda artmaktadır. Ayrıca kameradan alınan görüntünün boyutu arttıkça, az da olsa sinir ağının başarım oranı artmaktadır. Fakat görüntü boyutu büyüdükçe modelin eğitimi daha uzun sürmekte ve hesaplamalar daha fazla zaman almaktadır. Hesaplamaların fazla zaman alması da saniye başına düşen kare sayısını düşürmektedir. Otonom aracın az hatayla hareket etmesi için kare sayının önemi yüksektir. Eğer araç çok hızlı gittiği durumlarda fazla görüntü yakalayamazsa yoldan çıkma ihtimali yüksektir.

Gerçekleştirilen bu otonom araç projesinde temel olarak yapay sinir ağları kullanılmış ve derin öğrenme gerçekleştirilmiştir. Otonom araca konum ve mesafe belirlemeye yardımcı sensörler eklenerek aracın kontrolü iyileştirilebilir ve ek özellikler kazandırılabilir. Ayrıca Raspberry Pi yerine derin öğrenme modelleri için üretilen kartlar kullanılarak daha yüksek performans elde edilebilir.

Bir yapay sinir ağı modeli, insan beyni gibi karmaşıktır ve eğitmek için birçok veriye ihtiyaç duyar. Gerekli verinin toplanabilmesi için çok fazla zamana ihtiyaç vardır. Eğitim verisi toplamak için farklı metotlar belirlenip, daha kısa sürede veri elde etmeye önem gösterilmelidir.

KAYNAKLAR

Goodfellow, I., 2016, Deep Learning [online], MIT Press, http://deeplearningbook.org/ [Ziyaret Tarihi: 10 Ocak 2019].

Karpathy, A., 2018, CS231n Convolutional Neural Networks for Visual Recognition [online], http://cs231n.stanford.edu/ [Ziyaret Tarihi: 16 Ekim 2018].

Kızrak, A., 2018, Derine Daha Derine: Evrişimli Sinir Ağları [online], https:// medium.com /deep-learning-turkiye/derine-daha-derine-evrişimli-sinir-ağları-2813a2c8b2a9 [Ziyaret Tarihi:16 Ekim 2018].

Brownlee, J., 2016, Develop Your First Neural Network in Python With Keras Step-By Step [online], https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/ [Ziyaret Tarihi: 15 Kasım 2018].

Ronaghan, S., 2018, Deep Learning: Which Loss and Activation Functions should I use? [online],https://towardsdatascience.com/deep-learning-which-loss-and-activation -functions- should-i-use-ac02f1c56aa8 [Ziyaret Tarihi: 28 Kasım 2018].

Çarkacı, N., 2018, Derin Öğrenme Uygulamalarında En Sık Kullanılan Hiper Parametreler [online], https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4 [Ziyaret Tarihi: 4 Ocak 2018].

Anonim, 2018, Görüntü İşleme Tekniklerinde Yapay Zeka Kullanımı [online], https://medium.com/türkiye/görüntü-işleme-tekniklerinde-yapay-zeka-kullanımı-24101616cc97 [Ziyaret Tarihi:2 Kasım 2018].

Wang, Z., 2016, Self Driving RC Car [online], https://zhengludwig.Wordpress. com/projects/self-driving-rc-car/ [Ziyaret Tarihi: 7 Ekim 2018].

Ungurean, D., 2018, An Autonomous Car Model, Master's Thesis, Czech Technical University Faculty of Information Technology, Prague, 32-40.

Bruce, W., ; Otter, E. V., 2016, Artificial Neural Network Autonomous Vehicle , License Thesis, KTH Royal Institute of Technology, Stockholm, 23-24.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Feyyaz YAVAŞ
Uyruğu	**:**	T.C
Doğum Yeri ve Tarihi	**:**	Yatağan
Telefon	**:**	0551 404 21 26
e-mail	**:**	feyyazyavas@gmail.com

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Yatağan Mesleki ve Teknik Eğitim Merkezi, Yatağan, Muğla	2013
Üniversite	:	Selçuk Üniversitesi Elektrik Elektronik Mühendisliği, Selçuklu, Konya	

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2012	TKİ Yatağan Müdürlüğü	Stajyer
2018	Konsis Bilişim Hizmetleri	Stajyer

YABANCI DİLLER

İngilizce