T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ARAÇLAR ARASI HABERLEŞME KONTROLLÜ BİREYSEL TAŞIT SİSTEMİ

YÜKSEK LİSANS TEZİ

CİHAT DURMUŞ

DENİZLİ, EKİM - 2018

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ARAÇLAR ARASI HABERLEŞME KONTROLLÜ BİREYSEL TAŞIT SİSTEMİ

YÜKSEK LİSANS TEZİ

CİHAT DURMUŞ

DENİZLİ, EKİM - 2018

KABUL VE ONAY SAYFASI

CİHAT DURMUŞ tarafından hazırlanan "**ARAÇLAR ARASI HABERLEŞMELİ BİREYSEL TAŞIT SİSTEMİ**" adlı tez çalışmasının savunma sınavı 02.11.2018 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği ile Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Prof. Dr. Abdullah Tahsin TOLA	.....
Üye Doç. Dr. Selim ÖNCÜ	.....
Üye Dr. Öğr. Ü. H. Hilal EZERCAN KAYIR	.....

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 26/12/2018 tarih ve ...54/12... sayılı kararıyla onaylanmıştır.

.....

Prof. Dr. Uğur YÜCEL

Fen Bilimleri Enstitüsü Müdürü

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

CİHAT DURMUŞ

Cihat

ÖZET

ARAÇLAR ARASI HABERLEŞME KONTROLLÜ BİREYSEL TAŞIT SİSTEMİ
YÜKSEK LİSANS TEZİ
CİHAT DURMUŞ
PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI
(TEZ DANIŞMANI:PROF. DR. ABDULLAH TAHSİN TOLA)

DENİZLİ, EKİM - 2018

Three RC model cars are designed, the software are developed and the proposed algorithms are tested on the designed platform.

Günümüzde trafik kazalarının pek çoğu insan hatasına bağlı olarak yaşanmaktadır. Bu yüzden şoförü insan olmayan otonom araçlar üzerine çalışmalar yapılmaktadır. İnsansız araç teknolojisi çok hızlı bir şekilde gelişim göstermektedir. İlerleyen on yıllarda günümüzde kullandığımız insanlı araçların yerini insansız araçların alacağı düşünülmektedir. Yapılan tez çalışmasında yakın zamanda önerilen bir trafik sistemi olan MetroCar Sistemi üç adet model araba ile test edilmiştir. Tüm araçların sürücüsüz olduğu ve kendi aralarında bir haberleşme ağı üzerinden haberleştikleri düşünülmüştür. Her araç diğer araçların hız ve konum bilgilerini bilmekte ve bu şekilde kendi hızını güncellemektedir. MetroCar Sisteminde araçlar temel iki algoritma olan takip algoritması ve birleşme algoritmasına göre hareket etmektedirler. Aynı yolda arka arkaya hareket eden araçlar takip algoritmasına göre güvenli mesafeyi koruyarak hareket etmektedirler. Ana yola girmek için yan yoldan gelen araçlar, birleşme algoritmasına göre ana yoldaki arabaları aralarında güvenli boşluk oluşturmaya zorlamaktadır. Bu sistemde herhangi bir trafik ışığına gerek duyulmamış, arabalar kaza yapmadan sistemde hareket etmişlerdir. Üç adet model RC araba tasarlanmış, yazılım geliştirilmiş ve önerilen algoritmalar tasarlanan platformda test edilmiştir.

ANAHTAR KELİMELER: Otonom Araçlar, Bireysel Taşıt Sistemi, Akıllı Araçlar, Trafik Planlama

ABSTRACT

VEHICLE COMMUNICATION CONTROLLED INIVIDUAL VEHİCLE SYSTEM

MSC THESIS

CİHAT DURMUŞ

PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE
DEPARTMENT OF ELECTRİCAL AND ELECTRONİCS ENGİNEERİNG

(SUPERVISOR:PROF. DR. ABDULLAH TAHSİN TOLA)

DENİZLİ, OCTOBER 2018

Today, most of traffic accidents are experienced due to human error. Therefore, studies have been made on autonomous vehicles which driver is not a human. Unmanned vehicle technology is developing very quickly. In the following decades, the manned vehicles we use today are thought to be replaced by unmanned vehicles. In this thesis study, MetroCar System, a recently proposed traffic system, is tested using three model cars. It is considered that all vehicles are driverless and communicate with each other via a communication network. Each vehicle knows the speed and location of the other vehicles; therefore it updates their speed. In MetroCar System, the vehicles act according to the two basic algorithms, the tracking algorithm and the merging algorithm. The vehicles moving in a row on the same road follow the safe distance according to the tracking algorithm. The vehicles coming from the side road to enter the main road force the main road's vehicles to have safe gap to merge following the merging algorithm. In this system, no traffic light was required, and the vehicles transport without any accident. Three RC model cars are designed, the software are developed and the proposed algorithms are tested on the designed platform.

KEYWORDS: MetroCar System, Autonomous Vehicles, Intelligent Vehicles, Traffic Planning

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

ÖNSÖZ

Bu proje çalışması boyunca beni yönlendiren, karşılaştığım tüm zorlukları tecrübe ve bilgileri ile aşmama yardım eden hiçbir şekilde desteğini esirgemeyen değerli danışman hocam Prof. Dr. Abdullah Tahsin TOLA ’ya teşekkür ederim.

Yürüttüğüm proje çalışması boyunca benden desteğini esirgemeyen, bana moral veren aileme ayrıca teşekkür ederim.

1. GİRİŞ

1.1 Konu Özeti

Günümüzde trafik kazalarının %90'lık kısmı insan hatasına bağlı olarak yaşanmaktadır. Bu yüzden araçlar üzerine yapılan çalışmalar, insan iradesinin yerine sürücüsü olmayan kendi sürüş algoritmasına göre hareket eden araç tasarımı üzerine yoğunlaşmaktadır. İnsansız araç teknolojisi çok hızlı bir şekilde gelişim göstermektedir. İlerleyen yıllarda günümüzde kullandığımız insanlı araçların yerini insansız araçların alacağı düşünülmektedir.

Genel olarak MetroCar Sistem olarak tanımladığımız ve temel olarak araçlar arası bir haberleşme algoritmasına göre çalışan sistemimizi, toplu taşıma araçlarının ve özel araçların avantajları, dezavantajları ve günümüzde yapılan ve yapılması planlanan insansız araçların getirmiş olduğu güvenlik sorunlarını göz önüne alarak tasarladık.

Toplu taşıma araçları ve özel araçlara sırayla değinecek olursak; toplu taşıma araçları özel araçları göre trafikte daha az dur-kalk yaparak seyahat etmektedir. Buna bağlı olarak daha az strese maruz kalınmaktadır. Ayrıca toplu taşıma araçlarını kullanmak park yeri arama sorununu da ortadan kaldırmaktadır. Metro gibi toplu taşıma araçlarını kullanmak ise daha az trafiğe maruz kalınmasını ve daha az zaman kaybedilmesini sağlamaktadır. Bunların aksine toplu taşımanın getirmiş olduğu kalabalık bir araçla seyahat sorunu ve buna bağlı olarak gelişen konfor, hijyen, güvenlik, kötü kokular ve rahatsız edici davranış sorunlarına maruz kalınmaktadır.

Özel araçlar ise konfor, hijyen, güvenlik, yalnız seyahat etme imkanı ve oturarak seyahat imkanı gibi avantajlar sağlamaktadır. Bunun yanı sıra trafikte dur-kalk yapılması, zaman kaybı, stres, park yeri sorunu ve yakıt masrafı gibi dezavantajlara sahiptir.

MetroCar ile amaçlanan toplu taşıma ve özel araçların avantajları bir araya getirmektir. Bunu yaparken de güvenlik sorununu önemli bir parametre olarak alıp, yüksek güvenlikli bir sistem oluşturulması amaçlanmıştır. MetroCar Sistem öncelikle metro gibi kendine özel bir yola sahiptir. Ayrı bir yol oluşturulmasının temel sebebi sistemin öncelikle yüksek güvenlikte olmasını sağlamaktır. Bunun yanı sıra hızlı bir trafik akışı sağlanması, trafikte geçirilen zamanın verimli kullanılması, hijyen, konfor ve zaman kazancı bu trafik organizasyonunun önemli parametreleri olmuştur. Ayrıca geliştirilen insansız araçların trafiğe uyumunda insan ve hayvan faktörlerini göz önüne aldığımızda bazı sorunların yaşanması tahmin edilmektedir. MetroCar sistemle oluşturulan trafik düzeni içerisinde tüm araçlar otonom ve kendi aralarında veri alışverişi yapabilecekleri bir haberleşme ağına bağlıdırlar. Araçlar için tasarlanan özel yola sadece aynı tip birbiriyle haberleşebilen araçlar girebilmektedir. Araçlar bu yola girdiklerinde tam otonom olarak çalışabilmektedir. Ayrıca araçlar elektrikli araçlar olarak tasarlanmıştır.

1.2 Tezin Literatür Bilgisi

Sürücüsüz araç çalışmalarının başlangıcı General Motors'un 1939 yılında düzenlediği Futurama Dünya fuarı olduğu kabul edilmektedir. General Motors tarafından tanıtılan araç, şehir içi trafiğinde güvenli bir şekilde seyahat eden otomatik bir araç olarak tanıtılmıştır. (Özgüner 2011)

Mercedes-Benz mühendisi Ernst Dickmans 1980 yılında kameradan aldığı görüntü ile sürücüye ihtiyaç duymadan hareket eden araç tasarlamıştır. Ernst Dickmans yaptığı bu çalışma ve elde ettiği başarı sayesinde Avrupa Komisyonu'ndan 1987-1995 yılları arasında 800 milyon Avro destek almıştır. Bu destek ve çalışma otonom araç teknolojisinin geleceği yön vereceği kanıtlamıştır.(Özgüner 2011)

DARPA'nın desteklemiş olduğu 1987 yılında Amerika'da gerçekleştirilen insansız araç projesinde, kamera görüntüleri, lazer radarı ve robot kontrol yapıları kullanılmıştır. Arazi yapısında test sürüşü gerçekleştirilen araç 30 km/h hıza ulaşmıştır. (WEB_2)

Carnegie Mellon Üniversitesi 1995 yılında Navlab isimli insansız araç projesi gerçekleştirmiştir. Yapılan test sürüşünde araç %98.2 verimle 5000 km yol yapmıştır. Bu yıl içerisinde Dickmanns Mercedes-Benz aracıyla Münih ile Kopenhag arasında seyahat etmeyi başarmıştır.Dickmanns bu araçla 1600 km yolda 175 km/h'lik hıza ve %95 otomatik sürüşe ulaşmıştır.

ABD Savunma Bakanlığı insansız araçlar için 2002 yılında Grand Challange adlı bir yarış düzenlemiştir. Bu yarışı bitiren bir araç olmamıştır. 2005 yılında düzenlenen yarışta 217 km'lik yarış pistini 5 araç tamamlamıştır. 2007 yılında yapılan yarışta ise 6 araç bitiş çizgisine ulaşmış ve toplam ödülü paylaşmışlardır.

Parma Üniversitesi'nde Alberto Broggi tarafından yapılan çalışma sonrasında geliştirilen 4 adet insansız aracı 2010 yılında yapılan Şangay Expo'ya, İtalya'dan Çin'e 13.000 kilometre yol yaparak götürmeyi başarmıştır. Alberto Broggi'nin geliştirmiş olduğu araçlar elektrikli araçlardır. (Forret 2007)

2001 yılında Paruchuri ve arkadaşları araçların herhangi bir kontrol merkezine ihtiyaç duymadan düzensiz bir kavşak noktasından geçebilmeleri için çok etmenli simülasyon çalışması gerçekleştirmiştir. (Paruchuri 2001)

2001'de Wahle ve arkadaşları yürüttükleri çalışmada, kullanıcıya uygun yol güzergahlarını bulmak, trafikte oluşan sıkışıklığı minimuma indirmek ve trafik altyapısını verimli kullanabilmek için sürücü bilgi sistemini (ATIS) geliştirmişlerdir. Oluşturulan online simülasyonlar ile mevcut trafik bilgisi kullanılarak seyahat süreleri hesaplanmakta, sürücünün kişisel tercihleri göz önünde bulundurularak trafik yüküne göre en uygun trafik güzergahının belirlenmesi düşünülmektedir. (Wahle 2001)

Başkaya ve Öztürk dal-kesme metodunu kullanarak bir ekmek fabrikasının 5 satış şubesi için dağıtım problemini çözmek için çalışmışlardır. (Başkaya 2005)

Chong ve arkadaşları 2006 yılında trafik sorunu üzerine akıllı trafik konulu çalışma yapmışlardır. Yapılan çalışmada akıllı ulaşım organizasyonu için karar mekanizmalarının ve bilgi analizlerinin önemli olduğuna değinilmiştir. Karar mekanizmaları ve bilgi analizlerindeki eksikleri göz önünde bulundurarak grid bilgi işleme ve veri madenciliği adı altında yeni analiz sistemi önerilmiştir. Böylece sistem

içerisinde faydalı verileri işlemek daha kolay olmaktadır. Kullanılan sistem elde ettiği yararlı veriler sayesinde tavsiyelerde bulunabilmekteydi. Trafik verileri analizleri sonucunda oluşturulan sistemin trafik bilgi paylaşımındaki verimi artıracağı sonucuna ulaşılmıştır. (Chong 2006)

Lin ve arkadaşları 2009 yılında yaptıkları çalışmayla aynı yerden harekete geçen ve aynı yerde hareketini tamamlayan sürücülerin bu seyahatleri boyunca genellikle farklı güzergâhları tercih ettiği sonucuna ulaşmışlardır. Lin ve arkadaşları gerçekleştirdikleri çalışmada, sürücülerin güzergâh seçimindeki yaklaşımlarını da göz önüne alarak sürücülere yolları daha detaylı anlatan bir model geliştirmişlerdir. Bulanık nöral kılavuz sistemi adı verilen çalışmada kullanıcıların trafikteki önceki tutumları ve kararlarından yararlanılmıştır. Sistem ANFIS ile sürücüye gerek duymadan kendisini ayarlamasını sağlamıştır. Bu modellemeyle farklı kullanıcıların farklı tercihlerine göre gidilebilecek iyi güzergahları otomatik olarak sürücüye sunan bir sistem oluşturulmuştur. (Lin 2009)

Güvez ve arkadaşları 2012 yılında Kırıkkale ilinde tıbbi atıkları toplamak için en uygun rotayı belirleme problemini 0-1 tamsayılı programlama modelini kullanarak bulmaya çalışmışlardır. (Güvez 2012)

Güney Kore, otonom araçların test edilebilmesi için K-city adında özel bir şehir tasarladı. Projenin tamamlanarak hayata geçmesiyle birlikte dünya üzerinde sürücüsüz araçların test edilebilmesi için şuana kadar yapılan en büyük test alanı yapılmıştır. Ekim ayı içerisinde tamamlanan projeyle sürücüsüz araç şehri projesi hayata geçirilmiştir. Otonom araçların gerçek yollarda yaptığı testlerde meydana gelen kazalar neticesinde, otonom araçların test edilirken trafikte oluşturacağı riskleri ortadan kaldırıp güvenli bir ortamda araçların testlerini yapması sağlanmıştır. K-City şehri 17 milyon dolara mal olmuştur. Burada sürücüsüz araçlar trafiğe çıkmadan önce, gerçek trafikte karşılaşabilecekleri tüm senaryolarla karşılaşacak ve testler buna göre yapılacaktır. Bu da bu şehri sürücüsüz araçlar için çok önemli kılmaktadır. (Tunçer, 2017)

18 Mart 2018 yılında Uber'in sürücüsüz aracının ABD'nin Arizona eyaletinde karışmış olduğu bir kaza sonucunda Elaine Herzberg adlı yaya hayatını kaybetmiştir. Uber'in Volvo SC 90 otonom aracında kaza anında direksiyonda bir

kişi olmasına rağmen aracın otomatik modda seyahat ettiği tespit edilmiştir. Direksiyon başındaki kişinin yola değil telefon olduğu düşünülen bir şeye baktığı anlaşılmıştır. Yapılan araştırmada yetkililer yayanın yaya geçidi dışında bir noktada karşıdan karşıya geçtiğini belirlemiştir. Uber kaza sonrası tüm sürücüsüz araçlarının testlerini durdurmuştur. (Congar, 2018)

2018 yılı içerisinde Temsa Nvidia Europe GTC etkinliğinde MD9 ELECTRICITY elektrikli ve sürücüsüz aracını sergiledi. TEMSA 2022 yılında bu aracın seri üretimine başlamayı planlıyor.

TEMSA'nın teknolojik gelişiminin önemli bir aşaması olan otonom araçlar için yürütülen çalışmalar şirketin Adana'daki tesislerinde 2 yıldır devam etmektedir. Yapılan çalışmalarla geliştirilen otonom aracın ilk versiyonu 9-11 Ekim tarihlerinde Münih'te düzenlenen Nvidia Europe GTC etkinliğinde tanıtıldı. Aracın ilk versiyonu LIDAR, radarlar ve kameralarla çevresindeki nesneleri algılayabilmektedir. (Öğretmenoğlu, 2018)

Ford otomobil markası 2018 yılı içerisinde tanıtımını yaptığı insansız araç projesiyle trafik ışıklarında bekleme problemine çözüm bulmaya çalışmıştır. Ford firması yaptığı tanıtımla sürücülerin yılda 2 gününü trafikte geçirdiğine dikkat çekmiştir. Bunun yanında araç çarpışmalarının %60'dan daha fazlasının kavşak noktalarında meydana geldiği belirtilmiştir. Ford meydana gelen kazaların önüne geçmek ve trafik ışıklarında yapılan durma ve kalkmaları azaltmak için geliştirdiği otomobilinde araçlar arasında bir haberleşme ağı kurmuştur. Araçlar diğer araçlarla hız, konum ve hareketi yönü bilgilerini paylaşmaktadır. Kullanılan algoritmayla hangi aracın kavşak noktasına önce gireceği belirlenmektedir. Araçların kavşak noktasındaki sıralamaları belirlendikten sonra, çarpışma olmaması için araçların hızları artırılıp azaltılarak kavşak noktasında durmadan sırayla geçiş sağlanmaktadır.(Göküş,2018)

Nisan 2018'de Google firmasının otonom araçlarından Waymo Amerika'nın Arizona eyaletinde bir kazaya karışmıştır. Polisin yaptığı incelemeler neticesinde otonom aracın kazada herhangi bir suçunun olmadığı tespit edilmiştir. Waymo'ya çarpan aracın başka bir araca çarpmamak için yaptığı manevrayla Waymo otonom aracına çarptığı anlaşılmıştır. (Kundu, Mayıs 2018)

1.3 Tez Tanıtımı

Gerçekleştirilen yüksek lisans tez çalışmasında öncelikle yapılmak istenen tezin anlatıldığı konu özeti ve literatür bilgisi anlatılmıştır. İkinci bölümde tez çalışmasında kullanılan malzeme ve ekipmanlar hakkında genel bilgiler verilmiştir.. Burada DC motor kavramı, DC motor sürme teknikleri, araçlarda kullanılan lityum iyon bataryalar, sistemin genel kontrolünü sağlayan işlemci olan STM32F103C8T6 işlemcisinin genel özellikleri ve sistemin haberleşme yapısını oluşturan NRF24L01 WIRELESS modülü ele alınmıştır. Üçüncü konu olan donanım başlığı altında uygulaması yapılan sistemin çalışmasını sağlayan devreler ve malzemeler anlatılmıştır. Dördüncü konu olan yazılım kısmında ise, araçların çalışma diyagramı, sistemin çalışma diyagramı, araçların güvenli seyahatleri için çok önemli olan takip ve birleşme algoritmaları, araçların genel çalışma durumlarını kontrol etmek ve araçların sisteme girmesini sağlayan C# arayüz programı, araçların hız ölçümünün nasıl yapıldığı ve tezin uygulaması konuları ele alınmıştır. Beşinci bölümde, sistemin genel çalışma durumu, ileride geliştirilmesi gereken yanları ve gelecekte yapılması planlanan çalışmalar hakkında bilgiler sunulmaktadır.

2. TEMEL KAVRAMLAR

2.1 DC Motorlar

2.1.1 DC Motor Temel Prensibi

Manyetik alan içindeki bir iletkene enerji verilirse, iletkenden akım akacak ve manyetik alan manyetik kuvvet üretecektir. İletkende bir kuvvet oluşturmak için gerekli iki şart vardır. İletkenden akım geçmeli ve iletken manyetik alan içinde olmalıdır. Bu temel şartlar gerçekleştiğinde iletkene bir kuvvet uygulanacak ve bu kuvvet manyetik alana dik doğrultuda iletkeni hareket ettirmeye çalışacaktır. Bu tüm DC motorların çalışma prensibidir. Aşağıda Denklem 1.1'de iletken üzerinde oluşan kuvvet ifadesi gösterilmektedir.

$$F = i * l \times B \text{ (Newton)} \qquad (1.1)$$

Burada ;

B: Manyetik alan yoğunluğu

l: İletken Uzunluğu

i: İletkenden geçen akımı ifade etmektedir.

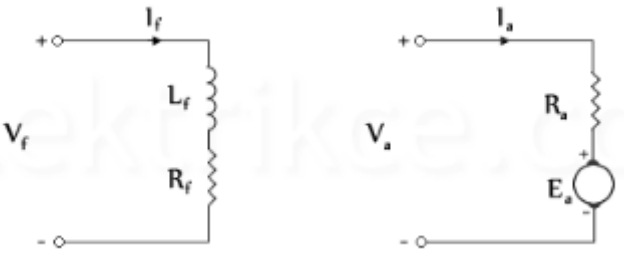
2.1.2 DC Motor Çalışma Prensibi

Akı yoğunluğu B olan manyetik alan içindeki bir bobini düşünelim. Bobinin iki ucu DC voltaj kaynağına bağlanmış ve bobinden akmaktadır. Manyetik alan ve elektrik akımının bir sonucu olarak bobin üzerine bir kuvvet uygulanır. Bobinin iki tarafındaki bu kuvvet bobini kuvvet yönünde döndürmeye başlar.

Gerçek bir DC motorda rotor üzerinde birden fazla bobin vardır. Hepsinde de dönmeyi sağlayacak kuvvet oluşur. Teldeki akım ve manyetik alan ne kadar büyük olursa o kadar hızlı hareket eder çünkü o kadar büyük bir kuvvet oluşur. Aynı zamanda manyetik alandaki iletkenin hareketi sonucu tork üretilmiş olur.

2.1.3 DC Motor Eşdeğer Devresi

DC motorun şematik diyagramı Şekil 2.1'de gösterilmiştir. Bir DC motor iki farklı devreye sahiptir: Alan devresi ve armatür devresi. Giriş elektriksel güç çıkış ise mekanik güçtür. Bu eşdeğer devrede alan sargısı ayrı bir DC voltaj olan V_f voltaj kaynağı ile beslenmektedir. R_f ve L_f alan sargısındaki direnç ve bobini temsil eder. Sargıda üretilen I_f akımı motorun çalışması için gerekli olan manyetik alanı oluşturur. Armatür devresinde V_t motora uygulanan gerilimdir, armatür devresinde akan akım I_a, armatür sargısının direnci R_a ve E_b armatürde indüklenen toplam gerilimi ifade etmektedir.



Şekil 2. 1 : DC Motor Eşdeğer Devresi. Kaynak : https://www.elektrikce.com

Gerilim denklemleri

Armatür devresinden :

$$V_t = E_b + I_a * R_a \tag{1.2}$$

Alan devresinden :

$$V_f = R_f * I_f \quad (1.3)$$

V_f alan sargısına uygulanan gerilim (manyetik alan oluşturur), R_f alan sargısının direnci ve I_f alan sargısının akımını ifade etmektedir.

Güç Aktarım Denklemleri :

Motorda oluşan tork ifadesi aşağıdaki şekilde verilir :

$$T_{dev} = K \, \Phi \, I_a \quad (1.4)$$

Geliştirilen güç, mekanik forma dönüştürülen güçtür ve aşağıdaki gibi ifade edilir :

$$P_{dev} = \omega_m * T_{dev} \quad (1.5)$$

Bu güç armatürde indüklenen gerilimden elde edilebilir ve şu şekilde verilir :

$$E_b * I_a(elektriksel\ güç) = \omega_m * T_{dev}\,(\,mekaniksel\ güç) \quad (1.6)$$

Dakikadaki devir hızı N, açısal hız ile ilişkilidir. (saniyede radyan cinsinden)

$$\omega = 2 * \pi * \frac{N}{60} \quad (1.7)$$

Projede kullanılan DC motor Şekil 2.2' de gösterilmiştir.

Şekil 2. 2 : Projede Kullanılan DC Motor

2.2 DC Motor Sürücüleri

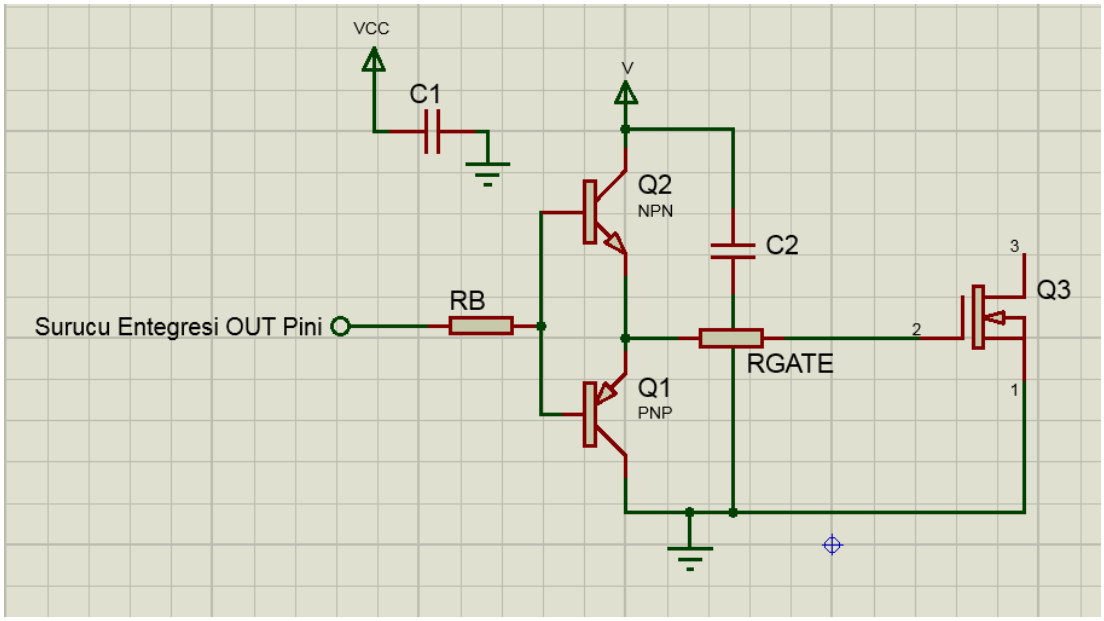
2.2.1 MOSFET Sürme Teknikleri

MOSFET gibi anahtarlama elemanları kullanılırken ideal bir anahtarlama zamanı elde edebilmek için MOSFET' in açılıp kapanma işlemlerini mümkün olduğunca hızlı yapabilmek gereklidir. MOSFET' leri anahtarlamak için MOSFET' in gate ucuna PWM sinyali uygulanır. Uygulanan sinyalin MOSFET' in gate ucuna düzgün gelmesi anahtarlama zamanını etkilemektedir. Bu yüzden kullanılan anahtarlama teknikleri ile MOSFET' in anahtarlama hızını artırmak amaçlanmaktadır. MOSFET' lerin anahtarlama süresinde MOSFET' in gate ucuna uygulanan voltaj değeri çok önemlidir. Bunun yanı sıra MOSFET' i hızlı bir şekilde deşarj etmek anahtarlama hızı için önemli bir diğer faktördür. MOSFET' i anahtarlarken gate ucunu bir kondansatör gibi düşünerek hareket edebiliriz. Kondansatörü hızlı bir şekilde şarj edip deşarj etmeliyiz. Bu yüzden gate ucuna

bağlanan direnç değeri yüksek olmamalıdır. Gate direncinin yüksek olması anahtarlama hızını yavaşlatacaktır.

2.2.2 Bipolar Totem Pole Sürme Tekniği

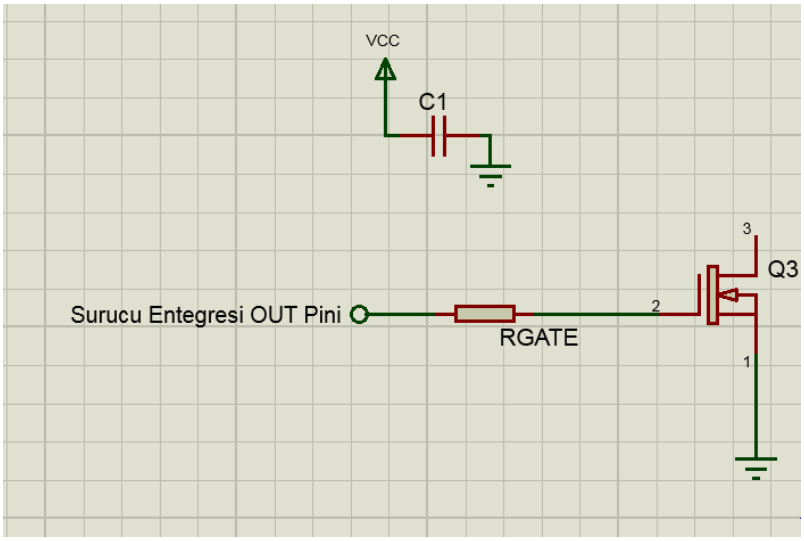
Şekil 2.3'de bipolar totem pole non-inverting sürme tekniği devresi gösterilmiştir. Çok kullanılan sürme tekniklerinde biridir. Entegre çıkışındaki sinyal R_B direnci üzerinden akarak birbirine bağlı PNP ve NPN transistörleri duruma göre anahtarlayarak MOSFET' i iletime ve kesime sokmaktadır. OUT kısmından lojik 1 sinyali geldiğinde akım üst koldan R_{GATE} direnci üzerinden akarak MOSFET' i iletime sokacaktır. OUT kısmından çıkacak lojik 0 sinyali ise alt koldaki PNP transistörü iletime sokacak ve böylece MOSFET' in gate ucu GND ye bağlanarak MOSFET kesime geçecektir. Bu sürme tekniği ani akım sıçramalarına dayanabilmektedir.



Şekil 2. 3 : Bipolar Totem Pole Sürme Tekniği

Bu teknikte iki bipolar transistör birbirlerini ters gerilim ve ters akıma karşı korumaktadırlar. Entegrenin OUT kısmından çekilen akımın yüksek olmasının istenmediği durumlarda bu teknik önemli avantaj oluşturmaktadır.

2.2.3 PWM Direkt Sürme Tekniği



Şekil 2. 4 : PWM Direkt Sürme

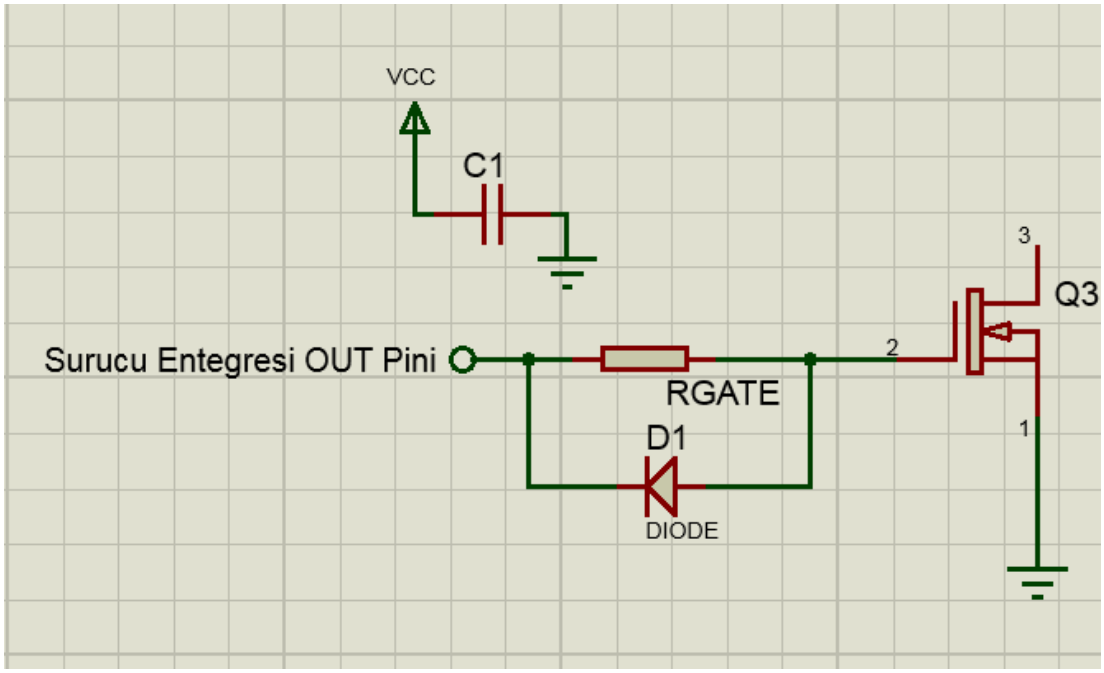
Şekil 2.4'de PWM Direkt Sürme tekniği görülmektedir. Bu teknikte entegrenin GND bacağı ile MOSFET' in source bacağı arasındaki mesafe anahtarlama hızı açısından çok önemlidir. Bu mesafe parazitik endüktans değerini etkilemektedir. Parazitik endüktans değeri olabildiğince düşürülmelidir. Bu da iki uç arasındaki yolların kalınlığı artırılarak ayarlanabilir. Burada entegre çıkışının MOSFET' i doğrudan anahtarlaması bir dezavantaj oluşturmaktadır. MOSFET' in gate ucundaki kapasitansın anahtarlama esnasında istediği akımı entegre karşılayamazsa entegre hasar görecektir. Entegre besleme uçları arasına konulmuş olan kapasitör sayesinde ani çekilen akımların önüne geçilmesi amaçlanmıştır.

2.2.4 Hız Artırma Teknikleri

MOSFET' i hızlı bir şekilde anahtarlamak için bazı devreler tasarlanmaktadır. Tasarlanan devreler sayesinde gate ucundaki kapasitör olabildiğince hızlı şarj edilerek MOSFET' i iletime sokmak ve kapasitörü mümkün olduğunca kısa bir sürede deşarj ederek MOSFET' i kesime sokmak amaçlanmıştır. Tasarlanan devreler ne kadar iyi olursa anahtarlama kayıpları o kadar az olacak ve MOSFET' lerde ısınma ve hasar da o kadar az olacaktır.

2.2.5 Diode TURN OFF Devresi

Şekil 2.5'de Diode TURN OFF devresi gösterilmiştir. Burada temel amaç R_{gate} direncine paralel olarak yerleştirilen D_{OFF} diyotu sayesinde MOSFET deşarj olurken akımın R_{gate} direnci ile karşılaşmadan diyot üzerinden akmasıdır. Burada diyotun toparlanma hızı çok önemli olduğu için diyot seçimi de büyük önem arz etmektedir.



Şekil 2. 5 : Diode TURN OFF Devresi

MOSFET iletime geçerken R_{GATE} direncini kullanmaktadır. MOSFET' in kesime geçtiği sırada diyotun aktif olabilmesi için Denklem (1.8) verilen formülün sağlanması gerekmektedir.

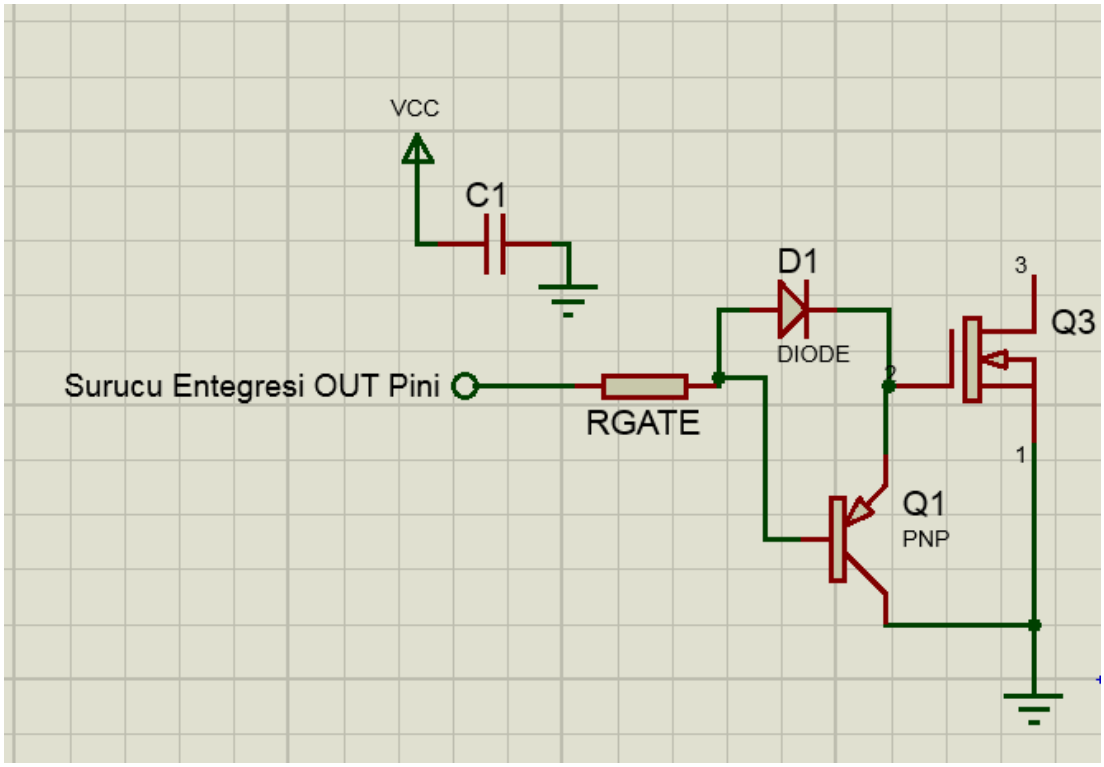
$$I_G > \frac{V_{D,FWD}}{R_{GATE}} \tag{1.8}$$

Burada :

$V_{D,FWD}$ diyotun FORWARD voltaj değeri olarak geçmektedir. Diyot ON olduğu durumda, üstünden akım geçirdiği zaman üzerinde oluşan voltajı ifade etmektedir. Bu teknikte entegrenin çıkış empedansına doğru akımın deşarj edilmesi dezavantaj oluşturmaktadır.

2.2.6 PNP TURN OFF Devresi

Şekil 2.6' da gösterilen PNP TURN OFF devresi, MOSFET' i daha hızlı bir şekilde deşarj edebilmek için kullanılan en popüler yöntemdir.



Şekil 2. 6 : PNP TURN OFF Devresi

Bu teknikle MOSFET' in deşarj edilmesi Q1 transistörü ile yapılarak deşarj süresi önemli ölçüde azaltılmaktadır. Burada şarj sırasında ve deşarj sırasında kullanılan yollar birbirinden farklıdır. Aynı zamanda deşarj sırasında akım herhangi bir dirençle karşılaşmadan daha kısa bir yolla GND ucuna akmaktadır. Kullanılan entegre deşarj sırasında herhangi bir işlem yapmamaktadır. Ayrıca Q1 transistörü deşarj sırasında oluşabilecek yüksek akımları kendi üzerinden akıtabilecektir.

2.3 Lityum İyon Piller

2.4 Lityum İyon Pillerin Genel Özellikleri

Enerji yoğunluğu yüksek olan piller için yapılan araştırmalar sonucu özgül ağırlığı en hafif ve elektronegatifliği en düşük olan lityum metali keşfedilmiştir. Teorik olarak enerji yoğunluğuna bakıldığında lityum metalinin enerji yoğunluğu en

yüksektir. Lityum metali periyodik cetvelde hidrojen ve helyumdan sonra en küçük atomdur. Bundan dolayı lityum metalinin anot-katot arası iyon mobilizasyonu çok kolaydır. Elektron verdiğinde diğer tüm metallerden çok daha yüksek standart potansiyel üretme yeteneğine sahiptir.

Li-on pilleri şarj etmek için pilin tamamen boşalması durumuna gerek yoktur. İstenildiği zaman şarj edilebillir, istenildiği seviyede şarjdan çekilebilir. Li-on bataryalar, kapasitelerini her yıl yüzde 20 ila yüzde 30 arasında kaybederler. Ortalama ömürleri 5 yıldır. Doğrudan güneş ışınlarından ve doğrudan ısıdan korunması gereken bu piller, halihazırda kullanılan en yaygın ve verimli olan pil/batarya türüdür.

2.4.1 Li-on Bataryaların Avantajları

- Kapalı hücrelerdir ve bakım gerektirmezler
- Uzun çevrim ömrüne sahiptirler
- Geniş çalışma sıcaklığı aralığına sahiptirler
- Yüksek hız ve yüksek enerjide deşarj olabilirler
- Yüksek spesifik enerji ve enerji yoğunluğuna sahiptirler
- Yüksek kolombik verim ve yüksek enerji verimliliğine sahiptirler
- Raf ömürleri uzundur
- Hızlı şarj edilebilirler
- Düşük kendiliğinden deşarj olma oranına sahiptirler

2.4.2 Li-on Bataryaların Dezavantajları

- Maliyetleri yüksektir
- Yüksek sıcaklıklarda bozunabilirler
- Koruyucu devre ekipmanlarına ihtiyaç duyarlar
- Gereğinden fazla şarj olmaları halinde kapasite kaybı ve termal kaçaklar olabilir

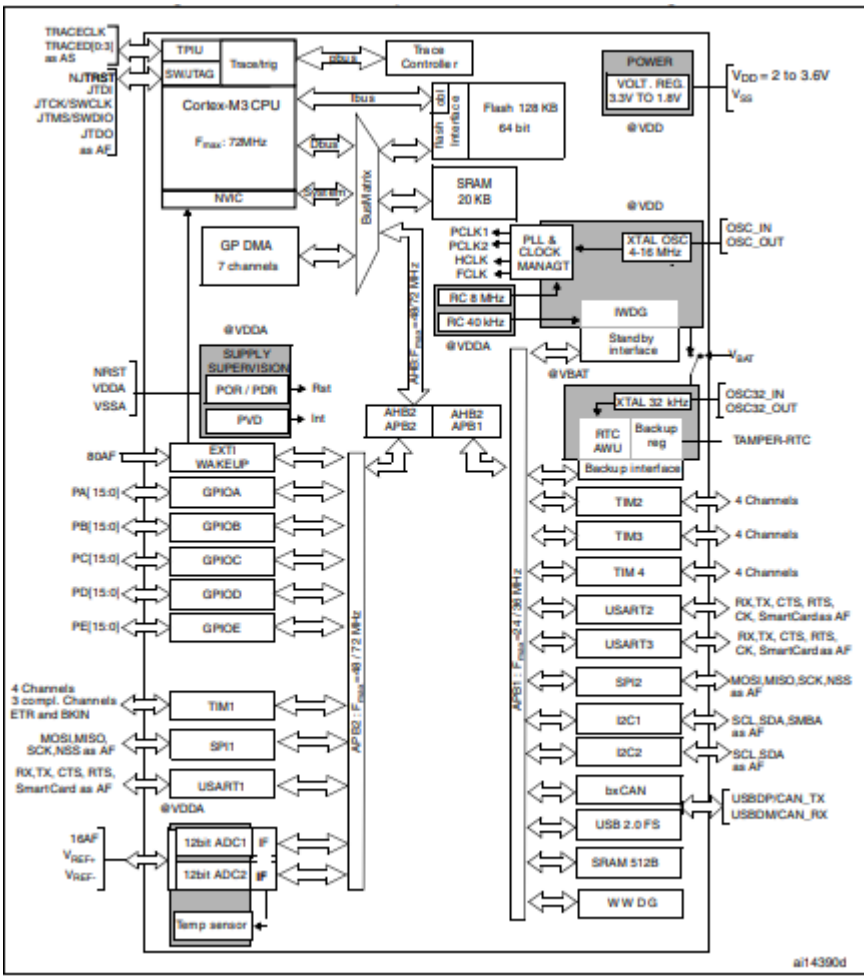
- Çarpma durumunda hava alma ve muhtemel termal kaçaklar olabilir
- Silindirik pillerde, Ni-Cd veya Ni-MH pillerinden daha düşük enerji yoğunluğu olmasıdır.

2.5 STM32F103XX Ailesi

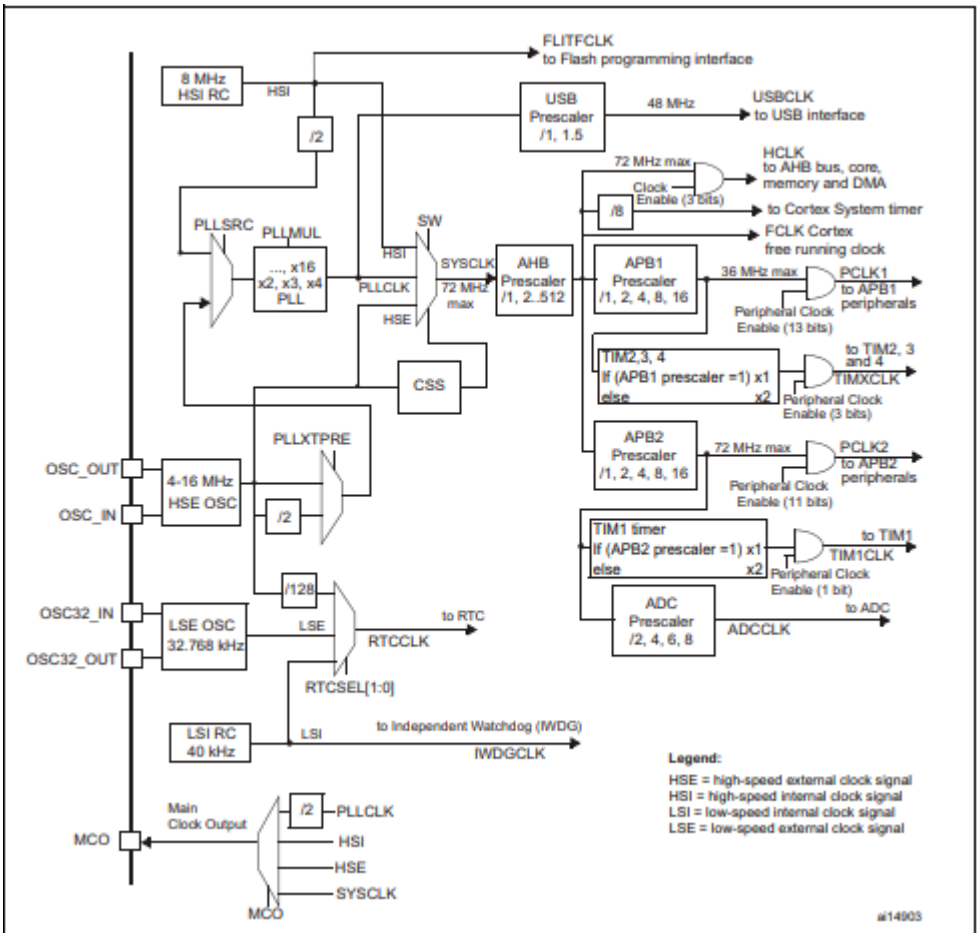
STM32F103xx işlemci ailesi yüksek performansa sahiptir. 72 MHz frekansta çalışan ARM Cortex-M3 32 bit RISC çekirdekli, yüksek hızlı gömülü 128 Kbyte Flash bellek ve 20 Kbyte SRAM, ve iki APB veri yoluna bağlı gelişmiş I/O çevre birimlerine sahiptir. 12-bit ADC, üç genel amaçlı 16-bit zamanlayıcı, bir adet PWM zamanlayıcısı, iki I2C, iki SPI, üç USART, bir USB ve bir CAN çevre birimlerini de sunar. İşlemci -40 derece ile +85 derece arası ve -40 derece ila +105 derece arasında çalışma sıcaklığı aralığına sahip cihazlar sunmaktadır. STM32F103xx ailesi 36 pimden 100 pime kadar değişen altı farklı cihaz içerir. Bu özellikler STM32F103xx ailesini, motor sürücüleri, uygulama kontrolü, tıbbi ve el aletleri, PC oyun ve çevre birimleri, GPS platformları, endüstriyel uygulamalar, PLC'ler, invertörler, yazıcılar, tarayıcılar, alarm sistemleri, görüntülü interkomlar, ısıtma, soğutma, havalandırma ve iklimlendirme uygulamaları için uygun hale getirir. STM32F103xx ailesinin genel özellikleri Şekil 2.7' de, blok diyagramı Şekil 2.8' de ve clock diyagramı Şekil 2.9' da verilmiştir.

Peripheral		STM32F103Tx		STM32F103Cx		STM32F103Rx		STM32F103Vx	
Flash - Kbytes		64	128	64	128	64	128	64	128
SRAM - Kbytes		20		20		20		20	
Timers	General-purpose	3		3		3		3	
	Advanced-control	1		1		1		1	
Communication	SPI	1		2		2		2	
	I^2C	1		2		2		2	
	USART	2		3		3		3	
	USB	1		1		1		1	
	CAN	1		1		1		1	
GPIOs		26		37		51		80	
12-bit synchronized ADC Number of channels		2 10 channels		2 10 channels		2 16 channels(1)		2 16 channels	
CPU frequency		72 MHz							
Operating voltage		2.0 to 3.6 V							
Operating temperatures		Ambient temperatures: -40 to +85 °C / -40 to +105 °C (see *Table 9*) Junction temperature: -40 to + 125 °C (see *Table 9*)							
Packages		VFQFPN36		LQFP48, UFQFPN48		LQFP64, TFBGA64		LQFP100, LFBGA100, UFBGA100	

Şekil 2. 7 : STM32F103xx Genel Özellikleri. Kaynak : https://www.st.com



Şekil 2. 8 : STM32F103xx Blok Şeması. Kaynak : https://www.st.com



Şekil 2. 9 : STM32F103xx Clock Diyagramı. Kaynak : https://www.st.com

2.5.1 ARM Cortex-M3 Çekirdeğinde Flash ve SRAM

ARM Cortex-M3 işlemcisi gömülü ARM işlemcilerinin en yenilerindendir. MCU (Mikroişlemcili Kontrol Ünitesi)'nun ihtiyaçlarını karşılayacak düşük maliyetli bir platformu geliştirilmiştir. Bu işlemci yüksek performans sağlarken düşük güç tüketimiyle avantaj oluşturmaktadır.

ARM Cortex-M3 32 bit RISC işlemcisi, olağanüstü kod verimliliği sunar. Bir ARM işlemciden beklenen yüksek performansı bellek boyutunda 8 bit ve 16 bit cihazlarla ilişkili olarak sağlar. Gömülü ARM çekirdeğe sahip STM32F103xx ailesi tüm ARM araçları ve yazılımı ile uyumludur.

2.5.2 Harici İnterrupt Kontrolörü (EXTI)

STM32F103xx ailesinde 16 adet harici kesme hattı mevcuttur. Harici kesme kontrolörü, kesme isteklerini üretmek için kullanılan 19 kenar detektör hattından oluşur. Her kenar, yükselen kenar, düşen kenar tetiklerini seçmek için bağımsız olarak yapılandırılabilir.

2.5.3 Clock ve Startup

Sistem saat seçimi başlangıçta gerçekleştirilir ancak CPU' nun clock ayarı varsayılan olarak 8MHz olarak seçilmiştir. Arıza olduğu durumlarda 4-16 MHz arası harici bir clock seçilebilir. Eğer arıza tespit edilirse, sistem otomatik olarak dahili RC osilatörüne geri dönecektir. Birkaç ön ayarlayıcı, AHB frekansının, yüksek hızlı APB (APB2) ve düşük hızlı APB' nin (APB1) etki alanlarının yapılandırılmasına olanak sağlar. AHB ve yüksek hızlı APB'nin maksimum frekansı 72MHz' dir. Düşük hızlı APB için izin verilen maksimum frekans ise 36 MHz' dir.

2.5.4 Ön yükleme Modları

Başlangıçta, ön yükleme pimleri üç ön yükleme seçeneğinden birini seçmek için kullanılır :

- Kullanıcı flashından ön yükleme
- Sistem hafızasından ön yükleme
- Gömülü SRAM' den ön yükleme

Ön yükleyici sistem hafızasına yerleştirilmiştir. USART1 kullanılarak flash bellek tekrar programlanabilir.

2.5.5 DMA (Direct Memory Access)

STM32F103xx işlemci ailesinde bulunan genel amaçlı 7 kanal DMA, hafızadan hafızaya, çevre birimlerinden hafızaya ve hafızadan çevre birimlerine veri transferini amaçlar. Her kanal, yazılımsal tetikleme desteği ile özel donanım DMA isteklerine bağlanır. Yazılım ve transfer boyutları tarafından yapılan konfigürasyon, kaynak ve hedef arasında bağımsızdır. DMA, SPI, I2C, USART, TIM ve ADC çevre birimleri tarafından kullanılabilir.

2.5.6 Gelişmiş Kontrol Zamanlayıcısı (TIM1)

Gelişmiş kontrol zamanlayıcısı altı kanallı 3 fazlı PWM gibi görülebilir. PWM çıkışlarının ölü zamanları programla ayarlanabilir. Ayrıca gelişmiş kontrol zamanlayıcısı, tamamlayıcı genel amaçlı zamanlayıcı olarak da görülebilir. TIM1' deki 4 bağımsız kanalın kullanım amaçları :

- Girişleri Yakalama
- Çıkışları Karşılaştırma
- PWM Üretimi
- Tek Darbe Modu Çıkışı

olarak sıralanabilir. Eğer 16 bit genel amaçlı zamanlayıcı olarak ayarlanmışsa, TIMx ile aynı özelliklere sahiptir. Eğer 16 bitlik PWM jeneratörü olarak yapılandırılmışsa, tam modülasyon kapasitesine sahip olur.

Hata ayıklama modunda, gelişmiş kontrol zamanlayıcı sayacı donabilir ve PWM çıkışları bu çıkışlar tarafından yönlendirilen herhangi bir güç anahtarını kapatmak için devre dışı bırakılabilir. Aynı mimariye sahip genel amaçlı TIM zamanlayıcıları birçok özellik bakımından benzerdir.

2.5.7 Genel Amaçlı Zamanlayıcı (TIMx)

STM32F103xx işlemci ailesi gömülü üç adet senkronize genel amaçlı zamanlayıcı içerir. Bu zamanlayıcılar 16 bitlik otomatik yeniden yükleme sayacına, 16 bitlik bir önyükleyiciye ve her biri dört bağımsız kanalda giriş yakalama, çıkış karşılaştırma, PWM veya tek darbe modu çıkışına sahiptir. Bu da en büyük paketlerde 12 giriş yakalama/çıkış karşılaştırma/PWM özelliklerini içerisinde barındırır. Herhangi bir genel amaçlı zamanlayıcı PWM sinyali üretmek için kullanılabilir. Hepsi bağımsız DMA yolu ile çalışabilir. Bu zamanlayıcılar, artan enkoder sinyallerini ve 1 ila 3 hall effect sensörlerinin dijital çıkışlarını kontrol edebilirler.

2.5.8 I2C Bus

STM32F103XX işlemci ailesinde, multimaster ve slave modlarında iki adet I2C bus arabirimi çalışır. Bunlar standart ve hızlı modlarda çalışabilirler. Ana modda çift bağımlı adreslemeyi (sadece 7-bit) ve 7/10 bit adreslemeyi desteklerler. Bu veri yolu DMA ile kullanılabilir ve SM Bus 2.0/PM veri yolunu destekler.

I2C haberleşmesinde kullanılan iki bağlantı ucu vardır. Bunlar serial data (SDA) ve serial clock (SCL) olarak adlandırılırlar. Bu iki bağlantı ucuyla çift yönlü haberleşme yapılabilir. Bu bağlantı uçlarına bağlı olan cihazlar master ve slave olarak kullanılabilirler. Master olarak kullanılan cihaz veri transferi boyunca sağladığı clock sinyaliyle veri transferini yönetir. Bu haberleşme protokolü üç farklı hızda veri transferini gerçekleştirir. Bunlar Standart Mode(100 kbit/s), Fast-Mode (400 kbit/s) ve High Speed Mode(3.4Mbit/s)'dir.

2.5.9 USART Birimi

USART birimlerinden biri 4.5 Mbit/s hıza kadar haberleşme yapabilmektedir. Diğer mevcut arabirimler 2.25 Mbit/s hızla haberleşme yapabilmektedirler. Tüm USART ara yüzleri DMA veri yolu ile kullanılabilir.

2.5.10 SPI Veri Yolu

Açılımı “Serial Periphal Interface” dir. Motorola tarafından geliştirilen senkron bir haberleşme türüdür. SPI veri yolu iki adede kadar, tam çift yönlü ve tek yönlü iletişim modlarında slave ve master modlarında 18Mbit/s’ ye kadar iletişim kurabilir. 3 bitlik ön ayarlayıcı 8 master mod frekansı ve ayarlanabilir 8 bitlik ve 16 bitlik bir yapı verir. Donanım SD kart modunu destekler. İki SPI veri yolu da DMA kontrolörü ile kontrol edilebilir.

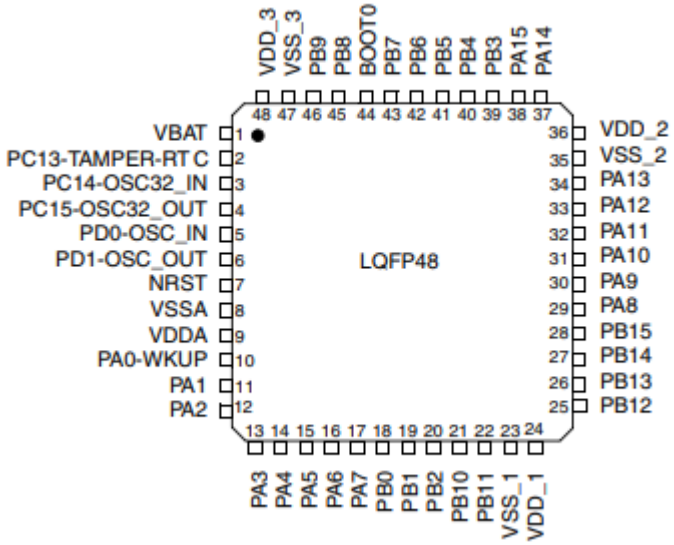
SPI veriyolu çift yönlü haberleşmeyi master slave ilişkisi ile sağlar. Master biriminden slave sinyal ve verinin gönderilmesi ile haberleşme başlamaktadır. SPI haberleşmesinde 3 adet bağlantı ucu vardır. Bunlar MOSI(Master Out, Slave In), MISO (Master In, Slave Out) ve SCK (Serial Clock)’dur. Bunların dışında slave birimini belirlemek için CS(Chip Select) veya başka bir adıyla SS(Slave Select) hattı bulunur.

2.5.11 CAN Bus Protokolü

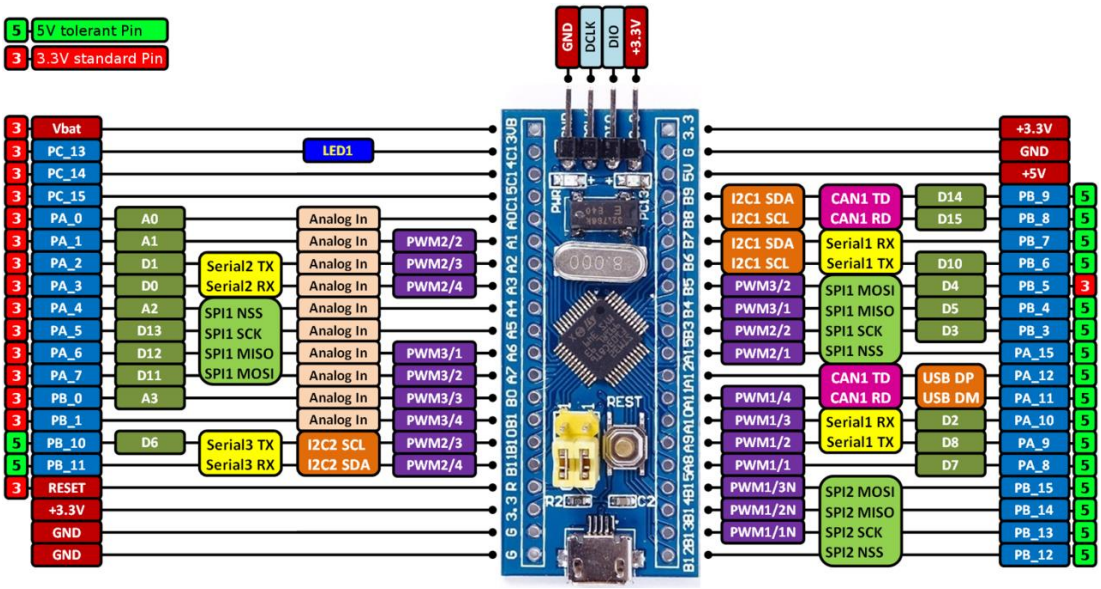
Açılımı “Controller Area Network Bus” yani “Kontrol Alan Ağı Veri Yolu” dur. CAN veri yolu 1Mbit/s hıza kadar çıkabilmektedir. Bu veri yolu 29 bit olarak tanımlanmış genişletilmiş yapının yanında 11 bit olarak tanımlanmış standart yapıyla veri alış verişi yapabilir.

CAN Bus protokolü Bosch’un otomotiv sektörü için geliştirdiği haberleşme standardıdır. Araçtaki ana kontrol birimi ile çevre birimler arasındaki haberleşmeyi sağlar. Günümüzde otomotiv sektörünün yanı sıra birçok endüstriyel alanda kullanılmaktadır. Can Bus protokolü ile aracın merkezi kontrol birimi araçtaki sensörler sayesinde aracın hız bilgisi, anlık yakıt tüketimi ve bunlar gibi birçok bilgiye hızlı ve güvenilir bir şekilde ulaşabilir. Bu protokol sayesinde her bir çevre birimden ayrı kablo gelmesinin önüne geçilmiş olur ve daha sade ve güvenilir bir sistem kurulmasını sağlar. Bu protokol araç üzerindeki birimlerden gelen binlerce veriyi çok güvenilir bir şekilde aktarabilme özelliğine sahiptir. Özellikle sistem güvenliğinin ön planda tutulduğu uygulamalarda kullanılır.

2.6 STM32F103C8T6 İşlemcisi



Şekil 2. 10 : STM32F103C8T6 Pin Gösterimi. Kaynak : https://www.st.com



Şekil 2. 11 : STM32F103C8T6 Pin Adlandırması. Kaynak : https://www.st.com

Şekil 2.10' da STM32F103C8T6 işlemcisinin pin gösterimi ve Şekil 2.11' de STM32F103C8T6 işlemcisinin pin adlandırması verilmiştir. STM32F103C8T6

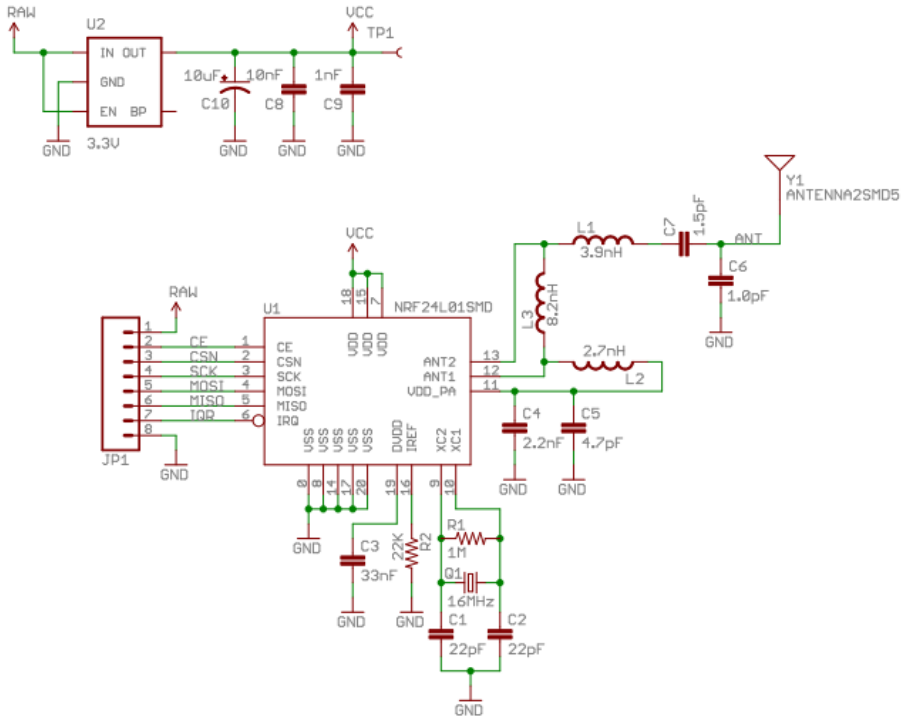
işlemcisi ARM Cortex M3 mimarisinin bir üyesidir. ARM bir kontrolcüden ziyade bir işlemci mimarisidir. Cep telefonlarında, tabletlerde, internet cihazlarında ve bunlar gibi gömülü sistemlerin girdiği birçok alanda ARM mimarisi kendisine yer edinmiştir. STM32F103C8T6 işlemcisinin CPU özelliklerini şöyle sıralayabiliriz :

- 32 bit ARM Cortex M3 işlemci mimarisi
- 64 Kbyte Flash
- 20 Kbyte SRAM
- 72 MHz maksimum frekans
- 2V ile 3.6V arası gerilimlerde çalışabilme özelliği

Dahili dış çevre birimleri yada içerdiği donanımlar ise :

- 37 tane giriş/çıkış pini
- Dahili RTC
- Tek çevrimde çarpma ve bölme yapabilme
- Dahili sıcaklık sensörü
- 7 kanal DMA modülü
- 2 IIC
- 2 SPI
- 3 USART
- 1 USB 2.0
- 1 CAN
- 1 gelişmiş kontrol zamanlayıcısı

2.7 NRF24L01 WIRELESS Modulü



Şekil 2. 12 : NRF24L01 Wireless Modül Devre Şeması. Kaynak : https://www.sparkfun.com

Şekil 2.12’ de NRF24L01 Wireless modülünün devre şeması gösterilmiştir. NRF24L01, gömülü bir temel bant protokolü altyapısına sahip tek bir çipli 2.4 GHz alıcı-verici ultra düşük güçte kablosuz uygulamalar için tasarlanmıştır. NRF24L01, bir seri çevresel arabirim (SPI) üzerinden yapılandırılmakta ve çalışmaktadır. NRF24L01 frekans kanalı gibi kullanıcı tarafından yapılandırılabilir. 250 KBps, 1MBps ve 2MBps gibi hızlarda haberleşme hızı seçilebilir. Ultra düşük güç tüketimine sahiptir. Çalışma voltajı aralığı 1.9V ile 3.6V arasında değişmektedir. Verici sinyal gücü +7dB, alıcı hassasiyeti 90dB’in altındadır. Haberleşme mesafesi açık alanda 250 metredir.

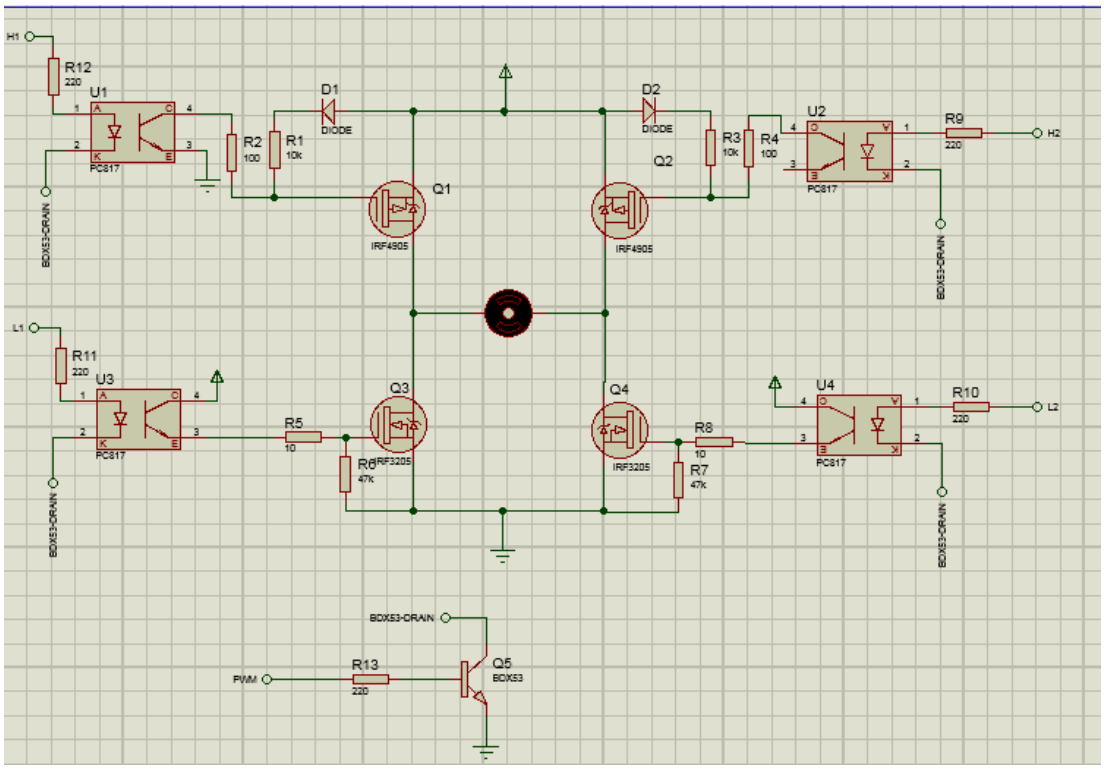
3. DONANIM

3.1 Hall Effect Sensörü

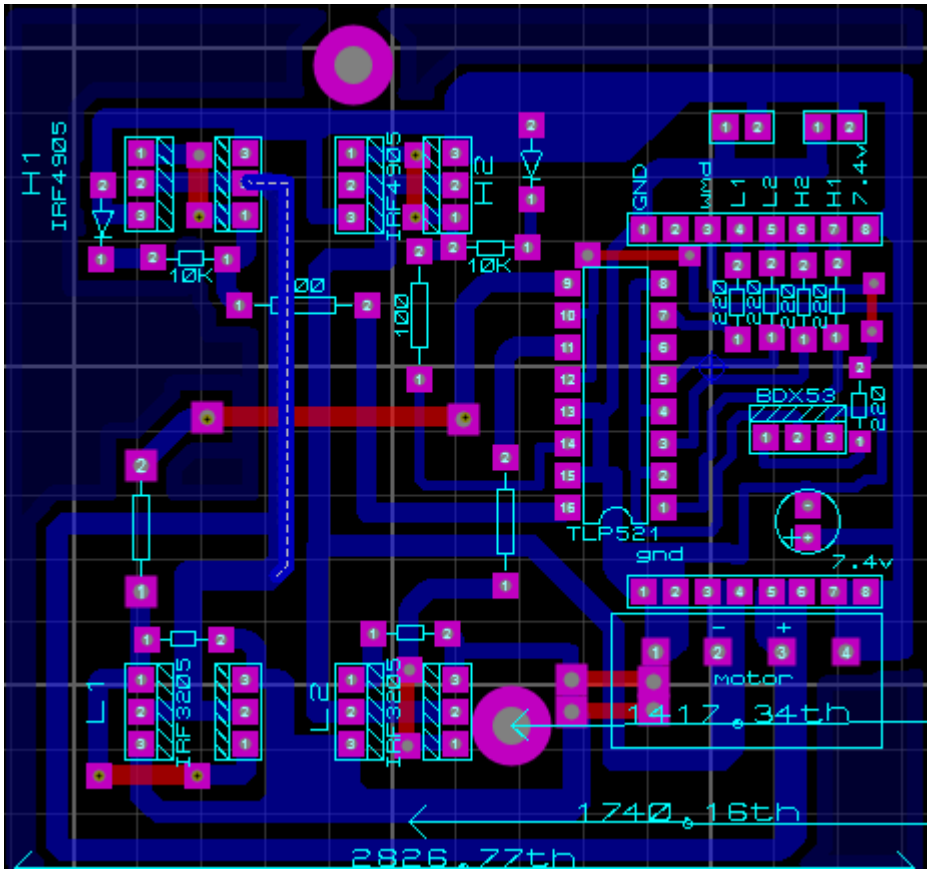
Araçlardaki hız ölçümünü yapabilmek için US1881 hall effect sensörü kullanılmıştır. Bu sensör manyetik alan değişimlerine karşı dijital çıkış vermekte ve bu şekilde araçların tekerlerine yerleştirilen mıknatıslarla araç hareket ettikçe oluşan manyetik alan değişimleri, hall effect sensör sayesinde algılanmakta ve bu şekilde aracın hızı tespit edilmektedir. Hall effect sensör ile tekerlerden alınan hız verisi ana kontrol kartına aktarılmaktadır. Böylece araçların ivmeli hızlanma, ivmeli yavaşlama ve sabit hızlı hareket ederken istenilen hızda gitmesi sağlanmaktadır. Kullanılan hall effect sensörünün kullanımının kolay olması ve maliyetinin düşük olmasından dolayı çeşitli alanlarda tercih edilmektedir.

3.2 DC Motor Sürücü Devresi

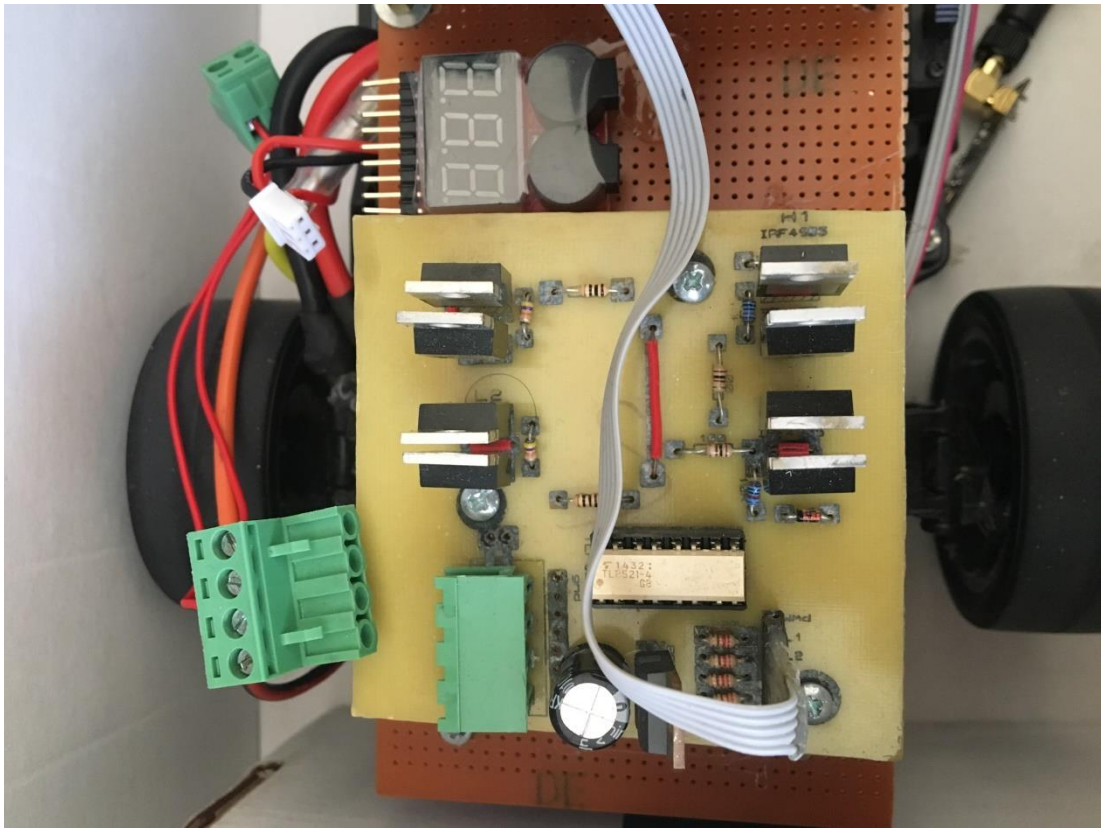
Araçların hız kontrollerini yapabilmek ve araçların hızlarını kontrol ederken MOSFET' lerin gate ucuna gelen PWM sinyalinin düzgün gelmesi için DC motor sürücü devresi tasarlanmıştır. MOSFET' lerin ısınmaması ve yanmaması için anahtarlama hızlarının olabildiğince hızlı olması gerekmektedir. Ayrıca anahtarlama hızlarının yanında MOSFET deşarj edilirken de mümkün olduğunca hızlı bir şekilde deşarj edilmelidir. Şekil 3.1' de proje kullanılan DC motor sürücü devresinin isis çizimi, Şekil 3.2'de ares çizimi, Şekil 3.3' de ise yapımı tamamlandıktan sonraki görseli görülmektedir.



Şekil 3. 1 : DC Motor Sürücü Devresi İsis Çizimi



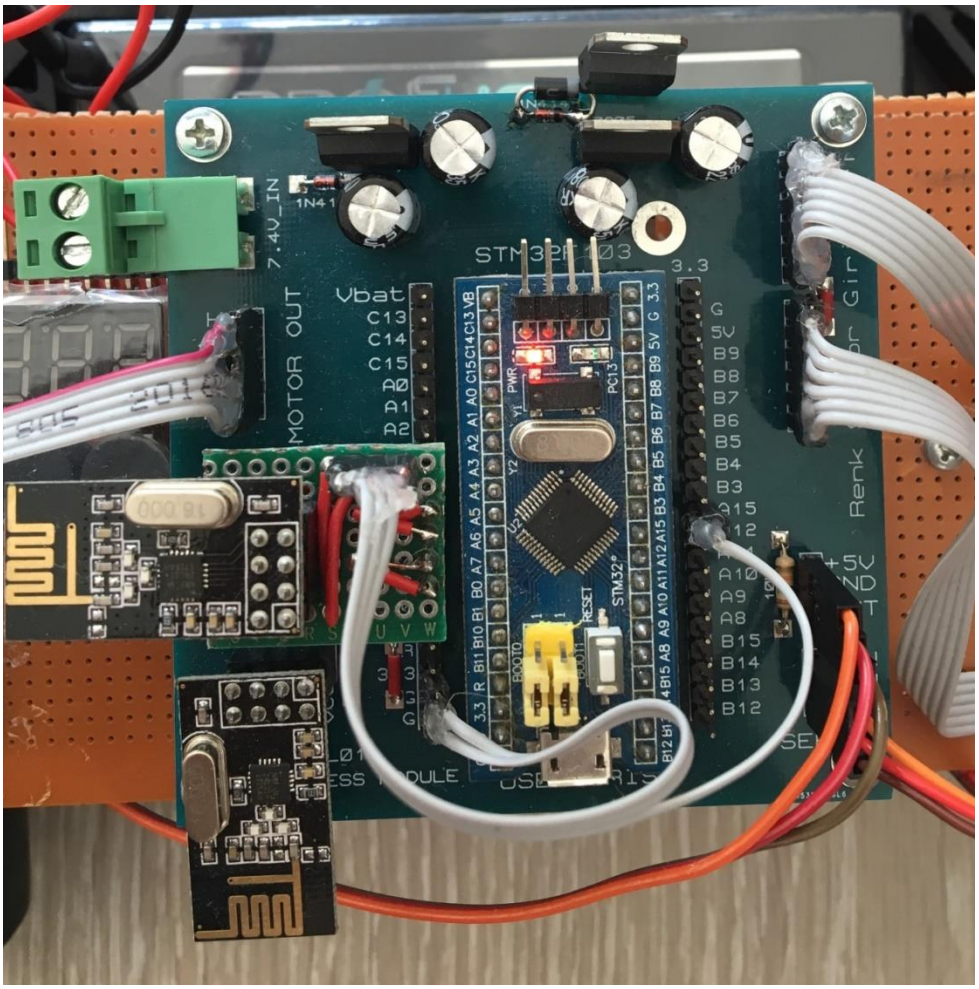
Şekil 3. 2 : DC Motor Sürücü Devresi Ares Çizimi



Şekil 3. 3 : DC Motor Devresi

3.3 Ana Kontrol Kartı

Şekil 3.4'de araçların genel kontrolü için tasarlanmış olan ana kontrol kartının devresi görülmektedir. Ana kontrol kartı üzerinde 2 adet alıcı-verici devresi, motor kontrol çıkış pinleri, hall effect sensör pinleri, servo motor kontrol ucu ve araçların yol takibi yapabilmeleri için renk sensörlerinden gelen giriş pinlerini barındırır. Ana kontrol kartı üzerinde ARM Cortex M3 mimarisinin işlemcisi olan STM32F103C8T6 bulunmaktadır.



Şekil 3. 4 : Ana Kontrol Kartı

Araçlar arasındaki haberleşme NRF24L01 WIRELESS modülü ile yapılmaktadır. Şekil 3.4' de görüldüğü gibi iki ayrı haberleşme kanalında haberleşme yapan iki ayrı modül kullanılmıştır. NRF24L01 modülü aynı anda hem alıcı hem de verici olarak kullanılabilmesine rağmen burada iki ayrı modül kullanılmasının sebebi, araçlar arasında bilgi alışverişi yapılırken herhangi bir gecikmeden kaynaklı sorun yaşanmaması ve araçlar arasındaki bilgi aktarım hızının maksimum hızda yapılması için böyle bir sistem tasarlanmıştır. Bu sistemde STM32F103C8T6 işlemcisinin iki ayrı SPI modülü kullanılmakta ve iki ayrı haberleşme modülünden alınan veriler veya gönderilen verilerin çakışması engellenmektedir.

Araçların tekerlerine manyetik alan oluşturması için mıknatıslar yerleştirilmiştir. Araçlar hareket ettikçe oluşan bu manyetik alan değişimlerini algılaması için US1881 hall effect sensörü kullanılmıştır. Bu sensör sayesinde

araçların hareketi boyunca oluşan manyetik alan değişimleri US1881 sensörünün çıkış pininin STM32F103C8T6 işlemcisinin interrupt pinine bağlanmasıyla her değişim bu şekilde algılanmakta ve hız hesabı yapılmaktadır. Hall effect sensörüyle algılanan her manyetik alan değişiminde araçlardaki hız ve konum verileri güncellenmekte ve bu veriler de diğer araçlara aktarılmaktadır. Bu yüzden yapılan hız hesabının doğruluğu bu noktada çok önemlidir.

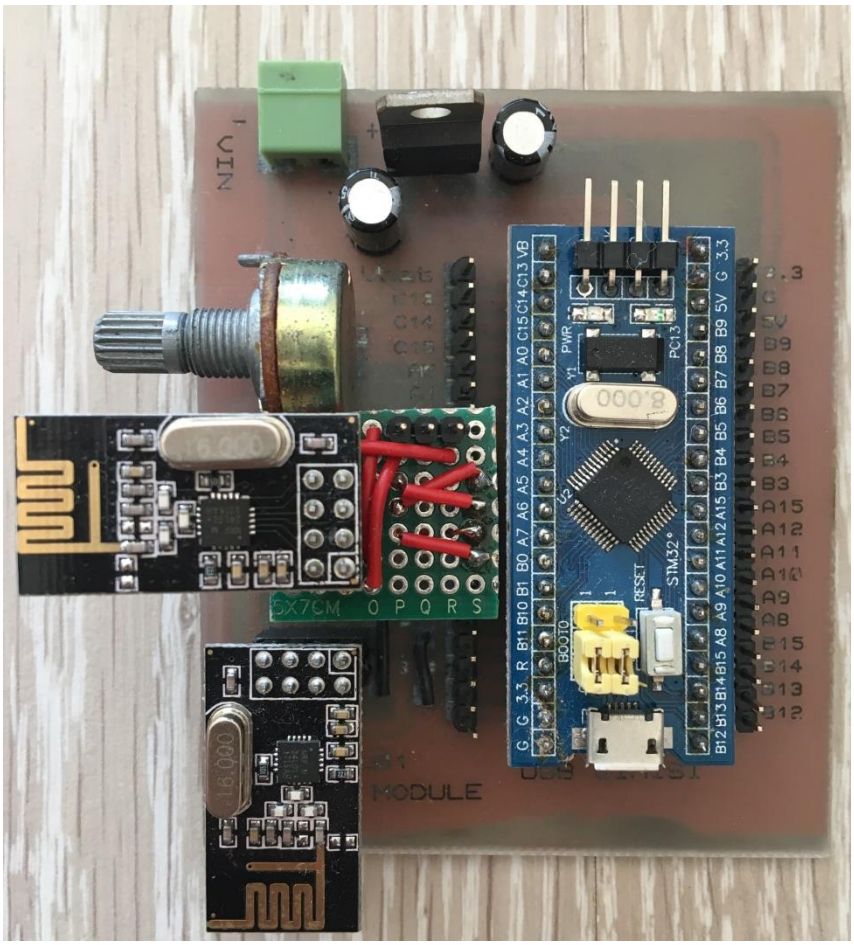
Araçların yol takibini yapabilmeleri için renk sensörü kullanılmıştır. Bu sensörler siyah ve beyaz çizgiye göre lojik "1" ve lojik "0" çıkış vermektedirler. Projede uygulama aşamasında beyaz yol üzerine siyah şerit yapılmış ve araçların bu siyah şeridi takip etmeleri sağlanmıştır. Aracın hareketi sırasında toplamda kullanılan 7 adet renk sensörünün hepsinden alınan verilere göre aracın ön tekerlerinin yönünü belirleyen RC servo motorun konumu güncellenmektedir. Bu şekilde aracın şerit dışına çıkmadan düzgün bir şekilde, tasarlanan mevcut trafik düzeni içerisinde hareket etmesi sağlanmaktadır.

3.4 Yol Takibi

Araçların yol takibini yapabilmeleri için renk sensörü kullanılmıştır. Bu sensörler siyah ve beyaz çizgiye göre lojik "1" ve lojik "0" çıkış vermektedirler. Projede uygulama aşamasında beyaz yol üzerine siyah şerit yapılmış ve araçların bu siyah şeridi takip etmeleri sağlanmıştır. Aracın hareketi sırasında toplamda kullanılan 7 adet renk sensörünün hepsinden alınan verilere göre aracın ön tekerlerinin yönünü belirleyen RC servo motorun konumu güncellenmektedir. Bu şekilde aracın şerit dışına çıkmadan düzgün bir şekilde, tasarlanan mevcut trafik düzeni içerisinde hareket etmesi sağlanmaktadır

.

3.5 Verici Devresi



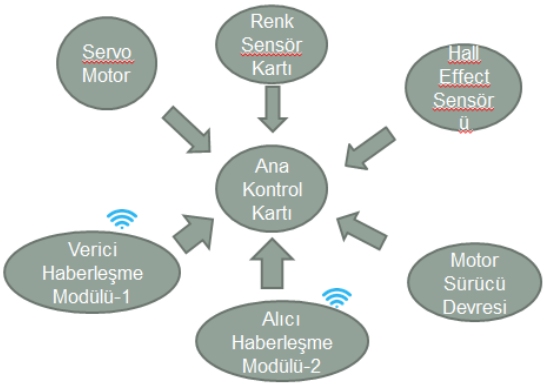
Şekil 3. 5 : Verici Devresi

Şekil 3.5'de verici devresi görülmektedir. Verici devresinde de araçlardaki kontrol kartında olduğu gibi veri alış verişi ayrı iki haberleşme kanalı kullanılarak yapılmaktadır. Burada amaç araçlarda olduğu gibi iletişim hızını artırmaktır. Verici devresiyle araçlardan alınan hız ve konum bilgisi C# ortamına aktarılmaktadır. Bu şekilde hem araçların gittikleri yol bilgisinin ve hız bilgisinin doğruluğu test edilmekte hem de gerçek ortamdan alınan verilerin bilgisayar ortamında simülasyonu yapılarak sonuçlar görülmektedir.

4. YAZILIM

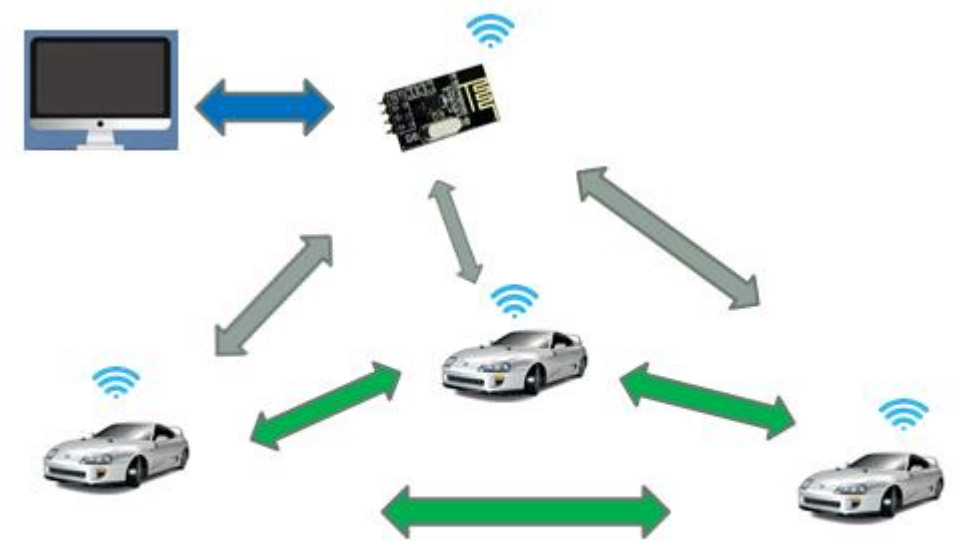
4.1 Araçların Çalışma Diyagramı

Şekil 4.1' de her bir aracın barındırdığı donanım yapısı ve blok şeması verilmiştir. Projede kullanılan her bir araç üzerinde ana kontrol kartı, yol takibi için renk sensör kartı ve servo motor, hız ölçümü yapmak için arka tekere yerleştirilmiş hall effect sensör, hız kontrolü yapmak için motor sürücü devresi ve her bir aracın diğer araçlarla veri alış verişini sağlayan bir adet alıcı modül, 1 adet de verici modül bulunmaktadır. Ana kontrol kartı renk sensörlerinden gelen bilgiye göre servo motorun açısını ayarlamaktadır. Bu şekilde aracın yolu takip etmesi sağlanmaktadır. Araçlar sistemdeki konumlarına ve hızlarına göre hızlarını sabit ivmeli olarak artırabilmekte, sabit ivmeli olarak azaltabilmekte veya sabit hızlı hareket yapabilmektedir. Araçlar gitmesi gereken hızlarını ayarlarken hall effect sensörden gelen bilgiyi kullanmaktadırlar. Hall effect sensör arka tekere bağlanan mıknatısları algıladığında dijital çıkış vermekte ve böylece araçların hızları bulunmaktadır. Araçların gitmesi gereken hızı ayarlamak için yazılımsal PID kullanılmıştır. Sistemin düzgün çalışabilmesi için çok önemli olan araçlar arası haberleşme 2 adet modül kullanılarak yapılmıştır. Burada her araç üzerinde bir adet verici ve 1 adet alıcı modül kullanılmıştır. Kullanılan modüller farklı kanallar üzerinden tek yönlü olarak çalışmaktadırlar.



Şekil 4. 1 : Araçların Çalışma Diyagramı

4.2 Projenin Organizasyon Şeması ve Çalışma Şekli



Şekil 4. 2 : Sistemin Çalışma Diyagramı

Yukarıda Şekil 4.2' de yapılan tez çalışmasının organizasyon şeması gösterilmiştir. Sistem genel olarak araçların kendi aralarındaki haberleşmesine göre işlemektedir. Kurulan haberleşme ağında her araç kendi bilgilerini 100ms örnekleme zamanı ile diğer araçlara göndermektedir. Örnekleme zamanının 100ms olmasının nedeni kullanılan modüle bağlı olarak haberleşme verimini artırmaktır. Örnekleme zamanının azaltılması haberleşmenin aksamasına neden olmaktadır. Yapılan testler sonucunda kullanılan haberleşme modülü de göz önünde bulundurularak haberleşme örnekleme zamanı bu şekilde seçilmiştir. Burada merkezi kontrol birimi uygulamada olabilecek olası hataların önüne geçmek, sistemin çalışmasını daha iyi takip etmek ve sistemin doğruluğunu test etmek için kullanılmıştır. Ayrıca merkezi kontrol sistemiyle C# ortamında hazırlanan arayüze girilen başlangıç değerleri araçlara gönderilmektedir. C# ortamına araçlardan gelen veriler aktarılarak araçların hareketi arayüzde görülmekte ve araçların anlık hız, konum ve birbirleri arasındaki mesafe farkları da ara yüzde gösterilmektedir. Araçlar diğer araçların verilerini merkezi kontrol biriminden almamakta bu verileri sadece kendi aralarındaki haberleşme ağından elde edebilmektedirler. Her araçta hem alıcı hem de verici olarak kullanılan

iki ayrı modül bulunmakta ve oluşturulan ayrı kanallarda veri transferini gerçekleştirmektedir. Organizasyon şemasında 3 aracın olduğu sistem düşünülmüştür. Sistemdeki araç sayısı artırabilir. Projede sisteme giren araçlar diğer araçlarla haberleşmeye başlamaktadır. Her araç diğer araçların konum, hız ve ivme gibi bilgilerini bilmekte ve bir sonraki anda nasıl hareket edeceklerine karar vermektedirler. Araçların yan yoldan ana yola girişleri belli kavşak noktalarından sağlanmaktadır. Ana yola girecek olan araçlar belirlenen belirli bölgelerde diğer araçlara bir sonraki kavşak noktasında ana yola dahil olacağını bildirmektedir. Bundan sonra tüm araçlar bilgilerini karşılaştırarak ana yola dahil olacak aracın ana yoldaki konumunu belirlemekte ve buna göre hızlarını ve konumlarını ayarlamaktadırlar. Birleşmenin olacağı kavşak noktasına gelindiğinde araç daha önce belirlenen konumuna göre ana yola dahil olacaktır. Ana yol ve yan yoldaki araçlar önlerindeki aracı takip algoritmasına göre güvenli mesafeyi bozmadan takip etmektedirler.

Araçların kendi aralarındaki haberleşmesi, sistemde kullanılan yolun ayrı olması ve sistemdeki araçların hepsinin aynı özellikte olması sistemin güvenliğini ve işleyişini olumlu yönde etkilemektedir. Araçlar arasındaki haberleşme hızının ve araçların hızlarının doğru ayarlanmasının sistem üzerinde çok önemli etkisi vardır. Araçların hareketini sağlayan DC motor yüksek torklu seçilmiştir. Bunun nedeni, hassas hız ayarları yapılırken, araçların kararlı bir şekilde hareket etmesi ve özellikle araçların hızlanma sırasında hareketi zorlayan etkenlere karşı daha iyi tepki vermesinin sağlanmasıdır.

Proje Şekil 4.2' de gösterilen organizasyon şemasına göre projenin uygulaması oluşturulan özel yollar üzerinde gerçekleştirilmiştir. Uygulamada 1/10 oranında küçültülmüş araçlar kullanılmıştır. Buna bağlı olarak araçların hızları da gerçek bir araca göre 1/10 oranında ayarlanmıştır. Araçlar ivmeli hızlanma, ivmeli yavaşlama ve sabit hızlı hareket etmektedirler. Burada araçların belirli bir ivme ile hızlanıp yavaşlaması algoritmanın doğru işlemesi açısından kritik öneme sahiptir. Sistemin tüm algoritması sabit ivmeli hareket üzerine kuruludur. Araçların hızlanma ve yavaşlama ivmeleri arasında 4 kat fark olacak şekilde ivmeler seçilmiştir. Burada amaç araçların hızlanırken hızlarının ayarlanmasının daha zor olmasıdır. Araçların hızlarının ayarlanabilmesi için yazılımsal PID kontrolör kullanılmıştır. Burada araçların ivmeli hareket yaparken anlık olarak gitmesi gereken hızda gitmesi çok

önemlidir. Bu hız ayarı PID kontrol ile yapılmıştır. Hedeflendiği gibi araçların güvenli bir şekilde seyahat etmeleri sağlanmıştır.

4.3 Takip Algoritması

Mevcut sistemde tüm araçlar teknik özellik bakımından aynı araçlar olarak seçilmiştir ve çalışma buna göre yürütülmüştür. Sistemde bulunan tüm araçlar gerekli olduğu durumlarda diğer araçlarla haberleşebilecekleri bir haberleşme ağına bağlıdırlar. Her araç belirlenen örnekleme zamanında diğer araçlarla hız, konum ve diğer gerekli tüm bilgilerini paylaşmaktadır. Araçların hızlarının artması veya azalması tamamen sistemin işleyişine göre otomatik olarak değişmektedir. İstenildiği takdirde bir merkezi kontrol birimi ile de sisteme müdahale edilebilir. Takip algoritmasına göre sistemdeki araçlar sadece önündeki aracın hareketine göre aradaki güvenli mesafeyi koruyarak hareket etmektedirler. Aşağıda Denklem (4.1)' de (Bozuyla, v.dğr. 2018) güvenli takip için gerekli denklem ve takip algoritması detaylı olarak verilmiştir.

$$X_B - X_A + \frac{Vb^2}{a_{dec}} - \frac{Va^2}{a_{dec}} \geq \Delta\text{xmin} \tag{4.1}$$

Burada:

X_B : B aracının konumu

X_A : A aracının konumu

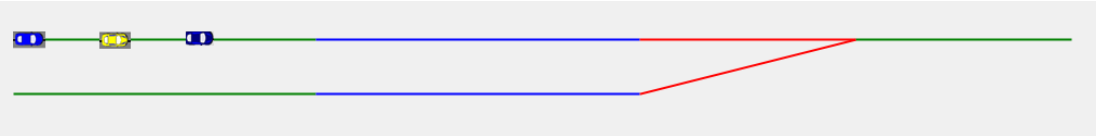
V_b : B aracının hızı

V_a : A aracının hızı

a_{dec} : Araçların yavaşlama ivmesi

Δxmin: Güvenli takip mesafesini ifade etmektedir.

Şekil 4.3'de C# ortamından alınan takip algoritması simülasyon görüntüsü verilmiştir. Yeşil renk ile gösterilen bölgeler serbest bölge, mavi ile gösterilen bölge kontrol bölgesi ve kırmızı ile gösterilen bölge ise kritik bölge olarak tanımlanmıştır.



Şekil 4. 3 : Takip Algoritması

Takip Algoritması

Bu algoritma verilirken öndeki araç B aracı arkadaki araç A aracı olarak düşünülmüştür.

EĞER(A ve B aracı arasındaki konum farkı < Δxmin) **İSE**

A aracının hızını azalt

A aracının konumunu güncelle

DEĞİLSE EĞER (B aracının hızı > A aracının hızı) **İSE**

A aracının hızını artır.

A aracının konumunu güncelle

DEĞİLSE EĞER $(X_B - X_A + \frac{Vb^2}{a_{dec}} - \frac{Va^2}{a_{dec}} < \Delta xmin)$ **İSE**

A aracının hızını azalt

A aracının konumunu güncelle

DEĞİLSE

A aracının hızını artır

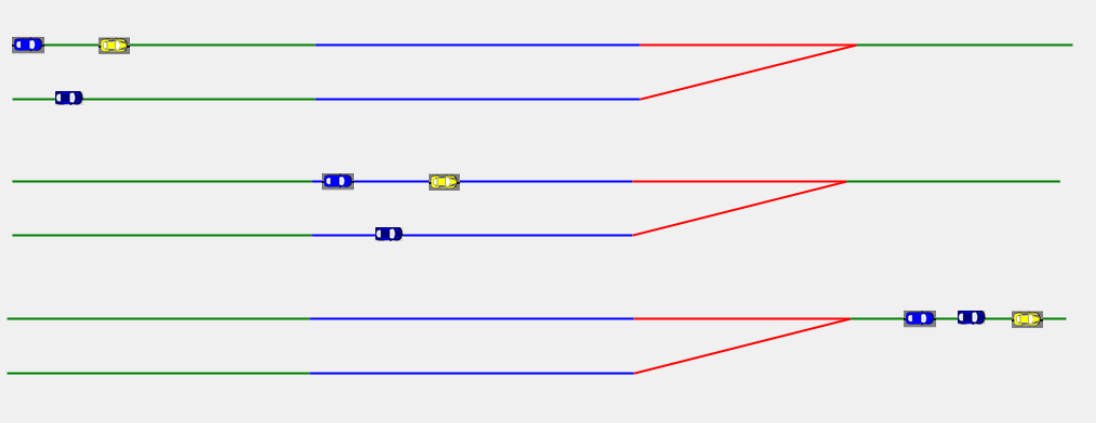
A aracının konumunu güncelle

(Bozuyla, v.dğr., 2018)

Sistemin daha hızlı daha etkin kullanımı ve oluşturulan trafik düzeni içerisinde araçların kaza yapmadan hareket etmesi takip algoritması ile sağlanmaktadır.

4.4 Birleşme Algoritması

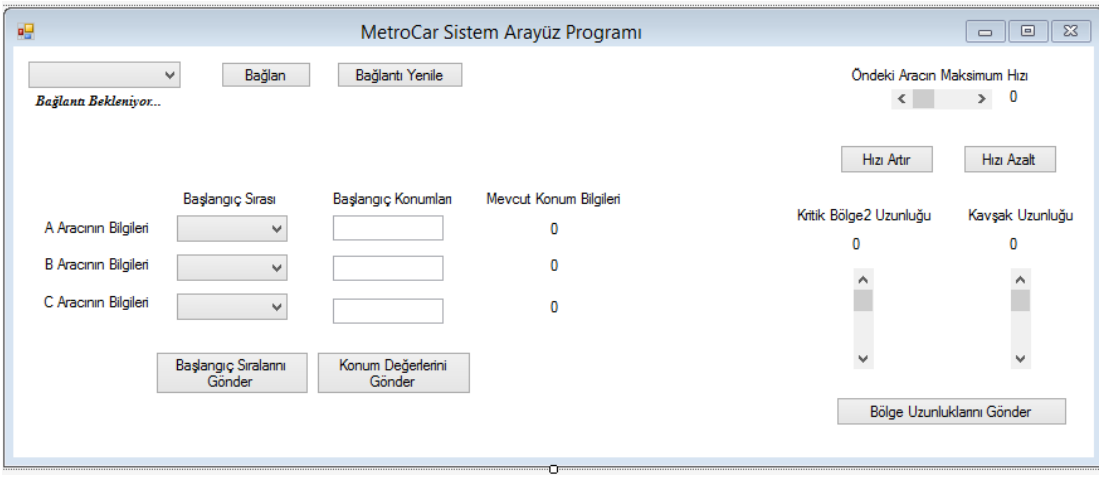
Birleşme algoritmasında yan yoldan gelen araçların kendisine paralel ana yola nasıl girecekleri ele alınmıştır. Yan yoldan gelen ve ana yola girecek tüm araçlar herhangi bir çarpışma olmadan ana yola girmek zorundadırlar. Yan yoldan gelen araçların maksimum hıza ulaşabilmeleri için yolun yeteri kadar uzun olması sağlanmıştır. Bunun sebebi ana yoldan gelen araçlarla yan yolda hareket eden araçların hızlarının benzer değerlerde olmasının sağlanmasıdır. Birleşme algoritmasındaki temel problem ana yol ve yan yoldan gelen araçların güvenli bir şekilde birleşmeyi sağlayabilmeleri için aradaki uygun mesafeyi oluşturabilmeleridir. Bunun için de hangi araçların hızlarını artıracakları hangi araçların hızlarını azaltacakları konusu oldukça önemlidir.



Şekil 4. 4 : Birleşme Algoritması

Şekil 4.4’ de birleşme algoritmasının çalışma şekli üç aşamada görülmektedir. Yeşil renk ile gösterilen bölgeler serbest bölge, mavi ile gösterilen bölge kontrol bölgesi ve kırmızı ile gösterilen bölge ise kritik bölge olarak tanımlanmıştır. Serbest bölgede araçlar sadece takip algoritmasına göre hareket eder. Araçlar bu bölgede sadece aynı yoldaki araçlarla haberleşerek kendi önündeki aracı takip eder. Yan yoldan gelen aracın ana yola birleşmesi söz konusu olduğunda kontrol bölgesinde araçlar arası haberleşme başlamaktadır. Bunun amacı hangi aracın birleşme noktasına daha yakın olduğunu belirleyerek öncelik sırasını belirlemektir. Ana yoldan ve yan yoldan sürekli bir trafik akışının olduğu düşünülürse yan yoldan ana yola birleşecek olan araç ana yolda hangi iki aracın arasına gireceğine kontrol bölgesinde karar verir. Kritik bölgeye geldiğimizde araçlar kontrol bölgesinde elde ettikleri bilgilere göre sıralanmaktadırlar. Bu bölgede daha önce belirlenen öncelik sırasına göre yan yoldan gelen araç ana yoldaki hangi iki aracın arasına girecekse o araçlar arasında uygun bir boşluk oluşacaktır. Birleşmenin olacağı kavşak noktasına gelindiğinde ise yan yoldan gelen araç kritik bölgede ana yoldaki araçlar arasında oluşan boşluğa güvenli bir şekilde girecek ve hareketine devam edecektir. Bu şekilde birleşme algoritması tamamlanacaktır. Burada önemli bir nokta ise birleşme algoritması gerçekleşirken takip algoritmasına aynen uyulması gerektiğidir.

4.5 C# Programı ve Programda Kullanılan Kodlar



Şekil 4. 5 : C# Arayüzü

Şekil 4.5’ de C# arayüz programı görüntüsü verilmiştir. Aşağıda C# programında kullanılan kodlar verilmiştir.

```
    private void Form1_Load(object sender, EventArgs e) //program açıldığında yapılacak işlemler…
    {
       portbul(); // Aşağıda fonksiyon detayı verişlmiştir.
       timer1.Enabled = true; //timer1 aktif edildi.
    }



 #region portbul
    private void portbul()
    {
       for (int i = 0; i < 30; i++)
       {
          try
          {  //Bilgisayarda kullanıma hazır tüm portları bulmak için her portu sırasıyla açtık ve kapattık. Kullanılabilir olanları combobox1 içine yazdık.
             serialPort1.PortName = "COM" + i.ToString();
             serialPort1.Open();
             comboBox1.Items.Add(serialPort1.PortName);
             serialPort1.Close();
          }
          catch (Exception)
          {
             continue;
          }
       }
    }
    #endregion

    private void timer1_Tick(object sender, EventArgs e) //timer1 aktif olduğunda yapılacak işlemler.
    {
       try
       {

          veri = serialPort1.ReadExisting(); //gelen veriyi okuduk. Araçlardan vericiye şifrelenerek gönderilen veriler burada yarıştırılıp programa işleniyor.

          if (veri[0] == 'A')   //A aracı için hesaplar yapılıyor...
          {
             indis_h = veri.IndexOf('A');
             indis_z = veri.IndexOf('Z');
             indis_s = veri.IndexOf('S');
             indis_f = veri.IndexOf('F');
```

```
                indis_uzunluk1 = indis_z - indis_h;
                indis_uzunluk2 = indis_f - indis_s;
                indis_uzunluk3 = indis_s - indis_z;

                a_aracinin_hizi = Convert.ToInt32(veri.Substring(indis_h + 1,
indis_uzunluk1 - 1));
                a_aracinin_aldigi_yol = Convert.ToInt32(veri.Substring(indis_z + 1,
indis_uzunluk3 - 1));


            }
            else if (veri[0] == 'B')   //B aracı için hesaplar yapılıyor...
            {
                indis_h = veri.IndexOf('B');
                indis_z = veri.IndexOf('Z');
                indis_s = veri.IndexOf('S');
                indis_f = veri.IndexOf('F');

                indis_uzunluk1 = indis_z - indis_h;
                indis_uzunluk2 = indis_f - indis_s;
                indis_uzunluk3 = indis_s - indis_z;

                b_aracinin_hizi = Convert.ToInt32(veri.Substring(indis_h + 1,
indis_uzunluk1 - 1));
                b_aracinin_aldigi_yol = Convert.ToInt32(veri.Substring(indis_z + 1,
indis_uzunluk3 - 1));

            }
            else if (veri[0] == 'C')   //C aracı için hesaplar yapılıyor...
            {
                indis_h = veri.IndexOf('C');
                indis_z = veri.IndexOf('Z');
                indis_s = veri.IndexOf('S');
                indis_f = veri.IndexOf('F');

                indis_uzunluk1 = indis_z - indis_h;
                indis_uzunluk2 = indis_f - indis_s;
                indis_uzunluk3 = indis_s - indis_z;

                c_aracinin_hizi = Convert.ToInt32(veri.Substring(indis_h + 1,
indis_uzunluk1 - 1));
                c_aracinin_aldigi_yol = Convert.ToInt32(veri.Substring(indis_z + 1,
indis_uzunluk3 - 1));

            }
            label7.Text = (a_aracinin_aldigi_yol).ToString();
            label8.Text = (b_aracinin_aldigi_yol).ToString();
            label9.Text = (c_aracinin_aldigi_yol).ToString();
            Invalidate();
```

```
            }
            catch (Exception)
            {


            }
        }

        private void button2_Click(object sender, EventArgs e) //buton2 aktif
olduğunda araçların sistemdeki sıraları belirlenmektedir.
        {
            mesaj = "X" + (comboBox3.Text).ToString() + "Y" +
(comboBox4.Text).ToString() + "Z" + (comboBox5.Text).ToString() + "S";
            serialPort1.Write(mesaj);
            Invalidate();
        }

        private void button3_Click(object sender, EventArgs e) //buton3 aktif
olduğunda araçların sistemdeki başlangıç konumları belirlenmektedir.
        {
            mesaj = "a" + (textBox13.Text).ToString() + "b" +
(textBox14.Text).ToString() + "c" + (textBox15.Text).ToString() + "f";
            serialPort1.Write(mesaj);
            Invalidate();
        }

private void button4_Click(object sender, EventArgs e)
        {
            portbul();
        }

        private void button5_Click(object sender, EventArgs e) // buton5 ile sistemdeki
araçlara başla komutu verilmektedir.
        {
            serialPort1.Write("H" + "BASLA" + "F");
        }

        private void button6_Click(object sender, EventArgs e) //Sistemdeki araçları
durdurmak için kullanılmıştır.
        {
            serialPort1.Write("H" + "DUR" + "F");
        }

        private void button7_Click(object sender, EventArgs e) //kritik bölge ve kavşak
uzunluğu araçlara gönderilir.
        {
            mesaj = "R" + (vScrollBar1.Text).ToString() + "V" +
(vScrollBar2.Text).ToString() + "Ş";
            serialPort1.Write(mesaj);
            Invalidate();
```

```
}

private void hScrollBar1_Scroll(object sender, ScrollEventArgs e) //burada araçların çıkabileceği maksimum hız ayarlanmaktadır.
{
   label13.Text = hScrollBar1.Value.ToString();
   serialPort1.Write("F" + hScrollBar1.Value.ToString() + "C");
}
```

4.6 Araçlarda Kullanılan Haberleşme Kodları

```
void nrf24l01() //nrf24l01 fonksiyonu araçların genel haberleşme kodlarını göstermektedir. Aşağıda verilen kodlar A aracı için örnek olarak verilmiştir.

{

 //-------------------------------------------------------------

 radio.openWritingPipe(pipes[1]);

 radio.openReadingPipe(0, pipes[0]);

 radio.startListening();

 if (radio.available())  //ana kontrol kartı üzerinde bulunan alıcıda veri olması durumdan yapılacak işlemler.

 {

  radio.read(veria,  sizeof(veria));  //radio kanalından gelen veri okunarak veria' ya tanımlandı.

 }

//gelen veriler kodlamaya göre farklı mesajlara aktarılarak ayrılmaktadır.

 if (veria[0] == 'X') gelen_mesaj2 = String(veria);

 else if (veria[0] == 'A') gelen_mesaj3 = String(veria);
```

```
else if (veria[0] == 'B') gelen_mesaj4 = String(veria);

else if (veria[0] == 'C') gelen_mesaj5 = String(veria);

else if (veria[0] == 'a') gelen_mesaj6 = String(veria);

else if (veria[0] == 'R') gelen_mesaj7 = String(veria);

else if (veria[0] == 'F') gelen_mesaj8 = String(veria);

if (durum1 > 0) //durum1 her 100ms de "1" olmaktadır. A aracının bilgileri diğer araçlara gönderrilmektedir.

{

 durum1 = 0;

 giden_mesaj = "A" + String(a_aracinin_hizi) + "Z" + String(alinan_yol) + "S"  giden_mesaj.toCharArray(verig, sizeof(giden_mesaj)); //gönderilecek mesaj char tipindeki verig'ye tanımlandı.

 radio1.openWritingPipe(pipes[0]);

 radio1.openReadingPipe(1, pipes[1]);

 radio1.stopListening();

 radio1.write(verig, sizeof(verig)); //radio1 kanalından veri transferi gerçekleştirildi.

 }

}
```

4.7 Araçların Hız Ölçümünün Yapılması

Araçların hız ölçümü yapılırken kullanılan US1881 hall effect sensörü manyetik alan değişimlerine karşı dijital çıkış vermektedir. Her bir aracın tekerine

eşit aralıklarla 5 adet mıknatıs yerleştirilmiştir. Araçların teker çapları 6.5 cm' dir. Her bir manyetik alan değişiminde araç 4.082 cm yol almış olmaktadır. US 1881 sensörünün çıkış pini işlemcinin interrupt pinine bağlanmıştır. Veri geldiğinde işlemci kesmeye girmekte ve aracın hızını hesaplamaktadır. Aşağıda hız ölçümü için kullanılan yazılım fonksiyonu verilmiştir.

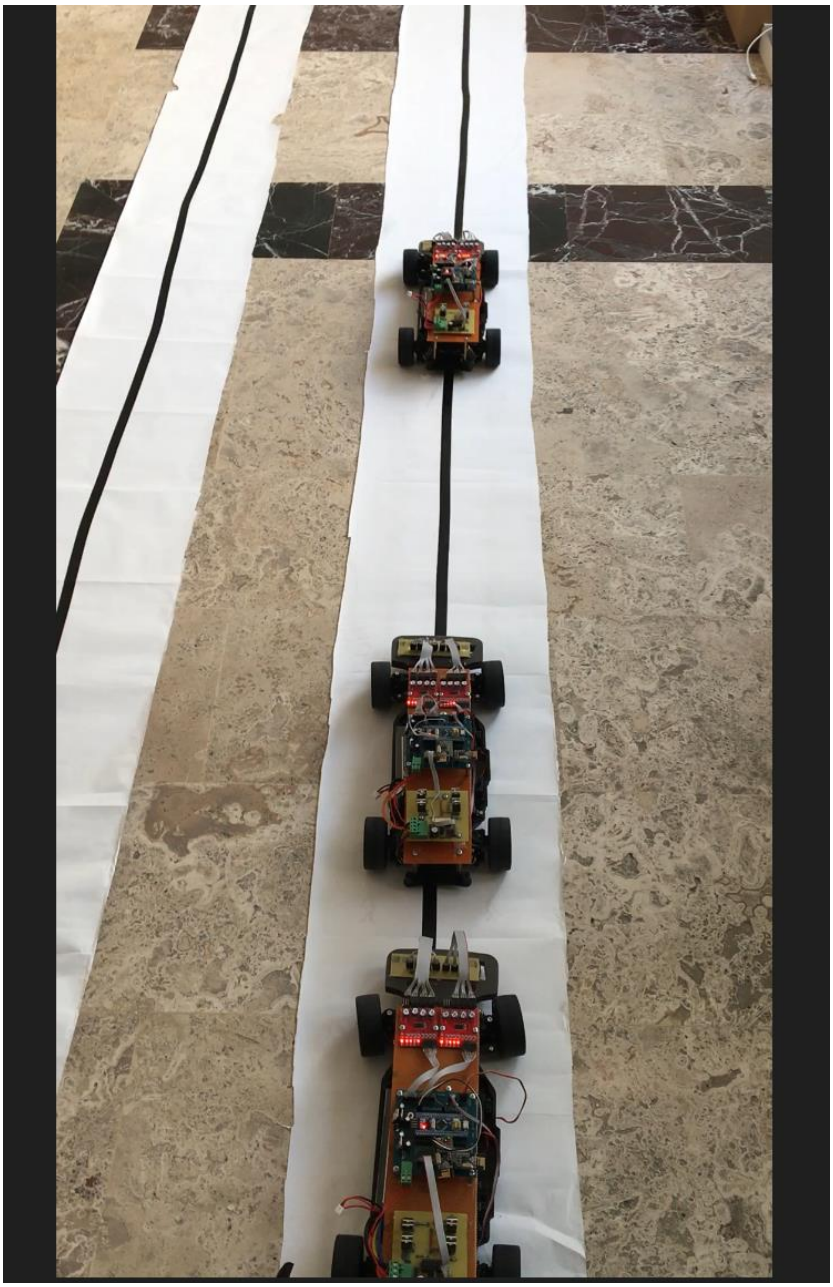
```
void interruptFunction() {

 son_zaman = micros();

 alinan_yol = alinan_yol + 0.4082; //araç tekerine 5 adet mıknatıs yerleştirildi. Her bir veride araç 4.082 cm yol alıyor.

 zaman = son_zaman - ilk_zaman;

 rps = 1000000 / zaman;

 hiz1 = (alinan_yol - yol) * rps + 0.5 ; //aracın gittiği hız hesaplanıyor.

 ilk_zaman = son_zaman;

 yol = alinan_yol;

 }
```

4.8 Tezin Uygulamaları

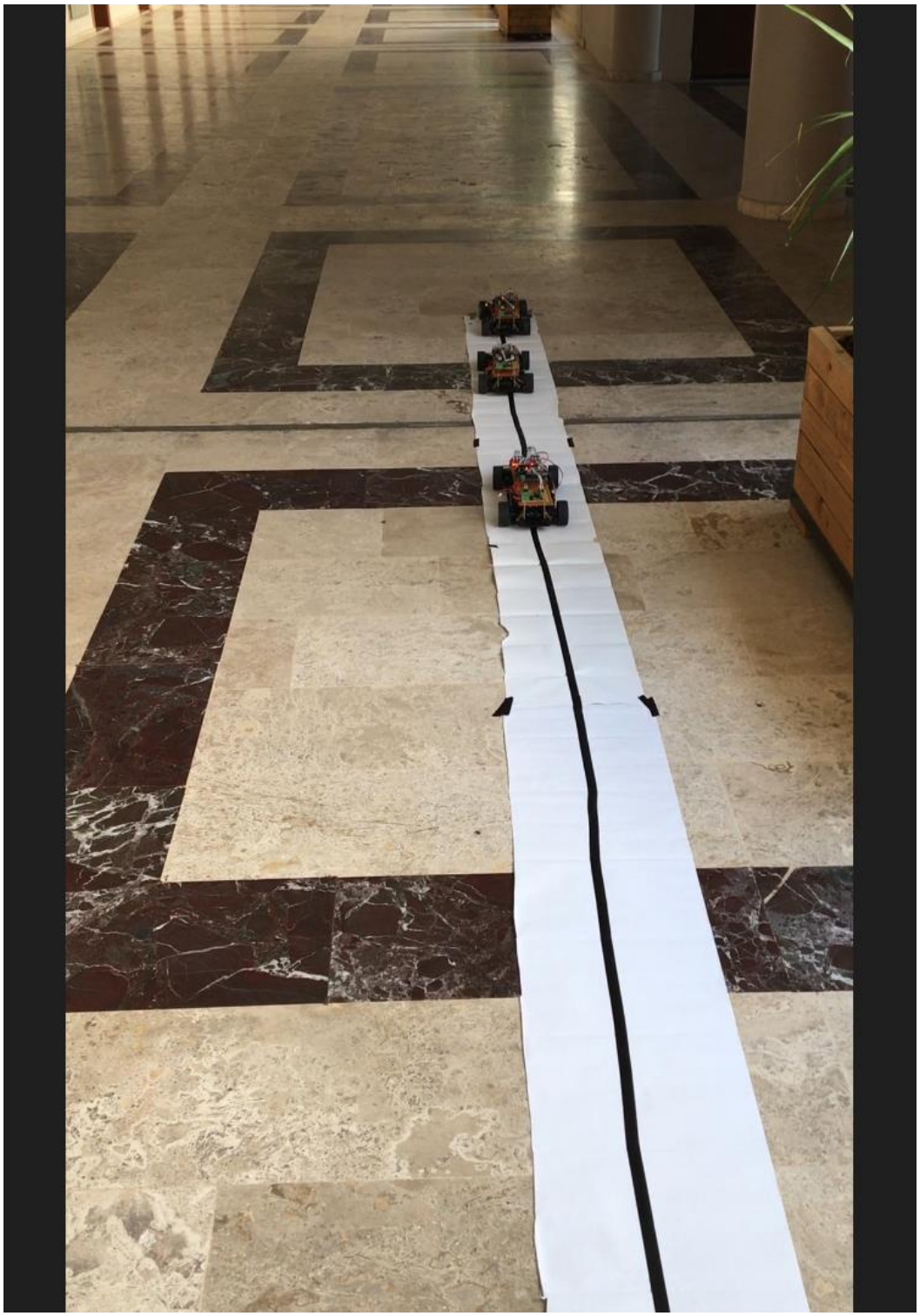
Tezin uygulama aşamasına geçilmeden önce takip ve birleşme algoritması C# programında oluşturulan simülasyon arayüzünde test edilmiştir. Bu testler sonucunda elde edilen veriler doğrultusunda hata oranın sıfıra yakın olduğu sonucuna varılmıştır. Tezin uygulamaları yapılırken 3 adet 1/10 oranında araç kullanılmıştır. Araçlar 2 m/s hıza ulaşabilmektedirler. Sistem testleri yapılırken önce takip algoritmasına göre tek yol üzerinde 2 araçlı ve 3 araçlı olmak üzere testler yapılmıştır. Takip algoritması testleri yapılırken araçların başlangıç sıraları ve başlangıç konumları C# üzerinde oluşturulan arayüz programıyla tanımlanmaktadır. Öndeki aracın harekete başlamasıyla diğer araçlar takip algoritmasına göre önündeki

aracı takip etmektedirler. Takip algoritması uygulamasından elde edilen sonuçlar sistemin +/- 5cm hatayla çalıştığını göstermiştir. Haberleşmede kullanılan zaman diliminin 100 ms olarak alınması bu hata üzerinde etkilidir. Haberleşme zaman diliminin 100 ms alınmasının nedeni, haberleşme kayıplarının önüne geçmektir. Şekil 4.6' da takip algoritmasında araçların başlangıç anındaki durumları gösterilmiştir. Şekil 4.7' de ise takip algoritmasında araçların yolun sonundaki durumları gösterilmiştir.

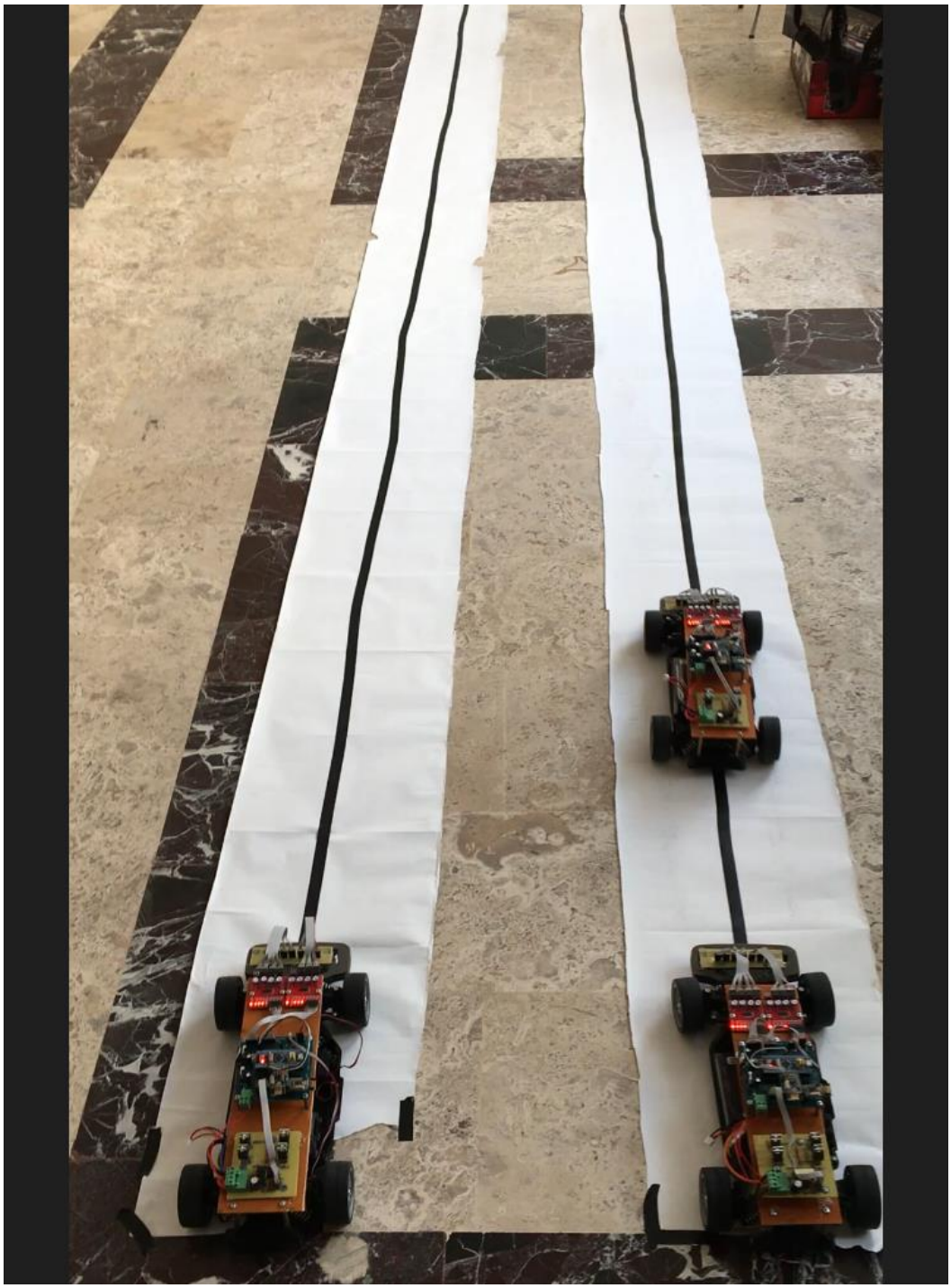
Şekil 4.8' de birleşme algoritması uygulamasında araçların başlangıçtaki durumları gösterilmiştir. Şekil 4.9' da araçların hareket ettikten sonraki durumları gösterilmektedir. Şekil 4.10' da araçların birleşme algoritmasına göre kavşak noktasına girişleri gösterilmiştir. Şekil 4.11' de iki farklı yoldan gelen araçların kavşak noktasında herhangi bir çarpma olmadan aynı yolda birleştikten sonraki durumları gösterilmiştir. Birleşme ve takip algoritmasına göre yapılan genel testler sonucunda sistemin +/- 5 cm hatayla çalıştığı tespit edilmiştir.



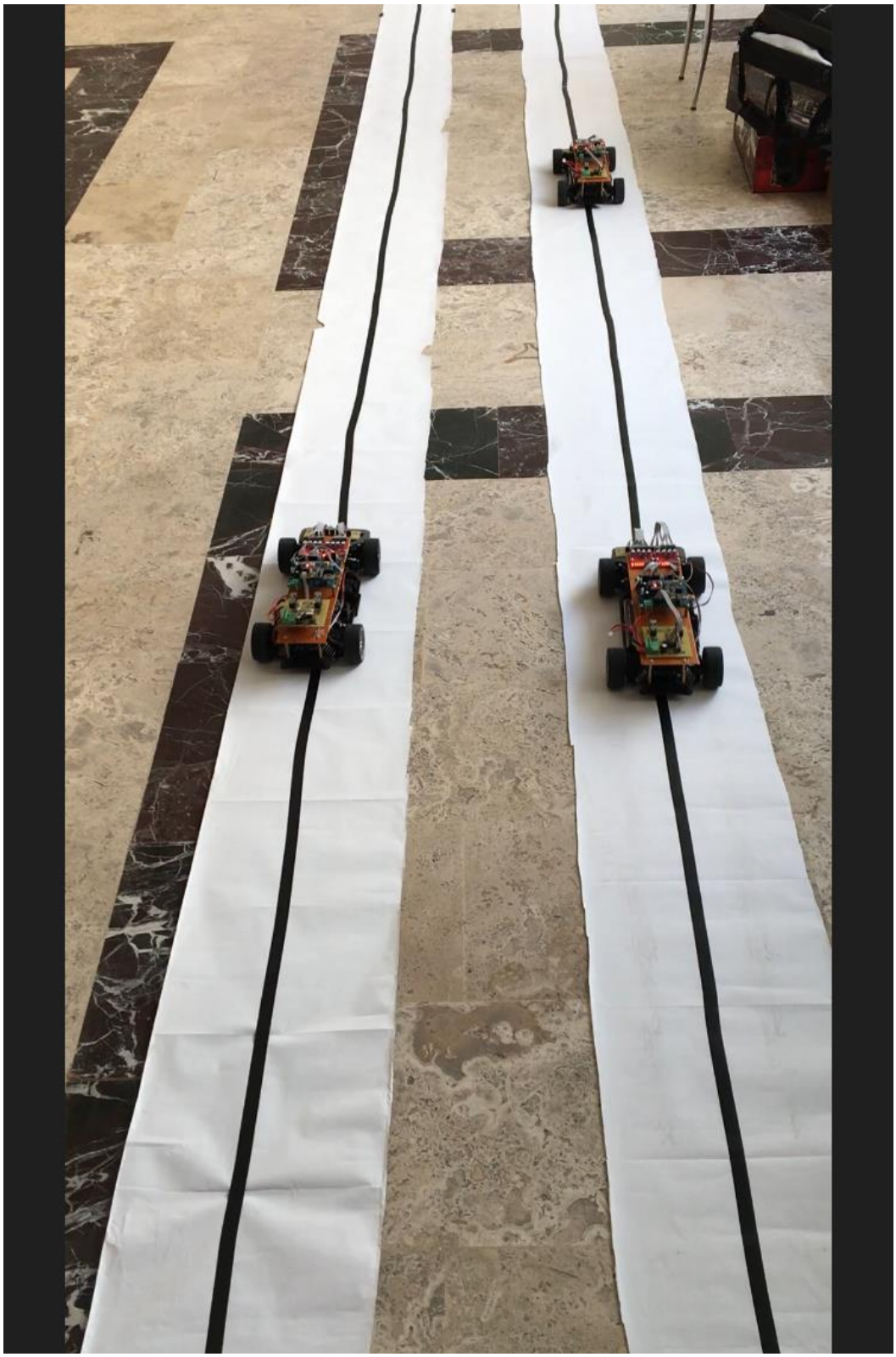
Şekil 4. 6 : Takip Algoritması Testi



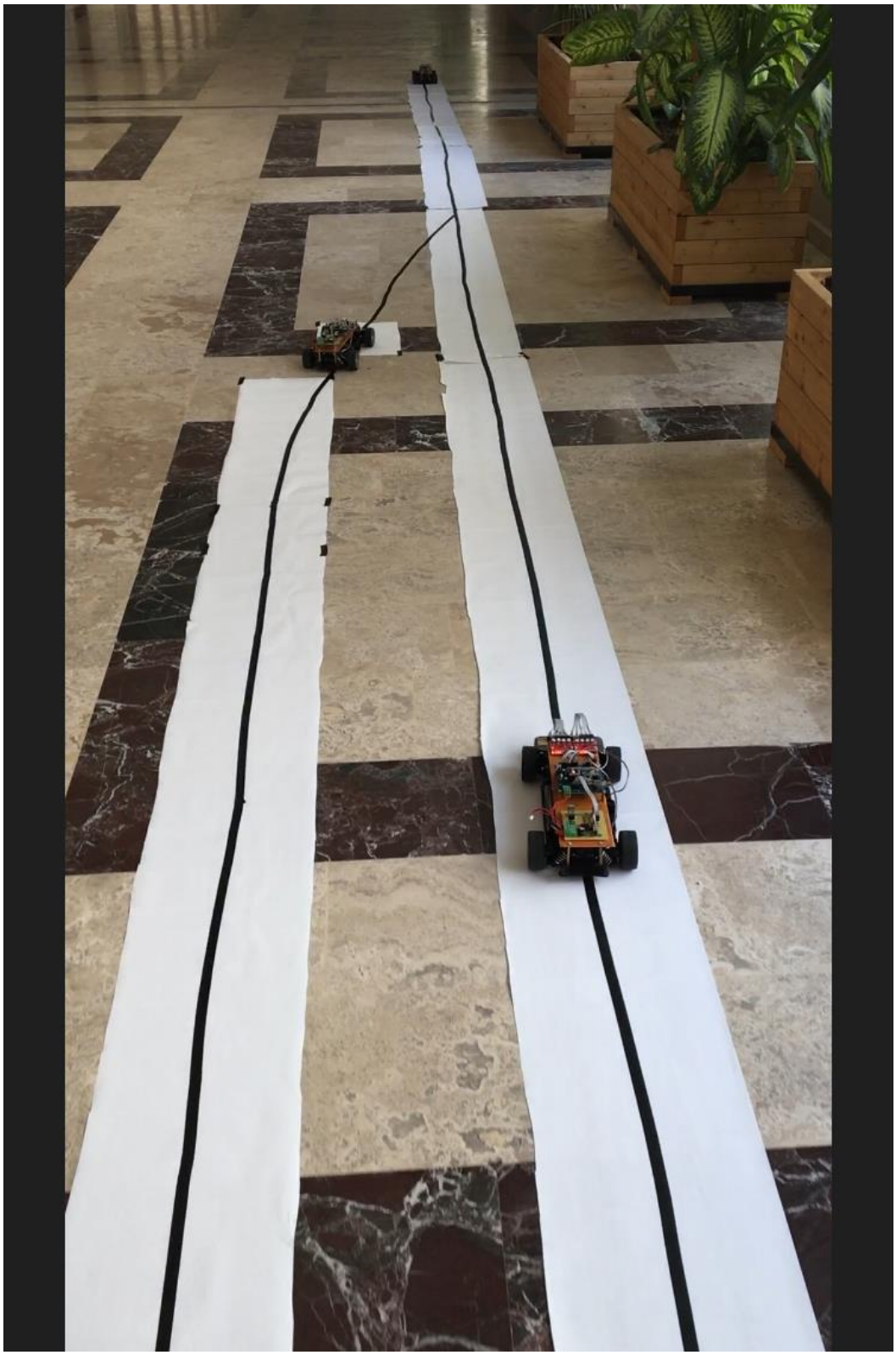
Şekil 4. 7 : Şekil 23 : Takip Algoritması Testi



Şekil 4. 8 : Birleşme Algoritması Testi



Şekil 4. 9 : Birleşme Algoritması Testi



Şekil 4. 10 : Birleşme Algoritması Testi

Şekil 4. 11 : Birleşme Algoritması Testi

5. SONUÇ

Yapılan tez çalışmasında iki ayrı yoldan sisteme giren araçların bir kavşak noktasına takip ve birleşme algoritmalarına göre güvenli bir şekilde girerek tek bir yolda seyahatlerine devam etmeleri sağlanmıştır. Çalışmada araçların seyahat yollarının ayrı olduğu ve bu yoldaki tüm araçların insansız araçlar olduğu kabul edilerek uygulama yapılmıştır. Tasarlanan sistemin altyapısı araçlara uygun olacak şekilde düşünülmüş çalışmalar buna göre yürütülmüştür. Uygulamada sisteme giren araçlar sistemdeki diğer araçlarla iletişim kurarak pozisyonu ve hızını belirlemekte ve sisteme kendini entegre etmektedir. Burada araçlardan alınan veriler C# ara yüz programına aktarılarak araçların uygulamadaki hareketleri, simüle edilip bilgisayar ortamında da görülmüştür. Sisteme giren tüm araçlar konumlarını ve hızlarını otomatik olarak ayarlamaktadır. Araçlar sisteme entegre olup harekete başladıktan sonra herhangi bir merkezi kontrol birimine gerek duymaksızın hareket edebilmektedirler. İhtiyaç duyulması halinde merkezi kontrol birimi tarafından sisteme ayrıca müdahale edilebilir. Araçların hız ölçümündeki hassasiyet, araçlar arasındaki haberleşme hızı ve araçların takip ve birleşme algoritmalarını uygularken kullandığı örnekleme zamanı sistemin doğru ve güvenilir bir şekilde işlemesi açısından çok önemli parametrelerdir. Sisteme giren herhangi bir aracın konumu sıfır değilse ve bu araç sisteme belirli bir noktadan dahil olacak ise bir merkezi sistem vasıtasıyla bu araç sisteme dahil edilmektedir. Sisteme belirli noktadan dahil olan aracın gerekli tüm bilgileri diğer araçlarla paylaşılmakta bu şekilde sisteme entegre edilmektedir.

Oluşturulan trafik düzeni içerisinde takip ve birleşme algoritmaları ile hareket eden araçların, yapılan uygulamalı testler sonucunda güvenli takip mesafesini +/- 5cm hata payı içerisinde koruduğu sonucuna ulaşılmıştır. Oluşan bu hatayı azaltmak için kullanılan haberleşme modülü yerine daha hızlı ve yüksek kapasiteli haberleşme modülleri kullanılabilir. Kullanılan işlemci yerine STM32F407xx, STM32F429xx veya Rasberry Pi gibi daha yüksek hızlarda çalışan ve daha gelişmiş işlemciler kullanılabilir. Hız ölçümünde daha hassas ölçümler yapılarak hatalar minimuma indirilebilir.

İleride yapılacak çalışmalarda yukarıda belirtilen ve geliştirilmesi gereken noktalara ek olarak, mevcut trafik düzeni içerisinde araç sayısı artırılması, kullanılan şerit sayısının artırılması ve en az bir kişinin seyahat edebileceği bir araç yapılması planlanmaktadır.

6. KAYNAK LİSTESİ

- Başkaya Z., Öztürk B. A., "Dal Kesme Yöntemi ve Bir Ekmek Fabrikasında Oluşturulan Araç Rotalama Problemine Uygulanması". *Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 24, 1, 101-114 (2005).
- CDIAC, "The United States Department of Energy's Carbon Dioxide Information Analysis Center for the United Nations". (2008)
- Chong L., Huapu L., "Study of Traffic Information Analysis and DecisionSupport System Based on Grid Computing, *International Journal of InformationTechnology*", 12(6), 69-77, (2006)
- Du Y., Gao X., "The new Navigation System for Automatic Guided Vehicle", (2008)
- Forret, A., Konca M., "Autonomous Cars and Society". (2007)
- Güvez H., Dege M., Eren T., "Kırıkkale'de Araç Rotalama Problemi ile Tıbbi Atıkların Toplanması". *International Journal of Engineering*, 4, 1, 41-45, (2012).
- Lari A., Douma F., Onyiah I., "Self-Driving Vehicles:Current Status of Autonomous Vehicle Developmentand Minnesota Policy Implications", Minnesota Üniversitesi, (2014)
- Lin C, Chou S.Y., Hsu H.Y., "Developing adaptive driving route guidance systems based on fuzzy neural network", IEEE International Conference on Systems, Man and Cybernetics (SMC 2009),San Antonio, USA, (2009).
- Ozguner, U., Acatman, T., Redmil, "*Autonomous Ground Vehicles*", (2011) [Book review]
- Paruchuri P., Pullalarevu A.R., Karlapalem K., "Multi Agent Simulation of Unorganized Traffic", International Conference on Autonomous *Agents*,ss : 176 – 183, (2001)
- Wahle J., Annen O., Schuster C., Neubert L., Schreckenberg M., "A Dynamic Route Guidance System Based On Real Traffic Data", *Europen Journal of Operation Research*, 131(2), 302-308, (2001)

- https://www.st.com/resource/en/datasheet/cd00161566.pdf
- https://www.sparkfun.com/datasheets/Components/General/Hall-US1881EUA.pdf
- https://www.sparkfun.com/datasheets/Components/SMD/nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf
- WEB_1, http://www.computerhistory.org
- WEB_2, http://www.autonomouscars.com
- WEB_3,http://wot.motortrend.com/volvo-autonomous-car-convoys-could-be-reality-2020-80731.html
- Kishalaya, Kundu.(2018). Google's Waymo Self-Driving Vehicle Involved in Nasty Car Crash in Arizona. Erişim adresi : https://beebom.com/googles-waymo-self-driving-vehicle-involved-in-crash-in-arizona/
- Can, T.(2017). Sürücüsüz Otomobil Şehri K-City. Erişim adresi : https://www.log.com.tr
- Kerem, C. (2018). Uber'in insansız aracı ölümlü kazaya sebep oldu. Erişim adresi : https://tr.euronews.com/2018/03/19/uber-in-insans-z-arac-olumlu-kazaya-sebep-oldu
- Ozan, Ö. (2018). Temsa imzalı sürücüsüz otobüsün çıkış tarihi belli oldu. Erişim adresi : https://www.log.com.tr/temsa-imzali-surucusuz-otobusun-cikis-tarihi-belli-oldu/
- Ali, G. (2018) Trafik ışığı ve trafik işaretlerini tarihe karıştıran teknoloji. Erişim adresi : https://www.log.com.tr/trafik-isigi-ve-trafik-isaretlerini-tarihe-karistiran-teknoloji-video/
- https://www.elektrikce.com/dogru-akim-dc-motor-cesitleri/
- Bozuyla, M., Tola, T. A., Yetis, S. M., "A Novel Safe Merging Algorithm for Connected Vehicles Using NetLogo", *Elektronika ir Elektrotechnika*, 24, 3, 3-7, (2018).

7. ÖZGEÇMİŞ

Adı Soyadı	:	CİHAT DURMUŞ
Doğum Yeri ve Tarihi	:	VAN / 10.05.1991
Lisans Üniversite	:	PAMUKKALE ÜNİVERSİTESİ
Y. Lisans Üniversite (varsa)	:	PAMUKKALE ÜNİVERSİTESİ
Elektronik posta	:	cihatdurmuss@gmail.com
İletişim Adresi	:	Karabük Üniversitesi Rektörlüğü C blok Yapı İşleri ve Teknik Daire Başkanlığı
Yayın Listesi	:	
Konferans listesi	:	