T.C.
KARABÜK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
MEKATRONİK MÜHENDİSLİĞİ

DENGE ROBOTU

BİTİRME TEZİ

Hazırlayanlar

Kağan ZORLUOĞLU 2014010225063

Kadir SEVİNÇ 2013010225037

Tez Danışmanı

Prof. Dr. Gökhan GÖKOĞLU

KARABÜK-2019

İÇİNDEKİLER

Sayfa

ÖZET

Bu çalışma PID kontrolörlü kendini dengeleyen robot üzerine yapılmıştır. Denge robotları hakkında bilgi toplanmış olup , kullanılan çeşitli materyaller ve PID kontrol teorisi hakkında teknik bilgiler verilmiştir.

KABUL VE ONAY

Kadir SEVİNÇ
Kağan ZORLUOĞLU tarafından hazırlanan "DENGE ROBOTU" başlıklı bu tezin Lisans Bitirme Tezi olarak uygun olduğunu onaylarım. 21./05/2019

Tez Danışmanı

Prof.Dr. Gökhan Gökoğlu

Bu çalışma, jürimiz tarafından oy birliği / oy çokluğu ile Mekatronik Mühendisliği Anabilim Dalında Lisans Bitirme Tezi olarak kabul edilmiştir 21../05/2019

Tez Jürisi

Başkan: Prof.Dr. Gökhan Gökoğlu

Üye : Dr. Öğr. Ü. Kenan Işık

Üye : Doç. Dr. İbrahim GAYIROĞLU

KBÜ Mühendislik Fakültesi, Mekatronik Mühendisliği Mezuniyet Komisyonu ve Bölüm Başkanlığı bu tezi Lisans Bitirme Tezi olarak onamıştır. 21./05/2019

Doç.Dr. İbrahim ÇAYIROĞLU
Mekatronik Müh. Bölüm Bşk.

ÖNSÖZ

Bugüne kadar çalışmalarımızda bizlere yol gösteren, yardımlarını esirgemeyen ve fikirlerini paylaşan tez danışmanımız Sayın Prof. Dr. Gökhan GÖKOĞLU'na,
bizlerin birer mühendis adayı olmasını sağlayan tüm saygıdeğer öğretmenlerimize ve her zaman yanımızda olan, her daim bizleri destekleyen ailelerimize teşekkürlerimizi ve minnettarlığımızı sunarız.

Kağan ZORLUOĞLU
Kadir SEVİNÇ

BÖLÜM 1

ROBOT SÖZCÜĞÜ VE TARİHÇESİ

1.1. ROBOT SÖZCÜĞÜ

Robot, otonom veya önceden programlanmış görevleri yerine getirebilen elektro-mekanik bir cihazdır. Güncel tanımı ile robotlar, elektronik ve mekanik birimlerden oluşan, algılama yeteneğine sahip olan ve programlanabilen cihazlardır. Başka bir tanımla robotlar, canlıların işlevlerini ve davranışlarını taklit edebilen, fiziksel yeteneklere ve yapay zekâya sahip, disiplinler arası öğeler içeren mühendislik ürünleridir[1]. Robot kelimesi Karel Čapek tarafından Çek dilindeki, hizmet eden manasına gelen "robota"dan türetilmiştir. Robot kelimesini ilk olarak Çek yazar Karel Čapek 1920'de yazdığı R.U.R. (Rossum's Universal Robots) adlı tiyatro oyununda kullanmıştır.

1.2. ROBOTUN TARİHÇESİ

Robot kelimesi ilk olarak 1920 yılında kullanılmış olsa da, robotlara ait ilk kavramlar ve robot benzeri ilk makinalara ait bilgiler M.Ö 3000 yıllarına kadar uzanmaktadır. Eski Mısır, eski Yunan ve Anadolu medeniyetlerinde otomatik su saatleri benzeri makinaların geliştirildiği bilinmektedir. Homerus'un İlyada eserinde insan yapımı kadın hizmetçiler anlatılmaktadır. M.Ö 100 yıllarında yaşamış olan İskenderiye'li bir mühendisin otomatik açılan kapılar, fıskiyeler vb. düzenekleri su ve buhar gücü ile çalıştırdığı eski kitaplarda yazılmaktadır. Daha yeni zamanlarda Leonardo da Vinci'nin yürüyen mekanik aslanı olduğu söylenmektedir. Batı dünyasında iyi bilinmeyen El Cezeri'nin robot teknolojisi konusunda fazla sayıda ve zamanına göre oldukça ileri uygulamaları bulunmaktadır. 1942 yılında Isaac Asimov ünlü serisiyle teknolojik açıdan tutarlı bir robot kavramı yaratır ve robotların amacının insana hizmet olduğunu, bir robotun kendi amaçlarını insanların amaçlarına karşı hiçbir zaman tercih edemeyeceğini belirttiği 3 adet Robot Yasasıyla belirler. Bu robot yasaları şu anda insan ve robot arasındaki ahlaksal ve hukuksal ilişkinin temelini oluşturur.

3 Robot Yasası ;

1.Bir robot bir insana zarar veremez ya da kayıtsız kalarak bir insanın zarar görmesine neden olamaz.

2. Birinci yasayla çatışmamak koşuluyla, bir robot insanlar tarafından verilen emirlere uymak zorundadır.

3. Birinci ve ikinci yasayla çatışmamak koşuluyla bir robot kendi varlığını korumalıdır.

BÖLÜM 2

KONTROL YÖNTEMLERİ VE KENDİNİ DENGELEYEN ROBOT

2.1. P - PI - PD - PID (Proportional - Integral - Derivative) Kontrol

PID içlerinde en yaygın kontrol türlerinden birisi olup , sistem hatasının üç ayrı matematiksel işlemden geçirilmesinin ardından toplanmasıyla sistem çıktısı oluşturulur. "PID" kelimesi bu üç ayrı matematiksel işlemin baş harflerinin bir araya getirilmesidir ve her bir işlemin etkisi farklıdır.

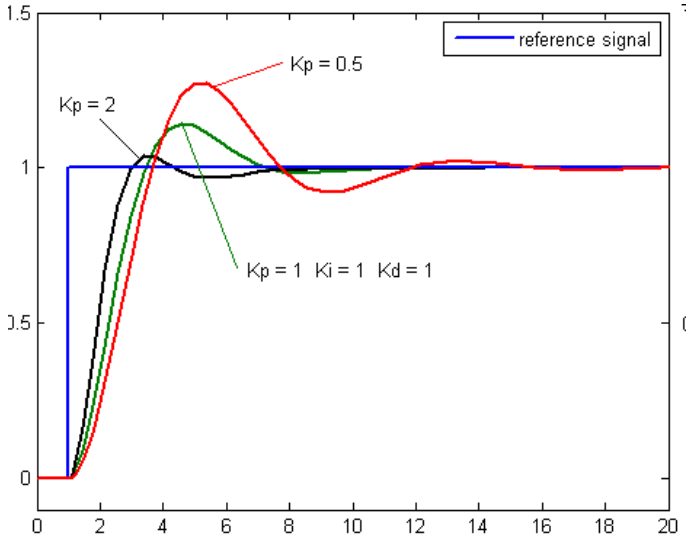
Buradaki P (Proportional), Oransal etkiyi temsil eder ve çıkışta sistemin hatasının belirlenmiş bir kazanç değeri ile çarpımı kadar çıkışı etkiler.

I (Integral) , integral etkiyi temsil eder. Sistem kontrolü başladığı andan itibaren hesaplamanın yapıldığı ana kadar geçen süredeki toplam hataların orantılı etkisidir ve sistemin geçmişte yapmış olduğu hataları ifade eder.

D (Derivative) , türevsel etkiyi temsil eder ve hatanın değişimi ile doğru bir orantıyla etkisini gösterir. Sistemde kullanılacak olan kontrolcü ise sistemin karakteristiğine göre seçilir.

2.1.1. Oransal (P) Kontrol

Bu kontrol türünde çalışma aralıksızdır ve sistemin enerji ihtiyacı değişiklik gösterebilir. Kontrolü sağlayan birim, ölçme elemanından edindiği veriye göre sürücüyle iletişim halinde bulunur. Sürücüde gücü sağlayan birime giren enerjiyi denetler. Ölçümü yapan eleman denetlenen değişkeni sürekli ölçer ve kontrol elemanına ölçümle ilgili sinyali gönderir. Sistemin belirli değerinde bir değişiklik olduğunda ölçüm elemanı kontrol elemanına sinyal gönderir ve kontrol elemanı bu sinyalle gelen bilgiyi referans değer ile karşılaştırıp bu döngüyü sistemde enerji olduğu takdirde devam ettirir. Hata azaldıkça kontrol etkisi azalır ve çıkış sinyalinin referans sinyale yavaşça yaklaşmasını sağlar fakat hiçbir zaman çıkış sinyali referansa ulaşamaz, Şekil 2.1'de olduğu gibi.

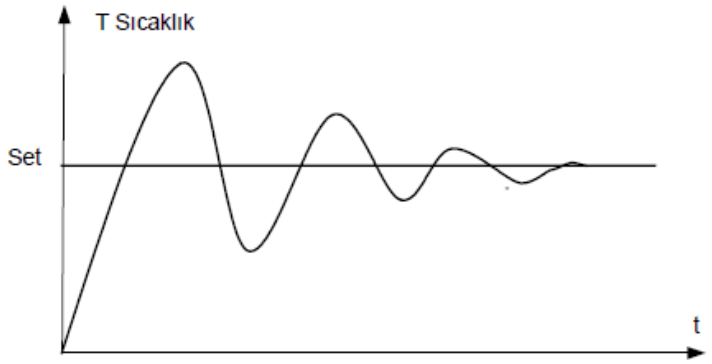


Şekil 2.1 oransal kontrol çıkış sinyalleri

Oransal kontrol genellikle sistem tepkisini hızlandırır , düzenli durum hatasını azaltır ve aşma değerini artırır.

2.1.2.Oransal - Integral (PI) Kontrol

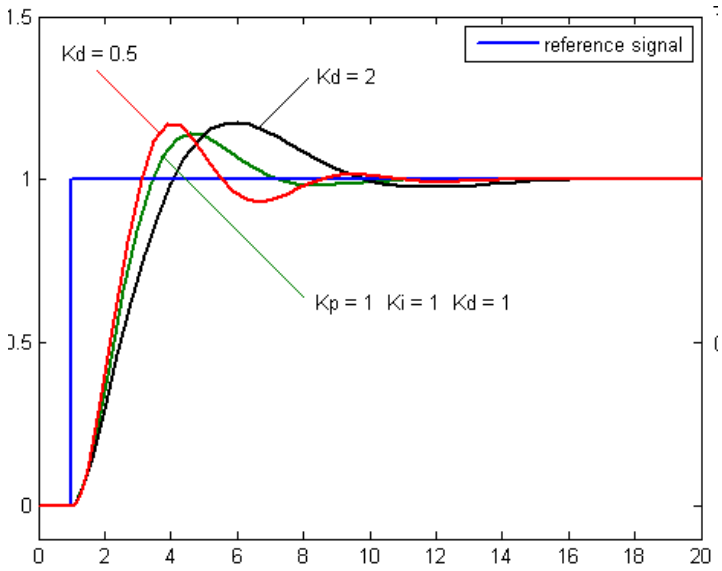
Oransal -İntegral denetleyici sistemin yükselme zamanını (Ki) azaltır ve yerleşme zamanını artırır. Sistemdeki kararlı hâl hatasını yok eder. Sistem değişkeninin ölçülen değeri ile set değeri arasındaki farkın sinyalinin zamana göre integrali alınır ve bu değer fark değer ile toplanır, oransal değer bandı kaydırılır. Bu şekilde sisteme verilmekte olan enerji miktarı otomatik olarak değiştirilebilir ve kontrol değişkeni tam olarak set değerine gelir. Bu yöntemin en belirgin sorunu ise , sistemin ilk çalışmaya başlamasında kontrol değişkeni beklenen değerin fazlasıyla üzerine çıkar ve bu duruma aşırma(overshoot) denir. Ardından ilk yapacağı salınımda ise beklenen değerin çok altına düşer ve bu duruma ise undershoot denir. İntegral etkisi ise oransal etkinin hesaplanmasına benzer şekilde hesaplanır, anlık hata değeri yerine sistemin aktif olduğu andan itibaren tüm zamandaki hataların toplamı olan kazanç ile çarpılır. Sistemin cevabının referans değerine ulaşması geciktiği süre boyunca integral kontrolünün etkiside artacaktır.



Şekil 2.2 PI kontroldeki overshoot örneği

2.1.3.Oransal - Türevsel (PD) Kontrol

Türevsel kontrolün etkisi sistemdeki hatanın değişimine orantılı olarak belirlenir. Türev işlemi, bir sistemin çıktısının hesaplandığı andan bir sonraki anda alacağı değere ilişkin bir değer çıkarır ve bu şekilde sistemdeki çıktıya göre bir öngörü kazandırır. Türevsel kontrolün etkisi, sistemdeki dalgalanmalar ile orantılı olduğundan dalgalanmalar büyük olursa sistem üzerindeki türevsel kontrol de büyük olur. Dolayısıyla sistemin çıktısı daha az dalgalı olacaktır. Türevsel kontrolün salınımları azaltma özelliği oransal kontrolün sebep olduğu aşma değerlerindeki artışı azaltır. Genellikle türevsel-oransal kontrolcüler sistemin aşma değerini azaltıp , sistemin reaksiyon hızını artırır.



Şekil 2.3 Türevsel kontrolün etkisi

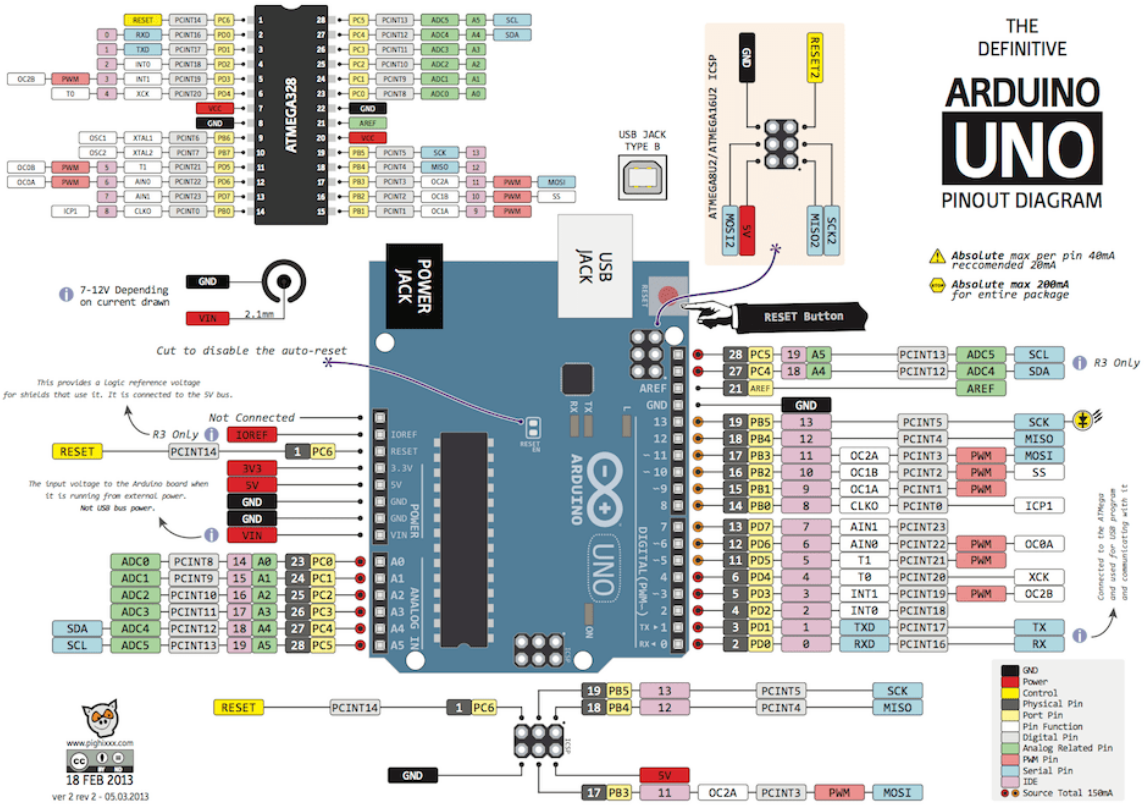
2.1.4.Oransal -Integral-Türevsel (PID) Kontrol

Oransal komponentin yükselme zamanını azaltmada etkisi vardır fakat çıkış sinyalini asla referans değere oturmasını sağlamaz. Integral komponentin ise kararlı hal hatasını sıfırlamakta etkilidir fakat yapısı gereği geçici etkiyi kötüleştirir. Türevsel komponent, değişkenin hızla yükseldiği takdirde çıkış sinyalini dengeler dolayısıyla sistem daha kararlı olur ve aşımı azaltır. Bu üç etkininde bir arada olması durumunda , sistemdeki kontrol etkisi genel olarak iyileşir. Parametreler iki şekilde ideal duruma getirilir. Bunlardan birincisi ; deneme-yanılma metodudur. İntegral (I) ve türevsel (D) parametreleri sıfır alınır, oransal (P) parametresi dilenen cevap hızına göre ayarlanır. Oransal parametre ayarlandıktan sonra integral parametresi salınımları durdurması için kademeli olarak artırılır ve en az kararlı hal hatasının olduğu duruma gelene kadar ayarlamalar yapılır. Oransal ve İntegral parametreleri ayarlandıktan sonra geriye kalan tek parametre olan türevsel parametresi ise sistemdeki aşım değeri-hız oranı kabul edilebilir bir noktaya gelinceye kadar artırılır. İkinci metod ise Ziegler-Nichols metodudur.

2.2. KENDİNİ DENGELEYEN ROBOT

2.2.1. Arduino Uno Rev3

Arduino, programlayarak bir çok işlemi yaptırabileceğimiz mikrodenetleyici platformudur. Gücü USB üzerinden ya da harici bir AC veya DC güç kaynağından alabilir. Çalışması için sürekli olarak USB'nin bilgisayara bağlı olması zorunlu değildir. Sadece batarya ya da sadece adaptör ilede çalıştırılabilir. Adaptör karta , kartın üzerindeki 2,1milimetrelik merkez pozitif güç soketinden ya da GND ve Vin pinlerinden bağlanabilir. Güç kaynağı olarak 6V-20V arası kullanılabilir ancak kart için önerilen besleme gerilimi 7V-12V'dur. Arduino'nun regülatörünün 7 voltun altında stabil olarak çalışmayacağı için öneriler değerler bu şekildedir. Kartın üzerindeki mikrodenetleyici 5V gerilim ile çalışır. Arduino Uno Rev3 üzerinde 14 dijital input/output pini bulunan ATmega328P tabanlı bir mikrodenetleyicili karttır. Ayrıca 6 analog giriş , USB bağlantı portu , güç jak'ı, devre içi seri programlama ve reset butonu mevcuttur.

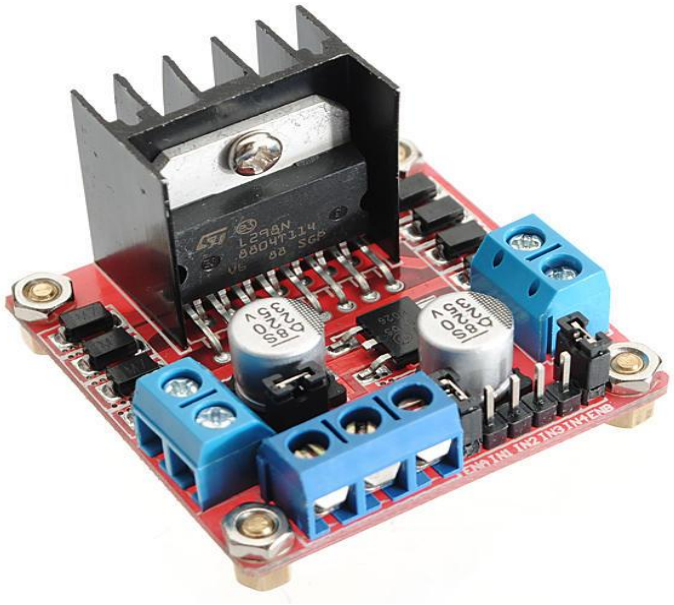


Şekil 2.4 Arduino Pinout Diyagramı

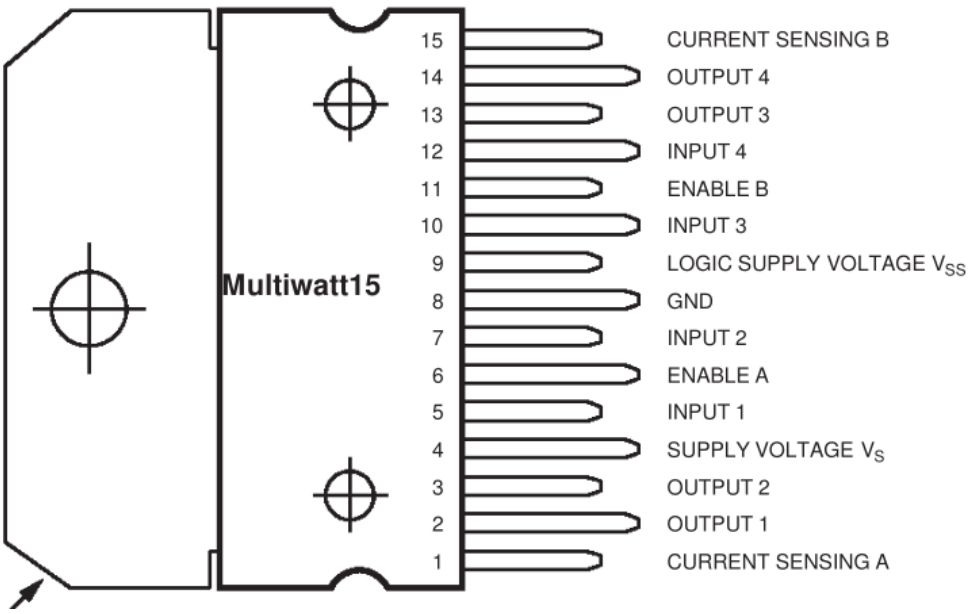
Arduino Uno Rev3'ün teknik özellikleri :

- ATmega328P Mikrodenetleyicisi
- 5V Çalışma gerilimi
- 7-12V Önerilen giriş gerilimi ve 6-20V Giriş gerilimi limiti
- 14 Dijital Input/Output pini ve bunlardan 6 adeti PWM output
- 6 Adet PWM dijital Input/Output pinleri
- 6 Adet dijital input pini
- Her Input/Output pini için 20mA DC akım
- 3.3V Pin için 50mA DC akım
- 0.5 KB'ı bootloader tarafından kullanılan 32KB'lık flash hafıza
- 2KB SRAM
- 1KB EEPROM
- 16MHz Saat hızı
- 68,6mm uzunluğunda ve 53,4mm genişliğinde , 25g ağırlığında

2.2.2. Voltaj Regülatörlü Çift Kanallı L298N Motor Sürücü



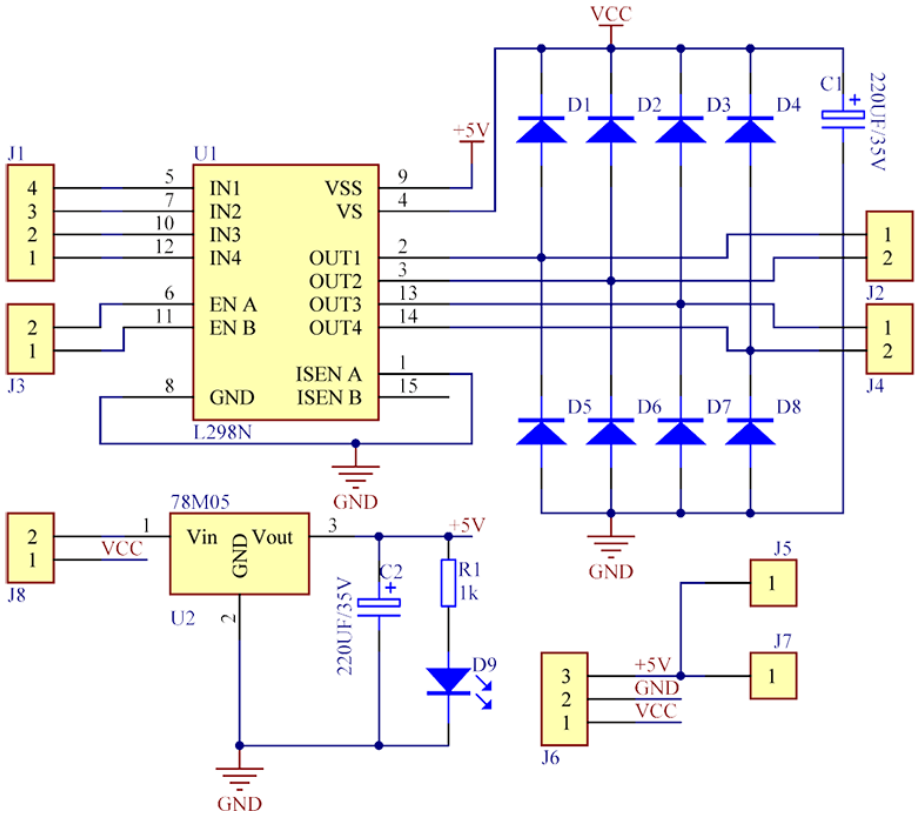
24V'a kadar olan motorları sürmek için hazırlanmış olan bu motor sürücü kartı, iki kanallı olup, kanal başına 2A akım vermektedir. Kart üzerinde L298 tabanlı olan L298N motor sürücü entegresi kullanılmıştır. DC motorlardan ayrı olarak step motor kontrolüne de imkan sağlamaktadır. Birbirinden bağımsız iki adet motoru kontrol edebilir. Kartın üzerinde dahili regülatörü ve soğutucusu bulunmaktadır. Yüksek sıcaklığa ve kısa devreye karşı koruması vardır. Akım okuma pinleri dışarıya verilmiştir. Kartı dilenen yere sabitlemek için 4 adet delik bulunmaktadır.



Şekil 2.5 L298N'in pin bağlantıları

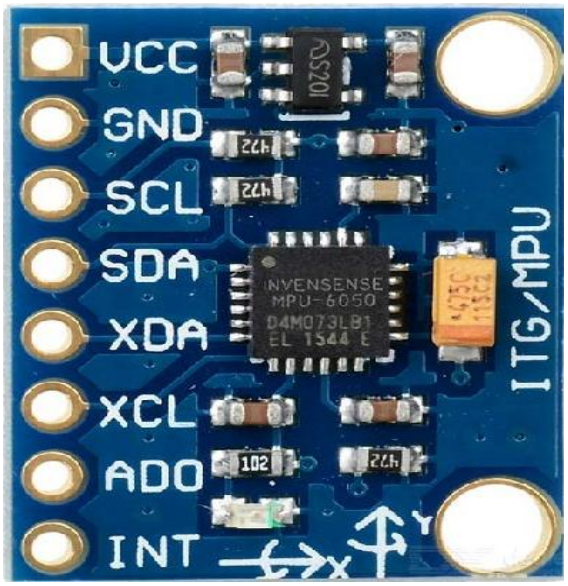
Sürücünün Özellikleri :

- 40V'a kadar çıkabilen yüksek çalışma gerilimi
- 3A'e kadar anlık çıkabilen yüksek çıkış akımı
- 25W'lık anma gücü
- DC ve Step motor sürmek için dahili 2 adet H-Köprüsü, yüksek voltaj, motorların dönüş yönünü kontrol etmeyi sağlayan tam köprü sürücüsü
- Kontrol için standart lojik sinyal seviyesini kullanması
- İki yada dört fazlı step motor kullanabilir
- İki fazlı DC motor kullanabilir
- Bir indüktif yükün yaratacağı ters akımdan, devreye bağlı cihazları koruması için yüksek kapasiteli filtre kapasitörü ve flyback diyotu bulundurması
- 5-35V sürücü gerilimi
- 5V Lojik gerilim
- 42mm x 42mm boyutlarında olması



Şekil 2.6 Motor sürücünün şematik diyagramı

2.2.3. MPU 6050 6 Eksenli Gyro ve İvme Ölçer



MPU-6050 çeşitli hobi, multicopter ve robotik projelerinde sıklıklı kullanılan üzerinde 3 eksenli bir gyro ve 3 eksenli bir açısal ivme ölçer bulunduran 6 eksenli bir IMU sensör kartıdır. Kart üzerinde voltaj regulatörü bulunduğundan 3 ile 5 V arası bir besleme voltajı ile çalıştırılabilir. İvme ölçer ve gyro çıkışlarının her ikisi de ayrı kanallardan I²C çıkışı vermektedir. Her eksende 16 bitlik bir çözünürlükle çıkış verebilmektedir. Pinler arası boşluk standart olarak ayarlandığı için breadboard veya farklı devre kartlarında rahatlıkla kullanılabilir.

Özellikleri :

- Gyro ölçüm aralığı : ±250, ±500, ±1000, ±2000 °/saniye
- Gyro hassasiyeti : 131, 65.5 , 32.8 , 16.4 LSB/ °/saniye
- Gyro gürültü yoğunluğu oranı : 0.005 , 0.005 , 0.005 , 0.005 dps/√Hz
- Açısal ivmeölçer ölçüm aralığı : ± 2 ± 4 ± 8 ± 16 g
- Açısal ivmeölçer hassasiyeti : 16384 , 8192 , 4096 , 2048 LSB/g
- İletişim türü : Standart I²C
- Lojik çalışma gerilimi : 1,8V± %5 ya da VDD
- Çalışma gerilimi : 2,375V-3,46V ± %5
- Boyutları : 4x4x0,9mm

dps = Gyro ölçüm aralığı

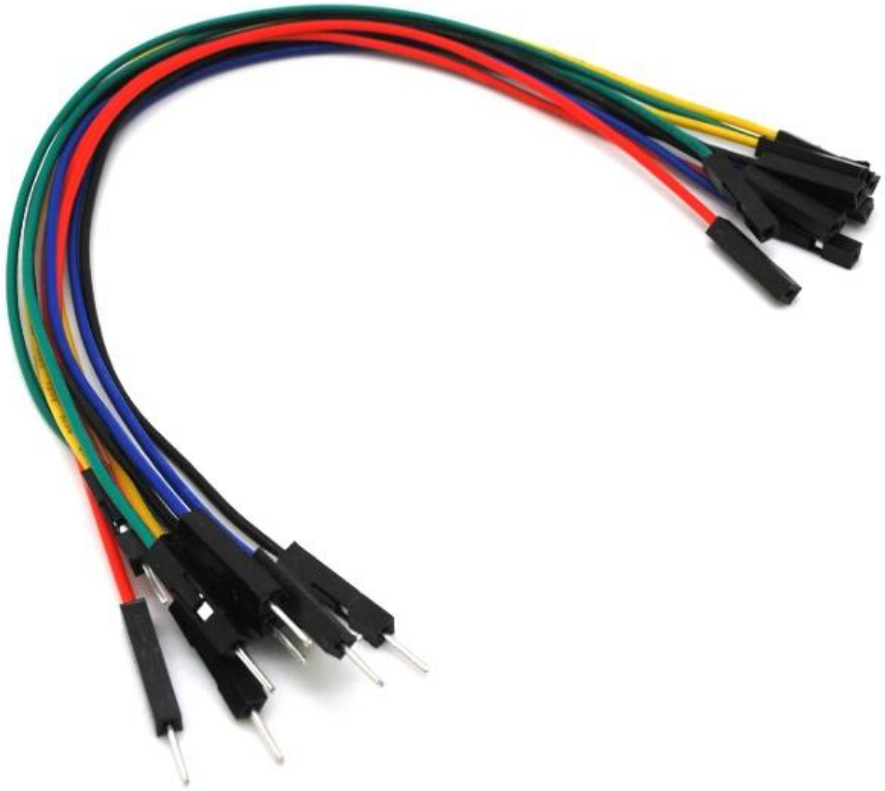
2.2.4. 6V , 250Rpm Plastik Redüktörlü DC Motor ve Tekerlek

Özellikleri :

- Çalışma gerilimi : 3V-6V
- Maksimum tork : 800gF/cm
- Hız : 250Rpm (yüksüz ve 6v gerilimdeyken)
- Yük akımı : 70mA-250mA
- Motor ağırlığı : 29g
- Tekerlek ağırlığı : 45g
- Tekerlek çapı : 70mm
- Tekerlek genişliği : 30mm

2.2.5. M5 Gijon ve M5 Somun

2.2.6. Jumper Kablolar



2.2.7. Li-Po Batarya

2.2.8. Potansiyometre

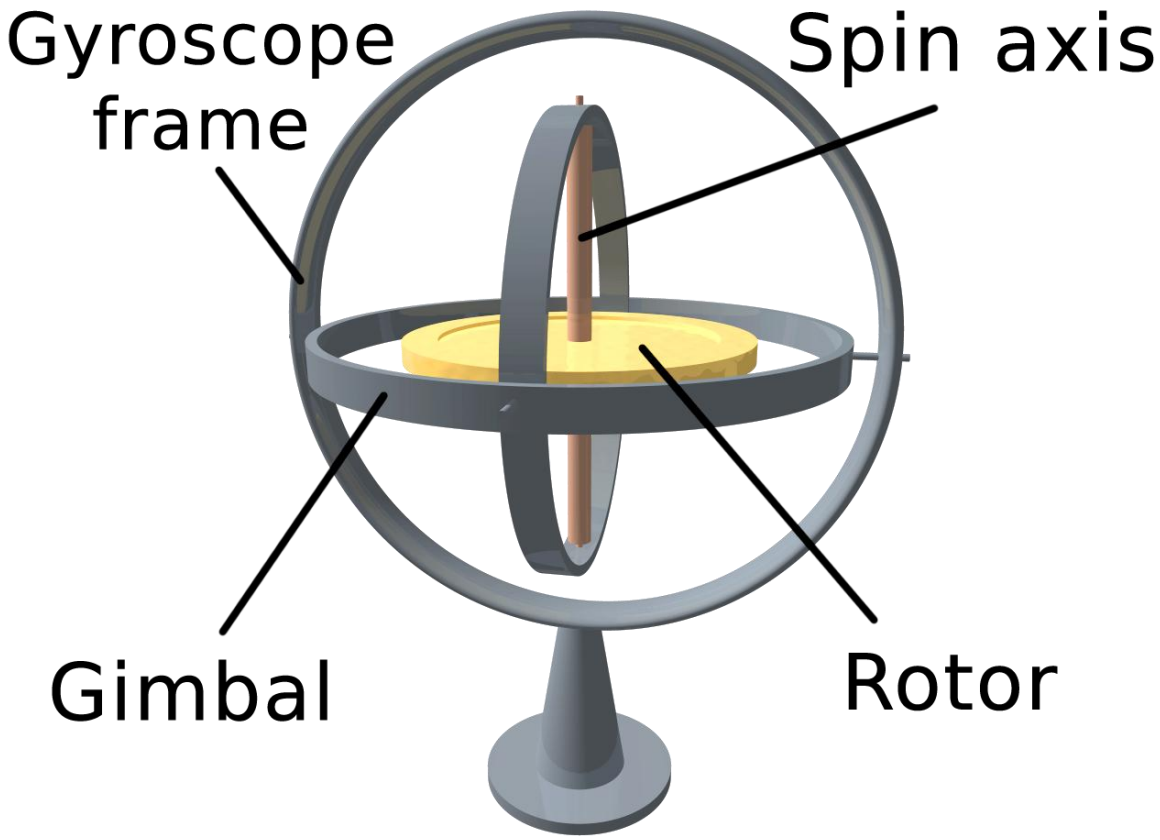
Potansiyometre , reosta olarakta bilinen bir çeşit direnç türüdür. Üç farklı bacağı bulunur. Yaygın olarak radyolarda ses ayarlamalarında , eski televizyonlarda kontrast, parlaklık vb. şeyleri değiştirmek için kullanılmıştır, fakat bu projede PID parametrelerinin (Ki, Kd, Kp) ayarlanması için kullanılacaklardır.

Özellikleri ;

- Toplam döndürme açısı : $300^{o} \pm 10^{o}$
- Döndürme torku : 2 - 20 mN/m arası
- Dönüş durdurucu gücü : 0.3N/m
- İtme/çekme gücü : 80N
- Direnç : 10K Ohm
- Direnç toleransı : ± %20
- Dayanıklılık : 15000 çevrim
- Şaft Uzunluğu : 25mm
- Şaft Kalınlığı : 6mm
- Maksimum çalışma gerilimi : 150V AC

BÖLÜM 3
JİROSKOP TÜRLERİ VE KULLANIM ALANLARI

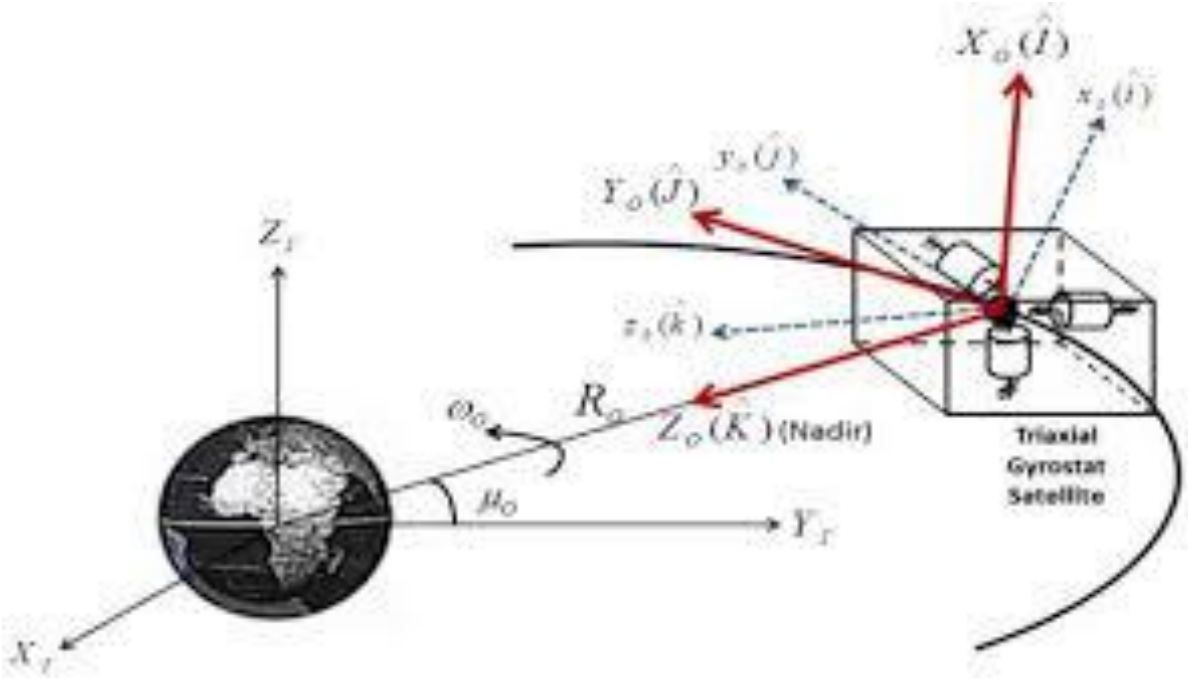
Jiroskop (sensör) , açısal hız sensörleri olarakta bilinirler , düzdönerlerin geliştirilmiş bir türüdür ve günlük hayatımızda farkında olmayıpta sıkça kullandığımız sensörlerden birisidir. Düzdöner ilk olarak 1817'de Johann Bohnenberger tarafından yapıldı ve açısal hızı ölçmek , dengesel durumu devam ettirebilmek amacıyla kullanılmıştır. Düzdöner , dönüş yönünü kendisi belirleyen bir çeşit disktir. Açısal momentumun korunması kuralından dolayı , dönerken , desteğinin dönüşünden ya da desteğin açısının değişiminden etkilenmez.



Şekil 3.1 Jiroskop

3.1. Jirostat

Jirostat , bir jiroskop türüdür. İlk jirostat Lord Kelvin tarafından tasarlanmıştır. Günümüzde uzay araçlarını ve uyduları yörüngelerine oturması için durum kontrolünde kullanılmaktadır.(şekil 3.2'de görüldüğü gibi)

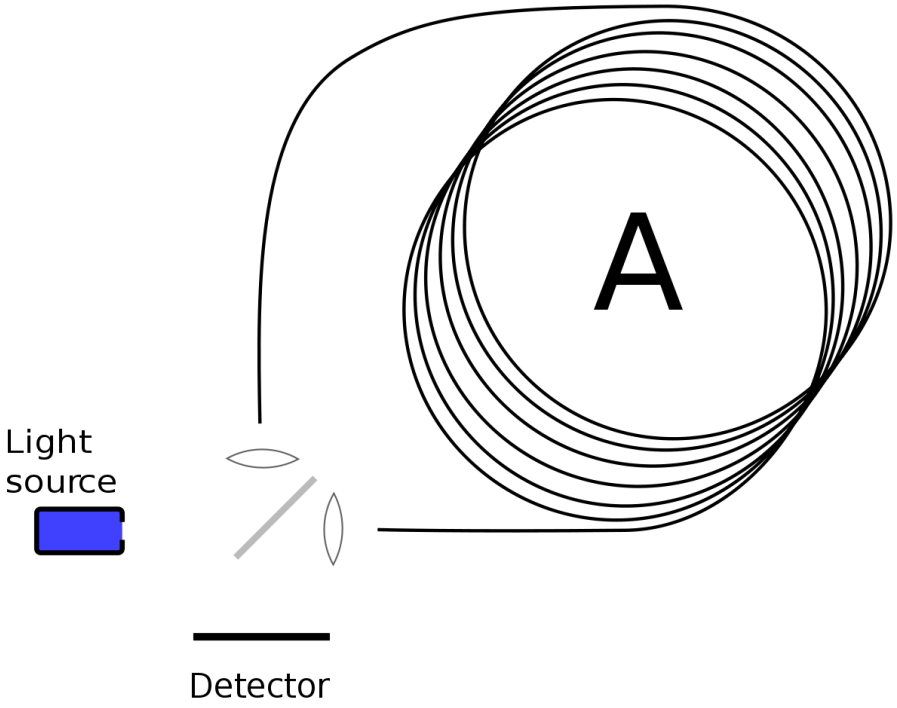


Şekil 3.2 Jirostatın Çalışması

3.2. Mikroelektronik-mekanik Sistem (MEMS)

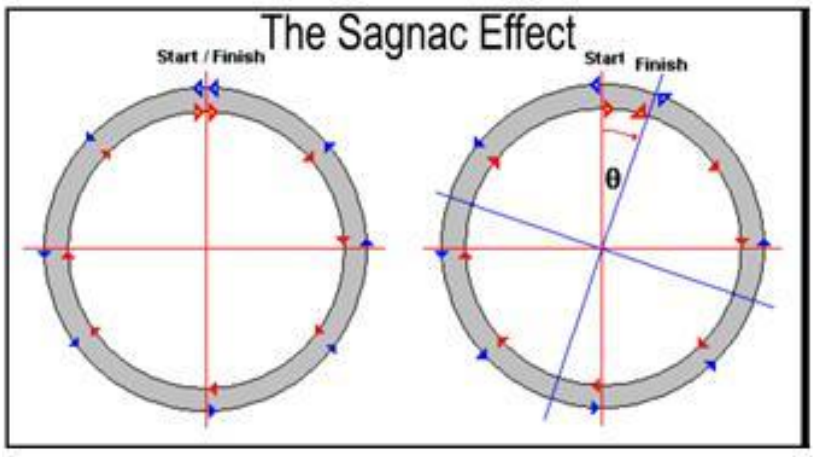
MEMS jiroskobu açısal hızı ölçebilmektedir. Titreyen bir yapı aracılığıyla dönüş açısını belirler. Çalışma prensibi, içerisinde bulunan titreyen obje , desteğinin yönü değişse dahi titremeye devam etmesidir. Objenin, Coriolis etkisiyle, desteği üzerine uyguladığı kuvvetin , büyüklüğünün ölçülmesiyle dönüş oranı belirlenir. Oldukça ucuz bir şekilde imal edilir ve günümüzde çok geniş bir alanda kullanılmaktadır. Bunlar cep telefonları , oyun konsolları , kameralar , hava yastığı sistemleri , görüntü stabilize etme gibi durumlardır.

3.3. Fiber Optik Jiroskop (FOG)



Şekil 3.3 Fiber Optik Jiroskop Çalışma Prensibi

Sagnac etkisiyle açısal durumdaki değişimi algılar, çalışma prensibi uzunluğu 5 kilometreye kadar çıkabilen fiber optik bobinin içerisinden ışığın geçerken yaptığı müdahaledir. Rotasyonal dönüş hakkında oldukça kesin değerler vermekle birlikte kapalı ve açık çevrimli sistemlerde kullanılabilmektedir. RLG'ye kıyasla daha fazla çözünürlüğü vardır. Yüksek performans beklenen uzay çalışmaları, askeri atalet navigasyon sistemlerinde, güdümlü füze sistemlerinde , otonom su altı araçlarında ve uzaktan kontrol edilen araçlarda kullanılmaktadır.



Şekil 3.4 Sagnac Etkisi

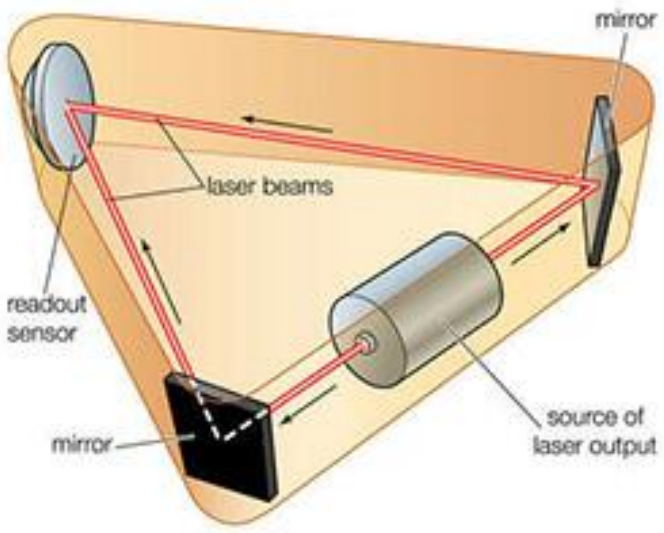
3.4. Yarı-Küresel Yankılayıcı Jiroskobu (HRG)

Şarap bardağı jiroskobu yada mantar jiroskobu olarakta bilinmektedir. Düşük gürültü ve yüksek performanslı rotasyon sensörüdür. Bir yarı küresel yankılayıcı jiroskobunu , ince ve yarı küresel bir kabuk ve kalın sap oluşturur. Kabuğu kaplayan kuartz yapılar, elektrotlar tarafından üretilen elektrostatik kuvvetler aracılığyla eğilme rezonansına itilir, jiroskopik etki , eğik duran dalgaların ataletinden elde edilir. Mekanik bir sistem olmasına rağmen hiç hareketli parçası bulunmamaktadır. Uzay araçları , uydular, uzay araçlarını fırlatan sistemler , hava saldırısının olacağı yeri belirleme , denizaltı , hava taşımacılığı gibi alanlarda kullanılmaktadır.

Şekil 3.5 Yarı-Küresel Yankılayıcı Jiroskobu

3.5. Halka Lazer Jiroskop (RLG)

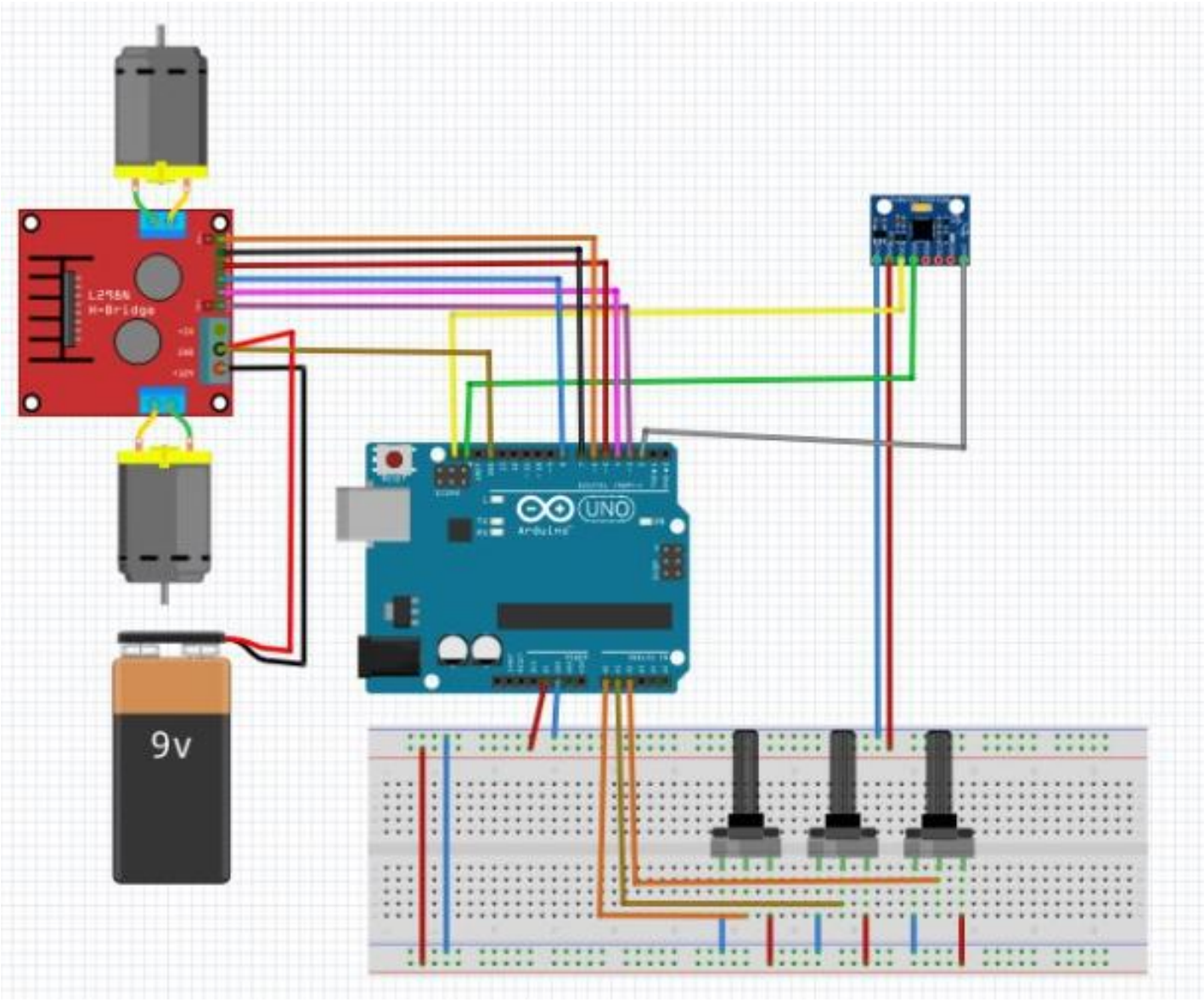
Sagnac etkisini temel alır. Fiber optik jiroskoba benzer şekilde çalışır. Bir halka lazerin aynı yol üzerinde karşılıklı olarak , birbirinden bağımsız iki yankı yaymasıdır. Bu yankı ışınları yayılırken aralarında bir frekans farkı oluşur, bu frekans farkı ise dönüşü belirler. Karşılık yönlerde yayılan ışınların arasındaki müdahale dışarıdan gözlemlenir ve bu dalgaların kalıplarında hareketlenme ile sonuçlanır. Dolayısıyla dönme olduğunu belirtir. Askeri ve yolcu uçakları , askeri helikopterler , anti-uydu füzeleri , uzay uygulamaları ve çeşitli güdümlü füze sistemlerinde kullanılmaktadır.



Şekil 3.6 Halka Lazer Jiroskobun Çalışma Prensibi

EKLER

Devre Şeması



KODLAR

```
//Kütüphanelerin tanımlanması ;
#include <PID_v1.h>
#include <LMotorController.h>
#include "I2Cdev.h"
#include "MPU6050_6Axis_MotionApps20.h"

#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
#include "Wire.h"
#endif

// Manuel Tuning ve Log Pid Constants 1 olursa potansiyometre üzerinden tanımlanır 0 olursa aşağıda verilen değer üzerinden tanımlanır

#define LOG_INPUT 0
#define MANUAL_TUNING 1
#define LOG_PID_CONSTANTS 1
#define MOVE_BACK_FORTH 0

#define MIN_ABS_SPEED 30 // Minumum Başlangıç Hızı

//MPU



MPU6050 mpu;

// MPU kontrol/durum değişkenleri
bool dmpReady = false;  // DMP init başarılı olmuşsa true değerini ayarlayın
uint8_t mpuIntStatus;   // MPU dn gerçek kesme durum baytı tutar
uint8_t devStatus;     // her cihazın çalışmasından sonraki durumu döndürür (0 = success, !0 = error)
```

```
uint16_t packetSize;    // beklenen DMP paaket büyüklüğü (varsayılan 42 bayttır)
uint16_t fifoCount;     // şuanda FIFO daki tüm bayt sayısı
uint8_t fifoBuffer[64]; // FIFO saklama baytı

// yönlendirme /hareket değişkenleri
Quaternion q;           // [w, x, y, z]         yönlendirme koordinatları
VectorFloat gravity;    // [x, y, z]            yerçekimi vektörü
float ypr[3];           // [yaw, pitch, roll]   yaw/pitch/roll koordinat ve yerçekimi vektörü


//PID


#if MANUAL_TUNING
  double kp , ki, kd;
  double prevKp, prevKi, prevKd;
#endif
double originalSetpoint = 174.29;
double setpoint = originalSetpoint;   //çift ayar noktası
double movingAngleOffset = 0.3;
double input, output;
int moveState=0; //0 = denge ; 1 = geri ; 2 = ileri

#if MANUAL_TUNING
  PID pid(&input, &output, &setpoint, 0, 0, 0, DIRECT);
#else
  PID pid(&input, &output, &setpoint, 70, 240, 1.9, DIRECT);
#endif
```

```
//MOTOR SURUCU PINLERININ TANIMLANMASI


int ENA = 3;
int IN1 = 4;
int IN2 = 8;
int IN3 = 5;
int IN4 = 7;
int ENB = 6;


LMotorController motorController(ENA, IN1, IN2, ENB, IN3, IN4, 0.6, 1);


//Zamanlama


long time1Hz = 0;
long time5Hz = 0;


volatile bool mpuInterrupt = false;     // MPU kesme pininin yüksek boşlukta olup olmadığını gösterir
void dmpDataReady()
{
   mpuInterrupt = true;
}


void setup()
{
```

```
 // 12C veriyoluna katıl (12C dev kütüphanesi bunu otomatik olarak yapmaz)
    #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
    Wire.begin();
    TWBR = 24; // 400kHz I2C clock (200kHz if CPU is 8MHz)
  #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
    Fastwire::setup(400, true);
  #endif

  // initialize serial communication (seri iletişimi başlatınız)
  // (115200 Teapot Demosu için gerekli olduğundan seçildi, fakat gerçekte size kalmış değerdir)
  Serial.begin(115200);
  while (!Serial); // Leonardo numaralandırma için bekleyin, diğerleri devam edecek

  // cihazı ilk haline getirmek
  Serial.println(F("I2C Kuruluyor..."));
  mpu.initialize();

  // bağlantısını doğrulayın
  Serial.println(F("Suruculer test ediliyor..."));
  Serial.println(mpu.testConnection() ? F("MPU6050 baglanti basarili") : F("MPU6050 baglanti basarisiz"));

  // DMP yi yükle ve yapılandır
  Serial.println(F("DMP kuruluyor..."));
  devStatus = mpu.dmpInitialize();

  // buraya gyro ofsetlerinizi tanımlayın , minimum hassasiyet için ölçeklendirildi
  mpu.setXGyroOffset(220);
  mpu.setYGyroOffset(76);
  mpu.setZGyroOffset(-85);
  mpu.setZAccelOffset(1788); // test çipim için 1788 üretim varsayılanı
  // çalıştığından emin olun  (returns 0 if so)
```

```
  if (devStatus == 0)
  {
     //  DMP yi açın, şimdi hazır
     Serial.println(F("Enabling DMP..."));
     mpu.setDMPEnabled(true);

// Arduino kesinti algılamsını etkinleştir
     Serial.println(F("Enabling interrupt detection (Arduino external interrupt 0)..."));
       //(kesintiyi algılamanın etkinleştirilmesi (Arduino haici kesinti 0))

     attachInterrupt(0, dmpDataReady, RISING);
     mpuIntStatus = mpu.getIntStatus();
     // DMP yi ready flag olarak ayarlayın böylece main loop() işlevi onu kullanmanın sorun olmadığını bilir
     Serial.println(F("DMP ready! Waiting for first interrupt..."));
     dmpReady = true;
// daha sonra karşılaştırma için beklenen DMP paket boyutunu al
     packetSize = mpu.dmpGetFIFOPacketSize();

     // PID kurulumu
     pid.SetMode(AUTOMATIC);
     pid.SetSampleTime(10);
     pid.SetOutputLimits(-255, 255);
  }
  else
  {
     // EHATA!
     // 1 = initial memory load failed (ilk bellek yüklemesi başarısız)
     // 2 = DMP configuration updates failed (DMP yapılandırma güncellemeleri başarısız)
     // (eğer kırılcaksa genelleklie kod 1 olacaktır)
     Serial.print(F("DMP Initialization failed (code "));
     Serial.print(devStatus);
     Serial.println(F(")"));
```

```
    }
}

void loop()
{
    // eğer programlama başarısız olursa hiçbir şey yapmaya çalışma
    if (!dmpReady) return;

// MPU kesintisi için bekleyin veya ek paket mevcut

    while (!mpuInterrupt && fifoCount < packetSize)
    {
        //MPU verisi yok–PID hesaplamaları yapılır ve motora çıkış sağlanır

        pid.Compute();
        motorController.move(output, MIN_ABS_SPEED);

        unsigned long currentMillis = millis();

        if (currentMillis - time1Hz >= 1000)
        {
            loopAt1Hz();
            time1Hz = currentMillis;
        }

        if (currentMillis - time5Hz >= 5000)
        {
            loopAt5Hz();
            time5Hz = currentMillis;
        }
    }
```

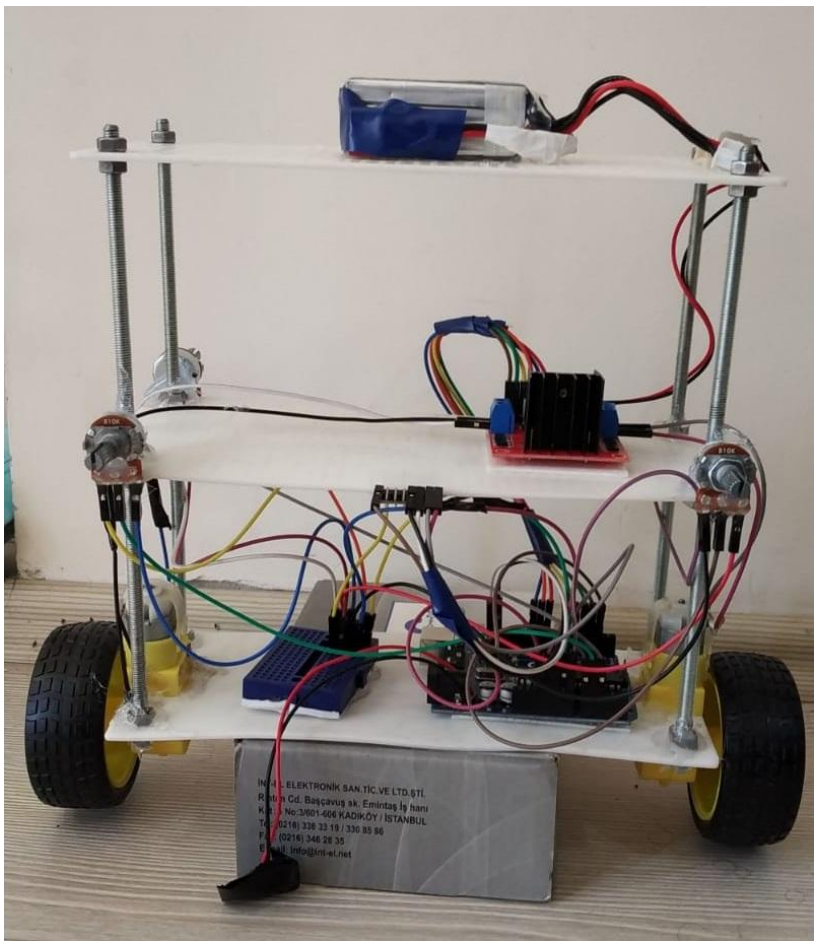
```
 // interrupt bayrağını sıfırla ve INT_STATUS baytını al
    mpuInterrupt = false;
    mpuIntStatus = mpu.getIntStatus();

    // geçerli FIFO sayısını al
    fifoCount = mpu.getFIFOCount();

    // aşırı akım olup olmadığını kontrol et  (kodumuz yetersiz olmadığı sürece bu asla olmaz)
    if ((mpuIntStatus & 0x10) || fifoCount == 1024)
    {
       // sıfırlayarak temiz bir şekilde devam edebiliriz
       mpu.resetFIFO();
       Serial.println(F("FIFO overflow!"));

    // DMP verilerinin hazır kesildiğini kontrol et   (bu sık sık kontrol edilmeli)
    }
    else if (mpuIntStatus & 0x02)
    {
       // doğru veri uzunluğu için bekle ,kısa bir bekleme
       while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();

       // FIFO dan bir paket oku
       mpu.getFIFOBytes(fifoBuffer, packetSize);

       // 1 paket mevcut olması durumunda FIFO burada takip edin
       // (bu bir kesinti olmadan hemen daha fazlasını okumamıza izin verir)
       fifoCount -= packetSize;

       mpu.dmpGetQuaternion(&q, fifoBuffer);
       mpu.dmpGetGravity(&gravity, &q);
```

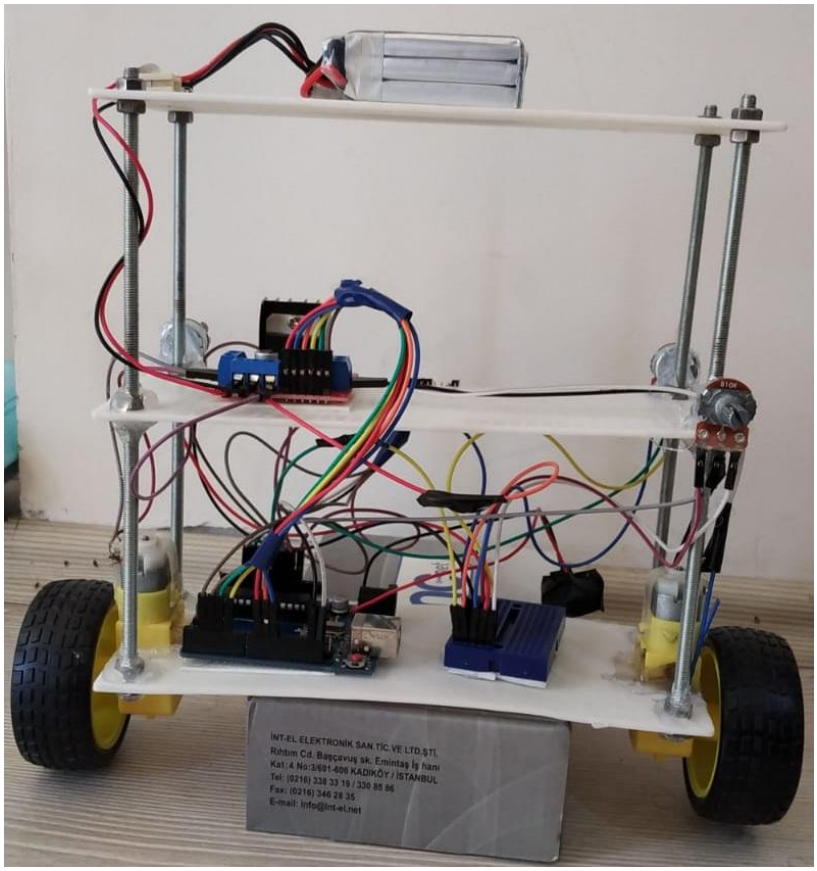
```
        mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);

        #if LOG_INPUT
            Serial.print("ypr\t");
            Serial.print(ypr[0] * 180/M_PI);
            Serial.print("\t");
            Serial.print(ypr[1] * 180/M_PI);
            Serial.print("\t");
            Serial.println(ypr[2] * 180/M_PI);
        #endif
        input = ypr[1] * 180/M_PI + 180;
    }
}


void loopAt1Hz()
{
#if MANUAL_TUNING
    setPIDTuningValues();
#endif
}


void loopAt5Hz()
{
    #if MOVE_BACK_FORTH
        moveBackForth();
    #endif
}
```

```
//ileri geri hareket etme


void moveBackForth()
{
  moveState++;
  if (moveState > 2) moveState = 0;

  if (moveState == 0)
    setpoint = originalSetpoint;
  else if (moveState == 1)
    setpoint = originalSetpoint - movingAngleOffset;
  else
    setpoint = originalSetpoint + movingAngleOffset;
}

//PID Tuning (3 potentiometers)

#if MANUAL_TUNING
void setPIDTuningValues()
{
  readPIDTuningValues();

  if (kp != prevKp || ki != prevKi || kd != prevKd)
  {
#if LOG_PID_CONSTANTS
      Serial.print(kp);Serial.print(", ");Serial.print(ki);Serial.print(", ");Serial.println(kd);
#endif

      pid.SetTunings(kp, ki, kd);
      prevKp = kp; prevKi = ki; prevKd = kd;
```

```
    }
}
//POTANSIYOMETRENIN TANIMLANMASI
void readPIDTuningValues()
{
    int potKp = analogRead(A0);
    int potKi = analogRead(A1);
    int potKd = analogRead(A2);

    kp = map(potKp, 0, 1023, 0, 25000) / 100.0; //0 - 250
    ki = map(potKi, 0, 1023, 0, 100000) / 100.0; //0 - 1000
    kd = map(potKd, 0, 1023, 0, 500) / 100.0; //0 - 5
}
#endif
```

PROJENİN FOTOĞRAFLARI





KAYNAKÇA

1- [Şişman B., "Eğitimde Robot Kullanımı", Eğitim Teknolojileri Okumaları 2016, İşman A., Odabaşı

2- https://www.st.com/resource/en/datasheet/l298.pdf (2019)

3- https://store.arduino.cc/usa/arduino-uno-rev3 (2019)

4- https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf (2019)

5- Classical PID Control by Graham C. Goodwin, Stefan F. Graebe, Mirio E. Salgado

6- https://www.sparkfun.com/datasheets/Components/General/R12-0-.pdf (2019)

7- Warren M. Macek and D. T. M. Davis, Jr. (1963) "Rotation rate sensing with traveling-wave ring lasers," Applied Physics Letters, vol. 2

8- Hervé Lefèvre, "The Fiber-Optic Gyroscope", 1993, ARTECH HOUSE, INC.

9- https://airandspace.si.edu/collection-objects/resonator-hemispherical-resonator-gyro (2019)

10- Cenk Acar, Andrei Shkel. "MEMS Vibratory Gyroscopes: Structural Approaches to Improve Robustness". 2008.

ÖZGEÇMİŞ

Kağan ZORLUOĞLU 1995'de Bursa'da doğdu; ilk ve orta öğrenimini aynı şehirde , Yıldırım İlköğretim Okulu'nda tamamladı; Hürriyet Mesleki ve Teknik Anadolu Lisesi , Makine Teknolojisi Alanında , Bilgisayar Destekli Makine İmalatı bölümünden mezun olduktan sonra 2014 yılında Karabük Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü'ne girdi. Halen bu bölümde eğitimini sürdürmektedir.

İletişim Bilgileri

E-posta: kaganzorluoglu@gmail.com

ÖZGEÇMİŞ

Kadir Sevinç 1995'te İstanbul'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı; 2 yıl Tekirdağ Namık Kemal Anadolu Lisesinde okuduktan sonra İstanbul/Silivri Hasan Sabriye Gümüş Lisesinde geçip oradan mezun olduktan sonra 2013 yılında Karabük Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü'ne girdi. Halen bu bölümde eğitimini sürdürmektedir. Mekatronik mühendisliğinin, mekanik ve elektrik alanları ile ilgilenmektedir.İş disiplinine sahip, verilen görevi zamanında yerine getiren, sosyal yönü kuvvetli, iletişim becerilerinde iyi bir kişiliğe sahiptir. İyi seviyede İngilizce bilmektedir.Autocad , Ansys , Solidworks , Arduino , Visual Studio programlarını iyi seviyede bilmektedir. (17.05.2019).

İletişim Bilgileri

E-posta: kadirsevinc@hotmail.com.tr

KENDİNİ DENGELEYEN ROBOT

Kadir SEVİNÇ – Kağan ZORLUOĞLU - Gökhan GÖKOĞLU
KARABÜK ÜNİVERSİTESİ MEKATRONİK MÜHENDİSLİĞİ

ÖZET

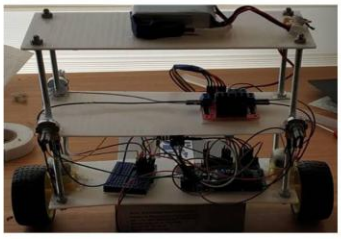
- Bu sistem, birden fazla değişkene sahip bir sistemdir. Bu sistemin kararlılık kontrolü ve robotun dengede kalması zor bir problemdir.
- Günümüzde denge sistemlerine verilen önem git gide artmaktadır . Uçan Araçlarda dengenin bozulması için kanat açılarının, motosikletlerde ön tarafın havaya kalkmaması için motora verilmesi gereken güç , araçlarda yağmurlu havalarda veya virajlarda lastiklerin yapabileceği maksimum açının hesaplanıp direksiyona iletilmesi vb. alanlarda kullanılmaktadır.

GİRİŞ

- PID(Proportional,Integral,Derivative) oransal-integral-türevsel denetleyici PID kontrol döngüsü yöntemi , yaygın endüstriyel kontrol sistemlerinde kullanılan genel bir kontrol döngüsü geribildirim mekanizmasıdır. Bir PID denetleyici ölçülü bir süreç içinde değişen ve istenilen ayar noktası ile arasındaki farkı olarak bir " hata" değerini hesaplar. Kontrolör proses kontrol girişini ayarlayarak hatayı en aza indirerek istenilen ayar değerine ulaşmak için çalışır.

AMAÇ

Bu proje , MPU6050 sensörünün diğer elemanlarla birlikte senkronize şekilde çalışarak, robotu dengede tutabilecek elektronik ve programlamanın bir arada kullanılması ile ; PID nin önemini vurgulayabilmeyi ve uygulayabilmeyi amaçlamıştır.

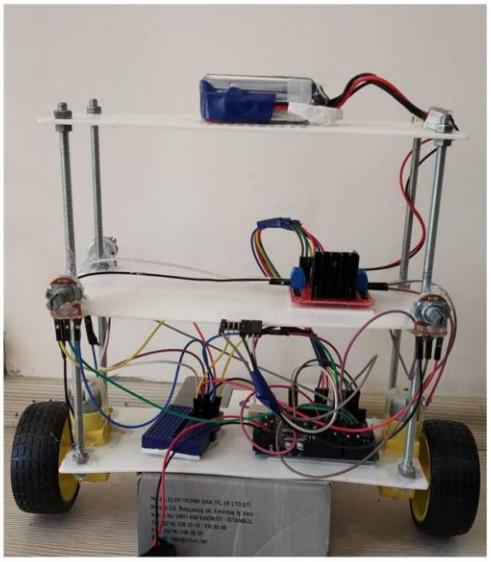


MATERYALLER

- MPU6050 Gyro ve İvme ölçer
- Arduino Uno Rev3
- L298N Motor Sürücü
- 6V 250rpm DC Motor ve Tekerlek
- 10K Potansiyometre
- Jumper Kablolar ve Li-Po Batarya
- M5 Gijon ve Somun
- Dikdörtgen Plaka (PLA)

PROJENİN TANITIMI

- Sistemin çalışması için gerekli PID algoritma kodlanmadan önce sistemin mekanik tasarımı yapılmıştır.
- Proje için gerekli olan malzemeler tasarıma uygun şekilde seçilmiştir.
- Projedeki asıl amaç ; iki tekerlek üzerinde olan robotun , 6 eksenli gyro ve ivmeölçerin yaptığı ölçümlerle robotun dengede olup olmamasına göre tekerlekleri belirli miktarda ve robotu eğildiği yönün tersine doğru döndürecek şekilde hareket ettirerek kendisini dengeye getirmesidir.
- Tasarım yapıldıktan sonra kod kısmına geçilmiştir ve Arduino Uno Rev3 kullanılmıştır.
- Kodlardaki PID değerleri , deneme yanılma yöntemiyle bulunmuştur. Kısaca PID'yi açıklayacak olursak ; «P» , salınım yaparken ki hızı ; «D», salınımı yaptıktan sonra geri gelme hızını ; «I» ise zaman katsayısıdır.



SONUÇ

- Robotun dengede kalması amaçlanmıştır.
- PID'nin anlamı , kullanım amaçları ve değerlerinin nasıl kullanıldığı ,
- Arduino'da PID kodlarını yazmak ve kodların amaçları ,
- DC Motorların, motor sürücülerin ve MPU6050'nin kullanım amaçları ve Arduino ile kontrolü , öğrenilmiştir.

Kaynakça

1. https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf (2019)
2. Norman S. Nise, Control Systems Engineering 6th edition, ISBN: 978-0-471-79475-2 (or 978-0-470-91769-5) John Wiley & Sons, Inc., New York, NY, 2011
3. Classical PID Control by Graham C. Goodwin, Stefan F. Graebe, Mirio E. Salgado

İletişim

kadirsevinc@hotmail.com.tr
kaganzorluoglu@gmail.com