Derinlik ve Yön Kontrol Uygulamaları İçin Deney Platformu Tasarımı

Proje No: 111E294

Yrd.Doç.Dr. Serhat YILMAZ
Yrd.Doç.Dr. Mehmet YAKUT
Müh. Sibel İNCE

Temmuz 2013
KOCAELİ

PSRYUK-02 Güncelleme Tarihi: 22/04/2011

Önsöz: Projenin amacı, sualtı aracı teknolojisindeki gelişmelere denetim yöntemleri açısından katkı sağlayacak bir geliştirme platformu tasarlamaktır. Deney platformu, mekanik, elektronik ve yazılım kısımlarından oluşan bir insansız bir sualtı aracı ve içinde hareket kontrol uygulamalarının yapılacağı bozucu etkiler oluşturulabilen su tankından oluşmaktadır. Sürücü ve algılayıcıları da içeren, bilgisayarla haberleşebilen elektronik anakart, aracın gereksinimlerini yerine getirebilecek şekilde tasarlanmış ve hazırladığımız kullanıcı kontrol arayüzü yazılımı ile bağdaştırılmıştır. Diğer yandan aracın bilgisayar ortamında benzetim ve tasarım uygulamalarını gerçekleştirebilmek için matematiksel modeli çıkarılmıştır. Geliştirilen platform temel derinlik ve yön kontrol uygulamalarını yapabilmektedir.

Sualtı aracının kendi imkanlarımızla tasarımı ve üniversitemiz laboratuarında yer alması, üzerinde çeşitli testlerin yapılabilmesi, yazılım geliştirilebilmesi ve farklı denetleyici yapılarının tasarlanabilmesi, üniversitemizde çalışan proje ekibine ve öğrencilerimize belirli bir yetkinlik kazandırmış ve yeni proje çalışmaları için altyapı hazırlamıştır. Ayrıca güncel literatürü takip etme, kendi uygulamalarımızla karşılaştırma imkanı vererek literatüre katkı potansiyeli olan çalışmaların önünü açmıştır. Çok sayıda lisans ve lisansüstü bitirme tezi ve proje öğrencimizin proje kapsamında toplanarak kolektif çalışmasını, ortaya bir ürün çıkarmada katkı koymalarını ve kaliteli ana tasarım tezleri sunmalarını sağlamıştır. Tez çalışmalarının sonuçları konferanslara ve bir dergiye gönderilmek üzere hazırlanmaktadır. Önümüzdeki yıllarda yapılacak olan Sualtı Araçları Yarışmalarına başvuru yapmak için altyapı hazırlıklarına başlanılmıştır [1,2]. Proje TÜBİTAK tarafından desteklenmiştir.

İÇİNDEKİLER

1. Tablo ve şekil listeleri

Özet:

İnsansız su altı araçları, günümüzde sualtı hareketlerinin izlenmesi, okyanus dibi sıcaklık haritalarının çıkarılması, gemi altı hasarlarının görüntülenmesine yönelik ekspertiz işlemleri, tehlikeli derinliklerde görüntü alma, batıkların incelenmesi, sahil güvenliğini sağlama, askeri bir takım görevleri yerine getirme, akarsuların denizlere döküldüğü alıcı su ortamlarının düzenli kirlilik analizi ve kirlilik haritalarının çıkarılması gibi çok geniş bir alanda kullanılmaktadır.

Projenin amacı, sualtı aracı teknolojisindeki gelişmelere denetim yöntemleri açısından katkı sağlamaktır. Uzaktan kumandalı veya otonom olarak çalışan sualtı araçları belirli bir açıya yönelme, belirli bir derinliğe inme, yanaşma ve seyir gibi temel dinamik hareketleri yapabilmelidir. Bu hareketlerin, sualtı akıntıları gibi bozucu etkiler karşısında da başarılı olması beklenmektedir.

Proje, iki yüksek lisans tezini desteklemektedir[3,4]. Projede testlerin yapılacağı bir sualtı aracı hazırlanmıştır. Aracın içinde çalıştırılacağı bozucu etki üreten bir deney tankı kurulmuştur. Araç derinlik bilgisini basınç sensöründen, yön bilgisini elektronik pusuladan almaktadır. Hareketler 4 motorla sağlanmaktadır. Aracın derinlik ve 2 eksende yön kontrolü için dinamik modeli çıkarılmıştır.

Aracın temel dalış ve hareketleri yapabilmesi için kablo bağlantısıyla uzaktan kumanda ile yönlendirilmesi sağlanmıştır. Hazırlanan kontrol arayüzünde istenen hareketleri sağlayabilmek için gerekli algoritmalar çıkarılmıştır.

Anahtar Kelimeler: Derinlik kontrolü, yön kontrolü, ROV, Sualtı Sistemleri

Abstract

Recently, unmanned underwater vehicles (uuvs) are operated in a wide range areas such as observations of underwater movements, determination of salty water layers, demining, maintenance and expertise operations of ship bilges, imaging in dangerous depths, investigation of sinks, execution of some military tasks, regular inspection and mapping of environmental pollution in receiving water bodies such as lake, freshwater resources.

This project will contribute to developments of underwater vehicle technology in terms of control methods. Remotely or autonomously operated underwater vehicles should handle basic dynamical behaviors such as standing to a certain angle, diving to a certain depth, docking and cruising. These behaivors are expected to be successful under distortional effects such as underwater flows.

The Project supports two M.Sc Thesis' [3,4]. In the project, an experimental underwater vehicle which desired tests are held is designed and realized. An experimental tank with distortion effects, on which the vehicle runs is set. Depth is sensed by pressure sensor where steering is determined by an electronic compass. Movements are actuated by 4 thrusters. Dynamical model of the platform for depth and 2 dimensional steering control is extracted.

The vehicle is remotely operated through a cable. In the Control Programme Interface the algorithms for the desired movements are established.

Keywords: Depth Control, Steering Control,ROV, Underwater Systems

4. Giriş

Bu çalışmada, derinlik ve yön denetimli su altı aracının tasarımı, modellenmesi ve gerçekleştirilmesi aşamaları verilmiştir. Su altı araçlarında denetimi sağlayabilmek için, aracın dinamik davranışları bir yazılım tarafından belirlenmelidir. Temel olarak, su altı araçlarında denetlenmesi gereken başlıca değişkenler, aracın derinliği, 3 eksende yönü ve hızıdır.
Konuyla ilgili olarak Tomotoka ve diğ. [5] mikroişlemci tabanlı bir sualtı robotunda derinlik kontrol sistemi geliştirmiştir. Çalışmada basınç sensöründen alınan derinlik bilgisi, mikrodenetleyiciye girilen istenen derinlik bilgisi ile karşılaştırılarak, aradaki hatayı düzeltecek şekilde motorları sürerek istenen derinliğe ulaşılmıştır.
DeBitetto [6] benzer bir çalışmada geleneksel denetim yöntemleri yerine bulanık denetleyicileri kullanmıştır. Burada Tomoka'nın çalışmasından farklı olarak, istenen derinlik ile gerçek derinlik arasındaki hata bilgisi ve onun türevi kullanılarak derinlik denetimi yapılmıştır. Burada derinlik hatası ve hatadaki artışa bağlı olarak aracın kalkış açısının büyüklüğü belirlenmiştir. Değişken büyüklükleri matematiksel bir model yerine sözel ifadelerle betimlenmiştir.
Kim ve Diğerleri [7,8], bir otonom sualtı aracının matematiksel modelini çıkarmışlardır. Kayan kipli gözlemci ve Kalman Filtresi kullanarak, aracın 6 serbestlik derecesinde hidrodinamik katsayılarını belirlemişlerdir. Buradan aracın blok şema modelini çıkararak, çeşitli derinlik ve yön referans değerleri için, aracın davranışının benzetimini yapmışlardır.

Sualtı araçları, alıcı su ortamlarında düzenli olarak kirlilik analizi sualtı yaşamının gözlenmesi (Eren ve diğ., 2006)[9], sualtı keşifleri, çekimleri ve haritalandırılması , arama kurtarma çalışmaları gibi pek çok uygulamada kullanılmaktadır.

Bu tür çalışmalarda su altı robot mekaniği olarak uygulamaya bağlı çeşitli prototipler kullanılmaktadır. Bu su altı araçlarının bir kısmı su yüzeyindeki sistemlere kablo ile bağlı olup (remotely operated vehicle-ROV) diğer grup kendi başına otonom hareket edebilme yeteneğine sahiptir (autonomous underwater vehicle-AUV). İkinci gruptaki su altı araçları genelde yüzeydeki sistemle akustik modem vasıtasıyla haberleşmektedir. Su altı araçlarının otonom hareketini sağlayabilmek için hem geleneksel (proportional integral derivative-PID) hem de yeni (yapay sinir ağları-YSA) ile kontrol tekniklerinden faydalanılmaktadır (Wang et al., 2003 [10]; Shi et al., 2005[11]) .

Projede bilgisayar ile kablo bağlantılı olarak denetlenen bir araç (ROV) tasarlanmış ve gerçeklenmiştir. Song ve diğerleri,2003 [12]'ni referans alarak dalış, istenen derinlik seviyesinin kontrolü, istenen konum açısının korunması gibi temel hareketlerin bilgisayar ortamında modellenmesi ve benzetim çalışmaları yapılmıştır.

5.Derinlik ve Yön Kontrol Uygulamaları İçin Deney Platformu Tasarımı

Araca ait genel bilgiler şu şekildedir;

Uzunluk: 46cm
Genişlik: 42,5cm
Çapraz köşeden köşeye: 47cm

Ağırlık : 13 kg

Motorların Gücü: 12V-3A yaklaşık 30W
Piller: 12V-7.400m A toplam 20 adet pil (Nikel MetalHidrit,1.2V, 3700 mAh her bir pil, aracın iki ayağında 10'ar adet seri pil bulunmakta bu iki ayak kendi içinde paralel bağlanmıştır)

Aydınlatma ledleri: 15W her biri.

5.1. Mekanik Tasarım

i) Aracın gövde tasarımı projenin ilk 6 aylık döneminde tamamlanmış ve dönem raporunda verilmiştir (Şekil.1)

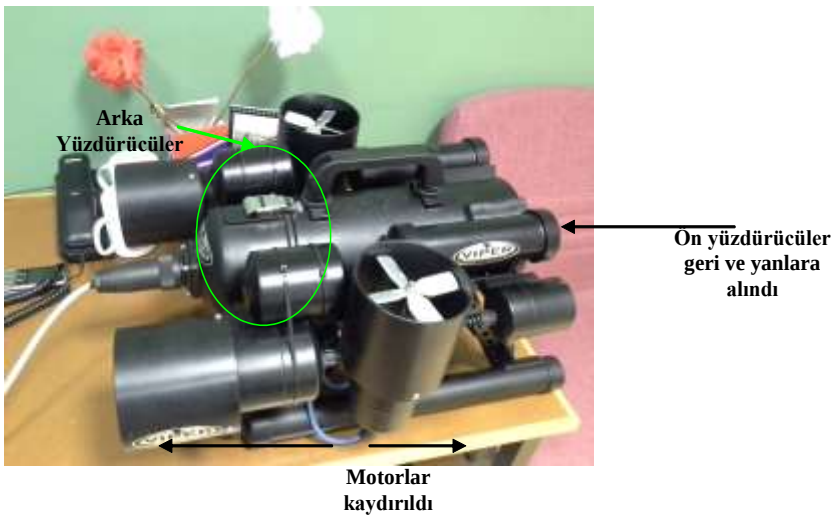
Şekil.1. Sualtı aracının ilk tasarımı [13]

ii) İkinci aşamada kontrol kartının, kamera ve diğer elektronik ekipmanların tamamlanıp araç içine yerleştirilmesinden sonra yeni ağırlık durumu göz önüne alınarak sudaki batmazlık oranının ayarlanabilmesi için aracın on kısmına yüzdürücüler , tüm ayaklara kurşun plakalar eklenmiştir (Şekil.2)

Şekil.2. Gövde tasarımındaki ilk değişiklik (ön yüzdürücülerin eklenmesi, kurşun plakalarla ince denge ayarının yapılması)

Araç su tankına indirilip derinlik kontrolünde kullanılacak olan dikey motorlar çalıştırıldığında araçta denge problemi ve su sızıntısı problemi yaşanmıştır.

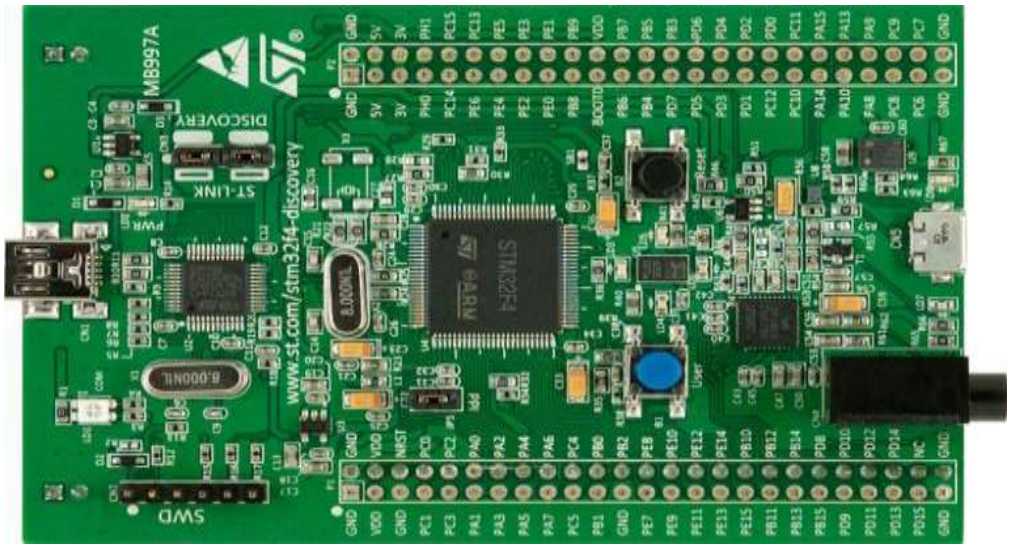
iii) Yatay motorlar geriye çekilip, dikey motorlar ortaya kaydırılarak, arkaya yüzdürücü eklenerek denge problemi giderilmiştir (Şekil.3). Sızdırma problemi O-ring değişikliği yapılarak ve arka kapakta değişiklik yapılarak ortadan kaldırılmıştır [14].



Şekil.3. Gövde tasarımındaki ikinci değişiklik (motor konumlarının kaydırılması ve arkaya yüzdürücü ilave edilmesi)

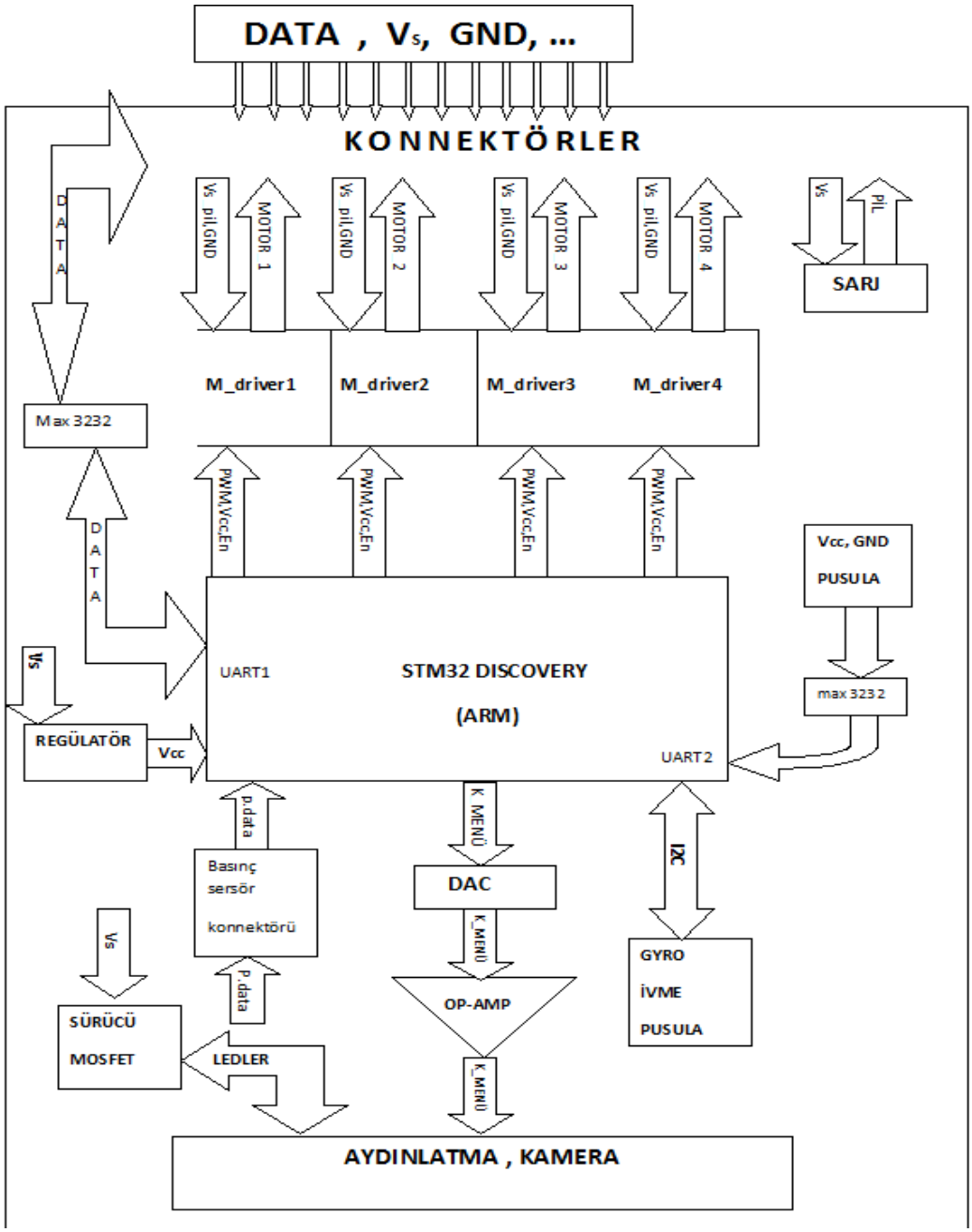
5.2. Elektronik Tasarım

5.2.1. Kontrol Kartı: Kart üzerinde ARM Cortex M4 tabanlı 168 MHz'lik bir mikrodenetleyici bulunan Discovery kit ile denetlenmektedir (Şekil.4).



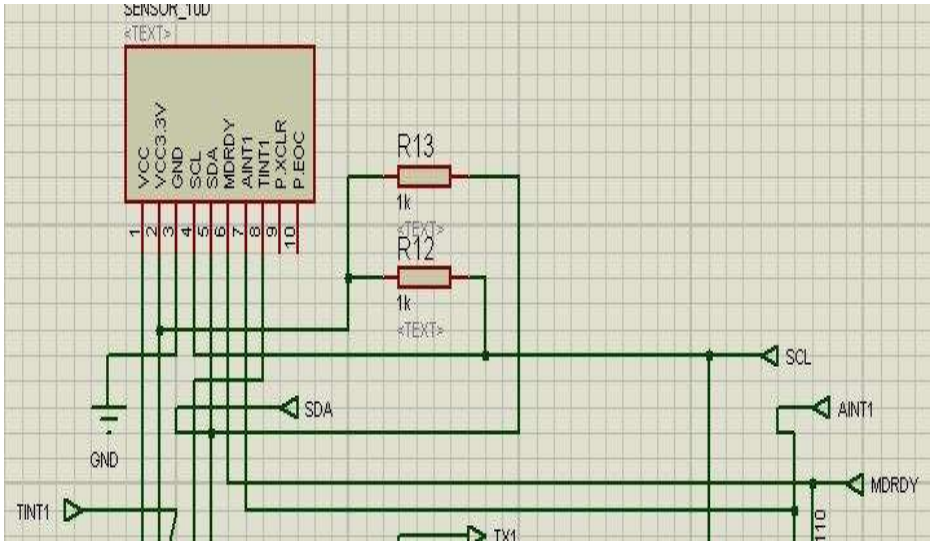
Şekil.4. STM32 Discovery kit

Kontrol kartının amacı bilgisayarla iletişim halinde olan Sualtı Aracımızın kamera görüntülerinden, basınç ve gyro sensörlerindeki verilerden faydalanarak bilgisayardan komut verip aracı yönlendirmektir.
IDE olarak Keil u-Vision kullanılmıştır ve programa bu ide ile C++ dilinde yapılmıştır. Elektronik pusula ve kamera bilgileri işlemciye fazla yük oluşturmamak için doğrudan bilgisayarda işlenmiştir. Bilgisayardan kontrol kartına PWM ve yön komutlarını gönderilmiştir. Derinlik bilgisi için Wika S-10 basınç sensörü kullanıldı, sensörün verileri işlemci tarafından okundu ve bu bilgi bilgisayara aktarıldı [15,16]. Kontrol kartını blok şeması Şekil.5'te verilmiştir.

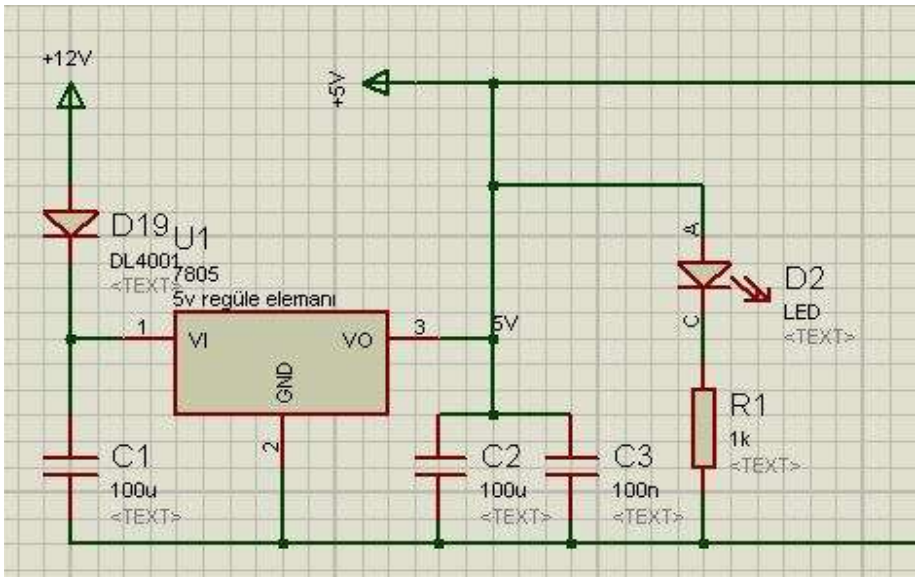


Şekil.5. Kontrol kartı blok şeması

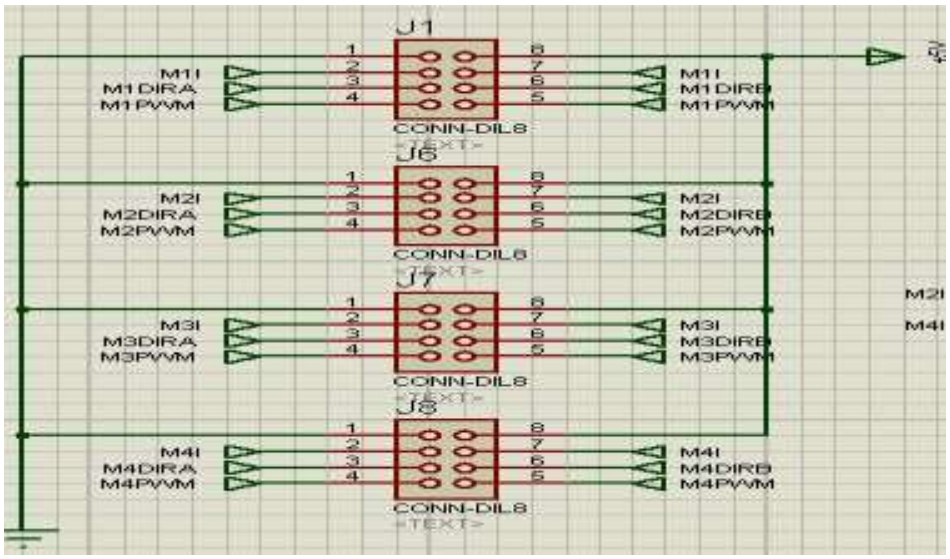
Kontrol kartı devresi aşağıdaki kısımlardan oluşmaktadır.



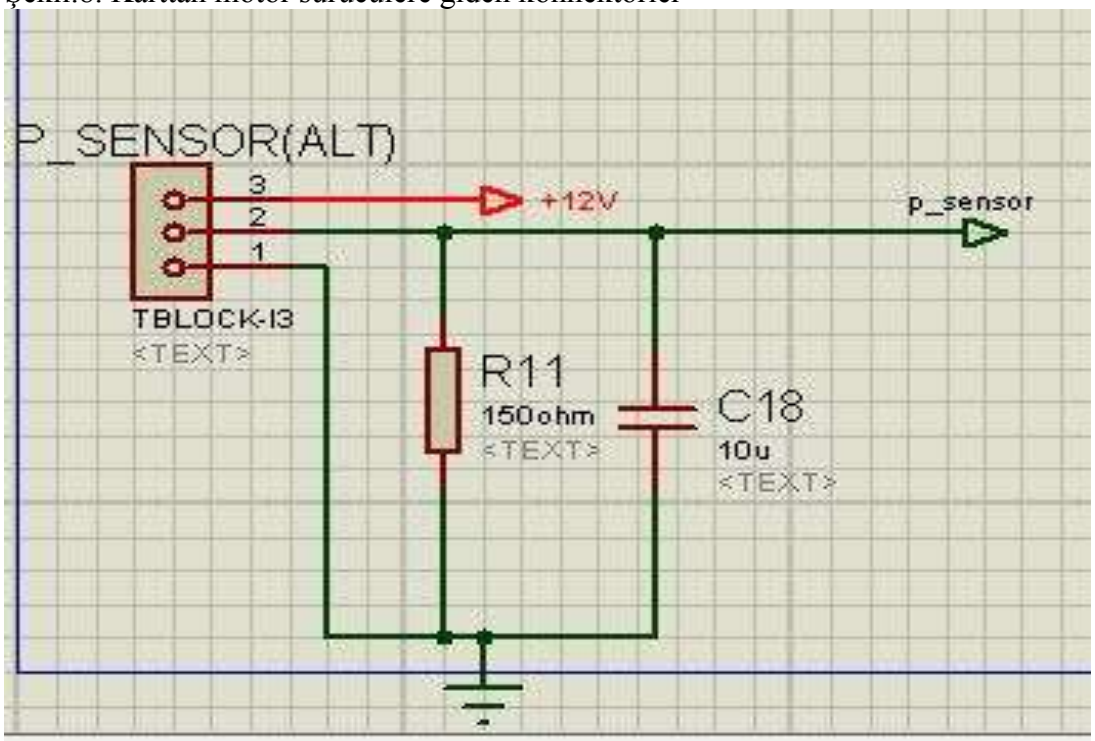
Şekil.6. Elektronik Pusula



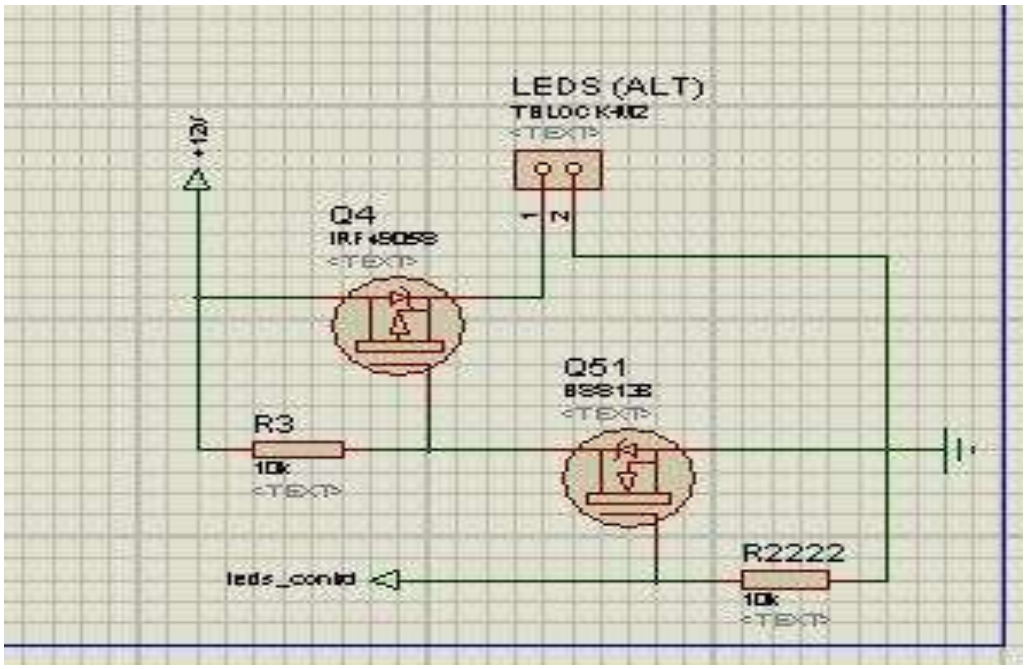
Şekil.7. 12V-5V Dönüştürücü Regüle Devresi



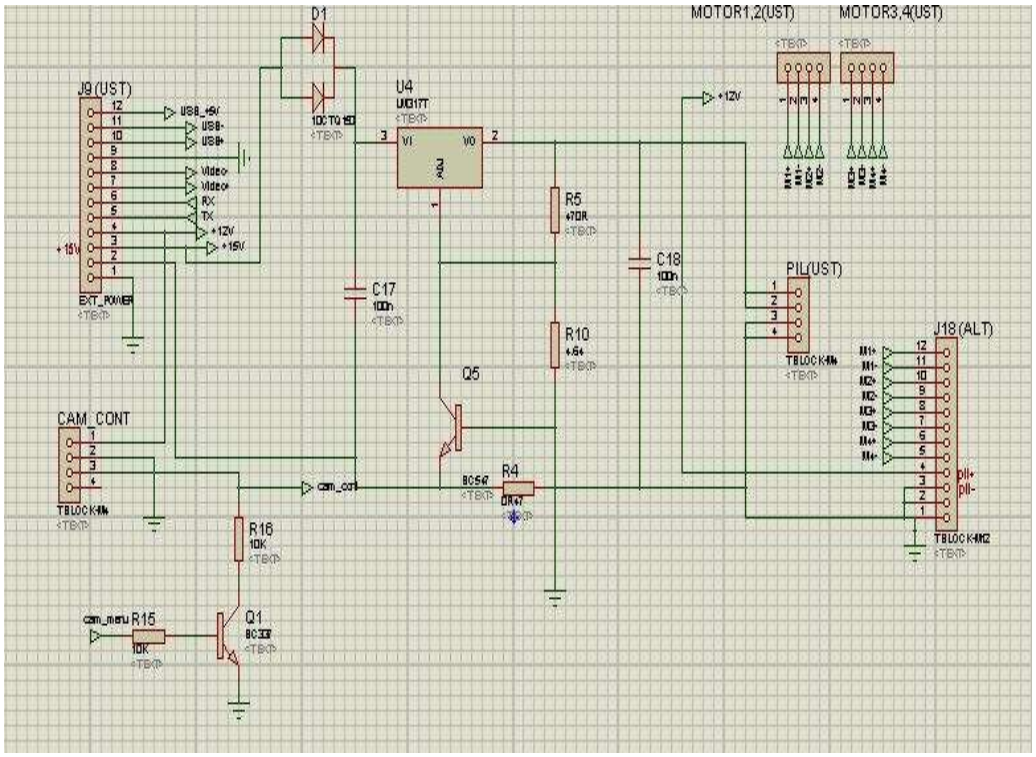
Şekil.8. Karttan motor sürücülere giden konnektörler



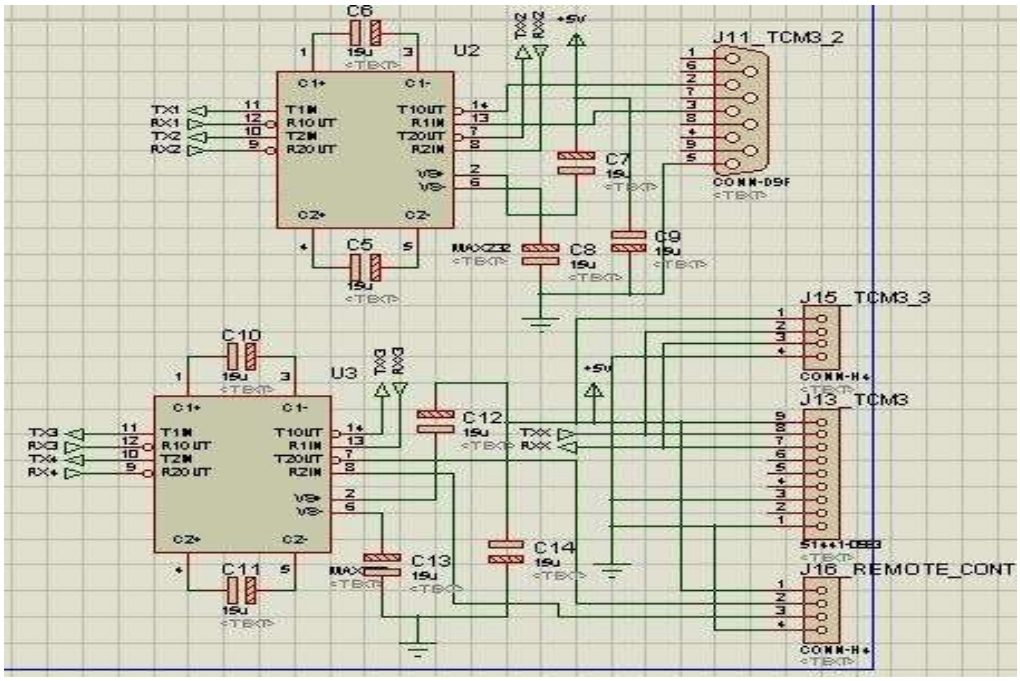
Şekil.9. 4-20mA akım çıkışlı basınç sensörünün çıkışının 0.6-3V arasında gerilime dönüştürülmesi



Şekil.10. Aydınlatma ledlerini süren devre

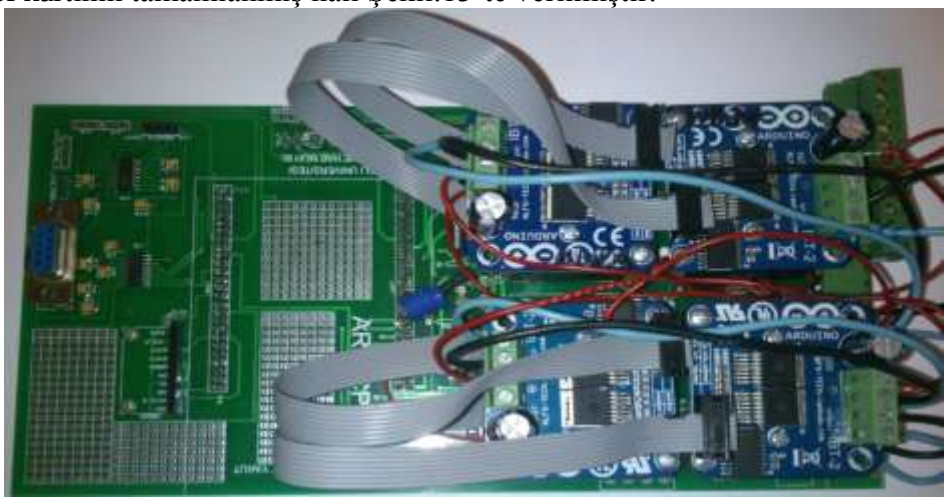


Şekil.11. Çıkış konnektörlerimiz, pil şarz devresi ve kamera menüsünü kontrol etmek için tasarlanan devre katı



Şekil.12. Seri haberleşme için Max3232 ile tasarlanan devre ve TCM3 elektronik pusula girişi, Arm ve TCM3'ün bilgisayarla haberleşmesi için

Kontrol kartının tamamlanmış hali Şekil.13'te verilmiştir.



Şekil.13. Montajı Yapılmış Kontrol Kartı

5.2.2. Kartı gerçeklemede karşılaşılan sorunlar:

- Aracın su yalıtımlı haznesi silindir şeklinde olduğundan ARM KİT planlanan konnektörlere takılamadı
- Kenarlardaki konnektörler planlanan yerlere takılamadı

Şekil.14. Kart hataları ve çözümleri

- Motor sürücünün pinlerinin sürücünün üzerinde yazılan notasyonda olmamasından dolayı kartın tasarımına uymadı. Flat kablonun 5. hattı 8. hat ile , 6. hattı 7. hat ile terslenerek sorun çözüldü.
- Motor sürücüleri hazır kullandığımız için kontrol kartına ve sualtı aracı haznesinde kapladığı alan isteğimiz dışında artmıştır. Bir sonraki kart tasarımında sürücü yolları karta çizilip, devre elemanları doğrudan karta monte edilmelidir.

Şekil.15. Ardunio motor sürücülerde karşılaşılan sorunlar

5.2.3. Kontrol Kartı Programı

Kartımızın PWM değerini ve motor yön bilgilerini bilgisayardan kontrol etmek ve sensör değerlerini okuyabilmek için ADC kodu ve haberleşme kodları yazıldı. Yazılan kodlara göre bilgisayardan 5 karakterlik komut bilgisi gönderilmesi kararlaştırıldı.

İlk karakter 1-5 arası olacak ve
1 için motor 1
2 için motor 2

3 için motor 3
4 için motor 4 seçilmiş olacak

İkinci karakter motor yön bilgisini temsil edecek ve “r” yada “l” olacak.

Sonraki üç karakter motor PWM değerini belirleyecek ve “000” ile “999” arasında olacak.
000 için motorlar duracak
999 için tam güç verilecek

Örneğin 1r500 2r400 3l400 4l400 gibi

Eğer 5SSSS gönderilirse aracımız bize derinlik bilgisini gönderecek.
Örneğin 02,55 gibi.

Bu doğrultuda PWM, ADC ve Haberleşme kodları yazıldı ve birleştirildi:

```
#include "STM32F4xx.h"
#include "stm32f4_discovery.h"

void Yon_Kontrol (void);

unsigned char WAdr,RAdr;
char RxBuf[128],data[5];
int controller=0,a=0,motor=0,count=0;
unsigned long basamak[3]={0,0,0};

uint16_t adc_val;
uint16_t adc_val1;
uint16_t adc_val2;
unsigned char say;

unsigned short PWM[4]={0,0,0,0};
unsigned short SRG[4];
unsigned short CNTR;

int mleft=0,  mright=0, mupleft=0,  mupright=0;
int mlefti=0,   mrighti=0, mrighti2=0, muprighti=0,  muplefti=0;
int K=2, L=2, yukari=0, ileri=0, direction=4;                    // K ve L degerleri baslangicda  yukari ve ileri degerlerine
esit olmasin diye 2 yapildi

/*****************************************************************************************************
      CPU PLL ile 168Mhz de kosturulur
      AHB frekansy 168 Mhz
      APB1 frekansy 42 Mhz
      APB2 frekansy 84 Mhz
*****************************************************************************************************/
void SystemInit()
{
unsigned int i;

    (*((int*)0xE000ED88))|=0x0F00000;
    for (i=0;i<0x00100000;i++);     // OSC oturtma ve kurtarma rutini
    RCC->CFGR |= 0x00009400;    // AHB ve APB hizlarini max degerlere set edelim
    RCC->CR |= 0x00010000;          // HSE Xtal osc calismaya baslasin
    while (!(RCC->CR & 0x00020000));// Xtal osc stabil hale gelsin
    RCC->PLLCFGR = 0x07402A04;      // PLL katsayilarini M=4, N=168, P=2 ve Q=7 yapalim
    RCC->CR |= 0x01000000;          // PLL calismaya baslasin  (Rehber Sayfa 95)
    while(!(RCC->CR & 0x02000000)); // Pll hazir oluncaya kadar bekle
    FLASH->ACR = 0x00000605;        // Flash ROM icin 5 Wait state secelim ve ART yi aktif edelim (Rehber Sayfa
55)
    RCC->CFGR |= 0x00000002;        // Sistem Clk u PLL uzerinden besleyelim
    while ((RCC->CFGR & 0x0000000F) != 0x0000000A); // Besleninceye kadar bekle
    RCC->AHB1ENR |= 0x0000000F;     // GPIO A,B,C,D clock'u aktif edelim

GPIOA->MODER  = 0x00001000;     // A Portunun 6. pini cikis tanimlandi
GPIOB->MODER  = 0x55400005;     // B Portunun 15 14 13 12 11 01 00 pinleri cikis tanimlandi
GPIOC->MODER  = 0x00000517;     // C Portunun 1 2 4 5. pinleri cikis tanimlandi
```

```
GPIOA->OSPEEDR= 0xFFFFFFFF;     // GPIOD nin tum cikislari en yuksek hizda kullanacagiz
GPIOB->OSPEEDR= 0xFFFFFFFF;     // GPIOD nin tum cikislari en yuksek hizda kullanacagiz
GPIOC->OSPEEDR= 0xFFFFFFFF;     // GPIOD nin tum cikislari en yuksek hizda kullanacagiz

 RCC->APB1ENR|=0x00000020;       // Timer7 CLK'u aktif edelim (84 Mhz)
TIM7->CR1=0x0080;               // Otomatik Reload
TIM7->PSC =839;                 // Prescaler degerimiz 839, Count frekansimiz = fCK_PSC / (Yuklenen Deger + 1) 84E6
/ (840) = 100 KHz
TIM7->ARR =1;                   // Counter, Decimal 1 olunca basa donsun. Her 20 mikrosaniye de bir timer int olusacak.

TIM7->DIER=0x0001;              // Update Int enable
NVIC->ISER[1] = 0X00800000;     // NVIC de Timer 7 interrupta izin verelim
TIM7->CR1|=0x0001;              // Counter Enable


RCC->APB2ENR|=0x00000100;  // ADC saat kaynagini aktif ettik.
ADC1->CR1 |=0x00000100; //  ADC scan modunda çalisacak.
ADC1->SQR3 |= 0x0000000A; // Çevrime ilk girecek kanal 10.
ADC1->CR2 |=0x00000003;   // AD converter'i açtik. Sürekli çevirim yapacagimizi belirttik.
}


void SendChar(char Tx)
{
     while(!(USART3->SR&0x80));  // TX Buffer dolu ise bekle
     USART3->DR=Tx;
}
void SendTxt(char *Adr)
{
     while(*Adr)
       {
         SendChar(*Adr);
         Adr++;
       }
}
char DataReady()
{
      return(WAdr-RAdr);
}
char ReadChar()
{
char Dat;

     Dat=RxBuf[RAdr];
     RAdr=(RAdr+1)&0x7F;

     return(Dat);
}
/*****************************************************************************
     USART3 modulunu kullanarak asenkron haberlesme (Hata kontrolu yapilmiyor)
*****************************************************************************/

void USART3_IRQHandler()
{

volatile int Sts;
    Sts=USART3->SR;
    RxBuf[WAdr]=USART3->DR;
    WAdr=(WAdr+1)&0x7F;

if(controller <= 5)                     // Bilgisayardan 5 karakter bekleniyor
{
data[controller]=USART3->DR;
controller++;
}

if(controller == 5)                     // 5 karakter geldi
{
controller = 0;
//SendTxt(data);                  // Testlerde haberlesme calisiyormu diye eklenen kontrol komutu

switch(data[0])                         // Motor secimi veya ADC bilgisi secenegi
{
case '1' : motor = 0; break;            //Motor 1
case '2' : motor = 1; break;            //Motor 2
case '3' : motor = 2; break;            //Motor 3
```

```
case '4' : motor = 3; break;                    //Motor 4
case '5' :                                      //ADC oku

if(ADC1->SR & 0x0002)
{ if(ADC1->SR & 0x0002){
ADC1->SR &= ~0x0002;
adc_val = ADC1->DR;
// adc_val= adc_val*3/4095;//gerilim hesaplamasi

adc_val2=adc_val;
adc_val2= (adc_val2-819.2)*250/(4095-819.2);
say = adc_val2/1000+48;
          SendChar(say) ;
             say = (adc_val2%1000)/100+48;
          SendChar(say) ;
             SendChar(',');
             say = (adc_val2%100)/10+48;
          SendChar(say) ;
say = adc_val2%10+48;
SendChar(say) ;
}break;
}
}

switch(data[1])                                 // Motor yon bilgisi
{
case 'l' : direction = 0; break;
case 'r' : direction = 1; break;
}

controller = 0;

for(count=0 ; count<=2 ;count++)                        //PWM degeri okunması
{
switch(data[count+2])
{
case '0' : basamak[count] = 0; break;
case '1' : basamak[count] = 1; break;
case '2' : basamak[count] = 2; break;
case '3' : basamak[count] = 3; break;
case '4' : basamak[count] = 4; break;
case '5' : basamak[count] = 5; break;
case '6' : basamak[count] = 6; break;
case '7' : basamak[count] = 7; break;
case '8' : basamak[count] = 8; break;
case '9' : basamak[count] = 9; break;
}
controller = 0;
}


PWM[motor] = 999- ( (basamak[0]*100)+(basamak[1]*10)+(basamak[2]));          //PWM degeri basildi

if( motor==0)
ileri=direction;

if(  motor==1)
ileri=direction;

if( motor==2)
yukari=direction;

if( motor==3)
yukari=direction;

Yon_Kontrol();
controller = 0;

        }
}


void UsartInit()
{
    WAdr=0;RAdr=0;

// USART3 MODULUNU AKTIF HALE GETIRELIM
```

```
      RCC->APB1ENR|=0x00040000;  // USART3 Clk Enable
      RCC->APB1RSTR|=0x00040000;  // USART3 Resetlendi
      GPIOC->AFR[1]=0x7700;  // PC10 PC11 pinleri USART3 ile alakalandirildi
      GPIOC->MODER|=0xA00000;  // GPIOC 10 11 icin alternatif fonksiyon tanimi

// USART3 MODULUNU AYARLAYALIM  // 1 Start, 8 Data, 1 Stop, No parity (Default degerler)
      RCC->APB1RSTR&=~0x00040000;  // USART3 Reseti kaldiralim
//    USART3->SR&=~0X03FF;   // Status registeri silelim
      USART3->BRR=0X1112;    // 9600 Baud

      USART3->CR1|=0x0000202C;  // USART3 enable
      NVIC->ISER[1]|=0x80;         // NVIC da USART3 interrupta izin verelim
}

void TIM7_IRQHandler()
{
unsigned short d,i,j;


    TIM7->SR=0;                  // Timer Int Flagini silelim
    d=GPIOC->ODR & 0xFF8D;                                          // PWM basilacak bitler sifirlandi

    CNTR++;
    if(CNTR>=1000)
    {
       CNTR=0;
     for(i=0;i<4;i++)
      {
if(PWM[i]>1001)
PWM[i]=1000;

SRG[i]=PWM[i];
       }
    }



  for(i=0;i<4;i++)
    {
if (i==0)                                    // Sag Motor

{
j= 0x0040;                                          //PWM  PA6
if (CNTR>=SRG[i])
d|=j;
GPIOA->ODR = d;                             // PWM BAS

}

if (i==1)                                    // Sol Motor

{
j=0x0010;                                   //PWM  PC4
if (CNTR>=SRG[i])
d|=j;
GPIOC->ODR = d;                                     // PWM BAS

}
if (i==2)                                   // Sag Üst Motor
{
j=0x0020;                                   // PWM  PC5

if (CNTR>=SRG[i])
d|=j;
GPIOC->ODR = d;                                     // PWM BAS

}
if (i==3)                                   // Sol  Üst Motor
{
j=0x0002;                                   // PC1

if (CNTR>=SRG[i])
d|=j;
GPIOC->ODR   = d;   // PWM BAS

                        }
```

```
            }
}


int main()
{
GPIOA->ODR =0x0;                                        //Başlangıçta portlar sıfırlandı
GPIOB->ODR =0x0;
GPIOC->ODR =0x0;

UsartInit();

ADC1->CR2 |=0x40000000;                 // ADC çevrimini başlattık.
while(1){}

}

void Yon_Kontrol (void)
{

if ( ileri==K )
{ }
else                                                    // Yon bilgisi değiştiyse
{
if ( ileri==0 )                                         //Motorlar geriye dogru çalissin
{
mrighti= 0x0001;                        //Yön PB0
mrighti2=0x0000;                        //Yön PC2
mlefti=0x1000;                          //Yön  PB1 ve PB12 diger kod   0x0002 olacak
}

if ( ileri==1 )                                         //Motorlar ileriye  Dogru çalissin
{
mrighti= 0x0000;                        //Yön PB0
mrighti2=0x0004;                        //Yön PC2
mlefti=0x0002;                          //Yön  PB1 ve PB12 diger kod   0x1000 olacak
}

GPIOB->ODR &=0xEFFC;                    //Diğer pinler etkilenmesin diye sadece yon pinleri sıfırlandı

GPIOC->ODR &=0xFFFB;                    //Diğer pinler etkilenmesin diye sadece yon pinleri sıfırlandı

GPIOB->ODR |=mrighti;
GPIOC->ODR |=mrighti2;
GPIOB->ODR |=mlefti;
K=ileri;

}


if ( yukari==L )
{ }
else                    // Yon bilgisi değiştiyse

{
if ( yukari==0 )                // Motorlar Yukari  Dogru çalissin
{
muprighti=0x8000;                       // Yön PB15 ve PB13 diger kod  0x2000
muplefti=0x4000;                                // YÖN PB14  ve PB11 diger kod  0x0800
}

if ( yukari==1 )                                        // Motorlar Asagi Dogru çalissin
{
muprighti=0x2000;                                       // Yön PB15 ve PB13 diger kod  0x8000
muplefti=0x0800;                                        // YÖN PB14  ve PB11 diger kod 0x4000
}

GPIOB->ODR &=0x17FF;                                    //Diger pinler etkilenmesin diye sadece yon pinleri sifirlandi

GPIOB->ODR |=muprighti;
GPIOB->ODR |=muplefti;
L=yukari;
}
}
```

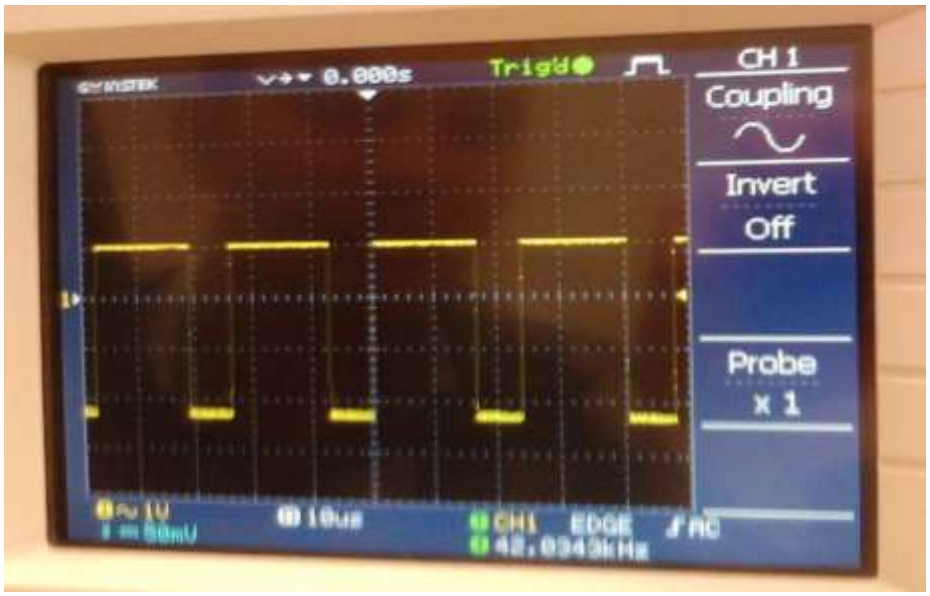
5.2.4. Touch Panel Eklentisi ile Sualtı Aracının Kontrol Arayüzü

Bilgisayardan kontrol için hazırlananan arayüzden bağımsız olarak aracı kontrol edebilmek için bilgisayara alternatif bir el terminali de tasarlanmıştır. Böylece bilgisayar taşımaya gerek kalmadan araca temel komutlar yollanabilecek, araçtan alınan bilgiler bu panelde izlenebilecektir [17]. Çalışmada amaçlanan otonom su altı aracının ARM STM32F4 DISCOVERY Touch Panel eklentisi (Şekil.16) ile arayüzünün oluşturulması ve 4 adet motorun ilgili sensör bilgileri ile PWM(Pulse Width Modulation) ayarlarının yapılması ve hareketinin panel üzerinde uygun bir benzetim ile gösterilmesiyle beraber bilgisayar arayüzünden bağımsız bir platform oluşturulmasıdır.

Şekil .16. Kullanılan dokunmatik panel TFT LCD Module : HY32C [10]

Bağımsız platform sayesinde verilerin gözlenebilmesi ve motor PWM değerlerinin kurulması gibi işlemlerin hazırlanan arayüzde kolayca gerçeklenmesi işlemi hedeflenmiştir. Çizimler oluşturulması aşamasında lcd. h kütüphanesi kullanılmış ve eklentileri 'C' dili ile yazılmıştır. Dokunmatik ekran 3 alana ayrılmıştır. İlk alanda sensör bilgileri, ikinci alanda X-Y-Z kartezyen koordinat, son olarak üçüncü ekranda ise PWM değerleri için sliderlar(kaydırıcılar) ve butonlar gösterilmiştir.

4 adet motorun sliderlar ile PWM kontrolü yapılabilmektedir. Osiloskop ekranından alınan ölçümler sonucu aşağıdaki gibi ekran çıktıları alınmaktadır. Çıktı olarak elde edilen PWM değerleri 42 Khz frekansında ve %70 ve %20 değerleri mertebesindedir. Programın tüm PWM değerleri için çıkış ekran görüntüleri çalışmanın **EK-1** kısmına konulmuştur.



Şekil.17. %70 PWM Değeri

Aşağıda x : 190-310, y: 198-213 koordinatları çerçevesinde çizdirilen sliderın kod parçacığı gösterilmiştir.

```
LCD_DrawLine(190, 198, 310, 198, Black);
LCD_DrawLine(190, 198, 190, 213, Black);
LCD_DrawLine(190, 213, 310, 213, Black);
LCD_DrawLine(310, 198, 310, 213, Black);
```

Gösterilen koordinatların % ' lik dilimlere bölünme aşamasında önceki sliderların bölünmesine benzer bir 10 ' luk sistem mevcuttur. **X-Y** butonunun basılmasıyla beraber derinlik ve yanal konumlar ile ilgili motorlara kurulan PWM değerleri gönderilmektedir. Bu motorlara gönderilen PWM'ler için belirlenen portlar ise **PA1** ve **PA0** portlarıdır.

Yanal yön seçimi için ilgili butondan yapıldığı durumlar çerçevesinde % ' lik dilimlere göre açılar belirlenmiştir. Belirlenen açılar yardımıyla da yatay düzlemde (x-y) aracın nasıl bir hareket sergileyeceği en uygun seviyede tespit edilmiştir.

Aracın slider tarafından seçilen PWM değerlerinin X ve Y koordinatlarına etkisi ve koordinat sisteminde aracın bölgesi Şekil.19'daki gibi olmaktadır.

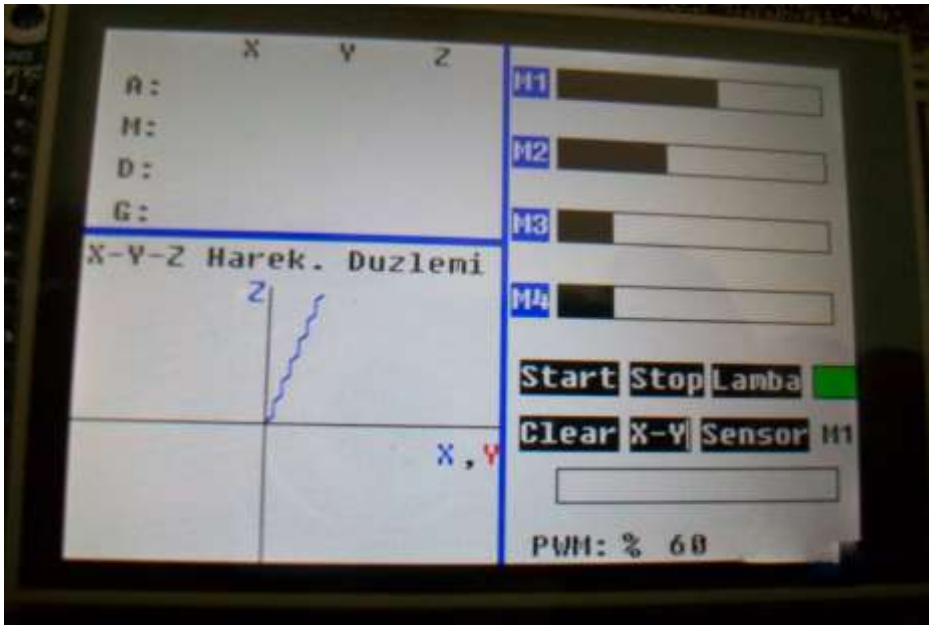
PWM	X	Y	Bölge
% 10	4	1	Birinci
% 20	2	3	Birinci
% 30	0	5	Y

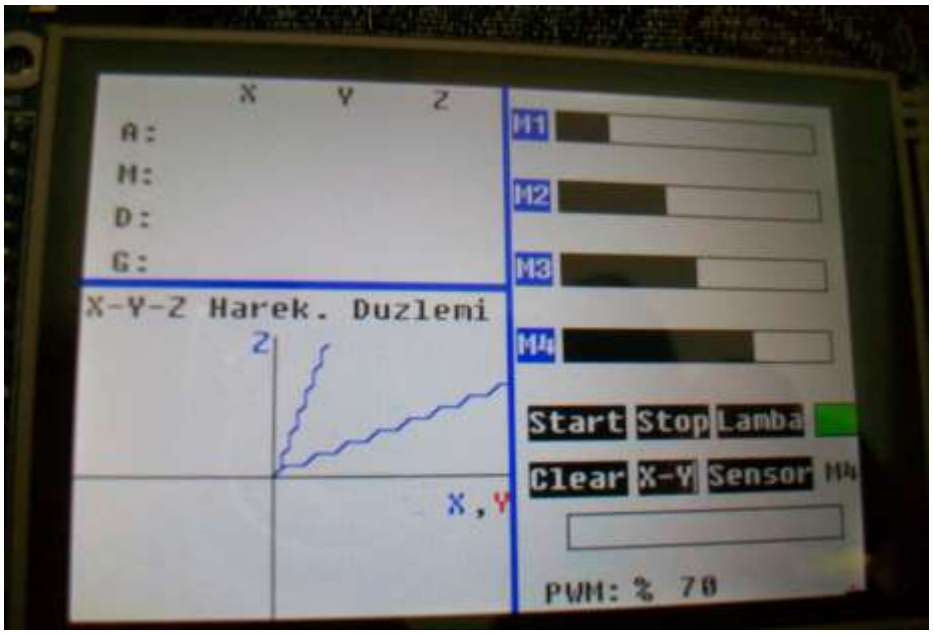
% 40	-2	3	İkinci
% 50	-5	0	-X
% 60	-4	-1	Üçüncü
% 70	-3	-2	Üçüncü
%80	3	-2	Dorduncu
%90	4	-1	Dorduncu
%100	5	0	+X

Şekil.18. Araç için Koordinat-PWM dönüşümleri

.

Derinlik ters kontrolü durumu ise **X-Y** butonu çerçevesine tek sayılarda dokunulduğunda meydana gelmektedir. Bu kontrol bazında sliderdan derinlik seçeneği tıklandığı zaman aracın 1. ve 2. motorları devreye girmekte ve derinlik ters kontrolü motorlarla ilgili sliderlardaki PWM değerlerine göre ve ekstra sliderdan alınan derinlik ters yön PWM değerine göre aracın +Z ve -Z hareketi çizdirilmektedir.



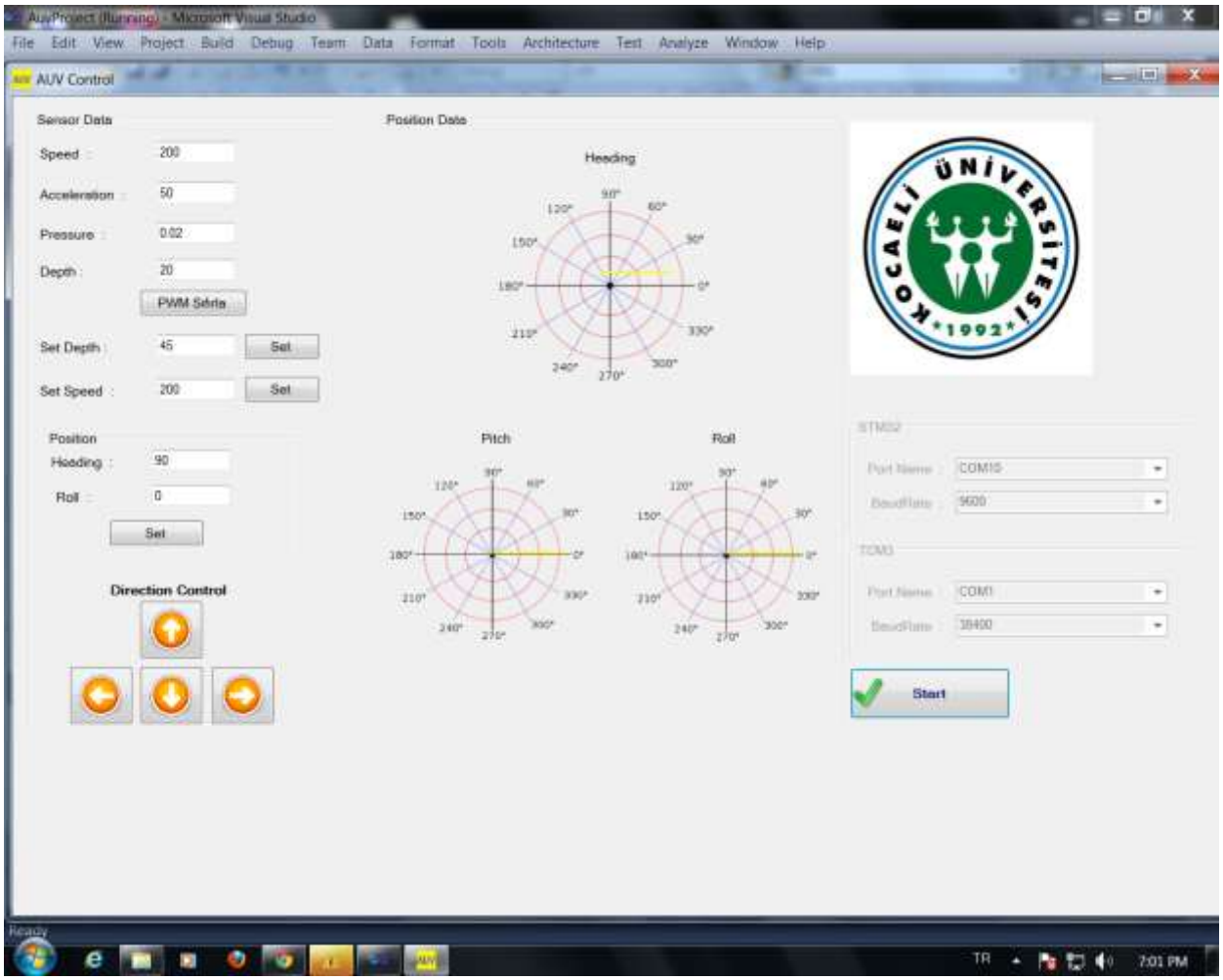


Şekil.19. Tasarlanan El Terminali

5.3. Sualtı Aracı Bilgisayar Arayüzü Yazılımı

Dört motorlu insansız sualtı aracının el ile ya da otomatik olarak kontrolü için bilgisayar arayüzü tasarlanmıştır. Arayüz için farklı proje öğrencileri tarafından farklı diller denenmiştir. Microsoft Visual Studio'da C#, Visual Basic, Matlab ve Delphi'de arayüzler hazırlanmış [18,19], grafik çizdirme, elektronik pusulaya seri porttan erişim, 3 eksen bilgisinin çözümlenmesi … gibi pek çok açıdan karşılaştırılmış ve son olarak C#'ta karar kılınmıştır.

Otomatik kontrol döngüsünde derinlik kontrol geri bildirim elemanı olarak Wika-S10 basınç sensörü ve yön geribildirim elemanı olarak PNI firmasının TCM3 elektronik pusula sensörü kullanılmış ve ilgili sensörlerden gelen verilere göre hangi motorun, hangi yöne ve ne kadar hızla dönmesi hesaplanarak, insansız sualtı aracının önceden belirlenen derinlik, yön ve hız ile dışarıdan sürekli müdahale olmadan otomatik olarak ilerlemesi hedeflenmiştir (Şekil.20).



Şekil.20. Sualtı Aracı Kontrol Arayüzü

Kullanıcı arayüzü tasarımı Microsoft Visual Studio'da C# programa dili kullanarak yapılmıştır [18]. Program kodu aşağıda verilmiştir.

Ugulama C# kod satırları

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO.Ports;

namespace AuvProject
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            //Eğer daha önceden seri portlar açıksa kapatıyoruz.
            if (serialPort1.IsOpen)
                serialPort1.Close();
```

```
            if (serialPort2.IsOpen)
                serialPort1.Close();

            //Kullanacağımız seri portların DataReceived eventlerini aktif hale getirdik.
            serialPort1.DataReceived                          +=                          new
SerialDataReceivedEventHandler(serialPort1_DataReceived);
            serialPort2.DataReceived                          +=                          new
SerialDataReceivedEventHandler(serialPort2_DataReceived);

            //Bağlı olan cihazların com portları otomatik olarak alınsın
            foreach (String portNames in SerialPort.GetPortNames())
                comboBox1.Items.Add(portNames);
            foreach (String portNames in SerialPort.GetPortNames())
                comboBox3.Items.Add(portNames);


            //Kullanacağımız baudrate çeşitlerini yazdık.
            comboBox2.Items.Add(38400);
            comboBox4.Items.Add(9600);
            comboBox4.Items.Add(38400);
        }

        private void Form1_KeyUp(object sender, KeyEventArgs e)
        {

            //Klavyeden yön tuşlarına basıldığında sualtı aracının
            //Heading ve Roll kontrolünü sağladığımız kod bloğu
            if (e.KeyCode == Keys.Up) button6_Click(sender, e);

            if (e.KeyCode == Keys.Down) button8_Click(sender, e);

            if (e.KeyCode == Keys.Right) button9_Click(sender, e);

            if (e.KeyCode == Keys.Left) button7_Click(sender, e);

        }
        private void button1_Click(object sender, EventArgs e)
        {
            //TCM3 için seri port1  atamalarını yapıyoruz.
            serialPort1.PortName = comboBox1.SelectedItem.ToString();
            serialPort1.BaudRate = (int)comboBox2.SelectedItem;
            serialPort1.Parity = 0;    // Parity'i buradan none olarak ayarladım.
            serialPort1.DataBits = 8;  // Databits'i de 8 bit olarak ayarladım.

            //STM32 için seri port2 atamalarını yapıyoruz.
            serialPort2.PortName = comboBox3.SelectedItem.ToString();
            serialPort2.BaudRate = (int)comboBox4.SelectedItem;
            serialPort2.Parity = 0;    // Parity'i buradan none olarak ayarladım.
            serialPort2.DataBits = 8;  // Databits'i de 8 bit olarak ayarladım.

            // Eğer farkli bir şeye ihtiyaç duyulursa serialPort1. dedikten sonra
            // kullanılabilecek özellikler arasında ihtiyacınız olanı seçip
            // ayarlayabiliriz.

            // Serial Haberleşmeyi başlatıyoruz.
            serialPort1.Open();
            serialPort2.Open();

            //TCM3 e veri gönderdiğimiz timer bloğunu aktif ediyoruz.
            timer1.Enabled = true;


            //Dışarıdan portlara ve baudrate kısmına erişimi kapatıyoruz.
            groupBox3.Enabled = false;
            groupBox4.Enabled = false;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            //Stop tuşuyla seri portları kapatıyoruz.
            serialPort1.Close();
            serialPort2.Close();
```

```
    //Dışarıdan port ve baudrate erişimini açıyoruz.
    groupBox3.Enabled = true;
    groupBox4.Enabled = true;

    //TCM3 e veri gönderdiğimiz timer bloğunu pasifleştiriyoruz.
    timer1.Enabled = false;


}

//Çizimde Kullandığımız parametreleri girdik.
//length=çizginin uzunluğu
//angleHeading,anglePitch ve angleRoll değerleri başlangıç koşullarına set ettik.
//Çizginin başlama kordinatını arayüz içerisinde tanımladık.
double length = 80;
double angleHeading = 180;
double anglePitch = 180;
double angleRoll = 180;
double x1 = 109;
double y1 = 109;

private void pictureBox1_Paint(object sender, PaintEventArgs e)
{

    //TCM3 sensöründen gelen Heading açı verisinden faydalanarak
    //grafik üzerindeki ibrenin açısının matematiksel hesaplamasına göre
    //kod tasarımı

    double angleRadians = (Math.PI / 180.0) * angleHeading;
    double x2 = x1 + (Math.Cos(angleRadians + Math.PI) * length);
    double y2 = y1 + (Math.Sin(angleRadians + Math.PI) * length);
    int intX1 = Convert.ToInt32(x1);
    int intY1 = Convert.ToInt32(y1);
    int intX2 = Convert.ToInt32(x2);
    int intY2 = Convert.ToInt32(y2);


    //Grafik çizim kod bloğu

    e.Graphics.DrawLine(
    new Pen(Color.Yellow, 2),
    new Point(intX1, intY1),
    new Point(intX2, intY2));
}

private void pictureBox2_Paint(object sender, PaintEventArgs e)
{
    //Headig açı çiziminde de olduğu gibi pitch açı çizimi
    double angleRadians = (Math.PI / 180.0) * anglePitch;
    double x2 = x1 + (Math.Cos(angleRadians + Math.PI) * length);
    double y2 = y1 + (Math.Sin(angleRadians + Math.PI) * length);
    int intX1 = Convert.ToInt32(x1);
    int intY1 = Convert.ToInt32(y1);
    int intX2 = Convert.ToInt32(x2);
    int intY2 = Convert.ToInt32(y2);

    e.Graphics.DrawLine(
    new Pen(Color.Yellow, 2),
    new Point(intX1, intY1),
    new Point(intX2, intY2));
}

private void pictureBox3_Paint(object sender, PaintEventArgs e)
{
    //Roll açı çizim kodu
    double angleRadians = (Math.PI / 180.0) * angleRoll;
    double x2 = x1 + (Math.Cos(angleRadians + Math.PI) * length);
    double y2 = y1 + (Math.Sin(angleRadians + Math.PI) * length);
    int intX1 = Convert.ToInt32(x1);
    int intY1 = Convert.ToInt32(y1);
    int intX2 = Convert.ToInt32(x2);
    int intY2 = Convert.ToInt32(y2);
```

```
    e.Graphics.DrawLine(
    new Pen(Color.Yellow, 2),
    new Point(intX1, intY1),
    new Point(intX2, intY2));
}

public float GetFloat(byte[] array)
{
    //TCM3 verisinden okudumuz Floating Point sayı dizisini
    //çeviren kod bloğu
    Array.Reverse(array);
    return BitConverter.ToSingle(array, 0);
}

//Seri port 1 in datarecieved eventi
private void serialPort1_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
    //Seri port kontrolü

    if (!serialPort1.IsOpen)
        serialPort1.Open();

    //Gelen veriyi okumaya başlıyoruz
    if (serialPort1.BytesToRead > 0)
    {
        byte[] bfr = new byte[serialPort1.BytesToRead];
        serialPort1.Read(bfr, 0, bfr.Length);
        string result = System.Text.Encoding.UTF8.GetString(bfr);

        //TCM3 sensöründen aldığımız veri paketinden
        //Heading,Pitch ve Roll değerlerini çekiyoruz ve bu değerleri çeviriyoruz.
        byte[] headingbyt = { bfr[5], bfr[6], bfr[7], bfr[8] };
        float heading = GetFloat(headingbyt);
        byte[] pitchbyt = { bfr[10], bfr[11], bfr[12], bfr[13] };
        float pitch = GetFloat(pitchbyt);
        byte[] rollbyt = { bfr[15], bfr[16], bfr[17], bfr[18] };
        float roll = GetFloat(rollbyt);

        angleHeading = heading;
        anglePitch = pitch;
        angleRoll = roll;


    }


}

//Derinlik ölçüm sonucu alınıp basılıyor.
private void serialPort2_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
    if (!serialPort2.IsOpen)
        serialPort2.Open();
    if (serialPort2.BytesToRead > 0)
    {
        byte[] bfr2 = new byte[serialPort2.BytesToRead];
        serialPort2.Read(bfr2, 0, bfr2.Length);
        string result = System.Text.Encoding.UTF8.GetString(bfr2);

        textBox5.Text = bfr2[0].ToString();

    }


}

//Timerdan her kesme gelişinde ne yapılacağını yazdık.
//Seri port1 den TCM3 sensörüne veri gönderme bilgisi gönderdik
//00-05-04-BF-71 (HEX)
//Seri port2 den STM32 ye basınç verisini göndermesini istedik
//35-77-77-77-77 (ASCII)
//Çizilen grafiği temizledik ve yeni çizime hazırladık
```

```
private void timer1_Tick(object sender, EventArgs e)
{
    byte[] numbers = new byte[5] { 0x00, 0x05, 0x04, 0xBF, 0x71 };
    serialPort1.Write(numbers, 0, 5);
    byte[] numbers2 = new byte[5] { 0x35, 0x77, 0x77, 0x77, 0x77 };
    serialPort2.Write(numbers2, 0, 5);
    this.Refresh();
}

//Motorlara PWM değerlerini ASCII olarak seri porttan göndermek için
//Atamaları yaptık
byte[] stop1 = new byte[5] { 0x31, 0x72, 0x30, 0x30, 0x30 };
byte[] stop2 = new byte[5] { 0x32, 0x72, 0x30, 0x30, 0x30 };
byte[] stop3 = new byte[5] { 0x33, 0x72, 0x30, 0x30, 0x30 };
byte[] stop4 = new byte[5] { 0x34, 0x72, 0x30, 0x30, 0x30 };
byte[] ileri1 = new byte[5] { 0x31, 0x72, 0x32, 0x30, 0x30 };
byte[] ileri2 = new byte[5] { 0x32, 0x72, 0x32, 0x30, 0x30 };
byte[] ileri3 = new byte[5] { 0x33, 0x72, 0x32, 0x30, 0x30 };
byte[] ileri4 = new byte[5] { 0x34, 0x72, 0x32, 0x30, 0x30 };
byte[] geri1 = new byte[5] { 0x31, 0x6C, 0x32, 0x30, 0x30 };
byte[] geri2 = new byte[5] { 0x32, 0x6C, 0x32, 0x30, 0x30 };
byte[] geri3 = new byte[5] { 0x33, 0x6C, 0x32, 0x30, 0x30 };
byte[] geri4 = new byte[5] { 0x34, 0x6C, 0x32, 0x30, 0x30 };

//Heading ve Roll açılarını istebilen değere göre ayarlama
private void button4_Click(object sender, EventArgs e)
{
    int counter = 0;


    while (counter != 2)
    {
        double farkH = Math.Abs(Convert.ToDouble(textBox4.Text) - angleHeading);
        double farkR = Math.Abs(Convert.ToDouble(textBox6.Text) - angleRoll);
        double isaretH = Math.Sign(Convert.ToDouble(textBox4.Text) - angleHeading);
        double isaretR = Math.Sign(Convert.ToDouble(textBox6.Text) - angleRoll);

        if (farkH != 0)
        {
            if (isaretH == 1)
            {
                serialPort2.Write(ileri2, 0, 5);
                serialPort2.Write(geri1, 0, 5);
            }
            if (isaretH == -1)
            {
                serialPort2.Write(ileri1, 0, 5);
                serialPort2.Write(geri2, 0, 5);
            }
        }
        if (farkH == 0)
        {
            counter++;
        }

        if (farkR != 0)
        {
            if (isaretH == 1)
            {
                serialPort2.Write(ileri4, 0, 5);
                serialPort2.Write(geri3, 0, 5);
            }
            if (isaretH == -1)
            {
                serialPort2.Write(ileri3, 0, 5);
                serialPort2.Write(geri4, 0, 5);
            }
        }
        if (farkR == 0)
        {
            counter++;
        }
    }
```

```
        }

        //Aracın derinliğini verilen değere göre ayarlama
        double depth;
        private void button5_Click(object sender, EventArgs e)
        {
            int counter1 = 0;
            double farkD = 1;
            double isaretD;

            while (farkD != 0)
            {
                farkD = Math.Abs(Convert.ToDouble(textBox8.Text) - depth);
                isaretD = Math.Sign(Convert.ToDouble(textBox8.Text) - depth);

                if (isaretD == 1)
                {
                    serialPort2.Write(ileri3, 0, 5);
                    serialPort2.Write(ileri4, 0, 5);
                }
                if (isaretD == -1)
                {
                    serialPort2.Write(geri3, 0, 5);
                    serialPort2.Write(geri4, 0, 5);
                }

            }
        }

        //Klavyeden ileri yön kontrolü
        private void button6_Click(object sender, EventArgs e)
        {
            serialPort2.Write(ileri1, 0, 5);
            serialPort2.Write(ileri2, 0, 5);
        }

        //Klavyeden geri yön kontrolü
        private void button8_Click(object sender, EventArgs e)
        {
            serialPort2.Write(geri1, 0, 5);
            serialPort2.Write(geri2, 0, 5);
        }

        //Klavyeden sağ yön kontrolü
        private void button9_Click(object sender, EventArgs e)
        {
            serialPort2.Write(ileri4, 0, 5);
            serialPort2.Write(geri3, 0, 5);
        }

        //Klavyeden sol yön kontrolü
        private void button7_Click(object sender, EventArgs e)
        {
            serialPort2.Write(ileri3, 0, 5);
            serialPort2.Write(geri4, 0, 5);
        }

        //PWM değerleri sıfırlama
        private void button10_Click(object sender, EventArgs e)
        {
            serialPort2.Write(stop1, 0, 5);
            serialPort2.Write(stop2, 0, 5);
            serialPort2.Write(stop3, 0, 5);
            serialPort2.Write(stop4, 0, 5);
        }
    }
}
```

SMT32'den Veri Alma

Araç üzerindeki basınç sensörü doğrudan STM32'ye bağlı olduğu için, derinlik bilgisini STM32'den almamız gerekmektedir. Başınç sensörü su basıncını bar

cinsinden ölçüp STM32'ye göndermekte ve STM32 içinde yapılan işlemler sonucu metreye çevirilmektedir. Bu derinlik bilgisini arayüzümüzde görmek için STM32'nin bize bu veriyi metre cinsine çevirip göndermesi gerekmektedir.

ARAYÜZ İLE SENSÖR HABERLEŞMESİ

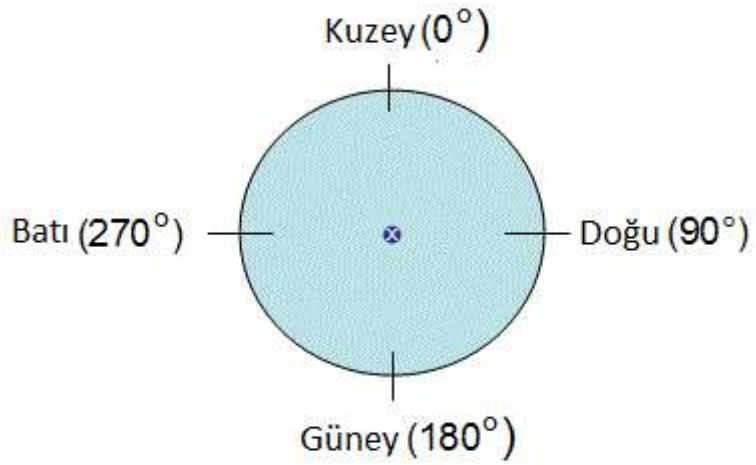
TCM3 sensörü, TCM sensörü ile arayüz haberleşmesi ve o verilere göre STM-32'ye gönderilecek olan pwmlerin belirlenmesi kısaca verilmiştir [20].

TCM3 SENSÖRÜ

İnsansız sualtı aracımızın su altındaki konumunu belirlemek veya istediğimiz konuma göndermek için üzerine bir elektronik pusula yerleştirdik. Elektronik pusulalarda dikkat edilen en önemli unsur ise gördüğümüz değerlerin doğruluğudur. Bu yüzden ölçümlerde çok hassas değişimleri bile görmemizi sağlayan PNI firmasının üretmiş olduğu TCM3 sensörünü tercih ettik.

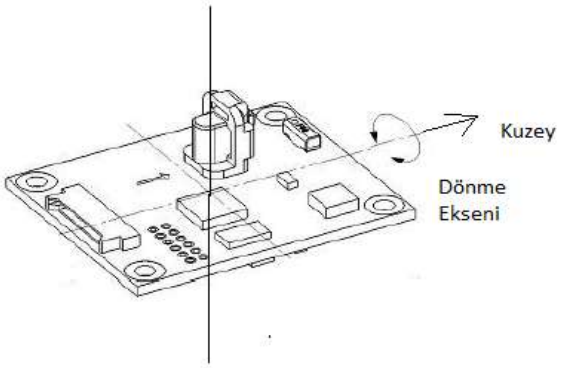
TCM 3, PNI firmasının ürettiği yüksek performanslı, düşük güç tüketimli, eğim-dengeli elektronik pusula modülüdür.TCM 3 elektronik pusula modülü üç eksenli ölçüm yapmaktadır (heading, pitch, roll). Bu ölçüm değerleri, C#'ta hazırlanan kontrol arayüzünde kullanılacak ve insansız sulaltı aracının sualtındaki durumu hakkında bilgi verecektir. Ayrıca insansız sulaltı aracının sualtında dengeli durabilmesi ve istenen doğrultuda ilerleyebilmesi için elde edilen ölçüm değerleri kontrol arayüzünde işlenip insansız sulaltı aracının motorlarına gerekli olan PWM değerleri gönderilecektir.

Rota (Heading) : İnsansız sulaltı aracının yönelme açısı değeridir, yani aracın gittiği yönü belirtir. Bu değer 0°-360° arasında değişmektedir. Örneğin 180°değeri aracın güney yönünde hareket ettiğini gösterir.



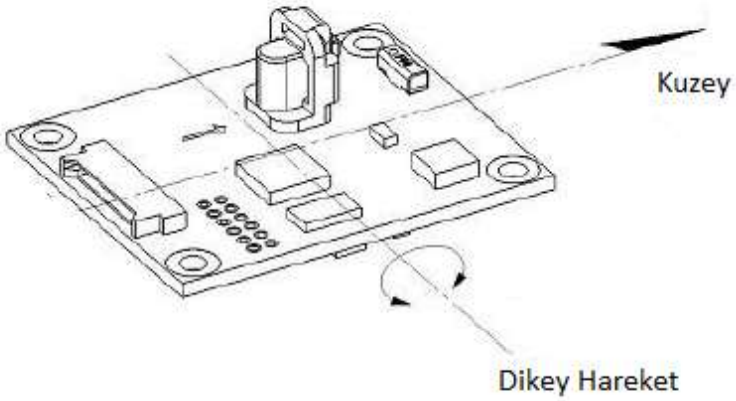
Sensörün yön ve derecesi

Dönme (Roll) : UUV'nin yatay eksendeki dönüş açısı değeridir. TCM 3 elektronik pusula modülü 0°-180° derece aralığında dönme açısı ölçer.



Roll haraketi

Dikey Hareket (Pitch) : UUV'nin dikey eksendeki yönelme açısı değeridir. TCM 3 elektronik pusula modülü 0°-90° aralığında dikey eksen hareketi ölçer.



Pitch hareketi

Denetim Algoritmaları

Gerek derinlik gerekse iki eksende yön kontrolü için gerekli program algoritmaları hazırlanmıştır. PID Denetim ve YSA denetim programlar taslakları aşağıdaki gibidir [23]. Zaman yetersizliğinden dolayı denetim programları ana programa eklenmemiştir.

PID Denetim Algoritması

e_1 : kT = E(z) – Örneklemeye ilişkin hata işareti

e_2 : k(T-1) = z^{-1}. E(z) - Bir önceki örneklemeye ilişkin hata işareti

e_3 : k (T-2) = z^{-2}. E(z) - İki önceki örneklemeye ilişkin hata işareti

P_1 : kT = P(z) - Örneklemeye ilişkin denetim işareti

P_0 : k (T-1) = z^{-1}. P(z) - Bir önceki örneklemeye ilişkin denetim işareti

olmak üzere,

$$(PID)_1 = \frac{2.Kd + 2Kp.T + Ki.T^2}{2.T} = Kp + \frac{Kd}{T} + \frac{Ki.T}{2}$$

$$(PID)_2 = \frac{Ki.T^2 - 2.Kp.T - 4.Kd}{2.T} = \frac{Ki.T}{2} - \frac{2.Kd}{T} - Kp$$

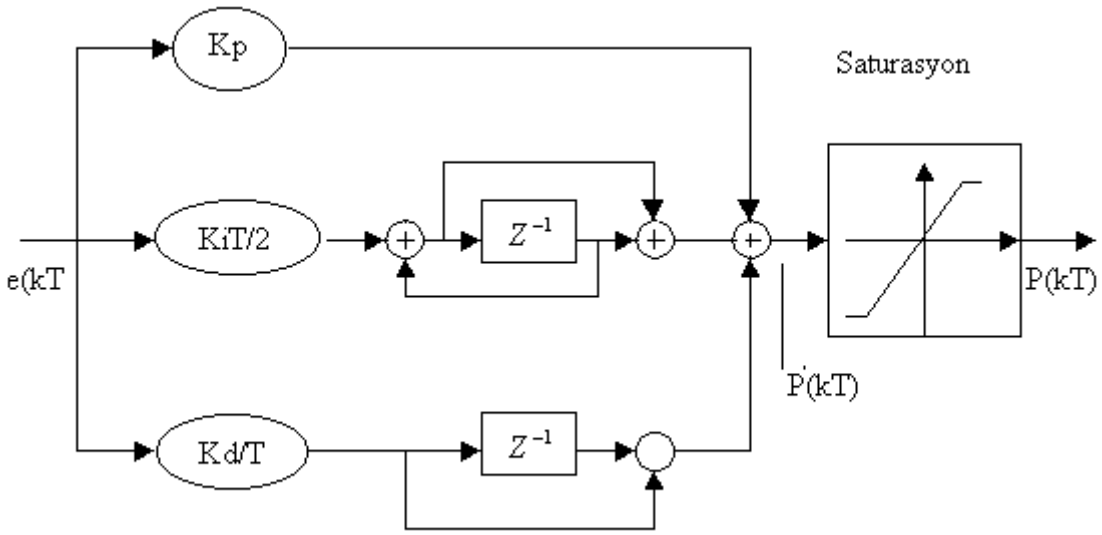
$$(PID)_3 = \frac{Kd}{T}$$

tanımlamaları ile her T örnekleme anı için PID denetleyici algoritmasının oluşturduğu denetim işareti,

$$P_1 = (PID)_1.e_1 + (PID)_2.e_2 + (PID)_3.e_3 + P_0$$

$$P_1 = P_0 + (e_1 - e_2).Kp + (e_1 - 2.e_2 + e_3).\frac{Kd}{T} + (e_1 + e_2).\frac{Ki.T}{2}$$

Yaklaşık trapezoidal integrasyon ve türev kullanılarak aşağıdaki PID algoritması elde edilebilir.

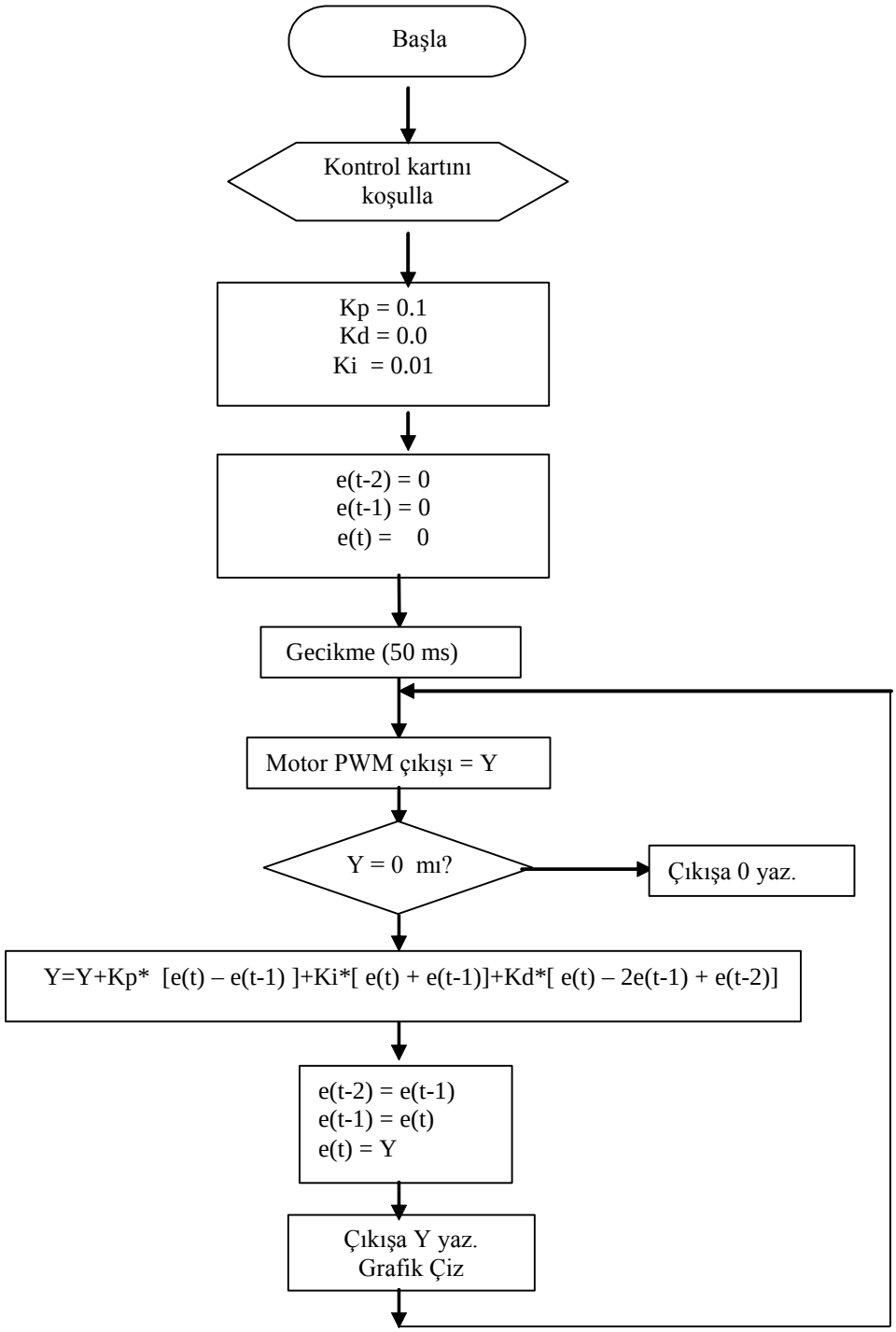


Gerçeklenecek PID Denetleyicinin Yapısı

Denklem programda kullanılmak üzere aşağıdaki gibi yeniden düzenlenir.

Y=Y+Kp* [e(t) – e(t-1)]+Ki*[e(t) + e(t-1)]+Kd*[e(t) – 2e(t-1) + e(t-2)]

Burada hata işareti derinlik kontrolü için istenen derinlik değeri ile basınç sensöründen okunan gerçek derinlik bilgisi arasındaki ilgili örneklemedeki farktır. Denetim işareti ise dikey motorlara uygulanacak PWM işareti tarafından belirlenen ortalama gerilimdir. Yön kontrolünde ise denetlenecek yön hangisi ise istenen açı değeri ile elektronik pusulada ona ait eksenden gelen gerçek açı değeri arasındaki fark hatayı temsil eder. Denetim işareti sözkonusu eksende dönüşü sağlayan motorlara uygulanan gerilimin karşılığı olan PWM işaretidir.



PID Denetim Programı Taslağı

```
program PID_pro;

uses
  Forms,
  PID_unit1 in 'PID_unit1.pas' {Form1},
  Unit2 in '..\Unit2.pas' {Form2};

{$R *.RES}
```

```
begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm2, Form2);
  Application.Run;
end.



unit PID_unit1;

interface

uses
  Windows, driver,math, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    OkuveYaz: TButton;
    Edit1: TEdit;
    Durdur: TButton;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    ProgramCikisi: TButton;
    Edit6: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Edit7: TEdit;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Edit8: TEdit;
    Label11: TLabel;
    Edit9: TEdit;
    Label12: TLabel;
    Edit10: TEdit;
    Label15: TLabel;
    Edit13: TEdit;
    Label17: TLabel;
    Edit15: TEdit;
    Label18: TLabel;
    Edit16: TEdit;
    Label19: TLabel;
    Edit17: TEdit;
    Timer1: TTimer;
    Label29: TLabel;
    Label30: TLabel;
    Edit27: TEdit;
    GrafikCiz: TButton;
    Label13: TLabel;
    Edit11: TEdit;
    Label14: TLabel;
    procedure OkuveYazClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure DurdurClick(Sender: TObject);
    procedure ProgramCikisiClick(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure GrafikCizClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }

  end;

var
  Form1: TForm1;
```

```
  tut : integer;
  girisverisi,normalizegirisverisi: integer;
  hata : longint;
  AGirisKosullamasi : PT_AIConfig;
  Analog_veriGirisi : PT_AIBinaryIn;
  cikisverisi : PT_AOBinaryOut;
  Gerilim,Basinc,E0,E1,E2: Real;
  Y:Extended;
  S,T: String;
  A:Boolean;
  degernum:integer;
  siv,I:integer;
   resim : TBitmap;
   point : PByteArray;
   zaman:integer;
   sayac:integer;
   Kp,Ki,Kd:Real;
implementation

uses Unit2;

{$R *.DFM}


procedure TForm1.FormCreate(Sender: TObject);

{Cihazın Açılması}
begin
I:=0;      {ilk değerleri sıfırla}
E0:=0;  E1:=0; E2:=0; Y:=0;
girisverisi:=0;
Gerilim:=0;Basinc:=0;
zaman:=0;  sayac:=0;
degernum:=23+9;  siv:=0;
Kp:=0; Ki:=0; Kd:=0;
Edit15.Text:=FloatToStr(0.1); {PID Katsayıları giriliyor}
Edit16.Text:=FloatToStr(0.01);
Edit17.Text:=FloatToStr(0);

hata := DRV_DeviceOpen(0,tut);
 case hata of
       Success : Edit1.Text := 'Handle başarılı! Cihaz Düzgün Çalışıyor'; {Sonucun mutlaka başarılı olması gerekiyor}
       end;
end;
 {Cihazın Açılması işlemleri burada bitiyor}


procedure TForm1.OkuveYazClick(Sender: TObject);
{Girişten veri okumak ve Çıkışa veri yazmak için gerekli   koşullamalar yapılıyor}

begin
   Kp:=StrtoFloat(Edit15.Text); {PID Katsayıları giriliyor}
   Ki:=StrtoFloat(Edit16.Text);
   Kd:=StrtoFloat(Edit17.Text);
A:=False;  {Durdur'a basana kadar true olmuyor ve okuma ve yazma işlemlerini
yapan repeat until döngüsü devam ediyor}

 AGirisKosullamasi.DasChan := 10;  {Giriş olarak 10. kanal kullanılıyor}
 AGirisKosullamasi.DasGain := 0;   {Giriş kazancı 1}
 hata := DRV_AIConfig(tut,AGirisKosullamasi);{Giriş koşullaması sonucunda geriye
                                    hata  döndürmemesi gerekir}
 case hata of
       Success : Edit2.Text := 'Giriş Koşullanmaya Hazır'; {Sonucun mutlaka başarılı (0) olması gerekiyor}
       end;
 analog_veriGirisi.chan := 10; {10. kanaldaki veri okunacak}
 analog_veriGirisi.TrigMode := 0;  {Yazılım tetiklemesi yok}
 {Koşullamalar bitti}


  Repeat {Dur'a başana kadar veri okuma ve yazma işlemleri devam edecek}
    sleep(50);{50 ms'de bir hata, bir önceki hata güncelleniyor}
    E2:=E1; edit10.text:=FloatTostr(E2);
    E1:=E0; edit9.text:=FloatTostr(E1);
    E0:=Y;  Edit8.text:=FloatTostr(E0);
   Repeat{Bu döngü okuma işlemi bitene kadar programı bekletiyor}
   Application.ProcessMessages;{Döngü içindeyken diğer prosedürleri çalıştıran butonlara basabilmemizi sağlıyor}
```

```
  analog_veriGirisi.reading := @girisverisi; {Girişteki veri okunuyor,bu verinin başlangıc adresi @verigirisi isimli
pointer'da tutuluacak}
      hata := DRV_AIBinaryIn(tut,analog_veriGirisi);
  until hata=Success;
   case hata of
        Success : Edit3.Text := 'Giriş Veri Okuma Başarılı'; {Sonucun mutlaka başarılı olması gerekiyor}
        end;

      normalizegirisverisi:=girisverisi-1995;
      if normalizegirisverisi<=0 then normalizegirisverisi:=0;
      if normalizegirisverisi>2048 then normalizegirisverisi:=2048;
       Gerilim:=((normalizegirisverisi)/409.6);
        Str(Gerilim:2:2,S);
        Edit4.text:=S;
        Edit27.text:=IntToStr(girisverisi);
        edit6.text:=InttoStr(normalizegirisverisi);
        basinc:=20+(normalizegirisverisi/204.8);

        Str(Basinc:2:2,T);
         edit7.text:=T;

 {PID DENETİM SÜRECİ BAŞLIYOR}

 {Kontrol Çıkışı Hesaplanıyor}
  Y:=normalizegirisverisi+Kp*(E0-E1)+Ki*(E0+E1)+Kd*(E0-2*E1+E2);
  if Y>2048 then Y:=2048; if Y<0 then Y:=0;
  sleep(100);
  Edit13.text:=FloatToStr(Y);
  Edit11.Text:=FloatToStr(5*Y/2048);

{Çıkışa Yazma işlemleri}

  cikisverisi.chan := 0;
cikisverisi.BinData :=Round(2*Y);{giris 0 ile 2048 arasında değişiyor oysa çıkış 0 ile 4096 arasında};

hata := DRV_AOBinaryOut(tut,cikisverisi);
 case hata of
        Success : Edit5.Text := 'Çıkışa Yazma Başarılı'; {Sonucun mutlaka başarılı olması gerekiyor}

    end;
{Çıkışa Yazma işlemlerinin sonu}


  Until A=True;
  Close;
  end;

procedure TForm1.DurdurClick(Sender: TObject);
begin
 A:=True;  { Durdur Düğmesine Basana Kadar Döngü Devam Eder}
  DRV_DeviceClose(Tut);

end;

procedure TForm1.ProgramCikisiClick(Sender: TObject);
begin
Halt;
Close;

end;

procedure TForm1.GrafikCizClick(Sender: TObject);
var x,y:Integer;
begin
  resim := TBitmap.Create;   {Çizdirilecek grafik Dosyası oluşturulur}
  resim.Height :=260;
  resim.Width := 650;
  resim.PixelFormat := pf8bit;
  for  x:=1 to 24 do
  begin
   resim.canvas.Textout(9+23+25*x-3,200+15,'+');
   resim.canvas.TextOut(9+23+25*x-3,211+15,inttostr(5*x));
  end;
  resim.canvas.TextOut(205,240,'t:zaman (dk)');
  for y:=0 to 10 do
  begin
     resim.canvas.TextOut(0+9,20*y+15,inttostr(30-y));
```

```
    resim.canvas.Textout(15+9,20*y+15,'+');

  end;
  resim.canvas.TextOut(0,0,'T: Derinlik (cm)');
  resim.canvas.MoveTo(30,215);
  begin
   timer1.Enabled:=True;
  end;
end;

 procedure TForm1.Timer1Timer(Sender: TObject);

begin
 degernum:=degernum+1;
 siv:=round((215+6.5)-(normalizegirisverisi/10.291457286432160804020100502512 6));
 point := resim.Scanline[siv];
 point[degernum]:=0; {siyah renk}
 Form2.Canvas.Draw(0,0,resim);
    resim.canvas.LineTo(degernum,siv);
  resim.canvas.moveto(degernum,siv);

 Form2.Edit1.Text:=T;
 sayac:=sayac+1;
 if sayac=5 then
 begin
  zaman:=zaman+1;
  Form2.Edit2.Text:=IntToStr(zaman);
  sayac:=0;
  resim.SaveToFile('c:\PIDDerinlikGrafigi.bmp');

 end;

end;

end.



unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, TeeProcs, TeEngine, Chart, Series, StdCtrls;

type
  TForm2 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Label23: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    Label26: TLabel;
    Label27: TLabel;
    Label28: TLabel;
    Label29: TLabel;
```

```
    Label30: TLabel;
    Label31: TLabel;
    Label32: TLabel;
    Label33: TLabel;
    Label34: TLabel;
    Label35: TLabel;
    Label36: TLabel;
    Label37: TLabel;
    Label38: TLabel;
    Edit1: TEdit;
    Label39: TLabel;
    Label40: TLabel;
    Edit2: TEdit;
    Label41: TLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

end.
```

Yapay Sinir Ağları Eğitim ve Denetim Algoritması

Denetim programı , YSA'nın eğitimi ve eğitilmiş YSA'nın denetleyici olarak kullanılması olmak üzere iki kısımdan oluşmaktadır.

YSA Eğitim Adımları

Önce sabitler tanımlanır.

Limit =22 : Bir katmanda en fazla 22 nöron olabilir.

İterasyon = 10^6 : Sistem, her defasında "çiftler" tane eğitim çifti kullanılarak, toplam 10^6 adımda eğitiliyor. Her iterasyon sonunda öğrenme adımının büyüklüğü tekrar ayarlanıyor. Küme hatası, "Eski Küme Hatası" adıyla güncelleniyor.

Eta= 0.01 : η, öğrenme adım büyüklüğü

Momentum=0,6 : μ, ani sıçramaları önlemek için bir önceki ağırlık değişimleri ($\Delta w_{ij}(k)$), bu katsayıyla çarpılarak parametre güncelleme terimine $\Delta w_{ij}(k+1)$ eklenir.

Gamma= 0,01 : γ, maliyet işlevindeki değişim negatif çıkarsa (azalırsa), adım büyüklüğündeki değişme $\Delta\eta$, 0,01 olacaktır.

Beta=0,05 : (- $\beta\eta$,maliyet fonksiyonu artıyorsa (değişim pozitif çıkıyorsa) öğrenme adımındaki değişme $\Delta\eta$=-0,05*0,01 olacak ve böylece öğrenme adımı küçülecektir.

Tolerans=0,00001 : Hata bu değerin altına düşene kadar veya eğitim adımlarının sonuna kadar, eğitim süreci devam ediyor.

Ciftler : Eğitim çifti sayısı. Her iterasyonda çiftler sayısı kadar eğitim yapılır.

1. Global değişkenler tanımlanır

- Kayıt tipi değişkenler

W : TAğırlıklar Tipi. Ardarda iki katmanda bulunan herhangi iki nöron arasındaki ağırlık değerleri

w[i,j]: Şu anki ağırlık değeri

w1[i,j]: Bir önceki ağırlık değeri,

w2[i,j]: İki önceki ağırlık değeri

Vektör :Tvektor Tipi. Bir katmandaki i. nörona ait bilgiler (i=[1..Limit])

Out[i]: Çıkış değeri

Delta[i]: Delta değeri

Thr[i]: Eşik değeri

SumJ[i]: Nörona gelen girişlerin net toplamı

EğitimÇiftleri : Tgecici Tipi. Eğitimde kullanmak için girişe verdiğimiz girişler ve bunlara karşılık çıkıştan almak istediğimiz çıkış değerleri

X[i,j] : j. eğitim çifti için ilk katmandaki i. nöronun girişi

D[i,j] : j. eğitim çifti için son katmandaki i. nörondan istenen çıkış

- Değişkenler

L : Gizli Katman Numarası

Katman[i] : i. Katmandaki Nöron Sayısı

Katman[0] : Giriş Katmanındaki Nöron Sayısı

Katman[L+1] : Çıkış Katmanındaki Nöron Sayısı

Sayıcılar

Sayac : 0'dan iterasyon sayısına(10^6) kadar her 100'lük eğitim çifti kümesinin bitiminde bir artan sayıcı

Örnek : Eğitim çiftlerinden örnek alınması sırasında o an kaçıncı örnekte olduğumuzu gösteren ve 1'den eğitim çifti sayısına kadar birer artan sayıcı

Ornek_Hata :$Jr=\frac{1}{2}\sum_{i=1}^{katman(L+1)}(d_i - y_i)^2$ bireysel hatası. Her eğitimde, o örnek çifti için çıkış katmanındaki (L+1. Katman) nöronların hatalarının toplamıdır. Böylece o eğitim için çıkış nöronlarına ait toplam hata bulunur. Örnek=Çiftler olana kadar her örnekte(her eğitim çiftinde) sıfırlanıyor.

Kume Hatası : $E=\sum Jr=\sum_{i=0}^{çiftler} Ornek\_hata(i)$ toplumsal hatası. ciftler adet örnek çifte ait örnek hatalar bulunup toplanıyor. Her iterasyonda o eğitim çifti kümesi için toplam hata, yani maliyet ($\sum$Jr) hesaplanıyor. Maliyet istediğimiz toleransın altına düşmemiş ise veya iterasyon 10^6'ya ulaşmamışsa bir sonraki iterasyonla eğitime devam edilir. Küme hatası her iterasyonun sonunda eski küme hatasını güncellendikten sonra sıfırlanır.

Eski Küme Hatası : Her iterasyon sonunda küme hatası buraya aktarılır. Böylece bir önceki iterasyondaki küme hatasıyla şimdikini karşılaştırarak, eğitim sonunda elde edilen hatada artış ya da azalma olup olmadığını anlaşılır ve buna göre eğitim adımının yönü ve büyüklüğü yeniden belirlenir.

- Yordamlar

Eğitim Çiftlerini Kaydet : Eğitimde kullanılacak *çiftler* adet eğitim çiftini bularak "Eğitim Çifti Dosyası"na kaydeder. Bunun için girişe karttan 1 dakikalık aralıklarla ,0.05V'luk adımlarla artan gerilim verilir. Çıkışta sıcaklık ölçülür.

İleri Yönde Hesaplama İşlemleri:

Yapısal Bilgileri Al : Eğitim çiftlerinin tutulduğu dosyanın adı, gizli katman sayısı, giriş nöronlarının kaç adet, çıkış nöronlarının kaç adet olacağı, aynı şekilde her bir gizli katmanda kaçar adet nöron olacağı gibi önceden tanımladığımız sabitler giriliyor.

İlk Değerleri Ata : Her katmandaki, her *bir önceki katman (k. katman) nöronu* (j. nöron) ile her *bir sonraki katman (k+1. katman) nöronu (i. nöron)* arasındaki ağırlıklar (*W(k).w(ij))* rastgele olarak atanır.

Eğitim Çiftlerini Yükle : İlk katmandaki giriş veya girişler (x[i j]) ile bunlara karşılık son katmanda istenen çıkış veya çıkışlar (d[i j]) eğitim çiftlerinin tutulduğu dosyadan sırayla hafızaya yüklenir.

Eğitim Çiftlerini Göster : Eğitim çiftlerini aynı sırayla ekrana yazdırır.

İleri Yön Değerlerini Hesapla: i) Her bir nöron için önce net toplam bulunur (S=SumJ=$\left(\sum w_{ij}\, o_j\right) - thr(i)$). Yukarıda belirlediğimiz ağırlıklar (*W(k).w(ij)),* bir önceki katmandaki ilgili nöron çıkışıyla ($o_j = \psi(S)$ = Vektor[k].out[j]) çarpılır. Bütün ağırlıklı çıkışlar toplanır. Eşik seviyesi de eklenerek bir sonraki katmanın i. nöronuna ait net toplam giriş (S=Vektör[k+1].SumJ[i]) bulunur.

ii) Daha sonra, nöronun bulunduğu katman gizli katmansa (k<L+1) nonlineer geçiş işlevinden geçirilir ($y = \psi(s) = \tanh(S)$). Son katmansa (k=L+1), lineer geçiş işlevinden geçirilir. ($y = \psi(s) = S$). Böylece çıkışlar bulunur.

Çıkışları Göster : 'İleri yön değerlerini hesapla' yordamında hesaplanan çıkışlardan son katmana ait çıkışlar (Vektor[L+1] .out[j]) ekrana yazdırılır. Grafik olarak çizdirilir. Aynı işlem, her eğitim çifti için tekrarlanır.

İleri Yönde Hesaplama İşlemleri Burada Sona Eriyor

Hata Geriye Yayma İşlemleri Başlangıcı:

Çıkış Delta Değerlerini Hesapla : $\delta_i^{k+1} = \left(d_i - o_i^{k+1}\right)\psi'(S_i^{k+1})$, çıkış katmanındaki her bir nöronun delta değeri hesaplanır. Yani maliyet işlevinin ilgili nöronun çıkışına; ilgili nöronun çıkışının girişindeki net toplama göre

değişimi hesaplanır. Böylece maliyet işlevinin girişteki net toplama göre değişimi bulunur. j: çıkış katmanındaki kaçıncı nöron olduğunu gösterir. Çıkış nöronlarında geçiş işlevi lineer olduğundan türevi $\psi'\left(S_i^{k+1}\right)$ =1 dir. Bu nedenle yazmaya gerek yoktur. Hata=istenen çıkış - gerçek çıkış $\rightarrow$ e=d-y idi.

$\Rightarrow$ Vektor[L+1].delta[j]=(d(j, örnek)-Vektor[L+1].out[j])

Gizli Delta Değerlerini Hesapla : $\delta_i^{k+1} = \left(\sum_{h=1}^{n_{k+2}} \delta_i^{k+2} \, w_{hi}^{k+1} \right) \psi'(S_i^{k+1})$, her bir gizli katmandaki (k: katman sırası, j: ilgili katmanda kaçıncı nöron olduğunu gösterir) her bir nöronun delta değeri hesaplanır. $\Rightarrow$

Toplam = Toplam+Vektor[k+1].delta[h](bir sonraki katmanın deltası)*W[k].w[h j]

Aynı zamanda geçiş işlevinin türevi alınır, $\psi'\left(S_i^{k+1}\right)$ ve gizli delta değeri o nöron için hesaplanır. $\Rightarrow$

Vektor[k].delta[j] *{Bir önceki katmanın deltası}*=Toplam*Türev

Her iterasyon başlangıcında Toplam sıfırlanmalıdır.

Parametreleri Güncelle : $\Delta w_{ij}^k(k+1) = \mu \, \Delta w_{ij}^k(k) + \eta \, \delta_i^{k+1}$ o_j^k =MomentA+DIFF

Yukarıdaki parametre güncelleme kuralı uygulanır. İlk katmandan son katmana kadar, katmanlar arasındaki ağırlık matrisini (önceki katmanla sonraki katman arasındaki her bir j. nöronla i. nöron arasındaki ağırlığı) tarar ve düzeltir. kk+1: Bir sonraki katman,

kk: k. katman $\Rightarrow$

DIFF= eta*Vektor[kk+1].delta[i]*Vektor[kk].out[j]

W[kk].w2[i,j]:=W[kk].w1[i,j]; (bir önceki $\rightarrow$ iki öncekine atarılır)

W[kk].w1[i,j]:=W[kk].w[i,j]; (şu anki $\rightarrow$ bir öncekine atarılır)

$\Delta w_{ij}^k(k)$ = şu anki-bir önceki= W[kk].w1[i,j]-W[kk].w2[i,j]); $\Rightarrow$

MOMENTA:=Momentum*(W[kk].w1[i,j]-W[kk].w2[i,j]); $\Rightarrow$

$\Delta w_{ij}^k(k+1)$=DIFF+MOMENTA idi. Bu artım değerini ekleyerek şu anki ağırlığımızı güncellersek;

W[kk].w[i,j]:=W[kk].w[i,j]+DIFF+MOMENTA olur.

Benzer şekilde eşik seviyeleri de güncellenir; eşik: nöron girişi
(o_j^k) = -1 olan bir giriş kabul edilir.

Vektor[kk+1].thr[i]:= Vektor[kk+1].thr[i]+ $\underbrace{eta * Vektor[kk+1].delta[i] * (-1)}_{\Delta\theta = DIFF = \eta\delta o_j},$

Hata Geriye Yayma İşlemleri Başlangıcı Burada Sona Eriyor.

Eğitilmiş Ağ Parametrelerini Kaydet : Eğitilmis ağ parametreleri, kacıncı iterasyonda olduğunu belirten sayac bilgisiyle birlikte bir dosyaya kaydedilir. Şayet iterasyon bitmeden küme hatası istenen tolerans değerine ulaşırsa bilgiler kaydedilir ve sayac iterasyon sonu olan 10^6ya ulaşmadan eğitim durdurulur programdan çıkılır.
Bu dosyada gizli katman sayısı, giriş ve çıkış vektörü boyutları, her bir gizli katmanda bulunan nöron sayıları rapor edilir. Daha sonra eğitilmiş ağ parametreleri (her katmandaki her bir nöron ile bir sonraki katmandaki her bir nöron arasındaki ağırlıklar) sırayla kaydedilir.

Eğitilmis YSA'nı Sına : Daha önce eğitim sırasında kullanılmamış olan test çiftleri oluşturulur ve Sınama Çiftleri adı altında kaydedilir. Bu yordam eğitilmiş ağ parametrelerini yükler, istenen sınama verileri ile ağı test eder. Ağın bilmediği girişler için doğru çıkışları sağlayıp sağlayamadığı gözler. Başarım ölçütü olarak küme hatası ekrana yazdırılır.

SınamaCiftleriniYukleVeGoster: Giriş KAtmanı ve Çıkış Katmanındaki nöronlarda sınama için kullanılacak değerler sırayla hafızaya yüklenir.

SınamaCikislariniGoster: Eğitilmiş Ağın Sınama girişleri, buna karşılık olması gereken çıkışlar ve Eğitilmiş ağın verdiği gerçek çıkışlar ekrana yazdırılır. Sınama girişlerine eğitilmiş ağın cevabı çizdirilir.

- İşlevler

Nonlineer İşlev(Toplam, eşik) : "İleri Yön Değerlerini Hesapla" yordamında kullanılan bu işlev gizli katmanlardaki aktivasyon fonksiyonlarını hesaplar.

$$\psi(s) = \tanh(S) = \tanh\left(\left(\sum w_{ij}\, o_j\right) - thr(i)\right) = \tanh(toplam - esik)$$

Nonlineer İşlev Türevi(Toplam) : "Gizli Delta Değerlerini Hesapla" yordamında kullanılan bu işlev gizli katmanlardaki aktivasyon fonksiyonlarınının türevlerini hesaplar.

Türev= $\psi'(S) = 1 - toplam^2$

YSA Denetim Adımları

Global değişkenler tanımlanır

- Kayıt tipi değişkenler

w[i,j] : TAğırlıklar Tipi. Ardarda iki katmanda bulunan herhangi iki nöron arasındaki ağırlık değerleri

Vektör :Tvektor Tipi. Bir katmandaki i. nörona ait bilgiler (i=[1..Limit])

Out[i]: Çıkış değeri

Delta[i]: Delta değeri

Thr[i]: Eşik değeri

SumJ[i]: Nörona gelen girişlerin net toplamı

- Değişkenler

L : Gizli Katman Numarası

Katman[i] : i. Katmandaki Nöron Sayısı

Katman[0] : Giriş Katmanındaki Nöron Sayısı

Katman[L+1] : Çıkış Katmanındaki Nöron Sayısı

Sayıcılar

zaman : Eğitim çiftlerinden örnek alınması sırasında o an kaçıncı örnekte olduğumuzu gösteren ve 1'den eğitim çifti sayısına kadar birer artan sayıcı

- Yordamlar

İleri Yön Değerlerini Hesapla: i) Her bir nöron için önce net toplam bulunur (S=SumJ=$\left(\sum w_{ij}\, o_j\right) - thr(i)$). Yukarıda belirlediğimiz ağırlıklar (*W(k).w(ij)),* bir önceki katmandaki ilgili nöron çıkışıyla ($o_j = \psi(S) =$ Vektor[k].out[j]) çarpılır. Bütün ağırlıklı çıkışlar toplanır. Eşik seviyesi de eklenerek bir sonraki katmanın i. nöronuna ait net toplam giriş (S=Vektör[k+1].SumJ[i]) bulunur.

ii) Daha sonra, nöronun bulunduğu katman gizli katmansa (k<L+1) nonlineer geçiş işlevinden geçirilir ($y = \psi(s) = \tanh(S)$). Son katmansa (k=L+1), lineer geçiş işlevinden geçirilir. ($y = \psi(s) = S$). Böylece çıkışlar bulunur.

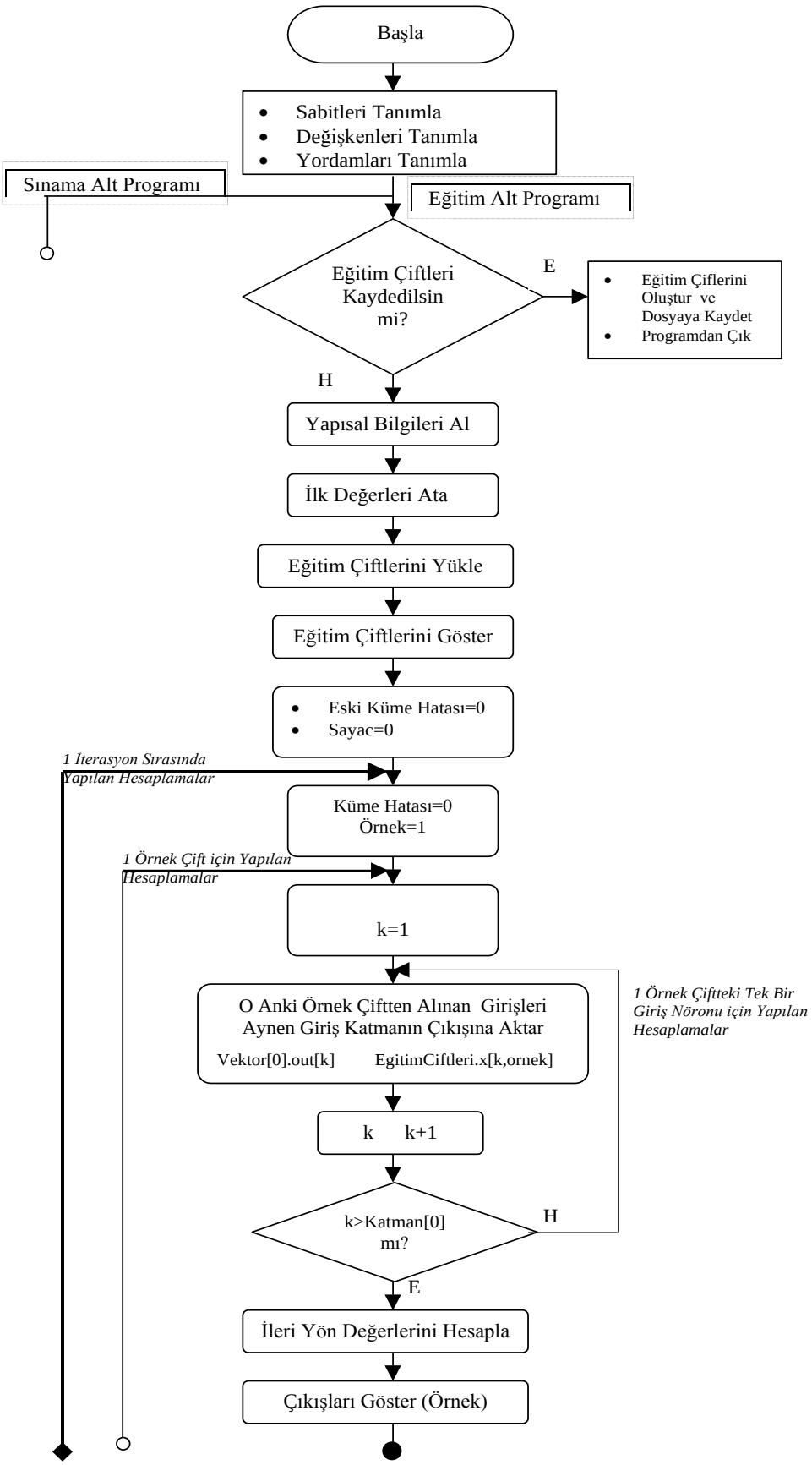
Çıkışları Göster : 'İleri yön değerlerini hesapla' yordamında hesaplanan çıkışlardan son katmana ait çıkışlar (Vektor[L+1] .out[j]) ekrana yazdırılır. Grafik olarak çizdirilir. Aynı işlem, her eğitim çifti için tekrarlanır.

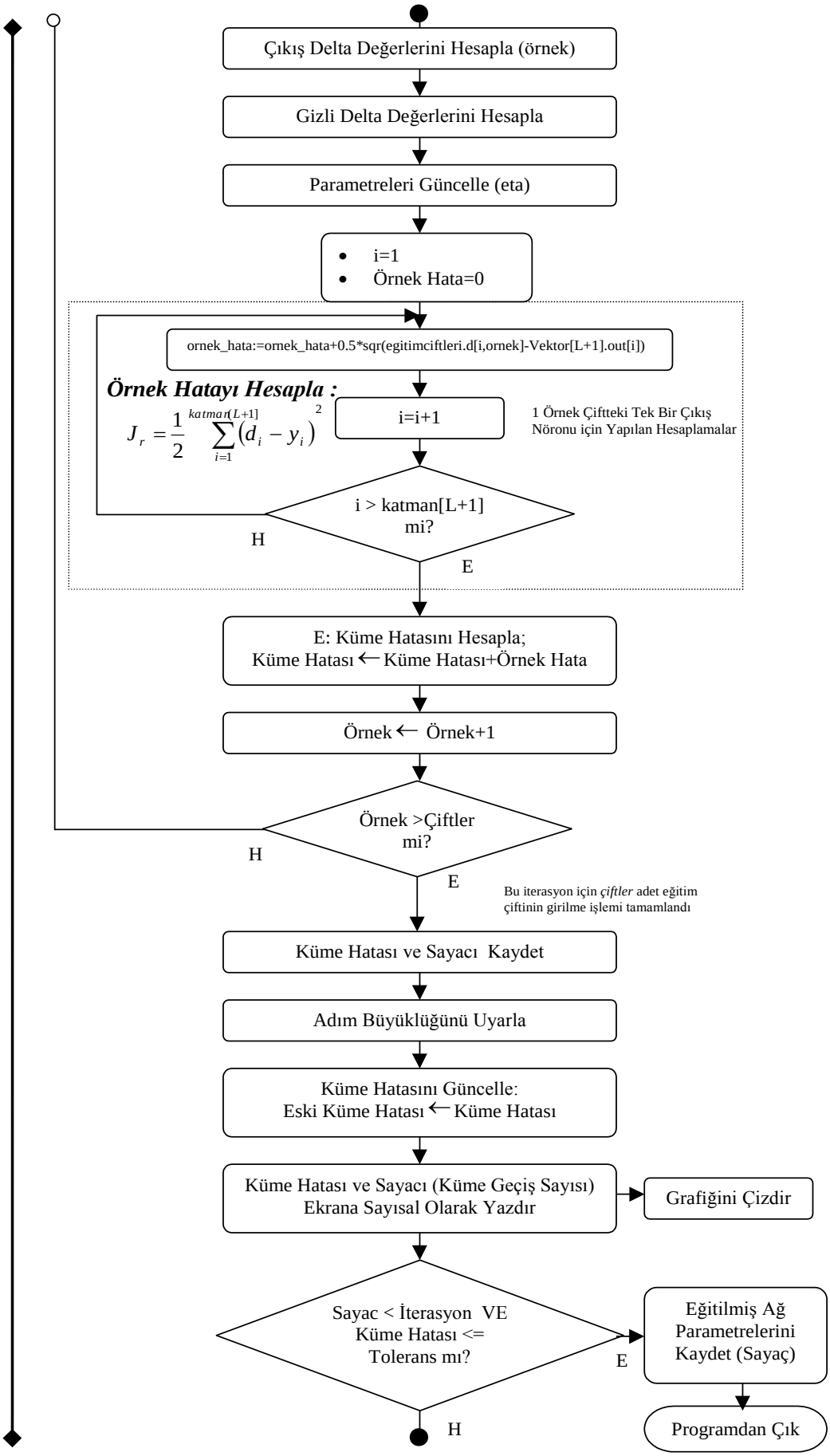
- İşlevler

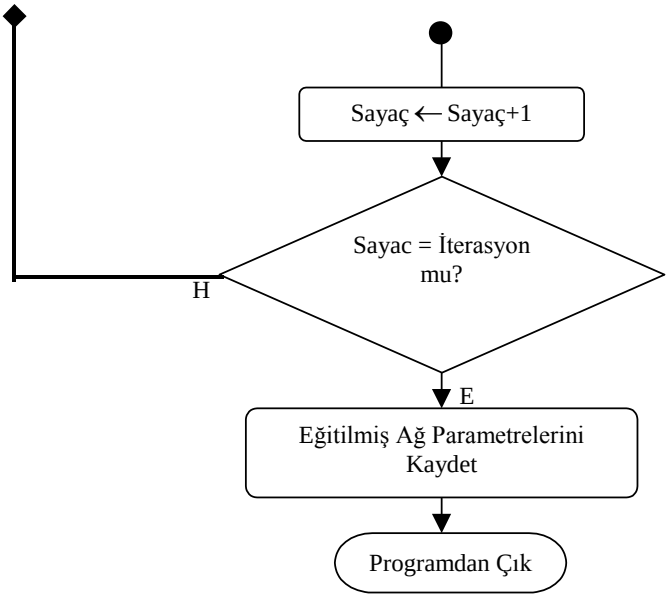
Nonlineer İşlev(Toplam, eşik) : "İleri Yön Değerlerini Hesapla" yordamında kullanılan bu işlev gizli katmanlardaki aktivasyon fonksiyonlarını hesaplar.

$$\psi(s) = \tanh(S) = \tanh\left(\left(\sum w_{ij}\, o_j\right) - thr(i)\right) = \tanh(toplam - esik)$$

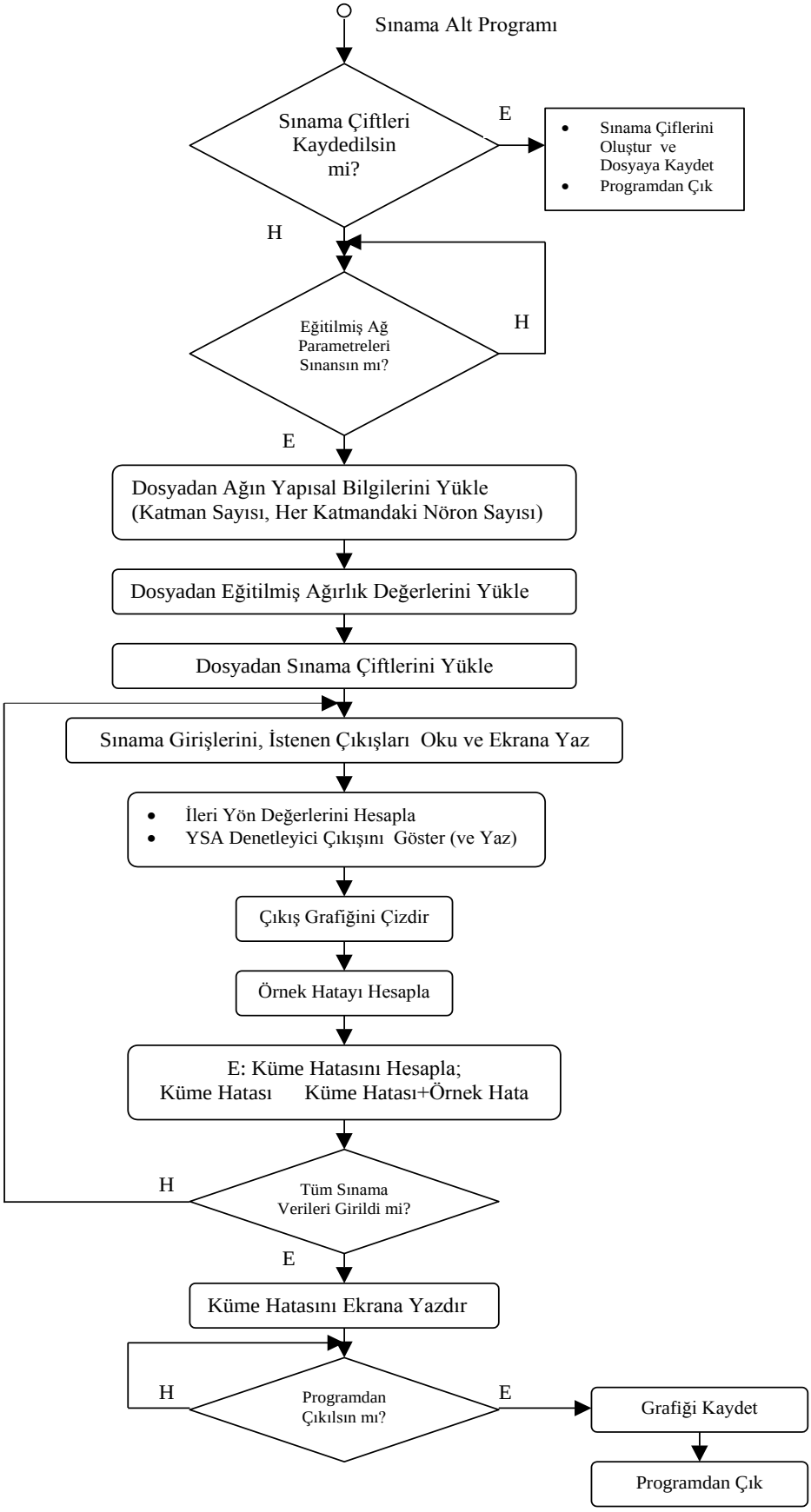
YSA EĞİTİM ALGORİTMASI




Çıkış Delta Değerlerini Hesapla (örnek)
Gizli Delta Değerlerini Hesapla
Parametreleri Güncelle (eta)
i=1
Örnek Hata=0
ornek_hata:=ornek_hata+0.5*sqr(egitimciftleri.d[i,ornek]-Vektor[L+1].out[i])
Örnek Hatayı Hesapla :
$J_r = \frac{1}{2} \sum_{i=1}^{katman[L+1]} (d_i - y_i)^2$
i=i+1
1 Örnek Çiftteki Tek Bir Çıkış Nöronu için Yapılan Hesaplamalar
i > katman[L+1] mi?
H
E
E: Küme Hatasını Hesapla;
Küme Hatası ← Küme Hatası+Örnek Hata
Örnek ← Örnek+1
Örnek >Çiftler mi?
H
E
Bu iterasyon için *çiftler* adet eğitim çiftinin girilme işlemi tamamlandı
Küme Hatası ve Sayacı Kaydet
Adım Büyüklüğünü Uyarla
Küme Hatasını Güncelle:
Eski Küme Hatası ← Küme Hatası
Küme Hatası ve Sayacı (Küme Geçiş Sayısı) Ekrana Sayısal Olarak Yazdır
Grafiğini Çizdir
Sayac < İterasyon VE Küme Hatası <= Tolerans mı?
E
H
Eğitilmiş Ağ Parametrelerini Kaydet (Sayaç)
Programdan Çık


Sayaç ← Sayaç+1
Sayac = İterasyon mu?
H
E
Eğitilmiş Ağ Parametrelerini Kaydet
Programdan Çık

YSA SINAMA ALTPROGRAMI



YSA Eğitim Programı

```
program NeuroDeneme_Project9;

uses
  Forms,
  Form1NeuroDeneme_Unit9 in 'Form1NeuroDeneme_Unit9.pas' {Form1},
  Form2EgitimCifleri_Unit9 in 'Form2EgitimCifleri_Unit9.pas' {Form2},
  Form3YSAEgitimGrafigi_Unit9 in 'Form3YSAEgitimGrafigi_Unit9.pas' {Form3};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm3, Form3);
  Application.Run;
end.



unit Form1NeuroDeneme_Unit9;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, FileCtrl,Math;
 const
    ciftler=360;
type
   Tagirliklar=Record
     w: array [1..2,1..2] of Real;
     w1: array [1..2,1..2] of Real;
     w2: array [1..2,1..2] of Real;
    end;
    TVektor=Record
     out: array [1..2] of Real; {çıkış değerleri}
     delta: array [1..2] of Real; {Delta Değerleri}
     thr: array [1..2] of Real; {Eşik Değerleri}
     SumJ: array [1..2] of Real;{ilgili Nöronun Ağırlıklı Toplamı}
    end;
    Tgecici=Record
     x:array[1..1,0..360] of Real;  {girişler}
     d:array[1..1,0..360] of  Real; {Çıkışlar}
    end;


  TForm1 = class(TForm)
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    Edit3: TEdit;
    Edit4: TEdit;
    Label5: TLabel;
    Edit5: TEdit;
    Label6: TLabel;
    Edit6: TEdit;
    Label7: TLabel;
    Edit7: TEdit;
    Label9: TLabel;
    Edit8: TEdit;
    Label10: TLabel;
    Edit9: TEdit;
    Label11: TLabel;
    Edit10: TEdit;
    Baslat: TButton;
    EgitimCiftleriniKaydet: TButton;
    Kapat: TButton;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
```

```
    procedure FormCreate(Sender: TObject);
    procedure BaslatClick(Sender: TObject);
    procedure EgitimCiftleriniKaydetClick(Sender: TObject);
    procedure KapatClick(Sender: TObject);


  private
    { Private declarations }
  public
    { Public declarations }
  end;

  const
   Yedekle=1000;{ * Eğitilmiş Ağ Parametrelerinin Düzenli olarak Yedeklenip kay
                dedilmesini sağlar. 1000 defa. Her eğitimden sonra 1 azalır}

var

  Form1: TForm1;
  resim1,resim2,resim3:TBitmap;
  renkkodu,degernum:integer;
  basla: boolean;
  limit,iterasyon,L:Integer;
  eta,momentum,gamma,beta,tolerans,kume_hatasi,eski_kume_hatasi,ornek_hata:real;
  katman:array [0..3] of Integer;
  egitimciftleri:Tgecici;
  W:array  [0..3] of Tagirliklar;  {Ağırlıklar kayıdı}
  Vektor: Array [0..3] of TVektor; {Bir katmandaki nöronların çıkış, }
                                   { delta, toplam, eşik  değerleri}

  sayac,ornek,line_to:integer;    { Sayaç 0 dan iterasyon sayısına kadar(1 000 000),
                          her iterasyonda Eğitim çifti sayısı kadar (180 tane)
                           örnek alır}


implementation

uses Form2EgitimCifleri_Unit9, Form3YSAEgitimGrafigi_Unit9;

{$R *.DFM}

procedure TForm1.EgitimCiftleriniKaydetClick(Sender: TObject);
Type
 Kayit=Record
  Giris:Array [1..1] of String[5];
  Cikis:Array [1..1] of String[5];
 End;
Var
 Dosya : File of Kayit;
 EgitimCifti: Kayit;
 zaman:integer;
begin
if form1.SaveDialog1.Execute then
begin
  AssignFile(Dosya,form1.SaveDialog1.FileName);
  Rewrite(Dosya);

 For zaman:=0 to 360 do
 Begin

  EgitimCifti.giris[1]:=FloatToStr(zaman);
  EgitimCifti.cikis[1]:=FloatTosTrf(Sin(pi*zaman/180),fffixed,5,3); {Buradaki 180, açıyı Delphi'nin işlem yaptığı radyana
çevirmek için}
  Seek(Dosya,FileSize(Dosya));
  Write(Dosya,EgitimCifti);
end;
 End;
 CloseFile(Dosya);
 Close;
end;

procedure YapisalBilgileriAl;
 Var
 i:integer;
 InputString: string;
begin
   limit:= StrToInt(Form1.edit1.text);
```

```
  iterasyon:=StrToInt(Form1.edit2.text);
  eta:=StrToFloat(Form1.edit3.text);
  momentum:=StrToFloat(Form1.edit4.text);
  gamma:=StrToFloat(Form1.edit5.text);
  beta:=StrToFloat(Form1.edit6.text);
  tolerans:=StrToFloat(Form1.edit7.text);

  L:=StrToInt(Form1.edit8.text);

 katman[0]:= StrToInt(Form1.edit9.text);
  katman[L+1]:=StrToInt(Form1.edit10.text);

  for i:=1 to L do
  begin

   InputString:= InputBox('Gizli Katman Nöronları', IntToStr(i)+'.Katmandaki Nöron Sayısını Giriniz','2');

   katman[i]:=StrToInt(InputString);

  end;
end;



Procedure ilkDegerleriAta;  {ilk Ağırlık Değerleri Atanır}
var
 i,j,k:integer;
Begin
  randomize ;
  for k:=0 to L do
  Begin
   for i:=1 to katman[k+1] do
   Begin
    for j:=1 to katman[k] do
    Begin
       W[k].w[i,j]:=random(10000)/10000-0.5 ; {-0.5 ile +0.5 arasında 0.00001}
       W[k].w1[i,j]:=random(10000)/10000-0.5 ;{hassasiyetle ratgele değerler}
       W[k].w2[i,j]:=random(10000)/10000-0.5 ;  {üretir}
    End;
    Vektor[k].thr[i]:= random(10000)/10000-0.5 ;
   End;
  End;

End;


Procedure EgitimCiftleriniYukleVeGoster;{ Giriş KAtmanı ve Çıkış Katmanındaki nöronlarda
                    Eğitim için kullanılacak  değerler sırayla hafızaya  yüklenir}
Type

 Kayit=Record
  Giris:Array [1..1] of String[5];
  Cikis:Array [1..1] of String[5];
 End;

 var
 i,j,Kod:integer;
 Ekran:Kayit ;
 Dosya: File of Kayit;



Begin
Form2.show;
AssignFile(Dosya,form1.OpenDialog1.FileName);
{$i-} Reset(Dosya);Kod:=IOResult;{$I+};
if Kod<>0 Then
Begin
if Application.MessageBox(
       'Dosya Yok_Programdan çıkalımmı',

       'Open Error',
       MB_OKCANCEL + MB_DEFBUTTON1) <> IDOK then    Halt;

End;

   j:=0;
```

```
 Repeat
    seek(Dosya,j);
    read(Dosya,Ekran);
    Form2.ListBox1.items.add( intToStr(j)+'.örnek Giriş');
    Form2.ListBox2.items.add( intToStr(j)+'.örnek Çıkış');
  for i:=1 to 1 do
  Begin

   Form2.ListBox1.items.add( intToStr(i)+'. Giris : '+Ekran.giris[i]);
   EgitimCiftleri.x[i,j]:=StrToFloat(Ekran.giris[i])/360;{Buradaki 360'a bölme işlemi, girişi 0 ile 1 arasına normalize
etmek için}
   Form2.ListBox1.Refresh;

  End;
  for i:=1 to 1 do
  Begin
   Form2.ListBox2.items.add( intToStr(i)+'. Cıkıs : '+Ekran.cikis[i]);
  EgitimCiftleri.d[i,j]:=StrToFloat(Ekran.cikis[i]);
   Form2.ListBox2.Refresh;
  End;
  j:=j+1;
  {Sleep(1000);}
  Application.ProcessMessages;

 Until Eof(Dosya);
 CloseFile(Dosya);

End;


Function Nonlineerislev(toplam,esik:Real):Real;
Begin
 Nonlineerislev:=Tanh((toplam-esik)*pi/180);
End;


Procedure ileriYonDegerleriniHesapla;
Var
 i,j,k:integer;
 Toplam:Real;

Begin
 for k:=0 to L do
 Begin
 For i:=1 to katman[k+1] do
  Begin
  Toplam:=0;
   for j:=1 to katman[k] do
   Begin
    Toplam:=Toplam+W[k].w[i,j]*Vektor[k].out[j];

   End;
   Vektor[k+1].Sumj[i]:=Toplam;
  if k<L then Vektor[k+1].out[i]:=Nonlineerislev(Vektor[k+1].Sumj[i],Vektor[k+1].thr[i])
  Else Vektor[k+1].out[i]:=Toplam-Vektor[k+1].thr[i];
  End;
 End;
End;

Procedure CikislariGoster(ornekcikis:integer);
Var

 j,zaman1:integer;
  x1,y1,Derinlik1,x2,y2,Derinlik2,x3,y3,Derinlik3:integer;
  Point1,Point2,Point3:PByteArray;
  NormalizeKume_Hatasi:Real;

Begin    { Main}
  if ornekcikis=0 then
  Begin  {I.1}
   Form2.ListBox3.items.add(' ');
   Form2.ListBox3.items.add(' ');
   Form2.ListBox3.items.add( intToStr(sayac)+'.İterasyonda');
   Form2.ListBox3.items.add(' ');
  End; {I.1 son}
```

```
 Form2.ListBox3.items.add('  '+intToStr(ornekcikis)+'.Örneğe ait');
 for j:=1 to katman[L+1] do
 Begin {I.2}
 Form2.ListBox3.items.add('    ' +intToStr(j)+'. Çıkış : '+FloatToStr(Vektor[L+1].out[j]));
 Form2.Listbox3.refresh;
 End; {I.2 son}
 Application.ProcessMessages;

 {Sadece Son İterasyona ait Çıkış Grafiğini Çizdir}
 begin  {II.1}

 if ornekcikis=0 then
 Begin{II.2}
    resim1.Create;
    renkkodu:=renkkodu+1;
    resim1.Height:=260;
    resim1.width:=420;
    resim1.PixelFormat:=pf8bit;
    for x1:=0 to 24 do
    Begin {II.3}
     resim1.Canvas.TextOut(32+15*x1,130,'+');
    End;  {II.3 son}

   for x1:=0 to 12 do
   Begin  {II.4}
    resim1.canvas.textout(35+30*x1,141,inttostr(30*x1));
   End; {II.4 son}
   for y1:=0 to 10 do
   Begin {II.5}
    resim1.canvas.textout(20,40+18*y1,'+');
   End;  {II.5 son}

   for y1:=0 to 10 do
   Begin {II.6}
    resim1.canvas.textout(0,40+18*y1,FloatToStr(1-0.2*y1));
   End; {II.6 son}

      resim1.canvas.Font.color:=Renkkodu;
      resim1.canvas.textout(250,5,IntTostr(sayac)+'.İterasyon Çıkışı');
      resim1.Canvas.Font.color:=0;
      resim1.canvas.textout(0,5,'Derinlik (+1,-1)');
      resim1.canvas.textout(360,120,'Açı(Derece)');

 End; {II.2 Son}


Derinlik1:=round(260-124-90*(Vektor[L+1].out[1]));
point1:=resim1.Scanline[Derinlik1];
point1[35+ornekcikis]:=renkkodu;
Form3.Canvas.Draw(3,0,Resim1);
End; {II.1 son}

 {Çıkış Grafiğini Çizdir}
  begin {III.1}
 if ornekcikis=0 then
 Begin {III.2}
    resim2.Height:=260;
    resim2.width:=420;
    resim2.PixelFormat:=pf8bit;
    for x2:=0 to 24 do
    Begin {III.3}
     resim2.Canvas.TextOut(32+15*x2,130,'+');
    End; {III.3 son}

   for x2:=0 to 12 do
   Begin {III.4}
    resim2.canvas.textout(35+30*x2,141,inttostr(30*x2));
   End; {III.4 son}
   for y2:=0 to 10 do
   Begin {III.5}
    resim2.canvas.textout(20,40+18*y2,'+');
   End;  {III.5 son}

   for y2:=0 to 10 do
   Begin  {III.6}
    resim2.canvas.textout(0,40+18*y2,FloatToStr(1-0.2*y2));
   End; {III.6 son}
```

```
   Resim2.canvas.Font.color:=renkkodu;
    resim2.canvas.textout(250,5,IntTostr(sayac)+'.İterasyon Çıkışı');
    resim2.Canvas.Font.color:=0;
    resim2.canvas.textout(0,5,'Derinlik (+1,-1)');
    resim2.canvas.textout(360,120,'Zaman(sn)');

 End;{III.2 son}


 Derinlik2:=round(260-124-90*(Vektor[L+1].out[1]));
 point2:=resim2.Scanline[Derinlik2];
 point2[35+ornekcikis]:=renkkodu;
 Form3.Canvas.Draw(3,270,Resim2);


end; {III.1 son}


{Hata Grafiğini Çizdir}
 begin {IV.1}
  if ornekcikis=0 then
  Begin {IV.2}


     for x3:=0 to 24 do
     Begin {IV.3}
      {resim3.Canvas.TextOut(32+15*x3,440,'+');}
     End; {IV.3 son}

    {for x3:=0 to 12 do
    Begin {IV.4}
     {resim3.canvas.textout(35+30*x3,141,inttostr(30*x3));
    End; {IV.4 son}


    for y3:=0 to 20 do
    Begin {IV.5}
     resim3.canvas.textout(20,40+20*y3,'+');
    End;  {IV.5 son}
    resim3.canvas.textout(0,0,'Jr(Küme Hatası)');
    resim3.canvas.textout(35,15,'(4,2 den daha büyük hatalar=4.2)');
    resim3.canvas.textout(325,435,'İterasyon');

    resim3.canvas.textout(0,20,'4,2 +');
    for y3:=0 to 20 do
    Begin  {IV.6}
     resim3.canvas.textout(0,40+20*y3,FloatToStr(4-0.2*y3));
    End; {IV.6 son}
     Resim3.canvas.Font.color:=renkkodu;
     resim3.canvas.textout(250,5,IntTostr(sayac)+'.Küme Hatası');

 End;{V.2 son}


 if ornekcikis=360 then
 begin   {küme hatası ciziliyor}
  degernum:=degernum+5;

  if Kume_Hatasi>4.2 then NormalizeKume_Hatasi:=4.2
  Else NormalizeKume_Hatasi:=Kume_Hatasi;
  Derinlik3:=round(resim3.Height -100*NormalizeKume_Hatasi);
   {point3:=resim3.Scanline[Derinlik3];
  point3[30+degernum]:=0;}
  if sayac=1 then
      begin
        Resim3.Canvas.MoveTo(25,20);
        line_to := 25;

      end
      else
      begin
        Resim3.Canvas.MoveTo(25+degernum-5,line_to);
        Resim3.Canvas.LineTo(25+degernum,Derinlik3);
        line_to := Derinlik3;
      end;

  Form3.Canvas.Draw(430,80,Resim3);
```

```
 End;  {küme hatası çiziliyor-Son}

end; {III.1 son}

End; {Main Son}



Function NonlineerislevTurevi(cikis:Real):Real;
Begin
 NonlineerislevTurevi:=(1-sqr(cikis));
End;


Procedure CikisDeltaDegerleriniHesapla(ornekdegeri:integer);   {çıkış katmanındaki katman [L+1]her bir}
 Var                 { nöronun delta değeri (e=d-y) hesaplanır,j çıkış katmanındaki kaçıncı nöron olduğunu gösterir}
  j:integer;
Begin
 for j:=1 to katman[L+1] do
 Begin
  Vektor[L+1].delta[j]:=(Egitimciftleri.d[j,ornekdegeri]-Vektor[L+1].out[j]);
 End;
End;

Procedure GizliDeltaDegerleriniHesapla;
{Gizli katmandaki her bir nöronun delta değerini hesaplar}
Var
 Toplam,Turev:Real;
 i,j,k:integer;{k katman sırasını gösterir}
Begin{main}
For k:=L downto 1 do
 Begin {1}
  for j:=1 to katman[k] do
  Begin{2}
   Toplam:=0;
   for i:=1 to katman[k+1] do
   Begin {3}
    Toplam:=Toplam+Vektor[k+1].delta[i]*W[k].w[i,j];
   End;  {3}
   Turev:=NonlineerislevTurevi(Vektor[k].out[j]);
   Vektor[k].delta[j]:=Toplam*Turev;
  End{2}

 End;{1}
End;{main}

Procedure ParametreleriGuncelle(eta_adimbuyuklugu:Real);
Var
 kk,i,j:integer;
 DIFF,MOMENTA:Real;
Begin{main}
 For kk:=0 to L do  {kk katmanlar arasındaki ağırlık matrislerini tarar}
 Begin{1}
  for i:=1 to katman[kk+1] do   {i,bir sonraki katmandaki bütün nöronları sırayla tarar}
  Begin;{2}
   for j:=1 to katman[kk] do  {j, bir önceki katmandaki bütün nöronları sırayla tarar}
   Begin{3}
     DIFF:=eta_adimbuyuklugu*Vektor[kk+1].delta[i]*Vektor[kk].out[j];
     W[kk].w2[i,j]:=W[kk].w1[i,j];
     W[kk].w1[i,j]:=W[kk].w[i,j];
     MOMENTA:=Momentum*(W[kk].w1[i,j]-W[kk].w2[i,j]);
     W[kk].w[i,j]:=W[kk].w[i,j]+DIFF+MOMENTA;
   End;{3}
   Vektor[kk+1].thr[i]:= Vektor[kk+1].thr[i]+ eta_adimbuyuklugu*Vektor[kk+1].delta[i]*(-1);
  End;{2}
 End;{1}
End;{main}

Procedure EgitilmisAgParametreleriniKaydet(Sayacdegeri:Integer);

Var
i,j,matris,Kod:integer;
Dosya : File of String[100];
s : string[100];

Begin {1}
```

```
if Form1.SaveDialog1.Execute then
begin
 AssignFile(Dosya,form1.SaveDialog1.FileName);
 {$i-}
 Reset(Dosya);
 Kod:=IOResult;
 {$I+}
  if Kod<>0 Then Rewrite(Dosya);

    Seek(Dosya,FileSize(Dosya));
    s := 'Eğitilmiş Ağ Parametreleri';
    Write(Dosya,s);

    Seek(Dosya,FileSize(Dosya));
    s := 'İterasyon : ' + inttostr(Sayacdegeri);
    Write(Dosya,s);

    Seek(Dosya,FileSize(Dosya));
    s := 'Gizli Katman Sayisi : '+ inttostr(L);
    Write(Dosya,s);
    Seek(Dosya,FileSize(Dosya));
    s := 'Giris Vektoru Boyutu : '+ inttostr(katman[0]);
    Write(Dosya,s);
    Seek(Dosya,FileSize(Dosya));
    s := 'Cıkıs Vektoru Boyutu : '+ inttostr(katman[L+1]);
    Write(Dosya,s);

 For i:=1 to L do
 Begin {3}
  Seek(Dosya,FileSize(Dosya));
  s := inttostr(i) + '. Gizli Katmandaki Nöron Sayısı : ' + inttostr(katman[i]);
  Write(Dosya,s);
 End;{3}
  Seek(Dosya,FileSize(Dosya));
  s := '========AĞIRLIK MATRİSLERİ========';
  Write(Dosya,s);
  For Matris:=0 to L do
  Begin{4}
   for i:=1 to katman[Matris+1] do
   Begin{5}
    for j:=1 to katman[matris] do
    Begin {6}
      Seek(Dosya,FileSize(Dosya));
      s := 'W['+inttostr(matris)+ '].w['+ inttostr(i)+ inttostr(j)+ '] = ' + floattostr(W[Matris].w[i,j]);
      Write(Dosya,s);
    End; {6}
    Seek(Dosya,FileSize(Dosya));
    s := 'Vektör['+ inttostr(matris+1) + '].thr[' + inttostr(i)+ '] = ' + floattostr(Vektor[Matris+1].thr[i]);
    Write(Dosya,s);
   End; {5}
   Seek(Dosya,FileSize(Dosya));
   s := '  ';
    Write(Dosya,s);
  End;{4}

    CloseFile(Dosya);
 end;
End;{1}

Procedure EgitilmisAgParametreleriniYSADenetleyiciicinKaydet;

Var
i,j,matris,Kod:integer;
Dosya : File of String[100];
s : string[100];

Begin {1}
if Form1.SaveDialog1.Execute then
begin
 AssignFile(Dosya,form1.SaveDialog1.FileName);
 {$i-}
 Reset(Dosya);
 Kod:=IOResult;
 {$I+}
  if Kod<>0 Then Rewrite(Dosya);

   Seek(Dosya,FileSize(Dosya));
```

```
   s := inttostr(L);
   Write(Dosya,s);           {Katman Sayısını}

  Seek(Dosya,FileSize(Dosya));
  s := inttostr(katman[0]);
  Write(Dosya,s);              {Giriş Vektörü Boyutunu}
  Seek(Dosya,FileSize(Dosya));
  s := inttostr(katman[L+1]);
  Write(Dosya,s);              {Çıkış Vektörü boyutunu}

 For i:=1 to L do
 Begin {3}
  Seek(Dosya,FileSize(Dosya));
  s :=inttostr(katman[i]);   {i. Gizli Katmandaki Nöron Sayısını gösterir}
  Write(Dosya,s);
 End;{3}


  For Matris:=0 to L do
  Begin{4}
   for i:=1 to katman[Matris+1] do
   Begin{5}
    for j:=1 to katman[matris] do
    Begin {6}
      Seek(Dosya,FileSize(Dosya));
      s:= floattostr(W[Matris].w[i,j]);
      Write(Dosya,s);
    End; {6}
    Seek(Dosya,FileSize(Dosya));
    s:=floattostr(Vektor[Matris+1].thr[i]);
    Write(Dosya,s);
   End; {5}

  End;{4}
    Seek(Dosya,FileSize(Dosya));
    s:=' ';
    Write(Dosya,s);
    CloseFile(Dosya);
 end;
End;{1}

procedure TForm1.FormCreate(Sender: TObject);

 begin
 Basla := False;
  edit1.text:=IntToStr(22);
  edit2.text:=IntToStr(1);
  edit3.text:=FloatToStr(0.01);
  edit4.text:=FloatToStr(0.6);
  edit5.text:=FloatToStr(0.0001);
  edit6.text:=FloatToStr(0.05);
  edit7.text:=FloatToStr(0.000001);

  edit8.text:=IntToStr(2);
  edit9.text:=IntToStr(1);
  edit10.text:= IntToStr(1);

 Form1.show;


end;

procedure TForm1.BaslatClick(Sender: TObject);{*     *     *     *     ***   ***}
var                                              {* *    * *  *   * *    *   * *   *}
  i,k:Byte;                                     {*****  *   * *  *****   * *   * *}
  InputString: string;                         {*      * *    * *     * *     *   *}

begin

if form1.OpenDialog1.Execute then
begin
   Application.ProcessMessages;
   Resim1:=TBitmap.Create;
   Resim2:=TBitmap.Create;
   Resim3:=TBitmap.Create;
   resim3.Height:=450;
   resim3.width:=368;
```

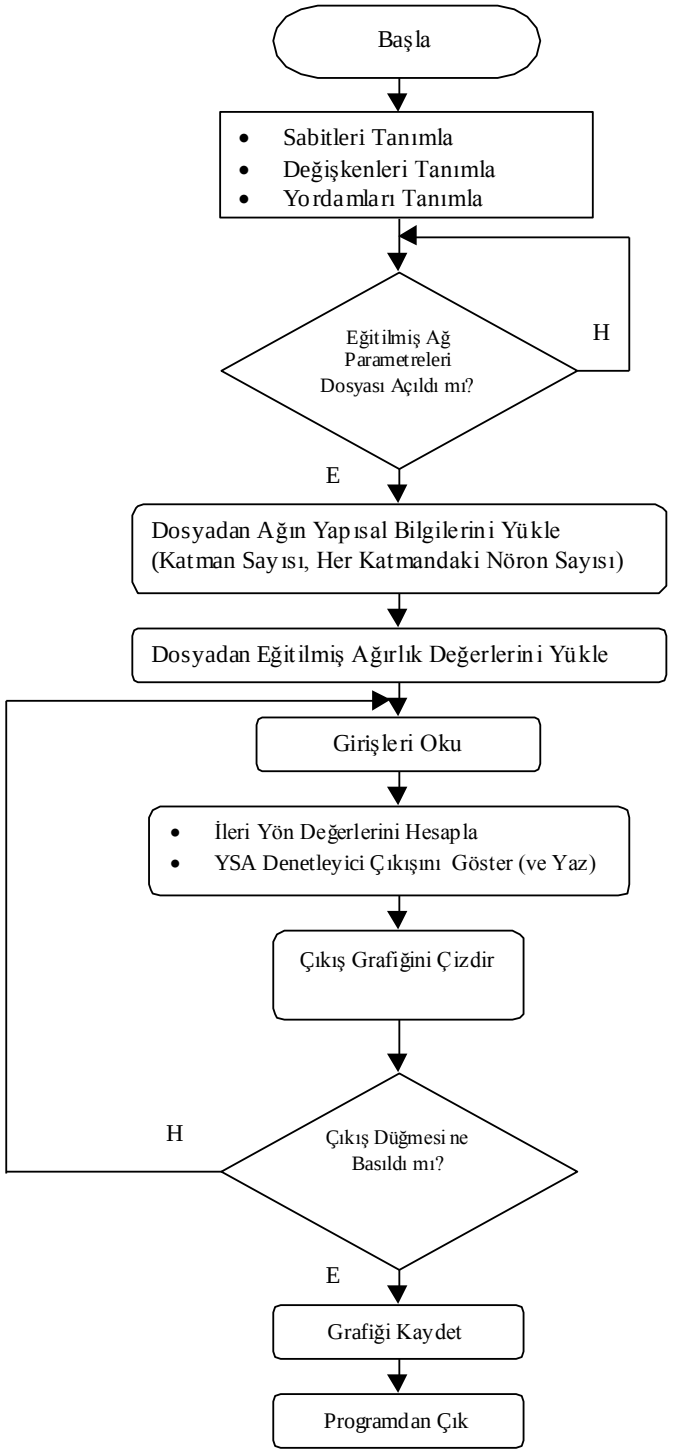
```
 resim3.PixelFormat:=pf8bit;
 renkkodu:=0; degernum:=0;
 Form3.Show;
 YapisalBilgileriAl;
 ilkDegerleriAta;
 EgitimCiftleriniYukleVeGoster;

eski_kume_hatasi:=0;


for sayac := 1 to iterasyon do
begin {5-iterasyon döngüsü}
 kume_hatasi:=0;
for ornek := 0 to ciftler do
 Begin {6-küme döngüsü}
  for k:=1 to katman[0] do
  Begin{6.5}
   Vektor[0].out[k]:=EgitimCiftleri.x[k,ornek]; {eğitim çiftlerinden alınan örnek}
  End; {6.5}                              {girişler doğrudan ilk katmanın çıkışına aktarılır}
   ileriYonDegerleriniHesapla;

   if (sayac=1)OR (Sayac=10) OR (Sayac=100)OR(sayac=200) OR (Sayac=300)
  OR (Sayac=400) OR (sayac=500)  OR (Sayac=600) OR (Sayac=700)OR (sayac=800)
  OR (Sayac=900)OR (Sayac=1000) OR (Sayac=10000)OR (sayac=20000) OR (Sayac=30000)
  OR (Sayac=40000) OR (sayac=50000)  OR (Sayac=60000) OR (Sayac=70000)OR (sayac=80000)
  OR (Sayac=90000) OR (Sayac=100000)OR (Sayac=110000)OR (sayac=120000) OR (Sayac=130000)
  OR (Sayac=140000) OR (sayac=150000)  OR (Sayac=160000) OR (Sayac=170000)OR (sayac=180000)
  OR (Sayac=190000) OR (sayac=200000)OR (Sayac=210000)OR (sayac=220000) OR (Sayac=230000)
  OR (Sayac=240000) OR (sayac=250000)  OR (Sayac=260000) OR (Sayac=270000)OR (sayac=280000)
  OR (Sayac=290000) OR (Sayac=300000)OR (Sayac=310000)OR (sayac=320000) OR (Sayac=330000)
  OR (Sayac=340000) OR (sayac=350000)  OR (Sayac=360000) OR (Sayac=370000)OR (sayac=380000)
  OR (Sayac=390000) OR (Sayac=400000)OR (Sayac=410000)OR (sayac=420000) OR (Sayac=430000)
  OR (Sayac=440000) OR (sayac=450000)  OR (Sayac=460000) OR (Sayac=470000)OR (sayac=480000)
  OR (Sayac=490000) OR (sayac=500000) OR (Sayac=510000)OR (sayac=520000) OR (Sayac=530000)
  OR (Sayac=540000) OR (sayac=550000)  OR (Sayac=560000) OR (Sayac=570000)OR (sayac=580000)
  OR (Sayac=90000) OR (Sayac=600000)OR (Sayac=610000)OR (sayac=620000) OR (Sayac=630000)
  OR (Sayac=640000) OR (sayac=650000)  OR (Sayac=660000) OR (Sayac=670000)OR (sayac=680000)
  OR (Sayac=690000) OR (Sayac=700000)OR (Sayac=710000)OR (sayac=720000) OR (Sayac=730000)
  OR (Sayac=740000) OR (sayac=750000)  OR (Sayac=760000) OR (Sayac=770000)OR (sayac=780000)
  OR (Sayac=790000)OR (sayac=800000) OR (Sayac=810000)OR (sayac=820000) OR (Sayac=830000)
  OR (Sayac=840000) OR (sayac=850000)  OR (Sayac=860000) OR (Sayac=870000)OR (sayac=880000)
  OR (Sayac=890000)OR (Sayac=900000)OR (Sayac=910000)OR (sayac=920000) OR (Sayac=930000)
  OR (Sayac=940000) OR (sayac=950000)  OR (Sayac=960000) OR (Sayac=970000)OR (sayac=980000)
  OR (Sayac=990000) OR  (Sayac=1000000)

  then CikislariGoster(ornek);
   {ileri yönde hesaplamalar bitti}

   {hatanın geriye yayılması işlemleri buradan başlıyor}
   CikisDeltaDegerleriniHesapla(ornek);
   GizliDeltaDegerleriniHesapla;
   ParametreleriGuncelle(eta);


   {adım büyüklüğünün güncellenmesi için önce örnek hataları bulunur, hepsi
   toplanıp küme hatası bulunur, her iterasyonda küme hatalarındaki değişime
   bakıp adım büyüklüğü buna göre uyarlanır}
   ornek_hata:=0;
   for i:=1 to katman[L+1] do
   Begin{7}
    ornek_hata:=ornek_hata+0.5*sqr(egitimciftleri.d[i,ornek]-Vektor[L+1].out[i]);
   End;{7}     {Jr=Toplam(1/2 (d-y)2}
   Kume_hatasi:=Kume_hatasi+ornek_hata;  {E=Toplam( Jr)}


 end;{6-küme döngüsü sonu}


 if (sayac=1) OR(Sayac=10) OR (Sayac=100)OR(sayac=200) OR (Sayac=300)
 OR (Sayac=400) OR (sayac=500)  OR (Sayac=600) OR (Sayac=700)OR (sayac=800)
 OR (Sayac=900)OR (Sayac=1000) OR (Sayac=10000)OR (sayac=20000) OR (Sayac=30000)
 OR (Sayac=40000) OR (sayac=50000)  OR (Sayac=60000) OR (Sayac=70000)OR (sayac=80000)
 OR (Sayac=90000) OR (Sayac=100000)OR (Sayac=110000)OR (sayac=120000) OR (Sayac=130000)
 OR (Sayac=140000) OR (sayac=150000)  OR (Sayac=160000) OR (Sayac=170000)OR (sayac=180000)
 OR (Sayac=190000) OR (sayac=200000)OR (Sayac=210000)OR (sayac=220000) OR (Sayac=230000)
 OR (Sayac=240000) OR (sayac=250000)  OR (Sayac=260000) OR (Sayac=270000)OR (sayac=280000)
 OR (Sayac=290000) OR (Sayac=300000)OR (Sayac=310000)OR (sayac=320000) OR (Sayac=330000)
```

```
   OR (Sayac=340000) OR (sayac=350000)  OR (Sayac=360000) OR (Sayac=370000)OR (sayac=380000)
   OR (Sayac=390000) OR (Sayac=400000)OR (Sayac=410000)OR (sayac=420000) OR (Sayac=430000)
   OR (Sayac=440000) OR (sayac=450000)  OR (Sayac=460000) OR (Sayac=470000)OR (sayac=480000)
   OR (Sayac=490000) OR (sayac=500000) OR (Sayac=510000)OR (sayac=520000) OR (Sayac=530000)
   OR (Sayac=540000) OR (sayac=550000)  OR (Sayac=560000) OR (Sayac=570000)OR (sayac=580000)
   OR (Sayac=90000) OR (Sayac=600000)OR (Sayac=610000)OR (sayac=620000) OR (Sayac=630000)
   OR (Sayac=640000) OR (sayac=650000)  OR (Sayac=660000) OR (Sayac=670000)OR (sayac=680000)
   OR (Sayac=690000) OR (Sayac=700000)OR (Sayac=710000)OR (sayac=720000) OR (Sayac=730000)
   OR (Sayac=740000) OR (sayac=750000)  OR (Sayac=760000) OR (Sayac=770000)OR (sayac=780000)
   OR (Sayac=790000)OR (sayac=800000) OR (Sayac=810000)OR (sayac=820000) OR (Sayac=830000)
   OR (Sayac=840000) OR (sayac=850000)  OR (Sayac=860000) OR (Sayac=870000)OR (sayac=880000)
   OR (Sayac=890000)OR (Sayac=900000)OR (Sayac=910000)OR (sayac=920000) OR (Sayac=930000)
   OR (Sayac=940000) OR (sayac=950000)  OR (Sayac=960000) OR (Sayac=970000)OR (sayac=980000)
   OR (Sayac=990000) OR  (Sayac=1000000)
   then
  begin
    Form3.ListBox4.items.add( intToStr(Sayac)+'. Küme Hatası : '+FloatToStr( Kume_hatasi));
    Form3.Listbox4.refresh;          {Küme Hatası ve Kaçıncı iterasyon için olduğu ekrana yazdırılır}

    Application.ProcessMessages;
  End;

    {Adım Büyüklüğü Uyarlaması}
    if   Kume_hatasi-Eski_kume_hatasi<0 then eta:=eta+Gamma
    Else if  Kume_hatasi-Eski_kume_hatasi>0  then eta:=(1-Beta)*eta;
    {Küme Hatası Güncellenir}
     Eski_kume_hatasi:= Kume_hatasi;


    {Ağ parametrelerinin çıkışta kaydedilmesi}
    if (sayac<iterasyon) AND (Kume_hatasi<=Tolerans)Then {Tüm iterasyonlar bitmeden hata toleransın altına
düşerse kaydet ve programdan çık}
     Begin{8}
      EgitilmisAgParametreleriniKaydet(Sayac);
      EgitilmisAgParametreleriniYSADenetleyiciicinKaydet;
     End;{8}
   if sayac = iterasyon then
   Begin
    EgitilmisAgParametreleriniKaydet(Sayac);
    EgitilmisAgParametreleriniYSADenetleyiciicinKaydet;
   End;
   Application.ProcessMessages;
   resim1.canvas.refresh;
 end; {5-iterasyon döngüsü sonu}

Basla := True;
end;

end; {Main}



procedure TForm1.KapatClick(Sender: TObject);
begin
resim1.FreeImage;
resim2.FreeImage;
resim3.FreeImage;
Close;
end;

end.



unit Form2EgitimCifleri_Unit9;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm2 = class(TForm)
    ListBox1: TListBox;
    ListBox2: TListBox;
    Button1: TButton;
```

```
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label11: TLabel;
    Label13: TLabel;
    ListBox3: TListBox;
    Label14: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

procedure TForm2.Button1Click(Sender: TObject);
begin
 Close;
end;

end.



unit Form3YSAEgitimGrafigi_Unit9;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm3 = class(TForm)
    Button1: TButton;
    ListBox4: TListBox;
    Label16: TLabel;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure FormClick(Sender: TObject);
    procedure Button2Click(Sender: TObject);


  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;

implementation

uses Form1NeuroDeneme_Unit9;

{$R *.DFM}

procedure TForm3.Button1Click(Sender: TObject);  {Sadece istenen değeri çizer}
var
 x,y,zaman,Derinlik:integer;
 Point:PByteArray;

 resim:TBitmap;
begin
  Resim:=TBitmap.Create;
 resim.Height:=260;
 resim.width:=420;
 resim.PixelFormat:=pf8bit;
```

```
 for x:=0 to 24 do
 Begin
  resim.Canvas.TextOut(32+15*x,130,'+');
 End;

 for x:=0 to 12 do
 Begin
  resim.canvas.textout(35+30*x,141,inttostr(30*x));
 End;
 for y:=0 to 10 do
 Begin
  resim.canvas.textout(20,40+18*y,'+');
 End;

 for y:=0 to 10 do
 Begin
   resim.canvas.textout(0,40+18*y,FloatToStr(1-0.2*y));
 End;
 resim.Canvas.Font.color:=clblue;
 resim.canvas.textout(125,5,'... İstenen Çıkış');
 resim.Canvas.Font.color:=0;
 resim1.canvas.textout(0,5,'Derinlik (+1,-1)');
 resim1.canvas.textout(360,120,'Açı(Derece)');
 for zaman:=0 to 360 do
 Begin
 Derinlik:=round(260-124-90*sin(pi/180*zaman));
 point:=resim.Scanline[Derinlik];
 point[35+zaman]:=4;
 if zaman/4=Trunc(zaman/4) then Form3.Canvas.Draw(3,0,Resim);

 End;

end;


procedure TForm3.FormClick(Sender: TObject);
begin
 Form3.Canvas.draw(3,0,Resim1);
 Form3.Canvas.draw(3,270,Resim2);
 Form3.Canvas.Draw(430,80,Resim3);
end;

{mevcut işlev çıkışı çizdirilirken istenen işlev (set) değerini de üstüne çizer}
procedure TForm3.Button2Click(Sender: TObject);
var
 zaman,Derinlik:integer;
  Point:PByteArray;
begin
 resim1.Canvas.Font.color:=clblue;
 resim1.canvas.textout(123,5,'... İstenen Çıkış');


 for zaman:=0 to 360 do
 Begin
  Derinlik:=read (PWM);
  point:=resim1.Scanline[Derinlik];
  point[35+zaman]:=4;
  if zaman/4=Trunc(zaman/4) then Form3.Canvas.Draw(3,0,Resim1);
 End;
 resim1.Canvas.Font.color:=0;
 resim1.canvas.textout(0,5,'Derinlik (0,-2)');
 resim1.canvas.textout(360,120,'Zaman(sn)');
end;

end.
```

YSA DENETİM ALGORİTMASI



YSA Denetleyici Programı

```
program YSADenetleyici_project;

uses
  Forms,
  Form1YSADenetleyici_unit1 in 'Form1YSADenetleyici_unit1.pas' {Form1},
  Form2YSADenetleyiciAgirliklar_Unit2 in 'Form2YSADenetleyiciAgirliklar_Unit2.pas' {Form2},
  Form3YSADenetleyiciGrafik_Unit3 in 'Form3YSADenetleyiciGrafik_Unit3.pas' {Form3};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm3, Form3);
  Application.Run;
end.



unit Form1YSADenetleyici_unit1;

interface

uses
  Windows,Math, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
    Tagirliklar=Record
     w: array [1..2,1..2] of Real;
     end;
     TVektor=Record
     out: array [1..2] of Real; {çıkış değerleri}
     delta: array [1..2] of Real; {Delta Değerleri}
     thr: array [1..2] of Real; {Eşik Değerleri}
     SumJ: array [1..2] of Real;{ilgili Nöronun Ağırlıklı Toplamı}
    end;

  TForm1 = class(TForm)
    Button1: TButton;
    OpenDialog1: TOpenDialog;
    Button2: TButton;
    Image1: TImage;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);


     private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Resim1:Tbitmap;
  L,zaman:integer;
  katman:array [0..3] of Integer;
 W:array  [0..3] of Tagirliklar;  {Ağırlıklar kayıdı}
 Vektor: Array [0..3] of TVektor;

implementation

uses Form2YSADenetleyiciAgirliklar_Unit2, Form3YSADenetleyiciGrafik_Unit3;

{$R *.DFM}


Function Nonlineerislev(toplam,esik:Real):Real;
Begin
 Nonlineerislev:=Tanh((toplam-esik)*pi/180);
End;
```

```
Procedure ileriYonDegerleriniHesapla;
Var
 i,j,k:integer;
 Toplam:Real;

Begin
 for k:=0 to L do
 Begin
 For i:=1 to katman[k+1] do
  Begin
  Toplam:=0;
   for j:=1 to katman[k] do
   Begin
    Toplam:=Toplam+W[k].w[i,j]*Vektor[k].out[j];

   End;
   Vektor[k+1].Sumj[i]:=Toplam;
  if k<L then Vektor[k+1].out[i]:=Nonlineerislev(Vektor[k+1].Sumj[i],Vektor[k+1].thr[i])
  Else Vektor[k+1].out[i]:=Toplam-Vektor[k+1].thr[i];
  End;
 End;
End;


Procedure CikislariGoster(zamancikis:integer);

Var
  j :integer;
  x1,y1,derinlik1:integer;
  Point1 :PByteArray;

Begin    { Main}

Form3.ListBox1.items.add('  '+intToStr(zamancikis)+'.Girişe ait');
  for j:=1 to katman[0] do
  Begin {I.2}
  Form3.ListBox1.items.add('    ' +intToStr(j)+'. Giriş : '+FloatToStr(zaman));
  Form3.Listbox1.refresh;
  End; {I.2 son}

   Form3.ListBox2.items.add('  '+intToStr(zamancikis)+'.Girişe ait');
  for j:=1 to katman[L+1] do
  Begin {I.2}
  Form3.ListBox2.items.add('    ' +intToStr(j)+'. Çıkış : '+FloatToStr(Vektor[L+1].out[j]));
  Form3.Listbox2.refresh;
  End; {I.2 son}
  Application.ProcessMessages;


  {Çıkış Grafiğini Çizdir}
   begin {III.1}
  if zamancikis=0 then
  Begin {III.2}
     Resim1.Height:=260;
     Resim1.width:=420;
     Resim1.PixelFormat:=pf8bit;
     for x1:=0 to 24 do
     Begin {III.3}
      Resim1.Canvas.TextOut(32+15*x1,130,'+');
     End; {III.3 son}

    for x1:=0 to 12 do
    Begin {III.4}
     Resim1.canvas.textout(35+30*x1,141,inttostr(30*x1));
    End; {III.4 son}
    for y1:=0 to 10 do
    Begin {III.5}
     Resim1.canvas.textout(20,40+18*y1,'+');
    End;  {III.5 son}

    for y1:=0 to 10 do
    Begin  {III.6}
     Resim1.canvas.textout(0,40+18*y1,FloatToStr(1-0.2*y1));
    End; {III.6 son}

     Resim1.canvas.textout(0,5,'Derinlik (+1,-1)');
```

```
    Resim1.canvas.textout(360,120,'Açı(Derece)');

 End;{III.2 son}


 Derinlik1:=round(260-124-90*(Vektor[L+1].out[1]));
 Point1:=Resim1.Scanline[Derinlik1];
 Point1[35+zamancikis]:=0;
 Form3.Canvas.Draw(3,270,Resim1);

end; {III.1 son}

End; {Main Son}



procedure TForm1.Button1Click(Sender: TObject);
type
 Kayit=String[100];

 Var
   i,j,k,Kod,Matris,sayac:integer;

 Dosya : File of Kayit;
 s : kayit;


begin  {a}
Form2.show;
Form3.show;

OpenDialog1.Execute;
Resim1:=TBitmap.Create;
AssignFile(Dosya,form1.OpenDialog1.FileName);
{$i-} Reset(Dosya);Kod:=IOResult;{$I+};
if Kod<>0 Then
Begin {b}
if Application.MessageBox(
       'Dosya Yok_Programdan çıkalımmı',

       'Open Error',
       MB_OKCANCEL + MB_DEFBUTTON1) <> IDOK then    Halt;

End; {bson}

sayac:=0;
Seek(Dosya,sayac);
   read(Dosya,s);
   L := StrToInt(s);          {Katman Sayısını}
   Form2.ListBox1.Items.Add('Katman Sayısı (L)= '+IntToStr(L));

   sayac:=sayac+1;
Seek(Dosya,sayac);
    read(Dosya,s);
    katman[0]  := StrToInt(s);    {Giriş Vektörü Boyutunu}
    Form2.ListBox1.Items.Add('Giriş (0) Katmanı Nöron Sayısı= '+IntToStr(katman[0]));

    sayac:=sayac+1;
Seek(Dosya,sayac);
    read(Dosya,s);
    katman[L+1]  := StrToInt(s);    {Çıkış Vektörü boyutunu}
    Form2.ListBox1.Items.Add('Çıkış ('+IntToStr(L+1)+ ') Katmanı Nöron Sayısı= '+IntToStr(katman[L+1]));

 For i:=1 to L do
 Begin {3}
 sayac:=sayac+1;
  Seek(Dosya,sayac);
    read(Dosya,s);
    katman[i]  := StrToInt(s);  {i. Gizli Katmandaki Nöron Sayısını gösterir}
    Form2.ListBox1.Items.Add(IntToStr(i)+'.Katman Nöron Sayısı= '+IntToStr(katman[i]));
 End;{3}


  For Matris:=0 to L do
  Begin{4}
   for i:=1 to katman[Matris+1] do
   Begin{5}
```

```
    for j:=1 to katman[matris] do
    Begin {6}
    sayac:=sayac+1;
     Seek(Dosya,sayac);
     read(Dosya,s);
     W[Matris].w[i,j] := StrToFloat(s);
     Form2.ListBox2.Items.Add('W['+IntToStr(Matris)+'].w['+IntToStr(i)+IntToStr(j)+']= '+s);
    End; {6}

    sayac:=sayac+1;
    Seek(Dosya,sayac);
    read(Dosya,s);
    Vektor[Matris+1].thr[i] := StrToFloat(s);
    Form2.ListBox2.Items.Add('Vektor['+IntToStr(Matris+1)+'].thr['+IntToStr(i)+']= '+s);
    Form2.ListBox2.Items.Add(' ');
   End; {5}

  End;{4}

    CloseFile(Dosya);
 {Eğitilmiş Ağırlık Değerlerinin Atanması Bitti}


 {girişlerin ağa girilmesi ileri yönde çıkışların hesaplanması}
 for zaman:=0 to 360 do
 Begin   {giriş}
    for k:=1 to katman[0] do
    Begin{6.5}
     Vektor[0].out[k]:=zaman/360; {girişler doğrudan ilk katmanın çıkışına aktarılır}
    End; {6.5}           {Buradaki 360'a bölme işlemi, girişi 0 ile 1 arasına normalize etmek için}
     ileriYonDegerleriniHesapla;
     CikislariGoster(zaman);
 End; {giriş}


end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 resim1.FreeImage;
 Close;
end;

end.




unit Form2YSADenetleyiciAgirliklar_Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm2 = class(TForm)
    ListBox1: TListBox;
    ListBox2: TListBox;
    Label1: TLabel;
    Label2: TLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

end.
```

```
unit Form3YSADenetleyiciGrafik_Unit3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm3 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label4: TLabel;
    Label11: TLabel;
    ListBox1: TListBox;
    ListBox2: TListBox;
    Label3: TLabel;
    procedure FormClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;

implementation
uses Form1YSADenetleyici_unit1;

{$R *.DFM}

procedure TForm3.FormClick(Sender: TObject);
begin
 Form3.Canvas.draw(3,270,Resim1);
end;

end.
```

6. Sualtı Aracının Modelleme Çalışmaları

Gelişme raporunda sualtı araçları için dinamik denklemler ve koordinat sistemleri ve dönüşümleri verilmişti [13]. Projenin bu kısımda su altı aracının hareket ve manevralarının benzetimi amacıyla Simulink ortamında derinlik, yalpa ve yönelme gibi hareketler için ayrı ayrı modeli belirlenmiştir [21,22].

6.1. HAREKET DENKLEMLERİ

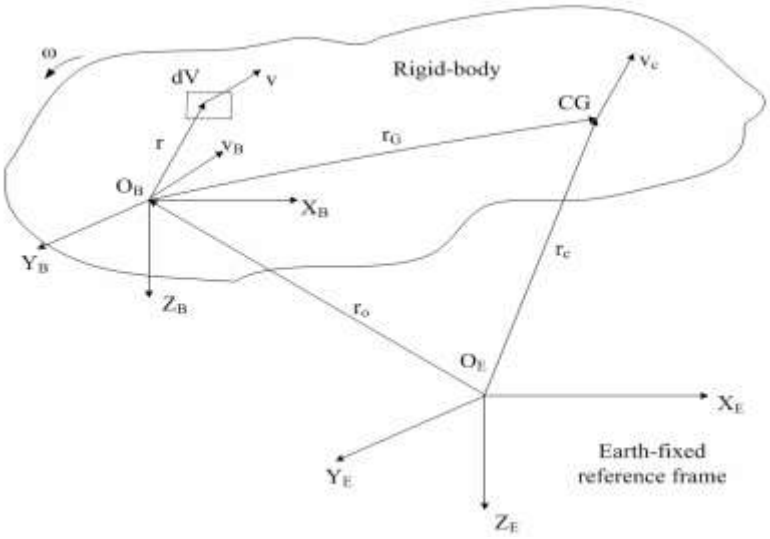
Sualtı araçlarını katı cisim olarak kabul edersek, Newton'un ikinci yasasına dayanarak hareket denklemlerini yazabiliriz.

Öteleme Hareketi

Bir deniz altı aracının öteleme hareketini,

$$p_c = f_c \qquad p_c = mv_c$$

Burada p_c aracın ağırlık merkezini referans alınarak doğrusal momentum, p_c zaman türevi v_c olup , ağırlık merkezinin hızı olarak adlandırılır. f_c harici güç, m aracın ağırlığı m ise katı gövdenin ağırlığıdır.



Eylemsizlik koordinat sistemine göre dönüştürülemeyen referans çerçevesi ve Araç-gövde koordinat sistemine göre dönüştürülebilen referans çerçevesi

O_B Araç-gövde koordinat sisteminin merkezi, O_E ise eylemsizlik koordinat sisteminin merkezidir. CG ise Rigit(katı) gövdenin ağırlık merkezini temsil eder. r, r_o, r_G ve r_c konum vektörlerini, v, v_B ve v_c vız vektörlerini belirtir. Son olarak ω ise rigit(katı) gövdenin eylemsizlik koordinat sistemindeki açısal hızını ifade eder. Yukardaki referans çerçevelerini kullanarak, Bu aşamadan sonra E eylemsizlik koordinat sistemindeki vektörleri, B ise araç-gövde koordinat sistemindeki vektörleri temsil edecektir. Formülleri oluşturalım.

$$E_{r_c} = E_{r_o} + E_{r_G}$$

Ağırlık merkezinin hızını ise;

$$E_{V_C} = E_{r'_C} = E_{r'_o} + E_{r'_G}$$

$X_E Y_E Z_E$ ve $X_B Y_B Z_B$ keyfi bir vektör olan c zaman vektörünün türevi olarak ilişkilendirirsek;

$$E_{c'} = B_{c'} + E_\omega \times E_c$$

$B_{r_G} = 0$ ve $E_{r'_o} = E_{v_o}$ (katı gövde için) olarak alırsak;

$$E_{r'_G} = B_{r'_G} + E_\omega \times E_{r_G} = E_\omega \times E_{r_G}$$

Böylece,

$$\begin{aligned} B_{r'_G} &= 0 \\ E_{r'_G} &= E_\omega \times E_{r_G} \end{aligned} \quad \text{ise} \quad E_{v_c} = E_{v_o} + E_\omega \times E_{r_G}$$

İvme vektörü olarak bulunabilir.

$$E_{v'_c} = E_{v'_o} + E_{\omega'} \times E_{r_G} + E_\omega \times E_{r'_G}$$

Ya da daha kullanışlı olarak;

$$E_{v'_c} = B_{v'_o} + E_\omega \times E_{v_o} + E_{\omega'} \times E_{r_G} + E_\omega \times (E_\omega \times E_{r_G})$$

Kullanılır. Denklemleri en son hali olarak Eylemsizlik koordinat sistemiyle yazarsak(6.10);

$$m(B_{v'_o} + E_\omega \times E_{v_o} + E_{\omega'} \times E_{r_G} + E_\omega \times (E_\omega \times E_{r_G})) = E_{f_o}$$

(6.18)Araç-gövde koordinat sistemine göre yazmak istersek;

$$m(B_{v'_o} + B_\omega \times B_{v_o} + B_{\omega'} \times B_{r_G} + B_\omega \times (B_\omega \times B_{r_G})) = B_{f_o}$$

Yukarıdaki oluşturmuş olduğumuz denklemler öteleme hareket denklemleridir.

Dönme Hareketi

Bir denizaltı aracının dönme hareketini;

$h_c = I_c \omega$, $h_c = m_c$

Burada h_c açısal momentum, m_c aracın merkezini referans alarak oluşturulan yerçekimi momenti, ω açısal hız vektörü ve I_c ise aracın merkezini referans alarak oluşturulan tensördür. Bulunan O_B vektörünü mutlak açısal momentum olarak adlandırmıştık. Bu bölümde ise;

$$B_{h_O} = \int_V B_r \times B_V \rho dV$$

ρ rigit(katı) gövdenin yoğunluğunu ifade eder. Eylemsizlik koordinat sistemine uygun yazmak istersek eğer;

$$E_{h_O'} = \int_V E_r \times E_{v'} \rho dV + \int_V E_{r'} \times E_v \rho dV$$

m_o sağ tarafın ilk moment vektörü ise ;

$$E_{m_O} = \int_V E_r \times E_{v'} \rho dV$$

olarak bulunur.

$$E_v = E_{r_O'} + E_{r'} \qquad E_{r'} = E_v - E_{v_O}$$

yazılabilir .

$(v \times v = 0)$ olarak kullanırsak;

$$E_{h'_o} = E_{m_o} - E_{v_o} \times \int_V E_v \rho dV$$

ya da;

$$E_{h'_o} = E_{m_o} - E_{v_o} \times \int_V (E_{v_o} + E_{r'}) \rho dV = E_{m_o} - E_{v_o} \times \int_V E_{r'} \rho dV$$

Aracın sabit kütlesi ile , O_B araç-gövde koordinat sisteminin orjinine olan uzaklığı ile tanımlamak istersek;

$$B_{r_G} = \frac{1}{m} \int_V B_r \rho dV$$

Eylemsizlik koordinatın;

$$mE_{r'_G} = \int_V E_{r'} \rho dV$$

$E_{r'_G} = E_w \times E_{r_G}$ eşitliğinden yararlanarak;

$$\int_V E_{r'} \rho dV = mE_w \times E_{r_G}$$

şeklinde yazabiliriz. Sonuç olarak ;

$$E_{h'_o} = E_{m_o} - mE_{v_o} \times (E_{w_B} \times E_{r_G})$$

İfadeyi genelleştirebiliriz. sağ tarafın ilk moment vektörünü kullanacak olursak;

$$\int_V E_r \times E_{v_o} \rho dV = (\int_V E_r \rho dV) \times E_{v_o} = mE_{r_G} \times E_{v_o}$$

Bu şekilde formülize edilir.

I_o aracın araç-gövde koordinat sistemin merkezinde oluşan atalet sensörü olsun.

$$I_o \omega = \int_V r \times (\omega \times r) \rho dV$$

Denklemimizi tekrar yazarsak;

$$E_{h_o} = E_{I_o} E_\omega + mE_{r_G} \times E_{v_o}$$

olarak elde etmiş oluruz. I_o eğer zamana göre sabit olarak varsayılır ise;

$$E_{h_o}^{'} = E_{I_o} E_{\omega}^{'} + E_\omega \times (E_{I_o} E_\omega) + m(E_\omega \times E_{r_G}) \times E_{v_o} + mE_{r_G} \times (B_{v_o}^{'} + E_\omega \times E_{v_o})$$

I_o'e göre formülü elde ederiz. $(\omega \times r_G) \times v_o = -v_o \times (\omega \times r_G)$ arasındaki bağlantıyı kullanarak E_{h_o} ;

$$E_{I_o} E_{\omega}^{'} + E_\omega \times (E_{I_o} E_\omega) + mE_{r_G} \times (B_{v_o}^{'} + E_\omega \times E_{v_o}) = E_{m_o}$$

bulunur . Araç-gövde koordinat sistemine göre yazmak istenirse;

$$B_{I_o} B_{\omega}^{'} + B_\omega \times (B_{I_o} B_\omega) + mB_{r_G} \times (B_{v_o}^{'} + B_\omega \times B_{v_o}) = B_{m_o}$$

Böylelikle dönme hareket denklemlerini yazmış olduk.

$$f_o = \tau_1 = \begin{bmatrix} X & Y & Z \end{bmatrix}$$

$$m_o = \tau_2 = \begin{bmatrix} K & M & N \end{bmatrix}$$

$$\omega_o = v_1 = \begin{bmatrix} u & v & w \end{bmatrix}$$

$$w = v_2 = \begin{bmatrix} p & q & r \end{bmatrix}$$

$$r_G = \begin{bmatrix} x_G & y_G & z_G \end{bmatrix}$$

Burada r_G araç-gövde koordinat sistemine göre aracın ağırlık merkezini belirtir. Araç-gövde koordinat sisteminde, altı dereceli serbest denklemlerden doğrusal olmayan dinamik denklemleri;

$$m(v_1^{'} + v_2 \times v_1 + v_2^{'} \times r_G + v_2 \times (v_2 \times r_G)) = \tau_1$$

$$I_o v_2 + v_2^{'} \times (I_o v_2) + m r_G \times (v_1^{'} + v_2 \times v_1) = \tau_2$$

Burada m aracın toplam kütlesi, I_o atalet tensörü ise;

$$I_o = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}, \qquad I_o = I_o^T > 0$$

Matris (6.39)'da I_{xx}, I_{yy}, I_{zz} X_B, Y_B, Z_B düzlemi için eylemsizlik momenti, aynı zamanda aralarındaki bağlantı ;

$$I_{xy} = I_{yx} \ , \ I_{yz} = I_{zy} \ , \ I_{xz} = I_{zx}$$

şeklindedir.

$$I_{xx} = \int_V (y^2 + z^2)\rho dV \quad ; \quad I_{xy} = \int_V xy\rho dV$$

$$I_{yy} = \int_V (x^2 + z^2)\rho dV \quad ; \quad I_{xz} = \int_V xz\rho dV$$

$$I_{zz} = \int_V (x^2 + y^2)\rho dV \quad ; \quad I_{yz} = \int_V yz\rho dV$$

Eylemsizlik momentlerini yukarıdaki formüllerden bulabiliriz. (Vücut kütle yoğunluğu: ρ)

Gerekli tanımlamaları yaptıktan sonra AUV için genel hareket denklemlerini yazabiliriz. Aşağıda oluşturulan ilk üç denklem AUV'un öteleme denklemleri, son üç denklem ise dönüş denklemleridir.

$$m[u' - vr + wq - x_G(q^2 + r^2) + y_G(pq - r') + z_G(pr + q')] = X$$
$$m[v' - wp + ur - y_G(r^2 + p^2) + z_G(qr - p') + x_G(pq + r')] = Y$$
$$m[w' - uq + vp - z_G(p^2 + q^2) + x_G(rp - q') + y_G(rq + p')] = Z$$
$$I_{xx}p' + (I_{zz} - I_{yy})qr - (r' + pq)I_{xz} + (r^2 - q^2)I_{yz} + (pr - q')I_{xy} + m[y_G(w' - uq + vp) - z_G(v' - wp + ur)] = K$$
$$I_{yy}q' + (I_{xx} - I_{zz})rp - (p' + qr)I_{xy} + (p^2 - r^2)I_{zx} + (qp - r')I_{xy} + m[z_G(u' - vr + wq) - x_G(w' - uq + vp)] = M$$

$$I_{zz}r' + (I_{yy} - I_{xx})pq - (q' + rp)I_{yz} + (q^2 - p^2)I_{xy} + (rq - p')I_{zx} + m[x_G(v' - wp + ur) - y_G(u' - vr + wq)] = N$$

Denklemler daha kompakt halde toplanabilir:

$$M_{RB}v' + C_{RB}(v)v = \tau$$

Burada doğrusal açı ve hız için kullanılan vektör $v = [u \quad v \quad w \quad p \quad q \quad r]^T$,

Dış kuvvetler ve momentleri ise $\tau = [X \quad Y \quad Z \quad K \quad M \quad N]^T$ ifade eder.

Ataletsel Matris: M_{RB}

Coriolis ve merkezcil Matris: (Coriolis kuvveti: dönen bir platformun merkezinden karşı tarafına yürümeye çalışan biri tarafından anlaşılabilir. Yürümek istediği tarafa doğru dik açıyla itildiğini görür. Benzer şekilde, dönen yer kürenin yüzeyi üzerinde hareket eden hava, kuzey yarım kürede hareket yönünün sağına, güney yarım kürede soluna saptırır. Bu saptırma gücüne *coriolis kuvveti* denir.) $C_{RB}(v)$ Temelde denklemleri bu şekilde kabul edeceğiz.

AUV için model oluştururken, AUV'un hareket denklemlerini katsayıları veya etki eden faktörleri tespit etmemiz gereklidir.

Ataletsel Matris; M_{RB}

Ataletsel Matris, tek ve simetrik bir matris olup değeri pozitiftir.($M_{RB} \in R^{6\times 6}$)

$$M_{RB} = \begin{bmatrix} mI_{3\times 3} & -mS(r_G) \\ mS(r_G) & I_o \end{bmatrix}$$

$$= \begin{bmatrix} m & 0 & 0 & 0 & mz_G & -my_G \\ 0 & m & 0 & -mz_G & 0 & mx_G \\ 0 & 0 & m & my_G & -mx_G & 0 \\ 0 & -mz_G & my_G & I_{xx} & -I_{xy} & -I_{xz} \\ mz_G & 0 & -mx_G & -I_{yx} & I_{yy} & -I_{yz} \\ -my_G & mx_G & 0 & -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

$I_{3\times 3}$:3x3'lük birim matrisi temsil eder.

$S(.)$:3x3'lük çarpık-simetri matrisini temsil eder. Örnek verecek olursak;

$$S(r_G) = -S^T(r_G) = \begin{bmatrix} 0 & -z_G & y_G \\ z_G & 0 & -x_G \\ -y_G & x_G & 0 \end{bmatrix}.$$

-Coriolis ve merkezcil Matrisi:

$$C_{RB}(v) = \begin{bmatrix} 0_{3\times 3} & -mS(v_1) - mS(v_2)S(r_G) \\ -mS(v_1) - mS(r_G)S(v_2) & -S(I_o v_2) \end{bmatrix} \quad C_{RB} = -C_{RB}{}^T$$

Dış Kuvvetler ve Momentler

Hareket denklemleri , dış kuvvetler ve momentler olarak 5 tane bileşenden oluşmaktadır.

$$\tau_{RB} = \tau_{hydrostatic} + \tau_{addedmass} + \tau_{drag} + \tau_{lift} + \tau_{control}$$

Hidrostatik Mukavement(Yerçekimi ve Kaldırma kuvveti dahil): $\tau_{hydrostatic}$

Araç üzerinde hidrodinamik kuvvet ve moment üçlü bileşenler: $\tau_{addedmass}, \tau_{drag}, \tau_{lift}$

Kitle tarafından oluşturulan kuvvet: $\tau_{addedmass}$

Araç üzerinde akışkan madde tarafından uygulanan kuvvet : τ_{drag}

Araca hareket halindeyken yaptığı açıya göre uygulanan kuvvet: τ_{lift}

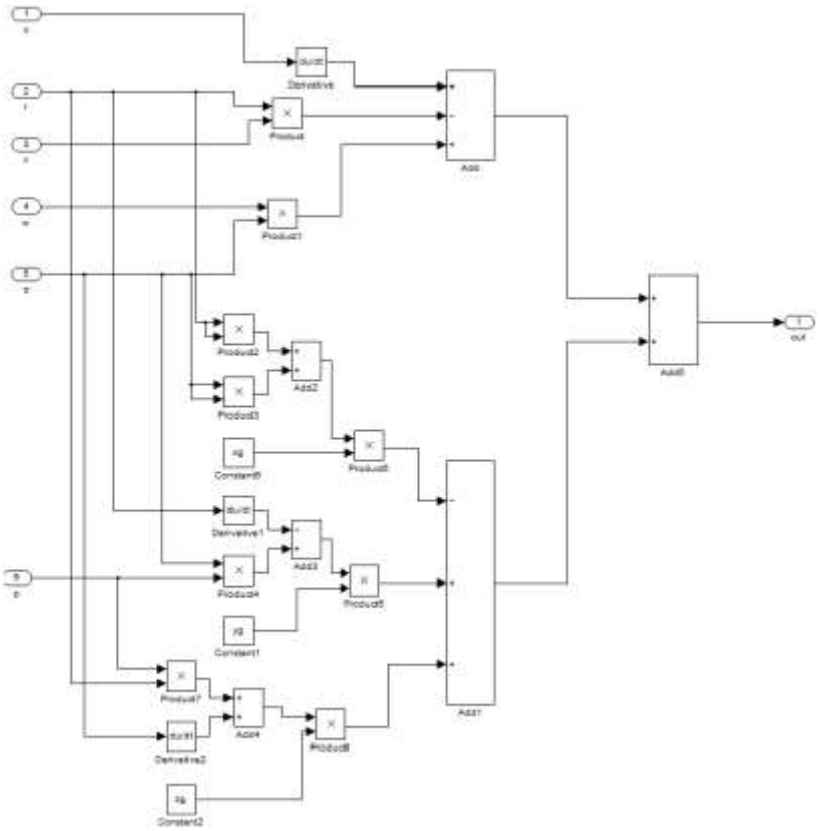
Aracı sürekli hareket halinde tutmak için iticilerde oluşturulan kuvvet: $\tau_{control}$

6.2. MATLAB İLE AUV SİMULASYONU

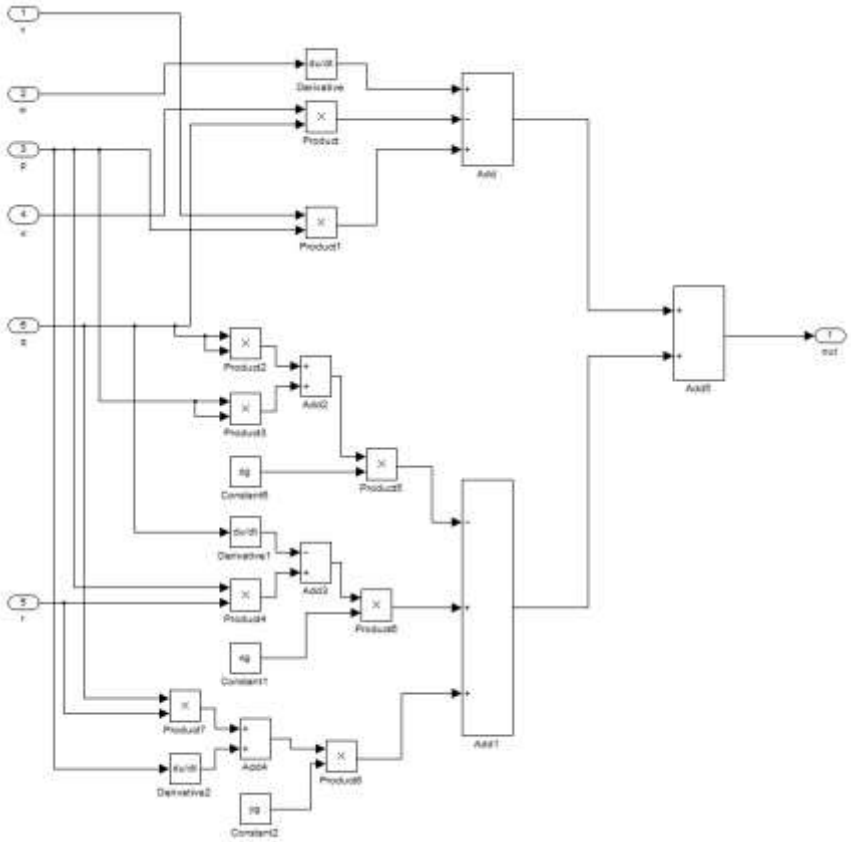
Su altı robotunun çalışma sistemi şu şekildedir. Yanlardaki iki itici (T1 ve T2) robotun ileri geri hareketini ve sağa sola dönüşünü sağlarlar.

Blok Diyagramları

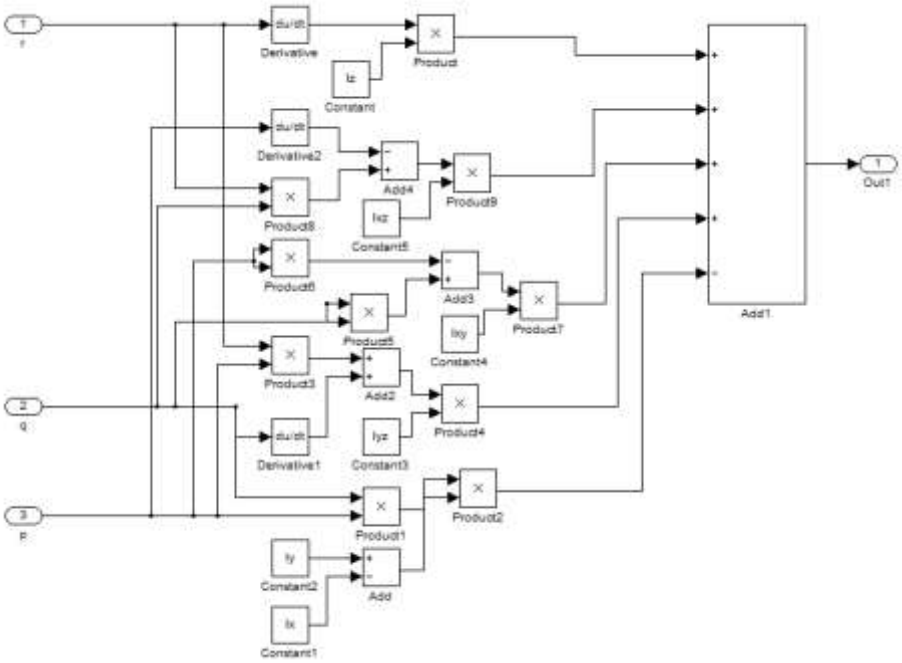
Öncelikle kullanılan denklemlerin oluşturulması gerekmektedir.



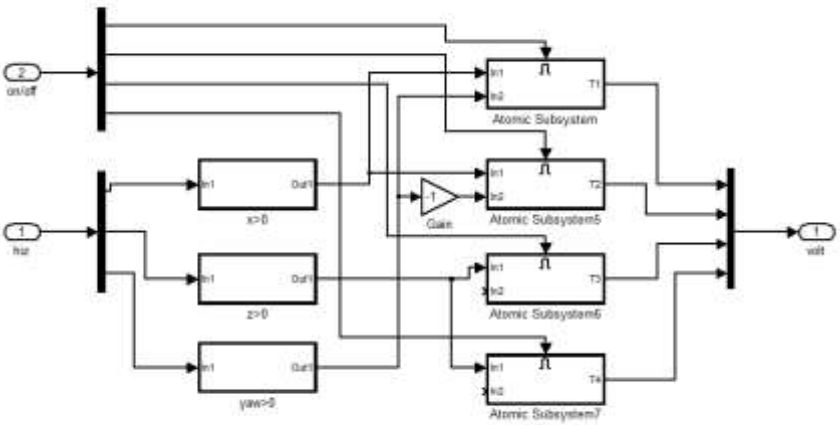
Matematiksel ileri yön hız denklemi



Matematiksel derinlik denklemi

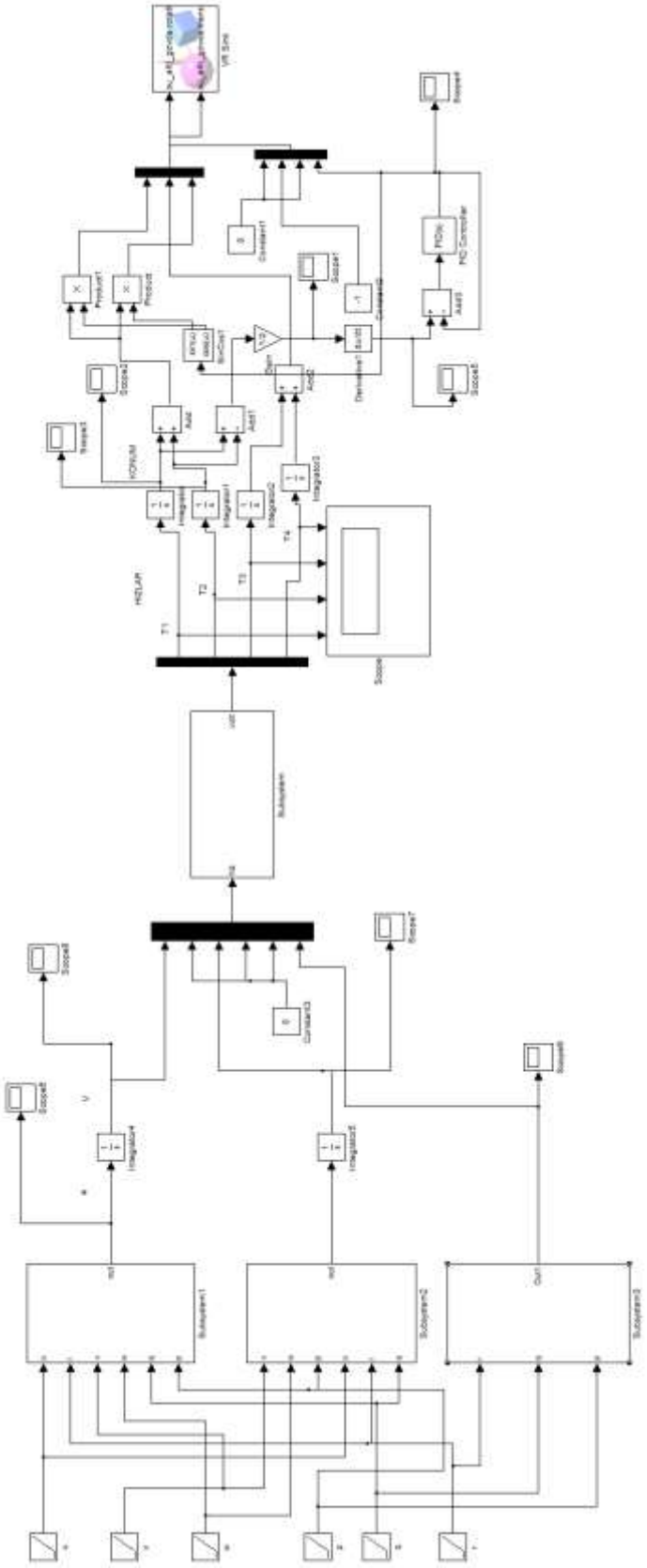


Matematiksel Dümen Denklemi



İstenilen hız ve açı referans değerleri girildiğinde iticilere gerilimi veren blok diyagramları

x>0 ise T1 ve T2 iticileri verilen değere göre çalışıyor. z>0 ise T3 ve T4 iticileri verilen değere göre çalışıyor. yaw açısının değeri varsa T1 iticisinin hızıyla T2 iticisinin hızlar arasındaki farktan açı değeri hesaplanabiliyor.



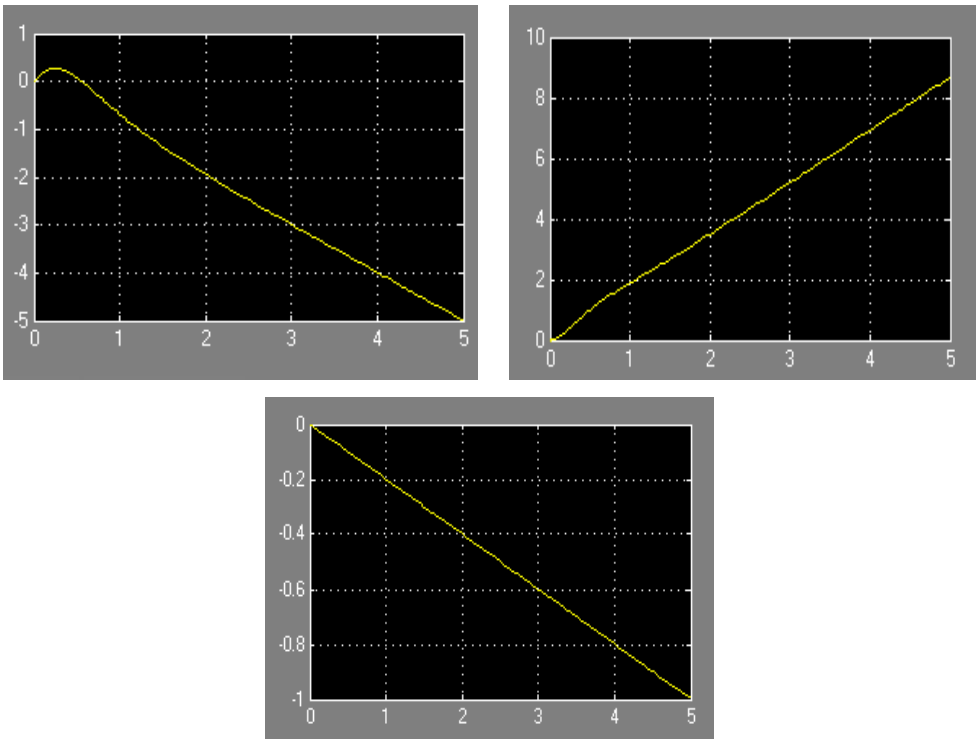
Şekil 21. Genel AUV Modelli

Simülasyonun Uygulanması

Şekil.22.'de blok diyagramına giren referans x, z eksenlerindeki öteleme hız değerleriyle birlikte, z eksenindeki dönme hareketi yapan, yaw açısının değişmesi için verilen açı değerinin karşılığında iticilerden çıkan hız değerlerini göstermektedir.

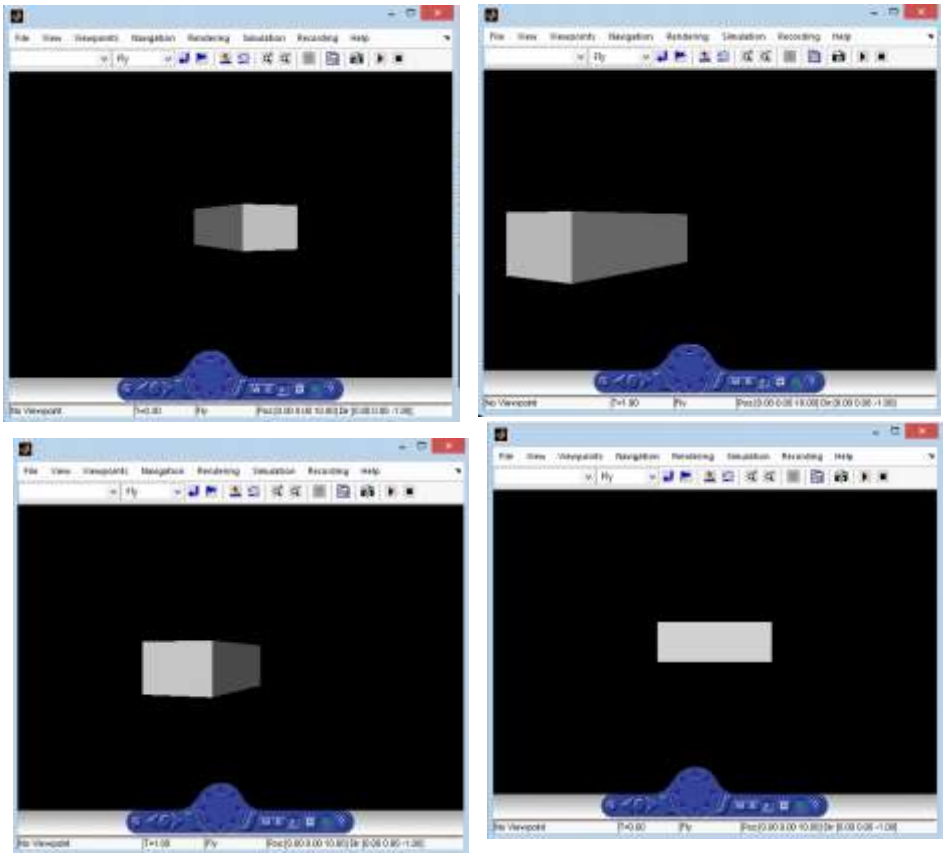
Hızlar bulunduktan sonra integralleri alınıp konumlar elde edilmiştir. x eksenindeki hareket T1ve T2 iticilerinin ötelediği kadardır, yönü de verilen açının cosinüsüne eşitttir.
y eksenindeki hareket, T1 ve T2 iticilerinin toplamları alınıp gelen açının sinüs değeri alınarak elde edilir.
Su altı robotunun dönme açısı, T1 ve T2 iticilerinin arasındaki fark alınarak elde edilir
z eksenindeki hareketini de xyatay eksene dik olarak yerleştirilen T3 ve T4 iticileri sağlar.

Şekil 22'deki örnekte 1 m/s hızla, 0 konumundan 2π/3 derece dönerek z ekseninde de -0.2 m/s hızda hafif alçalarak hareket etmesi isteniyor. Bunun sonucunda elde edilen grafikler, sırasıyla x, y ve z Şekil 23'deki gibidir.



Şekil 22. .x,y ve z eksenlerindeki hareketleri

Simülasyon sonucunda oluşan vr bloğu ise Şekil 24'deki gibidir.



Şekil 23. VR Bloğu

Araç, ilk önce sıfır konumundan başlayıp 120 derece döndükten sonra hareketine devam edip ilerlemeye başlamıştır.

6.3. Proje Grubu

15 kişilik proje gurubunun eşgüdümlü çalışabilmesi ve paylaşımların hızlı olması için periyodik toplantıların yanısıra AUVProject adında bir kapalı gurup oluşturularak sosyal paylaşım ağlarından yararlanılmıştır.

7. Deneysel Bulgular

İlk testlerde şu sorunlar karşımıza çıktı ve giderildi:

Konfigürasyonumuzda uygulama konusundaki bilgi eksikliğimizden dolayı dikey motorları aynı yönde sürüyorduk ve bu aracın bir yöne yalpa yaparak yatmasına neden oluyorlardı. Yazılımda motorlardan birine giden yön işareti terslenerek denge sağlandı. Sızdırma problemleri çözüldü. Bilgisayar- araç haberleşmesindeki sorunlar çözüldü.

Araç kontrol kartı uzaktan resetlenebilecek şekilde yeniden düzenlendi. Aracın dikey hareketlerde bozulan dengesi, motorların konumunda kaydırma yapılarak ve araca yüzdürücüler ve kurşun plakalar eklenerek düzeltildi.

Şekil.24. Test Uygulamaları

Derinlik kontrol uygulaması için şeffaf pleksiglass 80 cm çapında 2m boyunda silindir bir boru alındı. PVC bir taban üzerine aynı çapta yuva açtırılarak üzerine yerleştirildi ve yapıştırıldı. Bozucu etki üretmesi için santrifüj pompa alındı.

Şekil.24. Test Uygulamaları (Devam)

Şekil.24. Test Uygulamaları (Devam)

Şekil.24. Test Uygulamaları (Devam)

8. Sonuç ve Öneriler

Projede derinlik ve yön kontrolü uygulamaları için üzerinde yazılım geliştirilebilecek bir deney platformu tasarlanmış ve gerçeklenmiştir. Platformun, mekanik, elektronik ve yazılım kısımları tamamlanmıştır. Yazılımı gerek bilgisayardan el ile gerekse basınç ve elektronik pusuladan veri okuyup geri bildirime göre motorları sürebilmektedir. Kontrol algoritmaları yazılıma eklenmek üzere hazırlanmıştır. Ancak vakit kısıtlamasından dolayı PID ve Yapay Sinir Ağları ile denetim karşılaştırmaları yapılamamıştır. Önümüzdeki çalışmalarda deney platformu, tasarım amacına uygun olarak, kontrol algoritmaları geliştirmek ve sonuçlarını literatüre sunmak üzere kullanılacaktır. Çalışmaların, üniversitemizin ve ülkemizin bilimsel altyapısına katkıda bulunmasını hedefliyoruz. Çalışmalara daha sonra havuz ortamında devam edilecektir. Farklı kontrol stratejilerinin geliştirildiği ve karşılaştırıldığı uluslararası sualtı aracı yarışmalarına katılmak için gerekli altyapıyı oluşturmaya çalışacağız.

Referanslar

1) http://www.rovcontest.hk/
2) http://ewh.ieee.org/r10/singapore/oes/sauvc/

3) Sibel İnce, “Sualtı Deney Düzeneğinde Derinlik Kontrol Uygulamaları”, Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik ve Haberleşme Mühendisliği AD, Öğrenime Başladığı Tarih:09.09.2009, Öğrenim Durumu: Devam ediyor

4) Ahmet Furkan Ergan, "Sualtı Deney Düzeneğinde Kontrol Kartı Tasarımı ve Yön Kontrolü", Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik ve Haberleşme Mühendisliği AD, Öğrenime Başladığı Tarih:12.09.2011, Öğrenim Durumu: Devam ediyor
5) Tomotaka Inoue, Koji Shibuya and Akinori Nagano, "Underwater Robot with a Buoyancy Control System Based on the Spermaceti Oil Hypothesis - Development of the Depth Control System –" , The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 18-22, 2010, Taipei, Taiwan
6) Paul A. DeBitetto, "Fuzzy Logic for Depth Control of Unmanned Undersea Vehicles", IEEE JOURNAL OF OCEANIC ENGINEERING, VOL. 20, NO. 3, JULY 1995
7) Joonyoung Kim, Kihun Kim, Hang S. Choi, Woojae Seong, and Kyu-Yeul Lee, "Depth and Heading Control for Autonomous Underwater Vehicle Using Estimated Hydrodynamic Coefficients", Department of Naval Architecture & Ocean Engineering, Seoul National University, Seoul 151-742, Kore
8) Hyun-Sik Kim and Yong-Ku Shin, "Design of Adaptive Fuzzy Sliding Mode Controller using FBFE for UFV Depth Control", SICE-ICASE International Joint Conference 2006, Oct. 18-21, 2006 in Bexco, Busan, Korea
9) G. Eren, A. T. Naskal, E. Varol, A. Altınısık, S. M. Egİ,2006. CI-DRAGON:İnsansız Sualtı Aracı, Sualtı Robot Kontrolü İçin YazılımGeliştirme, http://www.idealteknoloji.com/documents/CI-Dragon.pdf
10) Wang, J. S., Lee, C. S. G., 2003. Self-Adaptive Recurrent Neuro-Fuzzy Control of an Autonomous Unerwater Vehicle. IEEE Transactions on Robotics and Automation, 19/2, 283-295.
11) Shi, X., Xiong, H., Wnag, C., Chang, Z., 2005. A New Model of Fuzzy CMAC Network with Application tothe Motion Control of AUV. Proceedings of IEEE International Conference on Mechatronics and Automation, 2173-2178.
12) Song, F., An,E., Folleco,A.2003. Modeling and Simulation of Autonomous Underwater Vehicles: Design and Implementation, IEEE JOURNAL OF OCEANIC ENGINEERING, VOL. 28, NO. 2,283-295.
13) Serhat Yılmaz, Mehmet YAKUT, Sibel İNCE, Tübitak Araştırma Projesi 1. Gelişme Raporu, Proje No:111E294
14) Özer Işık, "Sualtı Yarışmaları için Araç Tasarım İlkeleri", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
15) Murat Otcu, Baran Korkmaz, "İnsansız Sualtı Aracı için Kontrol Kartı", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, Yrd.Doç.Dr. Mehmet Yakut, 2013 Kocaeli
16) Dilek Bilici, Rukiye Acar,Emre Toydemir, Basınç sensöründen veri okuma ve motor sürme, Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, 6. Dönem Projesi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
17) Mehmet Can Döşlü, "Arm Discovery Touch Panel Eklentisi ile Sualtı Aracının

Kontrol Arayüzü", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli

18) Serhat Örs, Tayfun Yapıcı, İnsansız Sualtı Aracı Bilgisayar Arayüzü Tasarımı ", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
19) Gökhan Şenlik, Visual Basic ile İnsansız Sualtı Aracı Bilgisayar Arayüzü Tasarımı ", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Mehmet Yakut, 2013 Kocaeli
20) Arda Erten, TCM3 Elektronik Pusula Sensörü ve Bilgisayarla Seri Haberleşmesi", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
21) Raşit Eygi, "Otonom Sualtı Araçlarının Modellenmesi", Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, Lisans Tezi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
22) Ertuğrul Şahin, "Sualtı Araçlarının Modellenmesi",Kocaeli Üniversitesi, Mühendislik Fakültesi , Elektronik ve Haberleşme Bölümü, 6. Dönem Projesi, Danışman: Yrd.Doç.Dr. Serhat Yılmaz, 2013 Kocaeli
23) Serhat Yılmaz,DA Yüksek Gerilim Dirençsel Bölücülerde Ölçüm Belirsizliğinin Azaltılması için Yeni Yaklaşımlar, Doktora Tezi, Kocaeli Ün. Fen Bilimleri Enst.,2003, Danışman:Prof.Dr. Hasan Dinçer

TÜBİTAK
PROJE ÖZET BİLGİ FORMU

Proje No:111E294
Proje Başlığı: Derinlik ve Yön Kontrol Uygulamaları İçin Deney Platformu Tasarımı
Proje Yürütücüsü ve Araştırmacılar: Serhat YILMAZ, Mehmet YAKUT, Sibel İNCE
Projenin Yürütüldüğü Kuruluş ve Adresi: Kocaeli Üniversitesi Mühendislik Fakültesi Elektronik ve Haberleşme Bölümü, Umuttepe Kampüsü, 41380/Kocaeli
Destekleyen Kuruluş(ların) Adı ve Adresi: TÜBİTAK, Ankara
Projenin Başlangıç ve Bitiş Tarihleri:1 Haziran 2012-1 Haziran 2013
Öz (en çok 70 kelime) Projenin amacı, sualtı aracı teknolojisindeki gelişmelere denetim yöntemleri açısından katkı sağlayacak bir geliştirme platformu tasarlamaktır. Deney platformu, mekanik, elektronik ve yazılım kısımlarından oluşan bir insansız bir sualtı aracı ve içinde hareket kontrol uygulamalarının yapılacağı bozucu etkiler oluşturulabilen su tankından oluşmaktadır. Sürücü ve algılayıcıları da içeren, bilgisayarla haberleşebilen elektronik anakart, aracın gereksinimlerini yerine getirebilecek şekilde tasarlanmış ve hazırladığımız kullanıcı kontrol arayüzü yazılımı ile bağdaştırılmıştır. Diğer yandan aracın bilgisayar ortamında benzetim ve tasarım uygulamalarını gerçekleştirebilmek için matematiksel modeli çıkarılmıştır. Geliştirilen platform temel derinlik ve yön kontrol uygulamalarını yapabilmektedir.
Anahtar Kelimeler: Derinlik kontrolü, yön kontrolü, ROV, Sualtı Sistemleri
Fikri Ürün Bildirim Formu Sunuldu mu? **Evet** ☐ **Gerekli Değil** ☒ Fikri Ürün Bildirim Formu'nun tesliminden sonra 3 ay içerisinde patent başvurusu yapılmalıdır.
Projeden Yapılan Yayınlar:
Ekte Bulunan "ARDEB Başarı Öyküsü Formu", "Kazanımlar" Bölümünde Belirtilen Kriterlere Göre Proje Çıktılarınızın Başarı Öyküsü Niteliği Taşıdığını Düşünüyorsanız "ARDEB Başarı Öyküsü Formu"nu doldurunuz.

ARDEB BAŞARI ÖYKÜSÜ

Derinlik ve Yön Kontrol Uygulamaları İçin Deney Platformu Tasarımı	**Serhat YILMAZ**
	Proje No:111E294
	Destek Miktarı (TL) 24.500
	01.08.14-01.08.13
	Kocaeli Üniversitesi

Projenin Amacı ve Önemi Projenin amacı, sualtı aracı teknolojisindeki gelişmelere denetim yöntemleri açısından katkı sağlayacak bir geliştirme platformu tasarlamaktır.
Deney platformu, mekanik, elektronik ve yazılım kısımlarından oluşan bir insansız bir sualtı aracı ve içinde hareket kontrol uygulamalarının yapılacağı bozucu etkiler oluşturulabilen su tankından oluşmaktadır.
Sürücü ve algılayıcıları da içeren, bilgisayarla haberleşebilen elektronik anakart, aracın gereksinimlerini yerine getirebilecek şekilde tasarlanmış ve hazırladığımız kullanıcı kontrol arayüzü yazılımı ile bağdaştırılmıştır. Diğer yandan aracın bilgisayar ortamında benzetim ve tasarım uygulamalarını gerçekleştirebilmek için matematiksel modeli çıkarılmıştır. Geliştirilen platform temel derinlik ve yön kontrol uygulamalarını yapabilmektedir.

Proje ile Elde Edilen veya Beklenen Bilimsel, Teknolojik, Ekonomik ve Sosyal Kazanımlar

Sualtı aracının kendi imkanlarımızla tasarımı ve üniversitemiz laboratuarında yer alması, üzerinde çeşitli testlerin yapılabilmesi, yazılım geliştirilebilmesi ve farklı denetleyici yapılarının tasarlanabilmesi, üniversitemizde çalışan proje ekibine ve öğrencilerimize belirli bir yetkinlik kazandırmış ve yeni proje çalışmaları için altyapı hazırlamıştır.

Platform üzerinde geliştirilecek kontrol algoritmaları projenin tamamlanmasının ardından yayın ve patent potansiyeli taşımaktadır.

Otonom sualtı araçlarının katıldığı uluslararası yarışmalarda geliştirilen stratejiler, birçok ülkenin savunma bakanlıkları tarafından desteklenmektedir.

Proje için TÜBİTAK Desteğinin Önemi *TÜBİTAK'ın proje desteği, özellikle öğrenci guruplarının bir amaç etrafında toplanması, ortak çalışmaya katılımlarının sağlanması, kaliteli ana tasarım tezleri ortaya çıkarabilmeleri, lisansüstü öğrencilerin bir plan çerçevesinde çalışabilmeleri, buradan literatüre katkı sağlamamız ve yıllardır kafamızda olan şeyleri hayata geçirmemiz açısından bizlere bir çıkış noktası sağlamıştır.*

1. Proje yürütücüsü iletişim bilgileri:
Adı – Soyadı : Serhat YILMAZ
Unvanı :Yrd.Doç.Dr.
Telefon :05324713583
E-posta adresi : serhatyilmaz@ieee.org