



**T.C.
GEBZE TEKNİK ÜNİVERSİTESİ**

Bilgisayar Mühendisliği Bölümü

**RASPIJOY
Uçuş Simülatörü için
Kumanda**

Şevval MEHDER

**Danışman
Prof. Dr. Erkan ZERGEROĞLU**

**Ocak, 2019
Gebze, KOCAELİ**



**T.C.
GEBZE TEKNİK ÜNİVERSİTESİ**

Bilgisayar Mühendisliği Bölümü

**RASPIJOY
Uçuş Simülatörü için
Kumanda**

Şevval MEHDER

**Danışman
Prof. Dr. Erkan ZERGEROĞLU**

**Ocak, 2019
Gebze, KOCAELİ**

Bu çalışma / /2018 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Bölümünde Lisans Bitirme Projesi olarak kabul edilmiştir.

Bitirme Projesi Jürisi

Danışman Adı	Prof. Dr. Erkan ZERGEROĞLU	
Üniversite	Gebze Teknik Üniversitesi	
Fakülte	Mühendislik Fakültesi	

Jüri Adı	Dr. Öğr. Üyesi Alp Arslan BAYRAKÇI	
Üniversite	Gebze Teknik Üniversitesi	
Fakülte	Mühendislik Fakültesi	

ÖNSÖZ

Bu raporun ilk taslaklarının hazırlanmasında emeği geçenlere, raporun son halini almasında yol gösterici olan Sayın Prof. Dr. Erkan ZERGEROĞLU hocama ve bu çalışmayı destekleyen Gebze Teknik Üniversitesi'ne içten teşekkürlerimi sunarım.

Ayrıca eğitimim süresince bana her konuda tam destek veren aileme ve bana hayatlarıyla örnek olan tüm hocalarıma saygı ve sevgilerimi sunarım.

Ocak, 2019

Şevval MEHDER

İÇİNDEKİLER

ÖNSÖZ.....	6
İÇİNDEKİLER	7
ŞEKİL LİSTESİ.....	8
TABLO LİSTESİ	9
KISALTMA LİSTESİ	10
ÖZET	11
SUMMARY	12
1 GİRİŞ	13
2 ARKAPLAN	15
3 METOT	17
3.1 RASPBERRY Pİ ZERO’NUN USB AYGIT OLARAK TANITILMASI.....	17
3.2 KULLANICI GİRDİLERİNİN ALINMASI	22
3.3 RASPBERRY Pİ VE BİLGİSAYAR ARASINDAKİ İLETİŞİM.....	23
3.4 RASPIJOY’UN YER İSTASYONUNA TANITILMASI.....	24
4 DENEYLER VE BULGULAR	25
4.1 AŞAMALARIN GERÇEKLEŞTİRİLMESİ VE TEST EDİLMESİ.....	25
4.1.1 Raspberry Pi Zero’nun USB Aygıtı Olarak Tanıtılması.....	26
4.1.2 Bilgisayar ile İletişimin Sağlanması	29
4.1.3 RaspiJoy’un Yer İstasyonuna Tanıtılması	30
4.1.4 Kullanıcı Girdilerinin Alınması	31
4.2 BAŞARI KRİTERLERİNİN KONTROLÜ	33
5 SONUÇ	37
6 KAYNAKLAR.....	38

ŞEKİL LİSTESİ

ŞEKİL 1.1.1 SITL ve Gazebo Karşılaştırması.....	14
ŞEKİL 2.1 Mod 2 için kumanda kontrollerinin eşlenme şekli	16
ŞEKİL 3.1 Cihaz tanımlayıcısının yapısı ve sahip olduğu diğer tanımlayıcılar. Arayüz tanımlayıcısı cihazın fonksiyonlarını tanımlar, Endpoint ise cihazın üreteceği bilginin nasıl okunacağını bilgisini içerir. [4]	19
ŞEKİL 3.2 Kumanda üzerinde butonların yerleri ve görevleri.....	20
ŞEKİL 3.3 RaspiJoy için yazılan tanımlayıcı. Sağ tarafta bulunan değerler, sol taraftaki maddelerin onaltılık tabanda(hexadecimal) gösterimidir.	21
ŞEKİL 3.4 Kullanılacak kablonun bir ucu bilgisayarın USB portunda diğer ucu ise Rasperry Pi Zero'nun USB portunda olmalıdır.[8].....	21
ŞEKİL 3.5 Kullanıcı girdileri için kullanılan bileşenler. Solda 2 eksenli analog kumanda kolu, sağda 4 bacaklı 12x12x1 mm boyutunda buton.....	22
ŞEKİL 3.6 Projede kullanılan analog sinyali dijitale çeviren dönüştürücü. Solda MCP3008 çipi görünümü, sağda ise çipin pin çıkışları gösterilmiştir[9]	22
ŞEKİL 3.7 Sürücü ve USB arasındaki iletişim Control pipe ya da Interrupt pipe şeklinde gerçekleşebilir.....	23
ŞEKİL 4.1 usb_gadget klasörü altında kendi USB aygıtımız için bir klasör oluşturduğumuzda bir aygıt tanımlayıcısı oluşturabilmek için otomatik olarak oluşan şablon	26
ŞEKİL 4.2 Şablonun karar verilen uygun aygıt tanımlayıcısı şeklinde yazılması işlemi bir betik yardımı ile yapıldı.....	27
ŞEKİL 4.3 USB Aygıtın fonksiyonel karakteristiğini gösteren betik satırları.	27
ŞEKİL 4.4 ŞEKİL 4.4 Oluşturulan betik dosyası Raspberry'nin başlangıcında çalışması için /etc/rc.local dosyasına “exit 0” ifadesinin üstünde kalacak şekilde betiği çalıştıracak satır eklendi.....	28
ŞEKİL 4.5 lsusb komutu bilgisayara takılı olan USB aygıtları gösteren komuttur. Listedeki Linux Foundation Multifunction Composite Gadget RaspiJoy'dur.	28
ŞEKİL 4.6 Takılan aygıtın bilgisayar tarafında usbid sürücü ile bağlandığının kontrolü	29
ŞEKİL 4.7 Raspberry tarafından kumandanın 4. Axis değerinin değiştirilmesinin bilgisayar tarafında gözlemlenmesi	30
ŞEKİL 4.8 MAVProxy'de modülü yükleme komutundan sonra modül yüklenirken RaspiJoy ile eşleşilmiş ve yüklemiştir. Aktif kumandalar kontrol edildiğinde RaspiJoy'un aktif olduğu görülebilir.	31
ŞEKİL 4.9 Kanaldan okunan 10 bitlik değer 7 bite çevrilebilmek için 8e bölünmüştür	31
ŞEKİL 4.10 Butonların çalışma prensibi	32
ŞEKİL 4.11 Buton hareketlerinin MAVProxy konsolunda gerçekleşmesi.....	33
ŞEKİL 4.12 Sol kumanda kolu gaz değerinin ölçülmesi.....	34
ŞEKİL 4.13 Sağ kumanda kolu sağ ve sol hareketin gözlemlenmesi.....	34
ŞEKİL 4.14 Sağ kumanda kolu ileri ve geri hareketin gözlemlenmesi.....	35
ŞEKİL 4.15 Eksenlerin gecikmelerinin kontrolü.....	36
ŞEKİL 4.16 Örnek bir buton gecikmesinin kontrolü.....	36

TABLO LİSTESİ

TABLO 3.1 RaspiJoy'un ürettiği veri parçalarının sırası ve dizilimi	19
TABLO 4.1 Butonlar ve basamak değerleri	32

KISALTMA LİSTESİ

GTÜ	: Gebze Teknik Üniversitesi
İHA	: İnsansız Hava Aracı
UAV	: İnsansız Hava Aracı (Unmanned Aerial Vehicle)
SITL	: Döngüdeki Yazılım (Software In The Loop)
USB	: Evrensel Seri Veriyolu (Universal Serial Bus)
DIY	: Kendin Yap (Do It Yourself)
HID	: İnsan Arabilirim Cihazı (Human Interface Device)
USB-IF	: USB Uygulayıcılar Forumu (USB-Implementers Forum)
SPI	: Seri Haberleşme Protokolü (Serial Peripheral Interface)
RC	: Uzaktan Kontrol (Remote Control)
MSB	: En anlamlı bit (Most significant bit)

ÖZET

Günümüzde insansız hava aracı kullanımı hızla yaygınlaşmış ve bir hobi haline gelmiştir. Ancak tecrübesiz bir insanın uçurma çalışmaları sadece uçurduğu araca değil, kendisine ve çevresine de zarar vermesine sebep olabilmektedir. Böyle kazalar sonucu kişinin özgüveni de zedelenmektedir. Bu durumlarda oluşabilecek maddi ve manevi zararları en aza indirmek için, hiçbir donanım ihtiyacı duymadan uçak ve helikopter kullanmanıza izin veren simülasyon programları bulunmaktadır. Bu projede, bahsedilen simülasyon programlarından biri olan SITL için Raspberry pi Zero kullanılarak düşük maliyetli bir simülatör kumandası yapılmıştır.

RaspiJoy ismi verilen proje, İHA yapmaya ve uçurmaya meraklı olan kişilerin öğrenme sürecinde kendi ihtiyaç duyduğu fonksiyonları kumandada istediği yere istediği şekilde koymasına ve gerçek bir uçuş yapmadan önce bu kendi tasarladıkları kumanda sayesinde uçuş kontrollerine alışıp öğrenme sürecini en az maliyetle tamamlamasına olanak sağlar. Bu rapor, kullanıcının alışma sürecinde birçok fonksiyonu gerçekleştirebilen karmaşık kumandalar yerine uçuşa alışmasını kolaylaştıracak, basit ve bildiği bir kumandayı nasıl yapacağını anlatır.

SUMMARY

Nowadays, Unmanned Aerial Vehicle usage become widespread and a hobby. However when inexperienced players try to fly the drone, they will not only damage themselves, but damage the drone and the environment as well. In consequence of this accidents, they lose their self confidence. There are simulators that allow you to run drone without any hardware to minimize the pecuniary loss and intangible damages. In this project a low-cost joystick was designed and developed for SITL Simulator with Raspberry Pi Zero.

This project is named RaspiJoy allows people who have interest in fly the drone to design the flight pad according to their comfort zone during learning process. And player become familiar with flight controls, functions and modes with own handmade joystick before the real flight practice. This option decrease the costs and shorten the adaptation period. The report details making simple, own simulator joystick instead of general complex RC transmitters.

1 GİRİŞ

Kullanımı her geçen gün yaygınlaşan insansız hava araçlarını uçurmak birçok insanın hobisi haline geldi. Uluslararası yarışları da düzenlenmekte olan İHA uçurmanın öğrenilmesi için internette birçok tavsiye yazısı ve öğretici videolar bulunmakta. Bunlar ne kadar faydalı olursa olsun uzaktan kumanda ile gerçek uçuş denemesinde kazalar meydana gelmekte ve bu kazalarla kişi sadece İHA'ya değil, kendine ve çevresine de zarar vermektedir. Üstelik kazalar sonucu kişinin kendine olan özgüveni de zedelenebilmektedir.

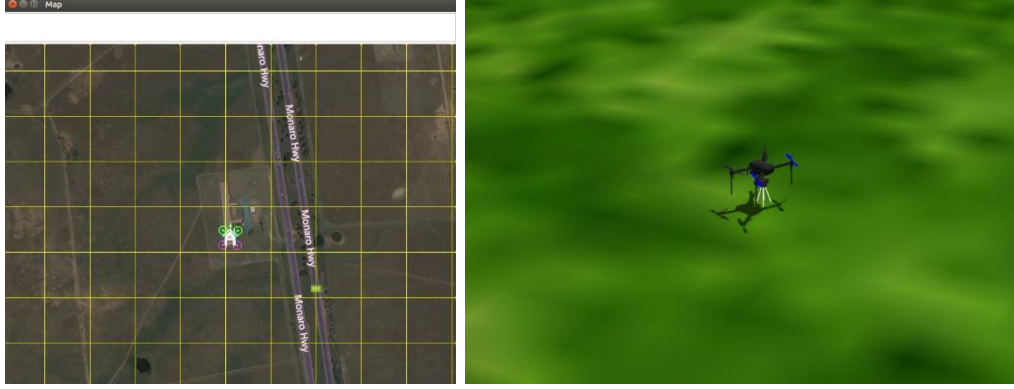
Öğrenme sürecindeki maaliyeti en aza indirmenin yolu bir uzaktan kumanda yardımı ile simülasyon programı kullanmaktır. Çoğu kendin yap(DIY) grupları oyun konsollarının uçuş kontrolleri için uyarlanması konusunda tavsiyede bulunur.[1]

Uçuş simülasyonları gerçek bir hava aracı donanımına sahip olmadan uçuşunuzu simüle etmeye yarayan yazılımlardır. Simülasyonlar, İHA'ların kontrolünü sağlayan kontrol kartlarının üstünde çalışan otopilot yazılımlarının bilgisayarınızın üstünde çalışmasını sağlar.

Projede Raspberry Pi Zero kullanılarak USB ile bilgisayara bağlanan, SITL simülasyon programı için bir kumanda yapıldı. Böyle bir kumanda yapılmasındaki en önemli motivasyon, hangi yaştan olursa olsun herkesin İHA uçurmayı öğrenme konusundaki istekliliği oldu. Yapılan araştırmalar sonucunda özellikle oyunlara ilgi duyan bir çok insanın oyun oynarken kullandığı oyun konsolunu kendi zevklerine göre dizayn etme isteği projenin gidişatında önemli bir rol oynadı.

SITL kendisine bilgi sağlayan yer istasyonu ile beraber 2 boyutlu bir harita sunmaktadır. Kullanıcının elindeki kumanda sayesinde sağladığı hareket neticesinde haritada izlenen rota çizilmektedir. Bu uçuş deneyiminin gerçeğe en yakın şekilde olabilmesi için 3 boyutlu bir robot simülasyon programı olan Gazebo kullanılmıştır. Programlar doğru bir şekilde haberleştikten sonra, kullanıcının kumanda ile yaptığı

hareketleri gerçek zamanlı olarak görüntüleyebildiği 3 boyutlu bir sistem oluşturulmuştur. İki görüntünün karşılaştırılması ŞEKİL 1.1’de gösterilmiştir.



(a) SITL görüntüsü

(b) Gazebo görüntüsü

ŞEKİL 1.1.1 SITL ve Gazebo Karşılaştırması

İHA kontrolünü sağlamak için mevcut otopilot yazılımları arasında en gelişmiş ve güvenilir olan açık kaynak kodlu ArduPilot yazılımı kullanıldı. ArduPilot İHA konusunda uçak ve helikopter(copter) ismi verilen iki farklı ürün sunuyor. Raporda bahsedilen bütün kontroller, uçuş modları, parametreler ve uçuş testleri 4 pervaneli helikopter(quadcopter) kullanılarak yapıldı.

Özetle RaspiJoy, içerisinde bilgisayar olarak görevlerini yapmaya devam eden Raspberry Pi Zero sayesinde kullanımı basit ve kolay erişilebilir bir kumanda sunar.

Rapor ilgili konulara değinen arkaplan ile devam ediyor. Bölüm 3’te kullanılan method ve bölüm 4’te bu methodun RaspiJoy için nasıl özelleştiği ve nasıl gerçekleştirildiğine dair deneyler anlatıldı. Ardından sonuçlarından ve ilerde yapılabilecek yeniliklerden bahsedildi.

2 ARKAPLAN

İnsansız hava araçlarında temel uçuş görevlerini yerine getiren sistemler, bir otopilot sistemi ve yer istasyonu yazılımıdır. Otopilot sistemleri İHA'ların kontrolünü sağlayan kontrol kartları üstünde çalışır ve sensörleri içerir. Yer istasyonu sayesinde de İHA'lar takip edilir, görev atamaları yapılır, veri alışverişi sağlanır. Ardupilot otopilot programı birçok yer istasyonu ile uyumlu çalışabilmektedir. Projede kullanılacak yer istasyonu olarak genelde geliştiricilerin kullandığı MAVProxy tercih edildi.

USB otomatik konfigürasyon yapabilen güvenli bir veriyoludur. Bu nedenle yapılacak kumandanın bilgisayar ile bağlantısında USB protokolü kullanılmasına karar verildi. Yapılan kumandanın yer istasyonu tarafından tanınabilir olması, USB port aracılığıyla alınan giriş değerlerinin okunması ve simülasyon için dönüştürülmesi konusunda MAVPROxy joystick modülü kullanıldı.

USB ile çalışılmaya karar verilmesi, kumandanın simülasyonun çalıştırılacağı bilgisayar USB port aracılığıyla ile bağlanacağını ifade eder. Ancak bilindiği üzere Raspberry Pi ürünleri bir USB aygıtı değil mini bir bilgisayardır. Bu nedenle bilgisayara bağlandığında USB protokolü onları bir cihaz olarak tanımaz. Bu noktada Andrew Mulholland'ın blog yazısından ilham alındı[2]. Andrew'in yazısına göre Pi Zero'nun donanım özelliği olarak, diğer Raspberry Pi modellerinin aksine, USB portun doğrudan işlemciye bağlı olması Zero'nun USB aygıtı gibi davranabilmesine olanak sağlıyordu. Bu nedenle projede Raspberry Pi Zero kullanılmasına karar verildi.

Kumandalar kanallara sahiptir. Her bir kanal kumandanın yapabileceği bir kontrol fonksiyonu ile eşleşir. Tek kanallı bir kumanda ile İHA üzerindeki tek bir hareketi kontrol edebilirsiniz. ArduPilot için varsayılan ayarlar şu şekildedir:

- Kanal 1: Sağ/Sol (Roll)
- Kanal 2: İleri/Geri (Pitch)

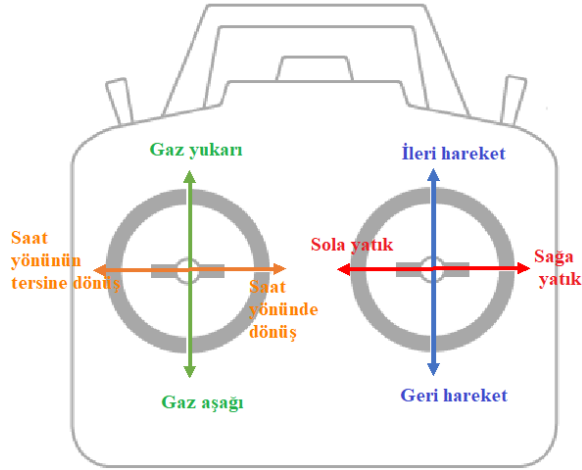
- Kanal 3: Gaz (Throttle)
- Kanal 4: Saat yönü/Saat yönünün tersi (Yaw)

RaspiJoy 6 kanallı bir kumandadır. Bahsedilen 4 kontrole ek olarak:

- Channel 5: Uçuş modları arasında seçim yapar
- Channel 6: Kalkış için pervaneleri motorlarını çalıştırır ve durdurur.

Bu kontroller için kumandada 2 tane 2 eksenli analog kumanda kolu sayesinde 4 temel eksen kontrolleri, 3 buton ile uçuş modları arasında geçiş ve diğer 1 butonla da kalkış ve iniş için pervanelerin kontrolü sağlanmıştır.

Gerek uçuşta kullanılan uzaktan kumandalarda gerekse oyunlar için kullanılan iki adet analog kola sahip oyun konsollarında bu kolların sağladığı kontroller konusunda 4 farklı dizilimde mod vardır. Bu dizilimlerde 4 temel kontrolün yerleri değişir. RaspiJoy 2. Modu kullanır. Mod 2 için kontroller şu şekildedir:



ŞEKİL 2.1 Mod 2 için kumanda kontrollerinin eşlenme şekli

3 METOT

Bu proje temelde biri Raspberry Pi Zero, diğeri normal bilgisayar olmak üzere iki bilgisayar üzerinde yapılacaktır. Pi Zero'nun bağlanacağı bilgisayar ev sahibi(host) olarak adlandırılır ve raporun devamında bu şekilde bahsedilecektir.

Basitçe yapılacak 3 aşama vardır:

1. Raspberry Pi Zero'nun USB aygıt olarak tanıtılması
2. Kullanıcı girdilerinin alınması
3. Raspberry Pi ve ev sahibi bilgisayar arasındaki iletişim
4. Ev sahibi bilgisayar tarafında bağlanan RaspiJoy'un Yer İstasyonuna tanıtılması

3.1 Raspberry Pi Zero'nun USB Aygıt Olarak Tanıtılması

Standart USB protokolü efendi/köle(master/slave) mimarisine sahiptir. USB aygıtlar slave, aygıtın takıldığı ev sahibi bilgisayar da master rolündedir. Bu ilişkide, ev sahibi bilgisayar cihaz ile iletişim kurar ve cihazın ürettiği girdileri alır. Raspberry Pi ise mini bir bilgisayardır. Ve USB port'a bağlandığında bir USB aygıtı olarak algınmaz. Bu nedenle projenin ilk adımında Raspberry Pi bilgisayarın bir USB aygıtıymış gibi davranması için çalışıldı.

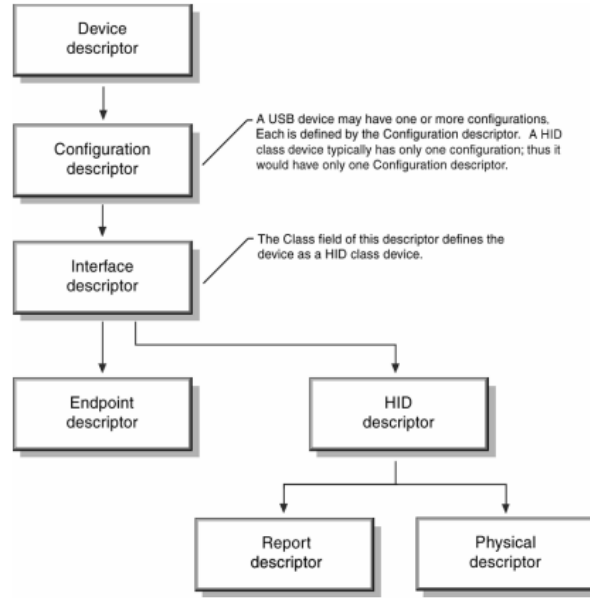
USB cihazları kendilerini üretici kimlik numaraları(Vendor ID) ve ürün kimlik numaraları(Product ID) ile tanıtır. Ev sahibi bilgisayar bu numara sayesinde cihazı tanır. Bu numaraların kontrolü usb.org tarafından yönetilir. Belli bir ücret karşılığı alınır.[3] Ürün kimlik numaraları ise, üretici numarasının sahibi tarafından yönetilir. Bu ikisinin kombinasyonu özgün olmalıdır. USB cihazlar bu iki numara ile beraber cihazın özelliklerini anlatan ve cihaz tanımlayıcısı ismi verilen bilgi paketine sahiptir. Cihaz tanımlayıcılarının sahip olabileceği bazı değerler RaspiJoy için şu şekilde planlandı:

- Yeni bir üretici kimlik numarası alınmadı. Proje Linux üzerinde yapılan açık kaynak kodlu bir proje olacağı için sahibi Linux Kuruluşu olan 0x1d6b üretici numarası kullanıldı.

- Ürün kimlik numarası olarak Linux Kuruluşu'nun çok işlevli aygıtlar için kullandığı 0x0101 ürün numarası kullanıldı. Bu numaralar Linux USB projesi sayfasında bulunmaktadır.[4]
- *bcdUSB* değeri için USB 2.0 portları kullanılacağı için 0x200 değeri,
- *bcdDevice* için RaspiJoy'un ilk versiyonu olduğu için 0x0100 değeri kullanıldı.
- *iManufacturer*, *iProduct*, *iSerialNumber* özellikleri karakter dizisi olarak gösterildi.
- *bNumConfiguration* değeri ise, güç tüketim konfigürasyonu yapılması için 1 adet olarak belirlendi.

USB cihazları, cihazın özelliklerine desteklenen isteklerine ve veri protokollerine dayalı olarak sınıflar halinde düzenlenmiştir. Her cihaz sınıfı benzer işlevler sunar ve ortak davranışlar sergiler. Örneğin monitörler görüntüleme(display) sınıfına, hoparlörler ses(Audio) sınıfına, hard diskler yığınsal depolama(mass storage) ve klavye joystick gibi cihazlar da insan arabirim(human interface) sınıfına dahildir. USB protokolü, cihazları başlangıçta veya çalışma zamanında yapılandırabilir [5] RaspiJoy, insan arabirim cihazı sınıfında yer alır. İnsan arabirim cihazları, bilgisayar sistemlerinin çalışmasını kontrol etmek için insanlar tarafından kullanılan cihazlardır.

Cihaz tanımlayıcıları içerisinde insan arabirim cihazları için HID sınıf tanımlayıcılarına sahiptir. Bu tanımlayıcılar arasındaki ilişki ŞEKİL 3.1'de gösterilmiştir.HID tanımlayıcılar, aygıtın veri paketini açıklayan sabit kodlanmış bir bayt dizisidir.[6] Bu tanımlayıcılar rapor tanımlayıcılar ve fiziksel tanımlayıcılar olarak ayrılır. Rapor tanımlayıcılar cihazın ürettiği veri parçalarını etiketi, tipi, boyutu yönünden tanımlarlar. Fiziksel tanımlayıcılar isteğe bağlı tanımlanır ve genelde insan vücudu ile kontrol edilecek cihazlarda kullanılır.



ŞEKİL 3.1 Cihaz tanımlayıcısının yapısı ve sahip olduğu diğer tanımlayıcılar. Arayüz tanımlayıcısı cihazın fonksiyonlarını tanımlar, Endpoint ise cihazın üreteceği bilginin nasıl okunacağını bilgisini içerir. [4]

RaspiJoy için fiziksel tanımlayıcıya ihtiyaç yoktur. Bir rapor tanımlayıcı yazmak için öncelikle yapacağımız kumandanın üreteceği veri planlandı. Bu veriler ve gönderim sırası Tablo 3.1’de gösterildiği gibidir:

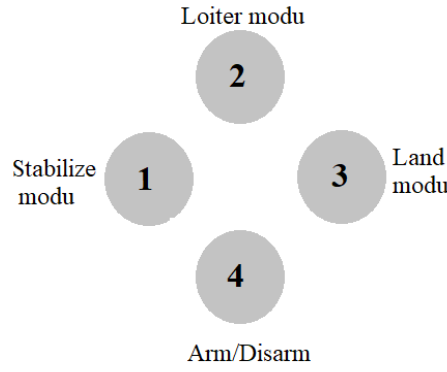
TABLO 3.1 RaspiJoy’un ürettiği veri parçalarının sırası ve dizilimi

	7. Bit	6. Bit	5. Bit	4. Bit	3. Bit	2. Bit	1. Bit	0. Bit
0.Bayt	Önemsiz	Önemsiz	Önemsiz	Öncemsiz	4.Buton	3.Buton	2.Buton	1.Buton
1.Bayt	Sol Kumanda Kolu X Eksen Hareketi							
2.Bayt	Sol Kumanda Kolu Y Eksen Hareketi							
3.Bayt	Sağ Kumanda Kolu X Eksen Hareketi							
4.Bayt	Sağ Kumanda Kolu Y Eksen Hareketi							

RaspiJoy üzerinde 4 buton olması planlandığı için buton verileri konusunda önemli olan veriler son 4 bitteki verilerdir. Ancak gönderilen veriyi 1 bayta tamamlamak gerektiği için 8 bit gönderilmiştir. RaspiJoy 8 butonu kontrol eden bir aygıt olarak

görülmektedir ancak bilgisayara gönderilecek veride ilk 4 bit her zaman 0(sıfır) değerini alacaktır.

Tabloda bahsedilen 1,2 ve 3 numaralı butonlar uçuş modları arasında geçişi sağlar. 1 numaralı buton stabilize(stabilize) moduna, 2 numaralı buton oyalanma(loiter) moduna, 3 numaralı buton iniş(land) moduna geçirir. 4 numaralı buton ise pervane motorlarının kontrolünü sağlayan(arm/disarm) butondur. Bahsedilen butonların, kumanda üzerinde nasıl sıralanacağı, hangi butonun hangi göreve karşılık geldiği bilgisi ŞEKİL 3.2’de gösterilmiştir:

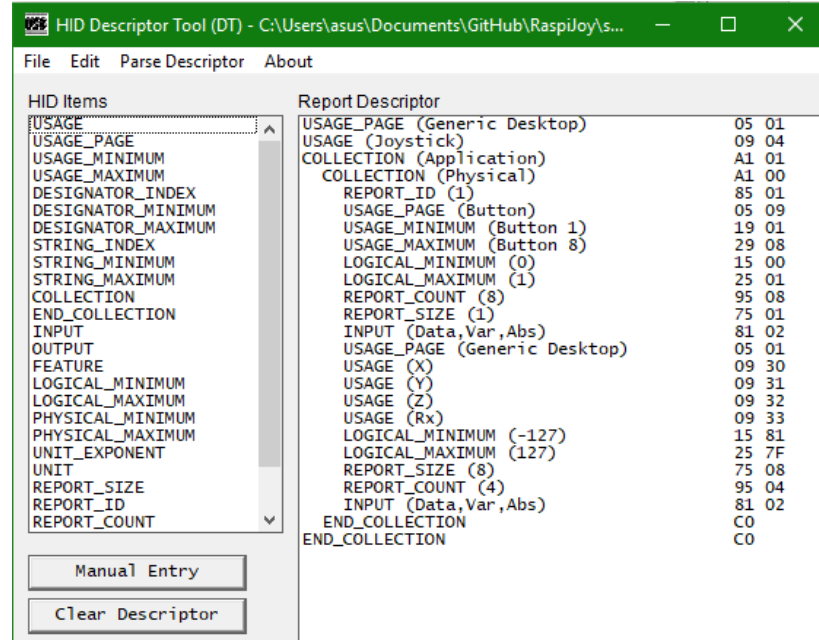


ŞEKİL 3.2 Kumanda üzerinde butonların yerleri ve görevleri

Rapor tanımlayıcılarda etiketini, tipini ve boyutunu tanımlamak zorunda olunan ana veri parçalarının her biri madde(item) olarak isimlendirilir. Bir rapor tanımlayıcı şu maddelere sahip olmak zorundadır:

- Girdi (Çıktı ya da özellik tipinde de olabilir)
- Kullanım şekli (Usage)
- Kullanım sayfası (Usage page)
- Mantıksal minimum
- Mantıksal maksimum
- Rapor boyutu
- Rapor sayısı

Bu tanımlayıcıları yazmak, yazılan tanımlayıcıyı düzenlemek için USB Uygulayıcılar Forumu(USB-IF) tarafından sunulan bir araç bulunmaktadır.[7] Bu araç kullanılarak RaspiJoy için yazılan tanımlayıcı ŞEKİL 3.3'te gösterilmiştir.



ŞEKİL 3.3 RaspiJoy için yazılan tanımlayıcı. Sağ tarafta bulunan değerler, sol taraftaki maddelerin onaltılık tabanda(hexadecimal) gösterimidir.

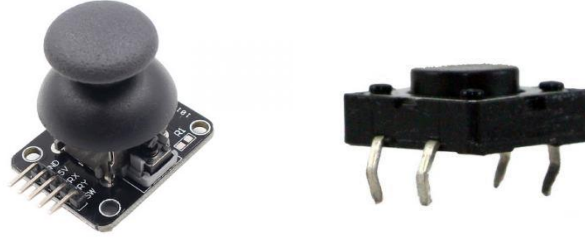
Raspberry Pi Zero USB ve POWER portlarına sahiptir. USB port, veri alışverişi yapılan port diğeri ise güce bağlandığı porttur. Raspberry Pi Zero'yu bilgisayara USB aygıtı gibi göstermek istediğimiz zaman, Pi Zero'daki USB portu yani veri alış verişi yapılan portu kullanmamız lazımdır. ŞEKİL 3.4'te de kablo USB port'a takılmıştır.



ŞEKİL 3.4 Kullanılacak kablonun bir ucu bilgisayarın USB portunda diğeri ucu ise Raspberry Pi Zero'nun USB portunda olmalıdır.[8]

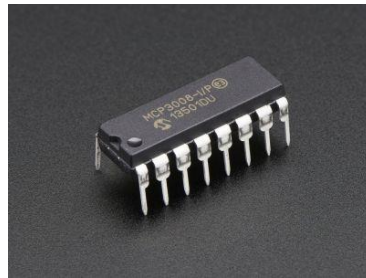
3.2 Kullanıcı Girdilerinin Alınması

RaspiJoy'da kullanıcıların kontrol fonksiyonlarını gerçekleştirebilmesi için Şekil 3.5'te görülen 2 eksenli analog kumanda kolundan 2 tane ve butonlardan da 4 tane kullanılmıştır.



ŞEKİL 3.5 Kullanıcı girdileri için kullanılan bileşenler. Solda 2 eksenli analog kumanda kolu, sağda 4 bacaklı 12x12x1 mm boyutunda buton

Raspberry Pi Zero analog girdi/çıkı pinine sahip değildir. Bu nedenle kullanılacak kumanda kolları raspberry ile doğrudan bağlanamaz. Arasındaki iletişimi sağlayabilmek için kumanda kolundaki analog sinyalleri dijital sinyallere çevirebilen dönüştürücüye ihtiyaç vardır. Projemizde bu görevi sağlaması için ŞEKİL 3.6'da görülen 8 kanallı 10 bit MCP3008 kodlu çipi kullandık. Bu çip, dijital ve analog sinyal arasında köprü görevi görür. Cihazlarla iletişim SPI protokolüyle gerçekleştirilir.



CH0	1	16	V _{DD}
CH1	2	15	V _{REF}
CH2	3	14	AGND
CH3	4	13	CLK
CH4	5	12	D _{OUT}
CH5	6	11	D _{IN}
CH6	7	10	CS/SHDN
CH7	8	9	DGND

ŞEKİL 3.6 Projede kullanılan analog sinyali dijitalle çeviren dönüştürücü. Solda MCP3008 çipi görünümü, sağda ise çipin pin çıkışları gösterilmiştir[9]

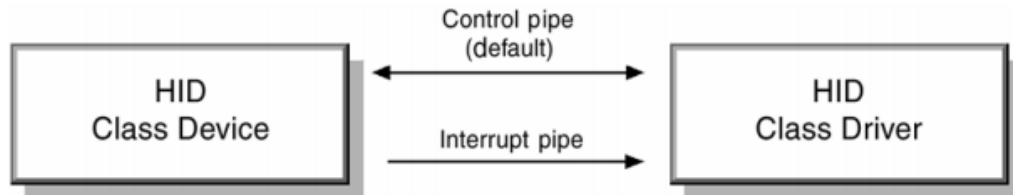
MCP3008, çalışabileceği en düşük voltaj değerinde(2.7V) 1.35 MHz frekansa sahiptir ki bu da yaklaşık 0.0007 milisaniyede bir okuma yapabildiğini gösterir.[9] Projede çipe 3.3 volt sağlandı ve çipten dolayı olabilecek gecikme olabildiğince düşürüldü.

Çipin Raspberry Pi Zero'ya bağlanması için, Raspberry'nin donanım SPI çıkışları kullanıldı. Bu pinler kullanılırken Raspberry Konfigürasyonlarından SPI etkinleştirilmelidir.

3.3 Raspberry Pi ve Bilgisayar Arasındaki İletişim

USB HID sınıfına ait bir cihaz tüm verileri yönlendirmek için karşılık gelen HID sınıfı bir sürücü kullanır(driver). Bu verilerin yönlendirilmesi ve alınması cihazın tanımlayıcıları ve sağladığı veriler incelenerek gerçekleşir. HID sınıfı bir sürücü tanımlayıcıları inceleyerek alacağı verinin boyutunu belirler.

Bir USB ve sürücü arasındaki iletişim ŞEKİL 3.7'de görülebileceği gibi iki farklı şekilde gerçekleşebilir. Kontrol veri yolu genelde ev sahibi bilgisayardan aygıtı bir bilgi aktarılacaksa olur ve cihazların birbirinden bilgi istemesi ile sağlanır. Kesmeli veri yolları(interrupt) ise ev sahibi bilgisayarın cihazdan senkron olmayan bir hareketle bilgi okuması demektir.



ŞEKİL 3.7 Sürücü ve USB arasındaki iletişim Control pipe ya da Interrupt pipe şeklinde gerçekleşebilir.

Küçük verilerde hızlı veri aktarımı için kesmeli veri yolu tercih edilir. Küçük veri ise 8 bayt ile sınırlandırılmıştır. 8 bayt ve daha küçük veriler kesmeli veri yolu ile gönderilirler. RaspiJoy daha önce belirlendiği üzere 5 bayt bilgi göndermekte ve ev sahibi bilgisayardan da bir bilgi almamaktadır. Bu nedenle RaspiJoy ve ev sahibi bilgisayardaki sürücü arasındaki veri kesmeli veri yolu ile gönderilir.

USB aygıtı, uygun sürücüye bağlandıktan sonra bir cihaz düğümü (device node) oluşur. Bu düğümler Linux'ta /dev klasörünün altında bulunur. USB aygıtlardan girdi aygıtı olanlar için /dev klasörünün altında input/ klasörü bulunmaktadır. Bu klasörün altındaki js#(js0, js1) şeklinde olanlar joystick içindir.

RaspiJoy bilgisayara takıldıktan sonra /dev/input/js0 dosyası oluşur ve dosyanın içi okunmaya çalışıldığında kumanda üzerinde hareket oluşturdukça yeni veriler eklendiği görülür.

3.4 RaspiJoy'un Yer İstasyonuna tanıtılması

Bilgisayarda beraber çalışması gereken programlar, bilgisayar üzerinde ArduPilot kontrollerini gerçekleştirebilen SITL simülasyonu, MAVProxy yer istasyonu yazılımı ve Gazebo 3 boyutlu robotik simülasyondur. Özellikle Gazebo için platform konusunda Ubuntu 16.04 önerilmektedir. Tüm testler de Ubuntu 16.04'te yapılmıştır. SITL simülasyonu otomatik olarak başka komuta gerek duyulmadan MAVProxy yer istasyonunu çalıştırır.

Simülasyondaki İHA'ların kontrolü yer istasyonları sayesinde sağlanır. Bu nedenle kullanılacak kumandalar yer istasyonuna bağlanmalıdır. MAVProxy bu konuda joystick ismi verilen modüle sahip. Bu modülü yüklediğiniz anda eşleşen bir kumanda bulunmuşsa sistemde aktif hale gelir ve kontrol sinyalleri bağlanır. Kumandanızı daha sonra takmanız durumunda da başka bir komutla uygun kumandaları tekrar tarar ve eşleşen bir kumanda olduğunda bağlanabilir. MAVProxy en fazla 8 kanallı bir kumandanın kontrol sinyallerini kontrol edebilir.

MAVProxy'nin joystick modülü için bazı kontrol tiplerine sahiptir. Modülün yaptığı şey temelde kumanda üzerindeki bu kontrolleri RC kanalları ile eşleştirmek.

Bu kontrol tipleri bazı parametreler ile kullanılır. Bu parametreler:

- inlow, inhigh : Kontrolün alabileceği girdi değerlerinin aralığını belirler. Varsayılan olarak -1 ve 1 değerlerini alır.
- outlow, outhigh: Kontrolün çıktı değerlerinin aralığını belirler. Varsayılan olarak 1000 ve 2000 değerlerini alır.

Bahsedilen kontrol tiplerinden RaspiJoy'da kullanılanlar de şu şekildedir:

- Eksenler: RaspiJoy'da 4 eksen kontrolü kullanılmıştır.
- Butonlar: Butona basıldığında, eşlendiği rc kanalı outhigh değerini alır. Bırakıldığında ise outlow değeri gönderilir. RaspiJoy'da 1 adet buton kontrolü kullanılmıştır.
- Çoklu buton: Aynı kanala birden fazla butonun bağlanması multibuton olarak adlandırılır. Genelde iki veya daha fazla seçenek arasında seçim yapmak için kullanılır. Her butona farklı bir değer atanır ve o butona basıldığında kanala ilgili değer gönderilir ancak bırakıldığında bir değişiklik yapılmaz. RaspiJoy'da 1 adet çoklu buton kontrolü vardır

MAVProxy python modülünün içerisinde joystick modülü klasöründe MAVProxy tarafından tanınan joystick'lerin tanımları YAML formatında saklanır ve simülasyon sırasında bu tanımlar dinamik olarak yüklenir. RaspiJoy için daha önce bölüm 3.1'de bahsedilmiş olan varsayılan kanal eksen atamalarına uyumlu olarak kanal-eksen ataması sağlandı.

4 DENEYLER VE BULGULAR

Method bölümünde detayları anlatılan 4 aşama birbirinden ayrı modüller şeklinde değildir. Her bir aşamanın sonunda testler sağlanmış ve diğer aşama onun üzerine inşa edilmiştir. Her aşama birbirini etkilemektedir. Son aşamanın da testinden sonra yer istasyonuna bağlanabilen kumanda yapmış oluruz. Bundan sonra başarı kriterlerinin sağlanıp sağlanamadığının kontrolü gerçekleştirilir.

4.1 Aşamaların Gerçekleştirilmesi ve Test Edilmesi

Method bölümünde anlatılan aşamaların gerçekleşmesi sırası şu şekilde oldu:

1. Raspberry Pi Zero'nun USB aygıtı olarak tanıtılması
2. Ev sahibi bilgisayar ile iletişimin sağlanması
3. RaspiJoy'un yer istasyonuna tanıtılması
4. Kullanıcı girdilerinin alınması

4.1.1 Raspberry Pi Zero'nun USB Aygıtı Olarak Tanıtılması

Bu işlemin gerçekleşmesi için öncelikle iki temel şart vardır:

1. Raspberry içerisindeki Raspian işletim sistemi güncel çekirdeğe sahip olmalı
2. Bağlantı için Raspberry'nin USB portu kullanılmalı

USB portu kullanılarak gereken güç bilgisayardan sağlanmakta ve güç portunun kullanılmasına gerek kalmamaktadır.

Andrew tarafından yazılan blog yazısında [2] söylendiği üzere dcw2 modülü ve kompozit aygıtlar için kullanılan libcomposit modülü yüklendi.

Configfs ismi verilen dosya sistemi Linux 3.11 ile gelen kullanıcıya özel USB kompozit aygıt tanımlamak için isteğe bağlı fonksiyonların ve konfigürasyonların tanımlanmasına izin veren arayüzdür. Eğer arayüz işletim sisteminizde kullanılabilir durumdaysa /sys/kernel/config klasörünün altında usb_gadget klasörü görünür. [10] Raspbian işletim sisteminin son çekirdek sürümünü kullandığımız için configfs zaten mevcut. Bu nedenle bu usb_gadget klasörü altında RaspiJoy'a özel bir klasör açıldı. Bu klasörün açılması ile doldurulmak üzere ŞEKİL 4.1'de görülen bir USB aygıt şablonu oluşturuldu.

```
pi@raspberrypi:/sys/kernel/config/usb_gadget/RaspiJoy $ ls
bcdDevice  bDeviceClass  bDeviceSubClass  configs  idProduct  os_desc  UDC
bcdUSB     bDeviceProtocol  bMaxPacketSize0  functions  idVendor  strings
pi@raspberrypi:/sys/kernel/config/usb_gadget/RaspiJoy $
```

ŞEKİL 4.1 usb_gadget klasörü altında kendi USB aygıtımız için bir klasör oluşturduğumuzda bir aygıt tanımlayıcısı oluşturabilmek için otomatik olarak oluşan şablon

Klasörün oluşturulmasından itibaren yapılan her bir adım tek tek terminal komutu olarak yazılmak yerine ŞEKİL 4.2'de görülen betik yazıldı. Betiğin 9. Satırında İngilizce için bir karakter dizisi klasörü açıldı. Bu şablon doldurulurken yazılan değerlerin detayları 3.1 numaralı bölümde anlatıldı. ŞEKİL 4.2'deki betiğin devamında güç tüketimi konfigürasyonu için klasör oluşturuldu.

```

1  #!/bin/bash
2  cd /sys/kernel/config/usb_gadget/
3  mkdir -p RaspiJoy
4  cd RaspiJoy
5  echo 0x1d6b > idVendor # Linux Foundation
6  echo 0x0104 > idProduct # Multifunction Composite Gadget
7  echo 0x0100 > bcdDevice # v1.0.0
8  echo 0x0200 > bcdUSB # USB2
9  mkdir -p strings/0x409
10 echo "raspi0123joy" > strings/0x409/serialnumber
11 echo "RaspiJoy" > strings/0x409/manufacturer
12 echo "Joystick" > strings/0x409/product

```

ŞEKİL 4.2 Şablonun karar verilen uygun aygıt tanımlayıcısı şeklinde yazılması işlemi bir betik yardımı ile yapıldı

Daha sonra RaspiJoy’un fonksiyonel karakteristik tanımları yapıldı. Bu fonksiyonlar protokolleri, cihazların alt sınıflarını, gönderilecek bilginin boyutunu ve rapor tanımlayıcıyı içerir. Bu değerler RaspiJoy için ŞEKİL 4.3’teki şekilde yapıldı. 22, 23 ve 24. satırdaki ifade tek bir satır olarak yazılmalıdır. Raporda gösterilebilmesi için bu şekilde parçalanmıştır. O satır ŞEKİL 3.4’te gösterilen rapor tanımlayıcısının bayt bayt yazılması ile oluşmuştur.

```

16 # Add functions here
17 # Functions -> protocol & subclass & report_length & report_desc
18 mkdir -p functions/hid.usb0
19 echo 3 > functions/hid.usb0/protocol
20 echo 1 > functions/hid.usb0/subclass # boot interface subclass
21 echo 8 > functions/hid.usb0/report_length
22 echo -ne \\x05\\x01\\x09\\x04\\xa1\\x01\\xa1\\x00\\x05\\x09\\x19\\x01\\x29\\x08\\x15\\x00
23 \\x25\\x01\\x95\\x08\\x75\\x01\\x81\\x02\\x05\\x01\\x09\\x30\\x09\\x31\\x09\\x32\\x09\\x33
24 \\x15\\x00\\x25\\x7f\\x75\\x08\\x95\\x04\\x81\\x02\\xc0\\xc0 > functions/hid.usb0/report_desc

```

ŞEKİL 4.3 USB Aygıtın fonksiyonel karakteristiğini gösteren betik satırları.

Fonksiyonların tanımlanmasından sonra oluşturulan USB aygıtı

"ls /sys/class/udc > UDC" komutu ile USB aygıt kontrolü (UDC) sürücüsüne bağlanmıştır. Son olarak yazılan bu betik /usr/bin klasörünün altına yerleştirilmiş ve Raspberry’nin açılış esnasında çalışması için /etc/rc.local dosyasına ŞEKİL 4.4’te gösterildiği gibi eklendi. Bu işlemten sonra Pi Zero tekrar başlatıldığında, /dev klasörü altında hidg0 aygıt dosyasının oluştuğu görülür.

```
GNU nano 2.7.4 File: /etc/rc.local
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
    printf "My IP address is %s\n" "$_IP"
fi

/usr/bin/raspijoy_usb

exit 0
```

ŞEKİL 4.4 ŞEKİL 4.4 Oluşturulan betik dosyası Raspberry'nin başlangıcında çalışması için /etc/rc.local dosyasına “exit 0” ifadesinin üstünde kalacak şekilde betiği çalıştıracak satır eklendi.

Bu aşamanın testi için Raspberry bilgisayara takılıp bilgisayar tarafında takılı aygıtlar görüntülenmelidir. Bilgisayara takılan USB aygıtları birden çok şekilde görüntüleyebilirsiniz. En az detay ile aygıtları gösteren yöntem, terminalde “lsusb” komut satırını çalıştırmaktır. Raspberry tarafında yapılan işlemler sonucu Raspberry bilgisayara bağlandığında Raspberry'nin açılması için bir süre beklenmelidir. Bu sürenin sonunda komutu çalıştırdığınızda ŞEKİL 4.5'teki görüntü ile karşılaşacaksınız. Bu çıktıda görünen USB üretici ve ürün kimlik numaralarının karşılıklarıdır. RaspiJoy için bölüm 3.1'de verdiğimiz numaralar ışığında çıkan sonuç doğrudur yani bilgisayar artık Raspberry'yi USB aygıt olarak tanımaktadır.

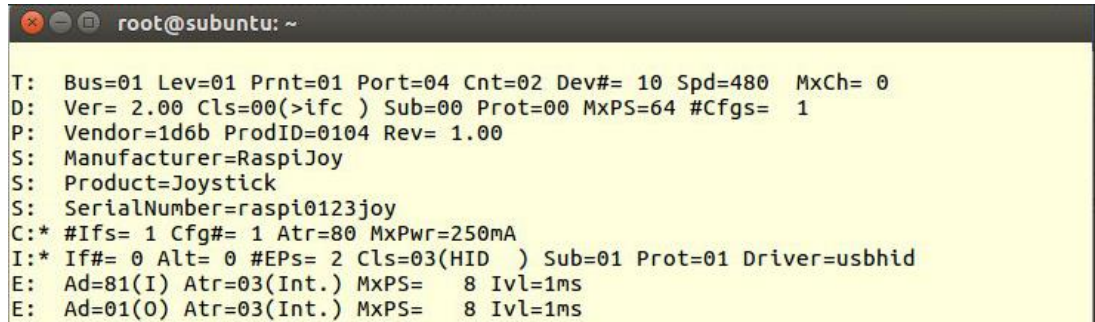
```
sevval@subuntu: ~/Desktop/FinalProject
sevval@subuntu:~/Desktop/FinalProject$
sevval@subuntu:~/Desktop/FinalProject$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 248a:8366
Bus 001 Device 010: ID 1d6b:0104 Linux Foundation Multifunction Composite Gadget
Bus 001 Device 002: ID 0bda:57de Realtek Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
sevval@subuntu:~/Desktop/FinalProject$
```

ŞEKİL 4.5 lsusb komutu bilgisayara takılı olan USB aygıtları gösteren komuttur. Listedeki Linux Foundation Multifunction Composite Gadget RaspiJoy'dur.

Önemli bir nokta olarak Raspberry Pi hala bir bilgisayar gibi çalışabilmektedir. Bir ekrana bağlayıp klavye ve fare ile her zaman kullanıldığı gibi kullanılmaya devam edilebilir. Bu aşamanın testi bu şekilde başarıyla tamamlanmıştır.

4.1.2 Bilgisayar ile İletişimin Sağlanması

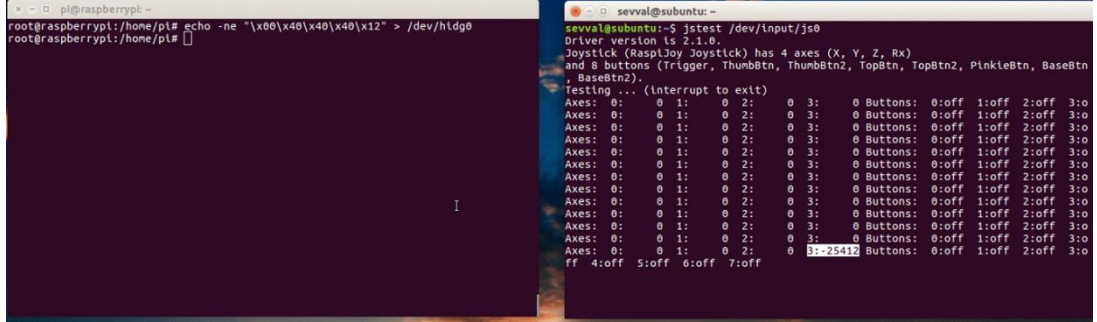
Bilgisayara takılı USB aygıtları daha fazla detay ile görüntülemenin yolu " cat /sys/kernel/debug/usb/devices" komut satırını çalıştırmaktır. RaspiJoy'un takılı olduğu bilgisayarda bu komutun ilgili çıktısı ŞEKİL 4.6'da gösterilmiştir. Bu çıktıda rapor tanımlayıcıda ayarladığımız değerleri gözlemleyebiliriz. Önemli nokta, bu aygıt için atanan sürücünün usbhid sürücüsü olmasıdır. RaspiJoy'un bilgisayar tarafında haberleşeceği sürücü usbhid sürücüsüdür. Bu sürücü Linux'un HID sınıfı aygıtlar için atadığı bir sürücüdür. Sürücü ve aygıt arasında kurulacak ilişkinin türü içinse I satırındaki #EPs değerine bakılır. Bu değer endpoint sayısını gösterir, 8 ve daha küçük değerler kesmeli şekilde okuyacağını söyler.



```
root@subuntu: ~
T: Bus=01 Lev=01 Prnt=01 Port=04 Cnt=02 Dev#= 10 Spd=480 MxCh= 0
D: Ver= 2.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS=64 #Cfgs= 1
P: Vendor=1d6b ProdID=0104 Rev= 1.00
S: Manufacturer=RaspiJoy
S: Product=Joystick
S: SerialNumber=raspi0123joy
C:* #Ifs= 1 Cfg#= 1 Atr=80 MxPwr=250mA
I:* If#= 0 Alt= 0 #EPs= 2 Cls=03(HID ) Sub=01 Prot=01 Driver=usbhid
E: Ad=81(I) Atr=03(Int.) MxPS= 8 IvL=1ms
E: Ad=01(O) Atr=03(Int.) MxPS= 8 IvL=1ms
```

ŞEKİL 4.6 Takılan aygıtın bilgisayar tarafında usbhid sürücüsü ile bağlandığının kontrolü

Veri gönderilmesinin doğru şekilde olup olmadığını kontrol edebilmek için ssh ile raspberry'e bağlanıldı. Bilgisayar terminalinde "jstest" ismi verilen kumanda kalibrasyon programı açıldı ve RaspiJoy için kararlaştırılan veri tipinde bilgiler Raspberry tarafında oluşan aygıt dosyasına yazıldı. Bilgisayardaki kalibrasyon programı ŞEKİL 4.7'deki gibi doğru değerler gösterdi. Raspberry tarafından gönderilen bilgi TABLO 3.1'de gösterilen RaspiJoy'un üreteceği 5 baytlık verinin onaltılık tabanda gösterimidir. Şekilde 4. eksen değeri değiştirilmektedir ve bilgisayar tarafında sonuç doğru okunmaktadır. Bu aşamanın testi de bu şekilde tamamlanmıştır.



```
pi@raspberrypi:~$ echo -ne '\x00\x40\x40\x12' > /dev/hidg0
root@raspberrypi:~$

seval@ubuntu:~$ jstest /dev/input/js0
Driver version is 2.1.0.
Joystick (RaspiJoy Joystick) has 4 axes (X, Y, Z, Rx)
and 8 buttons (Trigger, ThumbBtn, ThumbBtn2, TopBtn, TopBtn2, PinkieBtn, BaseBtn2).
Testing ... (Interrupt to exit)
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
Axes: 0: 0 1: 0 2: 0 3: 0 Buttons: 0:off 1:off 2:off 3:0
ff 4:off 5:off 6:off 7:off
```

ŞEKİL 4.7 Raspberry tarafından kumandanın 4. Axis değerinin değiştirilmesinin bilgisayar tarafında gözlemlenmesi

4.1.3 RaspiJoy’un Yer İstasyonuna Tanıtılması

Bilgisayarın kumandayı tanımasından sonra SITL çalıştırılarak MAVProxy programının bu kumandaya davranışı test edildi. MAVProxy bağlı olan bir kumanda görmekte, joystick modülünü yüklemekte ama kendi tanımlı olduğu joystickler ile eşleştirememektedir. Bu nedenle aktif kumanda gözükmemektedir.

RaspiJoy’un yer istasyonunda tanınabilmesi için MAVProxy python modülünün kaynak kodları içerisindeki joystick modülü klasöründeki tanımlı olan kumandaların bulunduğu klasöre RaspiJoy için de bir YAML dosyası eklendi. Kontrol kanal eşleşmesi konusunda eksenler varsayılan kanallar ile eşleştirildi. Diğer kontroller ise:

- Uçuş modları arasında seçim yapılabilen çoklu buton kontrolü 5. Kanal,
- Pervanelerin kontrollerini sağlayan buton 6. Kanal ile eşleşti.

5. Kanal genel olarak kumandalar arasında uçuş modu seçimi için kullanılan bir kanaldır. Ancak kanal 6’nın alabileceği farklı değerler vardır. Kanal 6’yı bu kontrol ile kullanmak için MAVProxy’nin içinden şu parametre ayarı yapılmalıdır:

“set param RC7_OPT 41”, 41 arm/disarm fonksiyonunun değeridir ve diğer seçenekler copter’in parametre listesinden bulunabilir. [11]

Bu aşamanın testi için SITL programı çalıştırıldıktan sonra programın içerisinde çalışmakta olan yer istasyonu MAVProxy’e kumanda yüklemek için ŞEKİL

4.8'deki gibi modül yüklenir, modülün yüklendiği ve eşleşen cihaz bulunduğu gözlenir. Ardından yüklenen kumandanın durumu kontrol edilir. MAVProxy programında RaspiJoy tanınmış ve yüklenmiş bu sayede bu aşamanın testi başarıyla sonuçlanmıştır.

```
STABILIZE>
STABILIZE>
STABILIZE> module load joystick
STABILIZE> pygame 1.9.4
Hello from the pygame community. https://www.pygame.org/contribute.html
MAVProxy.modules.mavproxy joystick: Found joystick (RaspiJoy Joystick)
MAVProxy.modules.mavproxy joystick: Using /usr/local/lib/python2.7/dist-packages
/MAVProxy/modules/mavproxy joystick/joysticks/RaspiJoy.yml ("RaspiJoy Joystick"
matches pattern "RaspiJoy Joystick")
Loaded module joystick

STABILIZE> joystick status
STABILIZE> Active joystick:
Path: /usr/local/lib/python2.7/dist-packages/MAVProxy/modules/mavproxy joystick/
joysticks/RaspiJoy.yml
Description: Support for RaspiJoy Joystick

STABILIZE> □
```

ŞEKİL 4.8 MAVProxy'de modülü yükleme komutundan sonra modül yüklenirken RaspiJoy ile eşleşilmiş ve yüklemiştir. Aktif kumandalar kontrol edildiğinde RaspiJoy'un aktif olduğu görülebilir.

4.1.4 Kullanıcı Girdilerinin Alınması

Modüllerin birbirleriyle bağlantısını sağlanmıştır. MCP3008 çipinin 4 kanalı, 4 eksen den bilgi okumak için kullanılmıştır. Raspberry'de girdilerin okunması için programlama dili olarak python kullanılmıştır ve MCP3008 çipiyle çalışmak için Adafruit MCP3008 Python kütüphanesinden yararlanılmıştır. Tasarıma istenildiği kadar buton hatta 3.bir analog kumanda kolu eklenebilir. Bu tamamen kişinin yapmak istediği kontrollere bağlıdır.

MCP3008 çipinden okunan değer 10 bit bir değerdir. Ancak RaspiJoy eksen değerlerini 8 bitte göndermekte ve bunun en anlamlı biti (MSB) de işaret bitidir. Sonuç olarak 10 bit olarak alınan değerlerin 7 bite döndürülmesi için ŞEKİL 4.9'deki gibi okunan değer 8'e bölünmüştür.

```
50
51 def readChannel(channel):
52     # read SPI data from MCP3008 given channel and divide by 2^3 because
53     # 7 byte data will send except sign bit but the readed data is 10 byte
54     return mcp.read_adc(channel)/8
55
```

ŞEKİL 4.9 Kanaldan okunan 10 bitlik değer 7 bite çevrilebilmek için 8'e bölünmüştür

Her buton bir bite karşılık geldiği için temsil ettikleri bir basamak değerleri vardır. Bu buton basamak değerleri TABLO 4.1’de gösterilmiştir. Butona basılıp bırakıldığında, butonların bağlı olduğu GPIO pininden yüksek girdi geldiği zaman, o butonun anlık değeri 1 olur. Ancak hemen bir sonraki gönderilecek veride, bu anlık değer 0 olarak ayarlanır. Her butonun anlık olarak aldığı 1 ya da 0 değeri, bu butonların basamak değerleri ile çarpılır ve sonra tüm butonlar için toplanır. Bu yapılan işlem ŞEKİL 4.10’da gösterilmiştir.

TABLO 4.1 Butonlar ve basamak değerleri

Buton numarası	4. buton	3. buton	2. buton	1. buton
Basamak değeri	8	4	2	1

Pervaneleri kontrol eden buton 4 diğerlerinden farklı çalışır. Çünkü pervanelerin dönmesini sağlayan bu buton, buton bırakıldığı anda dönen pervaneleri durdurur. Halbuki bir daha basılana kadar pervanelerin dönmesi gerekmektedir. Bu nedenle 4 butonun anlık değeri saklanır ve bir daha basılana kadar saklanan anlık değer ile işlem yapılır. Bu işlem de ŞEKİL 4.12’de gösterilmektedir.

```

67         buttonInfo = 0;
68         # Control the buttons
69         if not GPIO.input(23):
70             buttonInfo += val_but1
71         if not GPIO.input(24):
72             buttonInfo += val_but2
73         if not GPIO.input(25):
74             buttonInfo += val_but3
75         if not GPIO.input(12):
76             # If not pressed button4, press
77             # If pressed, release
78             if not but4Pressed:
79                 but4Pressed = True
80             else:
81                 but4Pressed = False
82             buttonInfo += val_but4
83
84         # If button4 already pressed, add the value of button4
85         if but4Pressed:
86             buttonInfo += val_but4

```

ŞEKİL 4.10 Butonların çalışma prensibi

Yazılan bu python kodu, bölüm 4.1.1’deki betiğe yapıldığı gibi /etc/rc.local dosyasına çalıştırma komutu ile yazıldı. Bu sayede Pi’nin açılır açılmaz kullanıcı


girdilerini okumaya ve yazmaya başlaması sağlandı. Bu aşamanın testi için Pi tarafında veriler gönderilirken bilgisayar tarafında daha önce de kullanılan “jstest” uygulaması ile gönderilen verilerin doğruluğu kontrol edildi.

4.2 Başarı Kriterlerinin Kontrolü

Projede iki adet başarı kriteri bulunmaktadır:

1. Kullanıcının RaspiJoy üzerinde sağladığı kontrol sinyallerinin simülasyon üzerinde gerçek zamanlı ve doğru bir şekilde gösterilmesi
2. RaspiJoy’un gecikme süresinin 50 milisaniyeden az olması

Kullanıcı girdilerinin bilgisayar tarafında doğru bir şekilde algılandığı bölüm 4.1.3’te test edilmişti. Simülasyonda doğru çalıştığının tespiti için, simülasyon programları çalıştırıldı, MAVProxy’e joystick modülü eklendi ve RaspiJoy aktif hale getirildi. Butonların yaptıkları değişiklikler MAVProxy’nin konsol olarak sunduğu arayüzden, eksenlerin yaptığı değişiklikler ise Gazebo’dan üç boyutlu olarak test edildi. Butonlara sırası ile basıldığında ŞEKİL 4.11’te görülebileceği gibi butonlar gerçekleştirmesi gereken fonksiyonları gerçekleştirdiler. Land modu iniş modudur, o modda pervaneleri çalıştırmaya izin vermediği için şekildeki hata alındı ve 2. Buton ile Loiter moduna geçildi.

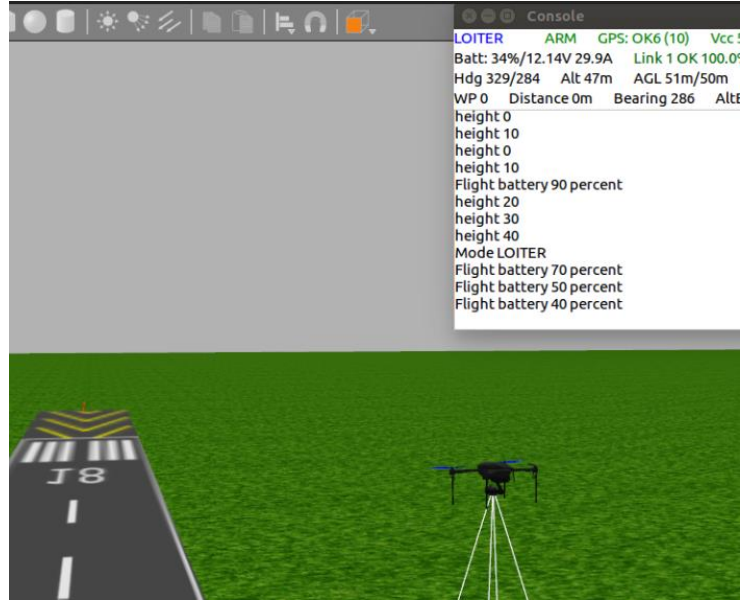


```
LOITER ARM GPS: OK6 (10) Vcc 5.00 Radio: - INS MAG AS RNG AHRS EKF LOG FEN
Batt: 98%/12.54V 5.0A Link 1 OK 100.0% (11589 pkts, 0 lost, 0.00s delay)
Hdg 0/271 Alt 0m AGL 0m/0m AirSpeed 0m/s GPSSpeed 0m/s Thr 0 Roll 0 Pitch 0 Wind -/-
WP 0 Distance 0m Bearing 0 AltError 0H AspdError 0.0H FlightTime 0:18 ETR 0:00
APM: EKF2 IMU0 Origin set to GPS
APM: EKF2 IMU1 Origin set to GPS
APM: EKF2 IMU0 is using GPS
APM: EKF2 IMU1 is using GPS
Mode STABILIZE
Mode LOITER
Mode LAND
APM: PreArm: Mode not armable
Mode LOITER
APM: Arming motors
ARMED
Arming checks disabled
```

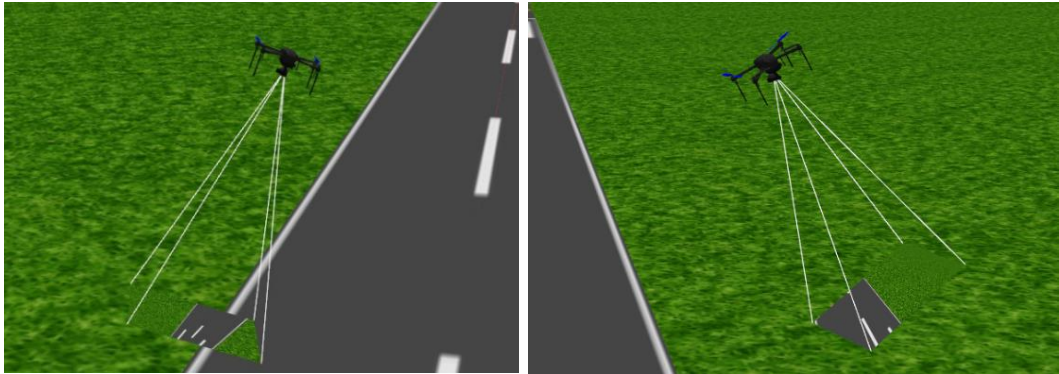
ŞEKİL 4.11 Buton hareketlerinin MAVProxy konsolunda gerçekleşmesi

Kumanda ile eksen kontrolleri için ise önce kalkış için gaz kontrolünü sağlayan sağ kolun y eksenini yukarı kaldırıldı. Bu hareket sonucunda ŞEKİL 4.12’de görülebileceği gibi konsolda İHA’nın yüksekliğinin artmıştır. Sonra bu koldaki x eksenini

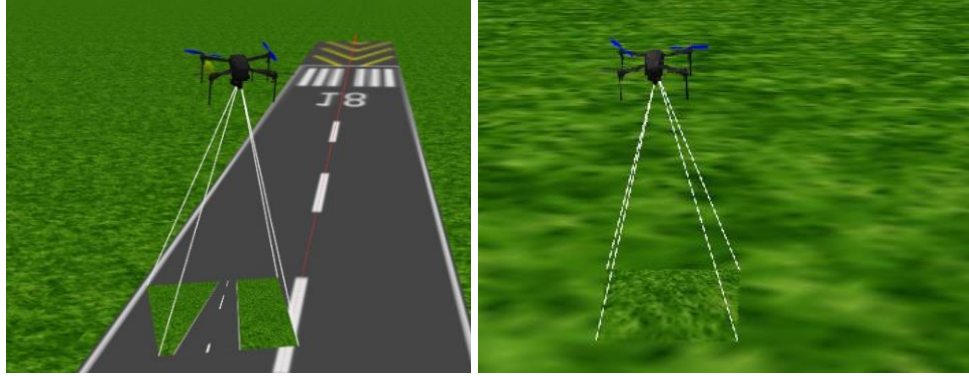
değiştirilerek İHA'nın saat yönünde ve saat yönünün tersine dönmesi hareketleri gözlemlendi. Mavi pervaneli kısım İHA'nın önü olmak üzere, ŞEKİL 4.13'te de sağa/sola yatış hareketleri, ŞEKİL 4.14'te ileri/geri hareketleri doğru bir şekilde gözlemlendi. Bu şekilde 1. Başarı kriteri sağlanmış oldu.



ŞEKİL 4.12 Sol kumanda kolu gaz değerinin ölçülmesi



ŞEKİL 4.13 Sağ kumanda kolu sağ ve sol hareketin gözlemlenmesi



ŞEKİL 4.14 Sağ kumanda kolu ileri ve geri hareketin gözlemlenmesi

2. başarı kriterinin belirlenmesinde, piyasadaki birçok kumanda markasındaki gecikmeleri ölçen ve sonuçları sunan bir yazı rol oynadı.[12] Yazıya göre en fazla 42 milisaniyelik bir gecikmeye sebep olduğu için başarı kriterinde gecikmenin 50 milisaniyeden az olması planlanmıştır.

MAVProxy’de kumandadaki kontrol tanımlarının test edilebilmesi adına kumandada yapılan hareketin hangi eksenin ya da hangi butonun kontrolünü sağladığını gösteren findJoy isminde bir program bulunmakta. Bu programın kaynak kodunda, sinyal ulaştığında zaman değerini ekrana bastıran değişiklik yapıldı. RaspiJoy’un da kullanıcı inputlarını alıp bilgisayara veri gönderen python programına aynı değişiklik yapıldı. ŞEKİL 4.15 ve 4.16’da üstteki terminalde findJoy programı alttakinde ise Pi’deki program çalışmaktadır. Aradaki sürenin hesaplanması için üst programdaki değerler alt programdan çıkarılmalıdır. Örneğin ŞEKİL 4.15’te ilk giden sinyal için saniye değerleri birbirinden çıkarıldığında 0,025070 saniye elde edilmektedir bu da 25 milisaniyeye denk gelmektedir. 4.16’daki buton örneğinde ise bu değer 4 milisaniyeye denk gelir.

```
sevval@ubuntu: ~  
Joystick: RaspiJoy Joystick  
Axes: 4  
Buttons: 8  
Hats: 0  
Balls: 0  
  
Axes control 1. event at time 2019-01-04 10:59:21.077221  
Axis 0  
Axes control 2. event at time 2019-01-04 10:59:21.077274  
Axis 1  
Axes control 3. event at time 2019-01-04 10:59:21.511353  
Axis 1  
Axes control 4. event at time 2019-01-04 10:59:21.511401  
Axis 1  
Axes control 5. event at time 2019-01-04 10:59:21.642680  
Axis 1  
  
pi@raspberrypi: ~  
1. user input is sending at time: 2019-01-04 10:59:21.052151  
2. user input is sending at time: 2019-01-04 10:59:21.488894  
3. user input is sending at time: 2019-01-04 10:59:21.491825  
4. user input is sending at time: 2019-01-04 10:59:21.619829  
5. user input is sending at time: 2019-01-04 10:59:21.624216  
6. user input is sending at time: 2019-01-04 10:59:21.629108  
7. user input is sending at time: 2019-01-04 10:59:21.633250  
8. user input is sending at time: 2019-01-04 10:59:21.638388  
9. user input is sending at time: 2019-01-04 10:59:21.642569
```

ŞEKİL 4.15 Eksenlerin gecikmelerinin kontrolü

```
sevval@ubuntu: ~  
Button control 97. event at time 2019-01-04 10:59:22.028546  
Button 4  
  
pi@raspberrypi: ~  
97. user input is sending at time: 2019-01-04 10:59:22.073267
```

ŞEKİL 4.16 Örnek bir buton gecikmesinin kontrolü

İki programda da hataya yer vermemek için Koordine Evrensel Zaman(UTC) değerleri ile zaman kontrolü yapılmıştır. Farklı zamanlarda yapılan birkaç test sonucu olarak ortalama gecikme süresi 26 milisaniyedir. Bu da 2. başarı kriterinin sağlandığını gösterir.

5 SONUÇ

Bu projede Raspberry Pi Zero kullanılarak SITL uçuş simülasyonu kontrolü için bir kumanda yapılmış ve sonuç olarak kullanıcıdan alınan kumanda kontrol sinyalleri ile uyumlu gerçek zamanlı bir görüntü Gazebo 3 boyutlu robotik simülasyonda gözlenmiştir. Bu çalışma referans alınarak kullanıcı kendi isteği kadar analog kumanda kolu ve butonla kendi kumanda tasarımını yapabilecektir. Proje Raspberry Pi Zero'nun USB aygıt olarak kullanılabileceği örnek fikirlerden biridir.

Bu kumanda, İHA uçurma ve yarışırma konusunda merakı olan tecrübesiz kullanıcıların öğrenme sürecinde sebep olacağı kazaları en aza indirmeyi hedefler. Bu sayede süreç boyunca maddi ve manevi zararlar azaltılıp, kişinin alışma süreci kısaltılabilir.

Tasarlanan kumandanın sadece simülasyon programı için değil, gerçek bir İHA'yı uçurması bu çalışmanın devamı niteliğinde bir proje olabilir.

6 KAYNAKLAR

- [1] Wood, Aaron Markus. *Flying with a game controller: RC through an Intel Edison*. Diss. 2015.
- [2] Mulholland, Andrew, *Raspberry Pi Zero – Programming Over USB* [online], <https://pi.gbaman.info/?p=699> [Ziyaret Tarihi: 8 Ekim 2018]
- [3] *Getting a Vendor ID* [online], <https://www.usb.org/getting-vendor-id>, [Ziyaret Tarihi: 3 Ocak 2019]
- [4] *List of USB ID'S* [online], <http://www.linux-usb.org/usb.ids>, [Ziyaret Tarihi: 25 Kasım 2018]
- [5] *Microsoft Word - HID1_11.doc* [online], https://www.usb.org/sites/default/files/documents/hid1_11.pdf. [Ziyaret Tarihi: 23 Kasım 2018]
- [6] *Human interface device* [online], https://en.wikipedia.org/wiki/Human_interface_device [Ziyaret Tarihi: 25 Kasım 2018]
- [7] USB Implementers, *HID Descriptor Tool* [online], <https://www.usb.org/document-library/hid-descriptor-tool> [Ziyaret Tarihi: 23 Kasım 2018]
- [8] *Raspaberry Pi Zero USB/ETHERNET Gadget Tutorial* [online] [Ziyaret Tarihi: 3 Ocak 2019]
- [9] *2.7V 4-Channel/8-Channel 10-Bit A/D Converters with SPI Serial Interface* [online], <https://cdn-shop.adafruit.com/datasheets/MCP3008.pdf> [Ziyaret Tarihi: 5 Aralık 2018]
- [10] Matt Porter Linaro, *Kernel USB Gadget Configfs Interface* [online], https://events.static.linuxfound.org/sites/events/files/slides/USB%20Gadget%20Configfs%20API_0.pdf
- [11] *Complete Parameter List* [online], <http://ardupilot.org/copter/docs/parameters.html>
- [12] *Arcade Stick Input Lag – Results* [online], <http://www.teyah.net/sticklag/results.html>