



**GÜMÜŞHANE ÜNİVERSİTESİ  
MÜHENDİSLİK ve DOĞA BİLİMLERİ FAKÜLTESİ**

**ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ**

**GPS ile YAYA TAKİBİ**

**TASARIM TEZİ**

**OMAR MANSOUR**

**HAZİRAN 2023  
GÜMÜŞHANE**

GÜMÜŞHANE ÜNİVERSİTESİ  
MÜHENDİSLİK ve DOĞA BİLİMLERİ FAKÜLTESİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ



Gümüşhane 2023

GÜMÜŞHANE ÜNİVERSİTESİ  
MÜHENDİSLİK ve DOĞA BİLİMLERİ FAKÜLTESİ  
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ

1907211601 - OMAR MANSOUR

tarafından hazırlanan

**GPS ile YAYA TAKİBİ**

Başlıklı tasarım çalışmasının teslim edilmesi uygundur ( ) uygun değildir ( ).

Tarih : 15/06/2023

Danışman : Dr. Öğr. Üyesi M. Taha KÖROĞLU .....  
.....

Başlıklı tasarım çalışması jürimizce değerlendirilmiştir.

Tarih : 15/06/2023

Danışman : Dr. Öğr. Üyesi M. Taha KÖROĞLU .....  
.....

Üye : Dr. Öğr. Üyesi Gökhan ÇETİN .....  
.....

Üye : Prof.Dr. Adı SOYADI 2 .....  
.....

Doç. Dr. Mustafa Engin BAŞOĞLU  
Bölüm Başkanı

## **ÖNSÖZ**

Tasarım projesinde bana yol göstermeye çalışan danışman hocama teşekkür ederim.

Omar MANSOUR

Gümüşhane, 2023

## **TEZ ETİK BEYANNAMESİ**

Lisans Tezi olarak sunduğum "GPS ile Yaya Takibi" başlıklı bu çalışmayı baştan sona kadar danışmanım Dr. Öğr. Üyesi M. Taha KÖROĞLU'nın sorumluluğunda tamamladığımı, verileri kendim topladığımı, deney ve analizleri kendim yaptığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynaklarda eksiksiz olarak gösterdiğim, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davranışımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim.

15/06/2023

Omar MANSOUR

## İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ . . . . .	III
TEZ ETİK BEYANNAMESİ . . . . .	IV
İÇİNDEKİLER . . . . .	V
ÖZET . . . . .	VI
SUMMARY . . . . .	VII
ŞEKİLLER DİZİNİ . . . . .	VIII
TABLOLAR DİZİNİ . . . . .	IX
SEMBOLLER . . . . .	X
1. GENEL BİLGİLER . . . . .	1
1.1. Mutlak Konumlama Sistemleri . . . . .	1
1.2. Göreceli Konumlama Sistemleri . . . . .	3
2. YAPILAN ÇALIŞMALAR . . . . .	5
2.1. Küresel Konumlama Sistemi (GPS) . . . . .	5
2.2. SparkFun GPS Modül Kit (EM-506 Alıcı) . . . . .	5
2.2.1. Default Kod Test - USB Kablo . . . . .	7
2.2.2. GPS Koordinat Dönüşümleri . . . . .	8
2.2.2.1. Ondalık Sayıdan Derece-Dakika-Saniye Formatına Dönüşüm . . . . .	8
2.2.3. Default Kod Test - XBee kablosuz RF modül . . . . .	9
2.2.4. Kendi Kodumuz ile Veri Toplama . . . . .	10
2.3. Cep Telefonu Dâhili GPS . . . . .	12
2.3.1. MATLAB Uygulaması . . . . .	14
2.3.2. Hem Hârici Hem Dâhili GPS ile Veri Toplanan Deneyler . . . . .	16
3. BULGULAR VE TARTIŞMALAR . . . . .	18
4. KAYNAKLAR . . . . .	19
5. EKLER . . . . .	21
ÖZGEÇMİŞ . . . . .	25

## Tasarım Tezi

## ÖZET

### GPS ile YAYA TAKİBİ

Gümüşhane Üniversitesi  
Mühendislik ve Doğa Bilimleri Fakültesi  
Elektrik-Elektronik Mühendisliği  
Danışman: Dr. Öğr. Üyesi M. Taha KÖROĞLU  
2023, 36 Sayfa

Son otuz yılda bilgisayar işlemci hızlarında ve algılayıcı teknolojilerinde yaşanan gelişmelerle Konum Tabanlı Hizmetler (KTH) artık herkes tarafından hem de ücretsiz olarak kullanılır hâle gelmiş durumda. Özellikle navigasyon ve konum takibi, günümüzde KTH uygulamalarının en vazgeçilmez olanlarından. Takip edilen yaya veya araç eğer dış ortamdaysa, tercih edilen standart teknoloji genellikle Küresel Konumlama Sistemi (GPS) olmaktadır. Bu çalışmada hobi devresi sayılabilen Arduino uyumlu hârıcı bir GPS alıcısından ve aynı zamanda akıllı bir cep telefonunun dâhili GPS alıcısından C++, MATLAB ve Python aracılığıyla yaya koordinat bilgileri (i.e., enlem-boylam) toplanmış, veriler hârıcı alıcı için gerçek-zamanda, dâhili alıcı için ise yürüyüş bittikten sonra görselleştirilmiştir. İlerleyen çalışmalarında diferansiyel GPS (dGPS) veya GPS RTK (Real-Time-Kinematics) teknolojilerinin kullanılmasıyla konum tahmin hatalarının minimize edilmesi hedeflenmektedir. Bu şekilde, yüksek hassasiyet ve daha yüksek yenileme frekansında çalışan GPS tabanlı bir konumlama sistemi kullanılarak (GPS verisi ile eş-zamanlı) yakalanan ataletsel ölçüm birimi (AÖB) sinyallerinden oluşan (mutlak konum verisi etiketlenmiş) bir veri kümesi (Ing. dataset) oluşturulabilir. Elde edilen bu veri kümesi ile yaya, araç veya dronlar için ataletsel odometri görevi görecektir. Tahmin algoritmaları (e.g., hata durum Kalman Filtresi, derin öğrenme modelleri) geliştirilebilir. Böyle altyapı bağımlılığı olmayan (i.e., kendi-kendine çözüm üretebilen) bir sistem, GPS'in sorunlu olduğu (e.g., ormanlık ve dağlık alanlar) veya hiç çalışmadığı (e.g., tüneller ve bina içleri) ortamlarda GPS'e alternatif olarak ve GPS'in çalıştığı açık alanlarda konum netliğini artıran bir biçimde (yâni GPS ile entegre edilerek) başarıyla kullanılabilir.

**Anahtar Kelimeler:** GPS, AÖB, konum tahmini, navigasyon, ataletsel odometri, ataletsel navigasyon sistemi, Kalman Filtresi, derin öğrenme, algılayıcı birleştirme, Arduino, C++, MATLAB, Python.

# Capstone Project

## ABSTRACT

### PEDESTRIAN TRACKING with GPS

Gumushane University  
Faculty of Engineering and Natural Sciences  
Electrical & Electronics Engineering  
Supervisor: Dr. Öğr. Üyesi M. Taha KÖROĞLU  
2023, 36 Pages

Due to advancements in computer processor speeds and sensor technologies over the past three decades, Location Based Services (LBS) have become widely accessible to the general public, often at no cost. Among the various LBS applications available today, navigation and location tracking are particularly essential. When it comes to outdoor tracking of pedestrians or vehicles, the commonly preferred technology is the Global Positioning System (GPS). In this study, pedestrian coordinate information, specifically latitude and longitude, was obtained from both an Arduino compatible external GPS receiver, which can be considered a hobbyist circuit, and the internal GPS receiver of a smartphone. The data collection process involved the use of C++, MATLAB, and Python for visualization, with the external GPS receiver providing real-time visualization and the internal GPS receiver data being visualized post-walk. In future research, positioning errors can be minimized by incorporating technologies such as differential GPS (dGPS) or GPS RTK (Real-Time Kinematics). This would enable the creation of an annotated dataset that includes inertial measurement unit (IMU) signals captured synchronously with higher precision and refresh frequency GPS data. Leveraging this new dataset, prediction algorithms, such as error state Kalman Filter and deep learning models, can be developed to serve as inertial odometry solutions for pedestrians, vehicles, or drones. Such self-contained systems offer an alternative to GPS in challenging environments like forests or mountainous areas, as well as in locations where GPS signals are unavailable, such as tunnels and indoor spaces. Additionally, they can be effectively utilized to enhance the accuracy of GPS solutions in open areas through GPS/IMU integration.

**Keywords:** GPS, IMU, position estimation, navigation, inertial odometry, inertial navigation system, Kalman Filter, deep learning, sensor fusion, Arduino, C++, MATLAB, Python.

## ŞEKİLLER

	<u>Sayfa No</u>
Şekil 1. Küresel Konumlama Sistemi (GPS) ile lokalizasyon. . . . .	1
Şekil 2. Yerel Konumlama Sistemi (LPS) ile lokalizasyon. . . . .	2
Şekil 3. Parmak izi (fingerprinting) metodu ile lokalizasyon. . . . .	3
Şekil 4. SparkFun EM-506 GPS modül kit. . . . .	6
Şekil 5. (a) Lehimlenmiş SF GPS modül kit ve Arduino Uno. (b) SF GPS modül shield'in Arduino Uno'ya takılmış hâli. . . . .	6
Şekil 6. SparkFun GPS Modülü testi (USB kablo ile bağlantı). . . . .	7
Şekil 7. SparkFun GPS Modülü testi (MATLAB ve Python seri port veri okuma). . . . .	8
Şekil 8. Dereceden dakikaya ve dakikadan saniyeye dönüşüm. . . . .	9
Şekil 9. Arduino, SparkFun GPS ve XBee kablosuz modüller ve Li-Po batarya. . . . .	10
Şekil 10. SparkFun GPS ve XBee kablosuz modüllerin Arduino Uno'ya bağlanması. . . . .	10
Şekil 11. SparkFun GPS Modülü kablosuz haberleşme testi (MATLAB veri okuma). . . . .	11
Şekil 12. SparkFun GPS - stadyum yürüyüş testi. . . . .	12
Şekil 13. SparkFun GPS - mühendislik binası testi. . . . .	13
Şekil 14. Google Play Store'da (Android) MATLAB uygulaması aratma sonucu. . . . .	14
Şekil 15. (a) MATLAB uygulaması ana sayfa, (b) Sensör seçimleri 1, (c) Sensör seçimleri 2. . . . .	15
Şekil 16. (a) MATLAB uygulaması algılayıcı yenileme frekansı seçimi. . . . .	15
Şekil 17. Cep telefonu dâhili GPS - stadyum yarışına testi. . . . .	16
Şekil 18. Hârıcı ve dâhili GPS - mühendislik binası testi. . . . .	17
Şekil 19. Hârıcı ve dâhili GPS - şehir merkezi testi. . . . .	17

## TABLALAR

Sayfa No

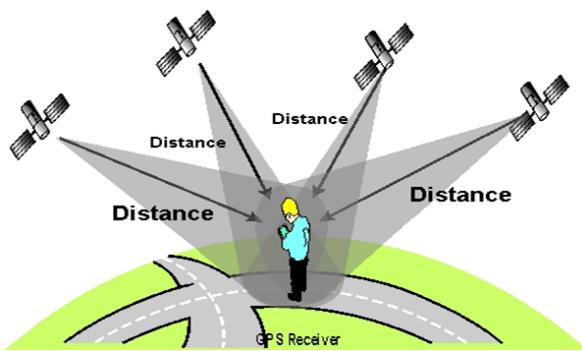
## **SEMBOLLER**

<b>KTH</b>	: Konum Tabanlı Hizmetler
<b>LBS</b>	: Location Based Services
<b>GPS</b>	: Global Positioning System
<b>GNSS</b>	: Global Navigation Satellite Systems
<b>LPS</b>	: Local Positioning System
<b>AÖB</b>	: Ataletsel Ölçüm Birimi
<b>IMU</b>	: Inertial Measurement Unit
<b>INS</b>	: Inertial Navigation System
<b>SLAM</b>	: Simultaneous Localization and Mapping

## 1. GENEL BİLGİLER

### 1.1. Mutlak Konumlama Sistemleri

Algılama (sensör) teknolojilerindeki ilerleme, bilgisayar işlemci hızlarında artış ve bu alanlarda kullanılan donanımların boyutlarının iyice küçülmesiyle beraber Konum Tabanlı Hizmetler (Location Based Services - LBS) hayatlarımızın değişilmez bir parçası hâline geldi. Özellikle navigasyon ve konum takibi, günümüzde global bir köye dönen dünyaımızda hem yayalar hem taşıtlar için neredeyse zorunluluk oldu. Kullanıcıların dış mekanlarda bulunduğu durumlarda konum bilgisini elde etmenin standart yolu Küresel Konumlama Sistemi (Global Positioning System- GPS)<sup>1</sup> kullanmaktadır.



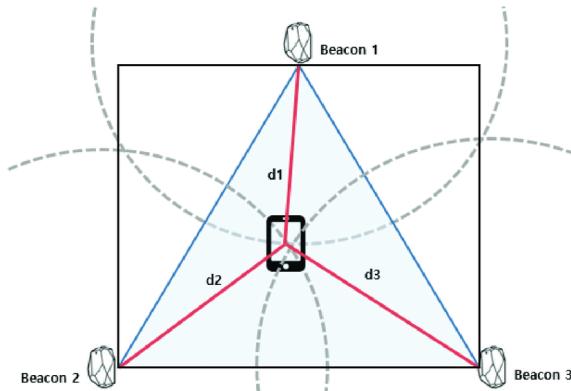
Şekil 1. Küresel Konumlama Sistemi (GPS) ile lokalizasyon.

Açık havada en az üç uydu sinyaline erişilebildiği durumlarda alıcının iki boyutlu koordinatlarını (enlem (latitude) ve boylam (longitude)), Şekil 1'de görüldüğü gibi en az dört uydu yakalandığı durumda ise alıcının üç boyutlu koordinatlarını (enlem-boylam ve yükseklik (altitude) bilgilerini) elde edebilen GPS teknolojisi, bu hesaplarda trilaterasyon (trilateration) denilen bir teknik kullanmaktadır. Yakalanan uydu sinyali sayısı arttıkça elde edilen pozisyon tahmini hatasının azaldığı GPS çözümünde, üç uydudan az uydu yakalanması durumunda kesin bir pozisyon elde edilemez.

Küresel bir kapsama alanı ve uzun süreli bir çözüm sağlamasına rağmen, özellikle yüksek binaların olduğu alanlarda sinyallerin yansıması sonucu oluşan çoklu yol (mul-

<sup>1</sup>Genel ismi Küresel Navigasyon Uydu Sistemleri (Global Navigation Satellite Systems - GNSS) olan bu teknolojide GPS, GLONASS, Galileo gibi isimleri uydulara sahip ülkeler tarafından belirlenen sistemler vardır. Mesela GPS ABD'nin, GLONASS Rusya'nın, Galileo ise Avrupa Birliği'nin uydu sistemlerine verilen adlardır. Bahsi geçen sistemlerin dışında dünyanın gelişmiş ülkelerinin de (e.g., Çin) uydu sistemleri vardır.

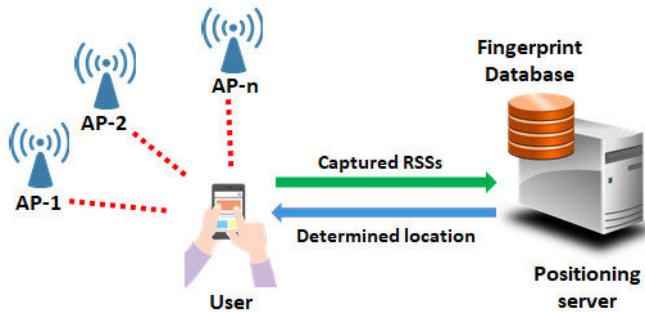
tipath), ormanlık alanlar gibi yerlerde gözlemlenen sinyal zayıflama (attenuation) ve iç mekanlarda ve tünelerde karşılaşılan sinyal engellenmesi (blockage) gibi problemlerden dolayı GPS teknolojisi yetersiz kalmaktadır. Bu gibi durumlarda ortama uyduların yerini alacak sinyal vericiler (beacon) yerleştirilirse, Şekil 2'de görüldüğü gibi GPS benzeri bir Yerel Konumlama Sistemi (Local Positioning System - LPS) kurulabilir. Burada sinyal vericilerden gönderilen Radyo Frekansı (Radio Frequency - RF) sinyali olarak genellikle Wireless-Fidelity (Wi-Fi), UltraWide-Band (UWB), Bluetooth Low Energy (BLE) kullanılırken, alıcı-vericilerin geometrisinden veya sinyallerin verici ile alıcı arasında harcadığı yolculuk zamanından (e.g., time of flight – ToF, time difference of arrival - TDoA) faydalananarak Triangulation, Trilateration veya Multilateration isimli metodlarla pozisyon tahmini yapılmaktadır [1].



Şekil 2. Yerel Konumlama Sistemi (LPS) ile lokalizasyon.

Yukarıda sayılan metodlara ek olarak, kullanıcı tarafından alınan sinyal kuvveti (received signal strength - RSS) kullanılarak yol kaybı modelleri (path loss models) oluşturulup konumlama yapılabilir. Şu ana kadar ismi geçen tekniklerin hepsi Non-Line-of-Sight (NLoS) denilen (verici ile alıcı arasında yer alan engellerden dolayı görüş kaybı olarak tercüme edilebilecek) problemden mustariptirler. Bu durumun gözlentiği ortamlarda Şekil 3'te gösterilen parmak izi (Fingerprinting) metodu kullanılabilir. Konumlama yapılacak mekânda (konumlama yapmadan önce mekâni tarayarak) RSS değerlerinden bir veri tabanı oluşturulan Fingerprinting tekniğinde, gerçek zamanda ölçülen RSS değerleri oluşturulmuş veri tabanıyla kıyaslanır (mesela bir makine öğrenmesi metoduyla [2]) ve böylece konum tahmini yapılır. Pozisyon tahminlerinin kesinliği yapılan RSS ölçüm sayısıyla doğru orantılı olacağından dolayı çok ölçüm pozisyon tahmini kesinliğini artırır, ancak bu oldukça fazla zaman alacağından bu işi zahmetli hale

getirir. Buna ek olarak, ortamda yapılan değişiklerde, veri tabanı güncellenmesi yapılması gerektiğinden Fingerprinting pratik bir yöntem değildir. Veri tabanı oluştururken SLAM, inter/ekstrapolasyon veya crowdsourcing kullanılarak vakitten ve emekten taraflar edilebilir ama aynı performans beklenmemelidir [3].



Şekil 3. Parmak izi (fingerprinting) metodu ile lokalizasyon.

Fingerprinting metodundan daha düşük performanslı ama daha pratik başka bir konumlama tekniği ise yakınlık tabanlı lokalizasyondur (proximity-based localization). Bu metot çoğunlukla Radio Frequency IDentification (RFID) ve bazen BLE teknolojilerinden faydalananmaktadır. Özellikle pasif RFID etiket (tag) kullanan çözümler, geniş iç mekânlarda ucuza konumlama yapabilme olanağı sağlamaktadır [4].

## 1.2. Göreceli Konumlama Sistemleri

Bahsi geçen tüm mutlak konumlama sistemleri pozisyon tahmini yaparken geçmiş tahminlerden bağımsız olduğundan uzun soluklu çözümler sağlayabilmektedir. Buna rağmen altyapı (infrastructure) kurulumu ve bakımı pahalı ve meşakkatli olduğundan, altyapı gerektirmeyen (infrastructure-free) daha ucuz ve pratik çözümler tercih edilmektedir. Dış sinyallere bağlı olmayan bu sistemler, başlangıç koşulları sağlandığında kendi kendilerine konum bilgisini hesaplayabilmektedirler. En çok kullanılan göreceli konumlama sistemi kamera geometrisinden faydalananmaktadır: Eş-zamanlı haritalama ve konumlama (Simultaneous Localization and Mapping - SLAM) olarak bilinen robotik bilim dalında mono, stereo ve RGB-D kameralarla gerçek-zamanda oldukça yüksek performansla pozisyon tahmini yapılabilmektedir [5]. Otonom araba ve robot uygulamalarında SLAM için tercih edilen bir başka popüler teknoloji, laser sensörlerle ortamı görünmez ışıkla tarayarak çalışan LIDAR'dır [6]. Kamera tabanlı SLAM'e göre daha

pahali olmasına rağmen, daha yüksek performans sağlayan LIDAR tabanlı SLAM [7] aynı zamanda mahremiyet konusunda da kullanıcıları memnun etmektedir [8].

Altyapı gerektirmeyen ve kendi kendine pozisyon hesaplayabilen bir başka göreceli konumlama sistemi atâlet sensörleri ile kurulabilir. İvmemetre (accelerometer) ve jiroskopdan (gyroscope) meydana gelen Atâletsel Ölçüm Birimi (AÖB) veya daha yaygın olan İngilizcesi ile Inertial Measurement Unit (IMU), son yirmi senede Mikro-Elektron-Mekanik-Sistemler (Micro-Electro-Mechanical-Systems - MEMS) teknolojisindeki gelişmelerle son derece yaygın hâle geldi. Hafif, ucuz, küçük ebatlı, düşük güç gereksinimli ve giyilebilir olan MEMS IMU, hareketli uygulamalarda (e.g., motion capture - MoCap) kendisinden çok daha pahalı olan alternatif kamera takip sistemlerine (e.g., VICON) tercih edilir oldu. Ayrıca tespit edilemeyecek şekilde pasif olması<sup>2</sup> ve kullanıcı mahremiyetine tehdit oluşturmaması IMU teknolojisinin ekstra avantajlarından. Bütün bu olumlu özelliklerine rağmen, tüketici sınıfı IMU (consumer grade IMU) verisinde gözlenen zamanla-değişen (time-variant) sensör yanılığı (bias) ve ölçek faktörü (scale factor) parametrelerinin tahminleri çok zor ve sinyallerde gözlemlenen sensör gürültüsü (noise) ise çok şiddetli. Bu problemlere ek olarak, bir de ölü-hesaplama (dead-reckoning - DR) metodıyla elde edilen Atâletsel Navigasyon Sistemi (Inertial Navigation System - INS) çözümünde biriken hatalar, "zamanla kayma" diye tercüme edebileceğimiz "drift" fenomenine dönüşerek navigasyon çözümünü çok kısa bir zaman için bile işe yaramaz hâle getirmekte. Bu sorunu engellemek için INS hatalarının büyümeyesine izin vermeden periyodik bir şekilde en az bir mutlak (e.g., WiFi [9]) veya göreceli (e.g., camera [10]) bir konumlama sistemini tamamlayıcı (complementary) olarak kullanan bir sensör füzyonu algoritması (e.g., Kalman Filtresi) ile konum tahmini güdüncellenebilir.

---

<sup>2</sup> Askeri uygulamalar için ideal.

## **2. YAPILAN ÇALIŞMALAR**

### **2.1. Küresel Konumlama Sistemi (GPS)**

Yayanın konum takibi açık alanda yapılıyorsa kullanılan ilk yol Küresel Navigasyon Uydu Sistemleri (Global Navigation Satellite Systems - GNSS) teknolojisi dir. Bina içlerine girildiğinde veya uydu sinyallerinin bloke olduğu veya zayıfladığı ortamlara (e.g., tünel, ormanlık alanlar) geçiş yapıldığında radyo sinyalleri kullanılarak (e.g., UWB) lokalizasyon devam ettirilebilir. Ama bunun için hem zaman hem işçilik hem de büyük masraflar gerektiren altyapı (infrastructure) kurulması gereklidir. Daha ucuz ve pratik bir yöntem olarak göreceli konumlama sistemleri kullanılabilir (e.g., kamera, LIDAR, SONAR veya RADAR kullanılarak SLAM, IMU ile ölü-hesaplama (dead-reckoning)). Ancak göreceli konumlama sistemlerinin üretmiş olduğu lokalizasyon bilgisi zaman içerisinde hataların (kümulatif olarak) birikmesinden dolayı gerçek konumdan uzaklaşmakta ve elde edilen çözüm kullanılamaz hâle gelmektedir. Genelde göreceli ve mutlak pozisyonlama sistemleri algılayıcı füzyonu (sensor fusion) metodu ile beraber kullanılarak optimal tahminler yapılmaktadır. Bu tasarım çalışmasında açık ortamlarda veri toplayacağımızdan dolayı GPS sensörünü tercih ettim.

### **2.2. SparkFun GPS Modül Kit (EM-506 Alıcı)**

Bu çalışmada yaya tarafından taşınabilecek, yâni haff ve portatif, ve de Arduino ile uyumlu bir GPS modülü kullanmak istiyoruz. Bu nedenle, SparkFun firmasının geçmişte satışa sunmuş olduğu *GlobalSat WorldCom Corporation*<sup>1</sup> firması tarafından üretilen, SiRF Star IV isimli 48-kanallı EM-506 GPS alıcısını<sup>2</sup> kullanmayı uygun gördük. SparkFun firması DIY projelerinde Arduino ile kolayca kullanılabilisin diye bu alıcıyı Şekil 4'te görüldüğü gibi bir GPS Shield Kit<sup>3</sup> olarak tasarlamış. Bir Arduino Uno ile beraber SparkFun GPS Shield Kit'in lehimlenmiş hâlini Şekil 5 (a)'da görebilirsiniz. Şekil 5 (b)'de görüldüğü gibi Arduino üzerine takılan GPS Shield hem Arduino'dan 5V ile beslenmiş hem de seri port ve I2C gibi haberleşme protokollerini kullanarak GPS

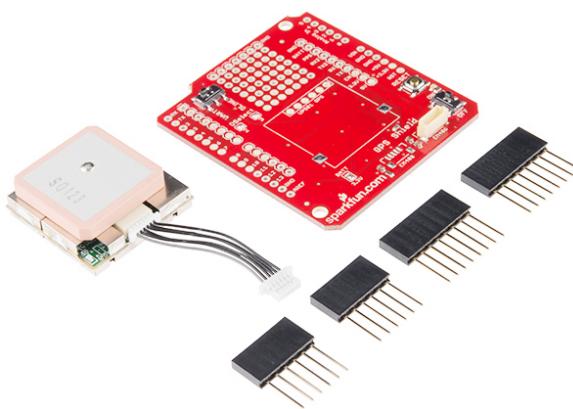
---

<sup>1</sup><https://www.globalsat.com.tw/>

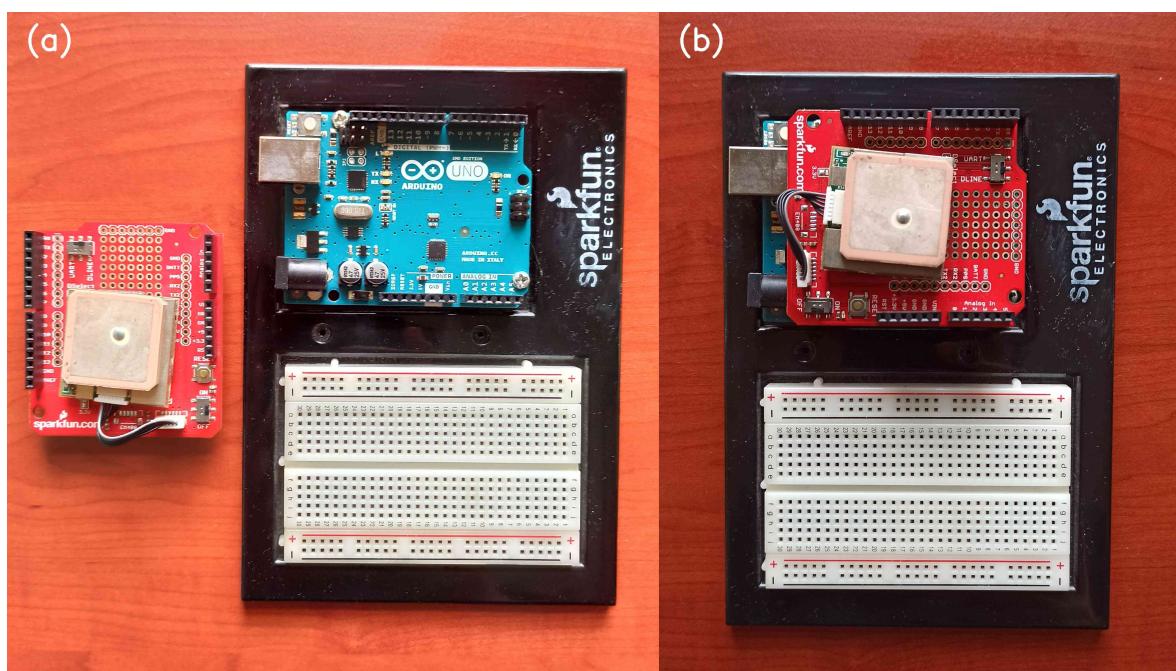
<sup>2</sup><https://www.sparkfun.com/products/12751>

<sup>3</sup><https://www.sparkfun.com/products/retired/13199>

koordinatlarını Arduino'ya göndermiştir.



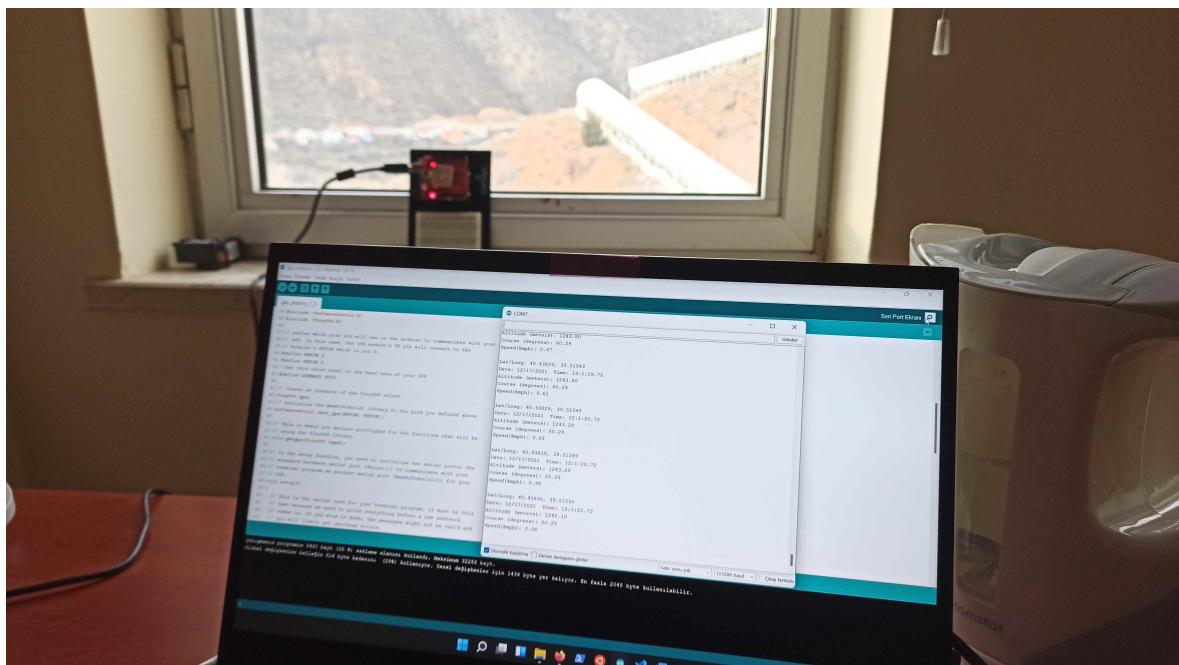
Şekil 4. SparkFun EM-506 GPS modül kit.



Şekil 5. (a) Lehimlenmiş SF GPS modül kit ve Arduino Uno. (b) SF GPS modül shield'in Arduino Uno'ya takılmış hâli.

### 2.2.1. Default Kod Test - USB Kablo

Müşterilerine her zaman örnek kodları sağlayan SparkFun firmasının web sitesinde alâkâlı GPS tutorial sayfasından<sup>4</sup> erişilen Mikal Hart'a ait github hesabında<sup>5</sup> yer alan **TinyGPS++** isimli kod deposundan (İng. repository) indirilen kodu (EK-1'de kodu görebiliyorsunuz) ilk önce Arduino'ya yükleyip sonra koşturduğumuzda, Arduino seri port penceresinde (doğru baud rate seçilmelidir) Şekil 6'da görüldüğü gibi GPS verilerini gözlemlemeye başladık.



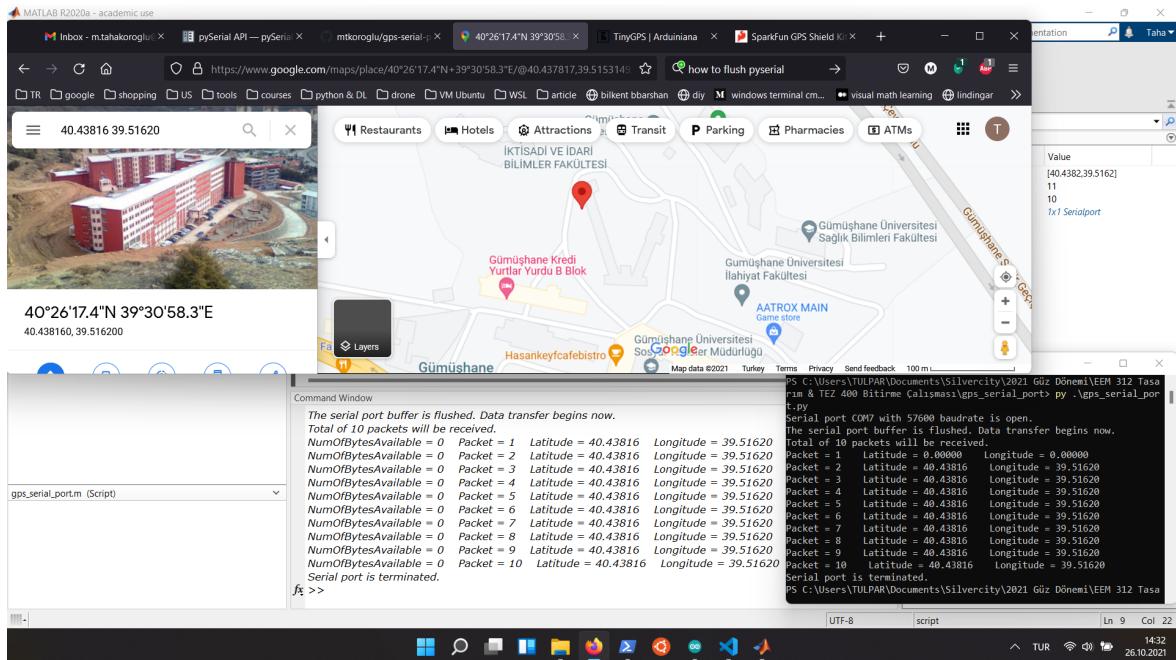
Şekil 6. SparkFun GPS Modülü testi (USB kablo ile bağlantı).

Default olarak bize sağlanan bu kodu anladıkten sonra kendi isteğimiz doğrultusunda değiştirerek seri port üzerinden hem MATLAB'a hem de Python aracılığıyla (*pyserial* paketinden faydalananarak) PowerShell konsol ekranına GPS koordinatlarını ve paket numaralarını okuduk. Alınıp konsola yazdırılan enlem ve boylam bilgilerini bir internet tarayıcısı açıp *Google haritalar*<sup>6</sup> sayfasına girdiğimizde Şekil 7'de görüldüğü gibi deneyi yaptığımız ofisin olduğu konuma oldukça yakın bir lokasyon tahmini yapıldığını gördük. Bölgenin dağlık yapısı ve bina içi bir deney olması göz önüne alındığında niteliksel olarak oldukça başarılı sonuç elde ettiğimiz söylenebilir.

<sup>4</sup><https://learn.sparkfun.com/tutorials/gps-shield-hookup-guide>

<sup>5</sup><https://github.com/mikalhart/TinyGPSPlus>

<sup>6</sup>[maps.google.com](http://maps.google.com)



Şekil 7. SparkFun GPS Modülü testi (MATLAB ve Python seri port veri okuma).

Şekil 7'de sayfanın sol üst tarafında GPS koordinatları girilen yere iki ondalık sayı enlem ve boylam bilgisi olarak girildiğinde çıkan bina resminin altında **derece-dakika-saniye** formatında temsil edilen koordinat bilgilerini görüyoruz. Yer bilimciler dünya üzerinde çalışırken bu formatı kullandığı için bu dönüşümün (ve ters dönüşümün) nasıl yapıldığını anlamak önemlidir.

## 2.2.2. GPS Koordinat Dönüşümleri

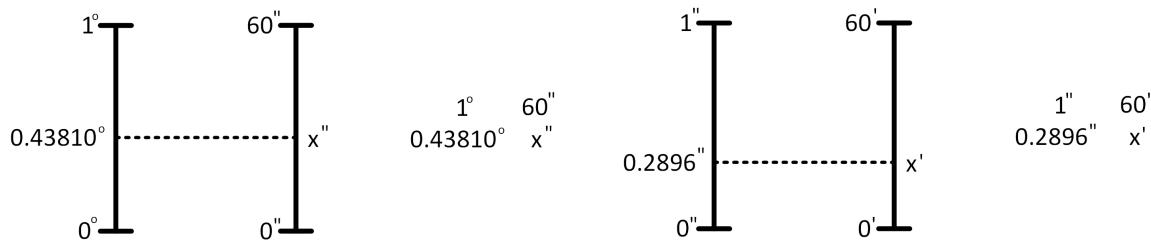
### 2.2.2.1. Ondalık Sayıdan Derece-Dakika-Saniye Formatına Dönüşüm

SparkFun GPS modül ile ilk veri yakaladığımız deney olan Şekil 7'de *Google Haritalar* sayfasında sol üstte koordinat bilgilerini girdiğimiz bölümün hemen altında görüldüğü gibi, okunan enlem-boylam bilgisi standart ondalıklı sayı<sup>7</sup> formatındadır. *Google Haritalar* bu iki ondalıklı sayıyı, bir başka ondalık sayı gösterimi olan dakika-derece-saniye formatına

<sup>7</sup>İng. floating point number. Birçok programlama dilinde **float** olarak bilinir. Matematikte **decimal** olarak da bilinir. Burada geliştirilen C++ (Arduino) ve Python kodlarında **float** olarak geçerken, MATLAB kodunda **single** veri tipi olarak tanımlanmıştır. Bu veri tipleri burada kullanılan Arduino Uno ve laptop üzerinde dört byte (otuz iki bit) yer kaplamaktadır.

$$\begin{pmatrix} 40.43816 \\ 39.51620 \end{pmatrix} \rightarrow \begin{pmatrix} 40^\circ 26' 17.4'' \\ 39^\circ 30' 58.3'' \end{pmatrix}$$

şeklinde dönüştürmüştür. Bu dönüşüm doğrusal bir dönüşüm olup Şekil 8'da görüldüğü gibi yapılmaktadır.



Şekil 8. Dereceden dakikaya ve dakikadan saniyeye dönüşüm.

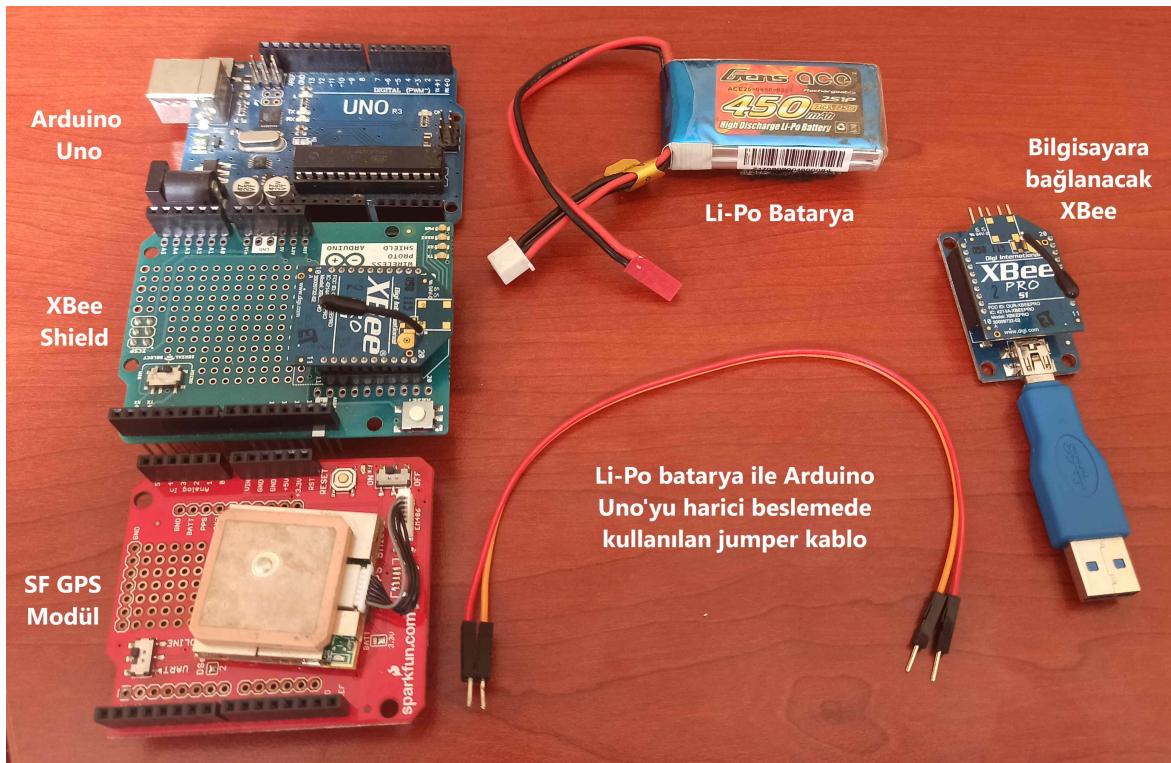
### 2.2.3. Default Kod Test - XBee kablosuz RF modül

Yukarıda USB kablo ile ilk testi yapılan GPS modülünü bilgisayarı yanımızda taşımadan kullanabilmek için kablosuz hâle getirmek istiyoruz. Bunun için Arduino ve SparkFun GPS Modül'e USB yerine hârici bir Li-Po batarya'dan güç verilmesi lâzım. Ayrıca, iptal edilen USB kablonun yerine, haberleşme vazifesini ede edecek bir kablosuz modül kullanılmalı. Bunun için kullanıcıya büyük kolaylık sağlayan XBee RF transceiver tercih edilebilir.<sup>8</sup> Bahsedilen elemanları Şekil 9'de görebilirsiniz. Arduino Uno, SF GPS modül shield ve XBee Shield'in bağlantıları Şekil 10'da görüldüğü gibi yapılip Şekil 9'de görülen Li-Po batarya ve jumper kablo ile güç verilirse kablosuz veri gönderimi gerçekleşecektir.

Şekil 9'de sağ tarafta görülen XBee alıcı Şekil 11'da görüldüğü gibi bilgisayara USB kablo ile bağlanırsa kablosuz veri alımı gerçekleşecektir ve kablosuz haberleşme sağlacaktır.

---

<sup>8</sup>USB kablo iptal edilip XBee RF transceiver'lar kullanıldığındada Arduino kodunda hiçbir değişiklik yapılması gerekmıyor. Öte yandan XBee RF modül yerine nRF24L01 gibi alternatif RF modüller kullanılırsa Arduino koduna mutlaka eklentiler yapmak gerekiyor.



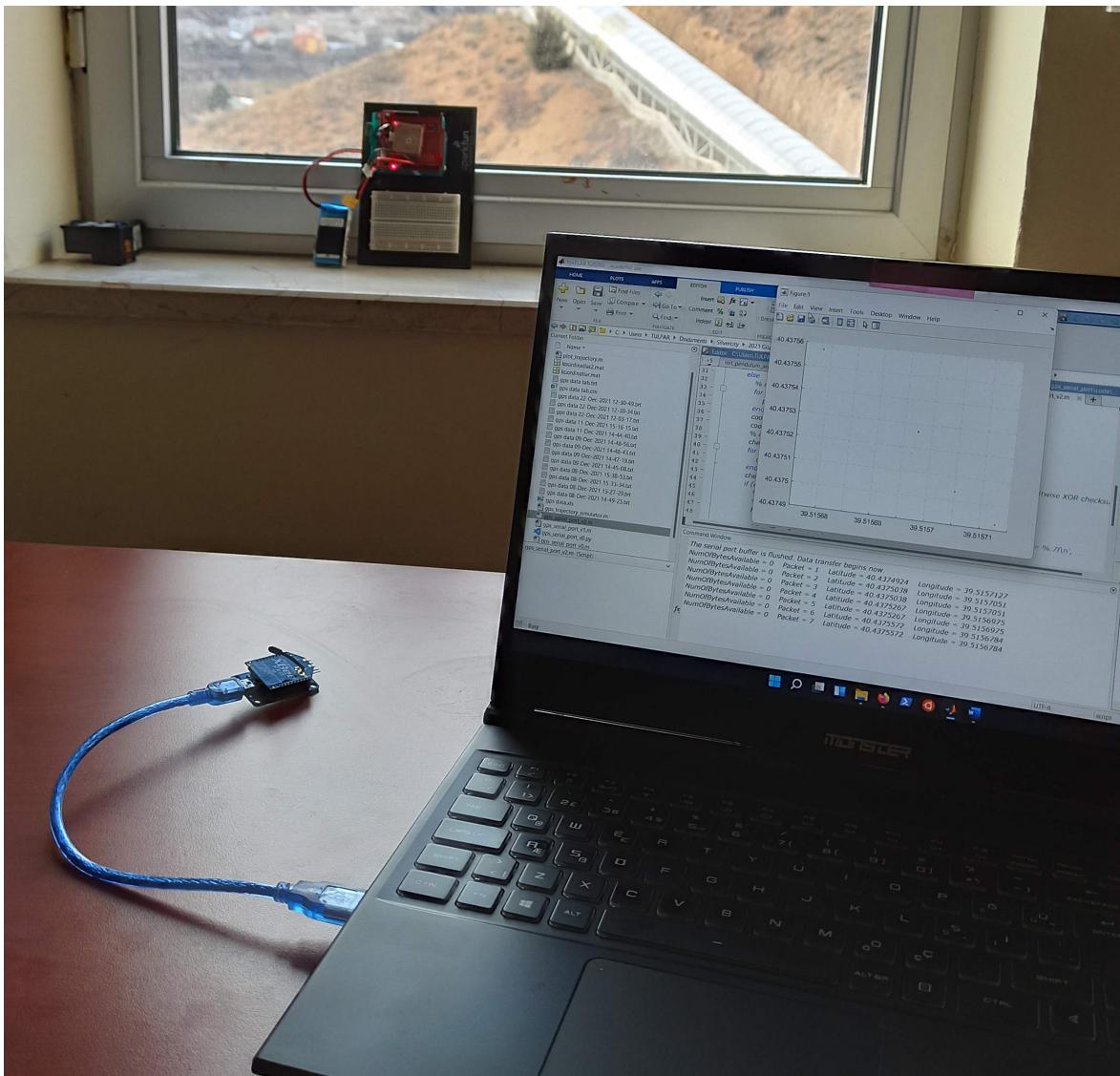
Şekil 9. Arduino, SparkFun GPS ve XBee kablosuz modüller ve Li-Po batarya.



Şekil 10. SparkFun GPS ve XBee kablosuz modüllerin Arduino Uno'ya bağlanması.

#### 2.2.4. Kendi Kodumuz ile Veri Toplama

Kullanıcının yürüyüş esnasında taşıyacağı SparkFun GPS modül tarafından yakalanan koordinat verileri, Arduino üzerinden XBee RF verici (transmitter) vasıtasıyla kullanıcı tarafından taşınma zorunluluğu olmayan bir bilgisayara bağlı XBee alıcıya



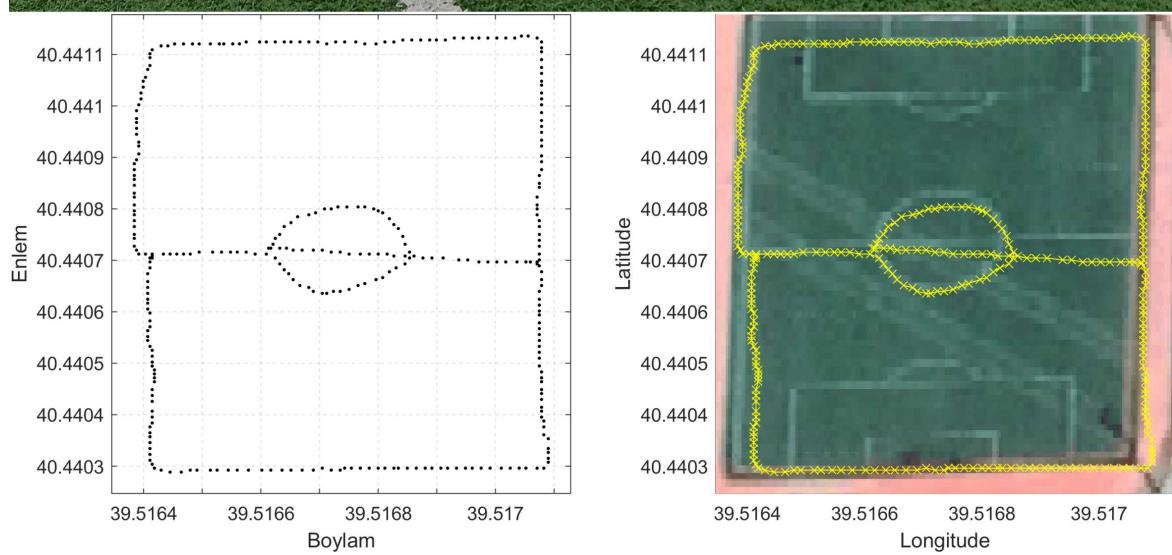
Şekil 11. SparkFun GPS Modülü kablosuz haberleşme testi (MATLAB veri okuma).

(receiver) kablosuz bir biçimde gönderilerek gerçek-zamanda (real-time) konum grafiği çizdirilebilir. Geçtiğimiz dönemlerde Gümüşhane Üniversitesi kampüsü içinde yer alan stadyumda yapılan deneylerden birinin sonucunu Şekil 12'de görebilirsiniz.<sup>9</sup>

Bir başka kullanıcı tarafından yürüyüse başlanan yere dönülecek biçimde Gümüşhane Üniversitesi Mühendislik ve Doğa Bilimleri binası etrafında dönlerek yapılan bir başka deneyin sonucunu ise Şekil 13'de görebilirsiniz.

---

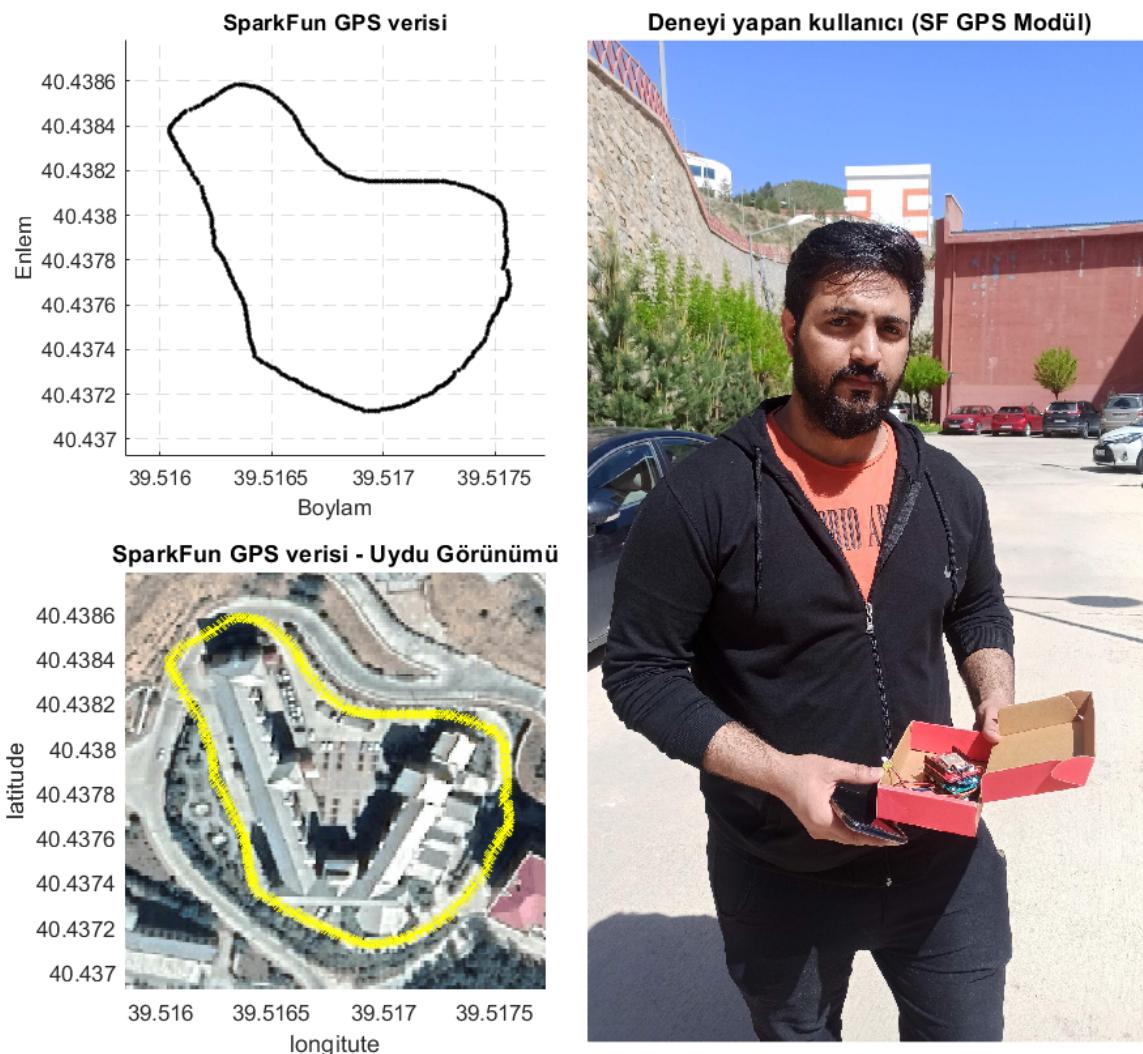
<sup>9</sup><https://www.youtube.com/watch?v=dK4XJg6-qIM>



Şekil 12. SparkFun GPS - stadyum yürüyüş testi.

### 2.3. Cep Telefonu Dâhili GPS

Konum Tabanlı Hizmetlerin (KTH) en çok yapıldığı cihazlar şüphesiz cep telefonları olduğundan dolayı cep telefonları standart olarak GPS alıcısı içermekte. Bu dâhili



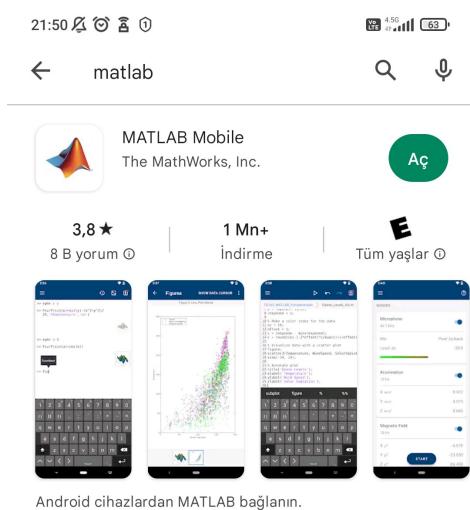
Şekil 13. SparkFun GPS - mühendislik binası testi.

GPS ile yukarıda kullandığımız hârici GPS modüle ihtiyacımız olmadan konum verilebilmizi kaydedebiliriz. Cep telefonu kullanarak yaptığımız deneylerde Arduino uyumlu hârici GPS ile veri toplarken yaptığımız gibi gerçek-zamanda veri görselleştiremezsek bile deney bittikten sonra kaydedilen dosyalardan veriyi okuyarak grafikleri çizdirebiliriz.<sup>10</sup>

<sup>10</sup> Aslında cep telefonlarındaki *Google Haritalar* gibi uygulamalar konumu gerçek zamanda göstermektedir ama yukarıda hârici GPS kullanılan deneylerde hem geçmiş konum bilgileri de gerçek-zamanda görselleştiriliyor hem de dosyaya kaydedebiliyoruz.

### 2.3.1. MATLAB Uygulaması

İster iOS olsun ister Android olsun hem ücretli hem ücretsiz bir çok algılayıcı verisi kaydedici (sensor registration veya sensor log) uygulama var. Bu uygulamalar arasında bizim en işimize geleni ve en güvenilir olanı MATLAB uygulaması. Android bir telefonda Google Play Store'da MATLAB uygulamasını aratınca karşımıza Şekil 14'de görülen sonuç geliyor.

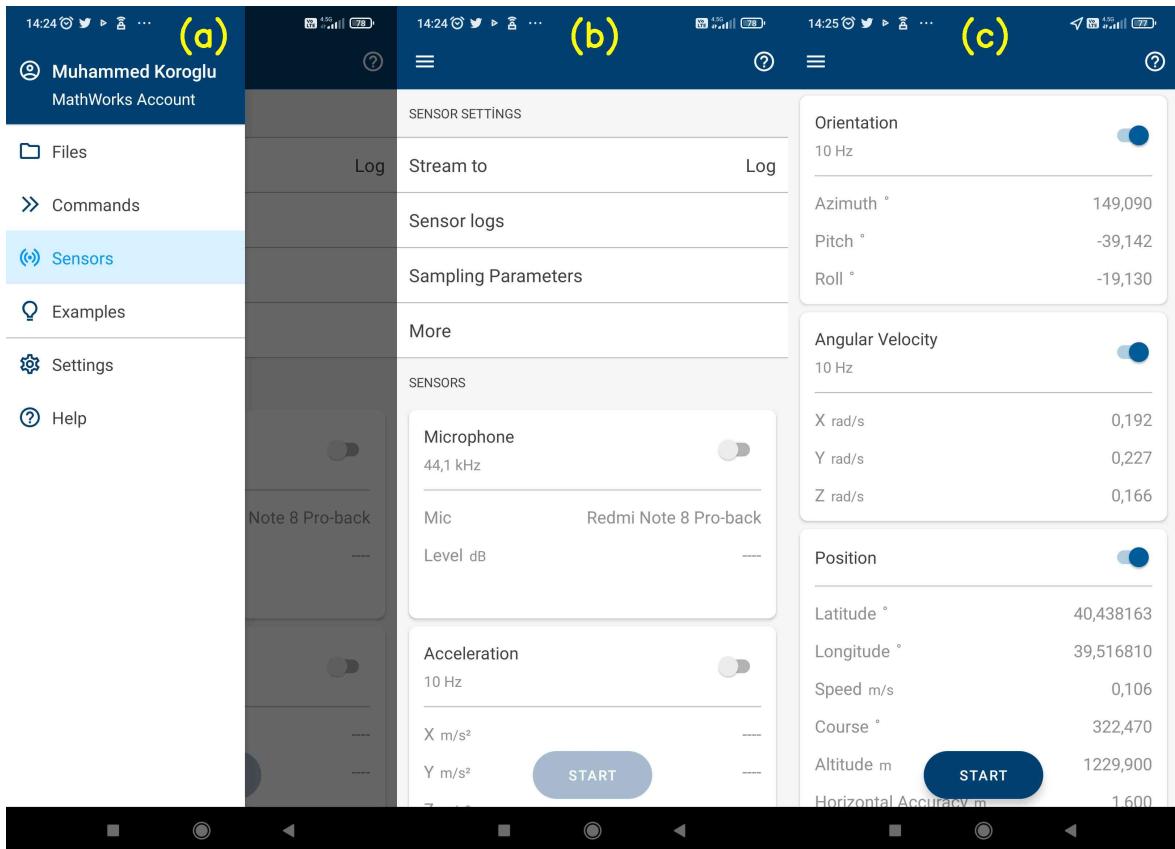


Şekil 14. Google Play Store'da (Android) MATLAB uygulaması aratma sonucu.

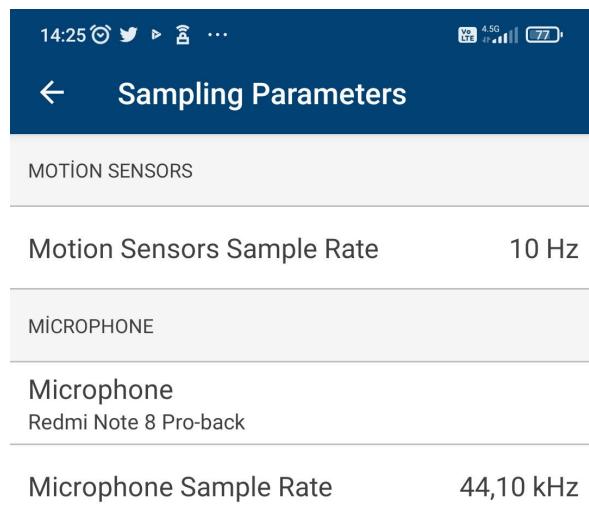
Uygulamayı indirip MATLAB programının üreticisi olan Mathworks hesap bilgi lerimizi girdiğimizde sensör verilerini zaman etiketli bir biçimde kaydetmek için uygulamayı ücretsiz olarak kullanabiliyoruz. Uygulamayı açtığımızda Şekil 15 (a)'da gözüken ekran karşımıza çıkıyor. Burada **Sensors** yazan yere tıklarsanız (b) geliyor ve ekranda aşağı inerseniz (c)'de görüldüğü gibi algılayıcıların (GPS en altta) aktifleştirildiğini görüyorsunuz. Burada **Start** yazan yere tıklayarak veri kaydını başlatabilirsiniz.<sup>11</sup>

Yukarıda Şekil 15 (b)'de **Sampling parameters** yazan yere tıklarsanız karşımıza Şekil 16'de görülen ekran geliyor. Burada algılayıcı yenileme hızını maksimum değer olan 100Hz yapmak mantıklı. Bununla beraber telefon en yüksek hangi örnekleme hızına destek veriyorsa o hızda veri yakalıyor. Örnek olarak, hareket sensörleri (i.e., ivmemetre ve jiroskop) verileri 200Hz gibi bir frekansla yakalanırken GPS verisinin örnekleme frekansı denenen bütün telefonlarda 1Hz olarak gözlemlendi.

<sup>11</sup>Veri toplarken önemli bir not: Ekran karardığında veri kaydı duruyor. Bu yüzden ekranın kapanmamasına dikkat etmelisiniz.

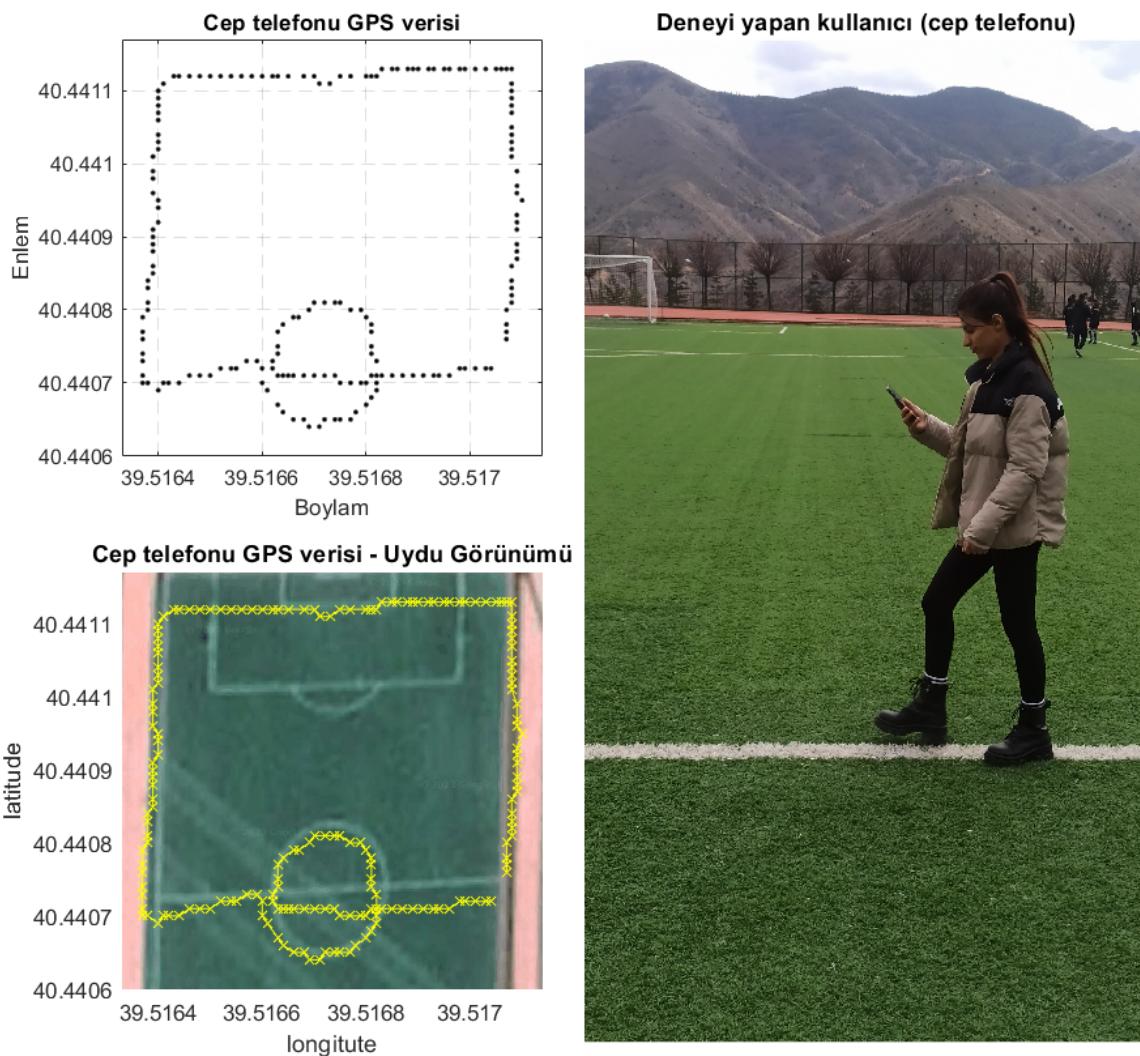


Sekil 15. (a) MATLAB uygulaması ana sayfa, (b) Sensör seçimleri 1, (c) Sensör seçimleri 2.



Sekil 16. (a) MATLAB uygulaması algılayıcı yenileme frekansı seçimi.

Bu ayarlamaları yaptıktan sonra sadece cep telefonu kullanarak yapılan bir deneyin sonuçlarını Şekil 17'da görebilirsiniz.

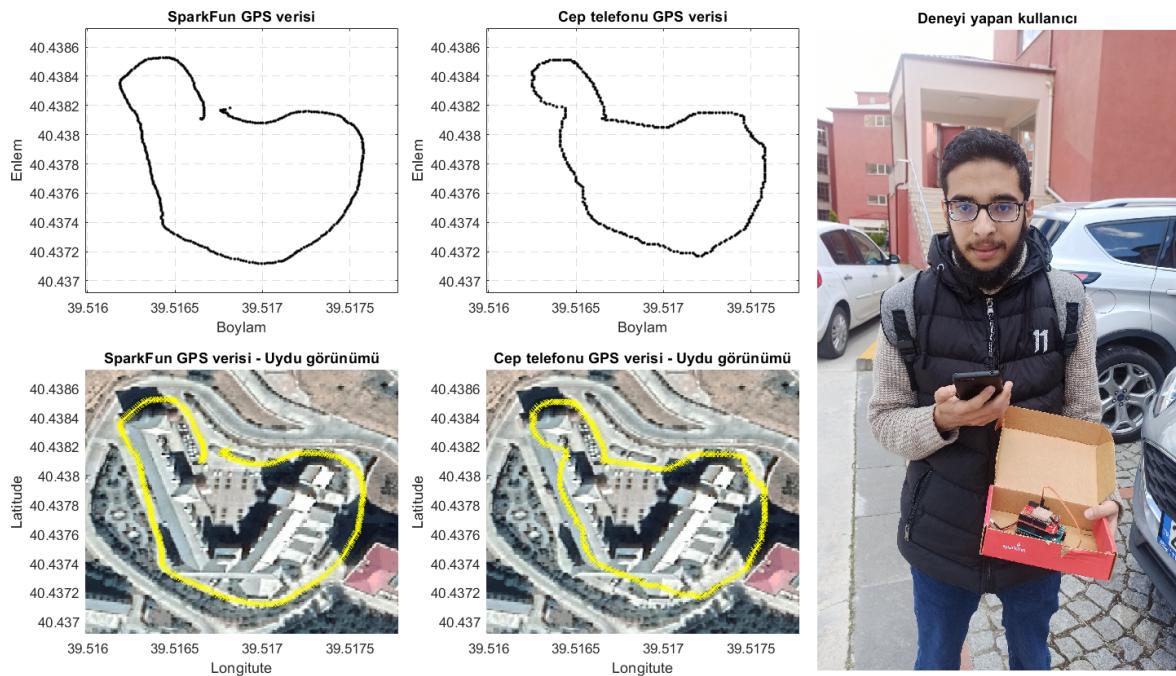


Şekil 17. Cep telefonu dâhili GPS - stadyum yarışaha testi.

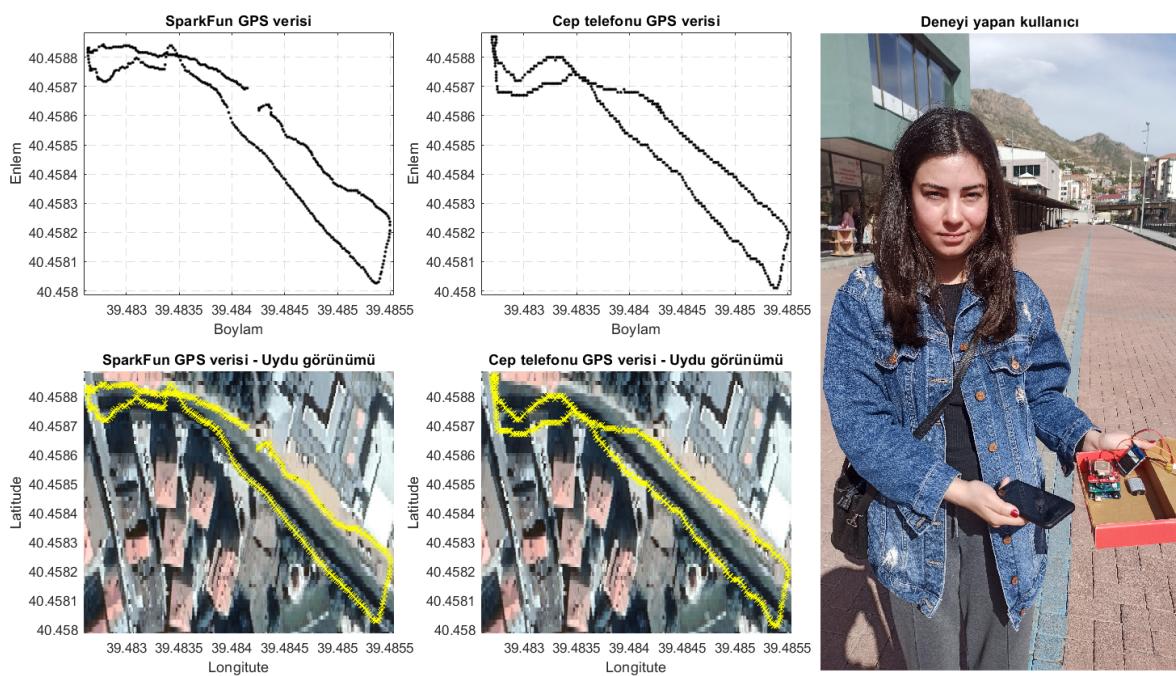
### 2.3.2. Hem Hârici Hem Dâhili GPS ile Veri Toplanan Deneyler

Şimdi hem hârici GPS'i açarak hem de cep telefonun dâhili GPS'ini aktif hâle getirerek yapılacak bir yürüyüşte iki ayrı GPS modülden alınan verilerin performanslarını kıyaslayabiliriz. Gümüşhane Üniversitesi Mühendislik binası etrafında bu sefer iki GPS alıcısıyla yapılan yeni bir deneyin sonuçlarını Şekil 18'da görebilirsiniz.

Gümüşhane merkezde yine iki GPS alıcısıyla yapılan başka bir deneyin sonuçlarını Şekil 19'da görebilirsiniz.



Şekil 18. Hârıcı ve dâhili GPS - mühendislik binası testi.



Şekil 19. Hârıcı ve dâhili GPS - şehir merkezi testi.

### **3. BULGULAR VE TARTIŞMALAR**

Arkadaşlar, kendi yorumlarınızı yazarsınız.

#### 4. KAYNAKLAR

1. J. Lategahn, M. Muller, and C. Rohrig, “EKF for a low cost TDoA/IMU pedestrian localization system,” in *11<sup>th</sup> Workshop on Positioning, Navigation and Communication (WPNC)*, pp. 1–6, March 2014.
2. M. D’Aloia, A. Longo, G. Guadagno, M. Pulpito, P. Fornarelli, P. N. Laera, D. Manni, and M. Rizzi, “IoT indoor localization with AI technique,” in *2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT*, pp. 654–658, 2020.
3. B. Jang and H. Kim, “Indoor positioning technologies without offline fingerprinting map: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 508–525, 2019.
4. M. Bolic, M. Rostamian, and P. M. Djuric, “Proximity detection with RFID: A step toward the internet of things,” *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 70–76, 2015.
5. R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
6. W. Hess, D. Kohler, H. Rapp, and D. Andor, “Real-time loop closure in 2D LIDAR SLAM,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278, 2016.
7. M. H. Sharif, “Laser-based algorithms meeting privacy in surveillance: A survey,” *IEEE Access*, vol. 9, pp. 92394–92419, 2021.
8. E. Guo, “A Roomba recorded a woman on the toilet. How did screenshots end up on Facebook?,” [MIT Technology Review]. Available: <https://www.technologyreview.com/2022/12/19/1065306/roomba-irobot-robot-vacuums-artificial-intelligence-training-data-privacy/>, Accessed: Mar. 29, 2023.

9. V. Guimarães et al., “A motion tracking solution for indoor localization using smartphones,” in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–8, Oct 2016.
10. T. Qin, P. Li, and S. Shen, “VINS-Mono: A robust and versatile monocular visual-inertial state estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

## 5. EKLER

### EK 1: SPARKFUN GPS MODÜLÜ (Default Arduino Kodu)

```
1 // In order for this sketch to work, you will need to download
2 // TinyGPS library from arduiniana.org and put them
3 // into the hardware->libraries folder in your ardiuno directory.
4 #include <SoftwareSerial.h>
5 #include <TinyGPS.h>
6
7 // Define which pins you will use on the Arduino to communicate with your
8 // GPS. In this case, the GPS module's TX pin will connect to the
9 // Arduino's RXPIN which is pin 3.
10 #define RXPIN 2
11 #define TXPIN 3
12 //Set this value equal to the baud rate of your GPS
13 #define GPSBAUD 4800
14
15 // Create an instance of the TinyGPS object
16 TinyGPS gps;
17 // Initialize the NewSoftSerial library to the pins you defined above
18 SoftwareSerial uart_gps(RXPIN, TXPIN);
19
20 // This is where you declare prototypes for the functions that will be
21 // using the TinyGPS library.
22 void getgps(TinyGPS &gps);
23
24 // In the setup function, you need to initialize two serial ports; the
25 // standard hardware serial port (Serial()) to communicate with your
26 // terminal program an another serial port (NewSoftSerial()) for your
27 // GPS.
28 void setup()
29 {
30     // This is the serial rate for your terminal program. It must be this
31     // fast because we need to print everything before a new sentence
32     // comes in. If you slow it down, the messages might not be valid and
33     // you will likely get checksum errors.
```

```

34     Serial.begin(115200);
35     //Sets baud rate of your GPS
36     uart_gps.begin(GPSBAUD);
37
38     Serial.println("");
39     Serial.println("GPS Shield QuickStart Example Sketch v12");
40     Serial.println("          ...waiting for lock...          ");
41     Serial.println("");
42 }
43
44 // This is the main loop of the code. All it does is check for data on
45 // the RX pin of the ardiuno, makes sure the data is valid NMEA sentences
46 // ,
47 // then jumps to the getgps() function.
48 void loop()
49 {
50     while(uart_gps.available())      // While there is data on the RX pin...
51     {
52         int c = uart_gps.read();    // load the data into a variable...
53         if(gps.encode(c))        // if there is a new valid sentence...
54         {
55             getgps(gps);          // then grab the data.
56         }
57     }
58
59 // The getgps function will get and print the values we want.
60 void getgps(TinyGPS &gps)
61 {
62     // To get all of the data into varialbes that you can use in your code,
63     // all you need to do is define variables and query the object for the
64     // data. To see the complete list of functions see keywords.txt file in
65     // the TinyGPS and NewSoftSerial libs.
66
67     // Define the variables that will be used
68     float latitude, longitude;
69     // Then call this function
70     gps.f_get_position(&latitude, &longitude);

```

```

71 // You can now print variables latitude and longitude
72 Serial.print("Lat/Long: ");
73 Serial.print(latitude,5);
74 Serial.print(", ");
75 Serial.println(longitude,5);
76
77 // Same goes for date and time
78 int year;
79 byte month, day, hour, minute, second, hundredths;
80 gps.crack_datetime(&year,&month,&day,&hour,&minute,&second,&hundredths)
81 ;
82 // Print data and time
83 Serial.print("Date: "); Serial.print(month, DEC); Serial.print("/");
84 Serial.print(day, DEC); Serial.print("/"); Serial.print(year);
85 Serial.print(" Time: "); Serial.print(hour, DEC); Serial.print(":");
86 Serial.print(minute, DEC); Serial.print(":"); Serial.print(second, DEC)
87 ;
88 Serial.print("."); Serial.println(hundredths, DEC);
89 //Since month, day, hour, minute, second, and hundr
90
91 // Here you can print the altitude and course values directly since
92 // there is only one value for the function
93 Serial.print("Altitude (meters): "); Serial.println(gps.f_altitude());
94 // Same goes for course
95 Serial.print("Course (degrees): "); Serial.println(gps.f_course());
96 // And same goes for speed
97 Serial.print("Speed(kmph): "); Serial.println(gps.f_speed_kmph());
98 Serial.println();
99
100 // Here you can print statistics on the sentences.
101 unsigned long chars;
102 unsigned short sentences, failed_checksum;
103 gps.stats(&chars, &sentences, &failed_checksum);
104 //Serial.print("Failed Checksums: "); Serial.print(failed_checksum);
105 //Serial.println(); Serial.println();
106 }
```

## EK 2: Python'da GPS Koordinat Dönüşümü

### GPS Koordinat Dönüşüm Fonksiyonları (transformation.py)

```
1 import math
2 import numpy as np
3
4 def float2dms(coordinateFloat):
5     absCoordinateFloat = abs(coordinateFloat)
6     absDeg = math.floor(absCoordinateFloat)
7     minFloat = 60 * (absCoordinateFloat - absDeg)
8     deg = np.sign(coordinateFloat) * absDeg
9     min = math.floor(minFloat)
10    sec = 60 * (minFloat - min)
11    return (deg, min, sec)
12
13 def dms2float(deg, min, sec):
14     return np.sign(deg) * (abs(deg) + min/60 + sec/3600)
```

### GPS Koordinat Dönüşüm Örnek Kod (gps-coordinate-transformation.py)

## ÖZGEÇMİŞ

Omar MANSOUR, ..... tarihinde .....’da doğdu. İlk ve orta öğrenimini ..... İlköğretim Okulu’nda, lise öğrenimini ise ..... Lisesi’nde tamamladı. .... yılında Gümüşhane Üniversitesi Mühendislik ve Doğa Bilimleri Fakültesi Elektrik Elektronik Mühendisliği Bölümü’nde lisans eğitimine başladı.