İNSAN HAREKETLERİ TABANLI GERÇEK RASGELE SAYI ÜRETİMİ

Yeliz GENÇ

YÜKSEK LİSANS TEZİ

Yazılım Mühendisliği AnaBilim Dalı

Danışman: Dr. Öğr. Üyesi Seda ARSLAN TUNCER

MART-2019

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İNSAN HAREKETLERİ TABANLI GERÇEK RASGELE SAYI ÜRETİMİ

YÜKSEK LİSANS TEZİ

Yeliz GENÇ

Anabilim Dalı: Yazılım Mühendisliği

Programı: Yazılım Mühendisliği

Danışman: Dr. Öğr. Üyesi Seda ARSLAN TUNCER

Tezin Enstitüye Verildiği Tarih: 01 Nisan 2019

MART-2019

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İNSAN HAREKETLERİ TABANLI GERÇEK RASGELE SAYI ÜRETİMİ

YÜKSEK LİSANS TEZİ

Yeliz GENÇ

(162137104)

Tezin Enstitüye Verildiği Tarih :
Tezin Savunulduğu Tarih : 01.03.2019

Tez Danışmanı : **Dr. Öğr. Üyesi Seda ARSLAN TUNCER (F.Ü)**

Diğer Jüri Üyeleri : **Dr. Öğr. Üyesi. Kazım HANBAY (B.Ü.)**

Doç. Dr. Fatih ÖZKAYNAK (F.Ü)

MART-2019

ÖNSÖZ

Tez konumun belirlenmesinde ve planlanmasında süreç boyunca yardımlarını esirgemeyen, deneyimlerini fikirlerini büyük bir özveri ile aktaran çok değerli danışmanım Dr. Öğr. Üyesi Seda ARSLAN TUNCER hocama sonsuz teşekkürlerimi sunarım.

Tez çalışmalarım boyunca maddi ve manevi olarak her zaman yanımda olan ve bana olan inançları ile beni yüreklendiren aileme teşekkürlerimi sunarım.

YELİZ GENÇ

ELAZIĞ-2019

İÇİNDEKİLER

Sayfa No

ÖZET

Rasgele sayıların kullanımı eski dönemlere dayanmakta ve günümüzde de bilgisayar bilimleri vb. alanlarda kullanılmaya devam etmektedir. Rasgele sayıların üretimi aşamasında rasgele sayı üreteçleri sözde, gerçek ve hibrit rasgele sayı üreteçleri olarak 3 grupta incelenmektedir. Gerçek rasgele sayı üreteci (GRSÜ) ile rasgele sayı üretmek için deterministik olmayan bir gürültü kaynağından yararlanılır. Rasgelelik derecesinin daha yüksek olması nedeniyle GRSÜ, SRSÜ sayı üreticisinden daha güvenli sayı üretirler.

Bu tezde, insan hareketleri ile rasgele sayı üreten bir GRSÜ çalışması yapılmıştır. Çalışmada GRSÜ, hemen hemen tüm insanların kullandığı mobil telefonlardaki ivme ve GPS sensörlerini kullanmaktadır. İlk olarak, mobil telefonu taşıyan kişinin 3-D ortamdaki hareketleri sonucunda ivme ve konum değişimleri Android tabanlı bir mobil cihazdan örneklenerek elde edilmiştir. Daha sonra, elde edilen bu veriler, normalizasyon işlemi uygulanarak ham sayı dizilerine dönüştürülmüştür. Son olarak, sayı dizilerinin istatistiksel özelliklerini iyileştirmek için XOR son işlemi uygulanmış ve rasgele sayı üretimi gerçekleştirilmiştir. Kişi yürürken, koşarken ve stabil konumda iken sensörlerden elde edilen toplamda 15 veri seti oluşturulmuştur. Sayıların istatistiksel özellikleri NIST Test Suite, Skala Index ve Otokorelasyon ile incelenmiştir. Çalışılan GRSÜ, cep telefonu platformu için uygun, evrensel ve düşük maliyetli olup, kişiye özgü rasgele sayı üretimini mümkün kılmaktadır.

Anahtar Kelimeler: Gerçek Rasgele Sayı Üreteci, İvme Sensörü, GPS, İstatistiksel Testler, Son işlem

SUMMARY

True Random Number Generation Based on Human Movements

The usage of random numbers dates back to old times and today it continues to be used in the areas such as computer science. Within the process of random number generation, random number generators are studied in three categories such as pseudo-random number generator (PRNG), true-random number generator (TRNG) and hybrid-random number generator (HRNG). Nondeterministic noise source is used in order to generate random number with true-random number generator (TRNG). Because it has higher degree of randomness, TRNG generates more secure numbers than PRNG.

In this thesis, TRNG, which generates random numbers by behavior of human, is studied. In the study, TRNG uses accelerometer and GPS sensor in cell phones used by almost all people. First, as a result of movements of a person carrying the mobile phone in the 3-D environment, the acceleration and position changes are obtained by sampling from an android based mobile device. Then, these obtained data are converted into raw number sequences by normalization process. Finally, to improve the statistical properties of the number sequences, XOR post processing is conducted and random numbers are generated. A total of 15 data sets are generated from the sensors while a person is walking, running orwaiting stably. The statistical properties of the numbers are examined by means of NIST Test Suite, Scale Index and Autocorrelation. The studied TRNG is suitable for mobile phone platform, universal and low cost, and it makes possible to produce random personal numbers.

Key Words: True Random Number Generator, Accelerometer Sensor, GPS, Statistical Tests, Post Processing.

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

SEMBOLLER LİSTESİ

KISALTMALAR

GPS : Global Positioning System
TRNG : True Random Number Generators
PRNG : Pseudo Random Number Generators
GRSÜ : Gerçek Rasgele Sayı Üreteçleri
SRSÜ : Sözde Rasgele Sayı Üreteçleri
ECG : Electrocardiogram
EEG : Electroensefalografi
EMG : Elektromiyografi
EOG : Elektrookulogram
XOR : Exclusive OR
NIST : Uluslararası Standartlar ve Teknoloji Enstitüsü
BBS : Blum Blum Shub Algoritması

SEMBOLLER

S_{obs} : Frekans testinde test istatistiği fonksiyonu
erfc : Tamamlayıcı hata işlevi
$S^{in}(s)$: Skala indeks testi iç skologram fonksiyonu
S_{min} : Minimum skala indeks değeri
S_{max} : Maksimum skala indeks değeri
$\oplus$: XOR Operatörü

1. GİRİŞ

Teknolojinin gelişmesiyle her geçen gün pek çok alanda ilerlemeler kaydedilmekte ve çalışmalar yapılmaktadır. Bu alanlardan biri de bilgisayar bilimleridir. Bilgisayar biliminde oyun programlama ve şifreleme, modelleme, simülasyon, eğlence gibi alanlarda rasgele sayı üretimine ihtiyaç vardır. Üretilen sayılar tahmin edilememe, tekrar üretilememe özelliği ve iyi istatistiksel özellikler içermelidir. Rasgele sayıların elde edilmesi amacıyla Gerçek Rasgele Sayı Üreteçleri (GRSÜ) ve Sözde Rasgele Sayı Üreteçleri (SRSÜ) olmak üzere iki üreteç vardır. Matematiksel bir fonksiyonun yardımıyla sayı üretiliyorsa, bu yolla üretilen rasgele sayılara sözde rasgele sayı adı verilir. Sözde rasgele sayılar üretmek için matematiksel fonksiyona bir başlangıç değer (tohum) verilir. Üretilecek sayılar tohuma bağlı olarak üretilir ve istendiğinde tohum değiştirilerek farklı rasgele sayılar üretilir. Her bir tohum değeri, ayrı bir rasgele sayı dizisi üretilmesine neden olur. GRSÜ gürültü kaynağı olarak kontrol edilemeyen ve tahmin edilemeyen gerçek fiziksel süreçleri kullanarak sayı üretir. GRSÜ'ler tarafından üretilen sayıların rasgeleliği fiziksel sürecin rasgeleliğine bağlıdır.

Literatürde rasgele sayı üretimi ile ilgili pek çok çalışma yapılmıştır. Literatürde gürültü kaynağı olarak elektronik devrelerde termal ve shot gürültüsü [1], jitter ve metastability [2,3,4], Brownian Motion[5], atmosferik gürültü ve nükleer bozulma [6] kullanılmıştır. Bunların yanı sıra ses, video, EEG (Elektroensefalografi), ECG (Elektrokardiyogram), Mouse hareketleri gibi insan kaynaklı gürültü kaynaklarından SRSÜ ve GRSÜ tabanlı üreteçler gerçekleştirilmiştir. Mousavi ve arkadaşları ECG (Elektrokardiyogram) sinyallerinden sözde rasgele sayı üretmişlerdir [7]. Üretilen rasgele sayıların kriptografik uygulamalar da anahtar olarak kullanılabilmesi için farklı iki yaklaşım sunmuşlardır. Bu yaklaşımlar Advanced Encryption Standard (AES) Algorithm ve ECG'nin Interpulse Interval (IPI) özelliği tabanlıdır. Her iki yaklaşım ile elde edilen sayılar NIST test suiti ile analiz edilmiş başarılı sonuçlar elde edilmiştir. Chen ve arkadaşları kriptografik sistemler için ECG sinyallerini kullanarak SRSÜ tabanlı sayı üreteci geliştirmişlerdir [8]. Geliştirilen sayı üreteci literatürde bilinen dokuz SRSÜ yapısı ile karşılaştırılmıştır. ECG tabanlı SRSÜ ile üretilen sayılar NIST istatistiksel testlerinden başarılı olmuş ve XOR (Exclusive OR

Generator), CCG (Cubic Congruential Generator) gibi SRSÜ'lerden daha iyi sonuçlar elde edilmiştir. Dang ve arkadaşları EEG (Electroensefalografi) sinyalleri kullanarak SRSÜ tabanlı bir üreteç önermişlerdir [9]. EEG sinyallerinin 0-1 sayı dizilerine dönüştürülmesi için modüler aritmetik kullanmışlardır. EEG veri seti alkolik kişilerden elde edilmiştir. Sayıların istatistiksel özellikleri NIST test suiti ile incelenmiş ancak bazı testlerden başarısız sonuçlar elde edilmiştir. Chen ve arkadaşları hem sağlıklı hem de hasta insanlardan alınan 5 farklı EEG işaretlerini analiz etmişlerdir [10]. EEG işaretlerinin gausian dağılımına uyduğunu göstermişlerdir. Üretilen sayılar NIST testinin non-periodic templates testinden başarısız olmuştur. Chen ve arkadaşları ses ve video görüntüler üzerindeki white noise sinyallerini gürültü kaynağı olarak kullanmışlardır [11]. GRSÜ ve SRSÜ tabanlı geliştirilen ses ve video rasgele sayı üreteci NIST testlerinden başarılı olmuştur. Nikolic ve arkadaşları ses kartı ve mikrofon yardımıyla elde ettikleri çevresel gürültü sinyallerini kullanarak GRSÜ tabanlı sistem geliştirmişlerdir [12]. Üretilen sayılar NIST, FIPS, Otokorelasyon testlerinden başarılı olmuş mükemmel kalitede sayı üretmişlerdir. Zhou ve arkadaşları mouse hareketlerinden rasgele sayı üretmek için GRSÜ önermişlerdir [13]. Sayı üreteci, kişisel bilgisayar platform için uygun, düşük maliyetli ve evrensel uygulamadır. Aynı kullanıcıların alışkanlıklarından kaynaklanan hareketleri yok etmek için kaotik hash fonksiyonu kullanmışlardır. x-y düzlemindeki mouse hareketleri 0-1 sayı dizilerine dönüştürülerek sayıların üretim hızı, difüzyon ve rasgelelikleri test edilmiş başarılı sonuçlar elde edilmiştir [13]. Xingyuan ve arkadaşları tek boyutlu kaotik harita ve mouse hareketleri kullanarak yeni bir GRSÜ geliştirmişlerdir. Üretilen sayılar NIST ve otokorelasyon testlerine tabi tutulmuş başarılı sonuçlar elde edilmiştir [14]. Hu ve arkadaşları mouse hareketlerinden 256 bitlik sayılar üretmek için yeni bir GRSÜ önermişlerdir [15]. Önerilen sistemde aynı kullanıcıların benzer hareketlerini yok etmek için Ayrıklaştırılmış 2D Kaotik Harita değişimi, spatio temporal kaos ve MASK algoritmalarını kullanmışlardır. Her üç algoritma ile üretilen sayılar istatistiksel testlerden başarılı olmuştur. Schulz ve arkadaşları kişiye özgü rasgele sayıların analizi için örüntü tabanlı analiz önermişlerdir. 20 sağlıklı insan tarafından her birinde 300 sayı bulunan ikişer adet sayı dizileri oluşturmuşlardır. İnsana özgü rasgele sayı dizisi içinde kişiye özel bilgilerin var olabileceği gösterilmiştir [16]. Tuncer ve arkadaşları EEG (Electroensefalografi), EMG (Elektromiyografi) ve EOG (Elektrookulogram) gibi biyoelektrik sinyallerinden ve GSR (Galvanic Skin Response) fiziksel sinyallerinden gerçek rasgele sayı üretimini sunmuşlardır. Bu sinyaller için BNCIHORIZON2020

veritabanı kullanılmıştır. Burada her bir sinyal normalleştirilmiş ve örnekleme işlemleri yapılmıştır. Örnekleme için kaotik bir lojistik harita kullanılmıştır. Örneklenen istatistiksel sayıların özelliklerini iyileştirmek için XOR son işlemi uygulanmıştır. Üretilen sayılara NIST istatistiksel testleri uygulanmıştır. Biyoelektrik ve fiziksel sinyallerden üretilen sayılar tüm testlerde başarılı olmuştur [17]. İnsan kaynaklı gürültü kaynakları kullanan rasgele sayı üreteçlerinin özeti Tablo 1.1'de verilmiştir.

Tablo 1.1. İnsan kaynaklı rasgele sayı üreteçleri ile ilgili çalışmalar

Referanslar	Gürültü Kaynağı	Uygulama Tipi	Testler	Performans
Chen ve ark.[8]	ECG sinyali	SRSÜ	NIST	Başarılı
Dang ve ark.[9]	EEG sinyali	SRSÜ	NIST	Kısmen Başarılı
Chen ve ark.[10]	EEG sinyali	SRSÜ	NIST	Kısmen Başarılı
Chen ve ark.[11]	Ses, Video	SRSÜ,GRSÜ	NIST	Başarılı
Nikolic ve ark.[12]	Ses	GRSÜ	NIST, FIPS, Otokorelasyon	Başarılı
Zhou ve ark.[13]	Mouse Hareketi	GRSÜ	Diffusion, NIST	Başarılı
Xingyuan ve ark.[14]	Mouse Hareketi	GRSÜ	NIST, Otokorelasyon	Başarılı
Hu ve ark.[15]	Mouse Hareketi	GRSÜ	NIST	Başarılı
Schulz ve ark.[16]	İnsan Kaynaklı	GRSÜ	NIST	Başarılı
Tuncer ve ark.[17]	Kaotik Sinyal	GRSÜ	NIST	Başarılı

Literatürdeki farklı kaynaklarla yapılan bazı çalışmalara bakıldığında Çiçek ve arkadaşları çift çekirdek entropili rasgele sayı üreteci tasarlamışlardır. Bu tasarımda 180 nm CMOS teknolojisinde üretilmiş, yapılan prototip çarpımından elde edilen sayılar herhangi bir post processing işlemi gerektirmeden tüm Ulusal Standartlar ve Teknoloji Enstitüsü 800.22 istatistiksel testleri uygulanmış %85 başarı elde edilmiştir [18]. Cao ve arkadaşları kuantum fiziği uygulayarak rasgele sayı üretmişlerdir. Üretikleri rasgele sayılar

üç gruba ayrılmış ve ayrılan sayıların rasgeleliği ve güvenirliği test edilmiştir. NIST istatistiksel testlerden geçen sayılar quantum tabanlı rasgele sayıların başarılı sayılar ürettiğini göstermiştir [19]. Özkaynak ve arkadaşları kaotik ek girdi ile şifreli olarak güvenli bir şekilde rasgele sayı üretmek için bir mimari tasarlamışlardır. Bu mimariyi tasarlarken rasgele bir jeneratörün gereksinimleri açıklanmış bu gereksinimlerden hareketle hibrit bir mimari sunulmuştur. Yapılan güvenlik analizi, önerilen jeneratörün çıktılarının rasgele göründüğünü göstermiştir [20]. Schingver ve arkadaşları ReConfig 2010'da keyfi dağılımlar ve kesinlikle kullanılabilen inversiyon tabanlı uygulamaya kıyasla %48 tasarruf sağlayan bir tasarım sunmuşlardır. Bu tasarımda inversiyon yöntemi, kayan noktalar metodu ile rasgele sayılar üretmişlerdir. Ürettikleri sayıları sentez ve kalite testlerinden geçirmişlerdir ve mevcut tasarıma göre avantajlı olduğu ortaya çıkmıştır [21].

1.1. Tezin Amacı

Bu tez çalışmasında, GRSÜ için insan hareketleri tabanlı insana özgü, düşük maliyetli rasgele sayı üretimi amaçlanmıştır. Bu amaç doğrultusunda kişinin yürüme, koşma ve durma aktivitelerinin ivme sensörü ve GPS üzerinden verileri alınarak bu veriler önce örnekleme daha sonra normalizasyon işlemi yapılmıştır. Sonrasında sırasıyla örnekleme, normalizasyon ve son işlem işlemleri uygulanmış bu adımlarla elde edilen sayı bitleri hesaplanarak rasgele sayılar üretilmiştir. Elde edilen sayılar literatürde bilinen NIST, Skala Indeks ve Otokorelasyon gibi belirli istatistiksel testler uygulanmıştır.

1.2. Tezin İçeriği

Tez içeriği 7 bölümden oluşmaktadır. İkinci bölümde rasgele sayılar ve tarihsel gelişimi ile genel bilgiler verilmiş, devamında rasgele sayı üreteçleri ile bilgiler yer almıştır. Üçüncü bölümde insan hareketleri biyomekaniği hakkında bilgi verilmiştir. Dördüncü bölümde sensörler ve GPS konuları açıklanmıştır. Beşinci bölümde tez çalışmasında kullanılan literatürde bilinen testlere yer verilmiştir. Altıncı bölümde tez çalışmasının adımları ifade edilmiştir. Yedinci bölüm yani son bölümde ise çalışma sonuçları ve öneriler hakkında bilgiler verilmiştir.

2. RASGELE SAYILAR VE RASGELE SAYI ÜRETEÇLERİ

Genel olarak rasgele sayılar matematiksel fonksiyonlarla ya da fiziksel yöntemlerle üretilen uzun diziler şeklinde ifade edilmektedir. Bir sayı tek başına tam olarak rasgele sayı değildir. Rasgele sayıdan bahsederken sadece bilgisayar bilimlerinin bir konusu olarak düşünmemek gerekir. Rasgele sayılar bilgisayar bilimlerinde dışında farklı alanlarda da kullanılmaktadır. Bunlardan bazıları; zar atma, sayısal loto, oyunlar, örnekleme, nümerik analiz, karar verme, simülasyon, modelleme, şifreleme vb. şeklindedir. Bu sayıların aslında günlük yaşantımızda da kullanıldığı görülmektedir.

2.1. Rasgele Sayıların Kullanıldığı Alanlar

- **Simülasyon:** Fiziksel olaylar simule edildiği zaman rasgele sayılara gerçeğe yakın durumlarda gereksinim duyulmaktadır. Bu alan nükleer fizik gibi birçok çalışmayı kapsamaktadır [22].
- **Modelleme:** Rasgele sayılar farklı alanlarda yapılan çalışmaların durumlarını modellemek için kullanılabilmektedir. İstatistiksel uygulamalar gibi bazı uygulamalar için modellemeler yapılabilmektedir.
- **Örnekleme:** İstatistiksel uygulamalarda olasılıksal yöntemlerle örnekleme seçimi yapılırken rasgele sayılar kullanılmaktadır. Bu sayılarla örnekleme seçimleri daha kolay gerçekleştirilmektedir.
- **Şifreleme:** Son dönemlerde kriptografik uygulamalarda rasgele sayılar etkin bir şekilde kullanılmaktadır. Örneğin bankacılık işlemlerinde kriptografik tasarım çalışmalarında gerektiği yerlerde rasgele sayılar yaygın olarak kullanılmaktadır.
- **Oyunlar:** Zar atma, iskambil kartları gibi eğlence oyunlarında da genel olarak rasgele sayılar kullanılmaktadır. Ayrıca son dönemlerde bilgisayar ve cep telefonları oyunlarında da rasgele sayılar yaygın olarak kullanılmaktadır.

2.2. Rasgele Sayı Üreteci

Rasgele sayı üreteci gerçek dünyadaki rasgele sayıları üretmek için oluşturulan yapının genel ismi şeklinde ifade edilmektedir. Rasgele sayı üretecinde sayı üretimi yapılırken belirli özelliklerin sağlanması gerekmektedir. Bunlardan temel olarak mümkün olduğu kadar rassal olması, büyük periyotlarda yani uzun bir seride rassallık sağlaması, üretilen rasgele sayıların yeniden üretilebilir, hesaplanabilir ve gerektiği durumda tekrar kullanılabilir olması gerekmektedir. Rasgele sayı üreteci ile ilgili farklı tekniklerle çalışmalar yapılmaktadır. Bu çalışmalar farklı tekniklerle üretilen sayıların bu bahsedilen özelliklerin gerçekleştirilmesini hedeflemektedir. Bu da genel olarak bir çalışmanın başarımını etkilemektedir.

2.3. Rasgele Sayı Üretecinin Tarihsel Gelişimi

Rasgele sayıların tarihi çok öncelere dayanmaktadır. Bunun ilk akla gelen örneklerinden biri zarlardır. Zar, bozuk para ve diğer cihazlar rasgele seçimler ve şans oyunlarında rasgele sayılar üretmek için uzun zaman önce kullanılmıştır. İran ve Irak'ta 5000 yıllık zarlar bulunmuştur. 4000 yıl önce de Hindistan, Çin, Mısır'da zarlar yaygın olarak kullanılmıştır [23].

Zarlar kriptolojinin ilk adımlarının atılmasına katkıda bulunmuştur. Kullanılmış ilk zarlar hayvanların ayak bileği kemiklerinden, pişmiş topraktan, bazı özel taşlardan yapıldığı dile getirilmektedir. Kullanılan maddeler dışında zarların yüzey sayıları da o dönemde değişiklik göstermiştir. Bunun bir örneği Mısır'da bulunmuş olup Helenistik döneme ait olduğu tahmin edilen 20 yüzlü zarlar olmuştur. Eski çağda insanlar zarların sonuçlarının gerçekte rasgele olmadığına sadece Tanrı tarafından karar verildiğine inanırlardı. Bulunduğu dönemde zarlar miras paylaşımı, başkanlık seçimi gibi kararların verilmesinde kullanılmıştır [23]. Zarların yanı sıra kağıt oyunları, madeni paralar, döner tekerlekler v.b nesneler de erken dönem rasgele sayı üreteçleri olarak adlandırılmaktadır. Çin'de, oyun kağıtları yani iskambil kağıtları 7. yüzyılın başlarında kumar oyunları için kullanılmıştır. 1300'lü yılların sonlarında Avrupa'da kullanılmıştır. 2000 yıl önce Roma İmparatorluğu'nda bazen hayat ve ölüm arasına tek bir bit karar vermiştir. Burada bahsedilen o dönemki paralar ile yazı-tura denemelerinin yapılmasıdır. Antik Roma'da

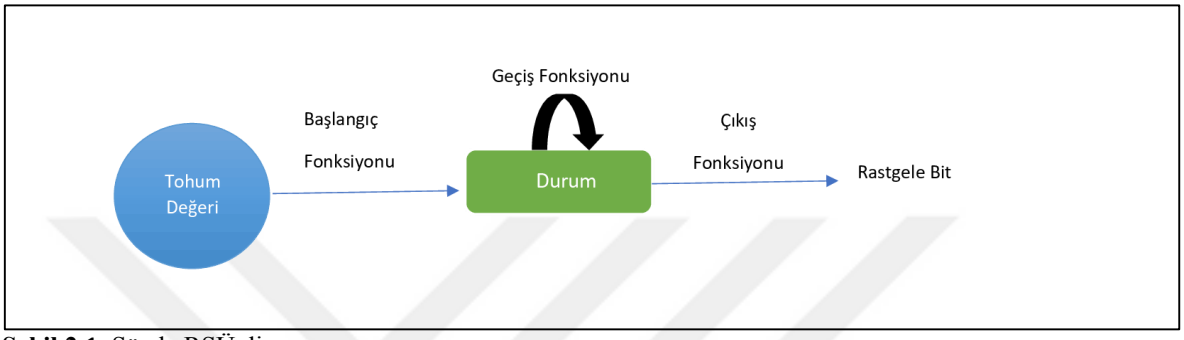
popüler olan bu uygulama zamanla Avrupa'ya yayılmıştır. Eski Yunanlılarda ortaya çıkan döner tekerlekler bahis aracı olarak kullanılmıştır. Buna göre tekerlekler oluşturulması için bir mızrak noktasında dengeli bir kalkan yapmışlardır. Kalkan bölüm olarak işaretlenmiş ve oyuncular kalkanın duracağı yere bahse girmişlerdir. Bu tekerlekler servet panosu haline gelmiş ileride rulete dönüşmüştür.

Rasgele sayıların ileriki zamanlarında kriptolojide Vernam şifreleme sistemi olarak isimlendirilen XOR (exclusive OR), veri dizisinin anahtarı şeklinde ifade edilen açık metinde (plaintext) gibi rasgele sayı metotları ortaya çıkmıştır. Bu dizi güvenli şifreleme protokolü olarak bilinmektedir. Bu şifreleme protokolünün temel uygulamalarda güvenilirliği sağlaması için her anahtarın (rasgele sayının) sadece bir defa kullanılması ve gerçek rasgele sayı kriterlerini sağlaması gerekmektedir. Bilgisayar tabanlı RSÜ' nün tasarım ve analizi için Knuth bir çalışma yayınlamıştır. Çalışmasında bilgisayar kullanıcılarının farklı uzman kişiler tarafından geliştirilen rasgele sayıları üretmek için oluşturdukları kütüphanelere erişim sorunlarını adreslemiştir [24]. Rasgele sayılarla ilgili çalışma yapan bir başka yazar Ripley [25] ise kişisel bilgisayarlar üzerinde yeterli olmayan RSÜ kullanıcı programları ile yer değiştirilmiş üstel, normal ve Poisson dağılımlı diziler içeren metotlar geliştirmiştir. Başka bir çalışmasında Monte Carlo hesaplamaları için sözde rasgele sayı üreteçlerinin kullanımı ile ilgili çalışması olmuştur.

2.4. Sözde Rasgele Sayı Üreteci (Pseudo Random Number Generator)

Sözde rasgele sayı üreteci genel olarak rasgelelik kriterine uygun olarak ilişki kurulması zor olan sayı dizisi üreten algoritma türüdür. Bir dizide her sayı eşittir özelliğine yaklaşma durumu bazen yeterli değildir. Örneğin; 1'den 100'e kadar olan dizi için her sayı birer kez bulunmaktadır. Burada rasgeleliğin düşük olduğu belirgindir. Genel olarak rasgele sayıların üretilirken bazı sayıların birden çok bulunması bazı sayıların da mümkün olduğunca olmaması gerekmektedir. Yani sözde rasgele sayı üreteçleri bu özellikleri içermeli ve geliştirilen birçok rasgelelik testi, homojenliğe dayalı testleri gibi testleri geçmesi gerekmektedir [26]. Sözde rasgele sayı üreteçleri Turing Makinesi gibi deterministik bir bilgisayarda çalıştıkları için deterministik algoritmalardır ve bu tür bir algoritma ile üretilen sayı dizisinde gerçek bir rasgele dizide olmayan periyot sınırlılığı gerçekleşecektir. Üreteç sabit kapasitede hafıza kullanmakta belirli sayıda döngü işleminden sonra aynı duruma ikinci defa gelmekte ve bu durum sonsuza kadar devam

etmektedir. Periyodik olmayan bir üreteç tasarlanabilir fakat böyle bir sistem çalıştıkça ihtiyaç duyacağı hafıza kapasitesi de artacaktır. Bunun yanı sıra sözde rasgele sayı üreteci belirlenen bir başlangıç yani çekirdek durumundan başlama verilebilir ve başlamasından itibaren özdeş bir sayı dizisi üretmektedir. Burada periyodiklik sınırlı bir durumdadır [26]. Sözde rasgele üreteçlerinin genel tasarım diyagramı Şekil 2.1'deki gibidir:



Şekil 2.1. Sözde RSÜ diyagramı

Sözde rasgele sayı üreteçleri adımları başlatma fonksiyonu, geçiş fonksiyonu ve çıkış fonksiyonu olarak tanımlanmaktadır. Giriş fonksiyonunda başlangıç değeri yani tohum değerini alır ve alınan tohum değeri başlangıç durumuna getirilir. Geçiş fonksiyonunda üretecin değerini çıkış fonksiyonuna girmesinden önce rasgele sayısını üretilebilir duruma dönüştürmektedir. Mevcut durumu yani sıfır ve bir bit dizisi üretmek için çıkış fonksiyonu kullanılır ve sürecin tamamlanması sağlanmaktadır. Genel olarak seçilen tohum değeri geçiş ve çıkış fonksiyonunu tekrar tekrar çağırarak elde edilmektedir.

Sözde rasgele sayı üretimi için uygulanan pek çok algoritma bulunmaktadır. Bu algoritmalar matematiksel fonksiyonlara dayanarak belli adımlarda sözde rasgele sayı üretmeyi gerçekleştirmektedir.

2.4.1. Orta Kare Tekniği

Bu teknik 1946 yılımda Vonneuman ve arkadaşları tarafından geliştirilmiştir. Rassal sayı üretiminde farklı uygulamalarda kullanılmıştır. Bu tekniğin adımları şu şekildedir:

- Üretilecek sayı kaç basamaklı olacaksa o basamak kadar x_0 diye tanımlanabilecek rasgele bir başlangıç değeri verilir.

- Sayının karesi alınır. Karesi alındıktan sonra ortasından belli sayıda değer alınır ve yeni sayı üretilir. Üretilen sayıda istenilen basamak değerinin eksikliği durumunda sayının sol kısmına sıfır eklenir.
- Hedeflenen sayılar elde edilene kadar adımlar bu şekilde devam eder ve gerçekleştiğinde işlem tamamlanır.

Örneğin; belirlediğimiz sayı 4 basamaklı 1920 sayısı olsun:

$$x_{0=}1920 \longrightarrow (x_0)^2=03686400$$

$$x_{1=}6864 \longrightarrow (x_1)^2=47114496$$

$$x_{2=}7114 \longrightarrow (x_2)^2=50608996...$$

Her yöntem ya da teknikte avantajlar olduğu kadar dezavantajlar da söz konusudur. Orta kare yönteminde de istatistiksel olarak tatmin edici olmaması, analizinde zorluklar yaşanması ve diziler arasındaki periyodu kısalığı gibi dezavantajları mevcuttur.

2.4.2. Lineer Benzerlik Algoritması

1951 yılında D.H.Lehmer tarafından geliştirilmiştir. Bu algoritmada, rasgele üretilmesi hedeflenen her sayı kendinden önceki sayıyla sabit bir sayının çarpımından oluşmaktadır. Sonrasında, mod işlemi uygulanmaktadır. Bir sonraki yalnız çarpım sonucunda değil, bu çarpımın önceden bilinen bir sayıya bölümünden elde edilen kalanla belirlenmiş olacaktır. Başlangıç çekirdek sayısına (s) denildiğinden adet rasgele sayı Denklem 2.1'de gösterildiği gibi ifade edilmektedir. Bu sayılar 0 ile m-1 aralığında olacaktır. s başlangıç değerini, c çarpım değerini, b ve m sabit tam sayı değerini ifade etmektedir [26].

$$u_n = s,$$

$$u_{n+1} = (c.u_n + b) \qquad \textbf{(2.1)}$$

$$0 < u_n < m$$

Bu algoritma şu şekilde çalışmaktadır. Örneğin; başlangıç değeri s=5212, çarpılacak değer c=6 ve b sabitinin değeri de 1 şeklinde verilsin. Sayıların mod 31 'e göre hesaplanması yapılsın. İşlemler aşağıdaki şekilde uygulanmaktadır ve ilgili adımlar işlem tamamlanıncaya kadar devam etmektedir.

$$\mathbf{u_1 = (6 * 5212 + 1) mod 31 = 31273\ (mod 31) = 25}$$

$$\mathbf{u_2 = (25 * 6 + 1) mod 31 = 151 (mod 31) = 27}$$

$$\mathbf{u_3 = (27 * 6 + 1) mod 31 = 163\ (mod 31) = 8}...$$

Bilgisayar programlarında üretilen rasgele sayılarda genelde üretilen sayılar ayrı bir dizide tutulmamaktadır. Lineer benzerlik algoritması işlemlerinde tekrarlamaların mümkün olduğunca olmaması için değerlerin belirli değerlendirmelere göre seçilmesi gerekmektedir. Bu teknikte diğer yöntemlerde olduğu gibi tekrarlı döngüye girmesi tekniğin dezavantajı olacaktır. Bu şekilde üretilmesi gereken bazı sayılara erken erişilmektedir.

2.4.3. Blum Blum Shub Algoritması

Blum Shub (BBS) 1986 yılında Lenore Blum, Manuel Blum ve Michael Shub (Blum ve diğerleri, 1986) tarafından geliştirilen bir sözde rasgele sayı üreteci algoritmasıdır [27]. BBS'nin kriptografik olarak güvenli bir sözde sanal jeneratör (CSPRBG) olduğu belirtilmektedir. Bu sanal jeneratör, bir sonraki bit testini geçişi tanımlamaktadır. Eğer dizinin ilk k bitleri verilmişse, bir sonraki bitin büyük ihtimalle 1 veya 0 olacağını tahmin edebilecek basit bir algoritma bulunmayabilir [28]. Kriptografik çalışmalar için güvenli bir SRSÜ bu algoritma güçlü bir güvenlik kanıtlamasına sahiptir fakat güvenli olması için büyük sayılar gerekmekte buna bağlı hesaplama yoğunluğunu artırmaktadır [29].

- Rasgele p ve q olmak üzere iki büyük asal seçilir. Burada p ve q farklı sayılar olmalıdır.
- M=p*q hesaplaması yapılır.
- s diye tanımlanan bir çekirdek sayısı seçilir. Bu değer 0 ile M-1 aralığındadır.

Başlangıç değeri Denklem 2.2'deki gibi hesaplanır:

$$x_0 = s^2 (mod\ M) \tag{2.2}$$

Rasgele sayılar Denklem 2.3'deki gibi üretilmektedir.

$$x_{n+1} = s^2 (mod\ m) \tag{2.3}$$

Yukarıdaki adımlara göre, p=7 ve q=9 değerleri seçilsin. Buna göre; M=p*q=7*19=133 olur. s çekirdek değeri 100 olarak seçilsin. Bu durumda başlangıç değeri $x_0 = 100^2 (mod 133) = 25$ elde edilir. İkinci sayıyı üretmek için birinci adımdaki kalan sayı yeni çekirdek değeri olur yani $x_0 = 25^2 (mod 133) = 93$ şeklinde olur. Diğer sayılarda benzer işlemler uygulanarak sayılar üretilir.

BlumBlumShub algoritmasında eşlik bitinin dikkate alınmasıyla rasgele bit dizilerinin elde edilmesi sağlanmaktadır. Bu algoritmanın avantajı pek çok kriptografik uygulamalarda kullanılabilir olmasıdır. Algoritmanın dezavantajı ise BBS'nin asal sayılarla işlem yapmasından dolayı yeterli hızda olmaması bu sebeple de modelleme gibi bazı uygulamalarda pek tercih edilmemesidir.

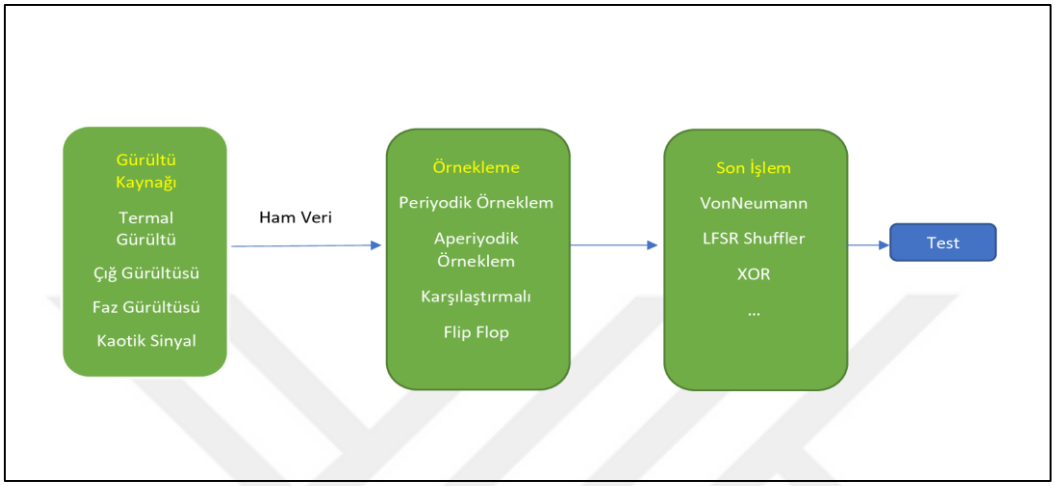
2.5. Gerçek Rasgele Sayı Üreteci (True Random Number Generator)

Genel olarak rasgele sayı üretiminde rassal özelliklerini sağlaması gerekmektedir. Matematiksel fonksiyonlar, bilgisayar programları kullanılarak üretilen rasgele sayılar deterministik bir biçimde çalıştıklarından dolayı tam olarak rasgele sayı üretebilir denilemez. Genel olarak kontrol edilebilen ve bir fiziksel işleyişle çalışan gerçek rasgele sayı üreteçleri kullanılarak elde edilmektedir.

GRSÜ'lerin rasgele sayıların üretilmesi için kaynak olarak fiziksel/doğal olaylardan yararlanılmaktadır. Örneğin; atmosferik gürültü, arka plan (beyaz) gürültü ve elektrik gürültüsü. Bazı değişken kaynaklar diğerlerinden daha rassal olarak düşünülebilir, ancak asıl durum, GRSÜ'lerin bir çeşit fiziksel kaynaktan okuması ve bunu veri işlemeye getirmesidir. Gerçek rasgele sayı oluştururken ayırt edilebilir bir model yoktur ve

gerçekten rasgele olan bir şey üretmek için sayılabilirler. Her bir okumanın sonucunda, sürekli olarak farklı bir çıktıya rastlamaktadır [30].

Genel olarak GRSÜ tasarım diyagramı Şekil 2.2'de gösterilmektedir.



Şekil 2.2. GRSÜ Diyagramı

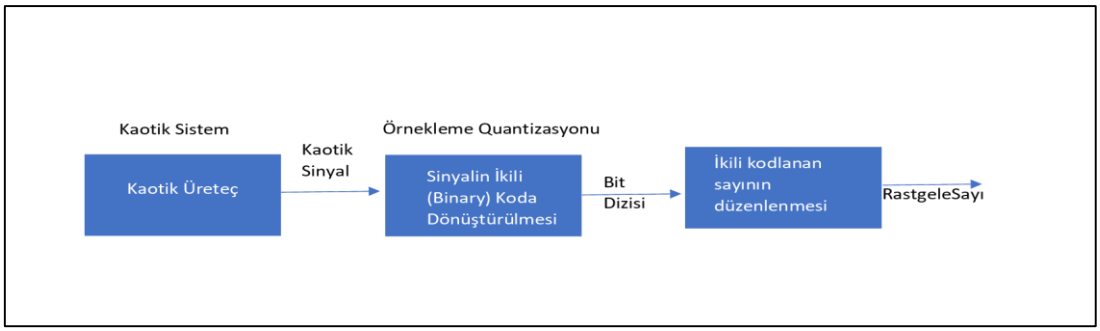
Literatürde gürültü kaynağı olarak termal gürültü, çığ gürültüsü, faz gürültüsü gibi kaynaklar bulunmaktadır. Bu kaynaklardan elde edilen ham veriler örnekleme ünitesine verilmektedir. Burada kullanılan yöntemler periyodik, aperiyodik vb. yöntemlerdir. Burada elde edilen verilere son işlem uygulanmaktadır. Son işlem uygulamasındaki amaç istatistiksel zayıflıkları ortadan kaldırmaktır. Son işlem için bir çok yöntem mevcuttur. Bunlardan biri eski ve pek çok çalışmada uygulanan VonNeuman yöntemidir. Bu yöntem en eski ve basit bir son işlem yöntemidir. Sayılardaki düzensizlikleri geliştirmek için pek çok zaman kullanılmaktadır. Burada 0 ve 1 sayıları bu son işlemle elde edilir. Örneğin; GRSÜ tarafında üretilen bir (1,0) dizisi için son işlemde çıkışı 1 olacaktır. Eğer (0,1) olması durumunda çıkışı sıfır olacaktır. Her iki değer 00 ya da 11 olması durumunda olursa o zaman çıkışı yoktur olacaktır. VonNeumann çıkış hızı üreticin çıkışına bağlı olduğundan sabit olmamaktadır. Bu yöntem sadece GRSÜ tarafından üretilen dizileri değerlendirmektedir. Bir başka son işlem yöntemi olan XOR işlemi de yaygın olarak kullanılan post processing yöntemidir. Burada da (0,1) ve (1,0) dizisinde XOR son işlem çıkışı 1 olacaktır. Diğer iki durumda yani (0,0) ve (1,1) olması durumunda çıkışlar 0 olacaktır. Son işlem yönteminden sonra üretilen sayıların testleri yapılmakta ve bahsedilen genel adımlarla süreç tamamlanmış olacaktır.

GRSÜ kaynaklarının SRSÜ'ye göre tahmin edilememe oranının yüksek olduğu belirlenmiştir. Rasgele sayı kaynaklarının çalışma durumları her ne kadar doğru olsa da üretilen sayıların problem oluşturmaması için kontrollerin yapılması gerekmektedir. Sıcaklık değişimleri, güç kaynağının gürültüsü, tasarlanmış cihazın yaşı, manyetik ve farklı dış alan etkileri devrenin kutuplanmasına neden olmaktadır. Problemin devam ettiği durumda sayı dizileri post processing işlemine girerek sayıların kalitesi artacaktır [31, 32]. Yazılım tabanlı üreteçlerde genel olarak mouse hareketleri gibi bilgisayar odaklı kaynaklardır ve bunlar gürültü kaynağı olarak kullanılmaktadır. Yazılım tabanlı RSÜ'leri gerçeklemek, donanım tabanlılara göre daha güçtür. Dış etkenlere karşı güvenirliği donanıma göre düşük olabilmektedir [25].

2.5.1. Kaos Tabanlı Rasgele Sayı Üreteci

Kaos sistemleri gürültü benzeri sinyaller üretmekte, periyodiklik özellikleri sergilemektedirler. Özellikle sinyallerinin gürültü özellikleriyle benzer durumlara sahip olmasından dolayı son dönemlerde rasgele sayılar, haberleşme, kriptografi, elektrik elektronik mühendisliği gibi alanlarda kaos sistemi de kullanılmaya başlanmıştır.

Genel olarak kaos tabanlı gerçek rasgele sayı üreteci yapılarının geliştirilmesi için çalışmalar yapılmaya özen gösterilmiştir [33]. Kaos tabanlı bir gerçek rasgele sayı üreteci yapısı Şekil 2.3'deki gibidir:



Şekil 2.3. Kaos Tabanlı RSÜ Diyagramı

Kaos temelli yapılar analog yapıya alternatif olarak etkilidir. Ama analog sinyallerde kullanılan GRSÜ sistemlerinin verici ve alıcı arasındaki senkronize durumu oldukça zor

olmaktadır. Bu devrelerde güçlü kaynaklar yerine fiziksel zayıf kaynaklar kullanılmaktadır. Kriptografi, haberleşme vb. uygulamalarında kullanılan kaotik GRSÜ'lerde nondeterministik güçlü kaynaklar kullanılmaktadır.

Kaotik temelli GRSÜ'lerde kullanılan haritalar mevcuttur ve bunlardan bazıları quadratic, tent ve logistic maptir. Tent map, gerçek rasgele sayı üretiminde kaos temelli uygulamalarda kullanılan dinamik haritalardandır. Haritanın yapılan çalışmalarda düzenli yoğunluk dağılımı ve asimptotik durumlarda kararlı davranışlar gösterdiği tespit edilmiştir. Bu haritada bit üretme süreci pek kolay olmamaktadır. Bu nedenle beklenileni karşılaşmadığı durumlarda idealleştirmek için gerekli olan son işlem karmaşık devre veya azalan bit oranıyla sonuçlandırılmaktadır [34]. Kuadratik harita kaotik sistemin bir örneğini oluşturmaktadır. Genel olarak klasik bir kuadratik haritanın kaos gibi zor bir sistem içinde durumları kolaylaştırma özelliği vardır. Haberleşme ve sinyaller ile ilgili çalışmalarda oldukça önemlidir. Diğer dinamik haritalarda kaotik sistemlerde etkin bir şekilde kullanılmaktadır.

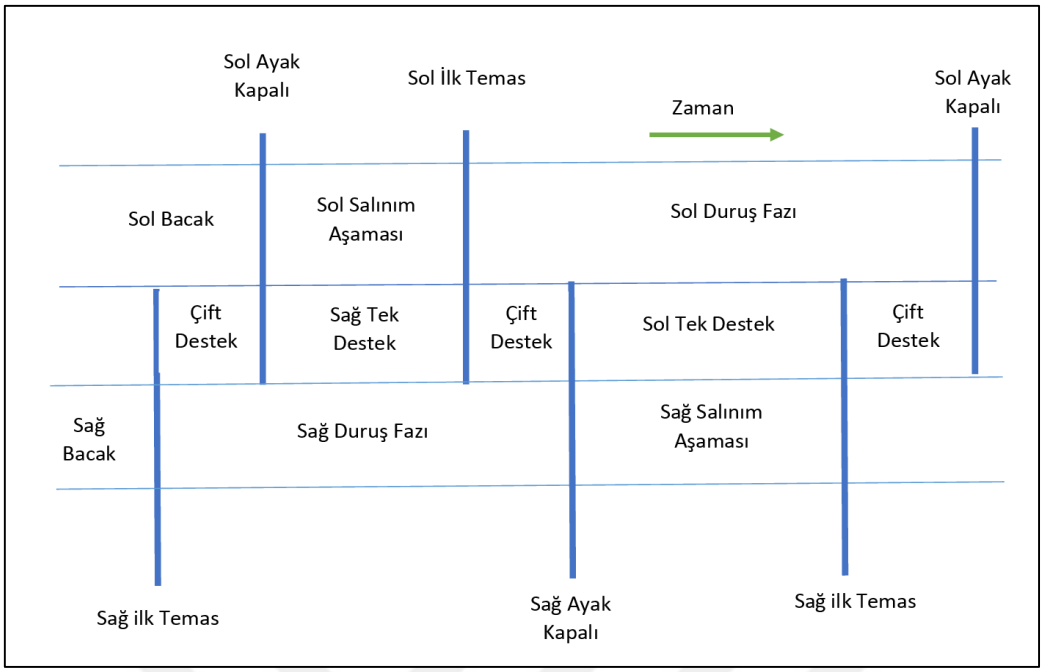
3. İNSAN HAREKET BİYOMEKANİĞİ

İnsan biyomekaniği, özellikle son 50 yılda ilgisi hızla artmaya başlamaktadır. Yürürken insan vücudunun biyomekanik parametrelerinin analizi, verilerinin ölçülmesi gibi birçok bilimsel çalışmalar vardır [35].

Yürüme, insan yer değiştirmesinin en temel yollarından biridir. Bilinçsiz bir şekilde yapıldığı için kolay bir hareket gibi görünmektedir, ancak vücudun gerçekleştirdiği en zor manevralardan biridir. Yürümenin amacı, vücut ağırlığının düz ve düzgün olmayan bir arazide güvenli ve etkili bir şekilde aktarılmasıdır. Yürüme sırasında tüm biyomekanik sistem, tüm dış kuvvetlerin yer değiştirme sırasında dengeyi sağlamak için olası etkileri hafifletmek istenilmektedir. Vücudun güvenli ve etkili bir şekilde sağlanması için, yürüyüş sırasında aşağıdaki adımları gerçekleştirilmektedir [35].

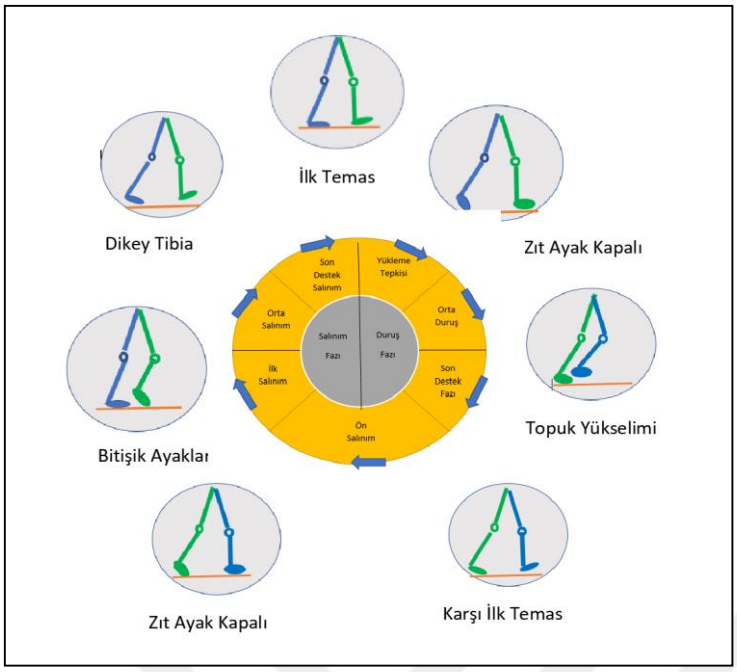
- Vücudun dik ve dengeli durması
- Ayakların zemin ile mesafesini korumak ve yere nazik bir temas sağlamak için alt ekstremitenin kontrol edilmesi
- Hareket yönündeki hızı kontrolü için mekanik enerji oluşumunu takibi.
- Vücudun dengesi için şok kuvvetinin etkileşimi şeklindedir.

Tüm bu özellikler yürürken vücut hareketlerinin bir parçasıdır. Hareket sırasında tüm vücut parçaları, kontrol kuvvetlerinin ve torklarında düzgün ve bilinçli olmayan bir duruma dönüşmesiyle beraber harekete başlamaktadır. Yürüme, vücudun dik ve ileri hareketinde bacağın hem sol hem de sağ olarak birinin diğerini destekleyen bir süreci ifade etmektedir. Burada yürüyüş döngüsü, birbirini izleyen tekrarlar arasındaki zaman aralığı olarak tanımlanmaktadır. Ayak yere temas ettiğinde başlar. Yürüyüş döngüsü Şekil 3.1'de gösterilmektedir [35].



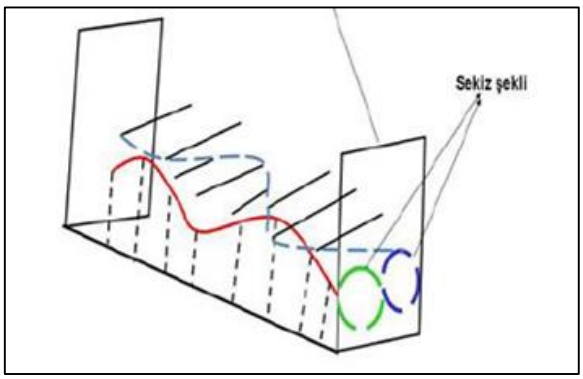
Şekil 3.1. Bir hareket döngüsü sırasında tek ve çift desteğin zamanlaması [35]

Duruş fazı; her iki ayağın yere temasına kadar olan evreleri tanımlanmaktadır. Bu fazın içindeki evreler; Loading response (Yükleme Tepkisi), Mid-stance (Orta Duruş), Terminal stance (Son destek fazı), Pre-swing (Ön salınım) şeklindedir. Salınım aşaması; sadece tek bir ayağın yere temasın kadar olan adımları tanımlamaktadır. Bu fazın da içindeki evreler; (Initial Swing) İlk salınım, (Middle Swing) Orta salınım, (Terminal Swing) Son destek salınımı şeklindedir. Bu aşamalar sadece bir bacak için sunulmaktadır. Diğer bacak aynı döngüyü yerine getirir ancak zaman içinde yer değiştirmektedir. Bu yürüyüş döngüsü, bir insanın yürüyüş sırasında izlediği tekrarlayan bir modeldir [35]. Yürüyüş döngüsünün alt fazlarını sağ ayağa göre ayıran olaylar Şekil 3.2'de gösterilmektedir.



Şekil 3.2. Tek bacak yürüyüşü sırasında bacakların sağ bacağın pozisyonu (yeşil renk)

Yürüme hızının artması sonucu koşma döngüsü oluşmaktadır. Koşma döngüsünde de iki ayağın da yerle temas etmediği iki adet süzülme dönemi yer alır. Hızlanma durumunda basma fazı kısalırken, salınım fazı uzamaktadır. Her iki durumdaki amaç vücudu istenilen hız ve doğrultuda, farklı yönlerde hareketi gerçekleştirmektedir. Bu işlem sırasında ayak, kol ve gövdedeki eklemler, bağlar aktif olarak kullanılmaktadır [35]. Hem yürüme hem de koşma sırasında ağırlık merkezi, dikey düzlemde aşağı yukarı, yatay düzlemde ise sağa-sola hareket etmektedir. Bu hareketlerin birleştirilmesi sonucu hareket örüntüsü oluşur. Şekil 3.3'de hareket esnasında ortaya çıkan örüntüyü göstermektedir. Buna bağlı olarak da x-y-z eksenindeki ivme ve konum sürekli değişkenlik göstermektedir.



Şekil 3.3. İnsan hareketi esnasında ortaya çıkan örüntü [35]

4. SENSÖRLER VE GPS

4.1. Sensör Nedir?

Sensörler genel olarak farklı türdeki birçok elektronik cihazda bulunan yerleşik aygıtlardır. Sensörleri bir canlının duyu organlarına benzetebiliriz. Bir canlının sensörleri olan duyu organı yaşamı için önemliliği varsa cihazlardaki sensörler de o cihaz için önem ifade etmektedir. Birçok elektronik cihazda olduğu gibi akıllı mobil cihazlarında büyük bir kısmında hareket sensörleri bulunmaktadır. Cihazların türüne göre farklı tip sensörler bulunmaktadır. Mobil cihazlara eklenen sensörler mevcut özelliklerinin dışında farklı niteliklerde özellikler sunmaktadır. Bu özelliklerden bazıları;insan aktivitelerini tanıma,davranışlarını izleme ve raporlama, nabız kontrolü gibi işlemler gerçekleştirmektedir. Mobil cihazlarda bulunan bazı sensörler ve genel açıklamaları Tablo 4.1'deki gibidir:

Tablo 4.1. Mobil Cihazlarda Kullanılan Sensörler [36].

Sensör	Algılanan Değer	Anlam	Açıklama
Accelerometer	3	X,Y,Z eksenleri sıralamasıyla	m/sn^2 cinsinden SensorManager. GRAVITY_XXX sabitleriyle anlamlandırılır.
Gyroscope	3	Azimut,Yunuslama,Yuvarlanma	Cihazın Yönetimi
Light	1	Işık	Lux cinsinden SensorManager.LIGHT_XXX sabitleriyle anlamlandırılır.
Magnetic Field	3	X,Y,Z eksenleri sıralamasıyla	MikroTesla cinsinden EMF
Orientation	3	X,Y,Z eksenleri sıralamasıyla	Yönelim
Pressure	-	Basınç	Kilopascal cinsinden
Proximity	1	Mesafe	Metre cinsinden
Temperature	1	Sıcaklık	Santigrat

Sensörler dışsal veriler algılayan aygıtlardır. Sensörler algıladıkları verileri elektronik cihazlarda işleyebilmesi için elektrik sinyallerine dönüştürmesi gerekmektedir. Bu dönüşümü transdiser denilen aygıtlar ile gerçekleştirmektedir. Sensör ve transdiser arasındaki farkı ayırt etmek çoğu zaman güçtür. Bu bakımdan ayırt dilmesi için ayrıntılı teknik bir araştırma yapılması gerekmektedir [36].

4.2. Android Sensörleri ve Genel Yapısı

Her cihazın sensör yapısı aynı olmamaktadır. Ama akıllı telefonlar üzerindeki sensörler çoğunlukla belirli sensörleri kullanmaktadır. Genel olarak Tablo 4.1'deki sensör türlerini barındırıyor diyebiliriz.

Genel olarak Android sensörleri, MEMS yani Mikro elektro-mekanik sensörler olarak adlandırılan bazı tekniklerle üretilmiş sensörlerdir. MEMS sensörü, tasarımlarında fiziksel olarak hareket eden veya titreşen bir kısmını birleştirenler anlamına gelmektedir. Mikro elektro-mekanik sensörler bilgisayar çiplerinin imalatındaki bazı teknikleri kullanarak genellikle silikon çiplerde küçük çapta üretilmiş sensörlerdir. Örneğin basınç sensörü, ivme ölçer gibi sensörler MEMS sensörleridir [37].

Sensörler için atıfta bulunulmuş, iki tipte incelenmiştir. Bunlar ham sensörler ve sentetik ya da sanal sensörlerdir. Ham sensörler Android cihaz içerisinde gerçek fiziksel bileşene içermektedir. Bu sensörler ham veri vermektedir. Sentetik sensörler ise uygulama kodu ile düşük seviyeli cihaz bileşenleri arasında sensörlerin ham verilerini birleştirerek ya da ham sensör verilerini değiştirerek bir soyutlama katmanı oluşturmaktadır. Sentetik sensörler iyi bir yöntemle belirlenebilmesi için öncelikle sensör verilerini okumaktadır. Örneğin manyetometre, ivme ölçer gibi ek sensörler kullanmadan jiroskop verilerini entegre ederek sensör verilerini okurken aynı zamanda değiştirebilmektedir. Sentetik sensörler de sensör tipi önemli değildir. Programcı sensör API'lerini kullanarak her tip sensöre aynı durumda erişebilmektedir. Bazı ham sensörler ve sentetik sensör tipleri Tablo 4.2'de verilmektedir [37].

Tablo 4.2. Ham ve Sentetik Sensörler

Ham Sensörler	Sentetik Sensörler
‰Sensor.TYPE_LIGHT	‰ Sensor.TYPE_ROTATION_VECTOR
‰ Sensor.TYPE_PROXIMITY	‰ Sensor.TYPE_LINEAR_ACCELERATION
‰ Sensor.TYPE_PRESSURE	‰ Sensor.TYPE_GRAVITY
‰ Sensor.TYPE_TEMPERATURE	‰Sensor.TYPE_ORIENTATION
‰ Sensor.TYPE_ACCELEROMETER	
‰ Sensor.TYPE_GYROSCOPE	
‰ Sensor.TYPE_MAGNETIC_FIELD	
‰Sensor.TYPE_RELATIVE_HUMIDITY	
‰ Sensor.TYPE_AMBIENT_TEMPERATURE	

Genel olarak Android sensör yapısı belirli sınıflar ve ara yüz içermektedir. İlk olarak sensörleri yöneten bir sistem servisi tanımlaması yapılmalıdır. Android sensörlerde cihaz donanımından sensör bilgilerini edinmek için sensör API'leri bulunmaktadır. Bu API'ler cihazdan sensör bilgilerini istemek ve işlemek için sınıflardan oluşmaktadır. API'ye giriş olarak bir uygulamanın sensör bilgisi talep etmesine ve sensör verilerini almak için kaydolmasına izin veren SensorManager sınıfıdır [37].

SensorManager, donanım sensörlerine bir uygulama erişimi sağlayan Android sistem servisidir. SensorManager uygulamaların sensörle ilgili olayları kaydetmesine ve kaydını silmesine izin vermektedir. Kayıt olduktan sonra, uygulama donanımdan sensör verilerini alır. Kaydedilme işlemi yapıldığında sensör verileri, belirli bir sensörden üretilen bilgileri içeren bir SensorEvent biçiminde bir SensorEventListener öğesine gönderilmektedir. Sensör verileri için bir uygulamanın kaydedilmesine izin vermenin yanı sıra, SensorManager ayrıca sensör verilerini işleyen yöntemler de sağlamaktadır [37]. SensorManager sınıfı içinde belirli bir sensöre erişmek yani sensör tipini belirlemek için uygulanacak yapı Şekil 4.1'de gösterilmektedir.

```
public SensorActivity() {
    mSensorManager = (SensorManager)getSystemService(SENSOR_SERVICE);
    mAccelerometer = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
}
```

Şekil 4.1. Sensör Tipi Belirleme

Bir uygulamanın sensörden bilgi alması içim SensorEvents almadan önce bir SensorEventListener uygulanması gerekmektedir [37]. SensorEventListener bir

interfacedir. Sensörden elde edilen veri değiştiğinde onSensorChanged() ve doğruluk değiştiğinde çalışan onAccuracyChanged() isimli iki callback fonksiyon içermektedir. Bu fonksiyonla bir sınıf ya da inner type ile implemente edilir, bu interface'i implemente eden türde nesne registerListener() fonksiyonuna parametre olarak geçmektedir. Böylece sensör verileri elde edilmektedir [36]. SensorEventListener içindeki bilgi alma fonksiyon yapısını gösteren kod parçası Şekil 4.2'de gösterilmektedir.

```
public class MainActivity extends AppCompatActivity implements SensorEventListener{

    private SensorManager sensorManager;

    public void onSensorChanged(SensorEvent event) {

        float[] values=event.values;

          //...
```

Şekil 4.2. Sensör Bilgilerini Alma Fonksiyonu

Bu kullanımdan sonra fonksiyon parametreleri SensorEvents türünde sensörün algıladığı verileri almaktadır. SensorEvent sınıfı veriler kayan noktalı formatında bir dizi referansını içermektedir. Genel olarak açıklanan sensörlerin genel kod gösterimi Şekil 4.3'deki gibidir:

```
public class MainActivity extends AppCompatActivity implements SensorEventListener {

    private final SensorManager xSensorManager;
    private final Sensor xAccelerometer;


    public MainActivity(){

        xSensorManager=(SensorManager)getSystemService(SENSOR_SERVICE);
        xAccelerometer=xSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);


    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {

    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int i) {
```

Şekil 4.3. Sensörler Android genel kod gösterimi

Sensör yapısından veri değerlerini ifade etmek için genelde standart 3 eksenli koordinat sistemini kullanmaktadır. Burada, X ekseni yatay ve sağ tarafı, Y ekseni dikey ve yukarı

doğru ve Z ekseni de ekranın dış yüzüne doğru işaret etmektedir. Hızlanma sensörü, Yerçekimi sensörü, Jiroskop, Doğrusal hızlanma sensörü, Jeomanyetik alan sensörü gibi sensörler üç eksenli koordinat sistemini kullanmaktadır.

4.2.1 Accelerometer (İvme Ölçer)

Accelerometer genel olarak mekanik uyarıma ya da eylemsizlik kuvvetine bağlı olarak oluşan fiziksel ivmeyi ölçmektedir. Yani hızın zamana göre değişimidir. İvme ölçer titreşim, dönme, eğim, çarpışma, yer çekimi ölçümlerinde etkin olarak kullanılmaktadır. İvme ölçer mekanik enerjiyi elektrik enerjisine dönüştürmektedir. Newton yasalarını kullanarak mekanik bir model oluşturulması bu sistemin anlaşılmasını kolaylaştırmaktadır [38].

Mobil cihazların birçoğunda ivme ölçer bulunmaktadır. Telefonlardaki ivme ölçer telefonun hareketlerini eksen tabanında takip etmektedir. Örneğin telefon kamerası yatay ve dikey konumuna getirilirken konuma göre ayarlanmasını ivme ölçer sağlamaktadır. İvme ölçer telefon ayarlarının yanı sıra indirilen oyun, video, sağlık vs. uygulamalarda bazı aktivitelerinde sensör kullanımını içermektedir. Akıllı telefonlardaki fitness uygulaması içerisinde bulunan adım sayar eklentisi ivme sensörünü kullanmaktadır. Telefondaki titreşim sayısına göre adımı sayar ve adım sayısını belirlemektedir. Accelerometer sensörü hareket sensörleri grubundadır. Android Accelerometer sensörü yapısında Bölüm 4.2.'de belirtilen genel adımlar oluşturulmaktadır. Dinleyicilerin etkin olması değerlerin erişimi için oldukça önemlidir. Daha sonra x,y,z ivme koordinatlarını almamız gerekmektedir. Bunun için eventvalues ile gerçekleştirilmektedir. İvme ölçerde hareketlerimizdeki anlık değişimlere bağlı ivme değişim metrikleri hesaplanabilmektedir. Accelerometer Android kod yapısı Şekil 4.4'deki gibi gösterilmektedir.

```
public class MainActivity extends AppCompatActivity implements SensorEventListener{

    private SensorManager sensorManager;

    @Override
    protected void onDestroy() {
        super.onDestroy();
        sensorManager.unregisterListener(this);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        sensorManager=(SensorManager)getSystemService(SENSOR_SERVICE);
        sensorManager.unregisterListener((SensorEventListener) this,sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER));
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
if(event.sensor.getType()==Sensor.TYPE_ACCELEROMETER){

    float[] values=event.values;
    float x=values[0];
    float y=values[1];
    float z=values[2];

    Log.d( tag: "MainActivity",String.format("x: %f y: %f z:%f",x,y,z));
```

Şekil 4.4. Android Accelerometer kod yapısı

4.3. GPS Nedir ve Kullanımı

İngilizce karşılığı Global Positioning System olan ve Küresel Yer Belirleme Sistemi anlamına gelen GPS, kodlanarak oluşturulmuş bilgiler gönderen bir uydu ağıdır. GPS, uydular arasındaki mesafeyi ölçmekte ve dünya üzerindeki yer tespiti yapmaktadır. GPS, ABD Savunma Bakanlığına ait olan bir sistemdir. GPS sistemi yörüngede sürekli olarak dönen 24 uydudan oluşmaktadır [39]. Bu uydular belli bir prensibe göre çalışmaktadır. Sistemin çalışma prensibi şu şekildedir: Yörüngede bulunan uydularda atomik saatler bulunmaktadır. Bu saatler yeryüzündeki saatler ve uydularla senkronize bir şekilde çalışmakta herhangi bir sapma durumunda günlük düzeltme işlemi yapılmaktadır. GPS uyduları saat bilgisini ve yörüngede bulundukları konumu sürekli Dünya'ya göndermektedir. Herhangi bir GPS alıcısının çalıştırılması durumunda kapsamı alanındaki uydulardan en az üçü sinyallere gereksinim duymaktadır. GPS alıcısı saat bilgilerinin mutlak zamandan ne kadar sapma yaptığını bulur ve her uydudan mesafeleri öğrenerek edindiği bu bilgilerle konumu hesaplamaktadır [40].

Günümüzdeki akıllı mobil telefonlar, navigasyonlar vs. bazı cihazlarda GPS alıcısını sağlayan harita uygulamaları bulunmaktadır. Bu harita uygulamasının kullanımı için internet bağlantısı olması gerekmektedir. Uygulama konumumuzu göstermekte herhangi

bir yeri seçilmesiyle arama yapabilir, hedeflenen yere ilerlememiz için gerekli rota bilgilerini hesaplar ve en kısa ya da en uygun yolu bize göstermektedir. Yol, konum gibi bilgiler dışında trafik bilgisi gibi bilgilere de ulaşılabilmektedir.

4.3.1. Android GPS Kullanımı

Android cihazların büyük bir bölümünde GPS alıcısı bulunmaktadır. GPS verilerine erişmek için telefondaki konum özelliğinin aktif olması gerekmektedir. Telefondaki pek çok uygulamada uygulama verileri ya da bilgileri dinleme ve değerlendirmesi için belli izinlerin verilmesi gerekmektedir. GPS uygulamalarında da söz konusu bu durum vardır.

Android GPS için izin işlemi AndroidManifest bölümüne ACCESS_FINE_LOCATION olarak tanımlanmaktadır. Bu izin, yüksek doğruluğa sahip konum bilgisi etmek için gereklidir. Bu izin dışındaki farklı izinlerde mevcuttur ve bazı yöntemler daha düşük doğruluğa sahiptir [36].

Android üzerinde GPS bilgilerini alabilmek için bazı tanımlamalar yapılması gerekmektedir. İlk olarak GPS için tanımlanan bir servis sınıfının tanımlanmasıdır. Tanımlanacak sınıf ilk olarak LocationManager sınıfıdır. LocationManager sınıfı bir sistem servisi olarak nitelendirilmektedir. Bu sınıf coğrafi lokasyon bilgilerinin alınmasını sağlayan çekirdek sınıftır. Sınıf içinde nesne doğrudan oluşturulmaz. Oluşturulma işlemi Şekil 4.5'deki gibi tanımlanması gerekmektedir:

```
LocationManager mng=(LocationManager)
        this.getSystemServices(Context.LOCATION_SERVICE);
```

Şekil 4.5. GPS LocationManager sınıfında servis gösterimi

Bu sınıf içerisinde cihazın belirlenen zamanda ve belli miktarda yer değiştirmesi halinde elde edilen lokasyon verilerinin alınması için requestLocationUpdates() fonksiyonu kullanılmaktadır. Bu fonksiyon içerisindeki sıklıkla kullanılan parametreler ve yapısı Şekil 4.6'deki gibi tanımlanmaktadır.

```
void requestLocationUpdates(){
        String provider; /* kullanılacak provider'ın ismini getirme */
        long minTime; /* Lokasyon verisi almada minimum zaman aralığını belirtme */
        float minDistance;  /* Lokasyon verisi almada minimum mesafeyi belirtme */
        LocationListener);  /* Lokasyon dinleme için bir referans */
```

Şekil 4.6. GPS Location parametreleri

Android GPS lokasyon bilgilerini modellemek için Location sınıfı kullanılmaktadır.Bu sınıfta enlem boylam v.b gibi bilgileri elde edebilecek fonksiyonları bulunduran sınıftır. Bunlardan bazıları enlem bilgisi için getLatitude, boylam bilgisi için getLongitude ve cihazın hız ile bilgisi için ise getSpeed fonksiyonlarıdır [36].

GPS için Android kod genel yapıları Şekil 4.7'deki gibi gösterilmektedir.

```
import ...

public class MainActivity extends AppCompatActivity {

    private LocationManager mng;
    private TextView txt1, txt2;
    private GpsReceiver receiver;

    private void init() {

        receiver = new GpsReceiver();
        mng = (LocationManager) this.getSystemService(Context.LOCATION_SERVICE);
        if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_G

            return;
        }
        mng.requestLocationUpdates(LocationManager.GPS_PROVIDER,  minTime: 1000L,  minDistance: 1.0F, receiver);
       txt1=(TextView)findViewById(R.id.txt1);
       txt2=(TextView)findViewById(R.id.txt2);

    }
```

```
public class GpsReceiver implements LocationListener{


    @Override
    public void onLocationChanged(Location location) {

        if(location!=null){

            double enlem=location.getLatitude();
            double boylam=location.getLongitude();
            double irtifa= location.getAltitude();
            float hiz=location.getSpeed();
            txt1.setText(String.format("Enlem:%f-Boylam:%f",enlem,boylam));
            txt2.setText(String.format("Hız:%f-İrtifa:%f",hiz,irtifa));
        }
        else{

            Toast.makeText( context: MainActivity.this,  text: "Konum Bilgisi alınamıyor.",Toast.LENGTH_LONG).show();



        }

    }
```

Şekil 4.7. Android GPS Uygulama Kod Gösterimi

5. TESTLER

Rasgele sayı üreteçlerinin ürettiği sayıların güvenirliğini ölçmek için belirli istatistiksel testler bulunmaktadır. Literatürde rastgele sayıların istatistiksel özelliklerini belirlemek için NIST, Otokorelasyon v.b testler kullanılmaktadır. Bu tezde Skala Indeks, Otokorelasyon ve NIST testleri uygulanmış ve bu bölümde açıklanmıştır.

5.1. NIST Testi

NIST testi, donanım ve yazılım tabanlı rasgele sayı üreteçlerinde, simülasyon, kriptografik vb. uygulamalarda farklı istatistiksel durumların incelenmesinde rasgele sayıların istatistiksel testini yapmak amacıyla kullanılan en yaygın testlerdendir. Bu testler bir dizide bulunabilecek rastlantısal olmayan durumlara odaklanmaktadır. Bu test suiti 15 testten oluşmaktadır.

5.1.1. Frekans (Monobit) Testi

Bu test, sıfırların ve tüm dizi için birlerin oranını belirlemektedir. Testin amacı bir sıradaki birlerin ve sıfırların sayısının gerçekten rasgele bir sıra için beklenildiği gibi olup olmadığının belirlenmesidir. Test bir dizideki 1 ve 0 sayılarının aynı olması gerektiğini değerlendirmektedir. Bu testin en önemli noktası sonraki tüm testlerin bu teste bağlı olmasıdır. Bu testin belli parametreleri şu şekildedir: n bit dizisi uzunluğunu, $\mathcal{E}$ test edilen GRSÜ veya SRSÜ tarafından üretilen bit dizisini ifade etmektedir. Bu fonksiyon $\varepsilon_1, \dots, \varepsilon_n$ yapıdadır. Bu testin adımları aşağıdaki gibidir:

- ± 1 Dönüşümü; Sıfırlar ve giriş dizisinin değerleri -1 ve +1 değerlerine dönüştürülmekte ve birlikte $S_n = X_1 + \dots + X_n$, $X_i = 2\varepsilon_1 - 1$ değeri üretmek için kullanılmaktadır.
- Test istatistiği olan S_{obs} Denklem 5.1'deki gibi hesaplanmaktadır.

$$S_{obs} = \frac{|s_n|}{\sqrt{n}} \quad \textbf{(5.1)}$$

Rasgelelik ölçüsü değeri olarak ifade edilen P_{value} değerleri Denklem 5.2'deki gibi hesaplanmaktadır.

$$erfc(\frac{S_{obs}}{\sqrt{2}}) \quad \textbf{(5.2)}$$

Burada efrc tamamlayıcı hata işlevini göstermektedir. Eğer hesaplanan $P_{value} < 0.01$ ise dizinin rasgele olmadığı sonucuna varılmakta aksi halde dizinin rastgele sonucuna varılmaktadır.

5.1.2. Blok Frekans Testi (Frequency Test with in Block)

Bu testte hedeflenen M-bit blokları içindekilerin oranını hesaplamaktır. Testin amacı M-bit bloğundakilerin sıklığının rastgele bir varsayım altında istenildiği gibi yaklaşık M/2 olup olmadığını belirlemektir. M=1 blok boyutu için bu test, 1 Frekans testini test etmek üzere bozulmaktadır.Bu testin adımları aşağıdaki gibidir:

- n, n bit dizisi uzunluğunu, $\mathcal{E}$ test edilen GRSÜ veya SRSÜ tarafından üretilen bit dizisini ifade etmektedir. Bu fonksiyon $\varepsilon_1, \dots, \varepsilon_n$ yapıyı ifade etmektedir.
- Her M-bit bloğundaki $\prod_i$ oranını $1 \leq i \leq N$ için Denklem 5.3'de kullanılmaktadır.
- X^2 değeri Denklem 5.3'deki formülle hesaplanmaktadır.

$$X^2(obs) = 4M \sum_{i=1}^{N} (\pi_i - \frac{1}{2})^2 \quad \textbf{(5.3)}$$

- P_{value} değerleri Denklem 5.4'deki formülle hesaplanmaktadır.

$$\text{igamc(N/2,}X^2(obs)/2\textbf{)} \quad \textbf{(5.4)}$$

Burada igamc tamamlanmış gamma fonksiyonunu ifade etmektedir.

- Eğer hesaplanan $P_{değeri} < 0.01$ ise dizinin rasgele olmadığı sonucuna varılmakta aksi halde dizinin rastgele sonucuna varılmaktadır.

5.1.3. Akış Testi (Run Test)

Bu test genel olarak, sürekli bir özdeş bit dizisi olduğu ve dizideki çalışma sayısını belirlemektedir. Bit uzunluğunun bir çevrimi, tam olarak ayrı bitlerden oluşur ve zıt değerlerden biraz önce ve sonrasında sınırlanmaktadır.

Bu testin amacı çeşitli uzunluklardaki birlerin ve sıfırların sayılarının rastgele bir sıra için beklenildiği gibi olup olmadığını belirlemektir. Özellikle bu test bu sıfırlar ve birler arasındaki çevrimin çok hızlı mı yoksa çok yavaş mı olduğunu belirlemektedir. Akış testi bir ön koşul olarak bir frekans testi gerçekleştirmektedir.

5.1.4. Bloktaki En Uzun Birlerin Akış Testi (Test for Longest Run of Ones in Block)

Bu testte hedeflenen M-bit blokları içinde en uzun birleri ifade etmektir. Bu testin amacı, test edilen dizinin en uzun birlerin uzunluğunun rastgele bir dizide beklenen en uzun çalışma süresinin uzunluğu ile tutarlı olup olmadığını test etmektir. En uzun zamanda beklenen uzunluktaki düzensizliğin en uzun sıfır uzunluğunun beklenen uzunlukta bir düzensizlik olduğunu ifade etmektedir. Bu nedenle bu test sadece birlerin testi için gereklidir.

5.1.5. İkili Matris Rankı Testi (Binary Matrix Rank Test)

Bu testte hedeflenen tüm dizinin ayrık alt matrislerinin sıralanması durumudur. Testin amacı orijinal dizinin sabit uzunluktaki alt dizileri arasında doğrusal bağımlılık olup olmadığını kontrol etmektir. Bu test DIEHARD testinde de görülmektedir [41].

5.1.6. Ayrık Fourier Dönüşümü Spektral Testi (Discrete Fourier Transform Spectral Test)

Bu testte, Ayrık Fourier dönüşüm tepe yüksekliklerine bağlı hesaplamalar yapılmaktadır. Testin amacı test edilen sıradaki rassallık varsayımından sapmayı gösteren periyodik özellikleri tespit etmektir. Yani %95'i aşan tepe sayısının olup olmadığını tespit etmektir [41].

5.1.7. Örtüşmeyen Şablon Eşleştirme Testi (Non Overplaning Template Matching Test)

Bu testte hedef, önceden belirlenen dizilerin gerçekleştirdiği durum sayısını belirlemektir. Testin amacı periyodik olmayan şablonun oluşumunu içeren jeneratörleri tespit etmektir.

5.1.8. Örtüşen Şablon Eşleştirme Testi (Overplaning Template Matching Test)

Bu test önceden belirlenen hedefin gerçekleşme sayısını tespit etmektedir. Bu test önceki test gibi m-bit kullanmaktadır. Aradaki fark, arama işlemine devam edilmeden önce şablonundaki değerlerin sadece bir bit kayması durumudur.

5.1.9. Maurer Evrensel İstatistiksel Testi (Maurer's "Universal Statistical" Test)

Bu testi Maurer 1992 yılında geliştirdiği testtir. Bu test bit başına akımın entropisi ile ilgisinden bahsetmektedir. Bu test gerçek kriptografik uygulamada önemlidir. Bir şifre sisteminde etkin anahtar boyutu olduğunda test özel bir model ya da istatistiksel hatayı tespit etmek için tasarlanmıştır. Genel olarak çok genel istatistiksel hata sınıflarda herhangi birini testi için tasarlanmıştır [41].

5.1.10. Doğrusal Karmaşıklık Testi (Linear Complexity Test)

Bu test, rasgeleliği test etmek için doğrusal karmaşıklık kullanmaktadır. LFSR'nin en güncel kısmında bu test uygulanmaktadır. Burada L'nin uzunluk bir kaydı her biri bir girişe ve bir çıkışa sahip olan L gecikme elemanlarından oluşmaktadır. LFSR başlangıç dizisi $(\varepsilon_{L-1} \ldots\ldots \varepsilon_0)$, çıktı dizisi $(\varepsilon_L \ldots . +1 \ldots)$ şeklindedir. LFSR başlangıç durumunda LFSR çıktısının olması ikili dizi oluşturduğunu göstermektedir [41].

5.1.11. Seri Testi (Serial Test)

Bu test, istatistiksel dağılımların tek düzeliğini test etmektedir. i_1'den i_m'ye kadar olan bir dizide 2^m olası tüm vektör özellikleri boyunca test çalıştırılmaktadır [41].

5.1.12. Yaklaşık Entropi Testi (Approxiamate Entropy Test)

Bu test, seri testinde olduğu gibi tüm dizi boyunca uyumlu m-bit modellerinin frekanslarını belirlemektir. Testin amacı iki ardışık uzunluktaki m ve m_{i+1} gibi üst üste gelen rastgele bir dizi için istenilen sonuç karşısında frekanslarını karşılaştırmaktadır [41].

5.1.13. Kümülatif Toplam Testi (Cumulative Sums Test)

Bu test dizideki (-1,+1) rakamlarının toplamının maksimum rastgele dolaşımın maksimum sıfırını belirlemektedir. Amaç test edilen kısmi sekansların kümülatif toplamının beklenilene göre çok büyük ve çok küçük olduğunu kabul etmektedir. Rastgele bir sıra için bu rastgele dolaşım sıfıra yakın olmalıdır. Bazı rastlantısal olmayan diziler için bu rastgele sıfırda yapılan adımlar büyük olacaktır.

5.1.14. Rastgele Yürüyüş Testi (Random Excursions Test)

Bu testte hedef, kümülatif toplam rastgele yürüyüşünde tam komşularına sahip döngü sayısıdır. Rastgele yürüyüş döngüsü başlangıçta başlayıp ve orijine geri dönmektedir. Rastgele alınan bir dizi uzunluktan oluşan bir dizi adımdan oluşmaktadır. bu testte amaç belirli bir döngü içindeki duruma göre yürüyüş sayısının rastgele bir sıra için istenilenden farklı olup olmadığını belirlemektir [41].

5.1.15. Rastgele Yürüyüş Varyant Testi (Random Excursions Variant Test)

Bu test, toplam rasgele yürüyüşte yürüyüş sayısını belirlemektedir. Bu testte amaç beklenen sayıda sapmaları tespit etmektir. Bu test başlangıç ve sonuç için 18 testten oluşmaktadır [41].

5.2. Otokorelasyon Testi

Bu test genel olarak sayı dizilerindeki bağımsızlık ilişkilerini inceleyen istatistiksel testlerden biri olup bilgisayar bilimleri, simülasyon vb. birçok çalışmada kullanılmıştır. Otokorelasyon testi rasgele sayılar üzerinde de üretilen sayıların bağımsız bir şekilde

üretilip üretilmediğini test etmek için kullanılmaktadır [42]. Otokorelasyon testi genel gösterimi Denklem 5.5'deki gibi ifade edilmektedir.

$$A(d) = \sum_{i=0}^{n-d-1} a_i \oplus a_{i+d} \tag{5.5}$$

Bu denklemdeki parametreler; n üretilen sayı dizisi, d değeri [1,*(n/2)*] aralığındaki bir sabit tam sayı değeri, $\oplus$ XOR operatörü ve a_i, *i.* sayı dizisi şeklinde ifade edilmektedir. Denklemde test değerinin hesaplanması *A(d)* tam sayı değeri Denklem 5.6'da kullanılmaktadır.

$$X_5 = \frac{2\left[A(d) - \frac{n-d}{2}\right]}{\sqrt{n-d}} \tag{5.6}$$

Bu denklem sonucunda elde edilen test değerleri $|X_5|$ <1.6449 sağladığında bu testin başarılı olduğu ifade edilmektedir.

5.3. Skala Indeks Testi

Skala Indeks testi, istatistiksel analizler olmak üzere farklı çalışmalarda kullanılan testlerdendir.Bu testte rasgele sayılar üzerinde periyodiklik durumları ve buna bağlı dereceleri belirlenmektedir. Literatürde GRSÜ ve SRSÜ çalışmalarında bu test kullanılmaktadır [43,44].

Skala indeks testi analiz işleminde iki teknik birlikte kullanılmaktadır. Bunlar Dalgacık Çoklu Çözünürlük (Wavelet Multi Resolution) ve Sürekli Dalgacık Dönüşümü (Continuous Wavelet Transform) teknikleridir [45]. Bu tekniklerin matematiksel gösterimi Denklem 5.7 ve Denklem 5.8 deki gibidir [43].

$$Wf(u,s) := \langle f, \varphi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t)\varphi_{u,s}^{*}(t)dt \quad \textbf{(5.7)}$$

$$(s) := \|Wf(u,s)\| = \left(\int_{-\infty}^{+\infty} |Wf(u,s)|^2 \, du \right) \quad \textbf{(5.8)}$$

Burada sürekli dalgacık dönüşümünde u ile belirtilen zamanda *s* ve *f* değerleri gösterilmiştir. bir s skalasına bağlı olarak Denklem 5.9'da *f*'nin iç skologramı hesaplaması yapılmaktadır [44].

$$S^{in}(s) := \|Wf(u,s)\|_{j(s)} = \left(\int_{c(s)}^{d(s)} |Wf(u,s)|^2 \, du \right)^2 \quad \textbf{(5.9)}$$

Burada, $J(s) = [c(s), d(s)] \subseteq I, \varphi_{u,s},$ tüm $u \in j(s)$ için Iiçinde yer alan en yüksek alt aralığı ifade etmektedir. $J(s)$ uzunluğunun skalaya bağlı olduğu durumda, farklı ölçeklerdeki iç skalogramın değerleri karşılaştırılamamaktadır. Bu durumda normalize edilmiş tercih edilir ve Denklem 5.10'da gösterildiği gibi tanımlanmaktadır [44].

$$\bar{S}^{in}(s) = \frac{S^{in}(s)}{(d(s) - c(s))^{\frac{1}{2}}} \quad \textbf{(5.10)}$$

Bu adımdan sonra [s_0, s_1] aralığındaki *f*'nin skala indeks değeri Denklem 5.11'deki gibi hesaplanmaktadır.

$$i_{scale} := \frac{S(S_{min})}{S(S_{max})}, \quad \textbf{(5.11)}$$

Burada skala indeks değeri olarak ifade edilen i_{scale} , [0,1] aralığında olmalıdır.

6. METOT VE YÖNTEM

Tez çalışmasında uygulanan GRSÜ yapısı Şekil 6.1'de gösterilmektedir. Genel olarak GRSÜ için önemli bileşenlerinden biri olan gürültü kaynağıdır. Gürültü kaynakları; elektriksel , atmosferik, ses, mouse hareketleri vb. şeklindedir. Tezde uygulanan GRSÜ yapısında insan hareketleri gürültü kaynağı olarak kullanılmıştır. İvme sensörü ve GPS kullanılarak kişinin yürüme, koşma ve durma durumundan değişim verileri elde edilmiştir. Elde edilen verileri Şekil 6.1'de gösterilen örnekleme, normalizasyon ve son işlem uygulanmıştır.

Şekil 6.1. Çalışma GRSÜ Yapısı

6.1. Örnekleme

İvme sensöründen ve GPS sisteminden elde edilen sayılar değişiklik gösterebilmektedir. Elde edilen bu sayılar kayan noktalı sayı formatındadır. Örnekleme işlemi yapılırken periyodik ya da periyodik olmayan işaretler kullanılmaktadır. Tez çalışmasında örnekleme işlemi periyodik olmayan kaotik davranış sergileyen tent map ile gerçekleştirilmiştir. Tent map, gerçek rasgele sayı üretiminde kaos temelli vb. uygulamalarda kullanılan dinamik haritalardandır. Bu harita ile ilgili yapılan çalışmalarda düzenli yoğunluk dağılımı ve asimptotik durumlarda kararlı davranışlar gösterdiği tespit edilmiştir. Tent map matematiksel gösterimi Denklem 6.1'deki gibidir.

$$f(x) = \begin{cases} \mu x_i, if\ x_i < 0.5 \\ \mu(1 - x_i), diğer\ durumda \end{cases} \quad \textbf{(6.1)}$$

Burada, $i \geq 0$ için$x_i \in [0,1]$aralığına sahiptir. Harita [0,1] aralığına dönüştürme yapar ve μ olarak adlandırılan tek bir kontrol parametresi içerebilmektedir. Burada kontrol parametresi [0,2]aralığındadır. x_0, sistemin başlangıç değeridir. Sonraki iterasyonlarda $i \geq 1$ için üretilecek sayılar x_1, x_2, ...olacaktır. Algoritma-1'de tent map yardımıyla üretilecek örnekleme dizisi için sözde kodu gösterilmektedir.

Algoritma-1 Tent map sözde kodu

Başlangıç parametrelerini x_0 , μ ayarla

$for i = 0 \; to \; m$

$if x_0 < 0.5 \; ise$

$f = \mu x_0$

$x_0 = f$

$S_i = 0 \; değilse$

$f = \mu(1 - x_0)$

$x_0 = f$

$S_i = 1$

end if

end

Örnekleme işleminde kontrol ve başlangıç parametresi sırasıyla 1.997 ve 0.32 olarak belirlenmiştir. Üretilen ilk 10 değer B={0,1,1,1,1,0,0,1,0,0}'dir. x ekseninden ivme sensöründen elde edilen kayan noktalı formatındaki değerler alınmıştır. Bu dizi b_x={ 6.18, 1.75, 1.80, 1.20, 3.05, 3.78, 2.41, 5.74, 3.39, 6.20} şeklindedir. Dizilerin örnekleme işareti sonucu Tablo 6.1'de gösterilmiştir.

Tablo 6.1 Tent map ile örneklenmiş işaretlerin elde edilmesi

A	6.18, **1.75, 1.80, 1.20, 3.05,** 3.78, 2.41, **5.74,**3.39, 6.20
B	0, 1, 1, 1, 1, 0, 0, 1, 0, 0
Örneklenmiş İşaret	**-, 1.75, 1.80, 1.20, 3.05, - - 5.74, - -**

6.2. Normalizasyon

Örneklenmiş işaret adımından sonra örneklenen işaretlerin kayan noktalı sayı formatındadır. Örneklenmiş işaret dizisinin normalizasyonu için tam sayıya dönüşümü gerekmektedir.Burada da her bir elemanın tamsayı formatına dönüşümü için 100 ile çarpılması sağlanmıştır. Modüler aritmetik kullanılarak 5 bit ile gösterilmiştir. 1.75 sayısı için 175 tamsayısına dönüştürülmüştür. 5 bitlik sayıya gösterimi a_0=01111 olmaktadır. Elde edilen 5 bitlik a_i 0 ve 1 üretmek için XOR işlemi uygulanmış, c_i dizisi elde edilmektedir. Böylece a_0 için üretilen rastgele sayı c_0=0 olacaktır. Normalizasyon süreci Algoritma 2'de gösterilmektedir.

Algoritma.2 Normalizasyon Süreci

$a = (a_1, \ldots., a_n)X$//Örneklenmiş veriler

$for\ i = 1\ to\ n$

$b_i = b_i * 100$//Tam sayıya dönüştürme

$b_i = a_i\ mod(32)$ // b_i [0-31] aralığında ikili formata dönüştürme

$for\ k = 1\ to\ 5$

$c_i = b_{i0}\ XOR, b_{i1}\ XOR, b_{i2}\ XOR, b_{i3}\ XOR, b_{i4}\ XOR$//XOR işlemi uygula

end

end

6.3. Son İşlem

Genel olarak GRSÜ tarafından üretilen rastgele sayıların istatistiksel zayıflıklarını gidermek için son işlem uygulanmaktadır. Pek çok çalışmada kullanılan son işlem fonksiyonları V.Neuman, XOR, Hash vb. şeklindedir. Bu çalışmada uygulanan son işlem fonksiyonu XOR'dur. c_i dizisinin ardışık bitlerinin XOR çıkış kuralı Tablo 6.2.'de verilmiştir.

Tablo 6.2. Ardışık bitler (XOR)

Ardışık Bit Çiftleri	Son İşlem Çıkışı
00	0
01	1
10	1
11	0

6.4. İstatistiksel Test Sonuçları

Kişinin yürüme, koşma ve durma esnasındaki değişimler ivme sensörü x,y,z düzlemlerindeki ve GPS'in de x,y eksenlerindeki hareketleri alınarak veriler elde edilmiştir. İvme sensörü için eksenlerine bağlı olarak 9 veri seti ve GPS için 6 veri seti oluşturulmuş, örnekleme, normalizasyon ve son işlem aşamalarından sonra test işleminde Skala Index, Otokorelasyon ve NIST testleri uygulanmıştır. Test sonuçları alt başlıklarda tablolar şeklinde verilmiş ve sonuçları ifade edilmiştir.

6.4.1. Skala Indeks Test Sonuçları

Tablo 6.3 'de koşma, yürüme, ve durma esnasında x,y,z koordinatlarındaki ivme değerleri ve x,y koordinatlarındaki konum değerlerinin skala indeks sonuçları gösterilmiştir.

Tablo 6.3. Konum ve ivme için Skala indeks test sonuçları

	İvme$_x$	**İvme$_y$**	**İvme$_z$**	**Konum$_x$**	**Konum$_y$**
Koşma	0.893	0.937	0.891	0.753	0.834
Yürüme	0.945	0.879	0.847	0.955	0.894
Durma	0.882	0.870	0.765	0.881	0.851

Bu çalışmada rasgele sayıların periyodikliğini belirlemek için bu test kullanılmıştır. Genel olarak skala indeks değerlerinin [0,1] aralığında olması gerekmektedir. Tablo sonuçlarına bakıldığında ivme ve konum değerlerine göre en yüksek skala indeks değeri koşma esnasında 0.937, yürüme esnasında 0.995 ve durma için 0.882 olarak tespit

edilmiştir. Bu testte, insan hareketlerinden elde edilen dizilerin nonlineer yapıda olduğu belirlenmiştir.

6.4.2. Otokorelasyon Test Sonuçları

Otokorelasyon testi sayı dizilerindeki bağımsızlık ilişkilerini incelemektedir. Skala indeks testi ile ilişkilendirildiğinde periyodiklik durumuna bağlı olarak sayıların bağımsız bir şekilde üretilip üretilmediği test edilmektedir. Üretilen rastgele sayı dizisi için d tam sayı değerleri d=8, d=10 ve d=13 olarak belirlenmiş ve test sonuçları Tablo 6.4'de verilmiştir.

Tablo 6.4. Otokorelasyon test sonuçları

	d=8			d=10			d=13		
	Koşma	**Yürüme**	**Durma**	**Koşma**	**Yürüme**	**Durma**	**Koşma**	**Yürüme**	**Durma**
İvme$_x$	0.187	-0.091	0.547	-0.879	0.274	0.912	1.602	-0.547	0.347
İvme$_y$	-0.906	0.492	0.911	-0.533	0.347	1.131	-0.493	0.803	0.911
İvme$_z$	-0.479	0.331	0.298	-1.066	1.185	0.124	0.795	0.912	0.196
Konum$_x$	-1.019	0.304	0.565	0.712	0.028	-1.130	-1.425	-0.111	1.002
Konum$_y$	-0.647	0.738	0.400	-0.388	0.101	0.100	0.104	-0.386	0.225

Bu çalışmada üretilen rasgele sayıların 0-1 değişimlerini belirlemek için otokorelasyon testi kullanılmıştır. Otokorelasyon testinde test değerinin $|X_5|$ <1.6449 şartını sağlaması gerekmektedir. Tablo sonuçlarında insan hareketlerinden alınan ivme ve konum değerlerine göre en yüksek test değerleri sırasıyla d=8 için 1.019, d=10 için 1.185 ve d=13 için 1.062'dir. Sonuç olarak bu testte, test değerlerinin istenilen şartı sağladığı elde edilen 0-1 sayı dizisinin birbirleriyle ilişkili olmadığı tespit edilmiştir.

6.4.3. NIST Test Sonuçları

Tez çalışmasında uygulanan testlerden son olarak NIST test suiti kullanılmıştır. İçerisinde 15 test bulunmaktadır [46]. NIST testinde test başarımı için iki önemli parametre bulunmaktadır. Bu parametreler önem seviyesi (θ) ve rasgelelik ölçüsü (P-value)dür. Burada önem seviyesi değerinin 0.01 olarak seçildiği durumda üretilen sayıların testin %99 güven değerine sahip olduğu ifade edilmektedir. Rasgelelik önem seviyesi değeri önem seviyesi değerinden büyük ya da eşit olması durumu test sonuçlarının olumlu olduğu ifade etmektedir. Bu tez çalışmasında test için önem seviyesi aralığı [0.001,0.01] olarak belirlenmiştir. İnsan hareketlerinden elde edilen ivme ve konum değerlerin NIST test sonuçları Tablo 6.5'de gösterilmiştir.

Tablo 6.5. Yürüme, koşma ve durma NIST test sonuçları

		1.Frekans Testi	2.Blok Frekans Testi	3.Akış Test	4. Bloktaki En uzun birlerin testi	5. İkili Matris Rankı Testi	6. Spektral Testi	7. Örtüşmeyen Şablon Eşleştirme Testi	8. Örtüşen Şablon Eşleştirme Testi	9. Maurer Evrensel İstatistiksel Testi	10. Doğrusal Karmaşıklık Testi	11.Seri Testi	12. Yaklaşık Entropi Testi	13. Birikimli Toplam Testi	14. Rasgele Yürüyüş Testi	15. Rasgele Yürüyüş Varyant Testi
YÜRÜME	$İvme_x$	0.236	0.665	0.907	0.231	0.741	0.877	0.941	0.496	0.125	0.121	0.956	0.459	0.163	0.472	0.428
	$İvme_y$	0.729	0.265	0.333	0.943	0.481	0.046	0.152	0.633	0.429	0.631	0.612	0.353	0.881	0.678	0.562
	$İvme_z$	0.121	0.906	0.014	0.965	0.741	0.747	0.734	0.838	0.580	0.849	0.175	0.017	0.235	0.218	0.302
	$Konum_x$	0.437	0.558	0.710	0.505	0.291	0.373	0.666	0.488	0.568	0.320	0.437	0.227	0.386	0.580	0.503
	$Konum_y$	0.569	0.751	0.499	0.582	0.793	0.859	0.746	0.108	0.351	0.238	0.569	0.321	0.241	0.624	0.621
KOŞMA	$İvme_x$	0.984	0.251	0.066	0.913	0.539	0.271	0.532	0.841	0.416	0.300	0.894	0.662	0.244	0.872	0.784
	$İvme_y$	0.462	0.182	0.018	0.371	0.550	0.363	0.156	0.627	0.365	0.165	0.462	0.244	0.109	0.783	0.706
	$İvme_z$	0.035	0.464	0.995	0.272	0.161	0.463	0.814	0.340	0.652	0.567	0.109	0.957	0.260	0.049	0.128
	$Konum_x$	0.052	0.625	0.823	0.041	0.693	0.596	0.607	0.886	0.279	0.985	0.143	0.734	0.993	0.097	0.186
	$Konum_y$	0.423	0.382	0.100	0.972	0.039	0.481	0.686	0.887	0.547	0.915	0.208	0.115	0.187	0.767	0.682
DURMA	$İvme_x$	0.512	0.918	0.682	0.342	0.176	0.160	0.158	0.703	0.296	0.969	0.754	0.715	0.951	0.516	0.456
	$İvme_y$	0.970	0.912	0.662	0.891	0.271	0.789	0.177	0.702	0.659	0.985	0.908	0.662	0.915	0.973	0.877
	$İvme_z$	**B**	0.116	**B**	**B**	0.196	0.201	0.801	0.838	**B**	**B**	**B**	0.016	**B**	**B**	0.230
	$Konum_x$	0.028	0.746	**B**	0.027	0.693	0.757	**B**	0.295	**B**	0.808	0.028	0.010	**B**	0.027	0.112
	$Konum_y$	0.331	0.647	0.019	0.180	0.693	0.024	0.817	0.488	**B**	0.985	0.039	0.018	**B**	0.631	0.465

B:Başarısız

7. SONUÇ VE ÖNERİLER

Bilgisayar biliminde gerçek rasgele sayıların üretiminde farklı gürültü kaynakları kullanılmaktadır. Bu kaynakların kullanım alanına göre farklılık göstermektedir. Tez çalışmasında gerçek rastgele sayı üretimi için insan kaynaklı bir GRSÜ geliştirilmiştir. Akıllı cep telefonu kullanan kişilerin yürüme, koşma ve durma durumları ivme sensörü ve GPS uygulaması kullanılarak kişi hareketlerinin değerleri elde edilmiştir. Elde edilen değerlerin güvenirliğini, başarımını test etmek için çalışmada skala indeks, otokolerasyon ve NIST testleri kullanılmıştır. Skala indeks testinde koşma, yürüme ve durma esnasında en yüksek skala indeks değerleri 0.937, 0.955 ve 0.882 olarak tespit edilmiştir. Bu testin sonucunda koşma, yürüme ve durma durumlarında doğrusal olmayan yani nonlineer değerlerin elde edildiği tespit edilmiştir. Bu durum üretilen sayıların genel olarak periyodik bir şekilde üretildiğini göstermektedir. Otokorelasyon testinde üretilen sayıların bağımlılıklarını inceleyen bir testtir. Bu testte sayıların birbirleriyle bağlantılı olmaması için test aralığını sağlanması istenmiştir. Otokorelasyon testi sonucunda en yüksek test değerleri, verilen d tam sayı değerlerine göre 1.019, 1.185 ve 1.602 olarak tespit edilmiştir. Test sonucunda da değerlerin aralığa uygun olduğu yani başarılı olduğu belirlenmiştir. Çalışma için uygulanan son test NIST test suiti olmuştur. NIST testinde testin güvenirliği için gereken uygun önem seviyesi aralığı seçilip test işlemi yapılmıştır. NIST testi sonucunda en yüksek test değerleri yürüme, koşma ve durma için sırasıyla 0.965, 0.985 ve 0.993 şeklindedir. Test sonucunda genel olarak test güvenirliği başarılı olarak sağlanmıştır. Ancak kişinin durma anında çok küçük hareket (milimetre seviyelerindeki değişimler) etmesi GPS'den elde edilen konumunlar da herhangi bir değişiklik göstermediğinden testlerden başarısız olunmuştur. Bunun yanı sıra durma esnasındaki çok küçük konum değişikliği x ve y eksenlerinde ivme sensörü tarafından algılanmış ve üretilen sayılar bazı testlerden başarısız olmuştur.

Sonuç olarak uygulanan çalışmada GRSÜ mobil telefon platformu için potansiyel olarak uygun, evrensel ve düşük maliyetli olup kullanıcıya özgü rastgele sayı üretiminin mümkün olduğu gösterilmiştir.

Genel olarak yüksek maliyetli üreteçlerin yanında maliyeti düşük ve kullanımı kolay kaynakların tercih edilmesi gerekmektedir. Yapılan çalışmada kullanıcının düşük maliyetli olarak rastgele sayı üretmesinin uygun olduğunu göstermektedir. Çalışmadaki insan hareketleri gürültü kaynağının literatürdeki diğer kaynaklarla karşılaştırılabilir ve değerlendirilmeleri yapılabilir. Çalışmadaki kullanılan testler dışında farklı testlerle de sonuçlar belirlenebilir ve karşılaştırılabilir. Gelecekte insan kaynaklı rasgele sayı üretimi ile ilgili geniş alanda uygulanabilecek çalışmalar yapılmalıdır.

KAYNAKLAR

[1] **Tokunaga, C., Blaauw, T. and Mudge, T**., True random number generator with a metasibility based quality control, *IEEE Journal of Solid-state Circuits*,43(1).

[2] **Avaroğlu, E.,Tuncer, T., Türk, M. ve Özer, A.B.,** A new method for hybrid pseudo random number generator, *J.Microelectron*, Electron Compon. Mater,4(4),303-311,2014.

[3] **Tuncer. T., Avaroğlu, E., Türk, M. ve Özer, A.B.,** Implementation of non-periodic sampling true random number generator on FPGA, *J.Microelectron*, Electron Compon. Mater, 4(4), 296-302, 2014.

[4] **Koyuncu, I., Özcerit, A.T., Pehlivan, I. ve Avaroğlu, E.,** 2014.Design and Implementation of chaos based random number generator on FPGA, *In 22nd Signal Processing and Communications Applications Conference (SIU)*, 236-239.

[5] **Wei, Z., Katoh, Y., Ogasahara, S., Yoshimoto, Y., Kawai, K., Ikeda, Y., Eriguchi, K., Ohmori, S. and Yoneda, S.,** True random number generator using current difference based on a fractional stohastic model in 40-nm embedded ReRAM,*IEEE Electron. Dev. Meet. 4.8.1-4.8.4,* San Fransisco, CA, USA, 2014.

[6] Walker, J., HotBits: Genuine Random Numbers Generated by Radioactive Decay, http://www.fourmilab.ch/hotbits, May 1996.

[7] **Moosavi, S.R., Nigussie, E., Virtanen, S. and Isoaho, J.,** 2017.Cryptographic Key Generation Using ECG Signal, *14th IEEE Annual Consumer Communications & Networking Conference (CCNC),*Las Vegas, NV, USA, 8–11 January 2017, 1024–1031.

[8] **Chen, X., Zhang, Y., Zhang, G. and Zhang, Y.,** 2012. Evaluation of ECG Random Number Generator for Wireless Body Sensor Networks Security, *5th International Conference on BioMedical Engineering and Informatics (BMEI 2012),*Chongqing, China, 16-18 Oct. 2012, 1308–1311.

[9] **Nguyen, D., Wanli, M., Tran, D., and Nguyen, K.,** 2017. EEG-Based Random Number Generators,in *Network and System Security*, pp. 245-256, Helsinki, Finland.

[10] **Chen, G.,** 2014. Electroencephalogram (EEG) signals pseudo-random number generators?, *Journal of Computational and Applied Mathematics*, **268,** 296-302.

[11] **Chen, I.T.,** 2013. Random Numbers Generated from Audio and Video Sources, *Mathematical Problems in Engineering*, Vol.2013, ArticleID 285373, 7 pages, 2013.

[12] **Nikolic, S., and Veinovic, M.**, 2016. Advancement of True Random Number Generators Based on Sound Cards Through Utilization of a New Post-processing Method, *Wireless Personal Communications*, **2**, 603-622, Springer Nature Switzerland AG.

[13] **Zhou, Q., Liao, X., Wong, K., Hu, Y. and Xiao, D.**, 2009. True random number generator based on mouse movement and chaotic hash function,*Information Sciences.*,**179(19),** 3442-3450.

[14] **Xingyuan, W., Xue, Q.and Lin, T.,** 2012. A novel true random number generator based on mouse movement and a one-dimensional chaotic map, *Mathematical Problems in Engineering*, Vol.2012, Article ID 931802, 9 pages, 2012.

[15] **Hu, Y., Liao, X.F., Wong, K. and Zhou, Q.**, 2009. A true random number generator based on mouse movement and chaotic cryptography,*Chaos, Solitons & Fractals.*, **40(3),** 2286-2293.

[16] **Schulz, A.M., Schmalbach, B., Brugger, P. and Witt, K.,**Analysing Humanly Generated Random Number Sequences:A Pattern-Based,ONE 7(7):e41531. doi:10.1371/journal.pone.0041531.

[17] **Tuncer, A. S.ve Kaya, T**., 2018. True Random Number Generation from Bioelectrical and Physical Signals,*Computational and mathematical methods in medicine,* Vol. 2018, Article ID 3579275, 11 pages.

[18] **Çiçek, İ.,Pusane, E.A. ve Dündar, G.**, 2017. An Integrated Dual Entropy Core True Random Number Generator,*IEEE Transactions on Circuits and Systems II:Express Briefs,* 64**,** 329-333.

[19] Ma, X., Yuan, X, Cao, Z., Qi, B. and Zhang, Z.,Quantum Random Number Generation,https://www.nature.com/articles/npjqi201621, 28 June 2016.

[20] **Özkaynak, F.,** 2014. Cryptographically secure random number generator with chaotic additional input, in *Nonlinear Dynamics*, 2015-2020,Springer Netherlands.

[21] **Schryver, C., Scmidt, D., Wehn, N., Korn, E., Marxen, H., Kostiuk, A. and Korn, R.,** A HardwareEfficient Random Number Generator for Nonuniform Distribution with Arbitrary Precision, *International Journal of Reconfigurable Computing*, vol.2012, 11.

[22] **Avaroğlu, E.**, 2014. Donanım Tabanlı Rastgele Sayı Üretecinin Gerçekleştirilmesi, *Doktora Tezi*, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ.

[23] https://www.matematiksel.org/gecmisten-gunumuze-rastgele-sayi-uretme-aracimiz-zar/,Geçmişten Günümüze Rastgele Sayı Üretme Aracımız: Zar, 14 Temmuz 2017.

[24] **Özdemir, K.** 2008. Sürekli-Zamanlı Kaos İle Rastgele Sayı Üreteci Tasarımı, *Yüksek Lisans Tezi*, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

[25] **Ripley, B.** 1983. Computer Generation of Random Variables:, *Yüksek Lisans Tezi,* İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

[26] **Nabiyev, V. V.,** 2013,Teoriden Uygulamalara Algoritmalar, Seçkin Yayıncılık, Ankara.

[27] https://tr.unionpedia.org/i/Algoritma#Blum Blum Shub, Algoritma, Unionpedi Anlamsal Ağ.

[28] Röck, A., 2005. Pseurandom Number Generators for Cryptographic Applications, https://www.rocq.inria.fr/secret/Andrea.Roeck/pdfs/dipl.pdf .

[29] Olsson, M., Gullberg, N.,2012. Blum Blum Shub on the GPU (A performance comparison between aCPU bound and a GPU bound BlumBlumShub generator),https://www.divaportal.org/smash/get/diva2:831071/FULLTEXT01.pdf.

[30] Naseem, A. S., Random Number Generator, https://www.slideshare.net/SayedAtif/Naseem/random number-generator-60772311. 11 April 2016.

[31] Random, 2008. http://www.random.org/randomness/.

[32] Hardware Random Number Generator,2008.http://en.wikipedia.org/wiki/Hardware_random number_generator

[33] **Tuna, M. ve Fidan C. B.,**2018. A study on the importance chaotic oscillators based on FPGA for true random number generating TRNG and chaotic systems. *Journal of the Faculty of Engineering and Architecture of Gazi University*,33(2), 469-486.

[34] **Nejati, H., Beirami, A. and Ali H. W.,** 2012. Discrete-time chaotic map truly random number generators: design, implementation and variability analysis of the zigzag map, *Analog Integr.Circ. Sig.Process.*,**73**, 363-374.

[35] **Milette, G. and Stroud, A**., 2012. *Professional Android Sensor Programming,* Wroks/E-books, London.

[36] **Taşdelen, A.,** 2015,Android Programlama Eğitimi,Pusula Yayıncılık, İstanbul.

[37] Ghosh, A.,Android Sensor Programming. http://docplayer.net/29179040-Android-sensor-programming-arindam-ghosh.html, 2014.

[38] Tılva, Y., Accelerometer and Gyroscope, https://www.slideshare.net/YASHTILVA2/accelerometer-and-gyroscope-52129081.

[39] https://www.bilgiustam.com/GPS-nedir-ve-nasil-calisir/, 2018.GPS Nedir ve Nasıl Çalışır?

[40] https://maker.robotistan.com/GPS-nedir/.GPS Nedir? Konum tespiti Nasıl Yapılır?, 2016.

[41] https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-22r1a.pdf**_2010**. **NIST SP 800-22, A Statistical Test Suite for Random NIST Page.**

[42] **Chan, M. M. J., Thulasiraman, P., Thomas, G. and Thulasiraman, R.,** Ensuring Quality of Random Numbers from TRNG: Design and Evaluation of Post Processing Using Genetic Algorithm, *Journal of Computer and Communications*,**4**, 73-92, 2016.

[43] **Benitez, R., Bolos, J. V. and Ramirez, E. M., K.,** A wavelet based tool for studying non-periodicty, *Comput. Math, Appl.*, Vol. 60, 634, 2010.

[44] **Yang, Y. G. and Zhao, Q. Q.,** Novel pseudo random number generator based on quantum random walks, *Scientific Reports*, Vol. 6, 20362, 2016.

[45] **Karakaya, B., Çelik, V. ve Gülten, A.,** Chaotic cellular neural network based true random number generator, *Int. J. Circ. Theor. Appl.,* Vol. 45,1885-1897, 2017.

[46] http://csrc.nist.gov/rng/rng2.html NIST Special Publication 800-22, 2001.

ÖZGEÇMİŞ

YELİZ GENÇ	
KİŞİSEL BİLGİLER	
DOĞUM YERİ/TARİHİ :	TEKİRDAĞ/ 17.10/1993
EPOSTA:	ylz.gnc.46@gmail.com
MEDENİ HALİ :	BEKAR
EĞİTİM BİLGİLERİ	
LİSANSÜSTÜ	**2017-2019,** FIRAT ÜNİVERSİTESİ, FEN BİLİMLERİ ENSTİTÜSÜ, YAZILIM MÜHENDİSLİĞİ ABD, ELAZIĞ
LİSANS :	**2011-2016,** FIRAT ÜNİVERSİTESİ,TEKNOLOJİ FAKÜLTESİ, YAZILIM MÜHENDİSLİĞİ, ELAZIĞ
LİSE :	**2007-2011**, HAFİZE ÖZAL MESLEKİ VE TEKNİK ANADOLU LİSESİ, MALATYA
BİLDİĞİ YABANCI DİLLER	
	İngilizce (B1)
ARAŞTIRMA DENEYİMİ	
✓ Kullanabildiğiniz Bilgisayar Programlama dilleri : (JAVA, C, C#, MATLAB, PYHTON) ✓ Kullanabildiğiniz bilgisayar Programları: (ANDROID STUDIO, ECLIPSE, NETBEANS, MATLAB, MS VISUAL STUDIO)	
İLGİLENDİĞİ DİĞER BİLİMLER	
✓ TİYATRO, SİNEMA, EDEBİYAT, GRAFİK TASARIM	