GENETİK ALGORİTMA İLE GRUP ASANSÖR SİSTEMLERİ İÇİN KABİN YÖNLENDİRME SİSTEMİNİN TASARIMI VE GERÇEKLEŞTİRİLMESİ

2019
YÜKSEK LİSANS TEZİ
MEKATRONİK MÜHENDİSLİĞİ

SEMİH PAK

GENETİK ALGORİTMA İLE GRUP ASANSÖR SİSTEMLERİ İÇİN KABİN YÖNLENDİRME SİSTEMİNİN TASARIMI VE GERÇEKLEŞTİRİLMESİ

SEMİH PAK

Karabük Üniversitesi
Fen Bilimleri Enstitüsü
Mekatronik Mühendisliği Anabilim Dalında
Yüksek Lisans Tezi
Olarak Hazırlanmıştır.

KARABÜK
Mart 2019

Semih PAK tarafından hazırlanan "GENETİK ALGORİTMA İLE GRUP ASANSÖR SİSTEMLERİ İÇİN KABİN YÖNLENDİRME SİSTEMİNİN TASARIMI VE GERÇEKLEŞTİRİLMESİ" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Dr. Öğr. Üyesi Ali UYSAL .....
Tez Danışmanı, Elektronik Sistemler Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği/çoğunluğu ile Mekatronik Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 15/03/2019

Ünvanı, Adı SOYADI (Kurumu)	İmzası
Başkan : Doç. Dr. Rıfat HACIOĞLU (BEÜ)	.....
Üye : Prof. Dr. Raif BAYIR (KBÜ)	.....
Üye : Dr. Öğr. Üyesi Ali UYSAL (CBÜ)	.....

...../...../2019

KBÜ Fen Bilimleri Enstitüsü Yönetim Kurulu, bu tez ile, Yüksek Lisans derecesini onamıştır.

Prof. Dr. Filiz ERSÖZ .....
Fen Bilimleri Enstitüsü Müdürü V.

"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."

Semih PAK

ÖZET

Yüksek Lisans Tezi

GENETİK ALGORİTMA İLE GRUP ASANSÖR SİSTEMLERİ İÇİN KABİN YÖNLENDİRME SİSTEMİNİN TASARIMI VE GERÇEKLEŞTİRİLMESİ

Semih PAK

Karabük Üniversitesi
Fen Bilimleri Enstitüsü
Mekatronik Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Ali UYSAL
Mart 2019, 74 sayfa

Bu çalışmada, grup asansör sistemlerinde kabin yönlendirme problemi genetik algoritma ile ele alınarak katlardaki yolcuların kabin bekleme süreleri minimize edilmektedir. Çalışma kapsamında 5 kabinli ve 16 katlı bir asansör sisteminin modellemesi yapılarak bir simülatör programı geliştirilmiş ve çağrılarının dışarıdan alınabildiği gömülü donanım tabanlı bir deney düzeneği hazırlanmıştır. Deney düzeneği sayesinde geliştirilen algoritmalar, gereken donanımlarında eklenmesi ile mevcut asansör kontrol kartlarına uygulanabilir. Hazırlanan simülatör programında geleneksel kontrol algoritması ve genetik algoritma tabanlı kontrol algoritması aynı trafik senaryoları için test edilerek, enerji tüketim değerleri ve ortalama cevaplama süreleri hesaplanmıştır. Kontrol algoritmaları simülasyon sonuçlarına göre karşılaştırıldığında, genetik algoritma tabanlı kontrol algoritmasının enerji tasarrufu sağladığı ve ortalama bekleme süresini azalttığı görülmektedir.

Anahtar Sözcükler : Asansör, genetik algoritma ve gömülü sistem.
Bilim Kodu : 929.1.182

ABSTRACT

M. Sc. Thesis

DESIGN AND IMPLEMENTATION OF THE CABIN ROUTING SYSTEM FOR GENERIC ALGORITHM AND GROUP ELEVATOR SYSTEMS

Semih PAK

Karabük University
Graduate School of Natural and Applied Sciences
Department of Mechatronics Engineering

Thesis Advisor:
Asst. Prof. Ali UYSAL
March 2019, 74 pages

In this study, cabin waiting time of the passengers on the floors is minimized by using the genetic algorithm in the cabin routing problem in group elevator systems. For GAS, a simulator program was developed by modeling 5 cabins and 16-storey elevator system and an embedded hardware-based test setup was prepared in which calls can be taken from outside. The algorithm developed by the experimental setup can be applied to the existing elevator control cards by the addition of necessary equipment. In the simulator program, traditional control algorithm and genetic algorithm based control algorithm were tested for the same traffic scenarios and energy consumption values and average response times were calculated. When the control algorithms are compared with the simulation results, it is observed that the genetic algorithm based control algorithm saves energy and reduces the average cabin waiting time of the passengers.

Key Word **:** Elevator, genetic algorithm and embedded system.

Science Code **:** 929.1.182

TEŞEKKÜR

Bu tez çalışması boyunca bilimsel temeller kapsamında yaptığı bütün yardım ve yönlendirmelerden ötürü sayın hocam Dr. Öğr. Üyesi Ali UYSAL'a ve genetik algoritmanın kodlanması hususunda anlatımları ve öğretimleri için sayın Dr. Öğr. Üyesi İlhan İLHAN'a teşekkür ederim.

Maddi ve manevi hiçbir desteğini esirgemeden arkamda duran aileme tüm kalbimle teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

SİMGELER

C_i : i indisli çağrının cevaplanma

Δ_i : i indisli kabinin konumu ile hedef kat arasındaki kat farkı

n : çağrı matrisindeki çağrı sayısı

t_i : i indisli çağrı cevaplama süresi

t_{ka} : kabin kapı açılma süresi

t_{kd} : kabin kalkma-durma süresi

t_{ka} : kabin kapı kapanma süresi

U : uygunluk değeri

V_{sh} : kabin seyir hızı

W_b : bireyin (yönlendirme tahmininin) enerji tüketim değeri

KISALTMALAR

AAA : Alan Ağırlık Algoritması
ABC : Artificial Bee Colony
AI : Artificial Intelligence
BM : Bulanık Mantık
BSA : Bulanık Sinir Ağı
ÇN : Çaprazlama Noktası
DA : Dinamik Algoritma
DE : Differential Evolution
GA : Genetik Algoritma
GAS : Grup Asansör Sistemleri
PSO : Parçacık Sürü Optimizasyonu
SAR : Simulated Annealing with Random Starts
TAA : Tabu Arama Algoritması
YSA : Yapay Sinir Ağları

BÖLÜM 1

GİRİŞ

Gelişen teknoloji sayesinde yeni yapıların kat ve içerisindeki insan sayısı da katbekat artmıştır. Bu gelişme ile bina içeresindeki katlar arası hareketliliği kolaylaştıran asansör sistemlerine olan ihtiyaçta giderek artmıştır. Günümüzde yapılan iş merkezleri ve konutlar için çıkarılan yeni imar yönetmelikleriyle birlikte belirli katlardan sonra insanlar için vazgeçilmez olan asansör sistemleri zorunlu hale getirilmektedir.

Başlarda tek kabinli olarak kullanılmaya başlanan asansörler zamanla artan insan trafiği nedeniyle yerini çok kabinli olan grup asansör sistemlerine (GAS) bırakmaktadır. Kabin sayısındaki bu artış ile kontrol algoritması değişerek daha kapsamlı bir hale gelmektedir. Ancak kabin sayısının artırılması trafiğin çözülmesi için tek başına yeterli olmayışından, enerji verimliliği ve bekleme süresinin optimizasyonu için çeşitli çalışmalar yapılmaya başlanmıştır.

Mikroişlemci teknolojisindeki gelişmeler sayesinde asansör kontrol sistemlerinde daha kapsamlı algoritmaları çalıştırabilecek donanımlar kullanılmaya başlanmıştır. İşlem kapasitesi yüksek bu işlemcilere çeşitli kontrol algoritmaları gömülerek asansör sistemlerinin performansları da geliştirilmektedir. Günümüzde özellikle gökdelenlerde ve yüksek katlı binalardaki grup asansör trafiğinin kontrol edilmesinde bilgisayar tabanlı kontrol sistemleri kullanılmaktadır.

Günümüzde gelişmiş asansör kontrol sistemlerinde geleneksel kontrol yöntemlerinin yerini yapay zekâ tabanlı kontrol algoritmaları almaktadır. GAS'ın yapay zekâ ile kontrol edilmesiyle birlikte yolcuların talepleri hızlı bir şekilde karşılanarak katlardaki bekleme süreleri azaltılmış ve bu sistemlerin daha verimli hale dönüşmesi sağlanmıştır [1].

Alander vd. (1995) çalışmalarında, GAS'ta kabin yönlendirme optimizasyonu için GA'nın uygulanabilirliğini konu almaktadır. Önerdikleri kontrol algoritması, gerçek bir asansör sisteminden ziyade bilgisayar ortamında hazırlamış oldukları 16 katlı 3 asansöre sahip iş merkezi için simüle edilmiştir. Simülasyon sonuçlarına göre önerdikleri algoritma, normal kontrol algoritmalarından bekleme süresi için en az 2 kat daha iyi sonuç vermektedir. Ayrıca çalışmalarında GA'nın GAS için avantajlarından detaylı bir şekilde bahsederek gelecek yıllarda GA'nın kullanılabilirliği noktasında çok doğru tahminlerde bulunmuşlardır [2].

İmrak (1996) çalışmasında, asansör trafiğinin optimizasyonunu bulanık mantık (BM) ve yapay sinir ağları (YSA) ile ele almaktadır. Ayrıca faklı bina tipleri için asansör trafik analizi yapabilen ve önerdiği algoritmaların test edilebildiği bir simülasyon programı hazırlamıştır. Geliştirdiği simülasyon sayesinde değişik bina tipleri için trafik analizleri yaparak GAS için algoritmalar tasarlamıştır. Simülasyon sonuçlarından aldığı grafikler yardımıyla uygun algoritmayı seçerek, asansör trafiğini düzenleyip bekleme zamanını minimize etmiştir [3].

Bolat vd. (2004) çalışmalarında, GA'nın tanımını ve çalışma prensibini ele alarak ve algoritma içiresindeki parametreleri tek tek incelemektedir. Ayrıca GA'nın mühendislik uygulamalarında nasıl kullanılabileceğini detaylı bir şekilde anlatarak tek değişkenli bir denklemin optimizasyonu ile konunun daha iyi anlaşılmasını sağlamaktadırlar [4].

Cortés vd. (2004) çalışmalarında, GAS'ta çağrılar için kabin yönlendirme optimizasyonunu GA ile ele almaktadırlar. Çalışmaları için ARENA v5.0 programı kullanarak GAS sistemini kapsamlı bir şekilde simüle etmişlerdir. Önerdikleri algoritmanın ARENA simülasyon sonuçlarını, endüstride kullanılan genel kontrol algoritmaları ile karşılaştırmışlardır. Algoritmaları, geleneksel grup kontrol sistemlerinden yaklaşık %25 oranında daha iyi performans vermektedir [5].

Bolat (2006) çalışmasında, GAS'ta kabin yönlendirme optimizasyonunu GA ile ele almaktadır. Önerdiği algoritmada gerçekleştirdiği optimizasyonun farklı bina karakteristiklerine de uyarlayabilmesi için birde simülasyon programı geliştirmiştir.

Bu simülasyon programı farklı bina tipleri ve karakterlerine göre trafik analizleri gerçekleştirebilmektedir. Analiz sonuçlarına göre çağrılara en uygun kabinleri yönlendirecek GA tabanlı optimizasyon algoritması hazırlayarak asansör trafiğini düzenlemiş ve insanların seyahat ve bekleme sürelerini %23 oranına kadar azaltmıştır [6].

Dağdelen vd. (2005) çalışmalarında, GAS için kabin yönlendirme problemini alan ağırlık algoritması (AAA) ile ele almaktadırlar. Çalışmaları kapsamında AAA'ya ekledikleri kabinlerin görev listesi kontrolü sistemdeki yolcuların bekleme sürelerini %31 'e kadar düşürmektedir [7].

Hiller ve Tuchscherer (2007) çalışmalarında, gömülü sistem üzerinde gerçek zamanlı uygulanabilen bir GAS kontrol algoritması önermektedir. Algoritmalarında bekleme ve seyahat süresini en aza indirmeyi hedeflemektedirler. Ancak kısıtlı işlemci kapasitesinden dolayı GA, parçacık sürü optimizasyonu (PSO) ve YSA gibi yüksek donanım isteyen algoritmaların yerine sadece gezgin satıcı probleminden çok iyi bilinen sezgisel eklemeler metodunu ile kabin yönlendirme işlemini başarılı bir şekilde gerçekleştirmektedirler. Önerdikleri algoritmanın etkileyici simülasyon çıktıları sonucunda algoritmaları, endüstri partnerleri olan Koll-Morgen Steuerungstechnik ile gerçek bir asansör sistemine aktarmıştır [8].

Dursun ve Sarıbaş (2008) çalışmalarında, 5 katlı bir binada tek kabinli bir asansör sisteminin YSA ile denetimini ele almaktadır. Binanın trafik analizini ve çalışmanın simülasyonu için C++ ortamında bir simülatör programı hazırlamışlardır. Simülatörden aldıkları trafik analizi doğrultusunda YSA üzerinde gelişmelere giderek algoritmanın bina için uygunluğunu artırmışlar ve önerdikleri YSA'yı aynı şartlar altında klasik kontrol yöntemleri ile karşılaştırdıklarında YSA'nın %17.5 oranında daha iyi sonuç verdiğini görülmektedir [9].

Yang vd. (2009) çalışmalarında, GAS'ta kabin yönlendirme probleminin optimizasyonunu ele almaktadır. Çalışmalarında trafik oranı ve yolcuların dağıtımına bağlı hızlı bir şekilde kat bölümlemesini ayarlayabilen dinamik bölümleme metodunu kullanmaktadır. Bu bağlamda optimizasyon için bulanık sinir ağlarını (BSA)

önerilmektedir. Sundukları asansör modelinin simülasyon sonuçları, algoritmalarının asansör sisteminin tüm performans değerlerini geliştirdiği için ektili bir yönlendirme algoritması olduğunu göstermektedir [10].

Jamaludin vd. (2009) çalışmalarında, GAS'ta kabin yönlendirme optimizasyonunu BM tabanlı olarak gerçekleştirmektedir. Çalışmalarında uygun bulanık kuralların ve üyelik fonksiyonlarının seçimi için tahmini trafik senaryoları yerine ortalama bekleme süresi ele alınmaktadır. Simülasyon sonuçlarına bakıldığında önerdikleri algoritmanın tüm kriterlerde, konvansiyonel grup kontrol performansından %66.3 oranında daha iyi sonuçlar verdiğini görmektedir [11].

Bolat vd. (2010) çalışmalarında, GAS'ta kabin yönlendirme optimizasyonunu GA tabanlı olarak ele almaktadır. Çalışmalarını gerçekleştirirken aynı algoritmayı kullananlardan farklı olarak yeni uygunluk fonksiyonu denemişlerdir. Algoritmalarını yüksek trafik yoğunluğundaki binalar için test ettiklerinde algoritmaları, geleneksel algoritmalardan yaklaşık %25 oranında daha iyi sonuç vermektedir [12].

Kocamaz ve Çiçekli (2010) çalışmalarında, GAS'ı da kapsayıcı nitelikte paralel makinelerin çizelgelenmesi problemini GA ile ele alarak mutasyon oranının GA üzerindeki etkisini incelemektedir. Gerçekleştirdikleri çalışma kapsamında 20 işi 3 paralel makineye optimum şekilde tevzi ederken en uygun mutasyon oranına ulaşmaya çalışmaktadırlar. Yaptıkları testler neticesinde paralel makineler için en uygun mutasyon oranının %5 ile %9 arasında olduğu görülmektedir [13].

Bayğın ve Karaköse (2011) çalışmalarında, GAS için kabin yönlendirme optimizasyonunu klonal seçim algoritması ile ele almaktadır. GAS için önerdikleri algoritmanın testi için 16 kata ve 7 asansöre sahip bir asansör sistemini simüle etmişlerdir. Önerdikleri optimizasyon algoritmasının simülasyon ortamındaki test ve analizleri sonucunda algoritmaları çağrı cevaplama süresi %20 oranında azaltmaktadır [14].

Cortés vd. (2013) çalışmalarında, yüksek katlı binalardaki GAS'ın çağrı optimizasyonunu viral sistem algoritması ile ele almaktadır. Algoritmalarını

MATLAB ortamında, literatürde en iyi sonuçları veren GA ve tabu arama algoritmaları (TAA) ile karşılaştırdıklarında, önerdikleri algoritmanın yüksek katlı ve çok asansörlü binalarda, GA ve TAA'dan daha iyi sonuç verdiği görülmektedir [15].

Bolat vd. (2013) çalışmalarında, GAS'ta kabin yönlendirme optimizasyon problemini PSO ile al ele almaktadır. Önerdikleri algoritmayı 2 - 6 kabinli ve 10 - 24 duraklı bina için test etmişlerdir. Test sonuçlarını GA ve TAA algoritmaları ile karşılaştırdıklarında önerdikleri algoritma GA'dan yaklaşık iki kat ve TAA'dan yaklaşık üç kat daha iyi sonuç vermektedir [16].

Beamurgia vd. (2015) çalışmalarında, GAS'ta kabin yönlendirme optimizasyon problemini GA ile ele almaktadır. Çalışmalarında GA'ya ait uygunluk fonksiyonuna en uzun bekleme süresi, en uzun seyahat süresi ve sonraki durak şeklinde birçok detaylandırılmış parametreler ekleyerek GA'nın en iyi sonuç verecek şekilde tasarlanmasını sağlamışlardır. Yapmış oldukları eklentiler ile GA'nın performansını önemli ölçüde artırmışlardır [17].

Debnath ve Serpen (2015) çalışmalarında, araç park sistemleri için grup asansör sisteminin optimizasyonunu GA ile ele almaktadır. Çalışmaları kapsamında 4-20 kat ve 400 park alanına kadar esnetilebilen ve gerçek zamanlı çalışan algoritma geliştirmişlerdir. Geliştirmiş oldukları algoritmayı MATLAB ortamında simüle ettiklerinde almış oldukları sonuçlar, algoritmanın optimum bekleme süresi ve servis sayısının optimize edildiğini göstermektedir [18].

Tartan ve Çiftlikli (2016) çalışmalarında, GAS'ta kabin yönlendirme problemini ele alarak bekleme zamanını optimize etmeyi hedeflemektedir. Optimizasyon işlemini geliştirilmiş GA kullanarak gerçekleştirmişlerdir. Önerdikleri kodlama sistematiği kodlanma işlemini çok daha basit hale indirgeyerek işlemcinin yükünü azaltmıştır. Önerdikleri algoritma geleneksel kontrol algoritmasından yaklaşık %47 oranında daha iyi performans göstermektedir [19].

Bolat vd. (2017) çalışmaları kapsamında GAS'da kabin yönlendirme optimizasyonu için, "çıkarımsal evrim", "rastgele yeniden başlatmalı benzetimli tavlama" (Simulated

Annealing with Random Starts, SAR), "yapay arı kolonisi" (Artificial Bee Colony, ABC), "yarasa algoritması bakteri otlama optimizasyon algoritması", PSO, GA ve TAA algoritmalarını karşılaştırmaktadır. Test sonuçları ABC ve TAA algoritmalarının daha iyi ortalama yolculuk zamanı verdiğini göstermektedir. Ayrıca çalışmalarında benzetimli tavlama algoritmasının yeni bir versiyonu olarak SAR'ı geliştirmişlerdir. Geliştirdikleri algoritma karşılaştırılan sekiz algoritma arasında en iyi sonucu veren üçüncü algoritmadır [20].

Sorsa vd. (2017) çalışmalarında, kabin yönlendirme probleminin gürbüz bir algoritma ile optimize edilmesini ele almaktadır. Önerdikleri algoritmada kabin yönlendirme probleminin modellenmesi "Geometrik Poisson Süreci" (geometric Poisson process) ile gerçekleştirmektedir. Deneyler sonucunda önerdikleri gürbüz kabin yönlendirme optimizasyonu belirsiz durumlar altında iyi yönlendirme tahminleri vermektedir [21].

Çiflikli ve Tartan (2018) çalışmalarında, asansör sistemlerinde kabinlerin kullanılmadıkları zamanlarda çağrıların optimum cevaplanması adına bina trafiğine uygun katlarda park edilmesi üzerine çalışmaktadır. Çalışmalarında kabin yönlendirme için hazırlanan optimizasyon algoritmalarının planlı kabin parkı ile desteklenmesi durumunda çağrıların cevaplanması için gereken sürede iyileştirme sağladığı görülmektedir [22].

Çetin ve Yurdusev (2018) çalışmalarında, atık su sistemleri için optimum boru çapını ve akış eğimi düşük maliyet ile gerçekleştirmek için GA kullanmaktadır. Çalışmalarında GA için en uygun kontrol parametrelerini belirlemek için 5 farklı popülasyon büyüklüğü için 20 farklı çaprazlama ve mutasyon oranı kombinasyonu kullanmışlardır. Sundukları çalışma, bir atık su projesinde optimum çözümü %5 daha düşük maliyet ile gerçekleştirmektedir [23].

Chou vd. (2018) çalışmalarında, yolcu taleplerine ve enerji tüketimine bağlı asansör kontrol yöntemlerinin geliştirilmesi üzerine bir çalışma gerçekleştirmektedir. Çalışmalarında asansör kontrol algoritmasına bekleyen yolcu sayılarının da eklenerek daha etkili bir kontrol yapılabileceğini ileri sürmüş ve bekleyen yolcu sayılarını kamera görüntüleri üzerine derin öğrenme uygulayarak belirlemişlerdir. Önerdikleri

asansör kontrol yöntemi geleneksel kontrol yöntemi ile karşılaştırıldığında yaklaşık aynı cevaplama süresi için yaklaşık %18 daha az enerji harcamaktadır [24].

Bu tez çalışmasında, GAS içi bir simülatör programı hazırlanarak gömülü donanım üzerinde GAS'ın simülasyonu ve kontrolü yapılmaktadır. Hazırlanan simülatör programı içerisine geleneksel ve GA tabanlı yönlendirme algoritması eklenerek kabinlerin kontrolü sağlanmaktadır. Ayrıca gömülü sistemin gerçek asansör kontrol kartlarına uyarlanabileceğini göstermek adına çağrıların dışardan alınabileceği bir deney düzeneği hazırlanmıştır.

Yapılan bu çalışma literatürde GAS için yapılan akıllı kabin yönlendirme optimizasyonlarından farklı olarak gerçek zamanlı çalışabilen bir gömülü donanım üzerinde gerçekleştirilmiştir. Ayrıca tercih edilen donanımın algoritma açısından uygunluğunun testide farklı bir çalışma olarak gerçekleştirilmiştir.

Bu çalışmanın birinci bölümünde asansör sistemlerindeki kabin yönlendirme problemi içi yapılan akademik çalışmalara yer verilmektedir. İkinci bölümünde asansör sistemlerinden ve gelişiminden bahsedilerek GAS'a ve kontrol yöntemlerine değinilmektedir. Üçüncü bölümünde kabin yönlendirme için kullanılan GA anlatılmaktadır. Dördüncü bölümde deney düzeneği ve hazırlanan simülatör programı detaylı bir şekilde anlatılmaktadır. Beşinci bölümde GA'nın asansör sistemine nasıl uyarlandığı adım adım anlatılmaktadır. Altıncı bölümde önerilen algoritma ile geleneksel kontrol algoritmasının karşılaştırılması yapılarak sonuçlar grafiklerle gösterilmektedir. Son olarak yedinci bölümde ise önerilen algoritma ile geleneksel kontrol algoritmasının sonuçları karşılaştırılarak sonuçlar ve değerlendirmeler verilmektedir.

BÖLÜM 2

ASANSÖR SİSTEMLERİ VE KONTROLÜ

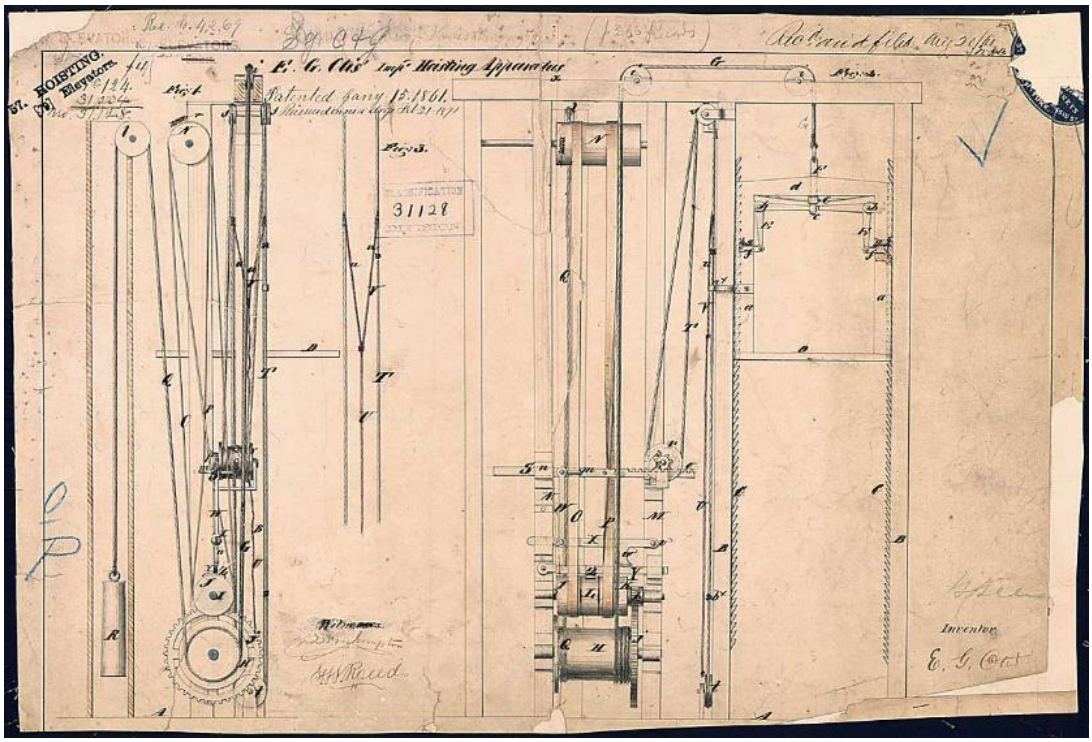
2.1. ASANSÖR SİSTEMLERİNİN TARİHSEL GELİŞİMİ

Asansör sistemlerinin gelişiminde büyük bir öneme sahip olan ve asansörün temelini teşkil eden vinç sisteminin ilk örneğini M.Ö. 236 yılında Arshimed gerçekleştirmiştir. İnsan taşıyan ilk asansör ise Fransız kralı XV. Louis için 1743 yılında Versailles sarayına kurulmuştur [25,26].

Asansör sistemleri özellikle sanayi devriminin ardından hızla gelişmeye başlamıştır. 1830 yıllarında İngiltere'de, doğrudan hidrolik tahrikli yük asansörleri ve 1835 yılında da buhar makinası ile çalışan asansörler yapılmıştır. Elisha Graves Otis'in 1853 yılında Crystal Palace New York'ta düşmeye karşı ilk emniyet düzeni olan ilk asansörü kurması, asansör sektörü için bir dönüm noktası olmuştur (Şekil 2.1) [27].

Şekil 2.1. Elisha Graves Otis'in 1853 yılında yaptığı güvenlik testi.

Elisha Graves Otis'in 1853 yılında Crystal Palace New York'ta düşmeye karşı ilk emniyet düzeni olan ilk asansörü kurması, asansör sektörü için bir dönüm noktası olmuştur (Şekil 2.1). Otis 1855'te, dönen bir milden aktarma organları ile hareket almak yerine kendi buhar makinesi ile çalışan asansör yapmıştır. Modern anlamda ilk asansör örneğini ise New York'ta bir iş merkezine 1857 yılında Otis firması tarafından kurulmuştur. Otis, yaptığı asansör için Amerikan Patent Ofisi'nden 15 Ocak 1861'de 31.128 numaralı patenti almıştır (Şekil 2.2). Aynı yıl Otis'in ölümüyle ailesi bu alandaki çalışmalarına devam ederek asansör konusunda birçok ilke imza atmış ve 1878 yılında hız regülatörü ve paraşüt düzenini geliştirmiştir. İlk grup asansör örneği ise 1879 yılında yine Otis firması tarafından dört kabin kullanılarak New York'da Boreel binasına kurulmuştur [25,27].



Şekil 2.2. Elisha Graves Otis'in asansör patent çizimi.

19. yüzyılın sonlarına doğru asansör sistemlerinde hareketi sağlamak için kullanılan hidrolik pistonların ve buhar makinelerinin kurulum ve kullanımdaki zorlukları nedeniyle asansör sistemleri için yeni güç sistemlerinin arayışına başlanmıştır. Alman bilim insanı Werner von Siemens 1880 yılında asansör kabininin altına bir elektrik motoru yerleştirerek elektrik ile hareket ettirilen ilk asansör sistemini

gerçekleştirmiştir. Bu gelişmenin ardından aynı yıl Siemens ve Halske firması 22 metre yüksekliğinde bir binaya ilk elektrikli asansörü yerleştirmişlerdir. Otis, 1889 yılında elektrik motoru ile çalışan asansör sistemine sonsuz vida mekanizmasını eklemiş ve kabini tutan halatların üzerime sarılabileceği bir tambur geliştirmiştir [27]. 2016 yılında Mitsubishi Electric firmasının 632 metre yüksekliğine sahip Shanghai Tower gökdelenine kurduğu asansör kabini saniyede 20.5 metre hızla dünya rekoru kırmıştır [28].

Mikroelektronik ve gömülü donanım alanlarındaki gelişmeler ile birlikte asansör sistemlerinde grup kontrollerine başlanmış ve 20. yüzyılın sonlarına doğru asansör kontrol sistemlerinde yapay zekâ uygulamalarına yer verilmeye başlamaktadır.

2.2. ASANSÖR KONTROL YÖNTEMLERİ

2.2.1. Tarihsel Gelişimi

Asansör sistemlerinin kontrolü temelde iki mühendislik problemini doğurur. Bunlardan ilki asansör kabinlerinin yukarı ve aşağı yönde hareketini sağlayacak komutların verilmesi ve hedef katlarda kabinlerin durdurulmasıdır. Diğer problem ise birden fazla kabinin bir arada bulunduğu GAS'ta kabinlerin çağrılara en uyun şekilde yönlendirilerek sistemin en verimli halde işletilme gereksinimidir.

Asansör sistemlerinin ilki olan buhar ve hidrolik tahrikli asansörler "el-kablosu" kontrolüyle kumanda edilirlerdi. Asansör hizmeti isteyen bir yolcu kabloya ulaşabilir ve asansörü çağırmak için kumanda edebilirdi. Operatör bulundurmayan bu sistemlerin kullanımını kolay olsa da çalışma tarzı nedeniyle emniyetsizdi. Elektrik motoru ile çalışan asansörlerin gelişmesiyle birlikte asansörün kontrolü kabin içerisindeki görevli operatör tarafından anahtar devresine eklenen bir kol yardımı ile yapılıyordu. Operatörlü kabin kontrolü ilk basit kontrol yönteminde, kabinin kata uygun seviyede olup olmadığını kontrol etmek gözleme dayanmaktaydı [6].

İlerleyen yıllarda yetersiz olan bu kontrolün yerine otomatik tahrik kontrolü ve tek kabin için sinyal sistemi geliştirilmiştir. Sinyal sistemi ile birlikte, kabinin hızlanma-

yavaşlama kontrolü ve kat ile kabin arası seviye kontrolü sağlanarak işletme hızları ve taşıma kapasitesi artmıştır.

Günümüz asansör sistemlerinin kontrolünde yüksek işlem kapasitesine sahip mikroişlemciye sahip mikrodenetleyiciler kullanılmaktadır. Özelikle bilgisayar alanındaki hızlı gelişmelerin yapay zekâ uygulamalarının asansöre uyarlanmasında önemli bir yeri vardır. Çizelge 2.1'de asansör sistemlerinin kontrolünde kaydedilen gelişmeler görülmektedir [25,27].

Çizelge 2.1. Asansör kontrol yöntemlerinin tarihsel gelişimi.

Zaman	**Asansör Kontrol Türü**
1850 – 1890	Basit mekanik kontrol
1890 – 1920	Operatör ve elektrikli kabin anahtar devreli kontrol
1920 – 1950	Operatör ve düğmeli kontrol
1950 – 1975	Grup kontrolü
1975 – Günümüz	Bilgisayar esaslı grup asansör kontrolü
1980 – Günümüz	Yapay zekâ tabanlı kontrol

2.2.2. Tek kabinli Asansör Kontrol Sistemleri

2.2.2.1. Otomatik Tip Kontrol

Düşük yolcu kapasitesine sahip asansör sistemleri için kullanılan kontrol yöntemidir. Bu sistemlerde kabinler genellikle tek yolcu taşırlar. Bekleme süresinin uzun olduğu bu sistemlerde yolcular gidecekleri katın düğmesine basarak kabini hareket ettirirler. Kabin hedef kata doğru hareket ederken diğer katlarda durmaz [25,27].

2.2.2.2. Toplamalı Tip Kontrol

Toplamalı tip kontrol yöntemi şehir içi otobüs taşımacılığına benzemektedir. Kabin aynı yöndeki çağrılara cevap verme mantığına dayalı kontrol edilir. Kabin bulunduğu kattan itibaren aldığı ilk yön dahilinde hareketine başlar ve aynı yöndeki son kata kadar hareketine devam eder. Bu kontrol yönteminde aynı yöndeki son çağrı cevaplanmadan kabinin hareket yönü kesinlikle değişmez. Bir çevrim içinde kabin bu şekilde hareket ederek her kata hizmet vermektedir. Bu yöntem kendi içinde üçe ayrılır [25,27];

a) Yönsüz toplama: Bu yöntemde katlarda asansörü çağırmak için sadece tek bir buton vardır. Yolcu kabini çağırmak için bu düğmeye basar ve gelen ilk çağrı ile kabinin yönü belli olur. Kabin hareket yönündeki her kata hizmet sunar ve son kata kadar yön değişmeden hareket devam eder.

b) Aşağı Toplama (Yukarı Dağıtma-Aşağı Toplama): Bu yöntemde de katlarda sadece tek bir çağrı butonu vardır. Burada hedef yolcuları hızlı bir şekilde yukarı taşınmasıdır. Kabin ara katlara sadece kabin içinden çağrı varsa uğrar aksi halde kabin son kata varmadan önce asla ara katlarda durdurulmaz.

c) Tam Toplama: Bu yöntemde ise her katta aşağı ve yukarı olmak üzere iki adet çağrı butonu vardır. Yolcu gitmek istediği katın yönüne bağlı olarak kabine bu düğmelerden biri ile çağrı gönderir. Eğer kabinin hareketi ile çağrının yönü aynı ve kabin yeni gelen çağrının bulunduğu katı geçmemiş ise çağrı cevaplanır. Aksi halde asansör aynı yöndeki son çağrıyı cevaplayıp bir sonraki çevrimde o çağrıyı cevaplar.

2.2.3. Grup Asansör Kontrol Sistemleri

Grup asansör sistemleri, yan yana konumlandırılmış iki ya da daha fazla kabinin bir araya gelerek oluşturduğu sistemlerdir [6,25]. Bina içerisindeki yolcu trafiğini karşılama noktasında tek kabinin yetersiz olduğu durumlarda birden fazla kabin yan yana konumlandırılır. Ancak çağrı trafiğin çözülmesi için kabinlerin yan yana konumlandırılmasının yanı sıra koordineli bir şekilde çalışmaları da gerekmektedir.

Bunun için bu sistemlerde tüm kabinlerin çağrı kayıt birimleri birbirine paralel bağlanır ve grup kontrolü aktif edilir. Sistem gelen çağrılar için hangi kabin uygunsa onu yönlendirir [3,25].

2.2.4. Bilgisayar Tabanlı Kontrol Sistemi

Bu yöntem asansör kontrol sistemleri arasındaki en kapsamlı ve verimli olan kontrol türüdür. Bu yöntemde birçok parametre bilgisayarın yüksek işlem gücü sayesinde bir arada değerlendirilerek daha esnek çözümler sunulmaktadır. Bilgisayarın hesaplama yeteneği sayesinde farklı senaryolar çalıştırılarak en uygun kabin yönlendirmesi gerçekleştirilmektedir. Bu sayede yolcuların bekleme süresi en aza inmekte ve sistemin verimi artmaktadır [2,5,7,28].

Bu kontrol yöntemi içerisinde en iyi sonucu yapay zekâ tabanlı kontrol algoritmaları vermektedir. Ancak yapay zekanın ihtiyaç duyduğu işlem gücü ve hafıza alanı için bu yöntem yalnızca gelişmiş işlemcili sistemlerde kullanılmaktadır. Günümüzde yüksek katlı yapıların artan asansör trafiği nedeniyle asansör kontrolünde yapay zekâ tabanlı kontrol sistemleri tercih edilmektedir.

BÖLÜM 3

GENETİK ALGORİTMA

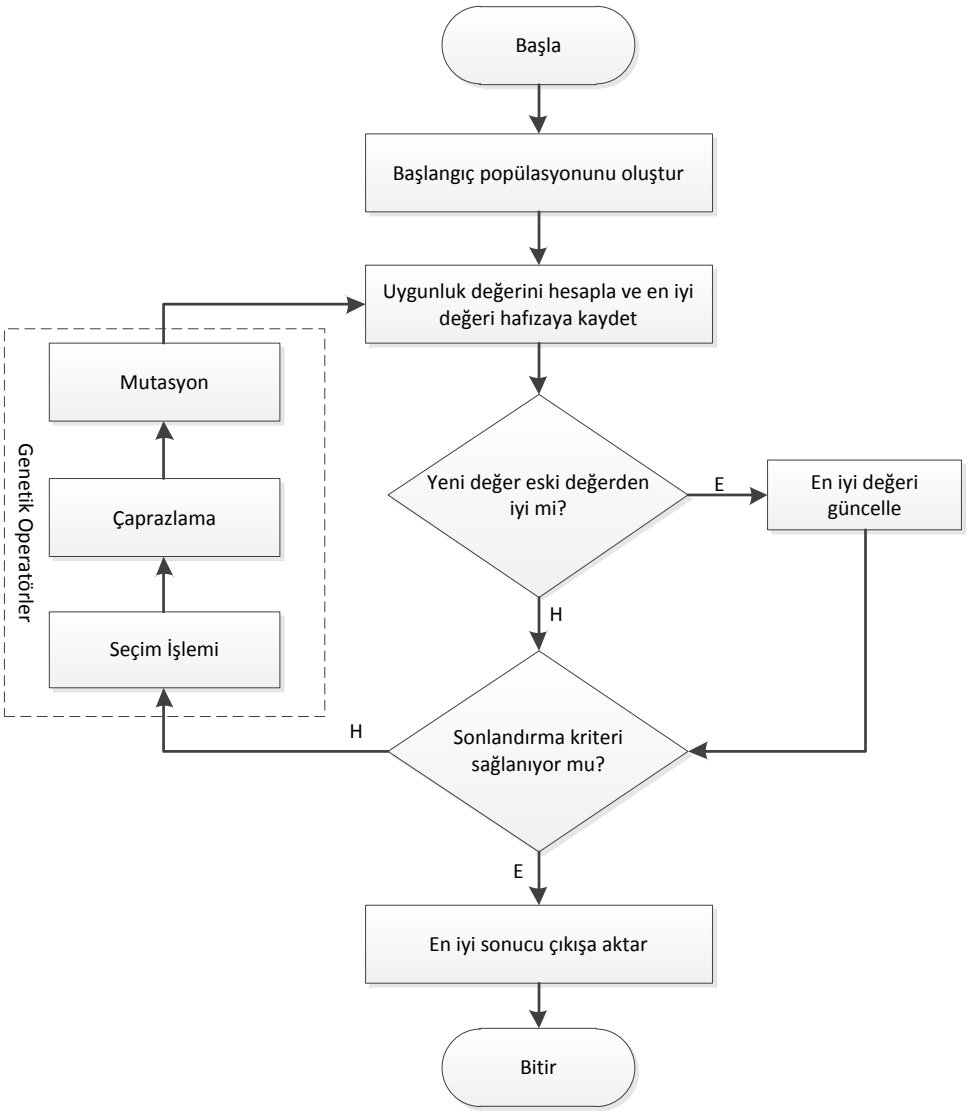
GA, canlıların doğal yaşamlarında hücresel ve bireysel anlamda yaşamak ve hayatta kalabilmek için geçirdikleri adımların taklit edilerek bir algoritmaya dökülmüş halidir. Canlıların milyonlarca yıldır başarılı bir şekilde yaşamlarını ve hayatta kalmalarını sağlayan bu algoritma, mühendislik problemlerinde en iyi sonucu veya en iyiye yakın sonucu bulmak için kullanılmaktadır.

"Genetik algoritma" terimi ilk kez 1967 yılında Bagley tarafından kullanılmıştır. 1975 yılında ise Holland GA'nın teorik bulgularını literatüre kazandırmıştır. Aynı yıl De jong GA'nın en iyileme amaçlı olarak kullanılabileceğini yapmış olduğu çeşitli deneysel çalışmalar ile göstermiştir [29–31]. 1980'li yılların başlarına kadar GA çok yaygın kullanılan bir algoritma değildi. GA ile yapılan çalışmaların çok azı dışındakiler ağırlıklı teorik çalışma olarak gerçekleştiriliyordu. Bu yıllarda Jong ve Hollstien, yaptıkları çalışmalar GA'nın yaygınlığının artırılmasında önemli katkılar sağlamıştır. Hollstein çalışmasında, farklı seçim ve eşleşme stratejilerinin GA'nın performansına ektisini analiz etmiştir. Jong ise çalışmasında GA'daki adaptif mekanizmanın dış hatlarını belirledi. 1990'lı yıllara Holland'ın öğrencisi Golldberg GA'nın başka alanlara da uyarlanabileceğini göstermesi ile GA hak ettiği ilgiye kavuşmuş ve birçok disiplinde çalışılmıştır. 1992 yılında Koza GA kullanarak genetik programlamayı geliştirmiştir. Devam eden yıllardaki uygulamalı ve teorik çalışmalar GA'nın uyarlanabilirliğini ve güvenilirliğini geliştirmiştir. Günümüzde birçok alanda GA etkili bir şekilde kullanılmaktadır [6].

3.1. GENETİK ALGORİTMANIN YAPISI

GA, araştırma ve optimizasyon problemlerinin çözümünde kullanılan adaptif metottur ve karmaşık bir probleme optimum bir çözüm ararken, biyolojik organizmaları genetik sürecine dayanan doğal seleksiyon yöntemini esas alır. Bu yöntemle probleme aranan

çözümler arasından en iyinin hayatta kalarak çözüm olarak seçilmesi sağlanır. GA'nın akış diyagramı Şekil 3.1'de verilmektedir [31,32].



Şekil 3.1. GA'nın akış diyagramı.

GA'da öncelikle problemin optimum çözümünü bulmak için rasgele çözümler üretilerek başlangıç popülasyonu oluşturulur. Oluşturulan tüm çözümlerin tek tek problem için uygunluğu hesaplanarak iyi bireylerin hayatta kalması sağlanır. En iyi birey ise hafızada saklanır. Hayatta kalan bireyler üzerinde bitirme kriteri sağlanana kadar sırayla GA'nın operatörleri olan seçim, çaprazlama ve mutasyon işlemleri gerçekleştirilir. Yeni nüfus üzerinde yeniden uygunluk değeri hesaplanır ve hafızada tutulan sonuçtan daha iyi bir sonucun olması dahilinde en iyi birey güncellenir ve çözüm olarak çıkışa aktarılır [29].

3.1.1. Genetik Algoritmada Kodlama Sistematiği

GA sezgisel bir algoritma olmasından dolayı problemin %100 doğrulukta çözümünü veremez. Bunun yerine bulduğu sonuçlar arasında en iyi olan çözümü çıkışa verir. Problemin çözümünü temsil eden bu her bir bireye kromozom denir. Kromozomlar problemin çözümü için gereken parametrelerin bir sıra dahilinde dizilmiş halidir. Parametre ise kromozom içeresindeki yer alan genler ile ifade edilir [6,32].

Kodlama sistematiği, algoritmanın programlanması esnasında kromozomun ne türde kodlanacağını ifade eder. Kodlama türleri, ikili (binary) kodlama, tamsayı (gerçel sayı) kodlama ve gray kodlama şeklindedir. GA'da problemin çözümü için yapılacak iyi bir kodlama, sistemin hızını artırmaya yardımcı olacaktır [6,32].

GA'da en yaygın kullanılan kodlama şekli ikili kodlamadır. İkili kodlamada değişkenler, genlere "1" ve "0" şeklinde değer verilerek ikilik sayı tabanına göre kodlanır. Çizelge 3.1'de bu gösterim verilmektedir.

Çizelge 3.1. Kromozomun ikili kodlama ile kullanımı.

Kromozomlar	**Genler**					
	Birinci Değişken			**İkinci Değişken**		
Kromozom 1	1	0	0	1	1	0
Kromozom 2	0	1	1	0	1	0

Kromozom içerisinde birden fazla değişken tutulacaksa kromozomun uzunluğu artırılır. Bu kodlamada sürekli olarak onluk sayı ile ikilik sayı arasında çevrim yapılacağı için çözümü bulma süresi artacaktır.

Gray kodlamada, ikili kodlama ile temelde aynı mantığa dayalıdır. Çizelge 3.2'de verildiği gibi değişkenlerin onluk tabandaki değerlerinin gray kodlu gösterimi ile ifade edilirler. Bu kullanım sayı tabanları arasındaki çevrimlerde ikinci bir algoritmaya ihtiyaç duymasından dolayı çözüm hızını negatif yönde etkilemektedir [6].

Çizelge 3.2. Kromozomun gray kodlama ile kullanımı.

Onluk Değer	İkili Kodlama	Gray Kodlama
0	[0 0 0 0]	[0 0 0 0]
1	[0 0 0 1]	[0 0 0 1]
2	[0 0 1 0]	[0 0 1 1]
3	[0 0 1 1]	[0 0 1 0]

Tam sayılı kodlamada ise, kromozom içerisinde yer alan değişkenler gerçel değerleri ile kodlanırlar. Çizelge 3.3'te içerisinde 4 değişkeni barındıran örnek kromozomlar verilmektedir. Bu kodlamada sayı tabanları arasında bir dönüşüm işlemi olmadığı için program daha hızlı çalışmaktadır. Ancak az sayıdaki değişken içeren problemlerde algoritmanın uygulanabilirliğini olumsuz etkilemektedir.

Çizelge 3.3. Kromozomun tamsayı kodlama ile kullanımı.

Kromozomlar	Değişkenler			
	Değ. 1	Değ. 2	Değ. 3	Değ. 4
Kromozom 1	10.271	1.028	-1.311	4.894
Kromozom 2	4.747	11.744	0.478	-8.167
Kromozom 2	2.457	-0.457	1.748	9.147

3.1.2. Başlangıç Popülasyonunun Oluşturulması

GA'da ilk adım probleme muhtemel çözüm olabilecek bireylerin (kromozomların) üretilerek başlangıç popülasyonunun oluşturulmasıdır. Başlangıç popülasyonu birçok problem için çoğunlukla rastgele oluşturulmaktadır. Ancak ele alınan problem ile ilgili tahmin edilebilen çözümlerin olması halinde başlangıç popülasyonu bu çözümlerin türetilmesi ile de oluşturulabilir. Optimizasyon problemlerinde rastgele oluşturulan bireyler problemlerin sınırları dışında çözümler üretebilir. Bu gibi durumlardan

kaçınmak için ele alınan probleme özgü sezgisel yöntemlerden yararlanılır [29,32].

3.1.3. Uygunluk Fonksiyonu ve Uygunluk Değeri

GA'da uygunluk fonksiyonu, ele alınan problemin optimizasyonu için gerekli olan matematiksel modeldir ve probleme özgüdür. GA sezgisel olmasından kaynaklı uygunluk fonksiyonunun modellemesi yapılırken sistemin birebir matematiksel modeline ihtiyaç duymaz. Ancak optimizasyonun gerçekleştirilebilmesi için matematiksel modelde belirleyici parametrelerin yer alması gerekmektedir [6,29].

Uygunluk değeri ise, popülasyonda yer alan bireylerin uygunluk fonksiyonu kullanılarak hesaplandığı çözüm değeridir. Problemin optimizasyonu yapılırken hesaplanan bu uygunluk değerleri kullanılır. Yapılan optimizasyon tipine göre her iterasyondaki en iyi birey saklanarak sistemi optimize edecek çözüm bulunur.

3.1.4. Seçim İşlemi

Genetik algoritmada seçim işlemi, mevcut popülasyon içerisinden bir sonraki nüfusun oluşumunda kullanılacak olan bireylerin belirlenmesidir. Popülasyon içerisinde yer alan bireylerin seçilme olasılıkları ağırlıklı olarak uygunluk değerleri ile orantılıdır. Tıpkı doğal yaşam sürecinde olduğu gibi uygunluk değeri yüksek olan veya başka bir ifade ile güçlü olan birey daha çok iterasyonda bulunarak daha uzun yaşarken uygunluk değeri düşük olan birey ise elenerek sonraki iterasyonlarda bulunamayacaktır [29,31,32].

Genetik Algoritmada kullanılmak üzere birçok seçim yöntemi bulunmasına rağmen rulet çemberi, turnuva ve elitizm seçim yöntemleri en yaygın kullanılanlardır. Rulet tekeri seçim yönteminde popülasyondaki tüm bireylerin uygunluk değerleri ile ilişkili olarak seçim işlemi yapılır. Öncelikle her bireyin rulet tekeri üzerindeki oranını bulmak için bireylerin uygunluk değeri popülasyondaki tüm uygunluk değerlerinin toplamına bölünür. Bu orana göre tüm bireyler küçükten büyüğe doğru sıralanır. İşlemci tarafından "0" ile "1" arasında rastgele üretilen bir değer sıra ile her bireyin rulet oranının kümülatif değeri ile karşılaştırılır ve rastgele değerden büyük kümülatif değere sahip bireyler bir sonraki popülasyon için havuza aktarılırlar. Çizelge 3.4'te örnek olarak yapılan bir rulet tekeri seçim hesaplaması verilmektedir [29,32].

Çizelge 3.4. Rulet tekeri seçim yöntemi örneği.

Birey	Uygunluk Değeri	Rulet Tekeri Değeri	Kümülatif Toplam
1	0,15	0,034	0,034
2	0,18	0,041	0,076
3	0,56	0,129	0,205
4	1,57	0,361	0,566
5	1,89	0,434	1,000

Turnuva seçim yönteminde de rulet tekerine benzer şekilde bireylerin uygunluk değerleri ile ilişkili seçim işlemi yapılır. Bu yöntemde kullanıcı tarafından bir turnuva büyüklüğü seçilir. Seçilen turnuva büyüklüğü kadar birey popülasyon içerisinden seçilir ve tüm bireyler uygunluk değerine göre birbiri ile kıyas edilir. Seçilen bu bireyler arasında en iyi olan birey havuza alınır. Yeni popülasyon tamamlanıncaya kadar bu işlem tekrar edilir. Turnuva büyüklüğünün küçük seçilmesi durumunda uygunluk değeri düşük olan bireylerin yeni popülasyonda bulunma ihtimali yükselecektir. Ancak turnuva büyüklüğünün popülasyon büyüklüğüne eşit seçilmesi durumunda yeni oluşturulacak popülasyonun tümü bir önceki popülasyondaki en iyi bireyden oluşacaktır. Turnuva büyüklüğünün en düşük değeri olan "1 birey" seçilmesinde ise bir sonraki popülasyon, tamamıyla rastgele seçilen bireylerden oluşacaktır [31].

Elitizm seçim yönteminde ise ana amaç popülasyondaki en iyi bireyin korunarak bir sonraki popülasyona aktarılmasıdır. Popülasyonun büyüklüğüne ulaşmak için gerekli olan diğer bireyler ise uygunluk değerlerini temel alan diğer seçim algoritmaları ile seçilir [6].

3.1.5. Genetik Operatörler

GA'da seçim işleminin tamamlanmış olması, optimum çözümün bulunması için tek başına yeterli değildir. Problemin optimum çözümü için her neslin yeni uygunluk değerine sahip bireylerle çeşitlendirilmesi gerekmektedir. Popülasyondaki bu genetik çeşitliliği sağlanmak için seçim işleminden farklı olarak çaprazlama ve mutasyon adlı genetik operatörler kullanılmaktadır [29,30].

3.2.5.1. Çaprazlama İşlemi

Çaprazlama işlemi, popülasyon içerisinden rastgele seçilmiş iki bireyin rastgele bir ve birkaç noktadan sıralı haldeki gen dizisi değişimidir. Bu gen dizisi tek bir noktadan başlayıp son gene kadar devam edebileceği gibi birçok noktadan bölünerek parçalı olarak da iki kromozom arasında değişebilir. Bu işlemdeki amaç, tıpkı doğal yaşam sürecinde olduğu gibi güçlü genlerin bir sonraki nesle aktarılarak daha güçlü bireylerin üretilmesini sağlamaktır. Kullanılan en temel çaprazlama yöntemi tek noktadan çaprazlamadır. Ancak bunun yanı sıra iki noktadan ve çok noktadan da çaprazlama işlemi yapılmaktadır [6,29,32].

Çaprazlama işlemi popülasyona belli bir oranda uygulanır ve buna çaprazlama oranı denir. Çaprazlama oran "0" ile "1" arasında sabit bir sayı olarak seçilebileceği gibi her iterasyonda yine bu aralıkta rasgele olarak da belirlenebilir. Çaprazlama işleminin yapılacağı bireylerin belirlenmesi için tıpkı seçim işleminde olduğu gibi her bir bireye "0" ile "1" arasında rasgele değerler verilir. Üretilen bu değerler çaprazlama oranı ile karşılaştırılarak çaprazlama oranından düşük olan bireyler çaprazlama işlemi için bir havuza alınır.

Çaprazlama işlemi için çift sayıda bireye ihtiyaç vardır. Çaprazlama havuzunda tek sayıda birey olması durumunda bireylerden biri çıkarılarak bu durum düzenlenir. Havuz içerisinden bireyler alınarak çaprazlama türüne göre rastgele noktalar belirlenir ve yeni bireyler üretilir.

Çaprazlama oranının yüksek seçilmesi popülasyon içerisinde daha fazla sayıda bireyin çaprazlamaya tabi tutularak yeni bireylerin üretilmesini sağlayacaktır. Bu sayede çözümün problemin yerel en iyi çözümlerine takılması engellenmiş olacaktır. Ancak bu oranın çok yüksek seçilmesi durumunda çözüm uzayının çok fazla noktasında arama yapılacağı için problemin istenen çözümü için harcanan zaman artacaktır [29,30].

Tek noktadan çaprazlama işleminde, rastgele seçilen bir çaprazlama noktasından (ÇN) başlayarak kromozomdaki son gene kadar gen dizisi karşılıklı olarak iki kromozom arasında Çizelge 3.5'te verildiği gibi değiştirilir.

Çizelge 3.5. Tek noktadan çaprazlama.

Kromozomlar	**Tek Noktadan Çaprazlama**							
						ÇN		
Kromozom 1	0	1	1	0	0	1	1	0
Kromozom 2	1	1	0	1	0	1	0	0
Kromozom 1	0	1	1	0	0	1	0	0
Kromozom 2	1	1	0	1	0	1	1	0

İki noktadan çaprazlama işleminde, kromozomun üzerinde rastgele iki nokta seçilir ve bu iki nokta arasında kalan gen dizisi Çizelge 3.6'da verildiği gibi değiştirilir.

Çizelge 3.6. İki noktadan çaprazlama.

Kromozomlar	**İki Noktadan Çaprazlama**							
		ÇN		**ÇN**				
Kromozom 1	0	1	1	0	0	1	1	0
Kromozom 2	1	1	0	1	0	1	0	0
Kromozom 1	0	1	0	1	0	1	1	0
Kromozom 2	1	1	1	0	0	1	0	0

Çok noktadan çaprazlama işleminde ise, iki noktadan çaprazla işlemine benzer şekilde birden fazla çift sayıda nokta belirlenerek bu nokta çiftleri arasında kalan gelen değiştirilir. Çizelge 3.7'de çok noktadan çaprazlama işlemine örnek verilmektedir.

Çizelge 3.7. Çok noktadan çaprazlama.

Kromozomlar	**Çok Noktadan Çaprazlama**							
		ÇN	**ÇN**			**ÇN**		**ÇN**
Kromozom 1	0	1	1	0	0	1	1	0
Kromozom 2	1	1	0	1	0	1	0	0
Kromozom 1	0	1	0	0	0	1	0	0
Kromozom 2	1	1	1	1	0	1	1	0

3.2.5.2. Mutasyon İşlemi

Mutasyon işlemi, popülasyon içerisindeki yakınsamayı ortadan kaldıran ve çeşitliliği sağlayan genetik operatördür. Mutasyon aynı zamanda ikili kodlamada kaybolan bitlerin yeniden oluşmasını da sağlamaktadır. Mutasyon işlemi popülasyon üzerinde belirli bir oranda gerçekleştirilir. Bu orana mutasyon oranı adı verilir. Bu oranın yüksek seçilmesi, popülasyondaki tüm bireylerin değişerek yüksek nitelikteki bireylerin kaybolmasına ve çözüm süresinin uzamasına neden olmaktadır[29,30,32].

Seçim işleminde kullanılan yönteme bağlı olarak seçim sonrasında tüm bireylerin aynı bireyden oluşma ihtimali vardır. Bu durumda genetik operatörlerden olan çaprazla işleminin aynı bireyler arasında yapılmış olması çeşitliliğe hiçbir katkı sağlamayacaktır. Çaprazlama işleminin aksine mutasyon işleminde bireyler tekil olarak ele alınır ve seçilen gen veya genler üzerinde mutasyon yapılarak çeşitlilik artırılır.

Mutasyon işleminde öncelikle popülasyonda yer alan tüm bireylerin bütün genleri için “0” ile “1” arasında rastgele değerler üretilir. Üretilen bu gen değerlerinin tümü mutasyon oranı ile karşılaştırılır ve mutasyon oranından küçük değerlerin olduğu genlerde mutasyon işlemi yapılır. Mutasyon operatörü problemin yapısına göre değilleme, ters çevirme, ekleme, yer değişimi ve karşılık değişim olarak beş farklı şeklinde uygulanır.

Değilleme yöntemi yalnızca ikili kodlamada kullanılan bir yöntemdir. Bu yöntemde mutasyona tabi tutulacak genin değeri “0” ise “1”, “1” ise “0” yapılır. Ters çevirme yönteminde aynı kromozom üzerinde belirlenen iki gen arasındaki gen dizilimi alınır ve ters çevrilerek tekrar yerine konulur. Ekleme yönteminde kromozom üzerinden rastgele seçilen bir parça alınır ve kromozom içerisinde başka bir yere konulur. Yer değişimi yönteminde kromozom üzerinden rastgele bir gen dizisi alınır ve aynı kromozom üzerinde rastgele bir yere yerleştirilir. Karşılıklı değişim yönteminde ise aynı kromozom üzerinden iki gen alınır ve kendi aralarında değiştirilir [6,29].

Çizelge 3.8’de örnek bir ana bireye mutasyon yöntemleri uygulanmış ve mutasyona tabi olan genler vurgulu şekilde belirtilerek verilmektedir.

Çizelge 3.8. Mutasyon yöntemleri örnekleri.

		Kromozom Genleri							
	Ana Birey	0	1	1	0	0	1	1	0
Mutasyon Yöntemleri	Değilleme	0	0	1	0	0	1	1	0
	Ters Çevirme	0	0	0	1	1	1	1	0
	Ekleme	1	0	0	1	1	0	0	1
	Yer Değişimi	0	1	1	1	0	1	0	0
	Karşılıklı Değişim	1	1	1	0	0	1	0	0

3.1.6. Durdurma Kriteri

GA problemlerin minimum veya maksimum çözümlerini ararken asla yüzde yüz doğru sonucu garanti etmez. GA sezgisel arama yapılmasından kaynaklı birçok noktada arama yapar ve bir durdurma kriteri belirtilmediği taktirde sonsuza kadar çalışır. GA’nın durdurulması hesaplama zamanı, optimizasyon hedef değeri ve minimum iyileşme şeklinde üç kritere bağlı olarak gerçekleştirilir. Hesaplama zamanı kriterinde, kullanıcı sonlandırma kriteri olarak belirli bir iterasyon sayısı veya hesaplama zamanı

belirler. Algoritma bu değerlerden birine ulaştığında sonlandırılır. Optimizasyon hedef kriterinde problem için ulaşılmak istenen uygunluk değeri bilinmektedir. Algoritmanın bu değere ulaşması durumunda algoritma sonlandırılır. GA sonuca yaklaştıkça çözüm değerindeki iyileşme çoğunlukla azalmaktadır. Minimum iyileşme kriteri de bu azalma oranını sonlandırma kriteri olarak belirler. Bu kriterde, her iterasyondaki en iyi çözüm bir önceki iterasyonda bulunan en iyi ile karşılaştırılır. Bu iki değer arasındaki iyileşme oranı kullanıcı tarafından belirlenen iyileşme oranından düşük olması durumunda algoritma sonlandırılır [29,32].

3.2. GENETİK ALGORİTMANIN KULLANIM ALANLARI

GA, problem çözümlerinde probleme özgü olarak kullandığı uygunluk fonksiyonu için sistemin matematiksel modeline yüzde yüz benzer bir fonksiyona ihtiyaç duymaz. GA için temel parametrelerin baz alınarak hazırlandığı bir uygunluk fonksiyonu yeterlidir. Karmaşık ve modellenmesi güç problemlerde bu durum benzetim için ciddi kolaylıklar sağlar ve kısa sürede problem için yüksek doğrulukta çözümler bulunabilir. GA sezgisel aramalar yapmasından ötürü çözüm uzayının geniş olduğu problemlerde yerel optimumlara takılmadan problemin en iyi çözümüne yakın değerler verebilmektedir. Bu üstünlüklerinden ötürü GA, optimizasyon tabanlı mühendislik problemlerinde, makine öğrenmesinde, ekonomik tahminlerde, sosyal sistemlerinin analizlerinde, oyun programcılığında ve matematik problemlerinin çözümleri gibi bir çok alanda başarılı bir şekilde kullanılmaktadır [6].

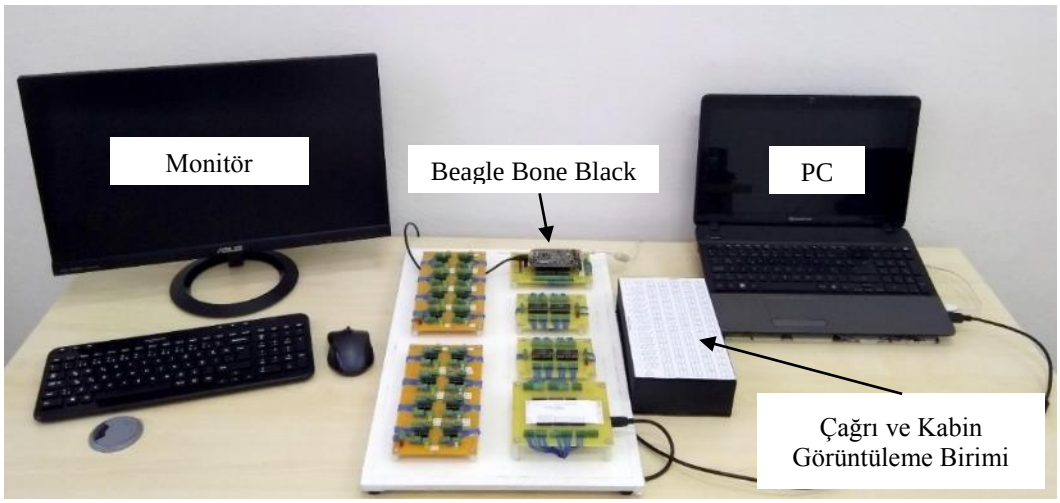
BÖLÜM 4

KABİN YÖNLENDİRME İŞLEMİNİN TESTİ İÇİN DENEY DÜZENEĞİ VE SİMÜLATÖR PROGRAMININ GELİŞTİRİLMESİ

Asansör kontrol sistemlerinin maliyetli ve kurulumun zorluğu nedeniyle kontrol ve optimizasyon algoritmaları ağırlıklı olarak benzetim programlarında ve deney düzenekleri üzerinde test edilerek geliştirilmektedir. Bu bölümde kabin yönlendirme optimizasyon algoritmasının testlerinin yapıldığı simülatör programı ve gömülü donanımın testlerinin yapıldığı deney düzeneği detaylı bir şekilde anlatılmaktadır.

4.1. DENEY DÜZENEĞİNİN GENEL YAPISI

Hazırlanan deney düzeneği temel olarak gömülü sistem etrafına konumlandırılmış baskı devre kartlarından ve gömülü sisteme bağlı klavye, mouse ve monitörden oluşmaktadır. Şekil 4.1'de deney düzeneğinin resmi verilmektedir.

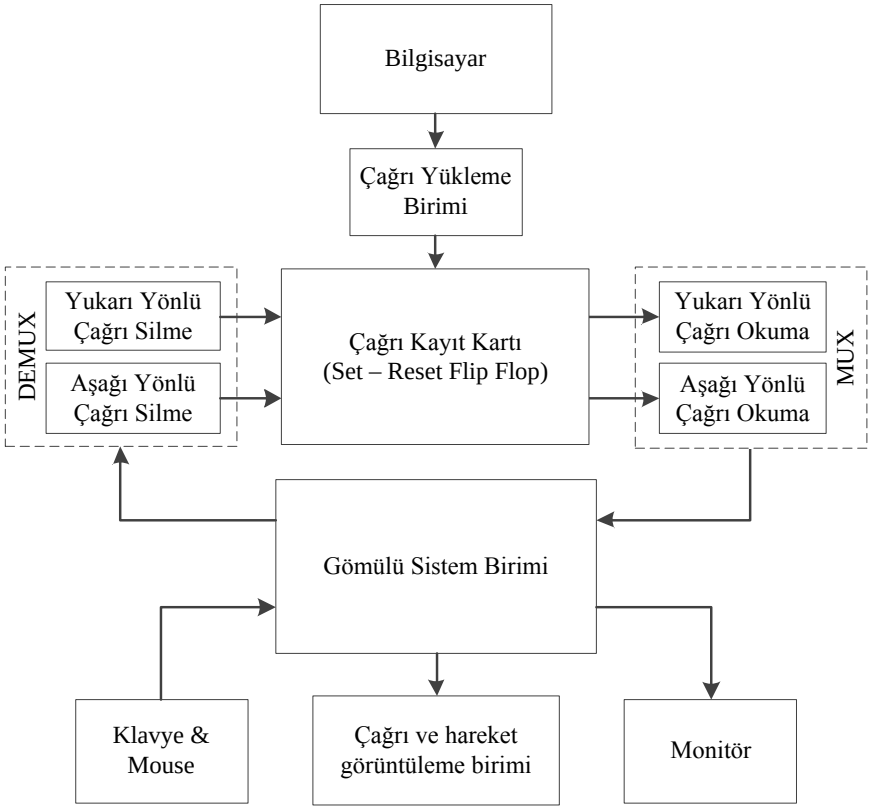


Şekil 4.1. Deney düzeneğinin resmi.

Gömülü donanım ve çevre birimleri 6 karttan ve genel olarak 5 birimden oluşmaktadır. Bunlar; çağrıların kaydedildiği "Çağrı kayıt kartı", çağrı senaryolarını kayıt kartına

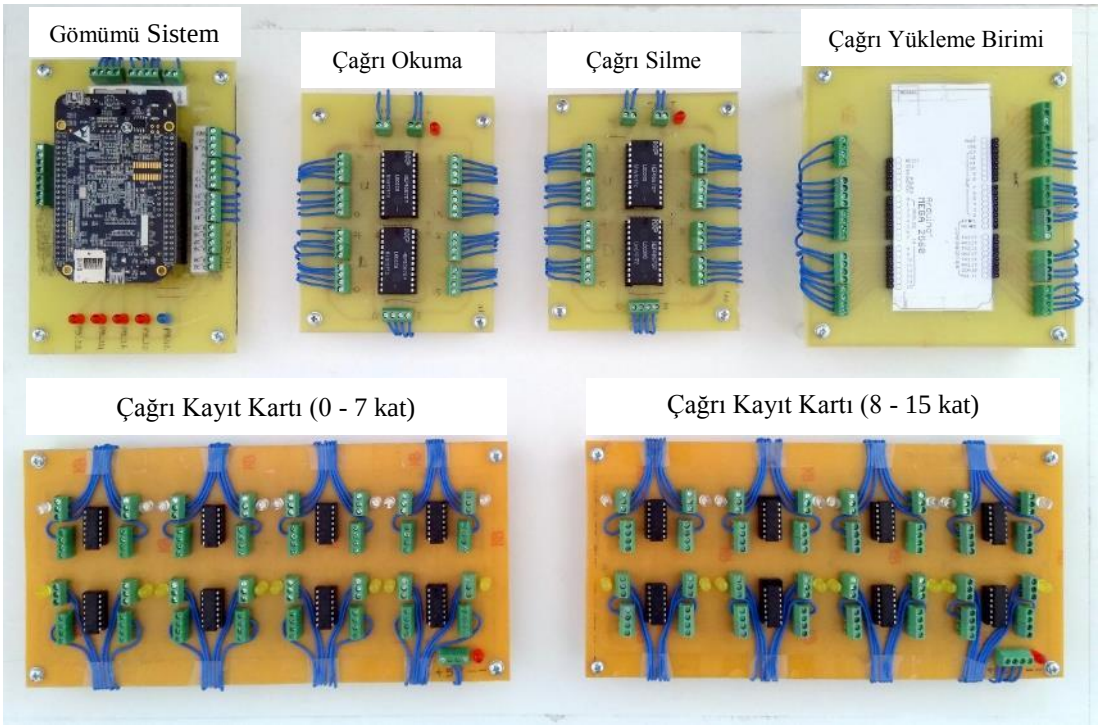
yükleyen “Çağrı yükleme birimi”, kayıt kartı üzerinden çağrının okunduğu ve silindiği “Çağrı okuma ve silme birimi”, simülasyon ile kontrol algoritmasını içerisinde barındıran “Gömülü sistem birimi” ve kat çağrılarının ve kabin hareketlerinin görüntülendiği “çağrı ve kabin görüntüleme birimi” şeklindedir.

Deney düzeneği temelde çağrıların yüklenmesi ve cevaplanarak kayıt kartından silinmesi şeklinde çalışmaktadır. Algoritmanın çalışması ile birlikte sırayla, çağrıların kayıt kartına yüklenmesi, kayıt kartından çağrıların okunması, yönlendirme optimizasyonunun yapılması, simülatör programının çalıştırılarak cevaplanan çağrıların kayıt kartından silinmesi ve kabin hareketlerinin görüntülemesi şeklindedir. Bu adımlar Şekil 4.2’deki blok şemasında verilmektedir.



Şekil 4.2. Deney düzeneğinin blok şeması.

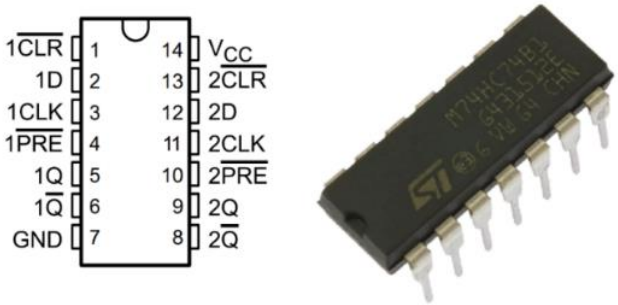
Gömülü sistemin mevcut asansör kartlarına uygulanabilmesi için çağrılar fiziksel ortamdan alınırken, algoritmanın testi ve uygunluğu için ise asansör hareketleri ve konumları simülatör yazılımı içerisinde sanal ortamdan alınmaktadır. Şekil 4.3’te hazırlanan donanımların resmi verilmektedir.



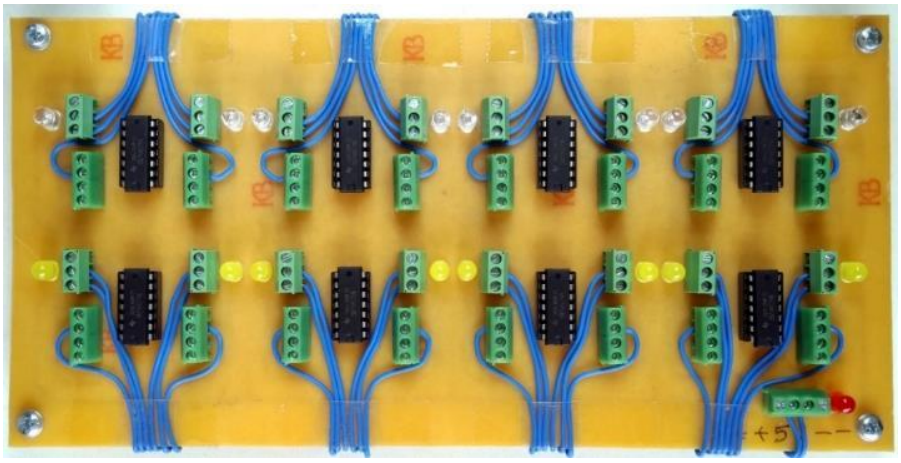
Şekil 4.3. Gömülü sistem ve baskı devre kartları.

4.1.1. Çağrı Kayıt Kartı

Bu birim, GAS'taki yukarı ve aşağı yönlü kat çağrılarını hafızada tutmak için kullanılmaktadır. Birim, benzetimi yapılan 16 katlı bina modeli için 15 yukarı yönlü ve 15 aşağı yönlü çağrıları kaydetmek için birbirinin aynısı iki karttan oluşmaktadır. Çağrı kayıt kartı tıpkı gerçek asansör kontrol kartlarında olduğu gibi set-reset flip-flop mantığına dayalı olarak çalışmaktadır. Bu birim, çağrıların hafızada tutulması için şekil 4.4'te verilen 74HC74N D tip flip-flop entegresi ile şekil 4.5'te verilen karttan iki adet kullanılarak oluşturulmuştur.

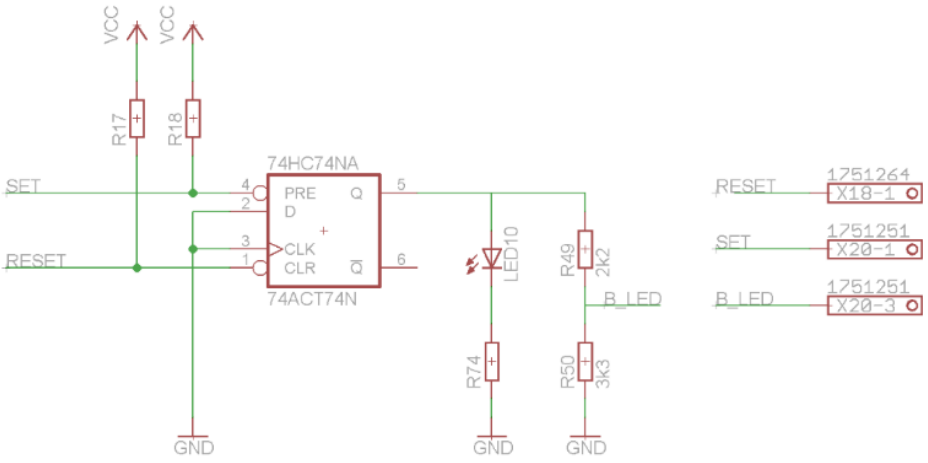


Şekil 4.4. 74HC74N entegresi ve bacak bağlantısı.



Şekil 4.5. Çağrı kayıt kartı üst görünüş.

Şekil 4.6'daki devre şemasında verilen "SET" etiketli klemensten gelen lojik sıfır değeri ile çağrı kaydedilmektedir. Kaydedilen çağrı "B_LED" etiketli klemens üzerinden okunmakta ve çağrının cevaplanması ile birlikte "RESET" etiketli klemense verilen lojik 0 değeri ile hafızadan silinmektedir. Bu birim şekil 4.6'da verilen temel devre şemasından 16 adet kurularak hazırlanmıştır.



Şekil 4.6. Çağrı kayıt kartı çağrı kaydetme devre şeması.

4.1.2. Çağrı Yükleme Birimi

Çağrı yükleme birimi gömülü sistem ve kontrol algoritmasının testleri için gerekli olan çağrı senaryolarının kayıt kartına yüklenmesinde kullanılmaktadır. Bu birimde gömülü donanım olarak Arduino Mega tercih edilmiştir. Arduino Mega'nın şekil 4.7'de resmi ve Çizelge 4.1'de teknik özellikleri verilmektedir [33].

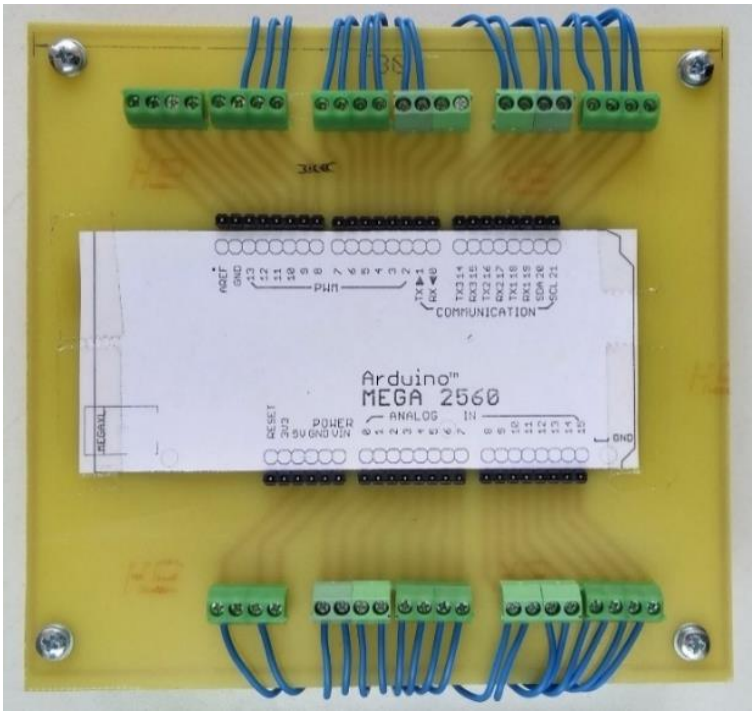
Arduino Mega'nın programlanması Windows tabanlı ve açık kaynak kodlu kendi yazılımı olan Arduino IDE isimli derleyici ile USB port üzerinden gerçekleştirilmektedir.

Şekil 4.7. Arduino Mega gömülü sistem geliştirme kartı.

Çizelge 4.1. Arduino Mega teknik özellikleri.

Özellikler	**Açıklamalar**
Kart işlemcisi	ATmega2560
İşlemci çalışma gerilimi	DC 5V
Kart çalışma gerilimi	DC 6 – 20 V
Dijital giriş çıkış sayısı	54 Adet (15 PWM)
Kart besleme gerilimi	DC 7 - 12 V
Analog giriş sayısı	16 Adet(10 Bit ADC)
Dijital pin akımı	20 mA
Programlama hafızası	256 KB
SRAM	8 KB
EEPROM	4 KB
Çalışma frekansı	16 MHz
Haberleşme protokolleri	4 Adet UART
Boyut / ağırlık	101.52 mm x 53.3 mm / 37g

Çağrı yükleme birimi ek bir elektronik komponent kullanmadan Arduino Mega için çıkışların dönüştürüldüğü bir klemens kartı şeklinde hazırlanmıştır. Hazırlanan kartın resmi Şekil 4.8'de verilmektedir.



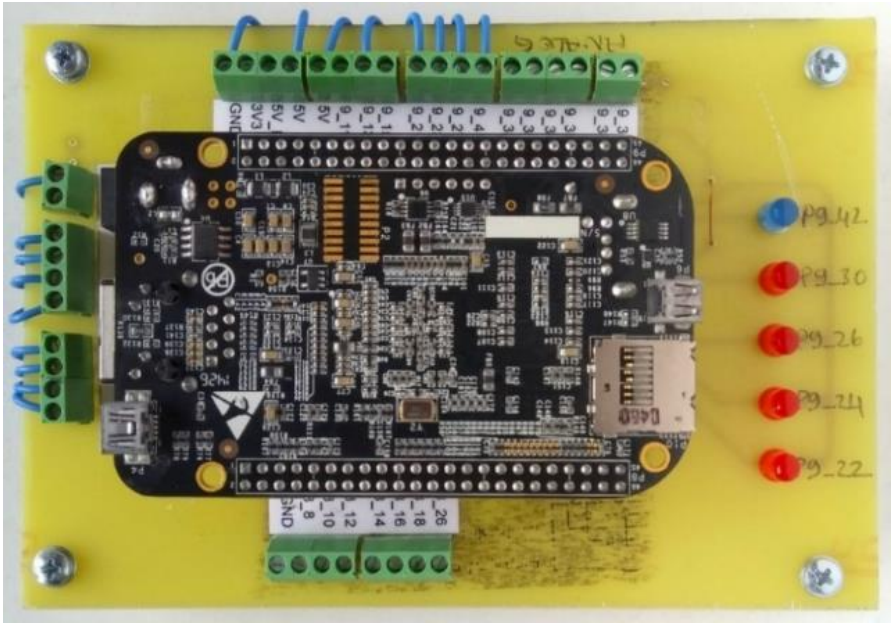
Şekil 4.8. Çağrı yükleme birimi üst görünüş.

Çağrı senaryoları, birim içerisinde bulunan Arduino Mega sayesinde hiçbir gecikme meydana gelmeden istenilen zaman ve sıklıkta simülatör için hazırlanmaktadır. Çağrıların buton sistemi yerine burada olduğu gibi bir işlemci ile kayıt kartına yazılmış olması, testlerde senaryoların sağlıklı işletilmesi açısından büyük kolaylıklar sağlamaktadır.

4.1.3. Gömülü Sistem Birimi

Gömülü sistem birimi, GAS'ta kabin yönlendirme optimizasyonunu ve asansör simülasyonunu gerçekleştirmektedir. Birim ek bir elektronik komponent kullanmadan Beaglebone Black için çıkışların dönüştürüldüğü bir klemens kartı şeklinde tasarlanmıştır.

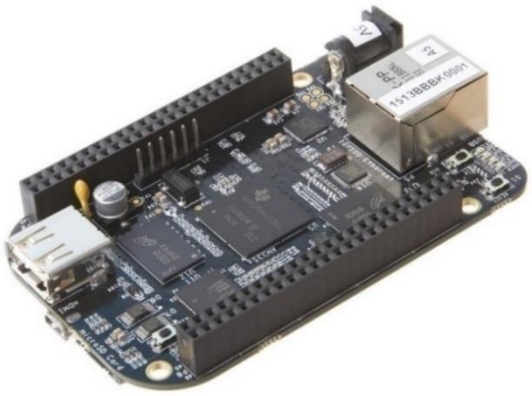
Yapay zekâ tabanlı optimizasyon işleminin gerçekleşebilmesi için yüksek işlem ve hafıza kapasitesine sahip bir işlemci gerekmektedir. Bu nedenlerden dolayı ARM (Advanced RISC Machines) mimarisine sahip Beaglebone Black Rev C geliştirme kartı gömülü donanım olarak seçilmiştir. Hazırlanan kartın görüntüsü Şekil 4.9'da ve gömülü sistemin Şekil 4.10'da resmi ve Çizelge 4.2'de özellikleri verilmektedir [34].



Şekil 4.9. Gömülü sistem birimi üst görünüş.

Çizelge 4.2. Beaglebone Black Rev C teknik özellikleri.

Özellikler	**Açıklamalar**
Merkezi işlemci	Sitara AM3358BZCZ100 ARM® Cortex-A8
Grafik motoru	SGX530 3D, 20M Polygons/S
Çalışma frekansı	1 Ghz
Hafıza	4 GB Embedded MMC
İşlemci çalışma gerilimi	DC 3.3V
Kart çalışma gerilimi	DC 5 V
Analog giriş sayısı	7 Adet (12 Bit ADC)
GPIO / PWM sayısı	67 / 8 Adet
SDRAM / EEPROM	512MB DDR3L 800MHZ / 4KB
Onboard Flash	4GB, 8bit Embedded MMC
Bağlantılar / Soketler	Micro HDMI, USB, Micro SD, ETHERNET
Video / Ses	16b HDMI, 1280x1024 (MAX) 1024x768, 1280x720, 1440x900, 1920x1080@24Hz w/EDID Support / HDMI portu ile streo ses
Genişletilebilir bağlantılar	Power 5V, 3.3V , VD.D_ADC(1.8V) 3.3V I/O on all signals McASP0, SPI1, I2C, GPIO(69 max), LCD, GPMC, MMC1, MMC2, 7 AIN(1.8V MAX), 4 Timers, 4 Serial Ports, CAN0, EHRPWM(0,2),XDMA Interrupt, Power button, Expansion Board ID (Up to 4 can be stacked)
Boyut / ağırlık	86.4 x 53.3 mm / 39 g



Şekil 4.10. Beaglebone Black Rev C gömülü sistem geliştirme kartı.

Beaglebone Black, Arduino Mega'dan farklı olarak içerisinde Linux işletim sistemine sahip bir geliştirme kartıdır. Bu nedenle kart, gömülü donanım olarak kullanılmak istenildiğinde Linux işletim sistemi üzerinde kartın mimarisine uygun uygulama geliştirilmesi gerekmektedir.

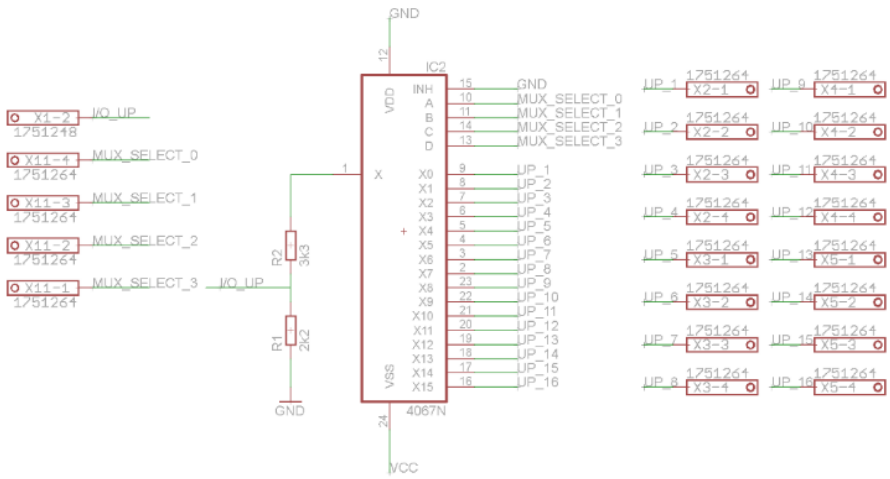
Bu çalışmadaki uygulamalar, Linux işletim sistemi içerisindeki Terminal'de doğrudan çalışabilen Python programlama dilinde geliştirilmiştir. Geliştirilen yazılımlar hata arama, görüntüleme ve saklamadaki kolaylıklar nedeniyle karta yüklü Linux'ta yer alan "Text Editör" yerine Şekil 4.11'de verilen Windows işletim sistemindeki "IDLE(Python GUI)" da hazırlanmıştır.

Şekil 4.11. IDLE (Python GUI) editörü (sol) ve yorumlayıcısı (sağ).

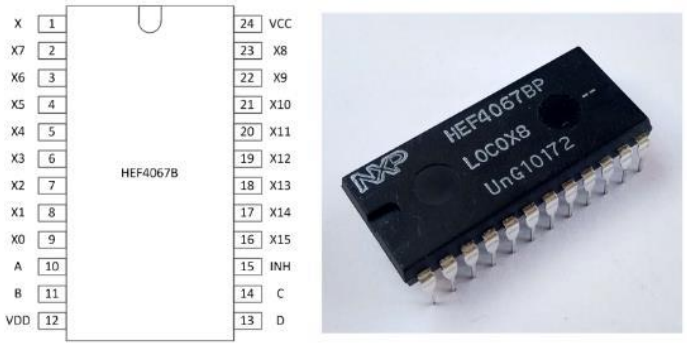
Windows tarafında hazırlanan uygulamalar, USB bağlantısı üzerinden Beaglebone'un kalıcı hafızasına kopyalanarak Linux tarafına aktarılmaktadır. Aktarma işleminin ardından karta yüklü Linux sistemi başlatılarak, Terminal üzerinden uygulama çağrılıp çalıştırıldığında simülatör ve yönlendirme işlemi gerçekleşmektedir.

4.1.4. Çağrı Okuma ve Silme Birimi

Çağrı Okuma ve Silme Birimi, "Gömülü Sistem Biriminin" çağrıları okumasında ve silmesinde yardımcı eleman olarak kullanılmaktadır. Birim ile kayıt kartındaki yukarı/aşağı yönlü 30 çağrı kayıt okunmakta ve silinmektedir. Çağrı okuma ve silme birimi HEF4067B 16 Kanal Analog Mux/Demux entegresi ile Şekil 4.12'de verilen temel devre şemasından iki adet kurularak hazırlanmıştır. HEF4067B entegresinin resmi Şekil 4. 13'te verilmektedir.

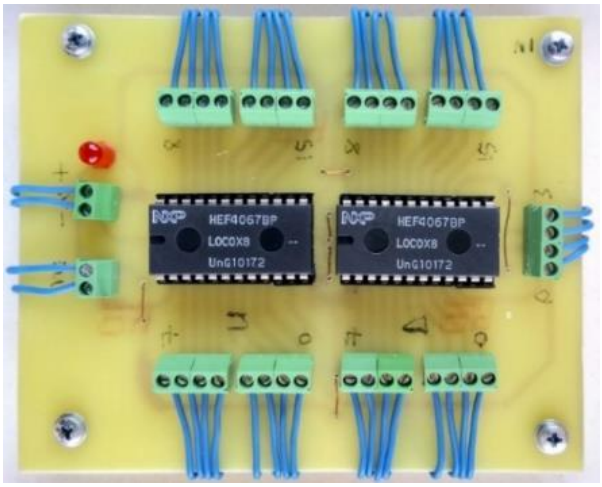


Şekil 4.12. Çağrı okuma silme birimi için port çoğullama devre şeması.



Şekil 4.13. HEF4067 entegresi ve bacak bağlantısı.

Çağrı okuma ve silme birimi Şekil 4.14'te verilen karttan iki adet kullanarak oluşturulmuştur. Kurulan bu alt birim sayesinde gömülü sistem üzerindeki port sayısı verimli şekilde kullanılmış ve 30 okuma/silme için ihtiyaç duyulan 60 dijital giriş çıkış sayısı 12'ye düşürülmüştür.



Şekil 4.14. Çağrı okuma silme birimi.

4.1.2. Çağrı ve Kabin Görüntüleme Birimi

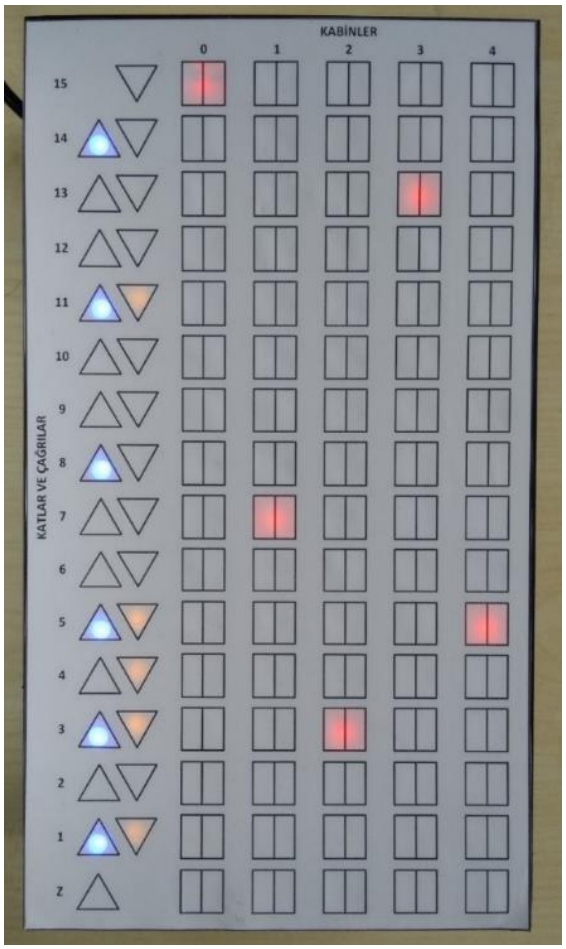
Çağrı ve kabin görüntüleme birimi, GAS'taki kabinlerin konumlarını ve katlardan gelen çağrıların görüntülenmesi için kullanılmaktadır. Bu birim sayesinde asansörlerin çağrıları cevaplamak için gerçekleştirdikleri hareketi görüntülemeye ve sistemin daha etkin yorumlanmasına imkân sunmaktadır. Birim dotmatrix led mantığına dayalı olarak hazırlanmıştır. Birim içerisinde MAX7219 sürücü kartı ve Arduino Uno denetleyicisi kullanılmaktadır. Arduino Uno'nun Şekil 4.15'te resmi ve Çizelge 4.3'te teknik özellikleri verilmektedir [35].

Şekil 4.15. Arduino Uno gömülü sistem geliştirme kartı [35].

Birim seri haberleşme ile gömülü donanım biriminden aldığı kabin konumları ve çağrıları görsel hale çevirmektedir. Şekil 4.16 ve Şekil 4.17'de birimin resmi verilmektedir.

Çizelge 4.3. Arduino Uno teknik özellikleri.

Özellikler	**Açıklamalar**
Kart işlemcisi / Frekansı	ATmega2560 / 16 Mhz
İşlemci / kart çalışma gerilimi	DC 5V / DC 6 – 20 V
Dijital giriş çıkış sayısı	14 Adet (6 PWM)
Analog giriş sayısı	6 Adet(10 Bit ADC)
Dijital pin akımı	20 mA
Programlama hafızası	32 KB
SRAM / EEPROM	2 KB
Haberleşme protokolleri	1 Adet UART
Boyut / ağırlık	68.6 mm x 53.4 mm / 25 g



Şekil 4.16. Çağrı ve kabin hareketleri görüntüleme birimi.

Şekil 4.17. Çağrı ve kabin hareketleri görüntüleme birimi devre resmi.

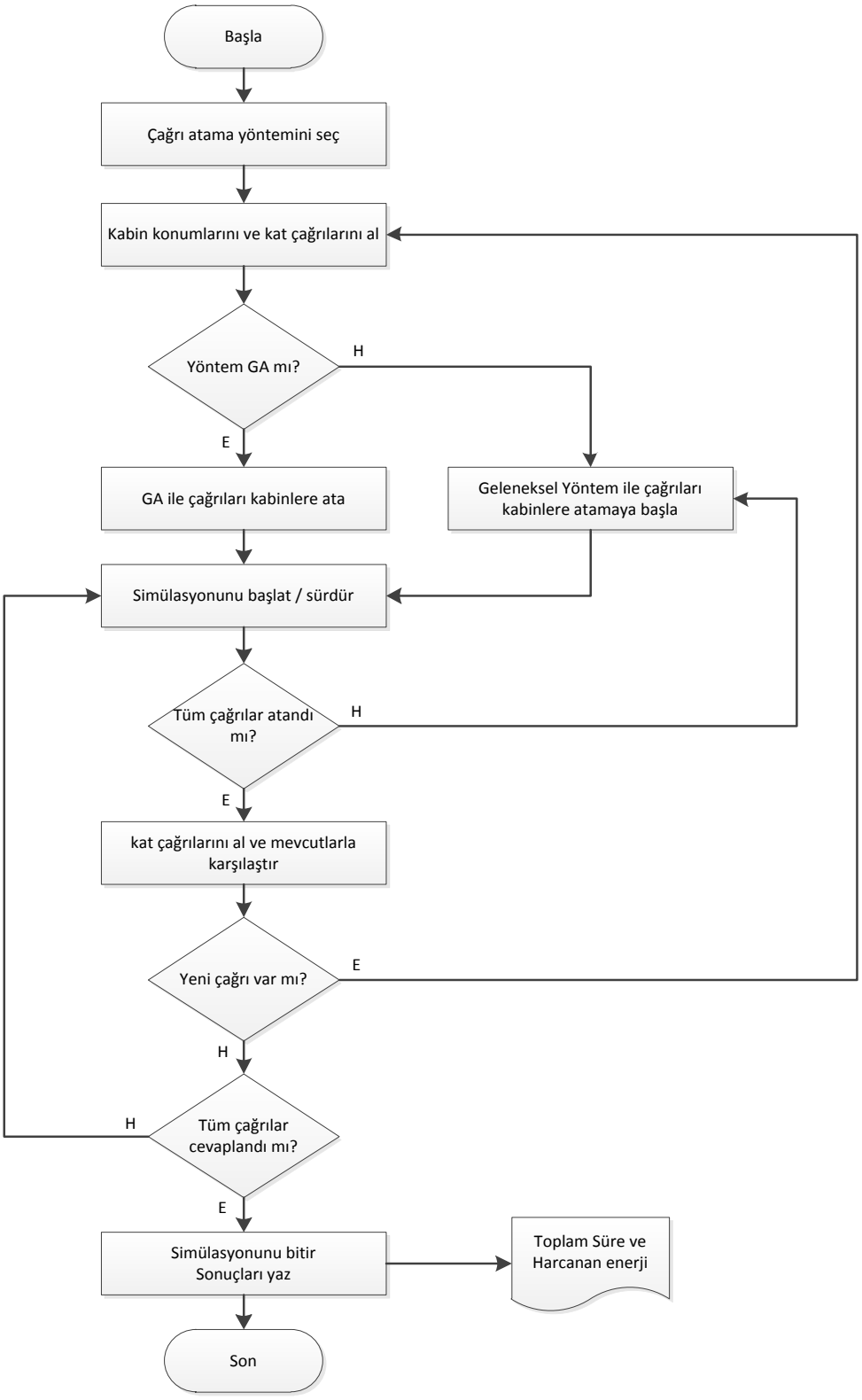
4.2. GELİŞTİRİLEN SİMÜLATÖR YAZILIMININ GENEL YAPISI

Asansör sistemlerinin kontrolü için önerilen algoritmaların testlerinde simülasyon programları kullanıcıya zaman, maliyet, okunabilirlik, gözlem kolaylığı ve anlık müdahale gibi birçok konuda ciddi kolaylıklar sağlamaktadır. Tüm bu durumlar göz önünde bulundurulduğunda, ihtiyaca dönüşen durum için çeşitli akademik çalışmalar gerçekleştirilmiş ve asansör algoritmalarının testleri için farklı bina ve trafik senaryolarını içeren simülatör programları geliştirilmiştir. Ancak hazırlanan bu simülatör programları gerçek dünyadan anlık çağrı ve trafik verilerini almaktan ziyade, bu verileri veri tabanından alarak yahut rastgele oluşturarak simülasyonlarını yapmaktadır. Ayrıca bu simülatörler gömülü bir sistemin testinden ziyade önerilen algoritmaların testine yönelik çalışmaktadırlar. Bu nedenlerden dolayı bu tez çalışmasında kullanılan simülatör programı gömülü donanımın ve önerilen algoritmanın testini yapacak şekilde donanım üzerine yazılmıştır.

4.2.1. Simülatör Programı Simülasyon Algoritması

Geliştirilen simülatör programında 5 asansörlü 16 katlı bir bina modellenmiştir. Simülatör programı Beaglebone Black üzerine kurulu Linux işletim sisteminde Python programlama dili kullanılarak kodlanmıştır. Simülatör programında öncelikle kabin atama yöntemi seçilir. Belirlenen kabin atama yöntemi sonrasında kabinlerin konumları ile kat çağrıları alınır ve atama işlemi yapılır. Atama işleminin ardından yönlendirme simülasyonu başlar. Yönlendirme esnasında yeni bir çağrı gelirse bu

işlemler tekrar edilir. Yeni bir çağrı gelmez ise simülasyon bitirilir. Sistemin akış diyagramı Şekil 4.18'de verilmektedir.



Şekil 4.18. GAS simülasyon programı akış diyagramı.

Simülatör programı temel olarak iki kısımdan oluşmaktadır ve iki kısımda aynı yazılım içerisinde birer fonksiyon gibi çalışmaktadır. Birinci kısım GAS'taki kabin hareketlerinin gerçek zamanlı olarak gerçekleştirilmesini kapsarken, ikinci kısım ise yönlendirme için seçilen kontrol algoritmasından oluşmaktadır. Daha önceden de bahsedildiği gibi simülatörün gömülü sistem üzerinde çalışıyor olması çağrıların gerçek dünyadan alınabilme ve gömülü donanımın testine de imkân sunmaktadır.

4.2.2. Simülatör Programı Enerji Tüketimi ve Çağrı Cevaplama Süresi Hesabı

Simülatör programında süre ve enerjinin hesaplanması, farklı kontrol algoritmalarının aynı senaryoya verdikleri sonuçların karşılaştırılması için yapılmaktadır. Süre hesabı, ilk kabin hareketi ile başlayıp son çağrının cevaplanmasına kadar geçen zamanı vermektedir. Çağrının cevaplanması sürecindeki kapıların açılıp kapanması, kabinin hızlanıp yavaşlaması ve katlar arasındaki seyir hızları birer parametre şeklinde simülatör programına eklenmiştir.

Enerji tüketim hesabı ise GAS'taki tüm kabinlerin çağrıları cevaplamak için harcadığı toplam enerjiyi göstermektedir. Enerji tüketimin hesabı için tek kabinli 6 kişilik bir asansör, tam yükle katlar arası farklı senaryolarla hareket ettirilip tüketim değerleri kaydedilmiştir. Çizelge 4.4'te asansöre ait tüketim değerleri verilmektedir.

Çizelge 4.4. 6 Kişilik bir asansör içi enerji tüketimi ve seyir süresi.

Başlangıç Katı	Bitiş Katı	Süre (s)	Enerji Tüketimi (Wh)
0	1	19.66	5
0	2	22.04	9
0	3	24.42	13
0	4	26.80	16
0	5	29.18	20
0 - 5 Sıralı hareket		98.30	29

Kaydedilen değerler ile tek kabinin enerji tüketim değerinin hesabı için gerekli olan "seyir tüketim değeri" ve "yavaşlama-hızlanma tüketim değeri" hesaplanarak

simülatör programında Eşitlik 4.1'de verilen fonksiyon ile kullanılmaktadır.

$$W_b(\Delta_i, n) = \sum_{i=0}^{n} \begin{cases} 0.5, & \Delta_i = 0 \\ 5, & \Delta_i = 1 \\ 5 + 3.85\Delta_i, & \Delta_i > 1 \end{cases} \quad (4.1)$$

Eşitlik 4.1'de verilen W_b bireyin (yönlendirme tahmininin) enerji tüketim değeri, Δ_i kabin konumu ile çağrı katı arasındaki mesafe ve n çağrı adedidir.

Cevaplama sürelerinin hesabı da benzer şekilde parçalı fonksiyon ile yapılmaktadır. Hesaplama için kullanılan fonksiyon Eşitlik 4.2'de verilmektedir.

$$C_i(\Delta_i) = \begin{cases} t_{ybi} + (t_{ka} + t_{kk}), & \Delta_i = 0 \\ t_{ybi} + t_{kd} + (t_{ka} + t_{kk}), & \Delta_i \leq 1 \\ t_{ybi} + \dfrac{(\Delta_i - 1)}{V_{sh}} + t_{kd} + (t_{ka} + t_{kk}), & \Delta_i > 1 \end{cases} \quad (4.2)$$

Eşitlik 5.2'de C_i i indisli çağrının cevaplanma süresi, Δ_i i indisli kabin konumu ile hedef kat arasındaki kat farkı, t_{ybi} yolcunun kabine binme-inme süresi, t_{ka} kabin kapı açılma süresi, t_{kk} kabin kapı kapanma süresi, t_{kd} kabin kalma-durma süresi ve V_{sh} kabin seyir hızıdır. Çizelge 4.5'te denklemlerde kullanılan parametrelerin değerleri verilmektedir.

Çizelge 4.5. Hesaplarda kullanılan parametreler ve değerleri.

Parametre	**Değer**
t_{ybi}	6 s
t_{ka}	3 s
t_{kk}	3 s
t_{kd}	7.66 s
V_{sh}	0.42 kat/s
Δ_i	0 – 14 kat
n	1 – 30

BÖLÜM 5

GENETİK ALGORİTMA İLE KABİN YÖNLENDİRME SİSTEMİNİN GERÇEKLEŞTİRİLMESİ

Gelişen işlemci teknolojisiyle birlikte gömülü donanımlar çok daha yüksek işlem ve hafıza kapasitelerine ulaşmışlardır. Bu gelişme özellikle yapay zekâ tabanlı kontrol algoritmalarının ihtiyaç duyduğu işlem gücünü ve hafıza alanını karşılayarak GAS'ın kontrolünde yapay zekanın kullanımını artırmıştır.

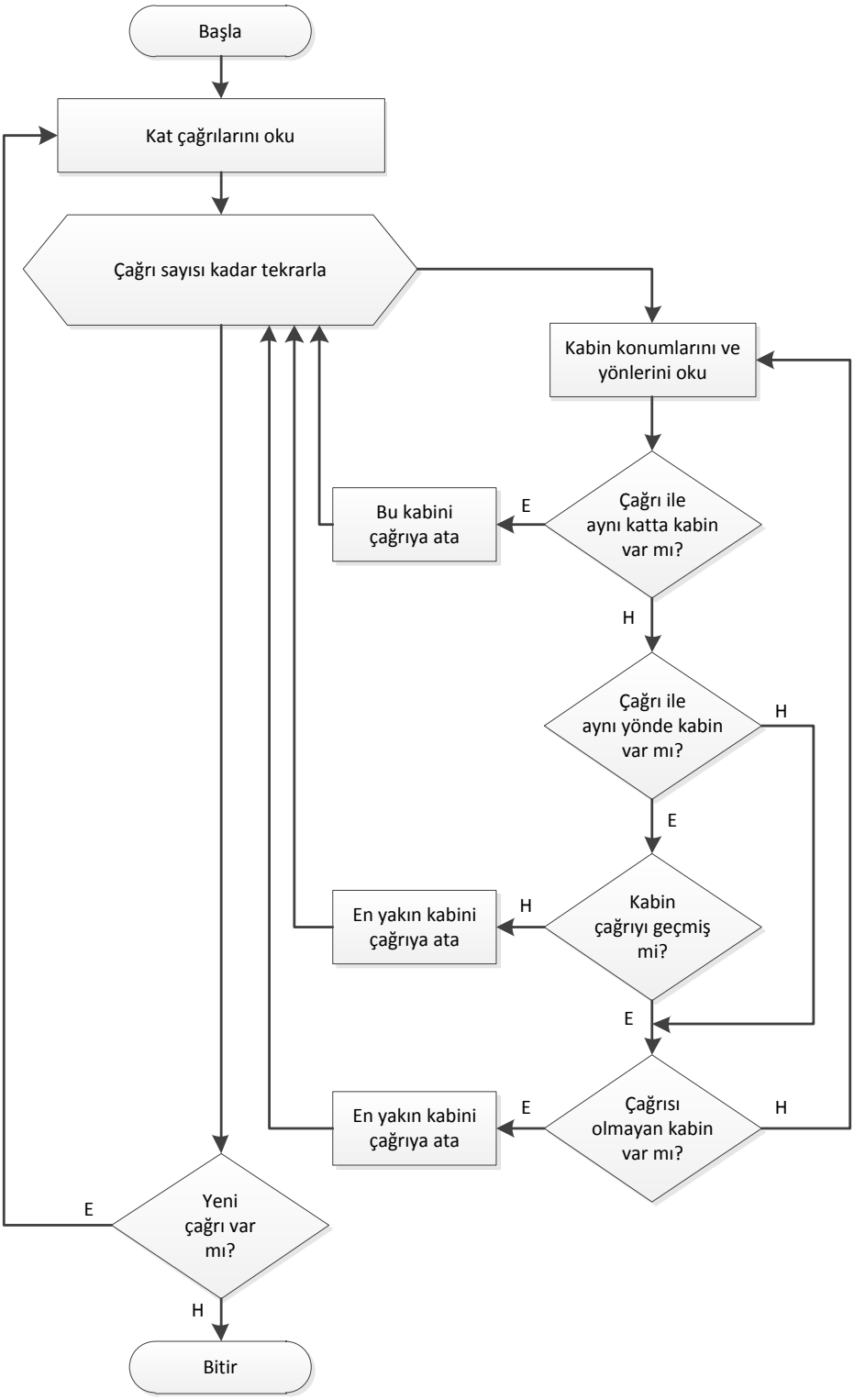
Bu bölümde GAS için geleneksel kontrol algoritması ve geliştirilen kontrol algoritması anlatılmaktadır. Geliştirilen algoritmanın uygunluğunun ölçülmesi için aynı çağrı setleri için her iki algoritma test edilmiştir. Son olarak kullanılan geliştirme kartının uygunluğu test edilerek sonuçlar grafikler şeklinde verilmektedir.

5.1. GELENEKSEL GAS KONTROL ALGROTİRMASI

Asansör sistemlerinde kabinlerin hareketleri, çağrıların cevaplama sırasına bağlı aşağı yönlü hareket, yukarı yönlü hareket ve yukarı-aşağı yönlü hareket şeklinde üç farklı başlık altında ele alınmaktadır.

Geleneksel kontrol algoritmasında kabinlerin hareketi tek yönlü seçilerek, yolcunun gitmek istediği istikametin aksine çıkılmadan en kısa sürede hedef kata götürülmesi amaçlanmaktadır [25]. Çift yönlü kabin yönlendirme hareketlerini içerisinde bulunduran algoritmalarda ise enerji tüketiminde iyileştirmeye gidilirken bekleme süresinde gecikmelere neden olmaktadır. Bu çalışma için en çok tercih edilen tek yönlü toplamalı tip ile kontrol edilen grup kontrol algoritması tercih edilmiştir. Algoritma aynı yöndeki çağrıları toplayarak kabinin son çağrıya kadar tek yönlü hareketini sağlar ve yolcunun aksi yönde taşınmasını engeller. Bu algoritmada çağrı ile aynı yöne giden kabin varsa ve kabin çağrıyı geçmemiş ise çağrı için yeni bir kabin yönlendirilmez.

Eğer uygun yönde kabin yoksa çağrı kabinlerden birinin tüm çağrılarını cevaplamasını bekler. Şekil 5.1'de geleneksel GAS kontrol sistemine ait akış diyagramı verilmektedir.

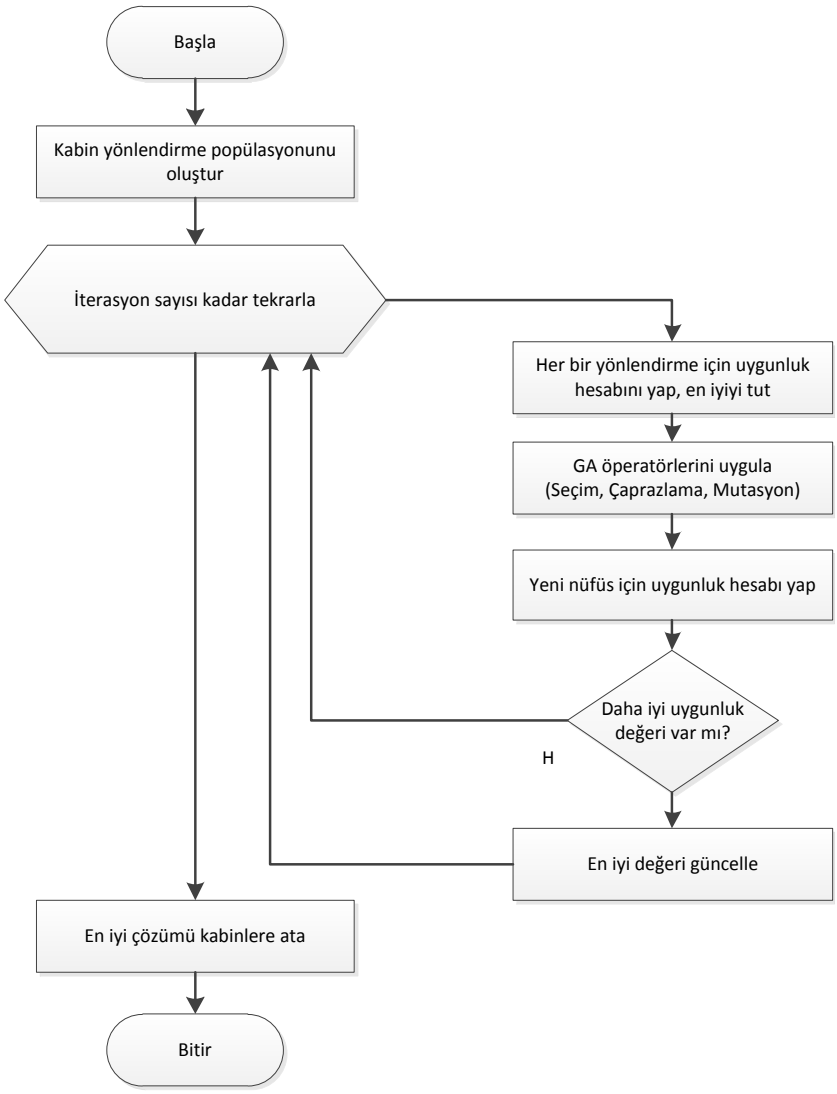


Şekil 5.1. Geleneksel GAS kontrolü akış diyagramı.

5.2. GELİŞTİRİLEN GAS KONTROL ALGROTİRMASI

Bu çalışmada, grup asansör sistemlerinde kabin yönlendirme işlemi için GA kullanılmaktadır. Geliştirilen algoritma Python dilinde gömülü sisteme uyarlanarak kodlanmıştır. Algoritmada çağrıların en kısa sürede cevaplandırılarak bekleme süresinin mimize edilmesi hedeflenmektedir.

Kabin yönlendirme algoritmasında ilk olarak kabin yönlendirme popülasyonu oluşturulmaktadır. Bu popülasyondaki her bir birey GAS'taki çağrılar için kabin yönlendirmelerini ifade etmektedir. Algoritmanın Akış diyagramı Şekil 5.2'de verilmektedir.



Şekil 5.2. Kabin yönlendirme algoritması akış diyagramı.

Algoritma için seçilen iterasyon sayısı kadar GA tabanlı yönlendirme işlemi tekrar edilmektedir. GA için ilk olarak popülasyondaki her bir bireyin uygunluk değeri hesaplanır. Hesaplama ardından farklı yönlendirmeler için popülasyonun çeşitlendirilmesi adına GA'nın operatörleri olan "Seçim", "Çaprazlama" ve "Mutasyon" işlemleri uygulanır.

Yeni popülasyon içerisindeki her bir yeni çözümün uygunluk değeri tek tek hesaplanarak en iyi sonuçla karşılaştırılır. Hesaplanan sonucun mevcut sonuçtan daha iyi çıkması durumunda mevcut değer yeni değer ile güncellenir. Tüm bu işlemler iterasyon sayısı kadar tekrar ederek en iyi sonuç bulunmaya çalışılır. Bulunan en iyi yönlendirme senaryosu, simülasyon programında işletilerek sistemdeki tüm çağrılar cevaplandırılır.

5.2.1. Çağrı Matrisinin Oluşturulması

Yapay zekâ tabanlı optimizasyonda kodlamadaki kolaylıklarından dolayı veriler matrislere atanır. Çalışmadaki çağrı matrisi GAS'taki tüm çağrılardan oluşmaktadır. Çağrı matrisi her bir çağrı için [çağrı katı, çağrı yönü] şeklinde boyutlandırılmıştır. Çağrı yönündeki "1" yukarı yönlü isteği ve "-1" ise aşağı yönü isteği ifade etmektedir. Çizelge 5.1'de örnek bir çağrı matrisi verilmektedir. GAS'ta eklenen her çağrı bu şekilde çağrı matrisi içerisine tanımlanarak yönlendirme algoritmasında kullanılmaktır.

Çizelge 5.1. GAS'a ait çağrı matrisi.

Çağrı matrisi	[1, -1]	[2, 1]	[3, 1]	[4, -1]	[6, 1]	[7, 1]

5.2.2. Başlangıç Popülasyonunun Oluşturulması

GA'da popülasyon, çözümleri ifade eden bireylerin oluşturduğu topluluğa verilen isimdir. Popülasyon her yönlendirme işlemi öncesi seçilen büyüklükte yeniden oluşturulur. Popülasyonun büyük olması yönlendirme kalitesini olumlu yönde etkilerden işlemcinin işlem yükünü artırdığı için çağrıların cevaplanmasını geciktirmektedir. Bu nedenle popülasyon büyüklüğü donanımın gücü ve ihmal

edilebilecek gecikme zamanı göz önünde bulundurularak seçilmelidir.

Bu çalışmada kullanılan bireylerin sütün sayısı ile çağrı matrisinin sütun sayısı aynıdır. Çizelge 5.2'de örnek çağrı ve birey verilmektedir. Birey içerisindeki sayılar çağrıya yönlendirilen kabini ifade etmektedir.

Çizelge 5.2. Grup asansör sistemi çağrı matrisi ve popülasyon bireyi.

Çağrı matrisi	[14, -1]	[9, -1]	[3, -1]	[0, 1]	[1, 1]	[3, 1]
Birey	4	4	4	2	3	3

5.2.3. Uygunluk Fonksiyonu ve Uygunluk Değeri Hesabı

Uygunluk fonksiyonu GA'da kilit unsurlardan biridir. Uygunluk fonksiyonunun doğru belirlenmesi kabin yönlendirme işleminin başarısını ciddi ölçüde etkilemektedir. Yönlendirmenin GA tabanlı oluşu nedeniyle uygunluk fonksiyonunun sistemin matematiksel modeline yakın olması yeterlidir. Ancak uygunluk fonksiyonu seçilirken sistemin detayları iyi belirlenmeli ve ihmallerin en aza indirgendiği bir modelleme işlemi yapılmalıdır. Yanlış seçilmiş bir uygunluk fonksiyonu, yönlendirme sonucunu tümüyle etkiler ve işlemini anlamsız hale getirir.

Bu çalışmada uygunluk fonksiyonu sistemdeki tüm çağrıların cevaplanma sürelerinin geometrik ortalaması olarak seçilmektedir. Bu sayede bir çağrıdaki meydana gelen en küçük değişim geometrik ortalamaya yansıyarak gerçek çözüme ulaşılmaktadır. Seçilen uygunluk fonksiyonu Eşitlik 5.1'de verilmektedir.

$$U(t) = \sqrt[n]{\prod_{i=0}^{n} t_i} \tag{5.1}$$

Eşitlik 5.1'de U uygunluk değeri, n çağrı matrisinde yer alan çağrı sayısı ve t_i her çağrının cevaplanma süresidir.

5.2.4. Seçim İşlemi

Seçim işlemi GA'da, bir sonraki popülasyonun oluşumundaki çeşitliliği etkileyen en önemli basamaktır. Seçim işlemi ile popülasyondaki zayıf bireyler elenerek güçlü bireylerin devam etmesi sağlanmaktadır. Bu çalışmada, her iterasyonda popülasyonun uygunluk değerinin aritmetik ortalaması alınarak ortalamanın altında kalan bireyler popülasyondan çıkarılmaktadır. Popülasyonun her iterasyonda aynı sayıda bireyden oluşması ve çeşitliliğin artırılması için çıkarılan her bireyin yerini rastgele üretilen yeni bireyler almaktadır.

5.2.5. Çaprazlama İşleminin Uygulanması

Çaprazlama işlemi genetik algoritmadaki çeşitliği artıran önemli adımlardan biridir. Çaprazlama işlemi sonrasında iki bireydeki iyi sonuçların birleşerek daha iyi sonuçların bulunması hedeflenir.

Bu çalışmada her iterasyonda popülasyondaki tüm bireyler için rastgele çaprazlama değeri atanmaktadır. Atanan çaprazlama değerleri seçilen çaprazlama oranı ile tek tek karşılaştırılmaktadır. Çaprazlama değerinin altında kalan bireyler çaprazlama işlemi için bir havuza aktarılmaktadır. Havuz içerisindeki birey sayısının çift olup olmadığı kontrol edilerek fazla olan birey çaprazlama havuzundan çıkarılır. Bu işlem ardından çiftler sıra ile eşleştirilip çaprazlama işlemine tabi tutulmaktadır. Çizelge 5.3'te çaprazlama işlemine tabi tutulan iki birey verilerek ve seçilen çaprazlama noktası (ÇN) koyu renkle vurgulanmaktadır.

Çizelge 5.3. Çaprazlama işleminin bireylere uygulanması.

Bireyler			**ÇN**			
Çaprazlama öncesi 1. birey	0	3	3	0	4	1
Çaprazlama öncesi 2. birey	1	1	0	3	0	4

İki bire bu noktadan kırılarak bu noktadan itibaren sağda kalan parçalar bireyler

arasında yer değiştirecek ve çaprazlama işlemi yapılacaktır. Geçekleştirilen tek noktadan çaprazlama işlemi sonucunda oluşan yeni bireyler Çizelge 5.4'te verilmektedir.

Çizelge 5.4. Çaprazlama işleminin bireylere uygulanması.

Bireyler			**ÇN**			
Çaprazlama sonrası 1. birey	0	3	0	3	0	4
Çaprazlama sonrası 2. birey	0	3	3	0	4	1

5.2.6. Mutasyon İşleminin Uygulanması

Mutasyon işlemi GA'daki çeşitliliği sağlayan diğer bir adımdır. Bu adımda noktasal dokunuşlarla her bireydeki kusurlu olan genin düzeltilmesi ve daha iyi sonucun bulunması hedeflenmektedir.

Bu çalışmada her iterasyonda popülasyondaki tüm bireyler için rastgele mutasyon değeri atanmaktadır. Atanan mutasyon değerleri seçilen mutasyon oranı ile tek tek karşılaştırılmaktadır. Mutasyon değerinin altında kalan bireyler mutasyon işlemi için bir havuza aktarılmaktadır. Havuz içerisindeki bireylerde mutasyona tabi tutulacak genler rasgele belirlenip mutasyon işlemi gerçekleştirilmektedir. Mutasyon işlemi ile çağrıya atanan asansör değiştirilmektedir. Mutasyona tabi tutulan birey ve mutasyon sonrası değişen iki gen koyu renkle vurgulanarak Çizelge 5.5'te verilmektedir.

Çizelge 5.5. Mutasyon işleminin bireye uygulanması.

Mutasyon İşlemi						
Mutasyon öncesi birey	0	3	3	2	4	1
Mutasyon sonrası birey	0	3	3	1	4	0

5.2.7. Normalizasyon İşleminin Uygulanması

Asansör sistemlerinde yapılan optimizasyon ile yolcunun en kısa süre içerisinde varmak istediği kata götürülmesi amaçlanmaktadır. Yolcunun gitmek istediği yönün aksi olan her kabin hareketi, bekleme süresini uzatmaktadır. Yön denetiminin olmadığı sistemlerde, sisteme dahil olan her yeni çağrı kabin içerisinden seçilen çağrıların gölgelenerek yolcunun kabinde uzun süre kalmasına veya gideceği kata asla varamamasına sebep olabilmektedir.

Bu çalışmada kabin yön denetimi normalizasyon başlığı altında ele alınarak ve çağrı yönünün aksine olan kabin hareketlerinde düzenleme işlemi gerçekleştirilmektedir. Normalizasyon işlemi GA'nın öperatörleri sonrasında doğru sonuçların çıkışa aktarılması için yapılması gereken önemli bir işlemdir. Her jenerasyonda yapılan çaprazlama, mutasyon ve yeniden birey oluşturma işlemleri ardından istenilen yön denetimin algoritmasının dışında sonuçlar üretilebilmektedir. Normalizasyon işlemi ile her bir bireyin sunduğu yönlendirme hareketi için yön denetimi yapılmakta ve aksi istikametin söz konusu olması durumunda uygun asansör ile değiştirme işlemi gerçekleştirilmektedir. Çizelge 5.6'da bireye uygulanan normalizasyonun örneği verilmektedir.

Çizelge 5.6. Normalizasyon işleminin bireye uygulanması.

Kat çağrıları ve yönleri (1 : yukarı, -1 : aşağı)	[1, -1]	[2, 1]	[3, 1]	[7, -1]	[10, 1]	[11, 1]
Normalizasyon işlemi öncesi birey	3	4	3	3	1	1
Normalizasyon işlemi sonrası birey	3	4	4	0	1	1

Çizelge 5.6'da normalizasyon işlemi öncesinde zemin katta bekleyen 3 numaralı kabin sırasıyla [1,-1], [3,1] [7,-1] ve [11,1] çağrılarına atanmıştır. Bu atama şekline baktığımızda kabin öncelikle birinci kattan aşağı gitmek isteyen yolcuyu alacak ve üçüncü kata çıkaracaktır. Yanlış yapılan bu yönlendirme ile ilk yolcu gereksiz yere kabin içerisinde bekletilecektir. Benzer şekilde ilk çağrı ile aynı yönde olmasına rağmen yedinci kattaki yönlendirme işlemi de ilk yolcunun gereksiz yere 6 kat

çıkmasına sebep olacağı için normalizasyon işlemi ile değiştirilmektedir. Normalizasyon işlemi sonrasında kabin hareketlerine baktığımızda her kabin almış olduğu yolcuyu gitmek istediği istikamet doğrultusunda taşımaktadır.

GA'nın tüm basamakları sabit 10 çağrılı tek bir senaryo için uygulandığında başlangıç ve bitiş popülasyonu Çizelge 5.7'de verilmektedir.

Çizelge 5.7. GA öncesi ve sonrasında popülasyon.

Sıra	GA Öncesi Bireyler		GA Sonrası Bireyler	
	Birey	**Uygunluk (s)**	**Birey**	**Uygunluk(s)**
1	[3, 3, 2, 0, 0, 1, 1, 0, 4, 1]	44,95346228	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
2	[3, 4, 2, 0, 0, 1, 1, 1, 1, 0]	45,22398119	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
3	[4, 2, 4, 1, 0, 3, 3, 0, 3, 0]	38,65073262	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
4	[2, 3, 3, 1, 0, 4, 4, 0, 0, 1]	50,56986994	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
5	[2, 2, 0, 1, 1, 3, 3, 1, 4, 4]	43,27348322	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
6	[3, 2, 3, 4, 0, 0, 1, 1, 0, 1]	47,43430468	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
7	[3, 3, 3, 0, 1, 0, 4, 1, 4, 2]	51,43762280	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
8	[1, 2, 2, 4, 0, 4, 3, 0, 3, 4]	49,04042529	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
9	[1, 4, 2, 3, 3, 0, 0, 3, 0, 0]	38,89519519	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
10	[3, 4, 3, 2, 0, 1, 1, 2, 0, 0]	40,62711095	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
11	[2, 3, 1, 0, 0, 4, 4, 0, 4, 0]	52,20974942	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
12	[2, 3, 1, 4, 0, 0, 0, 0, 0, 4]	54,27897530	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
13	[2, 0, 0, 3, 1, 4, 4, 3, 4, 4]	48,61317644	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
14	[2, 4, 3, 1, 0, 0, 0, 0, 1, 0]	48,89678855	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
15	[0, 0, 2, 3, 1, 4, 1, 3, 1, 4]	45,01913355	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
16	[2, 1, 0, 3, 4, 4, 3, 3, 4, 3]	47,05883143	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
17	[3, 2, 2, 0, 1, 4, 0, 4, 4, 0]	52,79903369	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
18	[4, 4, 0, 1, 3, 3, 2, 3, 1, 2]	31,92602706	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
19	[2, 0, 0, 3, 1, 4, 1, 4, 4, 4]	49,42806639	[4, 4, 1, 2, 3, 2, 0, 0, 0, 3]	24,41456093
20	[2, 0, 2, 4, 3, 1, 1, 4, 4, 1]	48,35450159	[4, 4, 1, 2, 3, 2, 0, 2, 0, 3]	25,07018472
21	[4, 4, 2, 3, 1, 0, 0, 0, 0, 1]	35,16450533	[4, 4, 1, 2, 3, 2, 0, 2, 3, 0]	25,15702027
22	[3, 3, 2, 4, 0, 4, 0, 1, 1, 0]	45,09063356	[4, 4, 1, 2, 3, 2, 0, 0, 0, 0]	25,37620221
23	[2, 2, 0, 1, 3, 4, 4, 1, 4, 4]	48,33269828	[4, 4, 1, 2, 3, 2, 2, 0, 3, 0]	26,62767261
24	[1, 2, 4, 0, 3, 3, 3, 0, 3, 0]	41,92936228	[4, 4, 1, 2, 3, 0, 2, 0, 3, 0]	27,83908093
25	[2, 2, 3, 0, 4, 0, 1, 1, 0, 0]	47,19266767	[4, 4, 0, 2, 3, 2, 1, 3, 3, 3]	27,91542695
26	[1, 2, 3, 0, 0, 4, 4, 0, 4, 0]	49,50698114	[4, 4, 1, 2, 2, 2, 0, 0, 2, 3]	28,46853306
27	[1, 2, 3, 4, 0, 0, 0, 0, 4, 0]	51,37937358	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
28	[1, 1, 4, 2, 0, 0, 3, 0, 3, 3]	40,24135389	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
29	[3, 3, 2, 0, 0, 1, 0, 0, 4, 4]	46,20903425	[4, 4, 4, 2, 3, 2, 0, 0, 1, 1]	23,94965932
30	[0, 1, 0, 3, 2, 3, 2, 4, 4, 3]	39,59142210	[4, 4, 1, 2, 3, 2, 0, 0, 3, 0]	24,36670224
	Ortalama	45,77761679	**Ortalama**	24,78073461

Bu tek senaryo için çizelge incelendiğinde 30 popülasyon büyüklüğüne sahip bir popülasyonun GA öncesi uygunluk ortalaması 45,77761679 iken GA'ya ait tüm işlemlerin tamamlanması ile birlikte bu ortalama en iyi uygunluk değeri olan 23,78829113'e yaklaşık %4'lük bir farkla yaklaşmıştır.

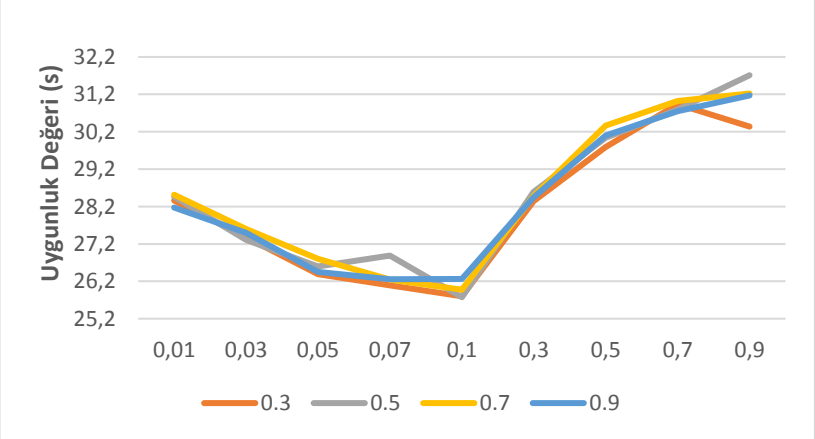
BÖLÜM 6

DENEYSEL ÇALIŞMALAR VE TARTIŞMA

6.1. GA İÇİN ÇAPRAZLAMA VE MUTASYON ORANININ SEÇİLMESİ

GA Algoritmada çaprazlama ve mutasyon oranlarının doğru seçilmesi yapılacak olan minimize işleminin kalitesini doğrudan etkilemektedir. Yüksel seçilen çaprazlama oranı erken yakınsamaya neden olacağı için doğru sonucu vermeyecektir. Benzer şekilde mutasyon oranının yüksek seçilmesi durumunda da popülasyondaki bireylerin aşırı mutasyona tabi kalarak en iyi çözümlerin kaybedilmesine sebep olacaktır [23,29,32].

GAS'ta kabin yönlendirme probleminde çaprazlama ve mutasyon oranının seçilmesi için literatürde en çok tercih edilen 4 farklı çaprazlama ve 9 farklı mutasyon oranı 10 çağrılı sabit bir senaryo için denenmiştir. Tüm testlerde başlangıç popülasyonu sabit seçilerek her test bitiminde en iyi sonuç kaydedilmiştir. Testlerde 50 popülasyon büyüklüğü ve 500 iterasyon sayısı için her oran ikilisi 20 kez çalıştırılıp aritmetik ortalamaları alınarak kaydedilmiştir. Test sonuçlarına ait grafikler Şekil 6.1'de ve veriler de Çizelge 6.1'de verilmektedir.



Şekil 6.1. GA parametre test sonuçları.

Çizelge 6.1. Çaprazlama ve mutasyon oranına bağlı uygunluk değerleri (s).

		Çaprazlama Oranları			
		0.3	**0.5**	**0.7**	**0.9**
Mutasyon Oranları	**0,01**	28,36406270	28,46291461	28,51235374	28,17614395
	0,03	27,37990691	27,30635952	27,59668649	27,49369134
	0,05	26,39551060	26,60121099	26,80187197	26,44953485
	0,07	26,09332128	26,88608998	26,24968598	26,25519325
	0,1	25,79984930	**25,77408784**	25,97567188	26,26675697
	0,3	28,34027786	28,59565765	28,48195157	28,44665443
	0,5	29,78964431	30,03563504	30,36778587	30,09491854
	0,7	30,93650915	30,78882703	31,02633764	30,74903726
	0,9	30,34181484	31,71119939	31,22129043	31,17021955

Verilen çizelge incelendiğinde en iyi çaprazlama oranı olarak 0.5 ve en iyi mutasyon oranı olarak 0.1 seçilerek diğer testler bu oranlar ile gerçekleştirilmektedir

6.2. ALGORİTMANIN TEST SONUÇLARI

Geliştirilen algoritmanın testleri grup asansör sistemi için Python dili kullanılarak hazırlanan simülatör yazılımında yapılmaktadır. Simülatör yazılımı, Beaglebone Black üzerine kurulu Linux içeresinde çalıştırılabiliyor olmasına rağmen gözlem, kontrol ve kayıt altına alma kolaylıkları nedeniyle Windows işletim sisteminde kurulu IDLE (Python GUI) programında çalıştırılmaktadır.

Algoritmanın testleri 3 farklı senaryo tipi kullanılarak gerçekleştirilmiştir. Bunlar, 10 tane 10 çağrı sayılı senaryo ve 10 tane 20 çağrı sayılı senaryo ve tane 4 çağrıdan 30 çağrıya kadar artan çağrı sayılı senaryo şeklindedir. Testler için kullanılan senaryolar Ek açıklamalar A'da sırası ile Çizelge Ek A.1, Çizelge Ek A.2, Çizelge Ek A.3 ve Çizelge Ek A.4'te verilmektedir.

Yapılan testler 16 katlı ve içerisinde 5 asansörü bulunduran bir bina modelin için yapılmaktadır. Bu testlerle geleneksel kontrol ile GA tabanlı kontrol arasındaki fark görülmek istenmektedir. Test için kullanılan her senaryo öncesinde GAS'ta yer alan

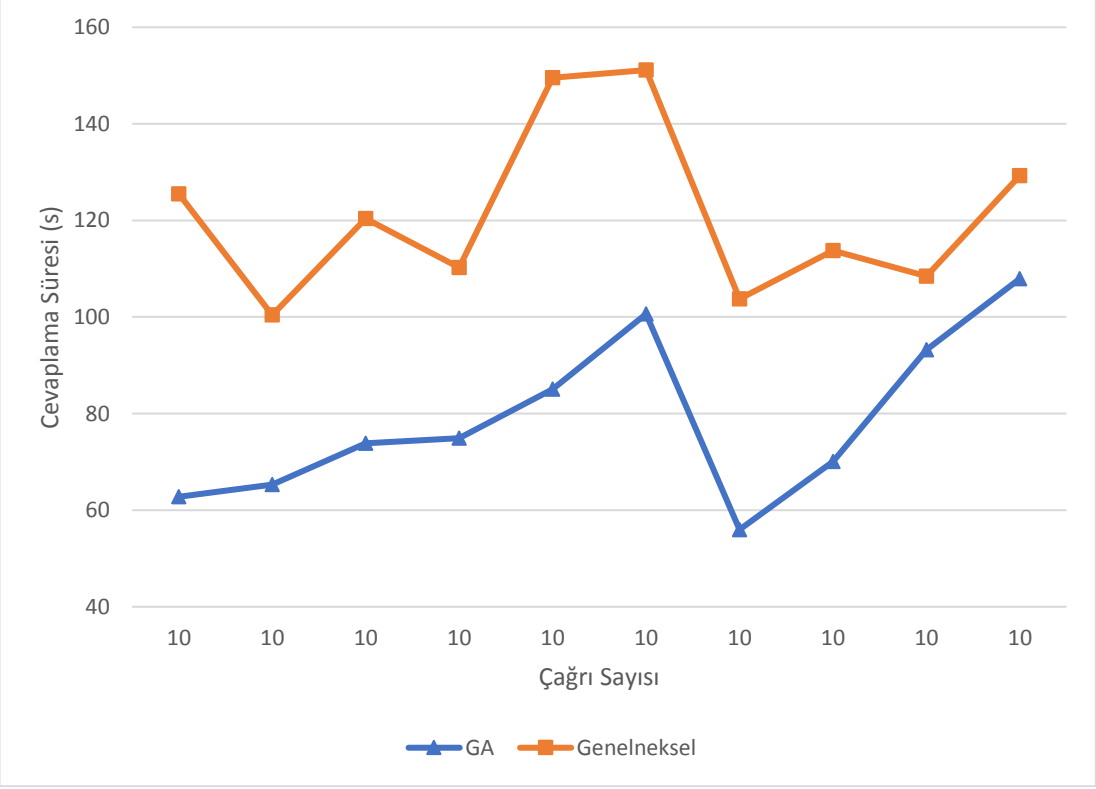
kabinlerin başlangıç konumları Çizelge 6.2'de verildiği gibi sabit olarak seçilmiştir.

Çizelge 6.2. Kabinlerin başlangıç konumları.

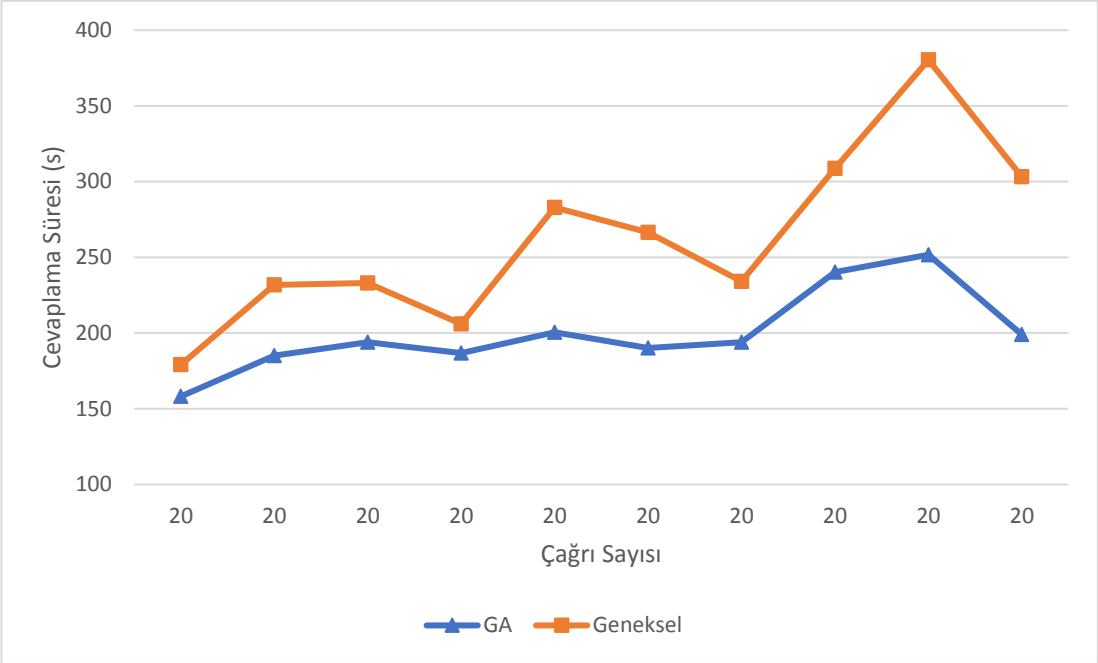
Kabin Başlangıç Konumları (Kat)				
1. Kabin	**2. Kabin**	**3. Kabin**	**4. Kabin**	**5. Kabin**
1	3	8	0	5

6.2.1. Çağrı Cevaplama Süresi Sonuçları

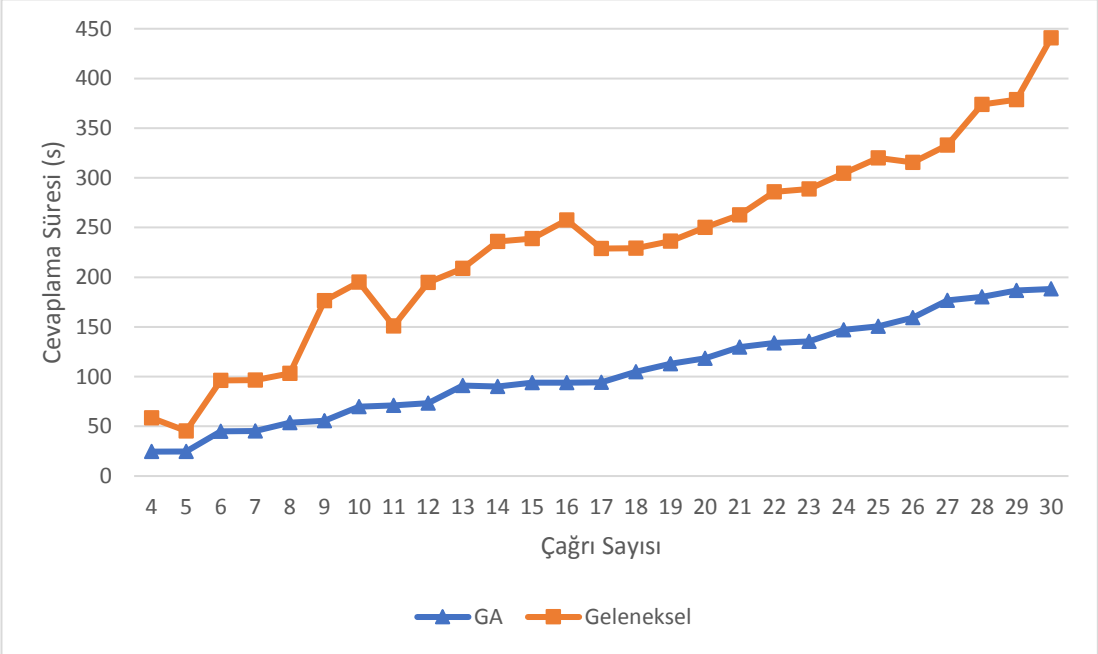
3 farklı senaryo tipi her iki algoritma içinde ayrı ayrı çalıştırılmış ve iki algoritmanın senaryoları cevaplama sürelerine ait sonuçların grafikleri Şekil 6.2 (Sabit 10 çağrılı senaryolar.), Şekil 6.3 (Sabit 20 çağrılı senaryolar.) ve Şekil 6.4'te (Artan çağrılı senaryolar.) verilmektedir. Testlere ait ayrıntılı sonuçlar ise Ek açıklamalar B'de sırası ile Çizelge Ek B.1, Çizelge Ek B.2, Çizelge Ek B.3 ve Çizelge Ek B.4'te verilmektedir.



Şekil 6.2. Algoritmaların sabit 10 çağrılı senaryoları cevaplama süreleri.



Şekil 6.3. Algoritmaların sabit 20 çağrılı senaryoları cevaplama süreleri.



Şekil 6.4. Algoritmaların artan çağrılı senaryoları cevaplama süreleri.

Verilen grafikler incelendiğinde GA kullanılarak yapılan kabin yönlendirme işleminin geleneksel yöntemden belirgin bir farkla daha iyi sonuç verdiği görülmektedir. Her iki kontrol algoritmasının 3 senaryo tipi için verdiği toplam cevaplama sürelerine bakıldığında 5 çağrılı senaryolardan sonra GA'nın belirgin bir farkla daha iyi sonuç

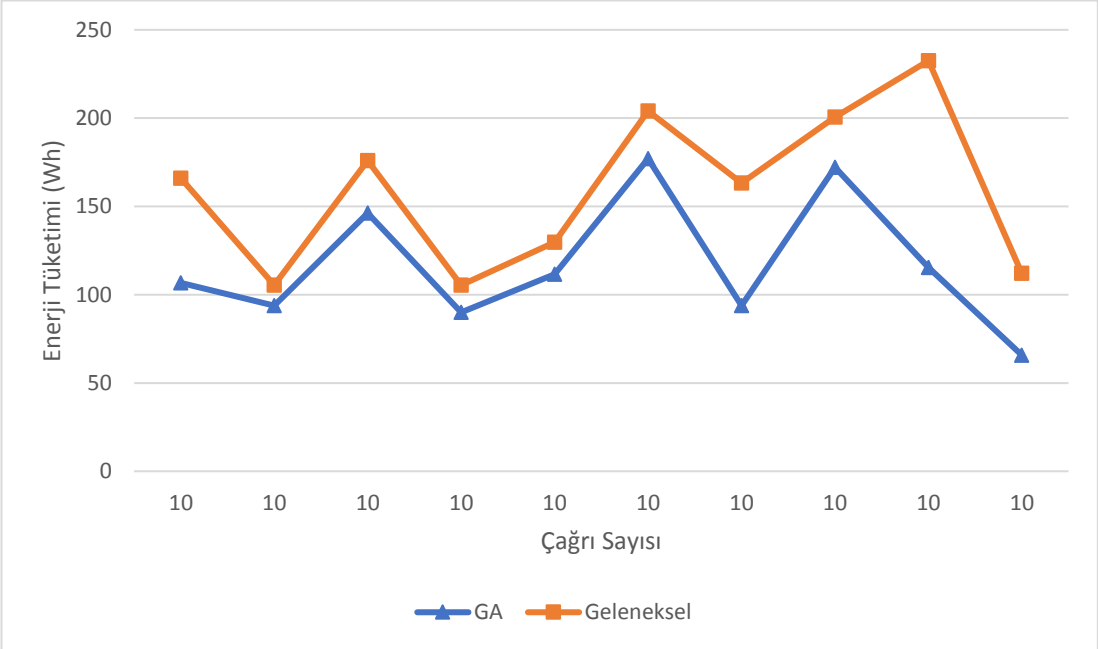
verdiği görülmektedir. Çizelge 6.3'te 3 farklı senaryo tipindeki tüm alt senaryoların cevaplama süreleri toplanarak GA ile gerçekleşen yönlendirme işleminin iyileştirme oranları verilmektedir.

Çizelge 6.3. Algoritmaların toplam cevaplama süreleri ve iyileşme oranları.

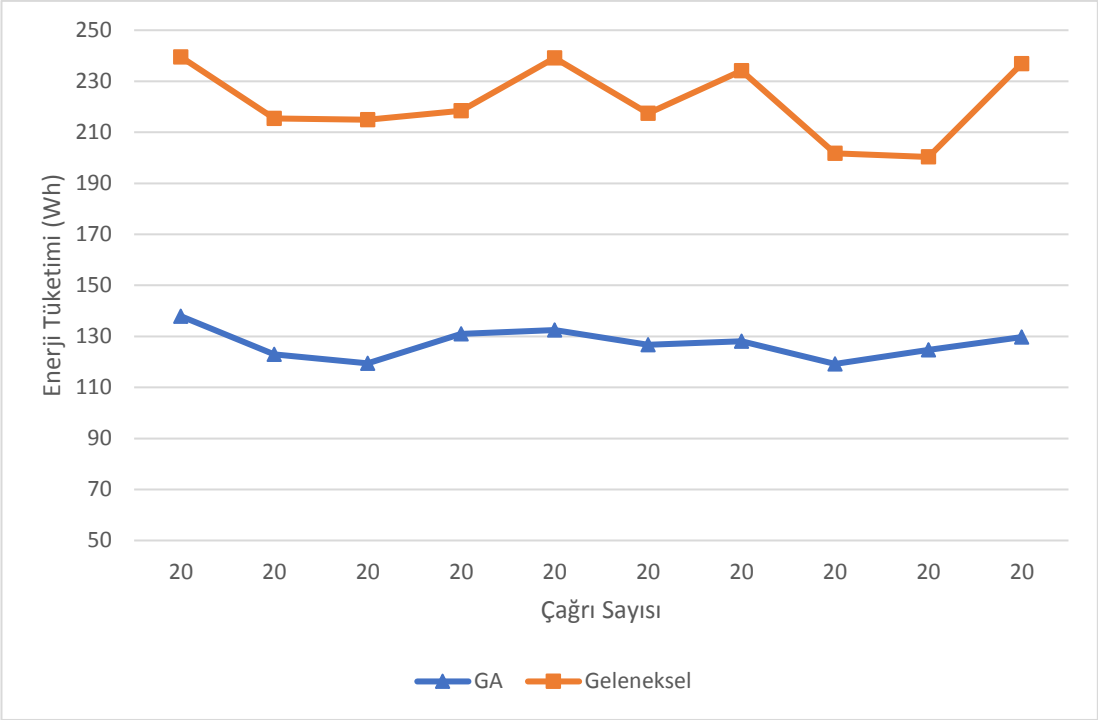
	Genetik Algoritma Toplam Süre (s)	**Geleneksel Yöntem Toplam Süre (s)**	**İyileştirme Oranı (%)**
10 Çağrılı Senaryolar	789,720	1212,35	34,85
20 Çağrılı Senaryolar	1271,98	2217,75	42,64
Artan Çağrılı Senaryolar	2850,11	3456,02	17,53
Toplam	4911,81	6886,12	28,67

6.2.2. Enerji Tüketim Sonuçları

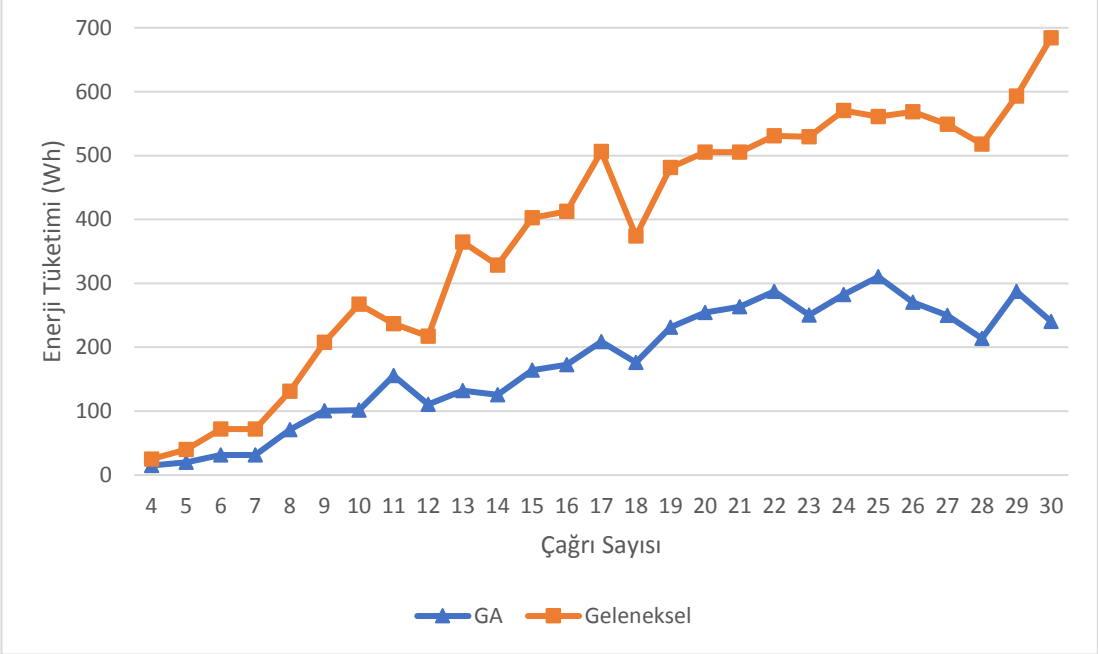
3 farklı senaryo tipi her iki algoritma içinde ayrı ayrı çalıştırılmıştır. İki algoritmanın senaryoları cevaplamak için harcadığı enerjiler grafiksel olarak Şekil 6.5, Şekil 6.6 ve Şekil 6.7'de verilmektedir. Testlere ait ayrıntılı sonuçlar ise Ek açıklamalar B'de sırası ile Çizelge Ek B 1, Çizelge Ek B 2, Çizelge Ek B 3 ve Çizelge Ek B 4'te verilmektedir.



Şekil 6.5. Algoritmaların sabit 10 çağrılı senaryoları için enerji tüketim değerleri.



Şekil 6.6. Algoritmaların sabit 20 çağrılı senaryoları için enerji tüketim değerleri.



Şekil 6.7. Algoritmaların artan çağrılı senaryoları için enerji tüketim değerleri.

Verilen grafikler incelendiğinde GA kullanılarak yapılan kabin yönlendirme işleminin geleneksel yöntemden tüm senaryolar için daha iyi sonuç verdiği görülmektedir. Çizelge 6.4'te 3 farklı senaryo tipindeki tüm alt senaryolardaki enerji tüketim değerleri

toplanarak GA ile gerçekleşen yönlendirme işlemindeki iyileştirme oranları verilmektedir.

Çizelge 6.4. Algoritmaların enerji tüketim değerleri ve iyileşme oranları.

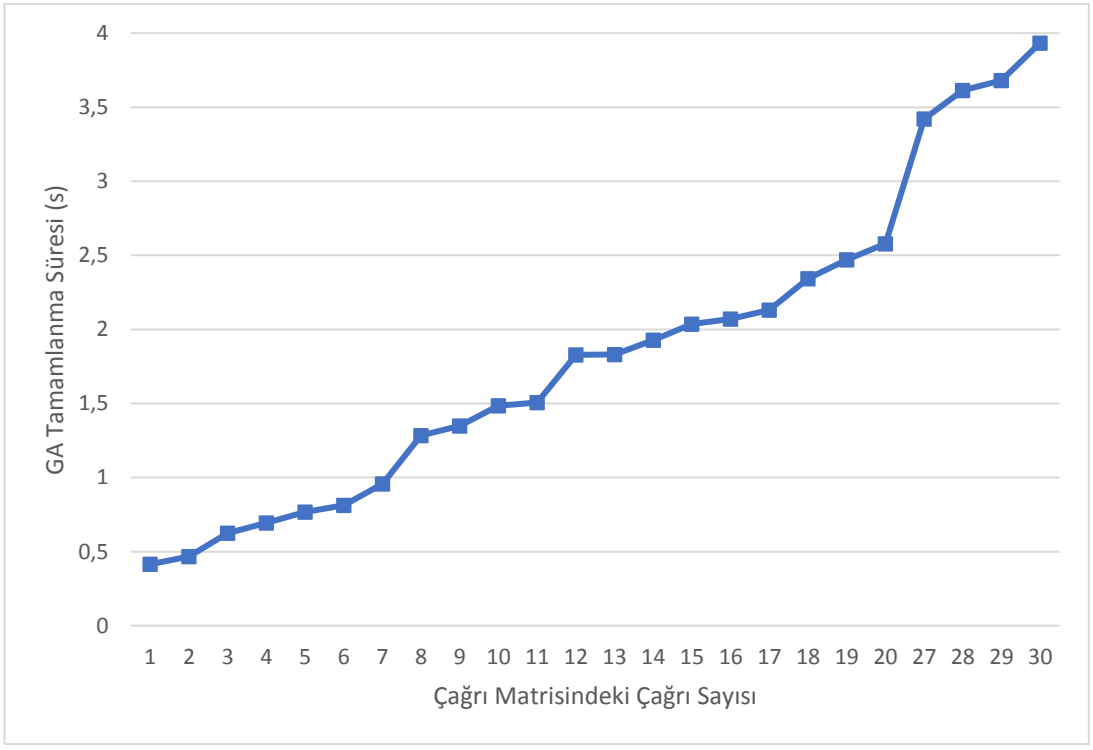
	Genetik Algoritma Toplam Enerji (Wh)	**Geleneksel Yöntem Toplam Enerji (Wh)**	**İyileştirme Oranı (%)**
10 Çağrılı Senaryolar	1172,709000	1594,799851	26,46
20 Çağrılı Senaryolar	1999,316800	2625,731678	23,85
Artan Çağrılı Senaryolar	4757,643894	5495,466000	13,42
Toplam	7929,669694	9715,998000	18,38

6.3. GÖMÜLÜ SİSTEMİN PERFORMANS TESTİ SONUÇLARI

Gömülü donanımın testleri, gömülü sistemin yönlendirme işlemini tamamlamak için harcadığı sürenin belirlenmesi şeklinde yapılmaktadır. Yönlendirme için seçilen GA'nın parametreleri arasında yer alan popülasyon büyüklüğü ve iterasyon sayısı işlemcinin işlem yükünü etkileyen en önemli parametrelerdir.

Bu iki parametrenin yüksek seçilmesi optimizasyon için olumlu etki oluştururken optimizasyonun tamamlanması için geçen sürenin artmasına sebep olmaktadır. Bu nedenle bu parametreler belirlenirken donanımın işlem kapasitesi ve tolerans edilebilecek optimizasyon tamamlama süresi göz önünde bulundurulmalıdır.

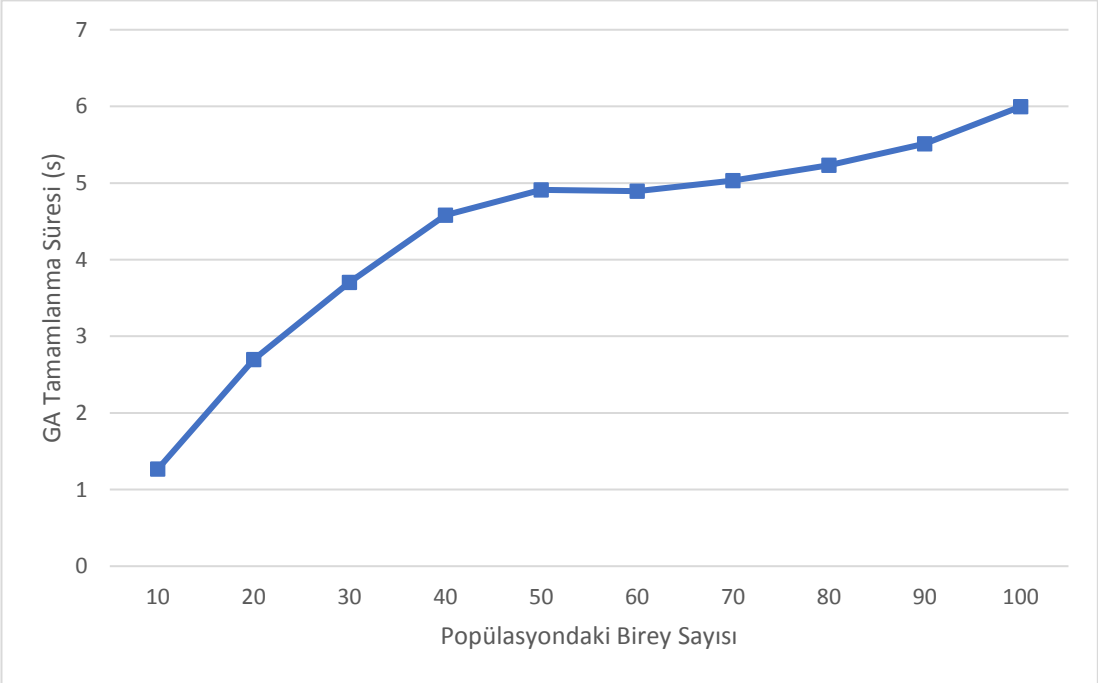
Gömülü donanımın testi çağrı sayısına ve popülasyon büyüklüğüne bağlı olarak iki farklı senaryo tipi ile gerçekleştirilmiştir. Şekil 6.8'de verilen grafikte otuz bireye ve otuz iterasyona sahip bir GA için çağrı sayısı kademeli olarak 30 çağrıya kadar artırılarak gömülü sistemin yönlendirme işlemini tamamlaması için harcadığı süreler verilmektedir.



Şekil 6.8. Gömülü sistemin çağrı sayısına göre optimizasyon için harcadığı süre.

Şekil 6.8'de görüldüğü üzere çağrı matrisinde yer alan çağrı sayısı arttıkça gömülü donanımın optimizasyonu tamamlamak için ihtiyaç duyduğu süre artmaktadır.

Şekil 6.9'da verilen grafikte ise, tezde model alınan 16 katlı bir binada oluşturulabilecek en fazla çağrı sayısı (15 aşağı- 15 yukarı yönlü çağrı) için artan popülasyon büyüklüğüne bağlı optimizasyon tamamlama süreleri gösterilmektedir.



Şekil 6.9. Gömülü sistemin popülasyona göre yönlendirme için harcadığı süre.

Şekil 6.9'da görüldüğü üzere popülasyondaki birey sayısı artarken yönlendirme atamasının tamamlanması için beklenen sürede artmaktadır.

BÖLÜM 7

SONUÇLAR VE DEĞERLENDİRME

Bu çalışmada GAS'taki kabin yönlendirme problemi ele alınarak katlardaki yolcuların bekleme sürelerini en aza indirgeyecek GA tabanlı bir yönlendirme algoritması geliştirilmiştir. Geliştirilen algoritmanın testi maliyet ve uyarlanabilirlik adına kişisel bir bilgisayar yerine mevcut asansör kontrol kartlarında kullanılabilecek nitelikte bir geliştirme kartı olan Beaglebone Black üzerinde gerçekleştirilmiştir.

Yapay zekâ tabanlı algoritmalar büyük çözüm kümeleri üzerinde çalışmalarından kaynaklı yüksek işlem gücü ve hafıza alanına ihtiyaç duymaktadır. Gömülü donanımdaki sınırlı hafıza alanı ve yeterince yüksek olmayan işlem gücü nedeniyle GA'nın kodlanması esnasında donanıma bağlı olarak algoritma şekillendirilmiş ve GAS için kabin yönlendirmesinin makul süre içerisinde tamamlanması sağlanmıştır.

Hazırlanan simülatör programında önerilen kontrol algoritması ve geleneksel kontrol algoritması farklı trafik senaryoları için test edilerek algoritmaların enerji tüketim değerleri ve çağrı cevaplama süreleri hesaplanmıştır.

Sonuçlar bekleme sürelerine göre karşılaştırıldığın önerilen kontrol algoritmasının az çağrılı senaryolarda belirgin farkla bir üstünlüğü olamamasına karşın çağrı sayısının arttığı senaryolarda üstünlüğü daha belirgin olmaktadır. Testler neticesinde GA tabanlı yönlendirmenin ortalama %28,67 daha iyi performans göstermektedir.

Sonuçlar enerji tüketim değerlerine göre karşılaştırıldığında ise önerilen algoritmanın geleneksel kontrol algoritmasına kıyasla ortalama %18,38 daha az enerji harcamaktadır.

Gömülü donanımın testinde ise modellenen yapıdaki çağrı sayısı kademeli olarak en

az sayıdan en çok sayıya kadar artırılarak yönlendirme işlemi gerçekleştirilmiş ve tamamlanma süreleri kaydedilmiştir. Testler sonucunda seçilen donanım, kabin kapısının açılıp kapanmasından daha kısa bir süre içerisinde sistemdeki maksimum çağrı sayısı için kabinlerin yönlendirme işlemini GA tabanlı olarak gerçekleştirmiştir.

Altıncı bölümde gerçekleştirilen gömülü donanım testleri ile yüksek hafıza ve işlem gücüne ihtiyaç duyan optimizasyon algoritmaların tasarımında parametre seçiminin sistemin ve donanımın performansını ne şekilde etkilediği gösterilmiştir.

Bu tez çalışmasında çalışılan kabin yönlendirme işleminde, kabinlerdeki ve katlardaki bekleyen yolcu sayıları algoritmaya dahil edilmeden yalnızca katlardan gelen çağrılar ve yönleri alınarak yapılmaktadır. Yapılacak olan sonraki çalışmalarda kabinlerdeki ve katlardaki bekleyen yolcu sayılarının ve yolcunun gitmek istediği kat bilgisinin önceden alınarak algoritmaya dahil edilmesi işlemcinin yükünü artırmasına rağmen yönlendirme işleminin kalitesini ciddi anlamda artıracaktır.

Sonraki yıllarda üretilecek olan donanımlardaki gelişmelerle birlikte çözüm uzayının büyüdüğü daha kapsamlı ve yüksek doğrulukla çözüm sunan yönlendirme algoritmaları asansör sistemleri için kullanılabilir.

KAYNAKLAR

1. Pak, S. ve Uysal, A., "Grup asansör sistemleri için deney düzeneğinin tasarımı ve gerçekleştirilmesi", ***Zeugma II. Uluslararası Multidisipliner Çalışmalar Kongresi***, 2150–2155 (2019).

2. Alander, J. T., Ylien, J., ve Tyni, T., "Elevator group control using distributed genetic algorithm", ***Artificial Neural Nets And Genetic Algorithms***, 400–403 (1995).

3. Imrak, C. E., "Asansör sistemlerinin trafik analizi, dizaynı ve simülasyonu", Doktora Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, (1996).

4. Bolat, B., Erol, K. O., ve İrmak, C. E., "Genetic algorithms in engineering applications an the function of operators", ***Journal Of Engineering And Natural Sciences***, 4 (212): 264–271 (2004).

5. Cortés, P., Larrañeta, J., ve Onieva, L., "Genetic algorithm for controllers in elevator groups: analysis and simulation during lunchpeak traffic", ***Applied Soft Computing Journal***, 4 (2): 159–174 (2004).

6. Bolat, B., "Asansör kontrol sistemlerinin genetik algoritma ile simülasyonu", Doktora Tezi, ***Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, (2006).

7. Dağdelen, U., Bağış, A., ve Karaboğa, D., "İyileştirilmiş alan ağırlık algoritmasına dayalı grup asansör kontrolü", ***Asansör Sempozyumu***, 171–179 (2005).

8. Hiller, B. ve Tuchscherer, A., "Real-time destination-call elevator group control on embedded microcontrollers", ***Operations Research Proceedings***, 26 (October): 357–362 (2007).

9. Dursun, M. ve Sarıbaş, Ü. Ş., "Asansör sistemlerinde kabin hareketinin yapay sinir ağları ile denetlenmesi", ***Politeknik Dergisi***, 11 (2): 115–122 (2008).

10. Yang, S., Tai, J., ve Shao, C., "Dynamic partition of elevator group control system with destination floor guidance in up-peak traffic", ***Journal Of Computers***, 4 (1): 45–52 (2009).

11. Jamaludin, J., Rahim, N. A., ve Hew, W. P., "Development of a self-tuning fuzzy logic controller for ıntelligent control of elevator systems", ***Engineering Applications Of Artificial Intelligence***, 22 (8): 1167–1178 (2009).

12. Bolat, B., Cortés, P., Yalçın, E., ve Alışverişçi, M., "Optimal car dispatching for elevator groups using genetic algorithms", ***Intelligent Automation And Soft Computing***, 16 (1): 89–99 (2010).

13. Çiçekli, U. G. ve Kocamaz, M., "Paralel makinaların genetik algoritma ile çizelgelenmesinde mutasyon oranın etkinliği", ***Ege Akademik Bakis (Ege Academic Review)***, 10 (1): 199–199 (2010).

14. Bayğın, M. ve Karaköse, M., "Adaptif yapay bağışık sistem tabanlı grup asansör kontrol algoritması", ***Elektrik-Elektronik Ve Bilgisayar Sempozyumu***, 205–210 (2011).

15. Cortes, P., Onieva, L., Munuzuri, J., ve Guadix, J., "A viral system algorithm to optimize the car dispatching in elevator group control systems of tall buildings", ***Computers And Industrial Engineering***, 64 (1): 403–411 (2013).

16. Bolat, B., Altun, O., ve Cortés, P., "A particle swarm optimization algorithm for optimal car-call allocation in elevator group control systems", ***Applied Soft Computing Journal***, 13 (5): 2633–2642 (2013).

17. Beamurgia, M., Basagoiti, R., Rodríguez, I., ve Rodriguez, V., "A modified genetic algorithm applied to the elevator dispatching problem", ***Soft Computing***, 20 (9): 3595–3609 (2015).

18. Debnath, J. K. ve Serpen, G., "Real-Time Optimal Scheduling of a Group of Elevators in a Multi- Story Robotic Fully-Automated Parking Structure", ***Procedia Computer Science***, 61: 507–514 (2015).

19. Tartan, E. O. ve Ciftlikli, C., "A genetic algorithm based elevator dispatching method for waiting time optimization", ***IFAC-PapersOnLine***, 49 (3): 424–429 (2016).

20. Bolat, B., Altun, O., Cortes, P., Yildiz, Y. E., Bolat, B., Altun, O., Cortes, P., Yildiz, Y. E., ve Topal, A. O., "A comparison of metaheuristics for the allocation of elevators to calls in buildings a comparison of metaheuristics for the allocation of elevators to calls in buildings", ***Journal Of Polytechnic***, 20(3): 519–529 (2017).

21. Sorsa, J., Ehtamo, H., Kuusinen, J. M., Ruokokoski, M., ve Siikonen, M. L., "Modeling uncertain passenger arrivals in the elevator dispatching problem with destination control", ***Optimization Letters***, 12 (1): 171–185 (2018).

22. Çiflikli, C. ve Tartan, E. Ö., "Elevator parking approach in nearest car method", ***26th Signal Processing And Communications Applications Conference (SIU)***, 1–4 (2018).

23. Cetin, T. ve Yurdusev, M. A., "Genetic algorithm for networks with dynamic mutation rate", ***Journal Of The Croatian Association Of Civil Engineers***, 69 (12): 1101–1109 (2018).

24. Chou, S., Budhi, D. A., Dewabharata, A., ve Zulvia, F. E., "Improving elevator dynamic control policies based on energy and demand visibility", ***2018 3rd International Conference On Intelligent Green Building And Smart Grid (IGBSG)***, (2018).

25. İmrak, C. E. ve Gerdemeli, İ., "Asansörler ve yürüyen merdivenler 1.Basım", ***Birsen Yayınevi***, İstanbul, (2000).

26. Uysal, A., "Asansör grubu kontrolü için bulanık kontrolör tasarımı", Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, (2007).

27. Dağdelen, U., "Grup asansörleri için zeki kontrol sistemleri", Doktora Tezi, ***Erciyes Üniversitesi Fen Bilimleri Enstitüsü***, Kayseri, (2006).

28. Internet: World, R. G., "Fastest lift (Elevator)", **http://www.guinnessworldrecords.com/world-records/fastest-lift-(elevator)** .

29. Karaboğa, D., "Yapay zeka optimizasyon algoritmaları 3. Basım", ***Nobel Akademik Yayıncılık***, Ankara, (2014).

30. Çetin Elmas, "Yapay zeka uygulamaları", ***Seçkin Yayıncılık***, Ankara, (2007).

31. Koç, İ. O., "Gezgin satıcı problemi için çok populasyonlu paralel bir genetik algoritma tasarımı, geliştirilmesi ve analizi", Doktora Tezi, ***Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü***, Eskişehir, (2007).

32. Taşkın, Ç. ve Emel, G. G., "Sayısal yöntemlerde genetik algoritmalar 1.Basım", ***Alfa Aktüel Yayınları***, Bursa, (2009).

33. Internet: Arduino, "Arduino mega", **https://www.arduino.cc/en/Main/arduinoBoardMega/** .

34. Internet: Beagleboard, "Beaglebone", **https://beagleboard.org/bone** .

35. Internet: Arduino, "Arduino uno rev3", **https://store.arduino.cc/usa/arduino-uno-rev3** (2019).

EK AÇIKLAMALAR A.

ALGORİTMA TEST SENARYOLARI

Çizelge Ek A.1. 10 çağrılı senaryolar.

Senaryo	Çağrılar ve Yönleri
1	[[0, 1],[1,-1],[2, 1],[3, 1],[4,-1],[6, 1],[7,-1],[9,-1],[13,-1],[14,-1]]
2	[[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,-1],[9,-1],[10,-1]]
3	[[2, 1],[4,-1],[6, 1],[7, 1],[9,1],[10, 1],[10,-1],[11,-1],[13,-1],[15,-1]]
4	[[1, 1],[1,-1],[2, 1],[3, 1],[5,1],[6, 1],[8,-1],[9,-1],[11,-1],[15,-1]]
5	[[2, 1],[3,-1],[5, 1],[7, 1],[9,-1],[11, 1],[12,-1],[13,-1],[13,1],[14,1]]
6	[[2, 1],[3,-1],[5, 1],[7, 1],[9,-1],[11, -1],[11,1],[13,1],[13,-1],[14,1]]
7	[[1, -1],[1,1],[2, -1],[3, -1],[5,-1],[6, -1],[8,1],[9,1],[11,1],[15,-1]]
8	[[0, 1],[2,-1],[6, -1],[7, -1],[7,1],[8, 1],[10,-1],[12,-1],[13,-1],[14,1]]
9	[[2, -1],[3,1],[5, -1],[7, -1],[9,1],[11, 1],[11,-1],[13,-1],[13,1],[14,-1]]
10	[[0, -1],[1,1],[2, -1],[3, -1],[4,1],[6, -1],[7,1],[9,1],[13,1],[14,1]]

Çizelge Ek A.2. 20 çağrılı senaryolar.

Senaryo	Çağrılar ve Yönleri
11	[[0, 1],[1,1],[1, -1],[2, 1],[2, -1],[3, -1],[4,1],[4,-1],[5, -1],[6, -1], [6,1],[7,1],[8,1],[9,1], [11,1],[12,-1],[13,1],[13,-1],[14,1],[15,-1]]
12	[[1, 1],[1, -1],[2,1],[2,-1],[4, -1],[5, 1],[5, 1],[7, -1],[7, -1],[8, 1], [8,-1],[9, 1],[9, -1], [10,1],[11, 1],[12,1],[12,-1],[13,1],[14,-1],[15,-1]]
13	[[0, 1],[1, 1],[1,-1],[2,-1],[5, -1],[5, 1],[6, -1],[7, 1],[7, -1],[8, 1], [8,-1],[9, 1],[9, -1], [10, 1], [11, 1],[12,1],[12,-1],[13,1],[14,-1],[15,-1]]
14	[[0, 1],[1,1],[1, -1],[2, -1],[3, 1],[3, -1],[4, -1],[4, 1],[6,1],[7,1], [7,-1],[8, -1],[8, -1],[10, 1],[11,1],[12,1],[12,-1],[14,1],[14,-1],[15,-1]]
15	[[1,-1],[1, 1],[2,1],[3,1],[5, -1],[6, 1],[6, -1],[7, 1],[7, -1],[8,1], [9,1],[9,-1],[10, -1],[10, 1],[11, 1],[12,1],[12,-1],[13,1],[14,-1],[15,-1]]
16	[[1,-1],[1, 1],[2,-1],[3,-1],[5, -1],[5, 1],[6, -1],[7, 1],[7, -1],[8, 1], [8,-1],[9,-1],[10, -1],[10, 1],[11, 1],[12,1],[12,-1],[13,1],[14,1],[14,-1]]
17	[[0, 1],[1, 1],[2,1],[2,-1],[5, -1],[6, 1],[6, -1],[7, 1],[7, -1],[8, 1], [8,-1],[9, 1],[9, -1],[10, 1],[11, 1],[12,1],[12,-1],[13,1],[14,-1],[15,-1]]
18	[[3, 1],[5,1],[5, -1],[6, 1],[7, 1],[8, -1],[9,-1],[7,1],[8, 1],[9, 1], [10, 1],[11,1],[11,-1],[12,1],[12,-1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
19	[[0, 1],[1,1],[2, -1],[3, 1],[4, 1],[8, -1],[9,-1],[7,1],[8, 1],[9, 1], [10, 1],[11,1],[11,-1],[12,1],[12,-1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
20	[[1, -1],[2, -1],[3,-1],[4,-1],[5, -1],[6, -1],[7, -1],[8, -1],[9, -1],[9,1], [10,1],[10, -1],[11,1],[11,-1],[12,1],[12, -1],[13,1],[13, -1],[14, -1],[15, -1]]

Çizelge Ek A.3. 4'den 30'a kadar artan çağrılı senaryolar 1-14.

Senaryo	Çağrılar ve Yönleri
21	[[1,-1],[3,-1], [3,1],[7,-1]]
22	[[1,-1],[2, 1],[3, 1],[4,-1],[7,-1]]
23	[[1,-1],[2, 1],[3, 1],[4,-1],[6, 1],[7,-1]]
24	[[0, 1],[1,-1],[2, 1],[3, 1],[4,-1],[6, 1],[7,-1]]
25	[[0, 1],[1,-1],[2, 1],[3, 1],[4,-1],[6, 1],[7,-1],[13,-1]]
26	[[0, 1],[1,-1],[2, 1],[3, 1, 5],[4,-1],[6, 1],[7,-1],[9,-1],[13,-1]]
27	[[0, 1],[1,-1],[2, 1],[3, 1],[4,-1],[6, 1],[7,-1],[9,-1],[13,-1],[14,-1]]
28	[[0, 1],[1,-1],[1, 1],[2, 1],[3, 1],[4,1],[6, 1],[7,-1],[9,1, 4],[13,-1],[14,1]]
29	[[0, 1],[1,-1],[1, 1],[2, 1],[3, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[13,-1],[12,-1]]
30	[[0, 1],[1,-1],[1, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[11,-1],[13,-1],[14,-1],[15,-1]]
31	[[0, 1],[1,-1],[1, 1],[2, 1],[3, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[13,-1],[14,-1],[15,-1]]
32	[[0, 1],[1,-1],[1, 1],[2, 1],[3, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[11,-1],[13,-1],[14,-1],[15,-1]]
33	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[11,-1],[13,-1],[14,-1],[15,-1]]
34	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[11,-1],[13,-1],[14,1],[14,-1],[15,-1]]

Çizelge Ek A.4. 4'den 30'a kadar artan çağrılı senaryolar 15-27.

Senaryo	Çağrılar ve Yönleri
35	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[4,1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1, 4],[11,-1],[13,-1],[14,1],[14,-1],[15,-1]]
36	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[4,1],[5, 1],[6, 1],[7,-1],[8,-1],[9,-1],[11,-1],[12,1],[13,-1],[14,1],[14,-1],[15,-1]]
37	[[0, 1],[1,-1,],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[4,1],[5, 1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[11,-1],[12,1],[13,-1],[14,1],[14,-1],[15,-1]]
38	[[0, 1],[1,-1,],[1, 1],[2, -1],[2, 1],[3, 1],[4,-1],[4,1],[5, 1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[11,-1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
39	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[11,-1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
40	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[11,-1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
41	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[11,-1],[11,1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
42	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,-1],[8,-1],[9,-1],[10,-1],[11,-1],[11,1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
43	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,-1],[9,-1],[10,-1],[11,-1],[11,1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
44	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,-1],[9,-1],[10,1],[10,-1],[11,-1],[11,1],[12,1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
45	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,-1],[9,-1],[10,1],[10,-1],[11,-1],[11,1],[12,1],[12,-1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
46	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,-1],[9,1],[9,-1],[10,1],[10,-1],[11,-1],[11,1],[12,1],[12,-1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]
47	[[0, 1],[1,-1],[1, 1],[2, -1],[2, 1],[3, 1],[3, -1],[4,-1],[4,1],[5, 1],[5, -1],[6, 1],[6, -1],[7,1],[7,-1],[8,1],[8,-1],[9,1],[9,-1],[10,1],[10,-1],[11,-1],[11,1],[12,1],[12,-1],[13,1],[13,-1],[14,1],[14,-1],[15,-1]]

EK AÇIKLAMALAR B.

ALGORİTMALARIN TEST SONUÇLARI

Çizelge Ek B.1. Sabit 10 çağrılı senaryoların deney sonuçları.

Yöntem	Senaryo	Kabin Ataması	Cevaplama Süresi (s)	Enerji Tüketimi (Wh)
GA	1	[3, 0, 3, 0, 2, 2, 4, 4, 1, 1]	62,775000100	106,6189431
GELENEKSEL	1	[3, 0, 1, 4, 2, 2, 2, 2, 1, 3]	125,46400000	165,9323981
GA	2	[0, 4, 4, 0, 1, 1, 2, 2, 2, 3]	65,317000150	93,90208377
GELENEKSEL	2	[1, 4, 4, 2, 0, 3, 2, 2, 2, 2]	100,40400000	105,4542613
GA	3	[2, 2, 1, 1, 3, 0, 4, 4, 4, 0]	73,81799984	146,2538025
GELENEKSEL	3	[2, 4, 1, 0, 4, 3, 3, 3, 3, 3]	120,3800001	175,9040423
GA	4	[2, 2, 4, 4, 3, 0, 0, 1, 1, 1]	74,93799996	90,05263850
GELENEKSEL	4	[2, 4, 1, 2, 0, 3, 3, 3, 3, 3]	110,2539999	105,4822161
GA	5	[2, 2, 2, 0, 3, 1, 1, 4, 4, 4]	85,07299995	111,6153211
GELENEKSEL	5	[2, 4, 1, 1, 0, 0, 0, 0, 0, 0]	149,5200000	129,7037953
GA	6	[3, 3, 3, 1, 0, 4, 4, 2, 2, 2]	100,6589999	177,0537962
GELENEKSEL	6	[0, 1, 2, 3, 4, 4, 4, 4, 4, 4]	151,1400001	204,0036923
GA	7	[3, 2, 2, 2, 0, 4, 4, 1, 1, 1]	55,97099996	93,90338108
GELENEKSEL	7	[0, 0, 0, 0, 1, 2, 3, 4, 2, 1]	103,7390001	163,2014613
GA	8	[3, 4, 3, 4, 2, 2, 1, 1, 0, 0]	70,03999996	172,0520230
GELENEKSEL	8	[3, 2, 2, 2, 4, 1, 0, 1, 3, 2]	113,7409999	200,5752962
GA	9	[0, 2, 2, 2, 1, 1, 1, 4, 4, 3]	93,21099997	115,4545123
GELENEKSEL	9	[0, 0, 0, 0, 1, 2, 3, 4, 1, 2]	108,4310000	232,5028919
GA	10	[2, 1, 1, 3, 0, 0, 4, 4, 4, 4]	107,9259999	65,80247782
GELENEKSEL	10	[4, 1, 0, 3, 2, 2, 2, 2, 1, 1]	129,2800000	112,0397965

Çizelge Ek B.2. Sabit 20 çağrılı senaryoların deney sonuçları.

Yöntem	Senaryo	Kabin Ataması	Cevaplama Süresi (s)	Enerji Tüketimi (Wh)
GA	11	[2, 2, 2, 4, 4, 4, 1, 1, 1, 3, 0, 3, 3, 3, 3, 3, 0, 0, 0, 0]	137,9080000	158,1584644
GELENEKSEL	11	[2, 4, 1, 2, 4, 4, 1, 0, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]	239,5359998	179,0595903
GA	12	[2, 2, 2, 2, 4, 4, 4, 4, 3, 3, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1]	122,9510000	185,1070654
GELENEKSEL	12	[2, 4, 1, 0, 2, 2, 2, 4, 1, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]	215,4240000	231,7660983
GA	13	[2, 2, 2, 2, 1, 1, 1, 0, 0, 0, 3, 3, 4, 3, 3, 3, 4, 4, 4, 4]	119,3720000	193,956259
GELENEKSEL	13	[2, 4, 1, 0, 2, 2, 2, 4, 1, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]	214,8629999	233,0081087
GA	14	[4, 4, 4, 2, 2, 2, 2, 1, 1, 1, 3, 0, 0, 3, 0, 3, 3, 0, 3, 0]	130,9830000	186,6557147
GELENEKSEL	14	[2, 4, 1, 2, 2, 2, 4, 1, 0, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]	218,3820000	205,9879985
GA	15	[3, 3, 3, 2, 2, 2, 2, 4, 4, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0]	132,4410000	200,5073683
GELENEKSEL	15	[2, 4, 1, 0, 3, 2, 2, 4, 0, 0, 0, 0, 4, 2, 2, 2, 2, 2, 2, 2]	239,0800002	283,0187796
GA	16	[0, 1, 0, 1, 1, 1, 1, 0, 0, 4, 4, 4, 4, 2, 2, 2, 2, 2, 3, 3]	126,6689999	190,1081587
GELENEKSEL	16	[0, 4, 2, 2, 2, 2, 2, 2, 2, 1, 3, 1, 1, 1, 4, 0, 4, 2, 2, 2]	217,4449999	266,4297808
GA	17	[2, 2, 2, 2, 4, 4, 4, 4, 4, 3, 0, 3, 1, 0, 1, 1, 0, 1, 0, 1]	128,1170001	193,9835753
GELENEKSEL	17	[2, 4, 1, 0, 2, 2, 2, 4, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]	234,1730001	234,052516
GA	18	[2, 2, 2, 2, 4, 4, 4, 4, 1, 3, 1, 0, 0, 1, 3, 0, 3, 3, 1, 0]	119,2019999	240,1767097
GELENEKSEL	18	[2, 4, 1, 0, 3, 2, 4, 4, 4, 4, 4, 2, 2, 2, 2, 2, 2, 2, 1, 3]	201,7070000	308,7266827
GA	19	[1, 1, 1, 1, 2, 2, 2, 2, 3, 0, 0, 3, 4, 4, 4, 0, 3, 3, 4, 0]	124,6780000	251,7061198
GELENEKSEL	19	[2, 4, 1, 0, 3, 2, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 4, 3, 0]	200,3239999	380,3333864
GA	20	[1, 1, 1, 1, 1, 3, 3, 3, 4, 4, 4, 4, 2, 2, 2, 2, 0, 0, 0, 0]	129,6619999	198,9573696
GELENEKSEL	20	[2, 2, 2, 2, 2, 4, 1, 0, 3, 4, 4, 1, 1, 0, 3, 1, 1, 1, 1, 1]	236,8160002	303,3487368

Çizelge Ek B.3. 4'den 16'ya artan çağrılı senaryoların deney sonuçları.

Yöntem	Senaryo	Kabin Ataması	Cevaplama Süresi (s)	Enerji Tüketimi (Wh)
GA	21	[2, 3, 0, 1]	24,54500008	15,00134690
GELENEKSEL	21	[2, 0, 1, 1]	34,03499985	10,04575042
GA	22	[0, 1, 2, 4, 3]	24,83000016	20,04795275
GELENEKSEL	22	[0, 1, 2, 4, 3]	20,50600004	20,00699766
GA	23	[2, 4, 0, 3, 1, 1]	45,04600000	31,55138198
GELENEKSEL	23	[2, 1, 0, 3, 4, 4]	51,00899982	40,40449242
GA	24	[2, 4, 0, 3, 3, 1, 1]	45,42700005	31,56818508
GELENEKSEL	24	[2, 1, 0, 3, 3, 4, 4]	51,05999994	40,42919578
GA	25	[4, 2, 2, 0, 3, 3, 1, 1]	53,56699991	71,19855696
GELENEKSEL	25	[2, 2, 1, 0, 3, 3, 4, 4]	49,78399992	59,69257239
GA	26	[3, 3, 1, 1, 0, 2, 2, 4, 4]	55,70000005	100,4511879
GELENEKSEL	26	[3, 0, 1, 4, 2, 2, 2, 2, 1]	120,6970000	107,0477695
GA	27	[3, 3, 1, 1, 4, 4, 2, 2, 0, 0]	69,62899995	101,6041826
GELENEKSEL	27	[3, 0, 1, 4, 2, 2, 2, 2, 1, 3]	125,4879999	165,9389069
GA	28	[2, 4, 4, 3, 0, 3, 1, 1, 3, 1, 0]	70,94599986	155,5032923
GELENEKSEL	28	[2, 2, 0, 3, 3, 1, 1, 1, 4, 4, 4]	80,13800001	81,25217119
GA	29	[1, 1, 2, 2, 4, 4, 3, 0, 3, 0, 3, 0]	73,27300000	110,4891084
GELENEKSEL	29	[2, 4, 2, 2, 1, 0, 3, 3, 3, 3, 3, 3]	121,4229999	106,6169798
GA	30	[3, 3, 3, 3, 2, 2, 2, 1, 1, 1, 4, 4, 0]	91,16499996	132,0045292
GELENEKSEL	30	[3, 0, 4, 4, 2, 1, 2, 2, 2, 1, 0, 3, 4]	117,6459999	232,5380027
GA	31	[3, 0, 0, 3, 0, 3, 2, 2, 2, 4, 4, 4, 1, 1]	89,97599983	125,4701514
GELENEKSEL	31	[3, 0, 0, 1, 4, 4, 2, 2, 2, 2, 2, 1, 3, 4]	145,9789999	202,9059513
GA	32	[3, 0, 0, 3, 0, 3, 4, 4, 4, 1, 1, 2, 2, 2, 2]	93,83400011	163,9529451
GELENEKSEL	32	[3, 0, 0, 1, 4, 4, 2, 2, 2, 2, 2, 1, 3, 4, 0]	145,1410000	238,6751756
GA	33	[3, 0, 0, 3, 0, 3, 4, 4, 4, 1, 1, 2, 1, 2, 2, 2]	93,88000011	172,8054099
GELENEKSEL	33	[3, 0, 0, 1, 4, 4, 2, 2, 2, 2, 2, 2, 1, 3, 4, 0]	163,7920001	239,8509096

Çizelge Ek B.4. 17'den 30'a artan çağrılı senaryoların deney sonuçları.

Yöntem	Senaryo	Kabin atama	Cevaplama Süresi (s)	E. Tüketimi (Wh)
GA	34	[3, 0, 0, 3, 0, 0, 3, 4, 4, 4, 1, 1, 2, 1, 2, 2, 2]	94,09600019	208,6051886
GELENEKSEL	34	[3, 0, 0, 1, 4, 4, 2, 1, 3, 1, 1, 4, 4, 0, 2, 2, 2]	134,6559999	297,5650168
GA	35	[3, 0, 0, 0, 0, 0, 3, 3, 4, 4, 4, 2, 2, 2, 2, 1, 1, 1]	104,7700000	176,3080920
GELENEKSEL	35	[3, 0, 0, 1, 1, 4, 4, 2, 3, 3, 0, 0, 2, 2, 1, 2, 2, 1]	124,3810000	197,5169316
GA	36	[3, 3, 3, 3, 3, 4, 4, 4, 4, 0, 2, 2, 2, 0, 2, 1, 1, 1, 1]	112,9440000	231,3483344
GELENEKSEL	36	[3, 0, 0, 1, 1, 4, 4, 2, 2, 3, 3, 4, 4, 4, 1, 0, 2, 2, 2]	123,1930001	249,8849395
GA	37	[3, 0, 0, 0, 3, 0, 0, 3, 3, 4, 4, 1, 1, 4, 2, 2, 2, 1, 2, 2]	118,5610001	254,4756171
GELENEKSEL	37	[3, 0, 0, 1, 1, 4, 4, 2, 2, 3, 3, 4, 4, 4, 4, 1, 0, 2, 2, 2]	131,6390002	251,0086841
GA	38	[3, 3, 3, 1, 1, 3, 3, 1, 1, 1, 4, 4, 0, 4, 4, 2, 2, 2, 2, 0, 2]	129,6990001	263,3366533
GELENEKSEL	38	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 3, 3, 4, 4, 4, 4, 0, 1, 1, 0, 2]	132,8810000	242,1152225
GA	39	[3, 3, 3, 1, 1, 3, 1, 1, 3, 1, 0, 0, 4, 4, 4, 2, 2, 2, 2, 0, 2, 2]	133,8640001	287,5691506
GELENEKSEL	39	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 1, 0, 1, 0, 2]	152,0369999	243,3035684
GA	40	[3, 0, 3, 0, 3, 3, 0, 3, 0, 0, 4, 4, 4, 4, 1, 2, 2, 2, 1, 1, 2, 2, 1]	135,5000000	250,2297144
GELENEKSEL	40	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 3, 3, 4, 1, 4, 4, 1, 1, 1, 0, 0, 2, 2]	153,3369999	279,1362279
GA	41	[3, 3, 3, 1, 1, 1, 3, 1, 3, 1, 1, 0, 4, 4, 0, 4, 2, 0, 0, 2, 2, 2, 2, 2]	146,9679999	282,2264517
GELENEKSEL	41	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 3, 3, 1, 1, 1, 1, 0, 4, 4, 4, 0, 3, 3]	157,5200000	288,0350000
GA	42	[3, 0, 0, 3, 0, 3, 0, 0, 3, 0, 3, 1, 4, 1, 1, 4, 4, 2, 1, 2, 2, 2, 2, 4, 2]	150,7270000	310,2787418
GELENEKSEL	42	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 4, 1, 0, 1, 0, 3, 3]	169,3020000	250,6129651
GA	43	[3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 3, 4, 1, 2, 2, 2, 2, 1, 2, 2, 0, 0, 0, 0, 0, 0]	159,3540001	270,2344564
GELENEKSEL	43	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 2, 3, 3, 1, 1, 1, 1, 0, 4, 4, 4, 0, 3, 3, 3]	156,2030001	297,9557862
GA	44	[3, 0, 0, 3, 3, 4, 4, 0, 4, 0, 4, 3, 0, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2]	176,7540002	249,8428428
GELENEKSEL	44	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 2, 2, 3, 3, 1, 1, 1, 1, 4, 0, 4, 4, 0, 3, 3, 3]	156,062000	299,2543358
GA	45	[3, 0, 0, 3, 3, 3, 0, 3, 3, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 4, 1, 1, 1, 1]	180,2419999	213,6891604
GELENEKSEL	45	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 2, 2, 3, 3, 1, 0, 1, 1, 0, 4, 4, 4, 3, 3, 3, 3, 3]	193,4250002	304,1742905
GA	46	[1, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 1, 4, 4, 2, 2, 3, 3, 2, 2, 3, 3, 0, 0, 0, 0, 0, 0, 0]	186,6550000	287,6144783
GELENEKSEL	46	[3, 0, 0, 1, 1, 4, 4, 2, 2, 2, 2, 2, 2, 2, 3, 3, 1, 0, 1, 1, 0, 4, 4, 4, 3, 3, 3, 3, 3]	192,1150000	305,3817199
GA	47	[4, 4, 4, 4, 4, 2, 2, 2, 2, 1, 2, 1, 1, 1, 1, 3, 0, 0, 0, 3, 0, 3, 0, 0, 0, 3, 3, 0, 3, 3]	188,1670001	240,2367820
GELENEKSEL	47	[2, 4, 1, 0, 3, 2, 2, 2, 4, 4, 1, 1, 0, 0, 0, 1, 4, 4, 4, 4, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3]	252,5799999	444,1165550

ÖZGEÇMİŞ

Semih PAK 1991 yılında Ankara'da doğdu. İlk, orta ve lise öğrenimini Ankara'da tamamladı. 2010 yılında Karabük Üniversitesi (KBÜ) Teknoloji Fakültesi Mekatronik Mühendisliği Bölümü'nde öğrenimine başlayıp 2014 yılında iyi bir derece ile mezun oldu. 2016 yılında KBÜ Fen Bilimleri Enstitüsü Mekatronik Mühendisliği Ana Bilim Dalı'nda yüksek lisansa başladı ve 2019 yılında mezun oldu. 2017 yılında KBÜ Teknoloji Fakültesi Mekatronik Mühendisliği Bölümü'nde araştırma görevlisi olarak işe başladı ve halen aynı yerde çalışmaktadır.

ADRES BİLGİLERİ

Adres : Karabük Üniversitesi
Teknoloji Fakültesi
Mekatronik Mühendisliği Bölümü
Demir Çelik Kampüsü 78050 / KARABÜK
Tel : 0370 418 9297
E-posta : semihpak@karabuk.edu.tr