Gazi Üniversitesi
Fen Bilimleri Dergisi
PART C: TASARIM VE TEKNOLOJİ

Gazi University
Journal of Science
PART C: DESIGN AND TECHNOLOGY

GU J Sci, Part C, 12(1): 108-116 (2024)

Robot Tutucu Problemi için Çok Stratejili Aritmetik Optimizasyon Algoritması

Mustafa Yusuf YILDIRIM[1*], Rüştü AKAY[2]

[1] *Niğde Ömer Halisdemir Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, Niğde*

[2] *Erciyes Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, Kayseri*

Makale Bilgisi

Araştırma makalesi
Başvuru: 31/10/2023
Düzeltme: 19/12/2023
Kabul: 25/12/2023

Anahtar Kelimeler

Robot Tutucu
Aritmetik
Optimizasyon
Çoklu-Strateji
Metasezgisel

Article Info

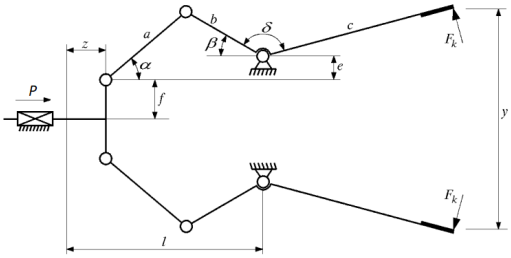
Research article
Received: 31/10/2023
Revision: 19/12/2023
Accepted: 25/12/2023

Keywords

Robot Gripper
Arithmetic
Optimization
Multi-Strategy
Metaheuristic

Grafik Özet (Graphical/Tabular Abstract)

Bu çalışmada, Aritmetik Optimizasyon Algoritmasının (AOA) orijinal güncelleme denkleminin değiştirildiği ve çoklu strateji konseptinin eklendiği yeni bir algoritma önerilmektedir. Önerilen algoritma, robot tutucu probleminde AOA ve diğer güncel algoritmalardan daha iyi performans göstermiştir. / In this study, a new algorithm is proposed in which the original update equation of Arithmetic Optimization Algorithm (AOA) is modified and the multi-strategy concept is added. The proposed algorithm outperformed AOA and other state-of-the-art algorithms in robot gripper problem.



Şekil A: *Robot tutucu modeli /* ***Figure A****: A robot gripper model*

Önemli noktalar (Highlights)

- *Keşif-kullanım dengesine odaklanmış bir güncelleme modifikasyonu / An update modification focused on exploration-exploitation balance*
- *Kendi kendine uyarlanabilir çoklu strateji konsepti / Self-adaptive multi-strategy concept*
- *Robot tutucu probleminde kayda değer iyileşme / Significant improvement in robot gripper problem*

Amaç (Aim): *Bu çalışmanın amacı, endüstriyel sistemlerde kullanılan robot tutucularının tasarım optimizasyonunu ele almak ve nesnelerin en az manevrayla ve zarar vermeden tutulabilmesine yönelik bir çözüm sunmaktır. / The aim of this study is to discuss design optimization of robot grippers used in industrial systems and to propose a method to hold objects without damaging them and with minimal maneuvering.*

Özgünlük (Originality): *Bu çalışmanın özgünlüğü, matematik esinli bir metasezgisel olan AOA'nın arama performansını iyileştirmek için Çok Stratejili Aritmetik Optimizasyon Algoritması (ÇSAOA) adında yeni bir algoritma önerilmesidir. ÇSAOA, farklı bir güncelleme mekanizmasının eklenmesi ve en iyi güncelleme stratejisine odaklanmasıyla kendinden uyarlanabilen bir yapıya sahipir. / The originality of this study is that a new algorithm called Multi-Strategy Arithmetic Optimization Algorithm (MSAOA) is proposed to improve search performance of AOA, a mathematics-inspired metaheuristic. MSAOA has a self-adaptive structure by adding a different update mechanism and focusing on the best update strategy.*

Bulgular (Results): *ÇSAOA robot tutucu probleminde, orijinal AOA'ya göre hem performans hem de hesaplama süresi açısından daha iyi sonuçlar elde etmiştir. Ayrıca önerilen algoritma literatürdeki diğer güncel algoritmalarla karşılaştırıldığında da en performanslı yöntem olmuştur. / MSAOA achieved better results in terms of both performance and computation time in the robot gripper problem than original AOA. In addition, the proposed algorithm was the most performant method when compared to other state-of-the-art algorithms in the literature.*

Sonuç (Conclusion): *ÇSAOA robot tutucuların nesnelerin en az manevrayla ve zarar vermeden tutulabilmesine yönelik etkili bir yöntemdir. Bu çalışma, endüstriyel robotik sistemlerde kullanılan robot tutucularının tasarımında değerli bir katkı sağlamaktadır. / MSAOA is an effective method for robot grippers to grip objects without damaging them and with minimal maneuvering. This study makes a valuable contribution to design of robot grippers used in industrial robotic systems.*

*Corresponding author, e-mail: myyildirim@erciyes.edu.tr
DOI: 10.29109/gujsc.1383797

GU J Sci, Part C, 12(1): 108-116 (2024)

Gazi Üniversitesi
Fen Bilimleri Dergisi
PART C: TASARIM VE TEKNOLOJİ

Gazi University
Journal of Science
PART C: DESIGN AND TECHNOLOGY

http://dergipark.gov.tr/gujsc

Robot Tutucu Problemi için Çok Stratejili Aritmetik Optimizasyon Algoritması

Mustafa Yusuf YILDIRIM[1*], Rüştü AKAY[2]

[1] *Niğde Ömer Halisdemir Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, ORCID:0000-0003-0302-8466, 51240, Niğde*

[2] *Erciyes Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, ORCID:0000-0002-3585-3332, 38030, Kayseri*

Makale Bilgisi

Araştırma makalesi
Başvuru: 31/10/2023
Düzeltme: 19/12/2023
Kabul: 25/12/2023

Anahtar Kelimeler

Robot Tutucu
Aritmetik
Optimizasyon
Çoklu-Strateji
Metasezgisel

Öz

Günümüzde endüstriyel sistemlerde nesnelerin kavranması, taşınması ve sabitlenmesi için kullanılan robot tutucular önemli araçlar olarak öne çıkmaktadır. Özellikle robotik sistemlerde, bir nesneyi en az manevrayla zarar vermeden tutabilme yeteneği büyük önem taşımaktadır. Bundan dolayı, son yıllarda robot tutucularının tasarım optimizasyonu ilgi çeken bir araştırma konusu haline gelmiştir. Bu çalışmada bu tasarım problemi için Aritmetik Optimizasyon Algoritması (AOA) iyileştirilmiş ve Çok Stratejili Aritmetik Optimizasyon Algoritması (ÇSAOA) adında yeni bir algoritma önerilmiştir. Bu algoritmada hem orijinal AOA'nın güncelleme mekanizmasını modifiye edilmiş, hem de farklı bir güncelleme mekanizması eklenilerek kendinden uyarlanabilen bir algoritma haline getirilmiştir. Bu yaklaşım, en iyi güncelleme stratejisine odaklanarak problemi daha verimli bir şekilde çözmeye olanak sağlamıştır. ÇSAOA, robot tutucu problemine uygulandığında, orijinal algoritmaya göre hem performans hem de hesaplama süresi açısından daha iyi sonuçlar ürettiği gözlemlenmiştir. Ayrıca, bu yeni algoritma literatürdeki diğer benzer algoritmalarla karşılaştırılmış ve önerilen ÇSAOA'nın daha performanslı algoritma olduğu görülmüştür.

Multi-Strategy Arithmetic Optimization Algorithm for Robot Gripper Problem

Article Info

Research article
Received: 31/10/2023
Revision: 19/12/2023
Accepted: 25/12/2023

Keywords

Robot Gripper
Arithmetic
Optimization
Multi-Strategy
Metaheuristic

Abstract

Today, robot grippers used for grasping, moving and fixing objects in industrial systems stand out as important tools. Especially in robotic systems, the ability to hold an object without damaging it with minimal maneuvering is of great importance. Therefore, the design optimization of robot grippers has become an interesting research topic in recent years. In this study, Arithmetic Optimization Algorithm (AOA) has been improved for this design problem and a new algorithm called Multi-Strategy Arithmetic Optimization Algorithm (MSAOA) has been proposed. In this algorithm, the update mechanism of original AOA was modified and a different update mechanism was added, making it a self-adaptive algorithm. This approach allowed solving the problem more efficiently by focusing on the best update strategy. When MSAOA was applied to the robot gripper problem, it was observed that it produced better results in terms of both performance and calculation time compared to the original algorithm. Additionally, this new algorithm was compared with other similar algorithms in the literature and it was found that the proposed MSAOA was the most performant algorithm.

1. GİRİŞ (INTRODUCTION)

Tutucular, bir nesnenin kavranacağı taşıyıcı araç ile arasında kritik bir bağlantı sağlayan alt sistemler olarak işlev görür. Bu bağlantı, nesnenin taşıyıcı araca göre konumunun belirlenmesi ve değiştirilmesi gibi önemli işlemlerde hayati bir rol oynar. Robotik teknolojide tutucular, işlevsellik açısından oldukça çeşitlilik gösteren temel birimler arasında yer alır. Bu durum, robotların esnek makineler olmalarına karşın, tutucuların belirgin ve özelleştirilmiş görevleri üstlenmelerinden kaynaklanmaktadır. Bu bağlamda, farklı ihtiyaçlara uyum sağlama ve güvenilir sistemler oluşturma arzusu, robot tutucu tasarımının giderek artan bir önem kazanmasına sebep olmaktadır. Robot uygulamalarında kullanılan tutucular, sadece montaj ve kaynak işlemleri gibi temel görevlerle

*Corresponding author, e-mail: myyildirim@erciyes.edu.tr
DOI: 10.29109/gujsc.1383797

sınırlı kalmazlar; parçaların taşınmasından yüksek sıcaklıklı kaynak operasyonlarına, radyoaktif maddelerin güvenli bir şekilde tutulmasına, bombaların güvenli patlatılmasına ve hatta mayınların veya gemi enkazlarının araştırılmasına kadar geniş bir yelpazede kullanılırlar. Bu mekanizmalar, iş parçalarını sadece kavramakla kalmaz, aynı zamanda onların konumlarını değiştirebilir ve istenen yönde hareket ettirebilirler. Ek olarak bu sistemler, sistemin içinde herhangi bir parçanın varlığını belirten algılayıcıları da içerebilirler. Bu nedenlerden dolayı, iyi bir tutucu tasarımı robotik sistemlere büyük katkı sağlar. Bu tasarım süreci hafiflik, küçük boyutlar, rijitlik, çoklu görev yeteneği, basitlik ve bakım gerektirmeme gibi güçlü kısıtlamaları dikkate almalıdır [1,2].

Optimizasyon çalışmalarının artan önemi ile birlikte, robotik araştırma alanlarında nesnelerin kavranması ilgi çekici bir araştırma konusu olmuştur. Özellikle robotik sistemlerde robot tutucuların, nesneye zarar vermeden etkili bir şekilde kavrama yeteneği, robot tasarımında dikkate alınması gereken önemli optimizasyon problemlerinden biri haline gelmiştir. Tutucu tasarım optimizasyonuna yönelik literatürde çeşitli güncel çalışmalar yapılmıştır. Yıldız vd. çekirge optimizasyon algoritması ve Nelder-Mead algoritmasının bir hibrit versiyonunu önermişlerdir. Yazarlar, hızlı ve doğru çözümler gerektiren bir robot tutucu mekanizması tasarlamayı amaçlamıştır. Ancak bu yeni yöntem, avantajlarını göstermek amacıyla bir araç yan çarpma tasarım problemi, çoklu kavrama disk problemi ve üretim optimizasyon problemini çözmek için de kullanılmıştır. Geliştirilen yöntem gerçek dünya mühendislik problemlerini etkili bir şekilde çözebilecek temel bir optimizasyon yaklaşımı olarak ortaya konmuştur [3]. Rao vd. maksimum ve minimum kavrama kuvvetleri arasındaki fark, kuvvet iletim oranı, kaydırma iletim oranı, tutucunun tüm elemanlarının uzunluğu gibi tutucuya ait farklı uygunluk fonksiyonlarını bir arada ele almışlar ve öğrenme-öğretme tabanlı algoritmayı uygulamışlardır. Bu algoritma, literatürdeki diğer optimizasyon algoritmalarına göre daha iyi veya rekabetçi sonuçlar vermiştir [4]. Datta vd. eyleyici modelinin entegre edildiği daha gerçekçi bir tutucu modeli kullanmışlardır. Burada çok amaçlı evrimsel algoritma robot tutucunun boyutları ve eklemlerinin açısı gibi tasarım değişkenlerini en iyi şekilde bulmak için uygulanmıştır. [5]. Mahanta vd. otomatik malzeme taşıma işlemlerinde optimum bir robot tutucunun konfigürasyonunun bulunmasına yönelik çok amaçlı karınca aslanı optimizasyon algoritması adında yeni bir yöntem önermişlerdir. Çalışmada, bu algoritma kullanılarak üç farklı robotik tutucunun optimum boyutları belirlenmiştir. [6]. Dong vd. 22 tasarım değişkeni içeren üç parmağa sahip iki robot tutucunun optimum tasarımına odaklanmışlardır. Bu tasarım değişkenleri üç parmağın uzunluğu ve genişliği, üç eklemin yarıçapı, avuç genişliği gibi parametreleri içerir. Yazarlar üç uygunluk fonksiyonu ve çok sayıda geometrik kısıtlama içeren bu modelin optimizasyonu için genetik algoritmayı uygulamışlardır [7]. Zhong vd. bal porsuğu algoritmasını lévy tabanlı bir iyileştirme ekleyerek geliştirmişler ve orijinal algoritmanın optimizasyon performansını artırmışlardır. Bu önerilen algoritma önce CEC2020 problemlerinde ve sonra da robot tutucu tasarım problemine başarıyla uygulanmıştır [8]. Dörterler vd. robot tutucu probleminde parçacık sürü optimizasyonu, yapay alg algoritması ve gri kurt optimizasyonu gibi birkaç metasezgisel algoritmanın çok amaçlı formunu kullanmışlardır. İki farklı robot tutucu konfigürasyonunda, uygulanan algoritmaların performansları pareto-ön eğriler ve hiper hacim metrikleri kullanılarak incelenmiştir [9]. Baskın olmayan sıralamalı genetik algoritmaların [10,11,12] ve topoloji optimizasyonunun [13,14,15] kullanıldığı çalışmalar da mevcuttur.

Literatüre göre, robot tutucu probleminin bazı metasezgisel algoritmalarla çözülmesine yönelik çalışmalar olsa da, matematik esinli ve güncel metasezgisellerin bu problem için kullanılmasına yönelik çalışmalara sık rastlanmamaktadır. Bu çalışmanın katkısı son zamanlarda geliştirilen ve matematik esinli bir metasezgisel olan Aritmetik Optimizasyon Algoritması'nın arama performansını geliştirmek için Çok Stratejili Aritmetik Optimizasyon Algoritması adında yeni bir algoritma önerilmesi ve bu algoritmanın robot tutucu problemine uygulanmasıdır. Çalışmada önerilen algoritmanın hem orijinal algoritmayla hem de literatürdeki benzer algoritmalarla performanslarının karşılaştırılması amaçlanmıştır.

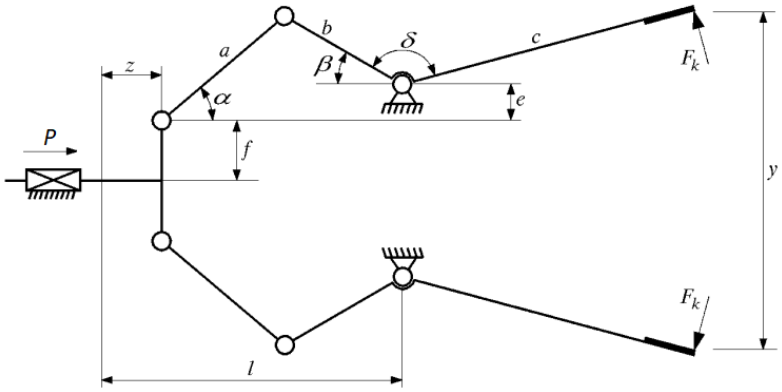
Çalışmanın bölümleri şu şekilde özetlenebilir: İkinci bölüm materyal ve metot bölümüdür. Burada problem tanımı, orijinal ve iyileştirilmiş algoritmadan bahsedilmektedir. Üçüncü bölümde simülasyon verileri ve bulgulara yer verilmiştir. Dördüncü bölüm ise sonuçtur.

2. MATERYAL VE METOD (MATERIALS AND METHODS)

Bu başlıkta robot tutucu problem tanımı, ardından orijinal AOA ve önerilen ÇSAOA algoritmalarından bahsedilmektedir.

2.1. Problem Tanımı (Problem Definition)

Bu çalışmada [16]'de sunulan ve Şekil 1'de gösterilen robot tutucu modeli kullanılmıştır.



Şekil 1. Robot tutucu modeli (The robot gripper model)

Bu tutucu modelinde 7 değişken mevcuttur, bu durumda problem boyutu 7'dir. Denklem 1'de gösterilen bu değişkenler tutucunun fiziksel özelliklerini tanımlar.

$$X = [a, b, c, e, f, l, \delta]^T \quad (1)$$

Burada a, b, c, e, f ve l tutucunun farklı bileşenlerinin boyutlarını temsil etmektedir. Tasarımın bu boyutlara göre şekillendirilmesi gerekmektedir. δ, b ve c uzuvları arasındaki açıdır ve bu açı tutucunun hareket kabiliyetini etkiler. α ve β, b ve c uzuvlarının yatay referans açılarıdır. Bu açılar, tutucunun nesneleri hangi açılarda yakalayabileceğini belirler. P tutucunun eyleyici kuvvetidir ve tutucunun çalışması için gereken itme veya çekme kuvvetini ifade eder. Tutucunun bir nesneyi kavrama veya taşıma kapasitesi üzerinde önemli bir etkisi vardır. F_k ise nesneye uygulanan kavrama kuvvetidir ve Denklem 2'deki gibi hesaplanır.

$$F_k = \frac{P.b.sin(\alpha+\beta)}{2.c.cos(\alpha)} \quad (2)$$

Bu eşitlikteki parametreler Denklem 3-6'da gösterilmiştir.

$$\alpha = cos^{-1}\left(\frac{a^2+g^2-b^2}{2ag}\right) + \phi \quad (3)$$

$$\alpha = cos^{-1}\left(\frac{a^2+g^2-b^2}{2ag}\right) + \phi \quad (4)$$

$$g = \sqrt{e^2 + (l-z)^2} \quad (5)$$

$$\phi = tan^{-1}\left(\frac{e}{l-z}\right) \quad (6)$$

Bu çalışmada Şekil 1'deki modelin ilk konfigürasyonuna odaklanılmıştır. Bu konfigürasyonun hedefi, tutulacak nesneyi sabit ve sıkı bir şekilde kavramaktır. Bu da, kavrama kuvvetinin (F_k) maksimum ve minimum değerleri arasındaki farkın minimizasyonu ile sağlanır. Bu minimizasyon Denklem 7'deki uygunluk fonksiyonu ile ifade edilir [17].

$$min \;\; f(x) = \max_z F_k(x,z) - \min_z F_k(x,z) \quad (7)$$

Ayrıca bu modelde tasarım değişkeniyle ilişkili ve Denklem 8-14'te gösterilen 7 eşitsizlik kısıtlaması vardır.

$$g_1(x) = Y_{min} - y(x, Z_{max}) \geq 0 \quad (8)$$

$$g_2(x) = y(x, Z_{max}) \geq 0 \quad (9)$$

$$g_3(x) = y(x, 0) - Y_{max} \geq 0 \quad (10)$$

$$g_4(x) = Y_G - y(x, 0) \geq 0 \quad (11)$$

$$g_5(x) = (a+b)^2 - l^2 - e^2 \geq 0 \quad (12)$$

$$g_6(x) = (l - Z_{max})^2 + (a-e)^2 - b^2 \geq 0 \quad (13)$$

$$g_7(x) = l - Z_{max} \geq 0 \quad (14)$$

Burada genel $y(x,z)$ fonksiyonu tutucunun iki ucu arasındaki mesafeyi temsil eder ve Denklem 15'teki gibi hesaplanır.

$$y(x,z) = 2.\left(e + f + c.sin(\beta+\delta)\right) \quad (15)$$

Y_{min} ve Y_{max} tutulacak nesnenin minimum ve maksimum boyutu, Z_{max} eyleyicinin maksimum

yer değiştirmesi, Y_G ise tutucunun maksimum yer değiştirme aralığıdır.

2.2. Orijinal Aritmetik Optimizasyon Algoritması (Original Arithmetic Optimization Algorithm)

Aritmetik Optimizasyon Algoritması (AOA), 2021 yılında Abualigah vd. tarafından geliştirilen matematik tabanlı bir optimizasyon algoritmadır [18]. Bu algoritma, matematiksel hesaplamaların temelini oluşturan aritmetik işlemlerin dağılım özelliklerini taklit etmek amacıyla tasarlanmıştır. AOA, aritmetik işlemlerine dayalı olarak çalışan bir yapıya sahiptir ve bu işlemler toplama, çıkarma, çarpma ve bölme işlemlerini içerir. Bu algoritma, birçok metasezgisel algoritma gibi, arama sürecini keşif ve kullanım olmak üzere iki temel aşamada gerçekleştirir. Keşif aşamasında, aday çözümlerin pozisyonları çarpma ve bölme işlemleri temel alınarak güncellenir. Bu, farklı ve çeşitli çözüm adayları üretmeye ve arama uzayını keşfetmeye yardımcı olur. Kullanım aşaması ise daha iyi çözümlere ulaşmak ve bu çözümleri iyileştirmek için toplama ve çıkartma işlemleri ile ilgilenir. Algoritmanın işleyişi şu şekildedir: Öncelikle rastgele bir başlangıç popülasyonu Denklem 16'daki gibi üretilir.

$$x = LB + (UB - LB).rand(p, d) \quad (16)$$

Burada x başlangıç popülasyonu, LB ve UB problemin sınır değerleri, p popülasyon sayısı ve d problem boyutudur. İteratif süreç başlamadan önce bu başlangıç popülasyonu uygunluk fonksiyonunda değerlendirilir. İteratif süreç başladığında ise matematik optimizasyon hızlandırıcı (math optimizer accelerated, MOA) ve matematik optimizasyon olasılığı (math optimizer probability, MOP) olmak üzere iki fonksiyon sırasıyla Denklem 17 ve 18'deki gibi hesaplanır.

$$MOA(t) = MOA_{min} + t\left(\frac{MOA_{max} - MOA_{min}}{T}\right) \quad (17)$$

$$MOP(t) = 1 - \frac{t^{1/\alpha}}{T^{1/\alpha}} \quad (18)$$

Burada MOA_{min} ve MOA_{max} hızlandırıcı fonksiyonun sınır değerleri, t mevcut iterasyon, T toplam iterasyon sayısı ve α kullanım doğruluğunu tanımlayan bir parametredir. MOA ve MOP fonksiyonları hesaplandıktan sonra her aday çözüm ve her boyut için [0, 1] aralığında rassal bir sayı (r_1) belirlenir. Bu sayısı mevcut iterasyondaki MOA değerinden büyükse keşif aşaması devreye girer ve aday çözümler Denklem 19'daki gibi güncellenir.

$$x_{i,j}^{t+1} = \begin{cases} x_{best,j}^{t} \,/\, (MOP + \epsilon).\left((UB_j - LB_j).\mu + LB_j\right), & r_2 < 0.5 \\ x_{best,j}^{t} \,.\, MOP.\left((UB_j - LB_j).\mu + LB_j\right), & r_2 \geq 0.5 \end{cases} \quad (19)$$

Burada $x_{i,j}^{t+1}$ i. çözümün j. boyutunun güncel değeri, $x_{best,j}^{t}$ j. boyutun mevcut optimum değeri, ϵ küçük bir tamsayı değeri, μ arama sürecini ayarlayan bir kontrol parametresi, r_2 [0, 1] aralığında rassal bir sayı, UB_j ve LB_j ise problemin j. boyutunun sınır değerleridir. Eğer r_1 rassal sayısı mevcut iterasyondaki MOA değerinden küçükse kullanım aşaması devreye girer ve aday çözümler Denklem 20'deki gibi güncellenir.

$$x_{i,j}^{t+1} = \begin{cases} x_{best,j}^{t} - (MOP + \epsilon).\left((UB_j - LB_j).\mu + LB_j\right), & r_3 < 0.5 \\ x_{best,j}^{t} + MOP.\left((UB_j - LB_j).\mu + LB_j\right), & r_3 \geq 0.5 \end{cases} \quad (20)$$

Burada r_3 [0, 1] aralığında rassal bir sayıdır. Güncel aday çözümler uygunluk fonksiyonunda tekrar değerlendirilir ve bu adımlar sonlandırma kriteri sağlanana kadar devam eder [19].

2.3. Çok Stratejili Aritmetik Optimizasyon Algoritması (Multi-Strategy Arithmetic Optimization Algorithm)

2.3.1. Orijinal güncelleme stratejilerinin modifiyesi (Modification of original update strategies)

Bu çalışmada Çok Stratejili Aritmetik Optimizasyon Algoritması (ÇSAOA) adında yeni bir algoritma önerilmiştir. Bu algoritmada orijinal AOA'nın arama performansını iyileştirmek için hem keşif hem de kullanım aşamasının dengesine odaklanılmış ve buna göre güncelleme stratejisi modifiye edilmiştir. Böylece orijinal algoritmanın güncelleme stratejisinde sadece optimum değere yakınsama sağlanırken, önerilen algoritmada optimum değere ek olarak hem mevcut değer de hesaba katılmış hem de rassallık artırılmıştır. Bu algoritmada r_1 rassal sayısı mevcut iterasyondaki MOA değerinden büyükse keşif aşaması devreye girer ve çözümler Denklem 21'deki gibi güncellenir.

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} \,/\, (MOP + \epsilon).\left((UB_j - LB_j).\mu + LB_j\right) + rand(0,1).\left(x_{best,j}^{t} - x_{i,j}^{t}\right), & r_2 < 0.5 \\ x_{i,j}^{t} \,.\, MOP.\left((UB_j - LB_j).\mu + LB_j\right) + rand(0,1).\left(x_{best,j}^{t} - x_{i,j}^{t}\right), & r_2 \geq 0.5 \end{cases} \quad (21)$$

Burada $x_{i,j}^{t+1}$ i. çözümün j. boyutunun güncel değeri, $x_{i,j}^{t}$ i. çözümün j. boyutunun mevcut değeri, $x_{best,j}^{t}$ j. boyutun mevcut optimum değeridir. Eğer r_1 rassal sayısı mevcut iterasyondaki MOA değerinden küçükse kullanım aşaması devreye girer ve çözümler Denklem 22'deki gibi güncellenir.

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} - (MOP + \epsilon).\left((UB_j - LB_j).\mu + LB_j\right) + rand(0,1).\left(x_{best,j}^{t} - x_{i,j}^{t}\right), & r_2 < 0.5 \\ x_{i,j}^{t} + MOP.\left((UB_j - LB_j).\mu + LB_j\right) + rand(0,1).\left(x_{best,j}^{t} - x_{i,j}^{t}\right), & r_2 \geq 0.5 \end{cases} \quad (22)$$

2.3.2. Çoklu strateji konsepti (Multi-Strategy concept)

Çoğu metasezgisel algoritma bir popülasyon içindeki aday çözümleri aynı güncelleme stratejisiyle günceller ve sürecin sonunda en iyi

çözümü seçer. Ancak, tek tip bir güncelleme stratejisi kullanmak algoritmanın arama performansını sınırlayabilir [20]. AOA da bu algoritmalardan biridir ve bu problemin üstesinden gelmek için algoritmaya çoklu strateji konsepti eklenmiştir. Bu konsept, popülasyondaki her aday çözümün kendini güncellerken farklı stratejileri kullanmasına olanak sağlar. Bu çalışma için eklenen güncelleme stratejisi Denklem 23'te gösterilmektedir.

$$x_{i,j}^{t+1} = \begin{cases} x_{R_1,j}^{t} + F\left(x_{R_2,j}^{t} - x_{R_3,j}^{t}\right), & rand < CR \\ x_{i,j}^{t}, & rand \geq CR \end{cases} \quad (23)$$

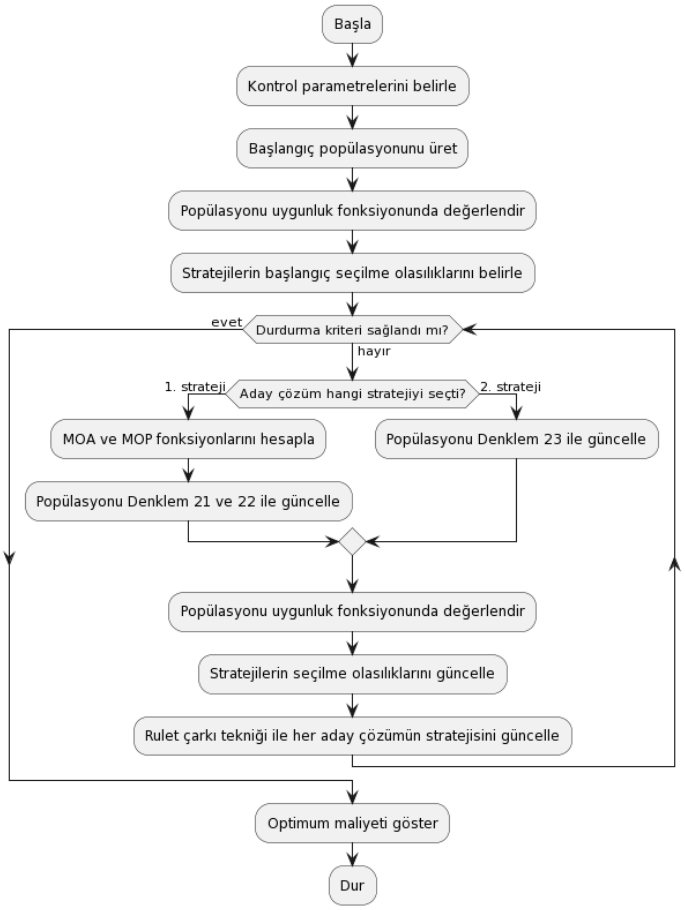
Burada, $x_{R_1,j}^{t}$, $x_{R_2,j}^{t}$ ve $x_{R_3,j}^{t}$ popülasyondaki i. çözümden farklı rastgele aday çözümlerin j. boyutu, F skala faktörü ve CR ise çaprazlama oranıdır. Böylece iki güncelleme stratejisinin olduğu bir strateji havuzu oluşmuştur. Başlangıçta her stratejinin seçilme olasılığı eşit olup %50 olarak ayarlanır. İteratif süreçte her aday çözüm rulet çarkı tekniğini kullanarak kendi stratejisini seçer. Böylece aday çözümler kendi stratejileri kullanılarak kendilerini günceller ve bu stratejilerin seçim sayaçları da güncellenir. Güncelleme sonrası yeni popülasyon uygunluk fonksiyonunda değerlendirilir ve her stratejinin olasılığı Denklem 24'teki gibi hesaplanır.

$$p_k^{t+1} = \frac{c_k^t}{\sum_{i=1}^{2} c_i^t}, k \in \{1,2\} \quad (24)$$

Burada p_k^{t+1} k. stratejinin güncel seçilme olasılığı ve c_k^t k. stratejinin mevcut seçim sayacıdır. Bu olasılıklara göre aday çözümler yine rulet çarkı tekniğini kullanarak kendi stratejilerini tekrar seçer, bu şekilde iteratif süreç devam eder. Böylece algoritma hangi stratejinin daha iyi sonuçlar ürettiğini öğrenir ve bu stratejiyi daha sık kullanır. Bu yaklaşım, farklı stratejilerin adaptasyonunu içerdiği için, farklı optimizasyon problemlerine daha iyi uyum sağlayabilir ve daha tatmin edici sonuçlar üretebilir. Bu sayede, optimizasyon işlemlerinde daha esnek ve etkili bir yol sunar. ÇSAOA'nın temel adımları Algoritma 1'de, akış diyagramı ise Şekil 2'de gösterilmiştir.

Algoritma 1: ÇSAOA'nın temel adımları (Basic steps of MSAOA)

Algoritmanın kontrol parametrelerini belirle
Denklem (16)'yı kullanarak başlangıç popülasyonunu üret
Popülasyonu bir uygunluk fonksiyonunda değerlendir ve optimum maliyeti belirle
Stratejilerin başlangıç seçilme olasılıklarını belirle
Rulet çarkı tekniği ile her aday çözümün stratejisini belirle
while (durdurma kriteri sağlanana kadar)
    Aday çözüm 1. stratejiyi seçmiş ise
        Denklem (17) ve (18)'i kullanarak MOA ve MOP fonksiyonlarını hesapla
        MOA değerine göre Denklem (21) ve (22)'yi kullanarak aday çözümü güncelle
    Aday çözüm 2. stratejiyi seçmiş ise
        Denklem (23)'ü kullanarak aday çözümü güncelle
    Güncel popülasyonu uygunluk fonksiyonunda değerlendir ve optimum maliyeti güncelle
    Denklem (24)'ü kullanarak stratejilerin seçilme olasılıklarını güncelle
    Rulet çarkı tekniği ile her aday çözümün stratejisini güncelle
end while
Optimum maliyeti göster



Şekil 2. ÇSAOA'nın akış diyagramı (Flow diagram of MSAOA)

3. BULGULAR VE TARTIŞMA (RESULTS AND DISCUSSION)

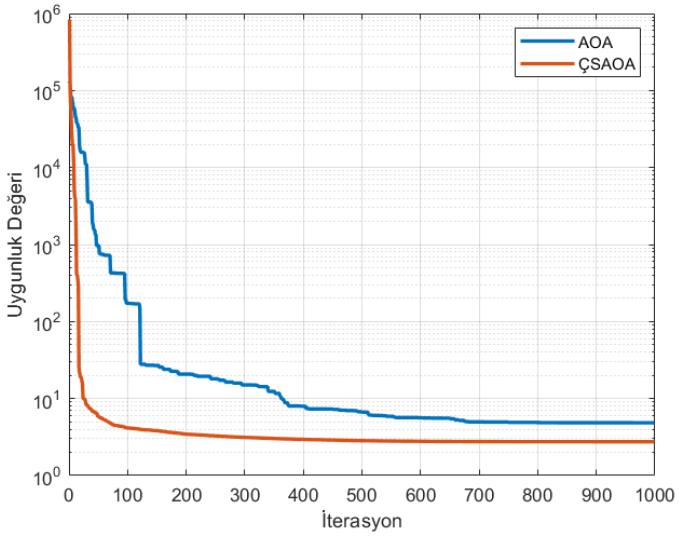
Bu çalışma MATLAB programlama dilinde kodlanarak gerçekleştirilmiş ve Windows 10 işletim sistemi, INTEL CORE i7 işlemcisi ve 16 GB RAM'e sahip bir bilgisayar kullanılmıştır. AOA ve ÇSAOA için popülasyon ve iterasyon sayıları sırasıyla 30 ve 1000 olarak belirlenmiştir. Hızlandırıcı fonksiyonun sınır değerleri [0.2, 1] iken, α ve μ sırasıyla 5 ve 0.5'tir. ÇSAOA için F ve CR değerleri sırasıyla 0.8 ve 0.95'tir. Problemin global değeri $f(x^*)$ 2.5287'dir. Algoritmalar 30 koşma ile çalıştırılmış ve algoritmalarının performansı ortalama uygunluk değerleri ve ortalama hesaplama süreleri açısından karşılaştırılmıştır. Çalışma sonucunca elde edilen bulgular Tablo 1'de, iyileşme oranları Tablo 2'de ve ortalama yakınsama eğrileri Şekil 3'te gösterilmektedir.

Tablo 1. AOA ve ÇSAOA'nın uygunluk değerleri ve ortalama hesaplama süreleri (Fitness and average calculation times of AOA and MSAOA)

Algoritma	Uygunluk Değeri				Ortalama Hesaplama Süresi (sn)
	Ortalama	Minimum	Maksimum	Standart Sapma	
AOA	4.8426	4.0795	6.5807	0.6790	22.9014
ÇSAOA	2.7507	2.5438	4.2893	0.5506	17.9213

Tablo 2. Ortalama uygunluk değeri ve ortalama hesaplama süresi açısından iyileşme oranları (Improvement rates in terms of average fitness value and average calculation time)

Kriter	İyileşme Oranları (%)
Ortalama Uygunluk Değeri	43.39
Ortalama Hesaplama Süresi	21.73



Şekil 3. AOA ve ÇSAOA'nın ortalama yakınsama eğrileri (Average convergence curves of AOA and MSAOA)

Tablo 1 ve 2, ÇSAOA'nın daha düşük uygunluk değerlerine ulaşarak problemi daha etkili bir şekilde optimize ettiğini göstermektedir. Daha düşük uygunluk değeri, robot tutucu probleminde daha verimli çözümler elde edildiği anlamına gelir. ÇSAOA aynı zamanda sonuçları daha hızlı üretmiştir. Genel olarak ÇSAOA'nın hem uygunluk değeri performansında hem de hesaplama süresinde belirgin bir iyileşme sağladığı görülmektedir. Şekil 3'teki yakınsama eğrisi de bu sonuçları desteklemektedir ve önerilen algoritmanın daha hızlı yakınsadığı söylenebilir. Bulgulara göre AOA ve ÇSAOA'nın optimum tasarım değişkenleri Tablo 3'te gösterilmektedir.

Tablo 3'e göre, minimum uygunluk değeri açısından önerilen algoritmanın yine daha üstün bir sonuçlar verdiği görülmektedir. Bu uygunluk değeri global değere çok yakındır. Bu nedenle önerilen algoritmayla elde edilen tasarım değişkenleri ile optimuma çok yakın bir robot tutucu tasarımı gerçekleştirilebilir. Ayrıca, önerilen algoritma literatürdeki benzer algoritmalarla ortalama uygunluk değeri açısından da kıyaslanmıştır. Bu kıyaslama Tablo 4'te gösterilmektedir.

Tablo 4'te görüldüğü gibi en performanslı algoritma ÇSAOA'dır. TLBO algoritması ortalama uygunluk değeri açısından en kötü sonuca sahipken, AOS ve MVPA algoritmaları daha düşük ve önerilen algoritmaya yakın sonuçlar üretmiştir. JAYA, TLOCTO ve sCMAgES algoritmaları ise TLBO'ya yakın sonuçlar elde etmiştir. Genel olarak bu tablo da önerilen algoritmanın etkinliğini göstermektedir.

Tablo 3. AOA ve ÇSAOA'nın optimum tasarım değişkenleri (Optimum design variables of AOA and MSAOA)

Algoritma	$f(x)$	a	b	c	e	f	l	δ
AOA	4.0795	150.00	141.75	200.00	0.00	150.00	182.84	2.60
ÇSAOA	2.5438	150.00	149.88	200.00	0.00	150.00	100.94	2.29

Tablo 4. Literatürdeki güncel algoritmalarla ortalama uygunluk değeri kıyaslaması (Average fitness value comparison with state-of-the-art algorithms in the literature)

Algoritma	**Ortalama Uygunluk Değeri**
TLBO [21] (2011)	4.9377
JAYA [22] (2017)	4.9376
sCMAgES [23] (2020)	3.3495
AOS [24] (2021)	2.7917
MVPA [25] (2021)	2.8100
TLOCTO [26] (2023)	3.0924
ÇSAOA	2.7507

4. SONUÇ (CONCLUSIONS)

Endüstriyel robotik sistemlerde robot tutucuların nesneleri zarar vermeden etkili bir şekilde tutabilmesi gerekir. Bundan dolayı, son yıllarda robot tutucularının tasarım optimizasyonu ilgi çeken bir araştırma konusu haline gelmiştir. Bu çalışmada bu tasarım problemi için aritmetik optimizasyon algoritmasına dayalı yeni bir model önerilmiştir. Önerilen modelde orijinal AOA'nın güncelleme mekanizması modifiye edilmiş ve çoklu strateji konsepti eklenerek arama performansı geliştirilmiştir. Önerilen algoritma robot tutucu problemine uygulandığında hem orijinal algoritmaya göre kayda değer iyileşmeler sağlandığı görülmüş, hem de diğer benzer algoritmalarla karşılaştırıldığında daha iyi performans göstermiştir. Bu bulgular, endüstriyel robotik sistemlerde kullanılan robot tutucularının tasarımında ÇSAOA'nın etkin bir şekilde kullanılmasının, nesnelerin zarar görmeden tutulabilmesi için önemli bir adım olduğunu göstermektedir. Bu çalışma, endüstriyel robotik teknolojilerinin geliştirilmesi ve güçlendirilmesi için değerli bir katkı sağlamaktadır.

TEŞEKKÜR (ACKNOWLEDGMENTS)

Bu çalışma Erciyes Üniversitesi Bilimsel Araştırma Projeleri programı kapsamında FDK-2021-11472 proje numarası ile desteklenmiştir.

ETİK STANDARTLARIN BEYANI (DECLARATION OF ETHICAL STANDARDS)

Bu makalenin yazarı çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler.

The author of this article declares that the materials and methods they use in their work do not require ethical committee approval and/or legal-specific permission.

YAZARLARIN KATKILARI (AUTHORS' CONTRIBUTIONS)

***Mustafa Yusuf YILDIRIM*:** Yazılım, görselleştirme, araştırma, kaynaklar, veri iyileştirme ve yazma (orijinal taslak).

Software, visualization, research, references, data curation, writing (original draft).

***Rüştü AKAY*:** Denetim, kavramsallaştırma, metodoloji, yazma (gözden geçirme) ve düzenleme.

Supervision, conceptualization, methodology, writing (reviewing) and editing.

ÇIKAR ÇATIŞMASI (CONFLICT OF INTEREST)

Bu çalışmada herhangi bir çıkar çatışması yoktur.

There is no conflict of interest in this study.

KAYNAKLAR (REFERENCES)

[1] Saravanan, R., Ramabalan, S., Ebenezer, N. G. R., Dharmaraja, C. Evolutionary multi criteria design optimization of robot grippers. Applied Soft Computing. 2009; 9(1): 159-172.

[2] Avder, A. Robot tutucuların optimum tasarımı için çok amaçlı hibrit bir yöntem önerisi, Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, Ankara. 2019.

[3] Yıldız, B. S., Pholdee, N., Bureerat, S., Yıldız, A. R., Sait, S. M. Robust design of a robot gripper mechanism using new hybrid grasshopper optimization algorithm. Expert Systems. 2021; 38(3): e12666.

[4] Rao, R. V., Waghmare, G. Design optimization of robot grippers using teaching-learning-based optimization algorithm. Advanced Robotics. 2015; 29(6): 431-447.

[5] Datta, R., Pradhan, S., Bhattacharya, B. Analysis and design optimization of a robotic gripper using multiobjective genetic algorithm. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2015; 46(1): 16-26.

[6] Mahanta, G. B., Rout, A., B.B.V.L. D., Biswal, B. B. An improved multi-objective antlion optimization algorithm for the optimal design of the robotic gripper. Journal of Experimental & Theoretical Artificial Intelligence. 2020; 32(2): 309-338.

[7] Dong, H., Asadi, E., Qiu, C., Dai, J., Chen, I. M. Geometric design optimization of an under-actuated tendon-driven robotic gripper. Robotics and Computer-Integrated Manufacturing. 2018; 50: 80-89.

[8] Zhong, J., Yuan, X., Du, B., Hu, G., Zhao, C. An lévy flight based honey badger algorithm for robot gripper problem. 2022 7th International Conference on Image, Vision and Computing (ICIVC). 2022; 901-905.

[9] Dörterler, M., Atila, Ü., Durgut, R., Şahin, İ. Analyzing the performances of evolutionary multi-objective optimizers on design optimization of robot gripper configurations. Turkish Journal of Electrical Engineering and Computer Sciences. 2021; 29(1): 349-369.

[10] Hassan, A., Abomoharam, M. Modeling and design optimization of a robot gripper mechanism. Robotics and Computer-Integrated Manufacturing. 2017; 46: 94-103.

[11] Datta, R., Deb, K. Optimizing and deciphering design principles of robot gripper configurations using an evolutionary multi-objective optimization method. KanGAL Report 2011002. 2011; 1-10.
[12] Jia, J., Sun, X., Liu, T., Tang, J., Wang, J., Hu, X. Structural optimization design of dual robot gripper unloading device based on intelligent optimization algorithms and generative design. Sensors. 2023; 23(19): 8298.
[13] Wang, R., Zhang, X., Zhu, B., Zhang, H., Chen, B., Wang, H. Topology optimization of a cable-driven soft robotic gripper. Structural and Multidisciplinary Optimization. 2020; 62: 2749-2763.
[14] Liu, C. H., Chen, T. L., Chiu, C. H., Hsu, M. C., Chen, Y., Pai, T. Y., Chiang, Y. P. Optimal design of a soft robotic gripper for grasping unknown objects. Soft Robotics. 2018; 5(4): 452-465.
[15] Sun, Y., Liu, Y., Pancheri, F., Lueth, T. C. Larg: A lightweight robotic gripper with 3-d topology optimized adaptive fingers. IEEE/ASME Transactions on Mechatronics. 2022; 27(4): 2026-2034.
[16] Osyczka, A., Krenich, S. Some methods for multicriteria design optimization using evolutionary algorithms. Journal of Theoretical and Applied Mechanics. 2004; 42(3): 565-584.
[17] Kumar, A., Wu, G., Ali, M. Z., Mallipeddi, R., Suganthan, P. N., Das, S. A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evolutionary Computation. 2020; 56: 100693.
[18] Hu, G., Zhong, J., Du, B., Wei, G. An enhanced hybrid arithmetic optimization algorithm for engineering applications. Computer Methods in Applied Mechanics and Engineering. 2022; 394: 114901.
[19] Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A. H. The arithmetic optimization algorithm. Computer Methods in Applied Mechanics and Engineering. 2021; 376: 113609.
[20] Akay, R., Yildirim, M. Y. Multi-strategy and self-adaptive differential sine-cosine algorithm for multi-robot path planning. Expert Systems with Applications. 2023; 120849.
[21] Rao, R. V., Savsani, V. J., Vakharia, D. P. Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Computer-Aided Design. 2011; 43(3): 303-315.
[22] Rao, R. V., Waghmare, G. G. A new optimization algorithm for solving complex constrained design optimization problems. Engineering Optimization. 2017; 49(1): 60-83.
[23] Kumar, A., Das, S., Zelinka, I. A modified covariance matrix adaptation evolution strategy for real-world constrained optimization problems. Genetic and Evolutionary Computation Conference Companion. 2020; 11-12.
[24] Azizi, M., Talatahari, S., Giaralis, A. Optimization of engineering design problems using atomic orbital search algorithm. IEEE Access. 2021; 9: 102497-102519.
[25] Uymaz, S. A. Evaluation of the most valuable player algorithm for solving real-world constrained optimization problems. Bilişim Teknolojileri Dergisi. 2021; 14(4): 345-353.
[26] Wu, X., Li, S., Wu, F., Jiang, X. Teaching-learning optimization algorithm based on the cadre-mass relationship with tutor mechanism for solving complex optimization problems. Biomimetics. 2023; 8(6): 462.