Journal of the Faculty of Engineering and Architecture of Gazi University 38:4 (2023) 2555-2563

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Optimal control of a class of nonlinear systems using Euler-Lagrange back integration method

Orhan Aksoy[1,2]*, Erkan Zergeroglu[1]

[1]Naval Combat Management Technologies Center, HAVELSAN A.S., Pendik, Istanbul, Türkiye
[2]Computer Engineering Department, Faculty of Engineering, Gebze Technical University, Gebze, Kocaeli, Türkiye

Highlights:

- Optimal control of nonlinear, linearizable input affine systems
- Euler-Lagrange back integration method
- Tessellation of optimal paths in the state-space

Keywords:

- Nonlinear optimal control
- HJB Equation
- Input affine nonlinear systems
- Breadth-first search
- Euler-Lagrange back integration method
- Tessellation

Article Info:

Research Article
Received: 06.02.2021
Accepted: 24.12.2022

DOI:

10.17341/gazimmfd.875563

Correspondence:

Author: Orhan Aksoy
e-mail: oaksoy@havelsan.com.tr
phone: +90 533 139 7380

Graphical/Tabular Abstract

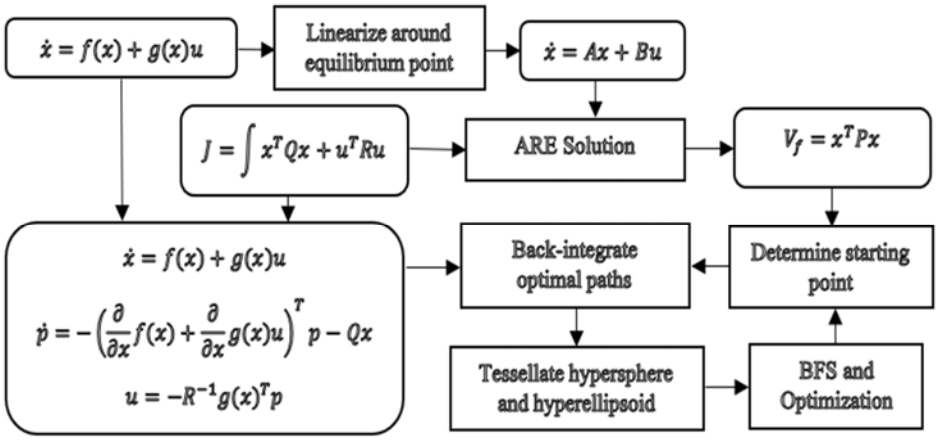


Figure A. Generation of optimal trajectories

Purpose:
The aim of this study is to achieve optimal control of nonlinear systems that are input affine and linearizable.

Theory and Methods:
Euler-Lagrange back integration approach enables generation of optimal trajectories of the system states. To use these paths in a control loop, these optimal paths should span the whole state space. A method has been proposed to accomplish this task for two dimensional systems (Holzhüter) (2004). This study presents a new method to generate such optimal trajectories for systems with dimensions greater than two, and is shown on Figure A. The proposed algorithm is applied in a three-dimensional nonlinear system in a simulation environment.

Results:
The results show that the proposed algorithm succeeds in spanning the state space with sufficient resolution and these paths can be used in a feedback loop. The performance of the controller is compared with an LQR regulator according to time integration of (a) error norm and (b) controller input. The proposed controller had better performance in both measures.

Conclusion:
Proposed algorithm can be used for optimal control of linearizable input affine nonlinear systems. The study shows that the algorithm succeeds in a three-dimensional case, however, the generation of optimal paths is a heavy task even in GPU architectures. The algorithm should be applied on systems with higher degree on higher performance computers, and with improvements aiming to improve efficiency.

Journal of the Faculty of Engineering and Architecture of Gazi University 38:4 (2023) 2555-2563

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Doğrusal olmayan bazı sistem sınıflarının Euler-Lagrange geri integrasyon yöntemi ile optimal denetimi

Orhan Aksoy[1,2]*, Erkan Zergeroglu[1]
[1]Deniz Savaş Yönetim Sistemi Teknolojileri Merkezi (DSYSTM), HAVELSAN A.Ş., Pendik, İstanbul, Türkiye
[2]Gebze Teknik Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Gebze, Kocaeli, Türkiye

ÖNEÇIKANLAR

- Doğrusal olmayan, doğrusallaştırılabilir, girişi doğrusal sistemlerin optimal denetimi
- Euler-Lagrange geri integrasyon yöntemi
- Durum uzayındaki optimal yolların üçgenlenmesi

Makale Bilgileri
Araştırma Makalesi
Geliş: 06.02.2021
Kabul: 24.12.2022

DOI:
10.17341/gazimmfd.875563

Anahtar Kelimeler:
Doğrusal olmayan optimal denetim, HJB denklemi, doğrusal girişli doğrusal olmayan sistemler, genişlik öncelikli arama, Euler-Lagrange geri integrasyon yöntemi, üçgenleme

ÖZ

Bu çalışmada, üç veya daha fazla boyutlu, doğrusallaştırılabilen, doğrusal girişli doğrusal olmayan sistemlerin optimal denetimi için Euler-Lagrange geri integrasyon yönteminin kullanımını geliştiren bir algoritma sunulmuştur. Algoritma, doğrusallaştırılmış sistemin sınırını temsil eden, denge noktasının yakınındaki bir hiper elipsoit üzerinde tekrarlı şekilde başlangıç koşulu seçerek doğrusal olmayan optimal dinamik üzerinde geri integrasyon yöntemini uygulamakta ve optimal yolları oluşturmaktadır. Daha sonra, bu optimal yollar optimal geri beslemede kullanılmaktadır.

Optimal control of a class of nonlinear systems using Euler-Lagrange back integration method

HIGHLIGHTS

- Optimal control of nonlinear, linearizable input affine systems
- Euler-Lagrange back integration method
- Tessellation of optimal paths in the state-space

Article Info
Research Article
Received: 06.02.2021
Accepted: 24.12.2022

DOI:
10.17341/gazimmfd.875563

Keywords:
Nonlinear optimal control, HJB equation, input affine nonlinear systems, breadth first search, Euler-Lagrange back integration method, tessellation

ABSTRACT

In this study, an algorithm is presented that aims to extend the use of Euler-Lagrange back integration method to optimal control of linearizable, affine-in-the-input nonlinear systems having more than two states. The algorithm generates optimal paths by selecting a starting point iteratively on the ellipsoid close to the equilibrium point, representing the optimal borderline of the linearized system, and backward integrating the optimal nonlinear dynamics. The generated paths are then utilized in the optimal control loop.

*Sorumlu Yazar/Yazarlar / Corresponding Author/Authors : *oaksoy@havelsan.com.tr, ezerger@bilmuh.gtu.edu.tr / Tel: +90 533 139 7380

1. Giriş (Introduction)

Doğrusal olmayan sistemlerin optimal denetimi, asimptotik kararlılığı kesin şekilde sağlayan en iyi denetleyici girişini bulmanın yanında önceden tanımlanmış bir performans metriğini de mümkün olan en küçük değerde tutmayı hedeflediği için oldukça zor bir görevdir. Bu metrik, genelde hem sistem durumlarının hem de girişin ikinci dereceden fonksiyonlarıdır. Böyle bir girişin tasarımı sistem dinamiğine karşılık gelen Hamilton-Jacobi-Bellman (HJB) denkleminin çözümü ile yapılabilir [1, 2]. Sistem dinamiğinin doğrusal olduğu özel durumlarda HJB denklemi analitik olarak çözülebilen Algebraic Riccati Equation (ARE) denklemine dönüşür; ancak doğrusal olmayan sistemlerde çözüm bir kısmi diferansiyel denklemin çözümüne bağlı olduğu için HJB denkleminin analitik bir çözümü mümkün olmayabilir [3]. Bu sebeple, HJB denkleminin çözümü için yaklaşım yöntemleri pek çok çalışmada uygulanmıştır. [4]'te HJB çözümünde kullanılan maliyet fonksiyonları yapay sinir ağları ile yaklaşık olarak temsil edilmiştir. [5] ve [6]'da HJB denklemlerinin yaklaşık çözümünde, değer, kritik ve tanımlayıcı yapay sinir ağları bileşenleri kullanılmıştır. [7]'de optimal denetleyicilerde Reinforcement Learning yaklaşımının kullanımı konusunda bir derleme sunulmuştur. [8]'de Adaptive Dynamic Programming yönteminin HJB denkleminin çözümünde kullanımı gerçekleştirilmiştir. Bu yaklaşım yöntemleri optimal değer fonksiyonunu temsil eden yapay sinir ağları kullanmakta olup sistem dinamiğinin optimal dinamiğe yakınsadığını göstermektedir.

[3]'te Euler-Lagrange modeli ile temsil edilen doğrusal olmayan sistemler için bir denetleyici tasarlanmış, modeli kullanarak doğrudan HJB denklemini çözmek yerine, denetleyicinin zaman içerisinde HJB denkleminin çözümü ile elde edilecek optimal denetleyiciye yakınsadığı gösterilmiştir. [4]'te doğrusal olmayan bir denetleyici yapısı öngörülerek denetleyici parametreleri iterasyonlarla optimal denetleyicinin parametrelerine yakınsatılmıştır. [5]'te önceden denetleyici yapısının öngörülmesine ihtiyaç olmadan, HJB denkleminin çözümü olan değer fonksiyonu, iterasyonlarla optimal denetleyici tasarımına yakınsatılmıştır (*Policy Iteration*). [6]'da aynı yaklaşım, belirsiz Euler-Lagrange sistemleri için uygulanmıştır. Bu çözümde tasarlanan denetleyicide, sistem parametreleri bilinmeden, değer fonksiyonunun optimal denetleyicinin değer fonksiyonuna yakınsamasını sağlanmıştır. Doğrusallaştırılmış halleri denetlenebilir olan doğrusal olmayan sistemler için optimal dinamik denklemleri zamanda geriye doğru çözülerek optimal denetleyici tasarlanabilir [9-12]. Bu yöntemde zamanda geriye doğru çözümün başlangıç noktası, doğrusallaştırılmış sistemin denge noktası yakınlarındaki bir hiper elipsoit yüzeyidir. Bu yüzey, $V_f = x^T P x$ denklemi ile temsil edilmekte olup buradaki P matrisi ARE denkleminin çözümü, V_f değeri de bu yüzey üzerinde optimal değer fonksiyonunun değeridir. Bu başlangıç noktaları uygun şekilde ve yeterli sayıda seçilerek durum uzayı yeterince yoğun optimal yollarla taranabilir. Bu optimal yollar üzerindeki her bir nokta, optimal durumu ve bu duruma karşılık gelecek optimal girişi barındırdığından bu veri kümesi üzerinde interpolasyon yaparak bir optimal geri besleme tasarlanması mümkün olur. Bu yöntemin uygulanmasındaki en büyük zorluk başlangıç noktalarının hiper elipsoit üzerinde seçileceği konumların belirlenmesidir. Bu noktaların yüzey üzerinde eşit aralıklarla seçilmesi, dinamiğin doğrusal olmayan doğası sebebiyle optimal yolların durum uzayının belirli bölgelerine yoğunlaşmasına ve dolayısıyla düzgün taranamamasına sebep olur. Bu yüzden başlangıç noktalarının seçilmesi için bir yönteme ihtiyaç vardır. İki boyutlu sistemler için [9] ve [10]'da bir yöntem önerilmiştir: İki boyutlu sistemler için durum uzayı iki boyutlu bir yüzeydir. Dolayısıyla başlangıç noktalarının seçileceği yerler bir elips üzerindedir. Öncelikle bu elips bir ξ açısıyla $e(\xi) = e_1\cos(\xi) + e_2\sin(\xi)$ şeklinde parametrik hale getirilir. Algoritma, öncelikle elipsin birbirine en uzak iki noktasını seçerek bu noktaları başlangıç koşulu olacak şekilde geriye integrasyon yöntemini uygular. İntegrasyonlar, belirli bir maliyet değerine varınca durdurulur. Daha sonra, her integrasyonun sonunda, ulaşılan son noktalar arasında birbirine en uzak iki tanesi seçilip bu integrasyonların başlangıç açıları kullanılarak yeni başlangıç açısı bunların ortalaması alınarak hesaplanır. Bu işlem belirli bir çözünürlük değeri elde edilinceye kadar tekrarlanır. Bu algoritmanın çalışma prensibi şudur: ξ_1 ve ξ_2 başlangıç noktalarından yola çıkıldığında geriye integrasyon sonunda ξ_1' ve ξ_2' noktalarına varılıyorsa, $(\xi_1 + \xi_1)/2$ noktasından yola çıkıldığında mutlaka ξ_1' ve ξ_2' arasında bir noktaya varılır ve böylece her iterasyon sonunda durum uzayının sınırındaki en uzak mesafe mutlaka kısalır. Durum vektörünün boyutunun ikiden fazla olduğu durumlarda, başlangıç noktaları bir elips değil bir hiper elipsoit oluşturur. Geriye integrasyon sonunda varılan noktalar da bir hiper yüzey oluştururlar. İki boyutlu uzaydaki algoritmaya benzer şekilde şöyle bir yöntem düşünülebilir: Bu hiper yüzey üçgenlere ayrılıp en büyük üçgen seçilir; bu üçgeni oluşturan optimal yolların başlangıç noktalarının ortası hesaplanır ve sonraki tekrar bu yeni noktadan başlar. Maalesef, doğrusal olmayan dinamiğin bir sonucu olarak, bu yeni noktadan başlayan optimal yolun durum uzayının sınırındaki hiper yüzeyde seçilen üçgenin içinde sonlanması gerekmez. Bu sebeple bu yaklaşım yalnızca iki boyutlu problemler için geçerlidir.

Bu çalışmada, ikiden fazla boyutlu sistemler için durum uzayını optimal yollarla kaplayacak şekilde başlangıç koşullarını seçen yeni bir algoritma sunulmuştur. Bu algoritma, tekrarlı şekilde, $V_f = x^T P x$ hiper elipsoit üzerinde bir nokta seçip optimal dinamiği bu noktadan zamanda geriye doğru çözerek durum uzayının sınırının çözünürlüğünü geliştirmektedir. Optimal yollar durum uzayını yeterince yoğun şekilde taradıktan sonra, bu yolları kullanan denetleyici performansı gösterilmiştir. Makalenin bundan sonraki bölümleri şu şekildedir: İkinci bölümde, optimal denetleyici problemi ve Euler-Lagrange Geri İntegrasyon Yaklaşımı sunulmuştur. Üçüncü bölümde, başlangıç noktaları ve optimal yolları oluşturan yeni algoritma gösterilmiştir. Dördüncü bölümde, örnek bir sistem için optimal yolları kullanan denetleyici tasarımı ve simülasyon sonuçları yer almaktadır. Beşinci ve son bölümde ise sonuçlar değerlendirilmiş ve sonraki çalışmalar için öneriler sunulmuştur.

2. Optimal Denetleyici Problemi ve Euler-Lagrange Geri İntegrasyon Yöntemi
(The Optimal Control Problem And Euler-Lagrange Back-Integration Method)

Bu çalışmada, doğrusallaştırılabilen, doğrusal girişli doğrusal olmayan sistemler ele alınmıştır. Bu sistemlerin dinamiği Eş. 1 ile ifade edilir.

$$\dot{x} = f(x) + g(x)u \tag{1}$$

Burada $x \in \mathbb{R}^n$ n boyutlu durumu vektörünü, $u(x) \in \mathbb{R}^m$ denetleyici girişini temsil etmektedir. Dinamikteki $f(x) \in \mathbb{R}^n$ ve $g(x) \in \mathbb{R}^{nxm}$ sisteme ait vektör alanları olup sistemin denge noktası etrafında kontrol edilebilir olduğu varsayılmıştır.

Optimal denetleyici problemi için kullanılacak sonsuz ufuklu (*İng. Infinite horizon*) maliyet fonksiyonu Eş. 2 ile gösterilmiştir.

$$J = \frac{1}{2}\int_0^\infty [x^T Q x + u^T R u]\, dt \tag{2}$$

Burada $Q \in \mathbb{R}^{nxn}$, sistem durumunun sabit, simetrik ve pozitif ağırlık katsayı matrisi, $R \in \mathbb{R}^{mxm}$ ise benzer şekilde girişin sabit, simetrik ve pozitif ağırlık katsayı matrisidir. Değişimler hesabı (*İng. Calculus of variations*) kullanılarak Eş. 2'de tanımlanan maliyeti en aza indirecek giriş için koşullar türetilebilir [1].

$$\dot{x} = f(x) + g(x)u \tag{3}$$

$$\dot{p} = -\left(\frac{\partial}{\partial x} f(x) + \frac{\partial}{\partial x} g(x)u\right)^T p - Qx \tag{4}$$

$$u = -R^{-1} g^T p \tag{5}$$

Burada $p \in \mathbb{R}^n$ yardımcı durum vektörüdür (*İng. costate*). Eş. 5'te gösterilen denklem optimal girişi tanımlamaktadır, ancak belirli bir sistem durumunda başlangıç koşulu bilinmeyen yardımcı durum dinamiğine bağlı olduğu için geri besleme girişi olarak doğrudan kullanılamaz. Ancak, sistem denge noktası etrafında doğrusallaştırıldığında, bu doğrusal sistem için optimal giriş, durum ve yardımcı durum doğrudan hesaplanabilir. Euler-Lagrange geri integrasyon yöntemi bu bilgiyi kullanmaktadır.

Eş. 1'de tanımlanan sistem Eş. 6 gibi doğrusallaştırılabilir.

$$\dot{x} = Ax + Bu \tag{6}$$

Burada, $A \in \mathbb{R}^{nxn}$ ve $B \in \mathbb{R}^{nxm}$ doğrusal sistemin sistem ve giriş matrisleridir ve doğrusal olmayan sistem Eş. 7 gibi denge noktasında ($x = \mathbf{0}$) doğrusallaştırılarak hesaplanır.

$$A = \frac{\partial}{\partial x}(f(x) + g(x)u)|_{x=\mathbf{0}}, B = \frac{\partial}{\partial u}(f(x) + g(x)u)|_{x=\mathbf{0}} \tag{7}$$

Eş. 6'da gösterilen doğrusal sistem için, Eş. 5'te gösterilen giriş $u = -R^{-1}B^T p$ haline dönüşür ve bu optimal giriş Eş. 2'de gösterilen maliyette kullanıldığında Eş. 8' deki optimal değer fonksiyonu elde edilir.

$$V^* = \frac{1}{2} x^T P x \tag{8}$$

Buradaki $P \in \mathbb{R}^{nxn}$ matrisi aşağıdaki ARE denkleminin çözümü ile bulunabilir (Eş. 9).

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{9}$$

Böylece, yardımcı durum vektörünün değeri Eş. 10'daki gibi hesaplanabilir.

$$p = Px \tag{10}$$

Buradan anlaşılacağı üzere, doğrusallaştırılmış sistemin herhangi bir sistem durumu için yardımcı durum vektörü hesaplanabilmektedir. Bu bilgi, denge noktasının yakınlarında, belli bir maliyet değeri için Eş. 8 numaralı denklemle temsil edilen hiper elipsoit üzerinde tüm durum ve yardımcı durum vektör değerlerinin bilindiği anlamına gelir. Euler-Lagrange Geri İntegrasyon yöntemi, bu hiper elipsoit üzerinde noktalar seçerek bu noktalardan başlayıp Eş. 3, Eş. 4 ve Eş. 5 ile temsil edilen optimal dinamiği zamanda geriye doğru çözerek optimal yolları bulmaya dayanır. Böylece, durum-yardımcı durum uzayında yeterince çok sayıda optimal yol üretilerek bu optimal yolların geri besleme çevriminde kullanılması sağlanır.

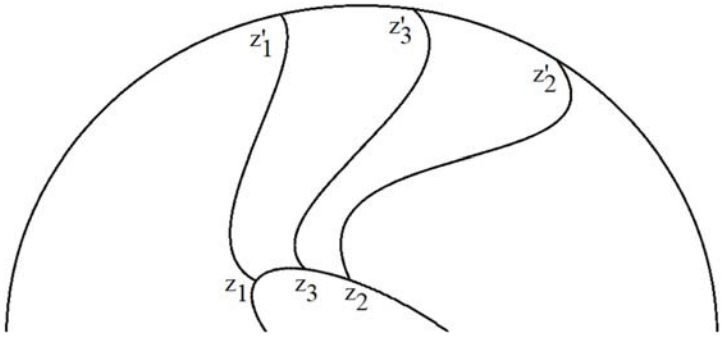
3. Optimal Yollarin Üretilmesi
(The Generation of Optimal Trajectories)

[9]'da iki boyutlu sistemlerde optimal yolların nasıl üretildiği gösterilmiştir. İki boyutlu sistemler üzerindeki bu çalışmada, başlangıç noktaları $V^*(x) = V_f$ denklemiyle ifade edilen elips üzerinde seçilir. Burada V_f, durum uzayında elipsin denge noktasına yeterince yakın olmasını sağlayacak maliyet fonksiyonu değeridir. Daha sonra, elipsin üzerindeki noktalar $\xi \in [0,2\pi]$ açı parametresiyle temsil edilir. Optimal yollar üretilirken her seferinde elips üzerinde bir noktadan başlayarak, Eş. 3, Eş. 4 ve Eş. 5 ile ifade edilen optimal dinamik, $\|x\|$ büyüklüğü bir maksimum değere ulaşıncaya kadar zamanda geriye doğru integrasyon uygulanır. Varılan son nokta, o zamana kadar varılan diğer noktalarla birlikte değerlendirerek sınırdaki en uzun boş bölge tespit edilir. Bu boş bölgenin sınırlarını temsil eden optimal yolların başladığı iki noktanın ortası bulunarak bir sonraki tekrarda geriye integrasyon yöntemi bu noktadan başlatılır. Bu algoritmanın doğru olarak çalışması, şu varsayıma dayanır: Şekil 1'de görüldüğü gibi, elips üzerinde $z_1 < z_2$ olacak şekilde z_1 ve z_2 noktalarından başlayan iki optimal yol z_1' ve z_2' noktalarında sonlanıyorsa, $z_1 < z_3 < z_2$ olacak şekilde seçilen bir z_3'ten başlayan optimal yol da $z_1' < z_3' < z_2'$ koşulunu sağlayan z_3''te sonlanacaktır. Bu varsayım iki boyutlu sistemler için geçerli olsa da daha fazla boyutlu sistemler için kullanılabilir değildir.

Üç ve daha fazla boyutlu sistemler için, başlangıç noktaları $V^*(x) = V_f$ denklemiyle ifade edilen hiper-elipsoit üzerinde seçilir. Sistem *n* boyutlu olduğunda, hiper-elipsoit de *n* boyutlu olur. Bu hiper-elipsoit Eş. 11'deki denklemle ifade edilebilir:

$$V_f = \frac{1}{2} x^T P x \tag{11}$$

Algoritmada bir optimal yol, Eş. 11'de ifade edilen hiper-elipsoit üzerinde başlar ve Eş. 3, Eş. 4 ve Eş. 5 ile ifade edilen optimal dinamik, $\|x\|$ değeri r_m maksimum değerine ulaşıncaya kadar zamanda geriye doğru integrasyon yapılarak optimal yol oluşturulur. Böylece, optimal yolların sınırı, durum uzayında *n* boyutlu bir hiper-küre oluşturur. Her yeni yolun tamamlanmasından sonra küre üzerindeki noktalar üzerinde üçgenleme yapılarak en büyük üçgen seçilir ve bu üçgen kullanılarak bir sonraki seferde hiper-elipsoit üzerinde hangi başlangıç noktasının seçileceği belirlenir. Algoritma 1, Tablo 1'de gösterilmiştir:



Şekil 1. İki boyutlu sistemler için optimal yollar (Optimal paths for two dimensional systems)

Tablo 1. Optimal Yolların Üretilmesi (Generating Optimal Paths)

Algoritma 1	
1:	*procedure* GENTRAJ $(f, g, Q, R, V_f, r_m, res)$
2:	(A, B)←LINMODEL (f, g)
3:	(P)←RICCATI (A, B, Q, R)
4:	$(\dot{x}, \dot{p}, u)$ ←OPTDYN (f, g, Q, R)
5:	(C_i, C_o) ←INITSS $(\dot{x}, \dot{p}, u, P, V_f, r_m)$
6:	$(a_{max}) \leftarrow \infty$
7:	*while* $a_{max} > res$ *do*
8:	(T_{in}) ←TESSELLATE (C_i)
9:	(T_{out}) ←TESSELLATE (C_o)
10:	$(t_{max}, a_{max}) \leftarrow max(T_{out})$
11:	(c_i, c_o)←BESTTRAJ $(t_{max}, C_i, C_o, \dot{x}, \dot{p}, u, r_m)$
12:	$(C_i) \leftarrow C_i \cup c_i$
13:	$(C_o) \leftarrow C_o \cup c_o$
14:	*end while*
15:	*return* $(C_i, C_o, T_{in}, T_{out})$
16:	*end procedure*

Algoritma 1, 2-6 numaralı satırlarda ilklendirme işlemlerini gerçekleştirir. Bunun için öncelikle 2 numaralı satırda, sistemin doğrusal modelini oluşturur. Bu işlemin sonunda Eş. 6'da gösterilen lineer sistemin A ve B matrisleri üretilir. 3 numaralı satırda, ARE denklemi çözülerek verilen Q ve R matrisleri için optimal dinamikte kullanılan P matrisi üretilir. 4 numaralı satırda, Eş. 3, Eş. 4 ve Eş. 5 ile temsil edilen optimal dinamik oluşturulur. 5 numaralı satırda, Algoritma 2'de gösterilen ilklendirme işlevi gerçekleştirilir. Algoritma 1, 6. satırda çözünürlük değerini sonsuz olarak belirler ve 7. satırdan itibaren gösterilen döngü içerisinde tekrarlı olarak, çözünürlük belirli bir değerin (*res*) altına inene kadar optimal yolları hesaplamaya başlar. Bunun için öncelikle 8 ve 9. satırlarda denge noktasının yakınındaki hiper-elipsoit ve durum uzayının sınırını oluşturan hiper-küre yüzeylerini n boyutlu üçgenlere ayırır. 10. satırda hiper-küre üzerindeki en büyük üçgen seçilir. 11. Satırda, Algoritma 3 ile gösterilen arama işlemi gerçekleştirilerek hiper-küre üzerinde bulunan en büyük üçgeni bölecek başlangıç noktası ve optimal yol bulunur. Algoritma, hiper-küre üzerindeki en büyük üçgenin büyüklüğü *res* değerinin altına inince durur.

8. ve 9. satırlardaki üçgenleme fonksiyonları (TESSELATE), hiper-elipsoit ve hiper-küre yüzeylerindeki nokta bulutları üzerinde Delaunay Üçgenleme görevini yerine getirir. Bu amaçla, kapalı birer hiper-yüzey üzerinde olduğu bilinen her bir nokta bulutunu kapsayan dışbükey zarf (*İng.: convex hull*) belirlenerek bu yapıyı oluşturan üçgenler kullanılır. Dışbükey zarfın oluşturulmasında *Quickhull* algoritması [13] kullanılmıştır.

Durum uzayını ilklendiren Algoritma 2, Tablo 2'de gösterilmiştir.

Algoritma 2, durum uzayını yeterince düzgün şekilde parçalayan *4n* adet optimal yol üretir. Bu şekilde, optimal yolların bitiş noktalarının üzerinde bulunduğu hiper-kürenin üçgenlere ayrılmasından sonra, en uzun üçgen kenarının merkezde yarattığı açı en fazla 180 derece olur. Algoritma 2, 2 numaralı satırda başlangıç ve bitiş noktalarının oluşturduğu kümelerin boş küme olarak belirlenmesi ile başlar. 3-9 arasındaki satırlarda algoritma *2n* adet başlangıç noktasını tüm eksenlerde pozitif ve negatif yönlerde birer adet olacak şekilde seçer (örneğin +x, -x, +y, -y, …). Bunun için öncelikle 4 numaralı satırda, her bir eksenin verilen bir maliyet değerinin temsil ettiği hiper elipsoitle kesiştiği nokta belirlenir. 5 ve 6. Satırlarda, geriye integrasyon yöntemi bu noktadan ve eksene göre simetrik noktasından başlayarak optimal yollar oluşturulur. 7 ve 8. Satırlarda bu optimal yolların başlangıç ve bitiş noktaları C_i ve C_o kümeleri içerisine eklenir. Bu işlem tüm eksenler için yapıldıktan sonra 10. satırda hiper-elipsoit üzerinde ilk üçgenleme gerçekleştirilir. Bu sırada hiper-küre üzerinde biten optimal yolların başlangıç noktaları hiper-elipsoit üzerinde düzgün dağılmış durumdadır, ancak sistemin doğrusal olmayan doğası sebebiyle aynı durum hiper-küre üzerindeki bitiş noktaları için geçerli değildir. Bu sebeple, 11-17. Satırlar arasında, hiper-küre üzerinde düzgün dağılmış bitiş noktalarını oluşturmak üzere *2n* adet daha optimal yol üretilmektedir. Her eksenin pozitif ve negatif yönleri için, 12. ve 14. satırlarda birer optimizasyon algoritması koşturulmaktadır. Bu algoritmada, tüm hiper-elipsoit yüzeyi, 10. satırda oluşturulmuş üçgenler ($t \in T_{in}$) ve bu üçgenlerin üzerindeki noktaların Barisentrik koordinatları ile ($v \in \mathbb{R}^{n-1}$) temsil edilmektedir. Optimizasyonda hedef, en uygun üçgeni ve bu üçgen üzerindeki en uygun noktayı başlangıç noktası seçerek ilgili eksenin hiper-küreyi kestiği noktaya en yakın noktada sonlanan optimal yolu bulmaktır. Bu satırlardaki BI $(c(t, v), \dot{x}, \dot{p}, u, r_m)$ ifadesi, seçili üçgen ve koordinatın hiper-elipsoit üzerinde karşılık geldiği noktadan $(c(t, v))$ başlayarak optimal yolun geri integrasyon yöntemiyle oluşturulması ve r_m yarıçaplı hiper küre üzerinde sonlanmasını ifade etmektedir. 13, 15 ve 16. satırlarda optimal yolların sonlandığı noktalar C_o kümesi içerisine eklenmektedir. Algoritma, son olarak 18. Satırda hiper-küre üzerindeki noktalar üzerinde ilk üçgenlemeyi gerçekleştirmektedir.

Hiper-küre üzerindeki en büyük üçgenin içinde sonlanacak optimal yolu arayan Algoritma 3, Tablo 3'de gösterilmiştir.

Tablo 2. Durum Uzayının İlklendirilmesi
(Initialization of the State Space)

Algoritma 2	
1:	*procedure* INITSS $(\dot{x}, \dot{p}, u, P, V_f, r_m)$
2:	$(C_i, C_o) \leftarrow (\emptyset, \emptyset)$
3:	*for all* a ← axes *do*
4:	$(c) \leftarrow a / \sqrt{a^T P a / V_f}$
5:	(c_1)←BI $(c, \dot{x}, \dot{p}, u, r_m)$
6:	(c_2)←BI $(-c, \dot{x}, \dot{p}, u, r_m)$
7:	$(C_i) \leftarrow C_i \cup \{c, -c\}$
8:	$(C_o) \leftarrow C_o \cup \{c_1, c_2\}$
9:	*end for*
10:	(T_{in}) ←TESSELLATE (C_i)
11:	*for all* a ← axes *do*
12:	$(t, v) \leftarrow \underset{\substack{t \in T_{in} \\ v \in \mathbb{R}^{n-1}}}{\operatorname{argmin}} \lVert \text{BI}\,(c(t, v), \dot{x}, \dot{p}, u, r_m) - a r_m \rVert$
13:	(c_1)←BI $(c(t, v), \dot{x}, \dot{p}, u, r_m)$
14:	$(t, v) \leftarrow \underset{\substack{t \in T_{in} \\ v \in \mathbb{R}^{n-1}}}{\operatorname{argmin}} \lVert \text{BI}\,(c(t, v), \dot{x}, \dot{p}, u, r_m) + a r_m \rVert$
15:	(c_2)←BI $(c(t, v), \dot{x}, \dot{p}, u, r_m)$
16:	$(C_o) \leftarrow C_o \cup \{c_1, c_2\}$
17:	*end for*
18:	(T_{out}) ←TESSELLATE (C_o)
19:	*end procedure*

Algoritma, 2. Satırda hiper-küre üzerinde verili üçgenin merkezini hesaplar. Algoritmanın hedefi, bu merkeze en yakın şekilde sonlanacak optimal yolun başlangıç noktasını bulmaktır. Önce, verili üçgeni oluşturan noktalara ait optimal yolların hiper-elipsoit üzerindeki başlangıç noktaları kullanılarak 3. Satırda bir "sanal" üçgen oluşturulur. Bu üçgenin sanal olarak nitelendirilmesinin sebebi, hiper-elipsoit yüzeyinin üçgenlenmesi ile oluşmuş kümenin (T_{in}) içerisinde yer almama ihtimalidir. 4. Satırda bu sanal üçgenin de merkezi bulunur. 5. Satırda, T_{in} kümesinin içindeki üçgenlerden hangisinin bu merkezi içerdiği bulunur ve 6. Satırda bu üçgen genişlik öncelikli aramanın üzerinde yapılacağı Q listesinin ilk elemanı olur. Bu üçgen, aramanın başlayacağı, hiper-elipsoit üzerindeki ilk üçgendir. 7, 8 ve 9. Satırlarda, genişlik öncelikli arama

algoritmasında ihtiyaç duyulan derinlik değerleri tüm üçgenler için sonsuza eşitlenir. 10. Satırda en iyi hata değeri sonsuza eşitlenir. Böylece genişlik öncelikli arama algoritmasının ilklendirilmesi tamamlanmış olur. 11-27. Satırlar arasında bu arama algoritması gerçekleştirilir. Arama, 12. Satırda her seferinde Q listesindeki ilk üçgeni ele alır. Listede üçgen kalmayınca veya yeterince iyi bir üçgen bulunması durumunda arama sonlanır. 13. Satırda eldeki üçgen üzerinde bir optimizasyon gerçekleştirilmektedir. Burada, optimizasyon yapılacak *n-1* boyutlu uzayda $p_i \in \mathbb{R}^{n-1}$ parametreleri, bu üçgen üzerinde Barisentrik koordinatlarda bir noktayı temsil etmektedir. Optimizasyon, p_i noktasının temsil ettiği başlangıç noktasından $(c(t, p_i))$ başlayan optimal yolun sonlandığı p_o noktasının, p_r, yani dış üçgenin merkezine uzaklığının en küçük değerini bulmayı hedefler. 14-16. Satırlar, bu hata değerinin belirli bir eşiğin altına inmesi durumunda algoritmanın sonlanmasını sağlar. 17-20. Satırlar, eldeki üçgen üzerinde bulunan en iyi noktanın, daha önceki üçgenlerde bulunan en iyi noktadan iyi olup olmadığını kontrol eder ve daha iyiyse en iyi nokta bilgisini günceller. 21-26. Satırlar eldeki üçgenin daha önce üzerinde arama yapılmamış olan komşularını tespit edip arama kuyruğuna ekler. 28. Satırda algoritma, arama süresince bulduğu en iyi optimal yolun başlangıç ve bitiş noktalarını geri döndürür. Genişlik öncelikli arama uzayı Şekil 2'de görülebilir. Dış küre üzerindeki bir t1-t2-t3 üçgeninin köşelerinin elipsoid üzerindeki başlangıç noktalarının orta noktası "x" ile işaretlenmiştir. Arama algoritması, bu noktayı içeren a-b-c üçgenini ve bu üçgenin birinci dereceden komşularını aradığında, arama uzayı Şekil 2'deki gibidir.

Optimal yolları üreten Algoritma 1, yinelemeli olarak hem başlangıç noktalarının üzerinde yer aldığı hiper-elipsoiti, hem de optimal yolların üzerinde sonlandığı hiper-küreyi üçgenlere ayırır. Algoritma, hiper-küre üzerindeki en büyük üçgenin büyüklüğü belirli bir eşiğin altına ininceye kadar bu işlemi tekrarlar. Yeterince yüksek çözünürlüğün tüm durum uzayında sağlanmasını garanti altına almak için bu algoritma artan r_m değerleri için birden fazla kez çalıştırılır.

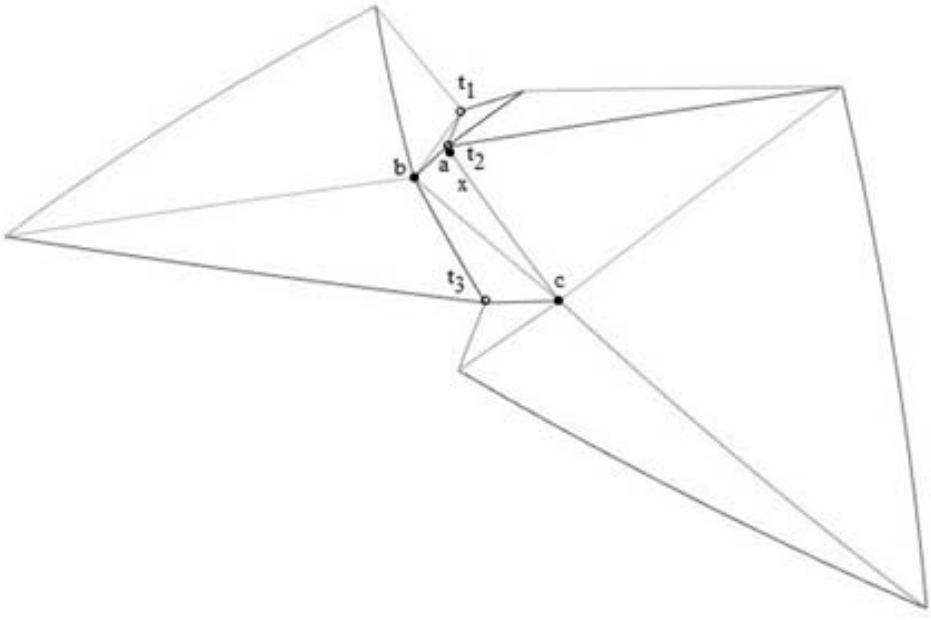
Algoritma 1 sonlandığında üretilen tüm optimal yollar kayıt altına alınmış olur. Kayıtlar, her bir r_m değeri için oluşan hiper-kürenin optimal yollarla kesiştiği noktalardan oluşur. Optimal denetleyici, geri besleme değerlerini bu kesişim noktalarını kullanarak hesaplar.

Tablo 3. Optimal Yol Arama Algoritması
(Optimal Path Search Algorithm)

Algoritma 3	
1:	*procedure* BESTTRAJ $(t, C_i, C_o, T_{in}, \dot{x}, \dot{p}, u, r_m)$
2:	$(p_r) \leftarrow$CENTER $(C_o[t])$
3:	t_i'=CREATETRI $(C_i[t(1)], C_i[t(2)], \dots, C_i[t(n)])$
4:	$(p_i) \leftarrow$CENTER $(C_i[t_i'])$
5:	$(t_i) \leftarrow$FINDTRI (p_i, T_{in})
6:	$(Q) \leftarrow \{t_i\}$
7:	*for all* $t \in T_{in}$ *do*
8:	$(d(t)) \leftarrow \infty$
9:	*end for*
10:	$(\varepsilon_{best}) \leftarrow \infty$
11:	*while* $Q \neq \emptyset$ *do*
12:	$(t) \leftarrow$pop (Q)
13:	$(p_i, p_o) \leftarrow \underset{p_i \in \mathbb{R}^{n-1}}{\operatorname{argmin}} \lVert$BI $(c(t, p_i), \dot{x}, \dot{p}, u, r_m) - p_r \rVert$
14:	*if* $\lVert p_o - p_{ref} \rVert \leq \varepsilon_{min}$ *then*
15:	*return* (p_i, p_o)
16:	*end if*
17:	*if* $\lVert p_o - p_{ref} \rVert \leq \varepsilon_{best}$ *then*
18:	$(p_i^*, p_o^*) \leftarrow (p_i, p_o)$
19:	$(\varepsilon_{best}) \leftarrow \lVert p_o - p_{ref} \rVert$
20:	*end if*
21:	*for all* t' adjacent to t *do*
22:	*if* $d(t') == \infty$ *then*
23:	$(d(t')) \leftarrow d(t) + 1$
24:	$(Q) \leftarrow Q \cup t'$
25:	*end if*
26:	*end for*
27:	*end while*
28:	*return* (p_i^*, p_o^*)
29:	*end procedure*

4. Optimal Denetleyici Tasarımı (Optimal Controller Design)

Eş. 5'te gösterilen giriş tasarımı, sistem durumuna (x), yardımcı sistem durumuna (p), girişin sistem dinamiğine etkisini temsil eden vektör alanına $(g(x))$ ve giriş ağırlık matrisine (R) bağlıdır.



Şekil 2. Genişlik öncelikli arama (Breadth first search)

Denetleyicinin giriş değerini hesaplarken x, $g(x)$ ve R değerlerine doğrudan erişimi vardır, ancak p yardımcı durum vektörüne yoktur. Bu sebeple, veri tabanı oluşturulurken, her kayıt noktası için optimal yolun o noktasındaki p değeri kullanılıp u girişi hesaplanarak kayda dahil edilmektedir.

Denetleyici, belirli bir x sistem durumu için girişi veri tabanında interpolasyon yaparak hesaplar. Her denetleyici iterasyonunda, önce x değerini içeren üçgen ve bu değerin üçgen üzerindeki konumu Barisentrik koordinatlarda bulunur. Üçgen üzerindeki bu sistem durumu $x_b \in \mathbb{R}^{n-1}$ Eş. 12'de gösterilmiştir.

$$x_b = [x_{b1} \quad x_{b2} \quad \cdots \quad x_{b(n-1)}]^T \tag{12}$$

Burada $0 \leq x_{bi} \leq 1$, üçgenin i numaralı köşesinin ağırlığını temsil etmektedir. Barisentrik koordinatların özelliği gereği $x_{bn} + \sum_{i=1}^{n-1} x_{bi} = 1$ olduğu için x_b, dolaylı olarak n numaralı köşenin ağırlığını da içermektedir. Bu bilgileri kullanılarak denetleyici tarafından giriş Eş. 13'deki gibi hesaplanır.

$$u = \sum_{i=1}^{n-1} x_{bi} u_i + \left(1 - \sum_{i=1}^{n-1} x_{bi}\right) u_n \tag{13}$$

Burada u_i, üçgenin köşelerindeki giriş değerlerini temsil etmektedir. Dolayısıyla giriş, n boyutlu üçgenin köşelerindeki giriş değerlerinin x_bağırlıklarına göre ortalaması olarak hesaplanmış olur.

5. Simülasyon Sonuçları (Simulation Results)

Önerilen optimal denetleyici, aşağıdaki üç boyutlu doğrusal olmayan sistemin denetimi için kullanılarak simülasyonu yapılmıştır (Eş. 14).

$$\dot{x} = \begin{bmatrix} -x_1 + x_2 \\ x_1 - x_2 + x_1 x_3 + u \\ x_1 + x_1 x_2 - x_3 \end{bmatrix} \tag{14}$$

Bu sistemin, Eş. 1'de belirtilen forma getirildiğindeki bileşenleri aşağıda gösterilmiştir (Eş. 15).

$$f(x) = \begin{bmatrix} -x_1 + x_2 \\ x_1 - x_2 + x_1 x_3 \\ x_1 + x_1 x_2 - x_3 \end{bmatrix}, g(x) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{15}$$

Geri integrasyon yönteminin başlangıç noktalarının üzerinde bulunacağı hiper-elipsoiti belirlemek üzere, sistem 0 noktasında Eş. 7 ile doğrusallaştırılarak Eş. 6'daki A ve B matrisleri elde edilir (Eş. 16).

$$A = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{16}$$

Eş. 2'deki durum ve giriş ağırlık matrisleri Eş. 17'deki gibi seçilmiştir:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = [1] \tag{17}$$

Eş. 9'daki ARE denklemi çözülerek aşağıdaki *P* matrisi bulunur (Eş. 18).

$$P = \begin{bmatrix} 1.2307 & 0.7639 & 0.2586 \\ 0.7639 & 0.8783 & 0.0898 \\ 0.2586 & 0.0898 & 0.4960 \end{bmatrix} \tag{18}$$

Eş. 3, Eş. 4 ve Eş. 5 kullanılarak optimal yardımcı durum dinamiği ve giriş bulunur (Eş. 19-20).

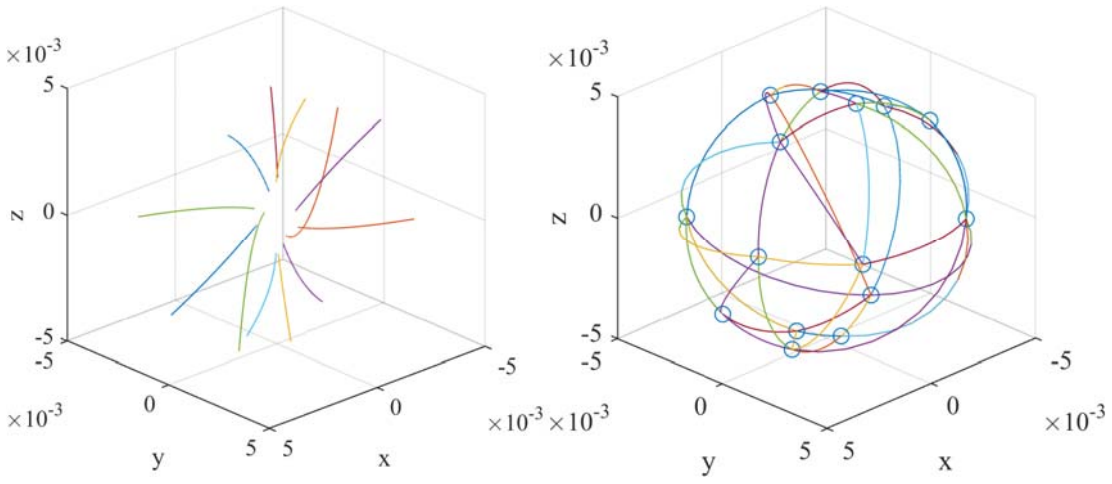
$$\dot{p} = \begin{bmatrix} p_1 - x_1 + p_2(x_3 - 1) - p_3(x_2 + 1) \\ p_2 - p_1 - x_2 - p_3 x_1 \\ p_3 - x_3 + p_2 x_1 \end{bmatrix} \tag{19}$$

$$u = -p_2 \tag{20}$$

Dinamiğin doğrusal olmayan doğasını görebilmek için Algoritma 1, $V_f = 1e-7$ ve $r_m = 0.005$ olacak şekilde çalıştırılmıştır. 20 iterasyon sonra oluşan optimal yollar ve küre üzerindeki üçgenleme Şekil 3'te görülebilir:

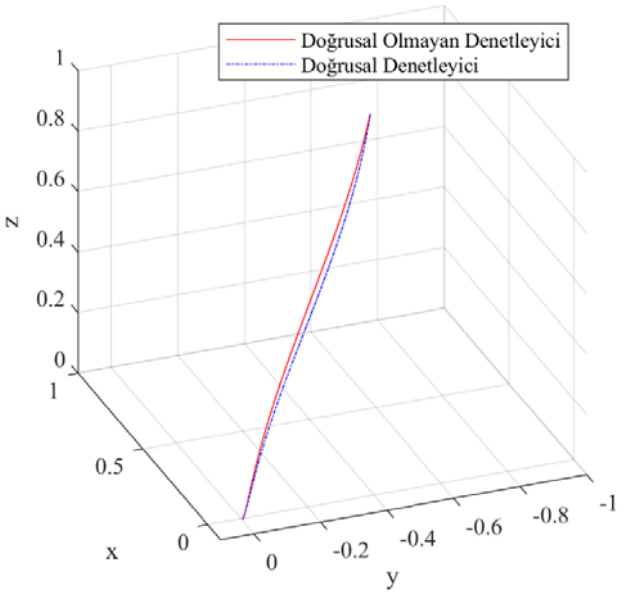
Solda, optimal yolların merkezdeki elipsoit yüzeyinden yola çıkışları ve izledikleri yollar, sağda ise optimal yolların üzerinde sonlandıkları küre ve bir sonraki başlangıç noktasını seçebilmek için ihtiyaç duyulan üçgenleme görülebilir.

Denetleyicinin performansını göstermek için algoritma 60.000 iterasyon çalıştırılmış ve optimal yollar yarıçapları artacak şekilde 8 katmanda kürelerle kesiştirilmiştir. Bu kürelerin yarı çapları [0,005 0,01 0,03 0,05 0,1 0,2 0,5 1] olacak şekilde belirlenmiştir. Yolları oluşturan Algoritma 1, her katman için en büyük üçgen çevresi o katmana ait çözünürlük sınırının altına ininceye kadar çalıştırılmıştır. Bu sınırlar: [0,003 0,003 0,005 0,005 0,01 0,02 0,05 0,1].

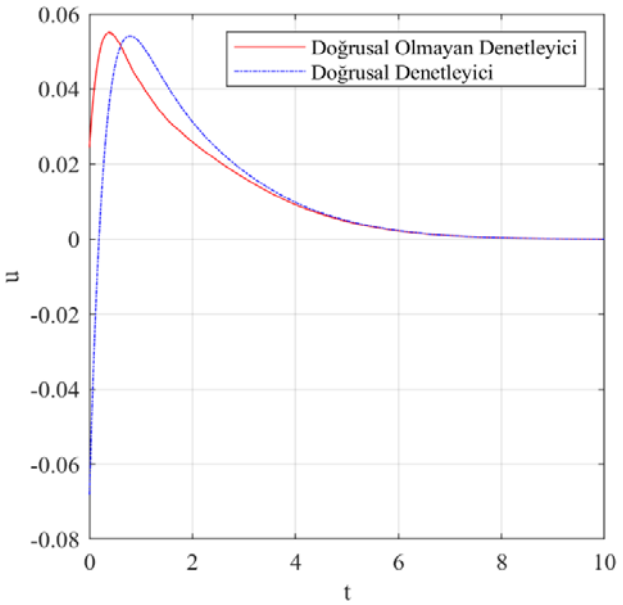


Şekil 3. 20 iterasyon sonunda optimal yollar (Optimal paths after 20 iterations)

Denetleyici, 1 KHz örnekleme frekansında 10 sn. çalıştırılmıştır. Başlangıç koşulları: $x_0 = [0{,}8 \quad -0{,}7 \quad 0{,}8]^T$ olacak şekilde belirlenmiştir. Denetleyicinin performansını değerlendirebilmek için, aynı sistem, doğrusallaştırılmış sistem kullanılarak tasarlanan bir LQR denetleyicisi ile de çalıştırılmıştır. Her iki sistemin sistem durumlarının zaman içerisindeki değişimleri Şekil 4'te, sistem girişlerinin zaman içerisindeki değişimleri de Şekil 5'te gösterilmiştir.



Şekil 4. Doğrusal olmayan ve LQR denetleyici ile durum vektör yolları (State trajectories with the nonlinear and LQR controllers)



Şekil 5. Doğrusal olmayan ve LQR denetleyicinin ürettiği girişler (Generated inputs of the nonlinear and LQR controllers)

Şekil 4 ve Şekil 5'te görüldüğü gibi, doğrusal olmayan denetleyici LQR denetleyiciye göre daha hızlı tepki vermekte ve sistemin denge noktasına daha kısa yoldan ulaşmasını sağlamaktadır. Her iki denetleyicinin optimallik konusundaki etkinliğini kıyaslayabilmek için çalıştıkları süre boyunca durum vektör normları ile giriş normlarını toplamları Tablo 4'te gösterilmiştir:

Tablo 4. Toplam denetleyici maliyetleri (Total controller costs)

Denetleyici	$\int \|x\|^2$	$\int \|u\|^2$	J
Doğrusal olmayan denetleyici	46,6747	2,0242	48,6989
LQR denetleyici	46,7976	2,1309	48,9285

6. Sonuçlar ve Tartışmalar (Conclusions and Discussions)

Bu çalışmada, Euler-Lagrange Geri İntegrasyon yönteminin üç ve daha fazla boyutlu doğrusallaştırılabilen, doğrusal girişli doğrusal olmayan sistemlerde kullanılabilmesine yönelik yeni bir algoritma sunulmuştur. Bu yöntemin iki boyutlu sistemlerde kullanımında optimal yolların belirlenmesi, bu yolların başlangıç noktalarının seçiminde durum uzayındaki boş bölgelerin karşılık geldiği elips arklarının ikiye bölünmesi gibi basit bir yöntemde gerçekleştirilir. Üç ve daha fazla boyutlu sistemlerde ise bu yöntem kullanılamaz. Bu çalışmada sunulan yöntemde her iterasyonda durum uzayının sınırını belirleyen hiper-küre yüzeyi üçgenlere ayrılarak en büyük üçgen belirlenmekte ve bu üçgenin içinde sonlanacak optimal yolun başlangıç noktası bir arama algoritması ve optimizasyon yöntemi ile bulunmaktadır. Daha sonra bu yolların oluşturduğu veri tabanı denetleyici tarafından denetim çevriminde kullanılmaktadır.

Tasarlanan denetleyicinin üç boyutlu doğrusal olmayan bir sistemin denetiminde simülasyonu gerçekleştirilmiştir. Aynı sistemin denetimi daha sonra bir LQR denetleyici ile gerçekleştirilerek her iki denetleyicinin performansı kıyaslanmıştır. Sonuçta, tasarlanan denetleyicinin hem hata normu hem de uygulanan giriş büyüklüğü açısından LQR denetleyiciye göre daha iyi olduğu gösterilmiştir.

Optimal yolların oluşturulması yüksek bilgisayar performansı gerektiren bir algoritma ile gerçekleştirilmektedir. Bu sebeple, algoritmanın uygulanması öncelikli olarak üç boyutlu bir sistemde gerçekleştirilmiştir. Gelecek çalışmalarda önce sistemin dört ve daha yüksek boyutlu sistemlerde kullanılabilmesi için optimal yolların oluşturulmasında kullanılan algoritmanın performansının artırılması hedeflenmektedir. Ayrıca, gelecek çalışmalarda kullanılacak doğrusal olmayan sistemin birden fazla denge noktası üzerinde çalışabilmesi hedeflenecektir.

Kaynaklar (References)

1. Kirk D. E., Optimal control theory: an introduction, Courier Corporation, 2004.
2. Lewis F. L., Optimal control, John Wiley & Sons, 2012.
3. Dupree K., Patre P. M., Wilcox Z. D., Dixon W. E., Asymptotic optimal control of uncertain nonlinear Euler–Lagrange systems, Automatica, 47 (1), 99-107, 2011.
4. Murad A. K., Lewis F. L., Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach, Automatica, 41 (5), 779-791, 2005.
5. Vamvoudakis K. G., Frank L. L., Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem, Automatica, 46 (5), 878-888, 2010.
6. Bhasin S., Kamalapurkar R., Johnson M., Vamvoudakis K. G., Lewis F. L., Dixon W. E., A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems, Automatica, 49 (1), 82-92, 2013.
7. Kiumarsi B., Vamvoudakis K. G., Modares H., Lewis F. L., Optimal and autonomous control using reinforcement learning: A survey, IEEE transactions on neural networks and learning systems, 29 (6), 2042-2062, 2017.
8. Liu D., Wei Q., Wang D., Yang X., Li H., Adaptive dynamic programming with applications in optimal control, Springer International Publishing, 2017.
9. Holzhüter T., Optimal regulator for the inverted pendulum via Euler–Lagrange backward integration., Automatica, 40 (9), 1613-1620, 2004.
10. Holzhüter T., Klinker T., Method to solve the nonlinear infinite horizon optimal control problem with application to the track control of a

mobile robot, International Journal of Bifurcation and Chaos, 17 (10), 3607-3611, 2007.

11. Nekoo S. R. and Rahaghi M. I., Recursive approximate solution to time-varying matrix differential Riccati equation: linear and nonlinear systems, International Journal of Systems Science, 49 (13), 2797-2807, 2018.

12. Aksoy O., Doğrusal olmayan ve belirsiz Euler-Lagrange sistemlerinin optimal çıkış geri beslemeli denetimi üzerine, Doktora Tezi, Gebze Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Kocaeli, 2016.

13. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.T., The Quickhull Algorithm for Convex Hulls, ACM Transactions on Mathematical Software, 22 (4), 469-483, 1996.