مجله مهندسی مکانیک مدرس، مهر 1396، دوره 17، شماره 7، ص‌ص 413-420

ماهنامه علمی پژوهشی

مهندسی مکانیک مدرس

mme.modares.ac.ir

مقایسه کنترلر گام به عقب بهینه‌شده با الگوریتم ازدحام ذرات و کنترلر LQR بر روی کوادروتور

نیلوفر پرهیزکار[1]، ابوالقاسم نقاش[2*]

1- کارشناسی ارشد، مهندسی هوافضا، دانشگاه صنعتی امیرکبیر، تهران

2- استاد، مهندسی هوافضا، دانشگاه صنعتی امیرکبیر، تهران

* تهران، صندوق پستی 1591634311، naghash@aut.ac.ir

اطلاعات مقاله

مقاله پژوهشی کامل

دریافت: 17 فروردین 1396

پذیرش: 05 خرداد 1396

ارائه در سایت: 13 مرداد 1396

کلید واژگان:

کنترل‌کننده گام به عقب

کنترل‌کننده LQR

الگوریتم ازدحام ذرات

تابع هزینه

بهینه‌سازی

چکیده

این مقاله به مقایسه عملکرد دو کنترلر گام به عقب بهینه‌شده با الگوریتم ازدحام ذرات و کنترلر LQR بر روی یک کوادروتور در حالت هاور می‌پردازد. کوادروتور یک سیستم دینامیکی پایدار نیست و توسعه کنترلرهای بهینه با کارایی بالا برای آن حائز اهمیت است. در ابتدا مدل دینامیکی کوادورتور معرفی شده و معادلات فضای حالت به منظور شبیه‌سازی مدل دینامیکی ارائه می‌شوند. سپس دو کنترلر گام به عقب و LQR برای کنترل ارتفاع و زوایای کوادروتور طراحی می‌شوند. به منظور بهینه‌سازی کنترلر گام به عقب، پارامترهای آن به کمک الگوریتم ازدحام ذرات به گونه‌ای تعیین می‌گردند که تابع هزینه در نظر گرفته شده برای کنترلر LQR کمترین مقدار خود را داشته باشد. هم چنین فرامین ورودی به موتورها به منظور نشان دادن توانایی کنترلر موردنظر محاسبه و ترسیم می‌شوند. برای رسیدن به مقایسه دقیق تر، تابع هزینه به ازای مقادیر مختلف ماتریس‌های وزنی Q و R برای دو کنترلر محاسبه شده و نتایج با یکدیگر مقایسه می‌شوند. نتایج نشان می‌دهند که کنترلر گام به عقب به کمک الگوریتم ازدحام ذرات توانایی بالاتری در کمینه سازی تابع هزینه دارد و به ازای ماتریس‌های وزنی مختلف تابع هزینه در کنترلر گام به عقب کمتر بوده است.

Comparison of Back Stepping Optimized via PSO Algorithm and LQR Controllers for a Quadrotor

Niloofar Parhizkar, Abolghasem Naghash*

Department of Aerospace Engineering, Amirkabir University of Technology, Tehran, Iran

* P.O.B. 1591634311, Tehran, Iran, naghash@aut.ac.ir

ARTICLE INFORMATION

Original Research Paper
Received 06 April 2017
Accepted 26 May 2017
Available Online 04 August 2017

Keywords:
Back stepping
LQR
PSO algorithm
Cost function
Optimization

ABSTRACT

Comparison of Back stepping method optimized via particle swarm optimization algorithm and LQR method for hovering control of a quadrotor is presented in this paper. Quadrotor is not a stable dynamical system and development of high performance controllers for it is important. First the dynamic model of a quadrotor is introduced and state-space equations are presented in order to simulate the dynamic model. Then two Back stepping and LQR controllers are designed to control Euler angles and height of the quadrotor. In order to optimize back stepping controller, its parameters are determined using particle swarm optimization algorithm to minimize cost function considered for LQR controller. Also, commands to the motors are calculated and plotted to show the feasibility of the controller. To obtain better comparison, the cost function is calculated for different weighting matrices of Q and R for two controllers and the results are compared. The results show that Back stepping controller has more ability to minimize the cost function in comparison to LQR and the cost function in Back stepping has fewer values for several choices of weighting matrices.

1- مقدمه

در دهه‌های اخیر مطالعات به منظور مدلسازی و کنترل ربات‌های پرنده روند رو به رشدی را به خود گرفته است و الگوریتم‌های کنترل موجود رو به افزایش است.

با بررسی الگوریتم‌های کنترلی طراحی‌شده برای کوادروتور ملاحظه می‌شود که با کامل تر شدن تحقیقات بر روی روش‌های کنترل کلاسیک، استفاده از روش‌های ترکیبی در سال‌های اخیر جایگاه ویژه‌ای پیدا کرده است.

تعیین پارامترهای طراحی کنترل‌کننده‌ها با استفاده از الگوریتم‌های هوشمند اخیراً روند رو به رشدی به خود گرفته است. روش‌ها و الگوریتم‌های بهینه‌سازی به دو دسته الگوریتم‌های دقیق و الگوریتم‌های تقریبی تقسیم‌بندی می‌شوند. الگوریتم‌های دقیق قادر به یافتن جواب بهینه به صورت دقیق هستند اما در مورد مسائل بهینه‌سازی سخت کارایی ندارند و زمان حل آنها در این مسائل به صورت نمایی افزایش می‌یابد. الگوریتم‌های تقریبی قادر به یافتن جواب‌های خوب (نزدیک به بهینه) در زمان حل کوتاه برای مسائل

Please cite this article using:

برای ارجاع به این مقاله از عبارت ذیل استفاده نمایید:

N. Parhizkar, A. Naghash, Comparison of Back Stepping Optimized via PSO Algorithm and LQR Controllers for a Quadrotor, *Modares Mechanical Engineering*, Vol. 17, No. 7, pp. 413-420, 2017 (in Persian)

[DOR: 20.1001.1.10275940.1396.17.7.47.9]

بهینه‌سازی سخت هستند. الگوریتم‌های تقریبی معمولاً به دو دسته الگوریتم‌های ابتکاری[1] و فرا ابتکاری[2] طبقه‌بندی می‌شوند. دو مشکل اصلی الگوریتم‌های ابتکاری، قرار گرفتن آنها در بهینه‌های محلی و ناتوانی آنها برای کاربرد در مسائل گوناگون است. الگوریتم‌های فراابتکاری برای حل این مشکلات الگوریتم‌های ابتکاری ارائه شده‌اند. در واقع الگوریتم‌های فراابتکاری، یکی از انواع الگوریتم‌های بهینه‌سازی تقریبی هستند که دارای راهکارهای برون‌رفت از بهینه محلی می‌باشند و قابل کاربرد در طیف گسترده‌ای از مسائل هستند. مثال‌هایی از این الگوریتم‌ها، الگوریتم ژنتیک، الگوریتم ازدحام ذرات، الگوریتم سرمایش تدریجی، الگوریتم رقابت استعماری، الگوریتم تکامل تفاضلی، الگوریتم مورچگان، الگوریتم جهش قورباغه، الگوریتم زنبور عسل، الگوریتم جست و جوی هارمونی، الگوریتم جست و جوی گرانشی، الگوریتم فاخته، الگوریتم خفاش‌ها و بسیاری الگوریتم‌های دیگر هستند.

در مرجع [1] الگوریتم ژنتیک و در مرجع [2] الگوریتم ازدحام ذرات برای تنظیم ضرایب کنترل‌کننده PID بر روی کوادروتور استفاده شده‌اند.

در مراجع [3] و [4] نیز کنترل‌کننده گام به عقب بهینه شده با الگوریتم ازدحام ذرات برای کوادروتور طراحی شده است.

ترکیب روش‌های کنترل کلاسیک با روش‌های تطبیقی مانند منطق فازی، شبکه‌های عصبی و ... نیز از روش‌هایی هستند که به منظور بهبود کارایی کنترلرها به کار می روند و در این روش‌ها پارامترهای طراحی یک کنترلر به صورت متغیر با زمان تنظیم می‌شوند.

روش‌های تطبیقی نیز به منظور ایجاد پاسخ مناسب در مقابل تغییرات آرام در سیستم و همچنین خطاهای مدل‌سازی مورد بررسی قرار گرفته‌اند. در مراجع [5] و [6] از الگوریتم فازی برای تنظیم ضرایب کنترل‌کننده‌های PID و مود لغزشی بهره گرفته شده است. هم چنین در مرجع [7] کنترل‌کننده LQR با الگوریتم فازی به منظور بهبود عملکرد و مقاوم تر شدن در برابر اغتشاشات و نامعینی‌ها ترکیب شده است. روش‌های شبکه‌های عصبی نیز به میزان کمتر استفاده شده اما دور از نظر نگه داشته نشده‌اند. در مراجع [8] و [9] الگوریتم شبکه‌های عصبی به ترتیب با PID و گام به عقب ترکیب شده‌اند. در [10]، برای جبران عدم قطعیت‌های پارامتری موجود در دینامیک مسی کوپتر از سه روش کنترلی غیرخطی تطبیقی مدل مرجع برای سه حالت متفاوت مبتنی بر معادلات خطی تک ورودی- تک خروجی و معادلات خطی چند ورودی- چند خروجی و معادلات غیرخطی چند ورودی- چند خروجی استفاده شده است.

2- معرفی کوادروتور مورد بررسی

کوادروتور شبیه‌سازی شده در این مقاله ای.آر.درون[3] ساخت شرکت پروت[4] است. این کوادروتور یک کامپیوتر داخلی با پردازنده 468 MHz ARM9 و حافظه اجرایی 128MB که تحت سیستم عامل لینوکس است با خود به همراه دارد و دارای سیستم کنترل اتوماتیک است که امکان بلند شدن و فرود آمدن کوادروتور را فراهم می‌کند. نمای کلی یک ای.آر.درون در شکل 1 مشاهده می‌شود.

مقادیر پارامترهای فیزیکی و مشخصات مورد نیاز برای شبیه‌سازی مدل غیرخطی این کوادروتور در مرجع [11] به صورت فیزیکی اندازه‌گیری شده‌اند. این مقادیر در جدول 1 قابل مشاهده هستند.

[1] Heuristic
[2] Metaheuristic
[3] AR.Drone
[4] parrot

Fig. 1 Overview of an AR.Drone

شکل 1 نمای کلی از ای.آر.درون

جدول 1 مقادیر پارامترهای فیزیکی کوادروتور [11]

Table 1 Value of physical parameters of the quadrotor [11]

پارامتر فیزیکی	توضیح	مقدار	واحد
m	جرم	335	gr
L	فاصله مرکز ملخ تا مرکز جرم (بازوی ممان)	18	cm
I_z	ممان اینرسی حول محور z	0.0047	kgm^2
I_x	ممان اینرسی حول محور x	0.0018	kgm^2
I_y	ممان اینرسی حول محور y	0.0018	kgm^2
b	ضریب تراست ملخ	5.7231×10^{-6}	Ns^2
d	ضریب گشتاور آیرودینامیکی	1.7169×10^{-7}	Nms^2
J_r	اینرسی روتور	1.85×10^{-5}	kgm^2

3- مدلسازی دینامیکی کوادروتور

فرضیات ساده کننده‌ای که در این مدلسازی مورد استفاده قرار می‌گیرند به شرح زیر هستند:

1- زمین مسطح فرض می‌شود.
2- سازه کوادروتور و ملخ‌ها صلب در نظر گرفته می‌شود.
3- فرض می‌شود که سازه متقارن است.
4- مرکز جرم و نیز مبدأ دستگاه مختصات بدنی منطبق بر مرکز تقارن سازه در نظر گرفته می‌شوند.
5- محورهای مختصات بدنی متصل به کوادروتور بر محورهای اصلی کوادروتور منطبق هستند در این صورت ماتریس ممان اینرسی، قطری شده و باعث ساده‌تر شدن معادلات می‌شود.
6- تراست و نیروی پسای ایجادشده توسط ملخ‌ها متناسب با مربع سرعت زاویه‌ای ملخ در نظر گرفته می‌شوند.

دستگاه مختصات اینرسیXYZ و دستگاه مختصات بدنیxyz مطابق با شکل 2 در نظر گرفته می‌شود.

3-1- استخراج مدل دینامیک دورانی کوادروتور

مومنتوم زاویه‌ای برابر با مومنتوم تولیدشده توسط دوران کوادروتور و سرعت

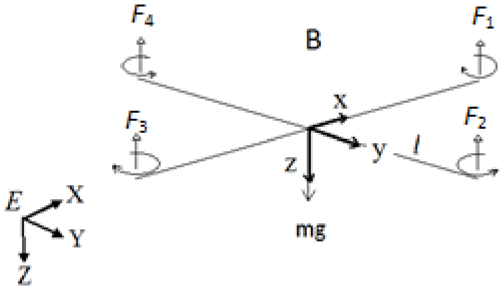

Fig. 2 Body and inertial frames

شکل 2 نمایش دستگاه‌های مختصات اینرسی و بدنی

[DOR: 20.1001.1.10275940.1396.17.7.47.9]

زاویه‌ای ملخ هاست. فرض می‌شود $\vec{M}$ برآیند گشتاورهای خارجی، ω_x ، ω_y و ω_z سرعت‌های زاویه‌ای و Ω_{blade} اختلاف بین سرعت زاویه‌ای ساعتگرد و پادساعتگرد باشد:

$$\Omega_{\text{blade}} = \Omega_1 + \Omega_3 - \Omega_2 - \Omega_4 \tag{1}$$

که در رابطه فوق $\Omega_{i,i=1,2,3,4}$ سرعت زاویه‌ای هریک از ملخ هاست. با استفاده از رابطه نیوتن-اویلر به شیوه مرجع [12] ، رابطه زیر برای گشتاور خارجی استخراج می‌شود:

$$\vec{M} = \begin{bmatrix} I_x\dot{\omega}_x + \omega_y\omega_z(I_z - I_y) + J_r\Omega\omega_y \\ I_y\dot{\omega}_y + \omega_x\omega_z(I_x - I_z) + J_r\Omega\omega_x \\ I_z\dot{\omega}_z + \omega_x\omega_y(I_y - I_x) \end{bmatrix} \tag{2}$$

اگر $T_{i,i=1,2,3,4}$ نیروی تراست تولیدشده توسط هر موتور باشد و $\tau_{i,i=1,2,3,4}$ گشتاور تولیدشده در جهت مخالف چرخش هر ملخ باشد که توسط هوا روی ملخ اعمال می‌شود، خواهیم داشت:

$$\tau_i = d\Omega_i^2 \quad , \quad T_i = b\Omega_i^2 \tag{3}$$

گشتاورهای رول، پیچ و یاو را به ترتیب با τ_x ، τ_y و τ_z نشان می‌دهیم. در این صورت خواهیم داشت:

$$\vec{M} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} -(T_2 - T_4)L \\ (T_1 - T_3)L \\ (-\tau_1 - \tau_3 + \tau_2 + \tau_4) \end{bmatrix} = \begin{bmatrix} -(T_2 - T_4)L \\ (T_1 - T_3)L \\ \frac{d}{b}(-T_1 - T_3 + T_2 + T_4) \end{bmatrix} \tag{4}$$

با ترکیب معادلات (2) و (4) ، بازنویسی معادله جدید برحسب مشتقات سرعت‌های زاویه‌ای، به‌کارگیری رابطه بین سرعت‌های زاویه‌ای حول محورهای بدنی و نرخ زوایای اویلر که در [12] محاسبه شده است و در نهایت با صرف‌نظر از ترمه‌ای بسیار کوچک و تبدیل معادلات به نمایش بر حسب زوایای اویلر (که با استفاده از ماتریس دوران صورت می‌گیرد) به رابطه زیر می‌رسیم:

$$\begin{bmatrix} \ddot{\varphi} \\ \ddot{\theta} \\ \ddot{\Psi} \end{bmatrix} = \begin{bmatrix} a_1(\dot{\theta}\dot{\Psi}) - a_4(\dot{\theta} + \varphi\dot{\Psi})\Omega + b_3\tau_z\theta + b_1\tau_x \\ a_2(\dot{\varphi}\dot{\Psi}) + a_5(\dot{\varphi} - \theta\dot{\Psi})\Omega - b_3\tau_z\varphi + b_2\tau_y \\ a_3(\dot{\varphi}\dot{\theta}) + (a_5\Omega\dot{\varphi} + b_2\tau_y)\varphi + b_3\tau_z \end{bmatrix} \tag{5}$$

که در آن:

$$a_1 = \frac{(I_y - I_z)}{I_x} + 1 \;, a_2 = \frac{(I_z - I_x)}{I_y} - 1 \;, a_3 = \frac{(I_x - I_y)}{I_z} + 1$$
$$a_4 = \frac{J_r}{I_x} \;, \quad a_5 = \frac{J_r}{I_y} \;, \quad b_1 = \frac{1}{I_x} \;, b_2 = \frac{1}{I_y} \;, b_3 = \frac{1}{I_z} \tag{6}$$

برای مطالعه سیستم در فرم فضای حالت متغیرهای حالت به صورت زیر در نظر گرفته می‌شوند:

$$\begin{cases} x_1(t) = \varphi(t) \\ x_2(t) = \dot{\varphi}(t) \\ x_3(t) = \theta(t) \\ x_4(t) = \dot{\theta}(t) \\ x_5(t) = \Psi(t) \\ x_6(t) = \dot{\Psi}(t) \end{cases} , \quad \begin{cases} U_2(t) = \tau_x \\ U_3(t) = \tau_y \\ U_4(t) = \tau_z \end{cases} \tag{7}$$

و با بازنویسی معادلات (5) و (6) به فرم نمایش فضای حالت به روابط نهایی زیر برای مدلسازی دینامیک دورانی کوادروتور می‌رسیم:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = a_1(x_4x_6) - a_4(x_4 + x_1x_6)\Omega \\ \quad +b_3.U_4.x_3 + b_1.U_2 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = a_2(x_2x_6) + a_5(x_2 - x_3x_6)\Omega \\ \quad -b_3.U_4.x_1 + b_2.U_3 \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = a_3(x_2x_4) + a_5(x_1x_2)\Omega \\ \quad +b_2.U_3.x_1 + b_3.U_4 \end{cases} \tag{8}$$

معادلات فوق غیرخطی هستند و حرکات رول، پیچ و یاو در همدیگر کوپل گردیده‌اند.

2-3- استخراج معادله حرکت ارتفاع

نیروهای اصلی عمل‌کننده بر روی کوادروتور شامل نیروی وزن و نیروی تراست هستند. مجموع تراست موتورها با U_1 نشان داده می‌شود. می‌دانیم که U_1 در راستای محور z بدنی و در جهت منفی آن است بنابراین برای به دست آوردن معادله حرکت ارتفاع در دستگاه اینرسی، باید U_1 از دستگاه بدنی به دستگاه اینرسی انتقال یابد و چون نیروی وزن نیز در راستای محور z دستگاه اینرسی است، در نهایت رابطه زیر برای حرکت در راستای محور z دستگاه اینرسی استخراج می‌شود [13]:

$$\ddot{z} = -g + \frac{1}{m}(\cos\theta\cos\varphi)U_1 \tag{9}$$

و با تعریف زیر:

$$[x_7 \;, x_8]^{\mathrm{T}} = [z \;, \dot{z}]^{\mathrm{T}} \tag{10}$$

و با در نظر داشتن رابطه (7) معادله (9) را به فرم فضای حالت در می‌آوریم:

$$\dot{x}_7 = x_8$$
$$\dot{x}_8 = -g + \frac{1}{m}(\cos x_3 \cos x_1)U_1 \tag{11}$$

4-طراحی کنترلر برای مدل دینامیکی غیرخطی

در این مقاله کنترل‌کننده گام به عقب و کنترل‌کننده LQR برای کنترل زوایای اویلر و ارتفاع کوادروتور طراحی می‌شوند. برای اینکه بتوانیم دو روش را به طور دقیق با یکدیگر مقایسه کنیم تابع هزینه‌ای را که استفاده از آن در طراحی کنترل‌کننده LQR مرسوم است برای هر دو کنترلر در نظر می‌گیریم:

$$J = \int_{t_0}^{t_f} (X^{\mathrm{T}}QX + U^{\mathrm{T}}RU)dt \tag{12}$$

به این صورت که برای کمینه سازی تابع هزینه در کنترلر گام به عقب از الگوریتم بهینه‌سازی ازدحام ذرات بهره می‌گیریم و با استفاده از این الگوریتم پارامترهای گام به عقب را به گونه‌ای تعیین می‌کنیم که تابع هزینه J مینیمم شود. خروجی‌های سیستم‌های کنترل U_1، U_2 ، U_3 و U_4 هستند. این خروجی‌ها باید به ورودی‌های مدل دینامیکی که سرعت‌های زاویه‌ای ملخ‌ها $\Omega_{i,i=1,2,3,4}$ هستند تبدیل شوند. می‌دانیم U_1 مجموع تراست موتورها و U_2 ، U_3 و U_4 گشتاورها هستند:

$$\begin{cases} U_1 = T_1 + T_2 + T_3 + T_4 \\ U_2 = -(T_2 - T_4)L \\ U_3 = (T_1 - T_3)L \\ U_4 = \frac{d}{b}(T_2 + T_4 - T_3 - T_1) \end{cases} \tag{13}$$

با بازنویسی روابط (13) بر حسب تراست ها (مشابه روشی که در [14] انجام شده است) و سپس مربع سرعت‌های زاویه‌ای ملخ‌ها به روابط زیر می‌رسیم:

$$\begin{bmatrix} {\Omega_1}^2 \\ {\Omega_2}^2 \\ {\Omega_3}^2 \\ {\Omega_4}^2 \end{bmatrix} = \frac{1}{b}\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} = \frac{1}{b}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -L & 0 & L \\ L & 0 & -L & 0 \\ -\frac{d}{b} & \frac{d}{b} & -\frac{d}{b} & \frac{d}{b} \end{bmatrix}^{-1} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{14}$$

1-4- کنترل‌کننده LQR

برای طراحی کنترلر LQR لازم است که معادلات (8) و (11) خطی سازی شوند. برای خطی سازی معادلات از روش ژاکوبین استفاده کرده و معادلات

خطی شده فضای حالت به صورت زیر درمی آیند:

$$\dot{X} = J_x\, X + J_u\, U \tag{15}$$

ماتریس‌های J_x و J_u ماتریس‌های ژاکوبین هستند که به ترتیب به صورت زیر تعریف می‌شوند:

$$J_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{16}$$

$$J_u = \begin{bmatrix} \frac{\partial f_1}{\partial U_1} & \cdots & \frac{\partial f_1}{\partial U_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial U_1} & \cdots & \frac{\partial f_n}{\partial U_n} \end{bmatrix} \tag{17}$$

که توابع f_i، سمت راست معادلات (8) و (11) هستند. مقادیر ماتریس‌های J_xو J_u با استفاده از روابط (16) و (17) حول حالت هاور که تمامی متغیرهای حالت و ورودی‌های کنترل صفر هستند ($\mathrm{x}_{i,i=1,2,\dots,8} = 0$, $U_{i,i=1,2,3,4} = 0$) به صورت زیر محاسبه می‌شوند:

$$J_x = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{18}$$

$$J_u = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{I_z} \\ 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 \end{bmatrix} \tag{19}$$

$$X = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6]^{\mathrm{T}}$$
$$U = [U_1 \quad U_2 \quad U_3 \quad U_4]^{\mathrm{T}} \tag{20}$$

و ماتریس‌های Q و R:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 90 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{21}$$

$$R = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 60 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \tag{22}$$

با حل معادله ریکاتی زیر و با در نظر گرفتن $A = J_x\, , B = J_u$:

$$A^{\mathrm{T}}P + PA - PBR^{-1}B^{\mathrm{T}}P + Q = 0 \tag{23}$$

و با جایگذاری P در رابطه زیر:

$$K = R^{-1}B^{\mathrm{T}}P \tag{24}$$

برای ماتریس بهره حالت داریم:

$$K = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1.34 & 0.948 \\ 0.141 & 0.023 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.129 & 0.022 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.707 & 0.082 \end{bmatrix} \tag{25}$$

و با این بهره بردار ورودی کنترل در مدل دینامیک دورانی را به دست می‌آوریم:

$$U = -KX \tag{26}$$

2-4- کنترل‌کننده گام به عقب

کنترل‌کننده گام به عقب در مرجع [15] به طور کامل توضیح داده شده است و در اینجا به اختصار بیان می‌شود. معادلات حالتی که در این روش به کار گرفته می‌شوند باید دارای فرم خاصی باشند. در واقع این معادلات از دو دسته تشکیل می‌شوند که به فرم زیر می‌باشند.

$$\begin{aligned} \dot{x}_1 &= f_1(x_1) + g_1(x_1)x_2 \\ \dot{x}_2 &= f_2(x_1, x_2) + g_2(x_1, x_2)u \end{aligned} \tag{27}$$

همان طور که مشاهده می‌شود به جز در انتهای معادله در دسته معادلات اول فقط متغیر x_1 وجود دارد. اصطلاحاً گفته می‌شود معادله اول نسبت به متغیر x_2 افاین[1] می‌باشد. همین طور معادله دوم نیز نسبت به ورودی کنترلی افاین است. اگر معادلات به فرم بالا باشند به راحتی می‌توان کنترل‌کننده گام به عقب را برای سیستم طراحی کرد.

کل روند طراحی گام به عقب دو مرحله می‌باشد. معادلات فضای حالت (8) را در نظر می‌گیریم و کنترل‌کننده گام به عقب را طبق روشی که در ادامه توضیح داده می‌شود طراحی می‌کنیم.

ابتدا معادلات فضای حالت (8) را به فرم زیر که در واقع همان رابطه (27) است در می‌آوریم:

$$\begin{aligned} \dot{\eta} &= f(\eta) + G(\eta)\zeta \\ \dot{\zeta} &= f_a(\eta, \zeta) + G_a(\eta, \zeta)u \end{aligned} \tag{28}$$

که در این صورت خواهیم داشت:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_3 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_4 \\ x_6 \end{bmatrix}$$
$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_4 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} a_1x_4x_6 - a_4(x_4 + x_1x_6)\Omega \\ a_2x_2x_6 + a_5(x_2 - x_3x_6)\Omega \\ a_3x_2x_4 + a_5x_1x_2\Omega \end{bmatrix} + \begin{bmatrix} b_1 & 0 & b_3x_3 \\ 0 & b_2 & -b_3x_1 \\ 0 & b_2x_1 & b_3 \end{bmatrix} \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{29}$$

به منظور رساندن متغیرهای حالت توسط ورودی کنترل به هر مقدار مطلوب، متغیرهای برداری جدید به صورت زیر تعریف می‌شوند:

$$\eta = \begin{bmatrix} z_1 \\ z_3 \\ z_5 \end{bmatrix} = \begin{bmatrix} x_{1d} - x_1 \\ x_{3d} - x_3 \\ x_{5d} - x_5 \end{bmatrix} \tag{30}$$

$$\zeta = \begin{bmatrix} z_2 \\ z_4 \\ z_6 \end{bmatrix} = \begin{bmatrix} x_{2d} - x_2 \\ x_{4d} - 4 \\ x_{6d} - x_6 \end{bmatrix} \tag{31}$$

که مقادیر x_{1d} تا x_{6d} متغیرهای حالت مطلوب هستند. با این تغییر متغیر و اعمال برخی تغییرات خواهیم داشت:

$$f(\eta) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, G(\eta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{32}$$

$$f_a(\eta, \zeta) = \begin{bmatrix} \ddot{x}_{1d} - a_1x_4x_6 + a_4(x_4 + x_1x_6)\Omega \\ \ddot{x}_{3d} - a_2x_2x_6 - a_5(x_2 - x_3x_6)\Omega \\ \ddot{x}_{5d} - a_3x_2x_4 - a_5x_1x_2\Omega \end{bmatrix} \tag{33}$$

$$G_a(\eta, \zeta) = -\begin{bmatrix} b_1 & 0 & b_3x_3 \\ 0 & b_2 & -b_3x_1 \\ 0 & b_2x_1 & b_3 \end{bmatrix} \tag{34}$$

و در نهایت:

$$u = \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{35}$$

[1] affine

[DOR: 20.1001.1.10275940.1396.17.7.47.9]

حال با آماده بودن فرم صحیح معادلات به طراحی کنترلر می‌پردازیم. در گام اول فرض می‌کنیم متغیر ζ ورودی کنترلی دسته معادلات اول باشد. دسته معادلات اول را با استفاده از کنترل‌کننده $\zeta=\phi(\eta)$ طوری پایدار می‌کنیم که بتوان یک تابع لیاپانوف برای همین دسته معادلات پیدا کرد:

$$\phi(\eta)=\begin{bmatrix}-k_1 & 0 & 0\\ 0 & -k_3 & 0\\ 0 & 0 & -k_5\end{bmatrix}\begin{bmatrix}z_1\\ z_3\\ z_5\end{bmatrix}=\begin{bmatrix}-k_1z_1\\ -k_3z_3\\ -k_5z_5\end{bmatrix}$$
$$k_1,k_3,k_5>0 \tag{36}$$

سپس از تئوری لیاپانوف استفاده می‌کنیم و تابع لیاپانوف را تعریف می‌کنیم. در این گام تابع لیاپانوف $V(\eta)$ را به صورت زیر در نظر می‌گیریم:

$$V(\eta)=\frac{1}{2}\eta^{\mathrm{T}}\eta=\frac{1}{2}(z_1^2+z_3^2+z_5^2) \tag{37}$$

مشتق $V(\eta)$ با توجه به روابط (36) و (37) به صورت زیر است:

$$\begin{aligned}\dot V(\eta)&=z_1\dot z_1+z_3\dot z_3+z_5\dot z_5=z_1z_2+z_3z_4+z_5z_6\\ &=-k_1{z_1}^2-k_3{z_3}^2-k_5{z_5}^2\\ &\rightarrow \dot V(\eta)<0\end{aligned} \tag{38}$$

بنابراین مشتق تابع لیاپانوف منفی است. در گام دوم تابع لیاپانوف را به صورت زیر در نظر می‌گیریم:

$$\begin{aligned}V_a&=V(\eta)+\frac{1}{2}[\zeta-\phi(\eta)]^{\mathrm{T}}[\zeta-\phi(\eta)]=\\ &=\frac{1}{2}(z_1^2+z_3^2+z_5^2)+\\ &+\frac{1}{2}((z_2+k_1z_1)^2+(z_4+k_3z_3)^2+(z_6+k_5z_5)^2)\end{aligned} \tag{39}$$

و در نهایت با در نظر گرفتن ورودی کنترلی u به صورت زیر مشتق تابع لیاپانوف در گام دوم را منفی می‌کنیم و دسته معادلات دوم نیز پایدار می‌شوند:

$$u={G_a}^{-1}\left[\frac{\partial\phi}{\partial\eta}(f+G\zeta)-\left(\frac{\partial V}{\partial\eta}\right)^{\mathrm{T}}-f_a-K(\zeta-\phi)\right] \tag{40}$$

که در آن ماتریس K به صورت زیر در نظر گرفته می‌شود:

$$K=\begin{bmatrix}k_2 & 0 & 0\\ 0 & k_4 & 0\\ 0 & 0 & k_6\end{bmatrix} \tag{41}$$

و در نهایت با جایگذاری روابط (31)، (32)، (33)، (34)، (36)، (37) و (41) در رابطه (40) داریم:

$$\begin{bmatrix}U_2\\ U_3\\ U_4\end{bmatrix}=\begin{bmatrix}b_1 & 0 & b_3x_3\\ 0 & b_2 & -b_3x_1\\ 0 & b_2x_1 & b_3\end{bmatrix}^{-1}\begin{bmatrix}\ddot x_{1d}+z_1+k_1z_2+k_2z_2\\ +k_1k_2z_1-a_1x_4x_6\\ +a_4(x_4+x_1x_6)\Omega\\ \ddot x_{3d}+z_3+k_3z_4+k_4z_4\\ +k_3k_4z_3-a_2x_2x_6\\ -a_5(x_2-x_3x_6)\Omega\\ \ddot x_{5d}+z_5+k_5z_6+k_6z_6\\ +k_5k_6z_5-a_3x_2x_4\\ -a_5x_1x_2\Omega\end{bmatrix} \tag{42}$$

و مشابه همین روش را روی معادله ارتفاع (11) اعمال می‌کنیم. با تعاریف زیر:

$$\begin{aligned}\eta&=z_7=x_{7d}-x_7\\ \zeta&=z_8=x_{8d}-x_8\end{aligned} \tag{43}$$

که در آن x_{7d} و x_{8d} مقادیر مطلوب هستند، به روابط زیر می رسیم:

$$\zeta=\phi(\eta)=-k_7z_7\ ,k_7>0 \tag{44}$$

$$V(\eta)=\frac{1}{2}{z_7}^2, V_a=\frac{1}{2}{z_7}^2+\frac{1}{2}(z_8+k_8z_8)^2 \tag{45}$$

$$f=0, G=1, f_a=\ddot x_{7d}+g,\ \ G_a=-\frac{1}{m}\cos x_1\cos x_3 \tag{46}$$

و در نهایت U_1 به صورت رابطه (47) به دست می‌آید:

$$U_1=\frac{m}{\cos x_1\cos x_3}(k_7z_{\mathbf{8}}+z_{\mathbf{7}}+g+\ddot x_{7d}+k_{\mathbf{8}}z_{\mathbf{8}}+k_7k_8z_{\mathbf{7}}) \tag{47}$$

3-4- الگوریتم بهینه‌سازی ازدحام ذرات[1]

الگوریتم بهینه‌سازی ازدحام ذرات یکی از روش‌های فراابتکاری است که در حل مسائل بهینه‌سازی پیوسته موفقیت بسیاری از خود نشان داده است. این روش در سال 1995 توسط کندی[2] و ابرهارت[3] معرفی شد [16]. این الگوریتم از روی زندگی جمعی و گروهی پرندگان الهام می‌گیرد تا به راه حل بهینه برسد. با استفاده از این روش می‌توان با مسائلی که جواب آنها یک نقطه یا سطح در فضای n بعدی می‌باشد، برخورد نمود. در این‌چنین فضایی، فرضیاتی مطرح می‌شود و یک سرعت ابتدایی به آنها اختصاص داده می‌شود، همچنین کانال‌های ارتباطی بین ذرات درنظر گرفته می‌شود. سپس این ذرات در فضای پاسخ حرکت می‌کنند، و نتایج حاصله بر مبنای یک «ملاک شایستگی» پس از هر بازه زمانی محاسبه می‌شود. با گذشت زمان، ذرات به سمت ذراتی که دارای ملاک شایستگی بالاتری هستند و در گروه ارتباطی یکسانی قرار دارند، شتاب می‌گیرند.

این الگوریتم با یک گروه از جواب‌های تصادفی شروع به کـار می‌کند. سپس برای یافتن جواب بهینه در فضای مسئله با بـه روز کردن موقعیت و سرعت هر ذره به جستجو می‌پردازد. هر ذره بـه صورت چندبعدی با دو مقدار $x_{i,j}$ و $v_{i,j}$ کـه بـه ترتیـب مکـان و سرعت مربوط به بعد j ام از i امین ذره هستند تعریف می‌شود. در هر مرحله از حرکت جمعیت، هر ذره با توجه به دو مقدار بهترین به روز می‌شود. اولـین مقـدار بهتـرین جـواب از لحـاظ شایستگی است که تاکنون برای هر ذره به طور جداگانه به دست آمده است. این مقدار بهترین تجربه فردی است که pbest نامیده می‌شود. مقدار بهترین دیگر که توسط PSO به دست می‌آید، بهترین مقداری است که تاکنون توسط تمام ذره‌ها در میان جمعیت به دست آمده است. این مقدار بهترین تجربه گروهی است که gbest نامیده می‌شود. پـس از یـافتن دو مقـدار pbest و gbest هر ذره سرعت v و مکان x جدید خود را با روابط (48) به روز می‌کند:

$$\begin{aligned}&{v_{i,j}}^{t+1}=wv_{i,j}^t+c_1r_1\left(p_{i,j}^t-x_{i,j}^t\right)+c_2r_2\left(g_j^t-x_{i,j}^t\right)\\ &x_i^{t+1}=x_i^t+v_i^{t+1}\end{aligned} \tag{48}$$

به طوری که w وزن اینرسی، c_1,c_2 ضرایب شتاب و r_1,r_2 اعداد تصادفی در بازه (0,1) می‌باشند. همچنین p و g به ترتیب pbest و gbest می‌باشند. 8 پارامتر $k_{i,i=1\dots8}$ را با الگوریتم ازدحام ذرات برای مینیمم کردن تابع هزینه به دست می‌آوریم. پارامترهای الگوریتم ازدحام ذرات در این مقاله به صورتی که در جدول 2 مشاهده می‌شود در نظر گرفته شده‌اند.

نحوه انتخاب بازه جست و جو برای پارامترها به این صورت است که در کنترلر گام به عقب، به طور کلی با افزایش پارامترهای $k_{i,i=1\dots8}$ مقدار خطا کاهش‌یافته و تلاش کنترلی افزایش می‌یابد. بنابراین در تابع هزینه در نظر گرفته شده برای کمینه‌سازی، هر چه قدر درایه‌های ماتریس R بزرگ تر باشند، پارامترهای کوچک‌تری توسط الگوریتم یافت می‌شوند و بالعکس هر چه درایه‌های ماتریس Q بزرگ‌تر باشند، پارامترهای بزرگ‌تری تعیین می‌شوند. بنابراین انتخاب بازه جست و جو برای بهینه‌سازی، بستگی به ماتریس‌های وزنی Q و R خواهد داشت. در این مقاله بازه جست و جو

[1] Particle Swarm Optimization (PSO)
[2] Kennedy
[3] Eberhart

[0-30] در نظر گرفته شده است. به طور کلی در الگوریتم‌های بهینه‌سازی، هر چه قدر بازه جست و جو مناسب تر انتخاب شود پاسخ در تعداد تکرار[1]های کمتری به مقدار بهینه خود همگرا می‌شود.

شکل 3 مقدار کمترین تابع هزینه بر حسب تعداد تکرار را نشان می‌دهد.

همان طور که در شکل 3 مشاهده می‌شود مقدار مینیمم تابع هزینه به کمترین مقدار ممکن خود همگرا شده است. ضرایب به دست آمده در آخرین تکرار به ترتیب زیر هستند:

$$k_{i,i=1\ldots8}=\begin{bmatrix}10.51\\9.32\\14.30\\11.91\\13.78\\12.78\\3.19\\3.13\end{bmatrix} \tag{49}$$

5- نتایج

برای مشاهده نتایج، به ارتفاع و زوایای اویلر مقدار اولیه داده شده است. نتایج به دست آمده از هر دو کنترلر در شکل‌های 4 تا 7 قابل مشاهده هستند.

مقدار تابع هزینه برای کنترلر گام به عقب 602.41 و برای کنترلر LQR 612.30 به دست آمده است. نکته‌ای که باید به آن توجه شود، بررسی ماکزیمم فرمان ورودی به موتور برای کنترل است تا یقین حاصل شود که کوادروتور مورد بررسی توانایی کنترل مقادیر اولیه و یا اغتشاش‌های مورد نظر را داشته باشد.

در کوادروتور مورد نظر فرمان به موتورها با دادن مقدار به پارامتری که n نامیده می‌شود امکان‌پذیر است و n می‌تواند از 0 تا 500 تغییر کند [11].

جدول 2 پارامترهای الگوریتم ازدحام ذرات

Table 2 Parameters of PSO algorithm

پارامترهای PSO	مقدار
تعداد متغیرهای فضای جست و جو ($k_{i,i=1\ldots8}$)	8
تعداد اعضای جمعیت	25
تعداد تکرارها	40
ضرایب شتاب c_1, c_2	2.05
وزن اینرسی w	0.73
بازه جست و جو برای پارامترها	[0-30]
زمان شبیه‌سازی	5 sec

Fig. 3 Best Cost per iteration

شکل 3 کمترین تابع هزینه بر حسب تکرار

[1] Iteration

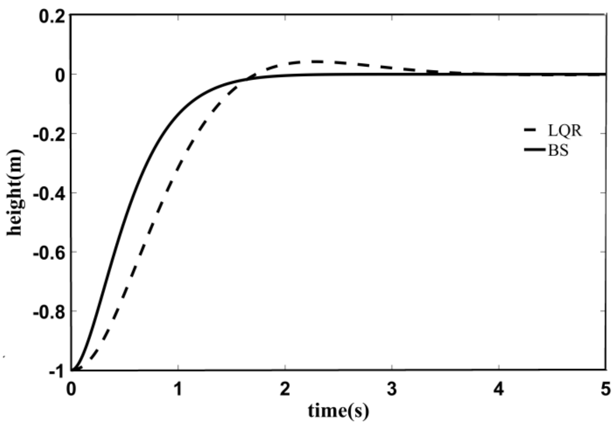

Fig. 4 Altitude of the quadrotor per time

شکل 4 ارتفاع کوادروتور بر حسب زمان

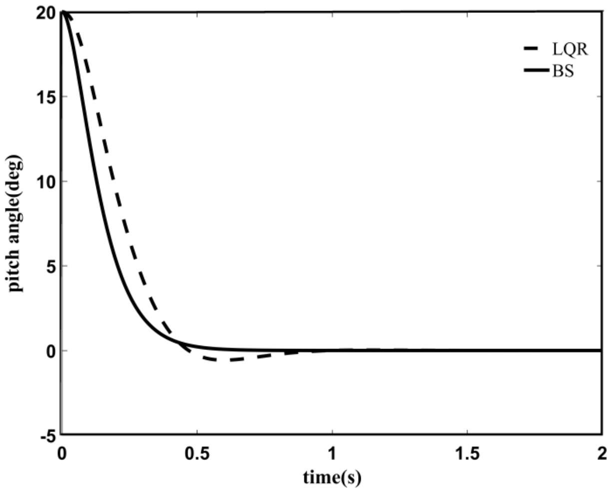

Fig. 5 Pitch angle per time

شکل 5 زاویه پیچ بر حسب زمان

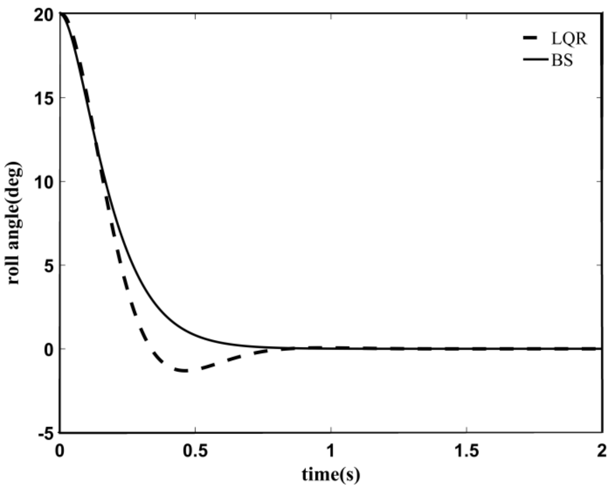

Fig. 6 Roll angle per time

شکل 6 زاویه رول بر حسب زمان

در مرجع [11]، با محاسبه نسبت دنده کوادروتور و بررسی رابطه بین سیگنال n و سرعت زاویه‌ای موتورها، رابطه بین فرمان ورودی به موتور و تراست به دست آمده است:

$$n = 1.3812\sqrt{\frac{T}{5.7231\times10^{-6}}} - 173.895 \quad (50)$$

شکل‌های 8 و 9 فرمان ورودی n برای موتورهای 1 و 3 را که فرمان ورودی بزرگتری نسبت به موتورهای 2 و 4 داشته‌اند را نشان می‌دهند.

همان طور که مشاهده می‌شود ماکزیمم فرمان ورودی به موتورها، در هر دو کنترلر از مقدار ماکزیمم 500 کمتر بوده است.

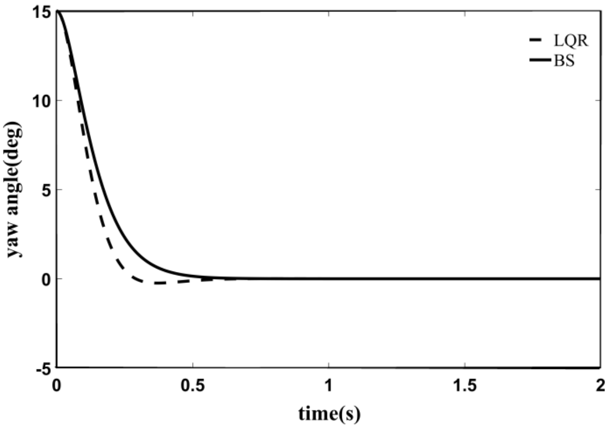

Fig. 7 Yaw angle per time

شکل 7 زاویه یاو بر حسب زمان

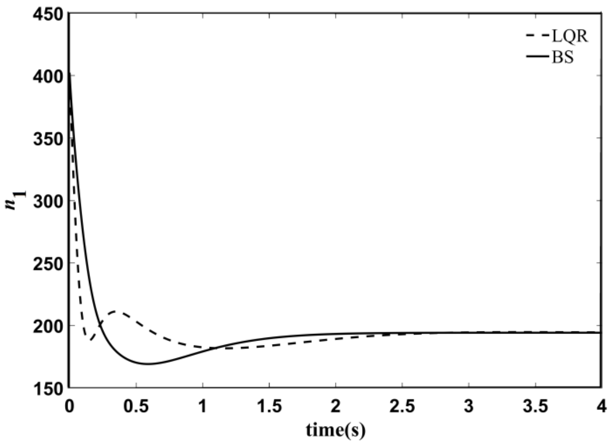

Fig. 8 Command to motor1 per time

شکل 8 فرمان ورودی به موتور 1 بر حسب زمان

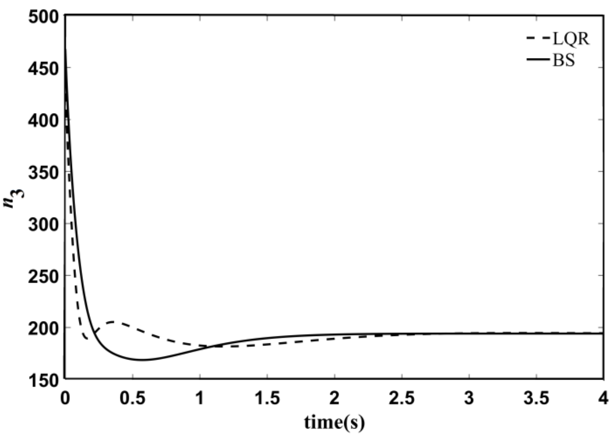

Fig. 9 Command to motor3 per time

شکل 9 فرمان ورودی به موتور 3 بر حسب زمان

جدول 3 تابع هزینه به ازای مقادیر مختلف وزن‌های ماتریس‌های Q و R

Table 3 Cost unction for different weights of Q and R matrices

LQR	گام به عقب	w_6	w_5	w_4	w_3	w_2	w_1
184.36	95.58	1	35	25	100	5	1
334.76	325.26	5	32	22	95	8	2
720.70	710.48	10	30	20	90	10	4
1032.56	1024.80	15	28	18	85	12	6
144.42	107.99	1	30	30	105	1	1
152.26	110.0	1	35	35	110	1	1
216.30	121.53	1	45	40	120	5	1
236.34	124.55	1	50	45	130	5	1
256.76	127.80	1	60	50	140	5	1
121.74	90.20	1	80	100	100	1	1
85.28	85.14	1	5	2	100	1	1
612.30	602.41	10	60	50	90	5	1

6- نتیجه‌گیری

این مقاله با هدف بهینه‌سازی یک کنترلر غیرخطی با استفاده از الگوریتم هوشمند و مقایسه آن با کنترلر LQR که خود یک کنترلر بهینه است و نیز مقایسه توانایی دو کنترلر در کمینه‌سازی تابع هزینه یکسان انجام شد. برای نتیجه‌گیری بهتر، ماتریس‌های Q و R در تابع هزینه را تغییر داده و مقدار تابع هزینه برای هر دو کنترلر محاسبه شده‌اند. نتایج در جدول 3 قابل مشاهده هستند. برای سادگی نمایش در جدول، ماتریس‌های وزنی به صورت زیر در نظر گرفته شده‌اند:

$$Q = \begin{bmatrix} w_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (51)$$

$$R = \begin{bmatrix} w_4 & 0 & 0 & 0 \\ 0 & w_4 & 0 & 0 \\ 0 & 0 & w_5 & 0 \\ 0 & 0 & 0 & w_6 \end{bmatrix} \quad (52)$$

بنابراین حداقل مقدار تابع هزینه در کنترلر گام به عقب به ازای ماتریس‌های وزنی مختلف کمتر از کنترلر LQR است و این نشان می‌دهد کنترلر گام به عقب به کمک الگوریتم ازدحام ذرات در مینیمم کردن تابع هزینه موفق‌تر بوده است.

7- مراجع

[1] A. Abdollahi, A. Foruzan Tabar, H. Khodadadi, Optimal controller design for quadrotor by genetic algorithm with the aim of optimizing the response and control input signals, *Science Journal (CSJ)*, Vol. 36, No. 3, pp. 135-147, Special Issue, 2015.

[2] J. Estevez, M. Lopez, M. Grana, Particle swarm optimization quadrotor control for cooperative aerial transportation of deformable linear objects, *International Journal of Cybernetics and Systems*, Vol. 47, Issue 1-2, pp. 4-16, 2016.

[3] M. Ariffanan Mohd Basri, K. Danapalasingam, A. Rashid Husain, Design and optimization of backstepping controller for an underactuated autonomous quadrotor unmanned aerial vehicle, *Transactions of FAMENA*, Vol. 38, No. 3, pp. 27-44, 2014.

[4] Y. Fouad, O. Bouhali, PSO optimization of integral backstepping controller for quadrotor attitude stabilization, *3rd IEEE International Conference on Systems and Control (ICSC'13)*, October, 2013.

[5] E. Abbasi Seidabad, S. Vandaki, A. Vahidian Kamyad, Designing fuzzy pid controller for quadrotor, *International Journal of Advanced Research in Computer Science & Technology*, Vol. 2, Issue. 4, pp. 221-227, 2014.

[6] S. Zeghlache, D. Saigaa, Backstepping sliding mode controller improved with fuzzy logic: Application to the quadrotor helicopter, *Archives of Control Sciences*, Vol. 22(LVIII), No. 3, pp. 315–342, 2012.

[7] Z. X. Liu, C. Yuan, Y. M. Zhang, A learning-based fuzzy lqr control scheme for height control of an unmanned quadrotor helicopter, *International Conference on Unmanned Aircraft System*, 27-30 May, 2014.

[8] M. Fatan, B. Lavi Sefidgari, A. Vatankhah Barenji, An adaptive neuro pid for controlling the altitude of quadcopter robot, *International Conference on Methods and Models in Automation and Robotics*, 26-29 August, 2013.
[9] T. Madani, A. Benallegue, Adaptive control via backstepping technique and neural networks of a quadrotor helicopter, *The International Federation of Automatic Control*, July 6-11, 2008.
[10] M. Navabi, H. R. Mirzaei, Dynamic modelling and nonlinear adaptive control of mesicopter flight, *Modares Mechanical Engineering*, Vol. 15, No. 5, pp.1-12, 2015. (In Persian فارسی)
[11] N. Parhizkar, A. Naghash, M. Naghshineh, Experimental investigation of rotational control of a constrained quadrotor using backstepping method, *International Micro Air Vehicle Conference*, 17-21 October, 2016.
[12] S. N. Ghazbi, A. L. I. Akbar, M. Reza, Quadrotor: Full dynamic modeling, nonlinear simulation and control of attitudes degrees of freedom and the movement, *Indian Journal of Scientific Research*, Vol. 1, No. 2, pp. 759-769, 2014.
[13] A. Naghash, M. Naghshineh, A. Honari, Minimum time trajectory optimization for flying a quadrotor in an 8 - shaped path, *International Micro Air Vehicle Conference*, 17-20 September, 2013.
[14] E. Davoodi, M. Rezaei, Dynamic modeling, simulation and control of a quadrotor using MEMS sensors' experimental data, *Modares Mechanical Engineering*, Vol. 14, No. 3, pp. 175-184, 2014. (In Persian فارسی)
[15] H. Khalil, *Nonlinear Systems*, pp. 588-601, Prentice Hall, 2002.
[16] J. Kennedy, R. Eberhart, Particle swarm optimization, *IEEE International Conference on Neural Networks*, Vol. 4, 27 November-1 December, 1995.