

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



دانشکده مکانیک

گروه مکاترونیک

طراحی کنترل کننده فازی بازوی رباتیک با مفاصل منعطف توسط الگوریتم بهینه سازی پرندگان

دانشجو: زهره صفرچراتی

استاد راهنما:

جناب آقای دکتر محمد مهدی فاتح

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

شهریور ۹۲



دانشگاه صنعتی شاهرود

مدیریت تحصیلات تکمیلی

فرم شماره (6)

باسمه تعالی

شماره:

تاریخ:

ویرایش:

فرم صورت جلسه دفاع از پایان نامه تحصیلی دوره کارشناسی ارشد

با تأییدات خداوند متعال و با استعانت از حضرت ولی عصر (عج) نتیجه ارزیابی جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای **زهرة صفرچراتی** رشته **مکاترونیک گرایش مکاترونیک** تحت عنوان **طراحی کنترل کننده فازی بازوی ربانیک با مفاصل منعطف توسط الگوریتم بهینه سازی پرندگان** که در تاریخ **92/6/27** با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح ذیل اعلام می گردد:

<input type="checkbox"/> قبول (با درجه : خوب - امتیاز ۱۷,۷۵)	<input type="checkbox"/> دفاع مجدد	<input type="checkbox"/> مردود
--	------------------------------------	--------------------------------

2- بسیار خوب (18 - 18/99)

1- عالی (19 - 20)

4- قابل قبول (14 - 15/99)

3- خوب (16 - 17/99) ✓

5- نمره کمتر از 14 غیر قابل قبول

امضاء	مرتبه علمی	نام و نام خانوادگی	عضو هیأت داوران
	استاد	مهدی صدیق پناه	1- استاد راهنما
			2- استاد مشاور
	استاد	مصطفی محمدی	3- نماینده شورای تحصیلات تکمیلی
	استاد	علیرضا اهری فرد	4- استاد ممتحن
	استاد	سعید آبرار	5- استاد ممتحن

رئیس دانشکده:

امضاء:

تعهد نامه

اینجانب **زهره صفرچراتی** دانشجوی دوره کارشناسی ارشد رشته **مکاترونیک** دانشکده مکانیک دانشگاه صنعتی شاهرود نویسنده پایان نامه طراحی کنترل کننده فازی بازوی رباتیک با مفاصل منعطف توسط الگوریتم بهینه سازی پرندگان تحت راهنمایی دکتر محمد مهدی فاتح متعهد می شوم.

- تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است.
- در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است.
- مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است.
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « دانشگاه صنعتی شاهرود » و یا « Shahrood University of Technology » به چاپ خواهد رسید.
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه رعایت می گردد.
- در کلیه مراحل انجام این پایان نامه ، در مواردی که از موجود زنده (یا بافتهای آنها) استفاده شده است ضوابط و اصول اخلاقی رعایت شده است.
- در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل رازداری ، ضوابط و اصول اخلاق انسانی رعایت شده است

تاریخ

امضای دانشجو

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج، کتاب، برنامه های رایانه ای، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد. این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود.
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.

* متن این صفحه نیز باید در ابتدای نسخه های تکثیر شده پایان نامه وجود داشته باشد .

تقدیم به پدر و مادر عزیزم

و همسر مهربانم

تقدیر و تشکر :

بر خود لازم می‌دانم از زحمات بی دریغ استاد دکتر محمد مهدی فاتح کمال تشکر و قدرانی را داشته باشم . بی شک بدون راهنمایی های دلسوزانه این بزرگوار هرگز این پایان نامه به نقطه مطلوب نمی‌رسید.

چکیده

در این پایان نامه ابتدا مرور اجمالی بر مسئله انعطاف در مفاصل و مشکلات ناشی از آن می‌شود. از آنجا که کنترل فازی عملکرد مناسبی را در کنترل ربات با مفاصل منعطف نشان داده است، به معرفی و طراحی کنترل فازی روی بازوی رباتیک با مفاصل منعطف می‌پردازیم. با وجود مزیت های فراوان کنترل کننده فازی، تنظیم پارامترهای کنترل فازی توسط فرد خبیره و پیچیدگی تحلیل پایداری آن همواره مشکلاتی را در برداشته است. بنابراین، استفاده از الگوریتم های هوشمند در طراحی کنترل فازی بهینه مطرح شده است. در میان الگوریتم‌های بهینه‌سازی هوشمند، الگوریتم بهینه سازی پرندگان پاسخ مناسبی در بهینه سازی مسائل غیرخطی و پیچیده با سرعت همگرایی بالا داشته است. در این پایان نامه از الگوریتم بهینه‌سازی پرندگان برای طراحی کنترل کننده ای با تضمین پایداری، استفاده شده است. در این راستا، طراحی کنترل کننده بهینه خطی با تضمین پایداری با بکارگیری الگوریتم بهینه سازی پرندگان برای سیستم های خطی مانند بازوی رباتیک تعمیم می‌یابد. بعد از آن با بکارگیری الگوریتم بهینه سازی پرندگان، کنترل کننده فازی به گونه ای طراحی شده است که پایداری آن تضمین گردد و کنترل کننده عملکرد بهینه داشته باشد. هر دو روش طراحی کنترل کننده فازی بهینه به صورت خارج خط و کنترل لحظه‌ای ارائه می‌گردد. نوآوری پایان نامه در تضمین پایداری کنترل کننده فازی توسط الگوریتم بهینه سازی پرندگان می‌باشد. مزیت آن سادگی طراحی در مقایسه با روش مرسوم پایداری لیاپانوف است که اجرای آن روی بازوی رباتیک با مفاصل منعطف مجهز به موتورهای الکتریکی با حضور کنترل کننده فازی دشوار است. همچنین طراحی کنترل کننده بهینه خطی با تضمین پایداری با بکارگیری الگوریتم بهینه سازی پرندگان برای سیستم های خطی و تعمیم آن برای سیستم های غیرخطی مانند بازوی رباتیک جدید است.

کلمات کلیدی:

بازوی ربات با مفاصل منعطف، الگوریتم بهینه سازی پرندگان، کنترل فازی، کنترل بهینه مربعی خطی

فهرست مطالب

شماره	عنوان	شماره بخش	صفحه
۱	فصل اول مقدمه		
۲	۱-۱ مرور کارهای قبلی		
۲	۱-۱-۱ مدل‌سازی ربات با مفاصل منعطف		
۳	۱-۱-۲ روش های کنترلی قبلی برای ربات با مفاصل منعطف		
۴	۱-۱-۳ استراتژی کنترل ولتاژ		
۵	۱-۱-۴ کنترل فازی		
۷	۱-۱-۵ الگوریتم بهینه سازی پرنندگان		
۱۰	۲-۱ اهداف پایان نامه		
۱۱	۳-۱ نمای کلی پایان نامه		
۱۲	فصل دوم مدل‌سازی ربات با مفاصل منعطف		
۱۳	۱-۲ مقدمه		
۱۳	۲-۲ معادلات دینامیکی ربات های با مفاصل منعطف		
۱۵	۳-۲ دینامیک محرکه ها		
۱۷	۴-۲ معادلات فضای حالت حاکم بر سیستم بازوی رباتیک با مفاصل منعطف		

۱۹	فصل سوم الگوریتم بهینه سازی پرندگان
۲۰	۱-۳ مقدمه
۲۰	۲-۳ شرح الگوریتم بهینه سازی پرندگان
۲۱	۱-۲-۳ الگوریتم بهینه سازی پرندگان در حالت استاندارد
۲۳	۲-۲-۳ مؤلفه های سرعت
۲۴	۳-۲-۳ پارامترهای اصلی
۲۵	۳-۳ تاثیر میزان جمعیت و ضرایب یادگیری در عملکرد الگوریتم بهینه سازی پرندگان
۲۶	۴-۳ انواع ساختارهای همسایگی در الگوریتم PSO
۲۹	۵-۳ توسعه الگوریتم بهینه سازی پرندگان
۲۹	۱-۵-۳ الگوریتم بهینه سازی پرندگان با سرعت آشفته‌گی تفاضلی
۳۰	۲-۵-۳ الگوریتم بهینه سازی پرندگان توسعه یافته STS-PSO
	فصل چهارم طراحی کنترل کننده بهینه خطی با تضمین پایداری با استفاده از الگوریتم بهینه سازی پرندگان
۳۲	پرندگان
۳۳	۱-۴ مقدمه
۳۳	۲-۴ تئوری کنترل بهینه مربعی خطی
۳۵	۳-۴ توسعه کنترل بهینه مربعی خطی با استفاده از الگوریتم بهینه سازی پرندگان
۳۶	۴-۴ تعمیم تئوری کنترل مربعی خطی به سیستم های غیرخطی

۳۹	۵-۴ شبیه سازی
۵۰	۶-۴ خلاصه بحث و نتیجه گیری
۵۱	فصل پنجم کنترل فازی بازوی ربات با مفاصل منعطف
۵۲	۱-۵ مقدمه
۵۲	۲-۵ طراحی کنترلکننده فازی توسعه یافته
۵۳	۱-۲-۵ کنترل کننده فازی PI
۵۵	۲-۲-۵ پایگاه استنتاج کنترل کننده فازی PI
۵۶	۳-۲-۵ کنترل کننده فازی PI اصلاح شده
۵۷	۳-۵ شبیه سازی
۶۰	۴-۵ خلاصه بحث و نتیجه گیری
۶۲	فصل ششم کنترل فازی بهینه بازوی ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان
۶۳	۱-۶ مقدمه
۶۴	۲-۶ مفاهیم پایداری
۶۵	۳-۶ کنترل کننده پایدار با استفاده از الگوریتم بهینه سازی پرندگان
۶۷	۴-۶ کنترل خطی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان
۶۸	۵-۶ کنترل فازی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه‌سازی پرندگان
۷۰	۶-۶ شبیه سازی

۷۰	۱-۶-۶ پیاده سازی کنترل خطی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان
۷۶	۲-۶-۶ پیاده سازی کنترل فازی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان
۸۱	فصل هفتم کنترل بهینه فازی بازوی ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان به صورت بر خط
۸۲	۱-۷ مقدمه
۸۳	۲-۷ تخمین تابع با استفاده از روش بهینه سازی پرندگان
۸۵	۳-۷ توابع موثر در کنترل ربات با مفاصل منعطف
۸۵	۱-۳-۷ تخمین تابع شتاب موتور
۸۷	۲-۳-۷ تخمین تابع مشتق چهارم موقعیت رابط
۸۷	۳-۳-۷ تخمین تابع موقعیت موتور
۸۸	۴-۷ ساختار سیستم فازی
۹۱	۵-۷ شبیه سازی
۹۴	۶-۷ نتیجه گیری
۹۵	فصل هشتم نتیجه گیری و پیشنهادات
۹۶	۱-۸ نتیجه گیری
۹۷	۲-۸ پیشنهادات
۹۸	منابع و ماخذ

فهرست شکل ها و جدول ها

۱۸	شکل (۲-۱) ساختار ربات با مفاصل منعطف
۲۴	شکل (۳-۱) جهت گیری ذرات در الگوریتم بهینه سازی پرندگان
۲۷	شکل (۳-۲) گراف حلقوی
۲۸	شکل (۳-۳) گراف ارتباط کامل
۲۸	شکل (۳-۴) شبکه ستاره
۲۹	شکل (۳-۵) شبکه درختی
۴۰	شکل (۴-۱) نرم خطای تنظیم در نقطه مطلوب
۴۰	شکل (۴-۲) تنظیم در نقطه مطلوب
۴۱	شکل (۴-۳) روند کاهش تابع هدف
۴۲	شکل (۴-۴) نرم خطای ردگیری سیستم خطی
۴۲	شکل (۴-۵) مسیر ردگیری سیستم خطی
۴۳	شکل (۴-۶) روند کاهش تابع هدف در الگوریتم PSO
۴۳	شکل (۴-۷) سیگنال کنترلی ورودی
۴۴	جدول (۴-۱) مشخصات موتورها
۴۴	جدول (۴-۲) مشخصات ربات

- ۴۵ شکل (۴-۸) بلوک دیاگرام کنترل کننده پیشنهادی
- ۴۶ شکل (۴-۹) مسیر ردگیری رابط یک
- ۴۶ شکل (۴-۱۰) مسیر ردگیری رابط دو
- ۴۷ شکل (۴-۱۱) نرم خطای ردگیری رابط یک
- ۴۷ شکل (۴-۱۲) نرم خطای ردگیری رابط دو
- ۴۸ شکل (۴-۱۳) سیگنال کنترلی ورودی رابط یک
- ۴۹ شکل (۴-۱۴) سیگنال کنترلی ورودی رابط دو
- ۴۹ شکل (۴-۱۵) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان
- ۵۴ شکل (۵-۱) بلوک دیاگرام سیستم فازی
- ۵۴ شکل (۵-۲) توابع تعلق ورودی
- ۵۴ شکل (۵-۳) توابع تعلق خروجی
- ۵۷ شکل (۵-۴) بلوک دیاگرام کنترل کننده فازی توسعه یافته
- ۵۷ شکل (۵-۵) بلوک AGC
- ۵۸ جدول (۵-۱) پارامترهای تنظیم شده توسط PSO
- ۵۸ شکل (۵-۶) خطای ردگیری رابط
- ۵۹ شکل (۵-۷) مسیر ردگیری رابط
- ۵۹ شکل (۵-۸) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان
- ۶۰ شکل (۵-۹) سیگنال کنترلی ورودی

- ۷۱ شکل (۶-۱) بلوک دیاگرام کنترل کننده خطی ربات با مفاصل منعطف
- ۷۲ شکل (۶-۲) مسیر ردگیری رابط یک
- ۷۲ شکل (۶-۳) مسیر ردگیری رابط دو
- ۷۳ شکل (۶-۴) نرم خطای ردگیری رابط یک
- ۷۳ شکل (۶-۵) نرم خطای ردگیری رابط دو
- ۷۴ شکل (۶-۶) سیگنال کنترلی ورودی دو رابط
- ۷۵ شکل (۶-۷) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط یک
- ۷۵ شکل (۶-۸) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط دو
- ۷۶ شکل (۶-۹) بلوک دیاگرام کنترل فازی بهینه ربات با مفاصل منعطف
- ۷۷ شکل (۶-۱۰) نرم خطای ردگیری رابط یک
- ۷۸ شکل (۶-۱۱) نرم خطای ردگیری رابط دو
- ۷۸ شکل (۶-۱۲) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط یک
- ۷۹ شکل (۶-۱۳) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط دو
- ۷۹ شکل (۶-۱۴) سیگنال کنترلی ورودی رابط یک
- ۸۰ شکل (۶-۱۵) سیگنال کنترلی ورودی رابط دو
- ۸۴ شکل (۷-۲) روش کار تخمین تابع در هر نمونه زمانی
- ۹۰ شکل (۷-۳) توابع تعلق ورودی
- ۹۱ شکل (۷-۴) بلوک دیاگرام کنترل کننده پیشنهادی

۹۲

شکل (۵-۷) تخمین تابع موقعیت موتور

۹۳

شکل (۶-۷) مسیر ردگیری موقعیت رابط

۹۳

شکل (۷-۷) تغییر پارامترهای سیستم فازی در طول شبیه سازی

فصل اول

مقدمه

۱-۱ مرور کارهای قبلی

۱-۱-۱ مدل سازی ربات با مفاصل منعطف

در دنیای امروز نیاز به تولید محصولات به صورت انبوه و با کیفیت بالا، افزایش یافته است. انجام بسیاری از کارها از عهده انسان خارج است. از جمله می توان به کار در محیط های خطرناک مانند فضا، دریا، زیردریا، محیط های شیمیایی و رادیواکتو اشاره کرد. همچنین برخی از کارها کسل کننده است و در بعضی موارد نیاز به دقت بسیار بالایی دارد و تکرار زیاد آن برای انسان احتمال خطا را افزایش می دهد. بنابراین، استفاده از ربات به جای انسان به عنوان یک طرح جالب مطرح شده است. به دنبال آن کنترل ربات برای انجام کارهای مورد نظر لازم است. کنترل ربات های صنعتی در دهه های اخیر نظر بسیاری از محققان را جلب نموده است و کارهای تحقیقاتی بسیاری برای کنترل این ربات ها انجام شده است.

بررسی ها نشان می دهد که ربات های صلب^۱ در حوزه های بسیاری کاربرد دارند. به موجب این امر تا به امروز روش های کنترلی کارآمدی برای آنها طراحی شده اند. اما با پیشرفت تکنولوژی و برای کاهش هزینه های ساخت و تولید، ربات های سبک تر به بازار آمدند. این ربات ها می توانند به دلیل وزن کمتر با سرعت بیشتری به انجام امور بپردازند. در عوض با مسئله انعطاف روبرو شده اند که ناشی از کاهش حجم و وزن ربات است. بر اساس تحقیقات روش های کنترل ربات های صلب نمی تواند عملکرد مطلوب را برای این ربات ها فراهم کند. چرا که انعطاف در مفاصل در عملکرد ربات تاثیر گذاشته و مانع بهره گیری مناسب از روش های کنترلی گذشته می شود. وجود انعطاف در مفاصل ربات باعث بروز ناپایداری می شود و باید قوانین کنترلی متناسب با این نوع ربات ها طراحی شوند.

انعطاف در مفاصل ممکن است به صورت عمدی به ربات اضافه شود یا ناشی از سیستم انتقال قدرت باشد. برای نمونه وجود تسمه، هارمونیک درایو و محورهای بلند باعث ایجاد انعطاف در ربات ها می شوند.

تا چندی پیش برای جلوگیری از انعطاف در مفاصل و در نتیجه کنترل ساده تر ربات ها سعی در ساخت ربات های با دینامیک صلب بوده است اما به جهت کاربردهایی که ربات با مفاصل منعطف^۲ فراهم می کند، امروزه مسئله انعطاف در مفاصل کاملاً پذیرفته شده است. برای نمونه در برخورد با اشیاء خارجی پیش بینی نشده وجود

^۱ Rigid Robot

^۲ Flexible joint Robot

انعطاف می تواند مفید باشد. در نتیجه اهمیت این نوع ربات ها باعث شده تا تحقیقات وسیعی بر روی آن ها صورت بپذیرد و روش های کنترلی متنوعی پیشنهاد شوند.

۱-۱-۲- روش های کنترلی قبلی برای ربات با مفاصل منعطف

در اولین تلاش ها برای کنترل ربات با مفاصل منعطف در [۱] با صرف نظر کردن از بخش های غیر خطی ربات کنترل کننده خطی پیشنهاد شده است که می تواند در سرعت های پایین حرکت بازو، پاسخ مناسبی را فراهم کند. همچنین در اوایل دهه ۸۰ تحقیقاتی بر روی موضوع کنترل پذیری ربات با مفاصل منعطف در [۳و۲] انجام شده است که در آن عدم توانایی خطی سازی پسخوری^۱ در کنترل این نوع ربات ها نشان داده شده است.

در تحقیقات ابتدایی به دلیل پیچیدگی مدل ربات با مفاصل منعطف مشکلاتی از قبیل عدم قابلیت خطی سازی پسخور یا نیاز به اندازه گیری شتاب و جرک^۲ در ارائه قانون کنترل وجود داشت. در مرجع [۴] نویسندگان با بکارگیری رویه ناوردا^۳ مدل ساده تری برای ربات پیشنهاد دادند که هرچند مشکلات پیشین را برطرف می کرد اما نیاز به سیگنال های بزرگ کنترلی، پیاده سازی عملی آن را با مشکل مواجه می کرد. تحقیقات بسیاری برای بهبود مدل ربات با مفاصل منعطف در [۵-۸] انجام شده است.

روش های کنترلی بسیاری برای ربات با مفاصل منعطف پیشنهاد شده است. که جزئیات آن در مراجع [۹و۱۰] تشریح شده است. از جمله می توان به کنترل PD [۱۱]، کنترل غیرخطی با استفاده از استراتژی کنترل ولتاژ [۱۲]، کنترل مود لغزشی [۱۳]، کنترل تطبیقی [۱۴] و کنترل عصبی تطبیقی [۱۵] آدرس داد. در این میان کنترل فازی به علت توانایی ارائه راه حلی موثر برای کنترل سیستم های پیچیده و دچار عدم قطعیت، بسیار مورد توجه قرار گرفته است. دو ویژگی کنترل کننده فازی که باعث گرایش های زیاد به آن شده است ۱- استفاده از دانش فرد خبره ۲- عدم وابستگی به مدل سیستم می باشد. از آنجا که در کنترل ربات با مفاصل منعطف مدل سیستم بسیار پیچیده می باشد و در بعضی موارد دسترسی به مدل سیستم امکان پذیر نیست بهره گیری از کنترل فازی در کنترل این نوع ربات بسیار مفید بوده است.

بسیاری از کنترل کننده های ارائه شده تاکنون از استراتژی کنترل گشتاور استفاده می کنند. در این پایان نامه به علت مزیت های فراوان استراتژی کنترل ولتاژ از این استراتژی استفاده شده است.

^۱ Feedback linearization

^۲ Jerk

^۳ Invariant manifold

۳-۱-۱ استراتژی کنترل ولتاژ

در کنترل سیستم های رباتیک به طور معمول، گشتاور موتورها به عنوان خروجی کنترل کننده در نظر گرفته می شود و برای کنترل حالت های سیستم از سیگنال گشتاور، به عنوان ورودی سیستم استفاده می شود. اما این سیگنال به طور مستقیم قابلیت اعمال به سیستم را ندارد و باید جریان مربوط به گشتاور مطلوب محاسبه شود، سپس به سیستم اعمال شود. در نظر گرفتن گشتاور به عنوان ورودی سیستم، موجب می شود از دینامیک محرکه ها صرف نظر شود. در نتیجه باعث کاهش درجه سیستم می شود. بنابراین، در سرعت های بالا مانع از ردگیری مناسب سیستم کنترلی می شود. همچنین روش کنترلی را وابسته به مدل غیرخطی سیستم می کند که در سیستم های پیچیده، محاسبات سیستم کنترلی را افزایش می دهد. در نتیجه بهره گیری از این روش در کنترل ربات با مفاصل منعطف با توجه به ویژگی های این ربات از جمله تزویج سنگین در مدل ربات و وجود عدم قطعیت ها در سیستم ربات دشوار می شود و مشکلاتی را برای طراح دربردارد.

استراتژی کنترل ولتاژ [۲۱] در سال های اخیر توانسته است بر مشکلات ناشی از استراتژی کنترل گشتاور غلبه کند. در این روش، ولتاژ به عنوان ورودی سیستم در نظر گرفته می شود و به طور مستقیم به موتورها اعمال می شود. بنابراین، دینامیک ربات نقش گشتاور بار را برای موتور ایفا می کند. در نتیجه این استراتژی مستقل از دینامیک پیچیده و غیرخطی ربات است. چرا که با کنترل موتورها امکان کنترل موقعیت زاویه ای رابطها در ربات فراهم می شود. در این روش در نظر گرفتن دینامیک محرکه ها باعث دقیق تر شدن مدل سازی سیستم می شود که این امر در طراحی یک کنترل کننده کارآمد برای ردگیری در سرعت های بالا [۲۲]، بسیار موثر است.

قابلیت های زیاد استراتژی کنترل ولتاژ باعث شده تا تمامی روش های کنترلی به سادگی در آن قابل اجرا باشند. برای نمونه می توان به کنترل مقاوم با استراتژی کنترل ولتاژ [۲۲] و کنترل فازی با استراتژی کنترل ولتاژ [۲۳] آدرس داد.

در این استراتژی از موتورهای الکتریکی به عنوان محرکه استفاده می شود. خطی بودن مدل این موتورها موجب می شود که تمام روش های خطی کنترلی برای کنترل سیستم هایی با مدل غیر خطی قابل پیاده سازی شود.

استراتژی کنترل گشتاور در کنترل ربات های صنعتی بسیار به کار گرفته شده است. اما در نظر گرفتن انعطاف در مفاصل پیچیدگی مدل سیستم را افزایش می دهد و این امر در این روش کنترلی به علت وابستگی آن به مدل سیستم به روابط پیچیده ای منجر می شود که کار کنترل سیستم را کند می کند. از این استراتژی در کنترل ربات با مفاصل منعطف نیز استفاده شده است. برای نمونه می توان به [۱۹ و ۲۴] اشاره کرد. اما به سبب وجود مزیت های فوق در استراتژی کنترل ولتاژ در این پایان نامه از این استراتژی در طراحی کنترل کننده استفاده شده است.

۱-۱-۴ کنترل فازی

با پیدایش منطق فازی^۱ تحول نوینی در امر کنترل سیستم های خطی و غیرخطی رخ داد. منطق فازی عملکرد مناسبی را به عنوان بخشی از سیستم کنترل یا به عنوان یک کنترل کننده مستقل در بسیاری از حوزه ها به خصوص در کنترل ربات ها از خود نشان داده است. اگر پارامترهای این کنترل کننده به طور مناسب تنظیم شود کنترل کننده فازی به خوبی می تواند سیستم های چند متغیره و غیرخطی با تزویج سنگین را کنترل کند. توانایی منطق فازی در تخمین توابع غیر خطی آن را به ابزار قدرتمندی برای مدل نمودن عدم قطعیت ها در سیستم های کنترلی تبدیل نموده است. این امر موجب شده است که در ترکیب با کنترل کننده های دیگر به عنوان تقریب زن قسمتی از قانون کنترل مورد استفاده قرار بگیرد.

در [۲۶] با توسعه کنترل کننده ی PI فازی ارائه شده در [۲۷] به کنترل ربات با مفاصل منعطف پرداخته است. کنترل کننده فازی طراحی شده در این مرجع بسیار ساده بوده و تنها از شش قانون تشکیل شده است. با استفاده از معادلات سیستم یک معادله کمکی برای تغییر اتوماتیک ضریب خروجی کنترل کننده در نظر گرفته شده است که بوسیله آن توانسته است پاسخ مناسبی برای ردگیری ارائه دهد. کنترل کننده طراحی شده در این مرجع همانند بسیاری از کنترل کننده های فازی به پارامترهایی که توسط طراح تنظیم می شود بسیار وابسته است. مزیت این طراحی، استفاده از دو متغیر حالت موقعیت موتور و سرعت موتور است و نیازی به اندازه گیری شتاب موتور وجود ندارد.

^۱ Fuzzy logic system

در [۲۹] ابتدا با ارائه کنترل کننده فازی PD به کنترل موقعیت قسمت صلب ربات پرداخته است و سپس با ترکیب الگوریتم های دیگر سه کنترل کننده ترکیبی PD FLC PID, PD FLC FLC, و^۱ PD FLC IS برای کاهش ارتعاش های ربات با مفاصل منعطف طراحی شده است. با پیاده سازی آن بر روی ربات Quanser پاسخ سه کنترل کننده با یکدیگر مقایسه شده است که نتایج نشان دهنده عملکرد مطلوب PD FLC IS نسبت به دو کنترل کننده دیگر است.

در مرجع [۳۰] برای کاهش تعداد قوانین فازی و در نتیجه کاهش محاسبات و پیچیدگی کنترل کننده، کنترل کننده را به دو بخش تقسیم نموده است. در هر بخش دو ورودی در نظر گرفته شده است. با توجه به اینکه تعداد ورودی های کنترل کننده چهار می باشد قوانین را از ۸۱ به ۱۸ کاهش داده است. در این مرجع ولتاژ موتور سرو به عنوان خروجی کنترل کننده در نظر گرفته شده است. هرچند در نهایت ارتعاش های خطای سیستم کامل از بین نرفته است اما در مقایسه ای که نویسندگان با روش^۲ LQR [۳۱] داشته اند، روش ارائه شده عملکرد مطلوب تری را در کاهش خطای سیستم و کاهش بالازدگی های خطای موقعیت، نسبت به روش مذکور داشته است.

در [۳۲] با تاکید بر عدم قطعیت های ربات و در دسترس نبودن مدل دقیق سیستم ربات با مفاصل منعطف به کارآیی کنترل کننده فازی تطبیقی در کنترل این نوع ربات پرداخته شده است. بر اساس تئوری پایداری لیاپانوف، پارامترهای کنترل کننده فازی تطبیقی مستقیم^۳ به صورت همزمان به گونه ای تنظیم می شوند که پایداری سیستم تضمین شود.

در مراجع [۱۶ و ۱۷] نویسندگان با ترکیب کنترل فازی و روش بازگشت به عقب^۴ قانون کنترل تطبیقی برای کنترل سیستم های غیرخطی پیشنهاد داده اند اما در این روش زیاد بودن پارامترهای کنترلی که در قانون تطبیق باید تنظیم شوند، در زمانی که درجه سیستم غیرخطی بالا باشد باعث همگرایی دیررس پارامترها می شود. در [۱۸] با توسعه کنترل کننده فازی بازگشت به عقب پارامترهای تطبیق را کاهش داده و تا حدودی این مشکل برطرف شده است. محدودیت دیگری که این نوع کنترل کننده با آن مواجه است مشکل انفجار پیچیدگی^۵ می باشد. در مرجع [۱۹] با پیشنهاد کنترل کننده تطبیقی فازی بازگشت به عقب، ربات یک محوره با مفاصل منعطف را به خوبی کنترل نموده است و بر محدودیت های بالا با در نظر گرفتن سطح همگرایی تا حدودی غلبه نموده است. در این مقاله با در نظر گرفتن دینامیک محرکه ها و استفاده از مشاهده گر تنها با اندازه-

^۱ PD Fuzzy Logic Controller input-shaper

^۲ Linear quadratic regulation

^۳ Direct Adaptive Fuzzy

^۴ Backstepping

^۵ Explosion of complexity

گیری موقعیت شفت موتور و زاویه محور توانسته است پاسخ مناسبی را فراهم کند و از منطق فازی برای تقریب بخش های غیرخطی استفاده شده است.

با وجود مزیت های فوق تحلیل پایداری کنترل فازی آسان نیست و تنظیم پارامترها معمولا فرایند زمان بری برای سیستم های غیرخطی و چند متغیره می باشد. همچنین کنترل فازی به علت استفاده از دانش فرد خبره پارامتر های زیادی برای تنظیم دارد که انتخاب صحیح پارامترها به نزدیک شدن سیستم به شرایط مطلوب کمک می کند.

این پایان نامه تلاش دارد با استفاده از الگوریتم بهینه سازی پرندگان یک کنترل کننده فازی را به صورت هوشمند برای سیستم غیرخطی با تزویج سنگین ربات با مفاصل منعطف ارائه کند.

۱-۱-۵ الگوریتم بهینه سازی پرندگان

یافتن کمینه توابع به دو صورت تحلیلی و عددی ممکن است. روش های تحلیلی در توابعی که مشتق پذیر نباشند و همچنین در توابعی که تابع هدف پیچیده دارند مشکل است. اما روش های عددی مانند الگوریتم بهینه سازی پرندگان^۱ و الگوریتم ژنتیک^۲ به صورت هوشمند فضای هدف را جستجو کرده و مشکلات فوق را ندارند. چرا که برای یافتن پاسخ بهینه به معادلات سیستم نیاز ندارند. در نتیجه شرط مشتق پذیری و پیوستگی تابع هدف مطرح نمی باشد.

الگوریتم بهینه سازی پرندگان [۳۶] به سبب سادگی معادلات به روز رسانی موقعیت ذراتش، سرعت همگرایی بالایی نسبت به دیگر روش های تکاملی^۳ دارد و در سال های اخیر به خوبی در بهینه سازی توابع پیچیده و غیرخطی موفق بوده است. اما پاسخ نهایی آن را نمی توان به طور قطع بهینه سراسری^۴ دانست. زیرا این الگوریتم ممکن است در کمینه محلی^۵ گرفتار شود. امروزه این نقطه ضعف را با روش های جدیدی به طور نسبی حل نموده اند [۳۳-۳۵]. همچنین با ترکیب با دیگر الگوریتم های تکاملی و یا گرادیان نزولی توانایی الگوریتم بهینه سازی پرندگان را در یافتن کمینه سراسری افزایش داده اند. برای بهبود عملکرد الگوریتم در نزدیکی نقطه کمینه، در [۳۷] سرعت حرکت ذرات را تابعی از تابع هزینه قرار داده شده است تا با کم شدن هزینه هپی تابع

^۱ Particle swarm optimization

^۲ genetic algorithm

^۳ Evolutionary techniques

^۴ Global minimum

^۵ Local minimum

خطی تابع هزینه، سرعت کاهش یابد. در نتیجه، الگوریتم به خوبی می تواند نقطه کمینه اصلی را جستجو نماید. روش های دیگری نیز برای بهبود عملکرد این الگوریتم ارائه شده است که در فصل سوم به تفصیل درباره آن توضیح داده شده است.

از الگوریتم بهینه سازی پرندگان برای کمینه سازی خطای سیستم در کنترل کننده های زیادی استفاده شده است. پژوهش های انجام شده در زمینه بهره گیری از الگوریتم بهینه سازی پرندگان در کنترل سیستم های متنوع از دو جهت قابل دسته بندی هستند. در بعضی از مقالات از الگوریتم بهینه سازی پرندگان تنها برای تنظیم پارامترهای قوانین کنترلی استفاده شده به این معنا که قوانین کنترل بر اساس روش های مختلف پیشنهاد شده و تنها برای بهبود عملکرد سیستم کنترلی پارامترهای آن را برای رسیدن به معیار عملکرد بهینه به وسیله PSO تنظیم می شود. اما در بعضی دیگر با ترکیب این الگوریتم با دیگر روش های کنترل، روش کنترلی جدیدی به وجود آمده که هر یک مزایای خود را دارند.

از جهتی دیگر در بعضی مقالات به صورت برخط^۱ از الگوریتم استفاده شده و پارامترها همزمان با کارکرد سیستم تنظیم می شود و در بعضی دیگر بصورت خارج خط^۲ پارامترها تنظیم می شوند و سپس به سیستم اعمال می شود. در اینجا مقالات متنوعی را مورد بررسی قرار می دهیم.

در [۳۸] از ویژگی سرعت همگرایی الگوریتم بهینه سازی پرندگان برای کاهش ارتعاش های سیستم تعلیق خودرو در زمان بسیار کوتاه استفاده شده است. در این مرجع ابتدا قانون کنترل با استفاده از روش کنترل نمایی، پایداری سیستم کنترل را فراهم نموده است و سپس در راستای تنظیم پارامترهای قانون کنترل، الگوریتم بهینه سازی تابع هدفی را که از پارامترهای قانون کنترل تشکیل شده است، برای رسیدن به عملکرد مطلوب بهینه می کند. برای تولید ذرات در هر مرحله، قیودی که روش کنترل نمایی برای تضمین پایداری بدست آورده است در نظر گرفته می شود.

در مرجع [۳۹] کنترل کننده ای با ترکیب روش کنترلی بازگشت به عقب و الگوریتم بهینه سازی پرندگان طراحی شده است. کنترل کننده پیشنهادی بدون نیاز به اطلاعات دقیق سیستم و با وجود عدم قطعیت ها نتایج خوبی را ارائه نموده است. با استفاده از ویژگی های روش بازگشت به عقب تابع هزینه مناسب پیشنهاد شده است و الگوریتم بهینه سازی سعی در بهینه سازی آن دارد. برای تولید جمعیت ذرات در هر مرحله از معیار پایداری لیاپانوف بهره گرفته شده است و بدین ترتیب پایداری سیستم کنترل تضمین می شود. در این طراحی

^۱ On-line

^۲ Off-line

الگوریتم بهینه سازی نقش مهمی را ایفا می کند و به صورت برخط پارامترهای کنترلی را تولید می کند. در بعضی دیگر از مقالات الگوریتم بهینه سازی تنها وظیفه کوچکی را برعهده دارد بدین معنی که ابتدا با روش های کنترلی قانون کنترل پیشنهاد می شود و سپس با استفاده از PSO پارامترهای عملکرد بهینه تولید می شود. برای مثال در [۴۰] الگوریتم بهینه سازی با روش مود لغزشی ترکیب شده است و توانسته است به خوبی ضعف این روش در لغزش سیگنال کنترلی را برطرف نماید. برای این امر از PSO برای مقدار دهی اولیه پارامترها استفاده شده است تا همگرایی به پاسخ مطلوب سریع تر روی دهد. روش کنترلی مود لغزشی قیودی را برای تضمین پایداری بر پارامترها تحمیل می کند، الگوریتم بهینه سازی پارامترها را در آن محدوده تولید می نماید در نتیجه پایداری سیستم تضمین شده است. ماهیت روش مود لغزشی به خوبی بر عدم قطعیت ها غلبه می کند در این طراحی تابع هزینه الگوریتم بهینه سازی با استفاده از این ویژگی روش کنترل مود لغزشی تعریف شده است.

مشابه روش پیشنهادی در مرجع [۴۰]، قبلا توسط نویسندگان مرجع [۴۱] با استفاده از الگوریتم بهینه سازی ژنتیک انجام شده است. اما با توجه به عملیات پیچیده این الگوریتم در هر مرحله برای تولید جمعیت در مرحله بعدی، زمان همگرایی این روش برای رسیدن به پاسخ مطلوب زیاد است.

در [۴۲] با استفاده از ترکیب الگوریتم بهینه سازی پرندگان و معیار پایداری لیاپانوف که در [۴۳] بیان شده است، کنترل کننده تطبیقی فازی به صورت غیرمتصل آموزش داده می شود. از این کنترل کننده برای کنترل ناوبری ربات با بینایی ماشین استفاده شده است. در اینجا دو کنترل کننده فازی تاگاکا- سوگونو از مرتبه صفر برای کنترل سرعت چرخ راست و چپ طراحی شده است. در این روش با قرار دادن ذرات در محدوده ای که پایداری مجانبی تضمین شده است، پارامترهای نامعلوم کنترل کننده در حالت غیرمتصل تنظیم می شود.

در مقایسه با الگوریتم ژنتیک که در [۴۴] اعمال شده است، الگوریتم بهینه سازی پرندگان سرعت همگرایی بسیار خوبی در رسیدن به موقعیت مطلوب را دارد چرا که در این الگوریتم برخلاف الگوریتم ژنتیک تولید جمعیت برای هر مرحله نیاز به عملیات پیچیده ندارد.

در مرجع [۴۵] با پرداختن به مزایا و معایب PSO، الگوریتم بهینه سازی پرندگان تقویت شده را با ترکیب این الگوریتم با Q-learning ارائه داده است. آنچه نویسندگان مقاله را بر آن داشته است تا PSO تقویت شده را ارائه کنند این است که پاسخ نهایی الگوریتم بهینه سازی پرندگان الزاما کمینه سراسری نیست و ممکن است الگوریتم در کمینه محلی گرفتار شده باشد. با استفاده از Q-learning تولید ذرات برای مرحله بعد هوشمند شده

است تا این احتمال را به حداقل برساند و البته سرعت همگرایی را نیز افزایش می دهد. از آنجا که طراحی سیستم کنترل فازی و تنظیم پارامتر توسط فرد خبره انجام می شود، تضمینی برای عملکرد بهینه سیستم کنترلی وجود ندارد. بنابراین، طراحی اتوماتیک سیستم فازی بسیار مورد توجه است. در این مقاله ضرایب PID کنترل کننده PID-FUZZY توسط الگوریتم بهینه سازی پرندگان تقویت شده، تنظیم شده است.

نویسندگان در مرجع [۴۶-۴۷] از الگوریتم بهینه سازی پرندگان برای تنظیم پارامترهای روش کنترل امیدانس استفاده نموده اند. PSO بعد از ارائه روش کنترلی تنها برای بهبود عملکرد ضرایب قانون کنترل را تنظیم می نماید. در [۴۷] با کنترل ربات با مفاصل منعطف با استفاده از کنترل امیدانس برای کمینه کردن خطای سیستم از PSO برای تنظیم پارامترها استفاده نموده است.

۱-۲ اهداف پایان نامه

این پایان نامه سعی بر آن دارد که طراحی ساختار کنترل کننده فازی را به صورت هوشمند با استفاده از الگوریتم بهینه سازی پرندگان به انجام برساند. به طوری که همزمان عملکرد بهینه و پایداری محقق شود. در این طراحی نیاز به استفاده از دانش فرد خبره به حداقل می رسد. روش پیشنهادی برای سیستم های با مدل پیچیده مانند بازوی ربات با مفاصل منعطف بسیار کارآمد می باشد. چراکه معادلات غیرخطی سیستم با تزویج سنگین، کار تحلیل پایداری کنترل کننده فازی را دشوار می نماید. ضمن آنکه به علت استفاده از الگوریتم بهینه سازی پرندگان ارتعاشها در سیستم ربات کاهش می یابد. به دلیل عدم وابستگی الگوریتم کنترلی ارائه شده به مدل سیستم، از طرح پیشنهادی می توان در کنترل انواع سیستم غیرخطی استفاده نمود.

۳-۱ نمای کلی پایان نامه

در این پایان نامه در فصل دو به معرفی معادلات حاکم بر سیستم ربات با مفاصل منعطف می پردازیم تا در فصول بعد برای پیاده سازی روش های کنترلی ارائه شده، از آن بهره ببریم. در فصل سوم الگوریتم بهینه سازی پرندگان تشریح می شود و روش های نوینی برای بهبود عملکرد آن بررسی می شود. در فصل چهارم الگوریتم بهینه سازی پرندگان با کنترل بهینه مربعی خطی ترکیب شده است و روش جدیدی ارائه شده است. در فصل پنجم کنترل کننده فازی را برای کنترل ربات با مفاصل منعطف بکار می گیریم و این کنترل کننده را که دانش فرد خبره طراحی می شود با کنترل کننده پیشنهادی پایان نامه مقایسه می نماییم. در فصل ششم روش جدید طراحی ساختار کنترل کننده بیان می شود و در ابتدا بر روی کنترل کننده خطی پیاده سازی می شود. سپس از این روش برای طراحی کنترل کننده فازی به صورت هوشمند استفاده می شود. در فصل هفتم با ارائه روشی جدید کنترل کننده فازی با استفاده از الگوریتم بهینه سازی پرندگان به طور برخط بخشی از مدل سیستم را تقریب می زند تا در قانون کنترل به طور مطلوب از آن استفاده شود.

فصل دوم

مدلسازی ربات با مفاصل منعطف

۲-۱ مقدمه

این فصل به بررسی دینامیک های سیستم های رباتیک، شامل بازوی ربات صلب با مفاصل منعطف و دینامیک محرکه ها می پردازد. ویژگی های این نوع ربات مورد بررسی قرار می گیرد و روابط حاکم بر ربات بیان می شود و پیچیدگی معادلات ربات با مفاصل منعطف تشریح می گردد. برای تشریح بهتر تزویج و پیچیدگی معادلات مدل سیستم، معادلات ربات دو رابط با مفاصل منعطف بیان می شود و از آن مدل برای پیاده سازی روش های کنترلی پیشنهادی در این پایان نامه در فصول بعد استفاده خواهد شد. در اینجا از بیان مسائل مقدماتی در زمینه دینامیک محرکه ها و جزئیات موتورهای الکتریکی خودداری شده است و فرض بر آن است که خواننده با این مسائل و همچنین روش اویلر-لاگرانژ^۱ در تحلیل مکانیکی بازوی ربات آشنایی دارد.

۲-۲ معادلات دینامیکی ربات های با مفاصل منعطف

برای مدل سازی ربات ها با مفاصل منعطف فرض می شود که مفاصل لولایی بوده و بوسیله محرکه های DC رانده می شوند. از ویژگی های موتورهای الکتریکی تولید گشتاور کم با سرعت بالا می باشد. بنابراین، این نوع موتورها باید بوسیله سیستم انتقال قدرت تجهیز شوند تا بتوانند گشتاور زیادی را در سرعت پایین برای به حرکت درآوردن ربات تولید کنند. در این صورت سیستم انتقال قدرت باعث ایجاد انعطاف در مفاصل می شود. این مولفه عامل اصلی لغزش در بازوهای ربات صنعتی است. در مقایسه با ربات صلب، درجات آزادی ربات با مفاصل منعطف دو برابر می باشد.

دستیابی به عملکرد با دقت بالا برای ربات با مفاصل منعطف تا زمانی که موقعیت رابط نتواند از موقعیت محور موتور پیروی کند به نظر مشکل می رسد. در یک طرح ساده از ربات با مفاصل منعطف [۲۰] اتصالات به صورت صلب در نظر گرفته می شود و موتورها به صورت انعطاف پذیر به مفاصل متصل شده است. در مرجع [۲۴] گشتاور موتور ها به عنوان ورودی در نظر گرفته شده است این فرض که می توان گشتاور را به عنوان ورودی به سیستم رباتیک اعمال کرد باعث می شود، دینامیک محرکه ها نادیده گرفته شود. در نتیجه درجه سیستم کنترلی به طور اشتباه کاهش می یابد. برای کنترل ربات در سرعت های بالا در نظر نگرفتن دینامیک محرکه ها موجب عدم ردگیری مناسب می شود.

^۱ Euler-Lagrange method

در این پایان نامه، ولتاژ موتورها به عنوان ورودی سیستم در نظر گرفته می‌شود. مدل ساده ای برای معادلات الکتریکی ربات در نظر گرفته شده است. اسپانگ در [۲۰] برای ساده سازی معادلات ربات فرض های زیر را در نظر گرفته است.

فرض ۱: انرژی جنبشی تنها ناشی از دوران است. به بیان دیگر حرکتها نسبت به دستگاه لخت به صورت دورانی خالص بوده و از حرکت انتقالی صرف نظر می‌گردد.

فرض ۲: لختی دورانی موتور نسبت به محور موتور متقارن است.

فرض ۳: انعطاف پذیری در هریک از مفاصل با فنر خطی مدل می‌گردد.

براین اساس و با حذف اثرات حرکت رابط بر روی انرژی جنبشی روتور، می توان نشان داد که انرژی جنبشی سیستم از معادله (۱-۲) قابل محاسبه می‌باشد.

$$T(\theta, \dot{\theta}, \dot{\theta}_m) = \frac{1}{2} \dot{\theta}^T D(\theta) \dot{\theta} + \frac{1}{2} \dot{\theta}_m^T J_m \dot{\theta}_m \quad (1-2)$$

که در آن $\theta \in R^n$ بردار زاویه مفاصل و $\theta_m \in R^n$ بردار زاویه موتورها می باشند. بنابراین، سیستم دارای $2n$ مشخصه به صورت $[\theta \ \theta_m]$ است. $D(\theta)$ ماتریس $n \times n$ اینرسی ربات که متقارن مثبت معین است و J_m ماتریس قطری لختی دورانی موتورها در سمت بازو می‌باشد [۲۰].

با توجه به فرض دوم، انرژی پتانسیل گرانشی تابعی از θ (مستقل از سرعت مفصلی) بوده و با توجه به فرض سوم، انرژی پتانسیل ناشی از انعطاف پذیری در مفاصل به صورت معادله (۲-۲) بیان می‌شود.

$$P_f(\theta, \theta_m) = \frac{1}{2} (\theta - \theta_m)^T K_s (\theta - \theta_m) \quad (2-2)$$

که در آن K_s ماتریس قطری ضرایب سختی فنرهای مدل کننده ی انعطاف پذیری در مفاصل است. در نتیجه، انرژی پتانسیل کل سیستم به صورت معادله (۳-۲) قابل محاسبه است.

$$P(\theta, \theta_m) = P_g(\theta) + P_f(\theta, \theta_m) \quad (3-2)$$

با توجه به مطالب فوق و بکارگیری روش لاگرانژ، معادلات حرکت با استفاده از معادله اولر-لاگرانژ در معادله (۴-۲) و جایگذاری ویژگی های ربات با مفاصل منعطف به شکل معادلات (۵-۲) و (۶-۲) بدست می آید.

$$L = T - P \quad (۴-۲)$$

$$D(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + g(\theta) = K(r\theta_m - \theta) \quad (۵-۲)$$

$$J_m\ddot{\theta}_m + B\dot{\theta}_m + rK(r\theta_m - \theta) = \tau \quad (۶-۲)$$

که در آن $C(\theta, \dot{\theta})$ ماتریس $n \times n$ گشتاور جانب مرکز و کوریولیس، $g(\theta)$ بردار $n \times 1$ گشتاور ثقلی، $\tau \in R^n$ بردار گشتاور موتورها می باشد. B و r ماتریس های قطری $n \times n$ برای ثابت میرایی چرخشی و ضریب چرخ-دنده می باشد.

برای ساده سازی مدل ضریب کاهشی چرخ دنده و ضریب فنر ثابت در نظر گرفته شده است و $g(\theta)$ گشتاور ثقلی تنها تابعی از موقعیت مفصل در نظر گرفته شده است. سیستم ارائه شده در معادلات فوق بسیار غیر خطی، دارای تزویج سنگین، پرمحاسبه و چندورودی-چندخروجی است.

یادآوری می نمایم که معادلات فوق معادلات حرکت بازو است و تنها در مواردی که سرعت بازو آهسته باشد، می توان از دینامیک محرکه ها، اغتشاشات، دینامیک های مدل نشده، دینامیک سنسورها و دیگر بخش های سیستم رباتیک صرف نظر کرد. در غیر اینصورت امکان ردگیری مطلوب فراهم نمی شود. پیچیدگی مدل چالش بسیار جدی در امر کنترل و مدلسازی ربات است. اگر دینامیک محرکه ها نیز در نظر گرفته شود این پیچیدگی بیشتر میشود. برای بدست آوردن ولتاژ موتورها به عنوان ورودی سیستم باید از معادلات الکتریکی موتور DC مغناطیس دائم استفاده شود.

۲-۳ دینامیک محرکه ها

در این بخش دینامیک محرکه ها که گشتاور ورودی معادلات دینامیکی ربات را فراهم می کند بررسی می شود. یک بازوی ربات ممکن است دارای چندین نوع محرکه نظیر محرکه های الکتریکی، پنوماتیکی یا هیدرولیکی باشد. برای کاربردهای رباتیک با سرعت بالا معمولاً از محرکه های الکتریکی نظیر محرکه های جریان مستقیم، محرکه های القایی و محرکه های DC بدون جاروبک استفاده می گردد. محرکه های مختلف دارای دینامیک-های مختلفی هستند. اگر چه ساده سازی های زیادی در مدل دینامیکی محرکه ها صورت میگیرد. اما باز هم

مدل دینامیک باقی مانده غیرخطی است و استفاده از یک روش کنترلی پیشرفته در این محرکه ها زمینه تحقیقاتی روز بوده است [۲۸]. در این بخش به معرفی محرکه های DC مغناطیس دائم می پردازیم. زیرا این محرکه ها به خاطر مدل دینامیکی خطی و ساده، در بازو های ربات استفاده های بسیاری دارند. اگرچه امروز با وجود پردازشگرهای سرعت بالا استفاده از سایر محرکه های AC در بازوی های ربات نیز به آسانی استفاده از محرکه های DC مغناطیس دائم است. دینامیک های محرکه های DC مغناطیس دائم به صورت معادله (۷-۲) است.

$$u = RI_a + L\dot{I}_a + K_b\dot{\theta}_m \quad (7-2)$$

این معادله به صورت ماتریسی می باشد و در آن $u \in R^n$ بردار ولتاژ موتورها می باشد، $I_a \in R^n$ بردار جریان آرمیچر و $\dot{\theta}_m$ بردار سرعت موتورها است. R, L, K_b ماتریس های $n \times n$ قطری بیان کننده ثابت مقاومت آرمیچر، اندوکتانس آرمیچر و ثابت ضدمحرکه است. بردار گشتاور موتور τ که به عنوان ورودی معادله دینامیکی (۶-۲) بوسیله جریان موتور از رابطه (۸-۲) بدست می آید.

$$K_m I_a = \tau \quad (8-2)$$

در این رابطه ماتریس قطری از ثابت گشتاور است. دینامیک معادلات (۵-۲) تا (۸-۲) در فرم فضای حالت در بخش بعد بیان می شود.

۴-۲ معادلات فضای حالت حاکم بر سیستم بازوی رباتیک با مفاصل منعطف

معادله (۹-۲) که پارامترهای آن در دسته معادلات (۱۰-۲) قابل مشاهده است، مدل سیستم در فرم فضای حالت را نشان می دهد.

$$\dot{X} = f(X) + bu \quad (۹-۲)$$

که:

$$f(X) = \begin{bmatrix} x_2 \\ D^{-1}(x_1)(-g(x_1) - Kx_1 - C(x_1, x_2)x_2 + Krx_3) \\ x_4 \\ J^{-1}(rKx_1 - r^2Kx_3 - Bx_4 + K_mx_5) \\ -L^{-1}(K_b x_4 + Rx_5) \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ L^{-1} \end{bmatrix}, \quad X = \begin{bmatrix} \theta \\ \dot{\theta} \\ \theta_m \\ \dot{\theta}_m \\ I_a \end{bmatrix} \quad (۱۰-۲)$$

برای ربات دورابط-لولایی پارامترها به صورت زیر بیان می شود:

$$\theta = [\theta_1 \quad \theta_2], \quad \theta_m = [\theta_{m1} \quad \theta_{m2}]$$

که در آن نشان دهنده موقعیت رابط اول و θ_2 نمایانگر موقعیت رابط دوم و θ_m برداری از موقعیت محور موتور اول و دوم می باشد. دیگر پارامترها به صورت زیر قابل بیان است:

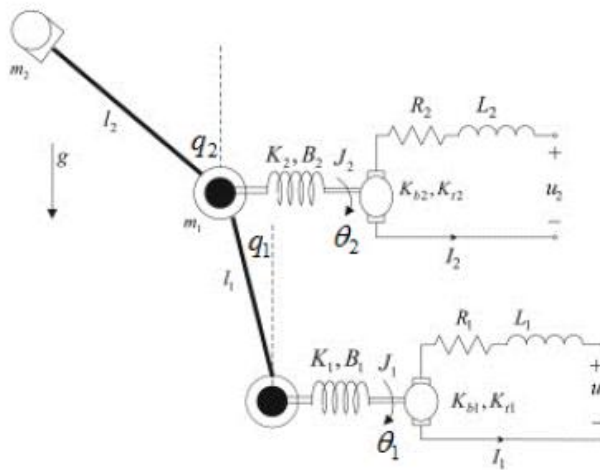
$$D(\theta) = \begin{bmatrix} m_1 l_{c1}^2 + m_2 (l_1^2 + l_{c2}^2 + 2l_1 l_{c2} \cos(\theta_2)) + I_1 + I_2 & m_2 (l_{c2}^2 + l_1 l_{c2} \cos(\theta_2)) + I_2 \\ m_2 (l_{c2}^2 + l_1 l_{c2} \cos(\theta_2)) + I_2 & m_2 l_{c2}^2 + I_2 \end{bmatrix} \quad (۱۴-۲)$$

$$C(\theta, \dot{\theta}) = \begin{bmatrix} -m_2 l_1 l_{c2} \dot{\theta}_2 \sin(\theta_2) & -m_2 l_1 l_{c2} (\dot{\theta}_1 + \dot{\theta}_2) \sin(\theta_2) \\ m_2 l_1 l_{c2} \dot{\theta}_1 \sin(\theta_2) & 0 \end{bmatrix} \quad (۱۵-۲)$$

$$G(\theta) = \begin{bmatrix} (m_1 l_{c1} + m_2 l_1) g \cos(\theta_1) + m_2 l_{c2} g \cos(\theta_1 + \theta_2) \\ m_2 l_{c2} g \cos(\theta_1 + \theta_2) \end{bmatrix} \quad (۱۶-۲)$$

که در آن طول رابط اول، l_2 طول رابط دوم، l_{c1} و l_{c2} فاصله بین مراکز جرم اول و دوم از مفاصل اول و دوم، m_2 و m_1 جرم رابط اول و دوم، I_2 و I_1 ممان اینرسی رابط اول و دوم در دستگاه متصل به مرکز جرم و g شتاب جاذبه زمین می‌باشد. ساختار ربات دورابط-لولایی با مفاصل منعطف در شکل (۱-۲) قابل مشاهده است. ماتریس ژاکوبینی که در شبیه سازی های فصول بعد استفاده شده به صورت زیر است:

$$J(\theta) = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \quad (17-2)$$



شکل (۱-۲) ساختار ربات با مفاصل منعطف

فصل سوم

الگوریتم بهینه سازی پرندگان

۳-۱ مقدمه

الگوریتم بهینه سازی پرندگان که به نام الگوریتم بهینه سازی گروهی ذرات نیز شناخته شده است، اولین بار توسط یک روانشناس اجتماعی با نام جیمز کندی و یک مهندس الکترونیک با نام راسل ابرهارت در [۳۶] با استفاده از تجربیات قبلی در رابطه با مدل کردن رفتارهای دسته جمعی که در بسیاری از انواع پرندگان قابل مشاهده است، توسعه داده شد. هر چند تعداد قابل ملاحظه ای از این گونه الگوریتم‌ها در آن زمان نظر محققان را به خود جلب کرده بودند، این دو محقق بیشتر بروی مدل‌هایی که توسط زیست شناس فرانک هپنر ایجاد شده بود تاکید کردند.

در الگوریتم PSO اعضای گروه با یکدیگر در مورد بهترین مکان یافت شده تا مرحله فعلی تبادل اطلاعات انجام می دهند. همچنین بهترین مکان یافت شده تا مرحله فعلی در بین یک همسایگی از اعضا نیز در بین اعضای این همسایگی مبادله می گردد. به عبارت دیگر، بهترین جواب یافت شده برای تمام اعضای گروه مشخص است و همچنین بهترین جواب یافت شده در یک همسایگی از اعضا تنها در بین اعضای همین همسایگی مشخص است. از این اطلاعات برای بروز رسانی موقعیت و سرعت ذرات در مراحل استفاده می گردد. بررسی های زیادی بر روی تنظیم پارامترهای معادلات بروز رسانی موقعیت و سرعت ذرات، به گونه ای که پاسخ مطلوب با سرعت همگرایی بالا بدست آید انجام شده است [۴۸-۴۹]. پاسخ بدست آمده در روش بهینه سازی پرندگان، لزوماً پاسخ سراسری نیست. ممکن است الگوریتم در کمینه یا بیشینه محلی گرفتار شود. برای حل این مشکل محققان روش‌های مختلفی را پیشنهاد داده‌اند و الگوریتم بهینه سازی پرندگان را توسعه داده‌اند [۵۰-۵۵].

۳-۲ شرح الگوریتم بهینه سازی پرندگان

الگوریتم PSO یک الگوریتم بر پایه جمعیت است که از نقطه نظرهایی شبیه به الگوریتم ژنتیک می باشد، با این وجود نحوه جستجو در فضای جواب توسط این دو الگوریتم بسیار متفاوت است. از شباهتهای این دو الگوریتم می توان موارد زیر را نام برد:

۱. هر دو الگوریتم بر پایه جمعیت می باشند. یعنی در ابتدا، تعدادی از اعضا تولید شده و در سرتاسر فضای جواب پخش می گردند.
۲. هر دو الگوریتم از یک تابع برازندگی برای ارزیابی جوابهای بدست آمده استفاده می کنند.
۳. هر دو الگوریتم، مراحل مشخصی را به تعداد دفعات معینی قبل از پایان الگوریتم انجام می دهند.

الگوریتم PSO دارای دو عملگر بروز رسانی سرعت و بروز رسانی موقعیت برای انجام عملیات می باشد. برخلاف الگوریتم ژنتیک، الگوریتم بهینه سازی پرندگان در بروز رسانی موقعیت پرندگان (در الگوریتم ژنتیک تولید نسل جدید)، از عملگرهای پیچیده استفاده نمی کند و همین امر زمان محاسبات این الگوریتم را کاهش داده است.

۳-۲-۱- الگوریتم بهینه سازی پرندگان در حالت استاندارد

PSO از تکنیکهای محاسبات تکاملی است و با تقلید از پرواز پرندگان و تبادل اطلاعات میان آنها ابداع شده است. در PSO هر راه حل تنها یک پرنده در فضای جستجو است و عضو نامیده می شود. تمام پرندگان یک مقدار شایستگی دارند که توسط تابع شایستگی که باید بهینه شود ارزیابی می گردند. در دسته ای متشکل از N جزء، موقعیت جزء j ام تحت اثر یک بردار مکانی n بعدی مطابق معادله زیر قرار دارد:

$$x_j = [x_{j1}, x_{j2}, \dots, x_{jn}]^T \in S \quad (۱-۳)$$

که در آن S فضای جستجو است این جزء همچنین دارای یک بردار سرعت به صورت معادله زیر می باشد:

$$v_j = [v_{j1}, v_{j2}, \dots, v_{jn}]^T \in S \quad (۲-۳)$$

بهترین موقعیت قبلی بدست آمده برای جزء j ام با استفاده از معادله زیر نمایش داده می شود

$$x_{pbestj} = [x_{pbestj1}, x_{pbestj2}, \dots, x_{pbestjn}]^T \in S \quad (۳-۳)$$

در نهایت موقعیت جدید اجزای دسته با استفاده از معادلات زیر بدست می آید:

$$v_j(k+1) = w(k)v_j(k) + c_1r_1(k)(x_{pbest,j}(k) - x_j(k)) + c_2r_2(k)(x_{gbest}(k) - x_j(k)) \quad (۴-۳)$$

$$x_j(k+1) = x_j(k) + v_j(k+1) \quad (۵-۳)$$

که در آن w وزن اینرسی می باشد c_1 پارامتری مثبت به نام پارامتر شناختی است و c_2 پارامتری مثبت به نام پارامتر اجتماعی است. c_1 و c_2 را ثابت های شتاب نیز می نامند.

استفاده از پارامتر وزن اینرسی باعث می شود که مصالحه ای بین توانایی اکتشاف سراسری و محلی دسته ایجاد گردد. وزن اینرسی بزرگ، محرکی برای بزرگتر شدن مقدار بردار سرعت ذرات در سراسر ناحیه جوابها (حرکت به سمت مناطقی از فضای جستجوی جواب مسئله که پیشتر تجربه نشده اند) بوده، در حالیکه یک وزن اینرسی کمتر سبب کوچک شدن فضای جواب مسئله در ناحیه کوچک فعلی می شود. در واقع وزن کمتر باعث می شود که جستجو در مناطقی که در گذشته تجربه شده اند، با دقت بیشتری ادامه پیدا نماید. مقدار w بین دو مقدار بیشینه و کمینه به صورت زیر تغییر می کند:

$$w(k) = w_{\max} - (w_{\max} - w_{\min})k / k_{\max} \quad (6-3)$$

که w مقداری بین $[0,1]$ است و همچنین r_1 و r_2 اعدادی تصادفی در بازه $[0,1]$ می باشد. بهترین موقعیت هر پرنده در کل مراحل طی شده از رابطه زیر بدست می آید:

$$x_{pbest,j}(k) = \begin{cases} x_{pbest,j}(k-1) & \text{if } f(x_j(k)) \geq f(x_{pbest,j}(k-1)) \\ x_j(k) & \text{if } f(x_j(k)) < f(x_{pbest,j}(k-1)) \end{cases} \quad (7-3)$$

و سپس x_{gbest} که نشان دهنده بهترین موقعیت همه پرنده ها می باشد، به صورت زیر در انتهای هر مرحله بدست می آید:

$$x_{gbest}(k) \in \{x_{pbest,1}(k), x_{pbest,2}(k), \dots, x_{pbest,n}(k)\} \\ f(x_{gbest}(k)) = \min \{f(x_{pbest,1}(k)), f(x_{pbest,2}(k)), \dots, f(x_{pbest,n}(k))\} \quad (8-3)$$

می توان الگوریتم اجتماع ذرات را به صورت زیر خلاصه کرد:

گام ۱: در مرحله نخست مقدار دهی تصادفی موقعیت و سرعت تمام پرندگان در فضای مشخص

گام ۲: محاسبه تابع هزینه برای هر پرنده

گام ۳: محاسبه x_{gbest} و x_{pbest} طبق معادلات فوق

گام ۴: به روزرسانی موقعیت و سرعت هر پرنده با توجه به معادلات فوق

گام ۵: افزایش گام $k=k+1$

گام ۶: بازگشت به مرحله ۲ و تکرار مراحل تا ارضا شدن شرط پایانی

شرط پایانی می تواند کوچک شدن خطا به مقدار مطلوب یا تعداد تکرار مشخص شده باشد.

در معادلات بروز رسانی، هر مؤلفه تاثیر به سزایی در همگرایی ذرات به سمت پاسخ مطلوب دارد که در ادامه به معرفی این مؤلفه ها می پردازیم.

۳-۲-۲- مؤلفه های سرعت

مؤلفه اینرسی (سرعت پیشین): حافظه ای است که جهت حرکت قبلی را نگه داری می کند. این مؤلفه حافظه، می تواند به عنوان اینرسی دیده شود که از تغییر شدید جهت حرکت، جلوگیری می کند.

مؤلفه ادراکی: کارایی هر ذره را نسبت به کارایی های قبلی اش می سنجد. مؤلفه ادراکی همانند حافظه ای، موقعیتی که ذره در آن بهتر بوده را حفظ می کند. تأثیر این مؤلفه در بازگشت ذره به بهترین موقعیت قبلی اش است که همانند تمایل افراد به بازگشت به موقعیت یا مکانی که آنها را در گذشته بیشتر ارضا کرده است، می باشد.

مؤلفه ای آموزش اجتماعی: کارایی هر ذره را نسبت به گروهی از ذرات یا همسایگانش می سنجد. از لحاظ مفهومی، مؤلفه ای اجتماعی همانند معیاری است که ذرات برای رسیدن به آن تلاش میکنند. تأثیر این مؤلفه این است که هر ذره به سمت بهترین موقعیت پیدا شده توسط همسایگانش کشیده می شود.

شکل (۳-۱) یک شمای کلی از جهت‌گیری هر ذره پس از در نظر گرفتن تمامی مولفه‌های سرعت را نشان می‌دهد.



شکل (۳-۱) جهت‌گیری ذرات در الگوریتم بهینه‌سازی پرندگان

۳-۲-۳- پارامترهای اصلی

تعداد ذرات: در حالت کلی تعداد ذرات وابسته به ابعاد مسئله است. افزایش تعداد ذرات از یکسو باعث پوشش بیشتر فضای طراحی شده و از سوی دیگر زمان محاسبات را افزایش می‌دهد. یک فضای جستجوی همواره نیاز به تعداد ذرات کمتری نسبت به یک فضای جستجوی ناهموار دارد، تا به جواب بهینه برسد.

اندازه همسایگی: اندازه همسایگی معرف بزرگی تبادلات اجتماعی درون همسایگی است و همسایگی‌های کوچکتر، تبادلات کمتری دارند. در نتیجه در همگرایی کندترند، ولی در همگرایی به بهینه سراسری قابل اطمینان‌تر هستند. همسایگی‌های بزرگتر مستعد در گرفتار شدن در بهینه محلی هستند. ولی همگرایی سریعتری دارند. برای استفاده از مزیت‌های هر دو نوع همسایگی می‌توانیم ابتدا اندازه همسایگی را کوچک بگیریم و با افزایش شماره ی تکرارها اندازه همسایگی را بزرگتر کنیم. این روش، پراکندگی زیاد اولیه، همگرایی سریعتر اجتماع و حرکت ذرات به سمت نواحی امیدوارکننده را تضمین می‌کند.

ثابت شتاب: ذره بهترین موقعیت در همسایگی خودش را با جایگزین کردن بهترین موقعیت جاری پیدا می‌کند و ذرات یک جستجوی محلی را انجام می‌دهند. اگر، هر ذره به سمت بهترین موقعیت شخصی خود کشیده شود که نتیجه اش سرگردانی است. در مقابل اگر، ذرات به سمت بهترین موقعیت سراسری کشیده شوند نتیجه‌اش هجوم پیش از موعد به سمت بهینه است. بزرگ شدن بیش از حد مقدار ثابت شتاب باعث می‌شود مسیر حرکت پرنده نوسانی شود در حالی که مقادیر کمتر مسیری هموارتر ایجاد می‌کند.

معمولاً c_1 و c_2 مناسب به صورت تجربی پیدا می شوند. کندی و همکاران [۵۲] در مقاله ای نشان دادند که باید حاصلجمع c_1 و c_2 کوچکتر از ۴ باشد تا از همگرایی شبکه اطمینان حاصل نمود. در غیر اینصورت سرعتها و موقعیتها به سمت بینهایت میل می کند.

وزن اینرسی: وزن اینرسی تأثیر سرعت های قبلی روی سرعت جدید را کنترل می کند. وزن اینرسی بیشتر، جستجو را در ناحیه گسترده تری با تغییرات ناگهانی همراه می سازد، اما وزن اینرسی کمتر جستجو را در ناحیه محدودتری متمرکز می کند. مقدار بهینه وزن اینرسی به مسئله و ثابتهای شتاب وابسته است. در حقیقت این سه پارامتر به صورت همزمان انتخاب میشوند. رابطه مقابل برای وزن اینرسی پیشنهاد شده و در صورت برقراری این رابطه از رفتار نوسانی و واگرایی پیشگیری خواهد شد.

$$w < \frac{1}{2}(c_1 + c_2) - 1 \quad (9-3)$$

در پیاده سازی های اولیه یک مقدار ثابت را برای تمام طول جستجو، برای تمام ذرات، در هر بعد استفاده می کردند. در پیاده سازی های بعدی از مقادیر متغیر برای مقادیر وزن اینرسی استفاده کردند. این روشها معمولاً با مقادیر زیادی برای اینرسی شروع می شدند که در طول زمان به مقادیر کم کاهش می یافتند. بدین ترتیب، ذرات در زمانهای اولیه جستجو، به اکتشاف و در زمانهای آخر به استخراج نواحی مورد علاقه می پرداختند. روشهای نظیر کاهش خطی [۵۴]، کاهش غیر خطی [۵۵]، تنظیم تصادفی [۵۶] برای وزن اینرسی پیشنهاد شده اند.

۳-۳ تاثیر میزان جمعیت و ضرایب یادگیری در عملکرد الگوریتم بهینه سازی

پرندهگان

تحلیل های جامعی برای تاثیر میزان جمعیت و ضرایب یادگیری در عملکرد الگوریتم PSO ارائه شده است که نشان می دهد افزایش ضرایب یادگیری، همواره باعث دیرتر همگرا شدن الگوریتم می شود. در عین حال این امر باعث بدست آمدن جواب های بهتری می شود [۵۷].

از آنجا که ضریب c_1 نشان دهنده اتکای هر پرنده به تجارب شخصی خود است، طبیعی است که افزایش آن باعث کندتر شدن همگرایی الگوریتم می شود و چون c_1 یک مولفه یادگیری است و بیشتر شدن آن باعث تقویت جریان تبادل اطلاعاتی می شود افزایش آن به بهتر شدن جواب ها کمک می کند. هرچند که بهترین پاسخ

ممکن توسط یک پرنده اتفاق نمی افتد اما به هرحال تصمیمی که با استفاده از تجارب گذشته باشد، قطعاً تصمیم نامناسبی نخواهد بود. ولی این تاثیر زیاد نخواهد بود به همین دلیل از مقدار میانه برای این ضریب یعنی عددی مانند ۰.۷۵ یا ۱ استفاده می شود.

با افزایش ضریب c_2 که نشان دهنده تبعیت هر پرنده از بهترین پرنده است، پاسخ بهتری بدست می آید. هرچند که همگرایی را کند می کند اما تاثیر آن بر کاهش سرعت همگرایی از ضریب c_1 کمتر است. در واقع حضور این ضریب در رسیدن به پاسخ صحیح نقش موثری دارد. یعنی زمانی که این ضریب صفر است، در پاسخ ها خطا دیده می شود اما در زمانی که این ضریب مقدار دارد این خطا کاهش می یابد.

در توسعه های دیگری از الگوریتم PSO در معادله به روز رسانی سرعت از مولفه دیگری که مبتنی بر استفاده از تجارب پرندگان همسایه است، نیز بهره می گیرند. در نتیجه داریم:

$$V_i(t+1) = wV_i(t) + c_1r_{1,i}(t)(P_{pbest}(t) - X_i(t)) + c_2r_{2,i}(t)(P_{gbest}(t) - X_i(t)) + c_3r_{3,i}(t)(P_{nbest}(t) - X_i(t)) \quad (10-3)$$

$P_{nbest}(t)$ نشان دهنده بهترین جوابی است که همسایگان پرنده i ام با یک شعاع مشخص بدست آورده اند. با افزایش ضریب c_3 نیز جواب نهایی بهبود می یابد و سرعت همگرایی کاهش می یابد. در بخش بعد به معرفی نوع همسایگی می پردازیم.

نتایج تجربی نشان می دهد که بیشتر شدن جمعیت اولیه در رسیدن به پاسخ بهتر بسیار موثر است اما در توابع مختلف افزایش تعداد جمعیت اولیه تاثیر چندانی در کاهش تابع هزینه ندارد. در بیشتر توابع میزان جمعیت اولیه ۳۲ یا ۶۴ مناسب است.

۳-۴ انواع ساختارهای همسایگی در الگوریتم PSO

پرندگان تمایل دارند که از پرندگان موفق در ارتباط با خود تاثیر بپذیرند. این همسایگان لزوماً پرندگانی نیستند که در فضای پارامتریک به آنها نزدیک تر هستند. بلکه پرندگانی هستند که براساس مکان شناسی همسایگی تعریف شده در ساختار اجتماعی پرندگان، به آنان نزدیک تر هستند. توپولوژی همسایگی به طرز عجیبی بر روی نرخ همگرایی الگوریتم PSO تاثیر گذار می باشد. از این رو باید به این مسئله توجه فراوانی شود.

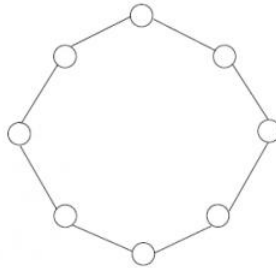
پرندگان می توانند براساس توپولوژی همسایگی تعریف شده با سایر پرندگان ارتباط برقرار کنند. از این رو بعضی از گراف ها و توپولوژی های همسایگی با توجه به معادله (۳-۱۱) ارائه شده در PSO عبارتند از:

$$V_i(t+1) = wV_i(t) + c_1r_{1,i}(t)(P_{pbest}(t) - X_i(t)) + c_2r_{2,i}(t)(P_{leader}(t) - X_i(t)) \quad (11-3)$$

که در آن P_{leader} پرنده ایست که به عنوان راهنما برای سایر پرندگان استفاده شده و سایر پرندگان را هدایت می-کند. P_{lbest} را پرنده ای تعریف می کنیم که بهترین پاسخ را در همسایگی یک پرنده معین دارد.

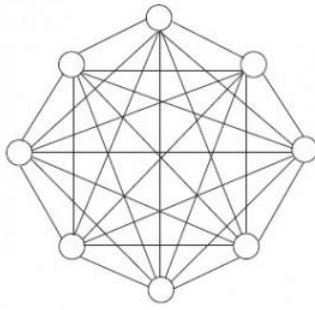
(۱) گراف تهی: در این توپولوژی پرنده تنها می باشد و فقط با خودش در ارتباط بوده و با سایر پرندگان ارتباط ندارد. در این مورد در معادله (۱۰-۳)، $c_2 = 0$ می باشد.

(۲) گراف حلقوی: در این توپولوژی هر پرنده از بهترین عملکرد k همسایه کناری خود تاثیر می پذیرد. اگر $k=2$ باشد گراف بصورت حلقوی می باشد در معادله (۱۰-۳) $P_{leader} = P_{lbest}$ می باشد. شکل (۲-۳) را مشاهده فرمایید.



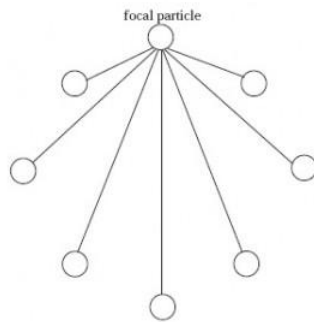
شکل (۲-۳) گراف حلقوی

(۳) گراف ارتباط کامل: در این توپولوژی بر خلاف گراف تهی هر پرنده از گروه با تمام اعضای گروه در ارتباط است. پرنده از حافظه تاریخی خود و بهترین جوابی که یافته (pbest) استفاده کرده، بعلاوه از بهترین جوابی که گروه تاکنون یافته است (gbest) نیز استفاده می کند. در معادله (۱۰-۳) $P_{leader} = P_{gbest}$ می باشد. شکل زیر را ببینید.



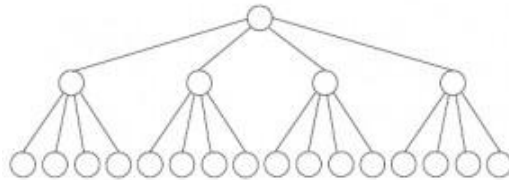
شکل (۳-۳) گراف ارتباط کامل

(۴) شبکه ستاره: در این توپولوژی یک پرنده (به نام focal) با همه پرندگان در ارتباط است اما سایر پرندگان با یکدیگر رابطه ندارند. یعنی سایر پرندگان عملکرد خود را با focal مقایسه می کنند. در معادله دوم $P_{leader} = P_{focal}$ می باشد. شکل (۴-۳) را ببینید.



شکل (۴-۳) شبکه ستاره

(۵) شبکه درختی: در این توپولوژی هر پرنده دارای یک والد بوده و با آن والد در ارتباط است اما سایر پرندگان با یکدیگر در ارتباط نیستند. این توپولوژی یک همسایگی دینامیک را ارائه می دهد. در معادله دوم $P_{leader} = P_{parent}$ می باشد. شکل زیر را ببینید.



شکل (۳-۵) شبکه درختی

۳-۵ توسعه الگوریتم بهینه سازی پرندگان

الگوریتم بهینه سازی پرندگان عملکرد مناسبی در حل مسائل غیرخطی و پیچیده دارد. این الگوریتم در مقایسه با الگوریتم ژنتیک به خاطر نداشتن عملگرهای پیچیده سرعت همگرایی بالایی دارد. اما پاسخ این الگوریتم های تکاملی را نمی توان به طور قطع نقطه کمینه یا بیشینه سراسری دانست و ممکن است در کمینه یا بیشینه محلی گرفتار شوند. محققان با توسعه الگوریتم بهینه سازی پرندگان سعی در رفع این نقطه ضعف نموده اند.

۳-۵-۱ الگوریتم بهینه سازی پرندگان با سرعت آشفتهگی تفاضلی

داس و همکاران در [۵۸] طرحی جدید از بروزرسانی سرعت ذرات در الگوریتم اجتماع ذرات را با استفاده از یک عملگر برداری دیفرانسیلی برگرفته از خانواده الگوریتم تکامل تفاضلی [۵۹] پیشنهاد کرده اند. همان طور که ذکر شد، سرعت هر ذره در الگوریتم اجتماع ذرات دارای سه قسمت اصلی است. در طرح ارائه شده توسط [۵۸]، قسمت دانش فردی ذره در بروز رسانی سرعت آن حذف شده و به جای آن یک قسمت جدید شامل اختلاف موقعیت دو ذره مجزا در فضای جستجو اضافه می گردد. این دو ذره به صورت تصادفی از میان ذرات موجود در فضای جستجو انتخاب می شوند. قسمت سرعت تفاضلی از به روز رسانی سرعت ذرات از طرح عملگر جهش در الگوریتم تکامل تفاضلی انتخاب شده و از این رو در مرجع [۵۸]، الگوریتم بهینه سازی اجتماع ذرات با PSO-DV پیشنهادی به عنوان سرعت آشفتهگی تفاضلی نام گذاری شده است.

در الگوریتم پیشنهادی، برای هر ذره i در فضای جستجو، دو ذره دیگر j و k ، $(i \neq j \neq k)$ به صورت تصادفی انتخاب می شوند. اختلاف بین موقعیت دو ذره انتخابی به عنوان یک بردار تفاضلی به صورت رابطه (۳-۱۲) تعریف می-شود.

$$\delta_i = P_k - P_j \quad (۳-۱۲)$$

سپس سرعت آزمایشی برای ذره مورد نظر λ_m به صورت رابطه (۳-۱۳) به روز رسانی می‌شود.

$$V_{i+1} = \begin{cases} K(wV_i + \beta\delta_i + C_{2i}(P_{gbest} - P_i)) & \text{if } r < CR \\ V_i & \text{otherwise} \end{cases} \quad (۳-۱۳)$$

که در آن CR احتمال وقوع عملگر تقاطع، r یک عدد تصادفی در بازه $[0, 1]$ و β نیز یک عدد اسکالر است که در بازه $[0, 1]$ انتخاب می‌شوند. با محاسبه سرعت جدید، موقعیت جدید آزمایشی برای ذره λ_m ، یعنی T_i از طریق جمع سرعت جدید حاصل از روابط (۳-۱۳) با موقعیت قبلی ذره به صورت رابطه (۳-۱۴) بدست می‌آید.

$$T_i = P_i + V_{i+1} \quad (۳-۱۴)$$

موقعیت T_i ، در صورتی جایگزین موقعیت فعلی می‌شود که نسبت به آن دارای برابری بهتری باشد. اگر هدف بهینه سازی، کمینه سازی تابع هدف F باشد، آنگاه ذره λ_m به صورت زیر بروز رسانی می‌گردد.

$$P_{i+1} = \begin{cases} T_i & \text{if } F(T_i) \leq F(P_i) \\ P_i & \text{otherwise} \end{cases} \quad (۳-۱۵)$$

۳-۵-۲ الگوریتم بهینه سازی پرندگان توسعه یافته STS-PSO

الگوریتم‌های بهینه سازی از مقادیر اولیه تصادفی شروع به کار می‌کنند و سعی می‌کنند به پاسخ بهینه نزدیک شوند. سپس بعد از رسیدن به معیار مشخصی کار جستجو را پایان می‌دهند. اما در بعضی مواقع قبل از رسیدن به معیار از پیش تعریف شده کار جستجو را راکد می‌شود و پیشرفتی حاصل نمی‌شود. این امر می‌تواند ناشی از گرفتار شدن الگوریتم در کمینه محلی باشد. در چنین شرایطی بعید است که مقدار کمینه سراسری در فضای جستجوی اولیه وجود داشته باشد. در نتیجه باید فضای جستجو را منتقل نمود. برای اولین بار ونگ و همکارانش [۶۱] الگوریتم انتقال فضای جستجو^۱ را معرفی نمودند.

در این الگوریتم اگر X بردار فضای کنونی باشد، با استفاده از معادله (۳-۱۶) به فضای X^* منتقل می‌شود.

^۱ Space Transformation Search

$$\begin{aligned}
X_{ij}^* &= k[a_j(t) + b_j(t)] - X_{ij}, \\
a_j(t) &= \min(X_{ij}(t)), b_j(t) = \max(X_{ij}(t)) \\
i &= 1, 2, \dots, Popsize, j = 1, 2, \dots, n
\end{aligned}
\tag{۱۶-۳}$$

$X \in \square$ در بازه ی $[a, b]$ قرار دارد.

در بسیاری از موارد در زمانی که سرعت پرندگان در نزدیکی نقطه کمینه زیاد باشد باعث می شود از نقطه کمینه پرش کند. یا اگر سرعت کم باشد باعث همگرایی دیررس می شود. برای جلوگیری از این مشکلات راه حل های مناسبی در مراجع ارائه شده است.

فصل چهارم

طراحی کنترل کننده بهینه خطی با تضمین پایداری با استفاده از الگوریتم
بهینه سازی پرندگان

۴-۱ مقدمه

تئوری کنترل بهینه مربعی خطی^۱ مبنای بسیاری از تئوری های کنترلی می باشد و به سادگی قابل پیاده سازی است. به همین علت بسیار مورد اهمیت است. این تئوری در کنترل بسیاری از سیستمها، مانند کنترل موتور القایی و کنترل میل لنگ خودرو به کار گرفته شده است [۶۳-۶۴]. در طراحی این روش کنترلی ماتریسهای وزنی وجود دارد که توسط طراح تنظیم می شود. در مورد انتخاب ماتریس های وزنی روش های مختلفی پیشنهاد شده است. کالمن [۶۵] برای اولین بار روشی را برای تعیین ماتریس های وزنی بر اساس قطب های داده شده ارائه نمود. سپس ونگ [۶۶] تحقیقات او را توسعه داد. اخیرا تلاش های بسیاری برای انتخاب این ماتریس ها توسط الگوریتم های بهینه سازی هوشمند مانند الگوریتم ژنتیک و الگوریتم بهینه سازی پرندگان [۶۷-۶۸] انجام شده است.

به دنبال یافتن روش کنترلی که با استفاده از الگوریتم بهینه سازی پرندگان، بتوان پایداری را تضمین نمود، باید تابع هدف مناسبی را یافت که برای انواع سیستم ها کارآمد باشد. در این فصل از تابع هدف در تئوری کنترل بهینه مربعی خطی، برای طراحی کنترل کننده با هدف فوق استفاده شده است. در اینجا با ترکیب الگوریتم بهینه سازی پرندگان و تئوری بهینه مربعی خطی روشی برای کنترل سیستم های خطی پیشنهاد شده است. سپس به مسائل غیر خطی و در نهایت ربات با مفاصل منعطف تعمیم داده می شود. نتایج حاصل در بخش شبیه سازی بر روی ربات دورابط با مفاصل منعطف پیاده سازی می شود. در نهایت مزایا و معایب روش پیشنهادی ارائه می شود.

۴-۲ تئوری کنترل بهینه مربعی خطی

سیستم خطی زیر را در نظر بگیرید.

$$\dot{X} = AX + bu \quad (1-4)$$

u ورودی سیستم و X متغیرهای حالت می باشد. ورودی سیستم را به صورت زیر پیشنهاد می دهیم:

$$u = -K^T X \quad (2-4)$$

در این روش هدف کمینه کردن تابع J در معادله (۴-۳) می باشد.

^۱ Linear Quadratic Regulator

$$J = \int_0^{\infty} (X^T Q X + r u^2) dt \quad (3-4)$$

Q و r ماتریس‌های وزنی هستند که به ترتیب نیمه معین مثبت و معین مثبت هستند. برمبنای تئوری لیاپانوف، تابع منتخب لیاپانوف زیر را پیشنهاد می‌دهیم:

$$V = X^T P X \quad (4-4)$$

که P ماتریس مثبت معین است. فرض کنید P وجود دارد که :

$$X^T Q X + r u^2 = -\frac{d}{dt} (X^T P X) \quad (5-4)$$

با جایگذاری معادله (5-4) در معادله (3-4) داریم:

$$J = X^T(0) P X(0) - X^T(\infty) P X(\infty) \quad (6-4)$$

نقطه تعادل سیستم $\dot{X} = AX$ صفر است، با فرض پایداری سیستم مقدار X در بی نهایت برابر صفر است در نتیجه:

$$X^T(\infty) P X(\infty) = 0 \quad (7-4)$$

پس معادله (6-4) به صورت زیر بازنویسی می‌شود:

$$J = X^T(0) P X(0) \quad (8-4)$$

در نقطه ای که تابع هدف کمینه می‌شود، مشتق تابع هدف نسبت به پارامترها برابر صفر است. در نتیجه داریم:

$$\frac{\partial J}{\partial K} = 0 \rightarrow \frac{\partial P}{\partial K} = 0 \quad (9-4)$$

مشتق تابع منتخب لیاپانوف V طبق معادله (5-4) منفی است. از معادله (4-4) و (5-4) می‌توان نوشت:

$$\dot{V} = -(X^T Q X + r u^2) \quad (10-4)$$

در نتیجه برمبنای تئوری لیاپانوف پایداری تضمین می‌شود.

با در نظر گرفتن رابطه (۴-۵) و بکارگیری $u = -K^T X$ می توان نوشت:

$$\begin{aligned} -(X^T QX + ru^2) &= \dot{X}^T P X + X^T P \dot{X} \\ &= (X^T A^T - X^T K b^T) P X + X^T P (A X - b K^T X) \\ &= X^T (A^T P + P A) X - X^T (K b^T P + P b K^T) X \end{aligned} \quad (۴-۱۱)$$

از طرفی داریم :

$$\begin{aligned} -(X^T QX + ru^2) &= -(X^T QX + X^T K r K^T X) \\ &= -(X^T (Q + K r K^T) X) \end{aligned} \quad (۴-۱۲)$$

بنابراین، از (۴-۱۱) و (۴-۱۲) می توان نوشت:

$$Q = 2K b^T P - K r K^T - A^T P - P A \quad (۴-۱۳)$$

در نتیجه با پیشنهاد ماتریس وزنی Q مثبت معین، مقدار P متناسب با آن از رابطه (۴-۱۳) بدست می آید. با روندگیری نسبت به K از طرفین (۴-۱۳) و در نظر داشتن آنکه P و Q ثابت هستند، می توان نوشت:

$$K = r^{-1} b^T P \quad (۴-۱۴)$$

پس قانون کنترل بدست می آید:

$$u = -r^{-1} b^T p X \quad (۴-۱۵)$$

برای پیشنهاد ماتریس Q به گونه ای که عملکرد بهینه ایجاد شود، روش های مختلفی از جمله استفاده از الگوریتم های بهینه سازی به کار گرفته شده است [۶۷].

۴-۳ توسعه کنترل بهینه مربعی خطی با استفاده از الگوریتم بهینه سازی

پرنندگان

در این روش برای یافتن پارامترهای کنترل کننده در معادله (۴-۱۵) که نیاز به محاسبه P دارد، دیگر به سراغ معادله (۴-۱۳) نمی رویم و به طور مستقیم با استفاده از الگوریتم بهینه سازی پرنندگان، ماتریس P را مقدار

دهی می‌کنیم. در این روش، الگوریتم بهینه سازی پرندگان در حالتی که سیستم خارج خط است، سعی در کمینه کردن تابع J در معادله (۳-۴) دارد. الگوریتم بهینه سازی پرندگان، با تغییر پارامتر P در هر بار تکرار، مقدار J را تا مقدار کمینه کاهش می‌دهد.

با توجه به آنکه برقراری رابطه (۱۵-۴) از بهینه سازی تابع هدف (۳-۴) نتیجه شده است. بنابراین، روش پیشنهادی گه با الگوریتم بهینه سازی پرندگان به بهینه سازی (۳-۴) می‌انجامد، همین نتیجه را خواهد داد. ضمن آنکه تشکیل ورودی سیستم از معادله (۱۵-۴) خود تضمین کننده پایداری سیستم است.

می‌توان نتایج حاصل را به ردگیری تعمیم داد. برای این منظور داریم:

$$\dot{e} = Ae + bv \quad (۱۶-۴)$$

که در آن v ورودی سیستم ردگیری می‌باشد و در رابطه (۱۷-۴) به صورت خطی تعریف می‌شود.

$$v = Ke \quad (۱۷-۴)$$

همچنین مقدار خطا برابر رابطه (۱۸-۴) می‌باشد.

$$e = X_d - X \quad (۱۸-۴)$$

جایگزینی مقدار خطا در معادله (۱۶-۴) داریم:

$$\begin{aligned} \dot{X}_d - (AX + bu) &= Ae + bv \\ \dot{X}_d - (AX + bu) &= A(X_d - X) + bk(X_d - X) \\ u &= b^{-1}(\dot{X}_d - AX_d - bk(X_d - X)) \end{aligned} \quad (۱۹-۴)$$

همانطور که در معادله (۱۹-۴) مشاهده می‌شود، ورودی سیستم به ماتریس A وابسته می‌شود.

۴-۴ تعمیم تئوری کنترل مربعی خطی به سیستم های غیر خطی

برای استفاده از این روش در سیستم غیر خطی باید سیستم در نزدیکی نقطه تعادل خطی سازی شود. سیستم غیرخطی زیر را در نظر بگیرید:

$$\dot{X} = f(X, u) \quad (۲۰-۴)$$

نقطه تعادل جایی است که سیستم در آن به آرامش می رسد. یعنی:

$$\dot{X} = 0 \rightarrow f(X, u)|_{X^*, u^*} = 0 \quad (21-4)$$

که در آن X^*, u^* نقطه $(0,0)$ می باشند. حال فرض می کنیم که تغییر کوچکی در اطراف نقطه $(0,0)$ رخ داده باشد.

$$\begin{aligned} X &\rightarrow X^* + \Delta X \\ u &\rightarrow u^* + \Delta u \end{aligned} \quad (22-4)$$

بنابراین، داریم:

$$\dot{X} = f(x, u) \rightarrow \dot{X}^* + \Delta \dot{X} = f(X^* + \Delta X, u^* + \Delta u) \quad (23-4)$$

با توجه به بسط تیلور می توان نوشت:

$$f(z) = \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{\partial^n f(z_0)}{\partial z^n} (z - z_0)^n \right) \quad (24-4)$$

$$z = (X, u)$$

$$z_0 = (X^*, u^*)$$

$$f(z) = f(z_0) + \left. \frac{\partial f}{\partial X} \right|_{z_0} \Delta X + \dots + \left. \frac{\partial f}{\partial u} \right|_{z_0} \Delta u + \dots \quad (25-4)$$

با صرف نظر کردن از مولفه های مرتبه بالاتر داریم:

$$f(z) = \left. \frac{\partial f}{\partial X} \right|_{z_0} \Delta X + \left. \frac{\partial f}{\partial u} \right|_{z_0} \Delta u \quad (26-4)$$

در نتیجه در نزدیکی نقطه $(0,0)$ داریم:

$$f(z) = AX + bu$$

$$A = \left. \frac{\partial f}{\partial X} \right|_{z_0}, \quad b = \left. \frac{\partial f}{\partial u} \right|_{z_0} \quad (27-4)$$

می توان نتایج حاصل در ردگیری سیستم های خطی را نیز به سیستم های غیرخطی تعمیم داد. برای این منظور داریم:

$$\dot{e} = f(e, v) \quad (28-4)$$

که در آن $e = X_d - X$ می باشد. اگر سیستم در اطراف نقطه $(0,0)$ خطی سازی شود، داریم:

$$\dot{e} = Ae + bv \quad (29-4)$$

که در آن v ورودی سیستم ردگیری می باشد. در تعمیم این معادله به روش ردگیری باید توجه داشت که برای سیستم، نقطه تعادل صفر مطلوب می باشد. زیرا سیستم در حالت پایدار به سمت نقطه تعادل خود میل می کند. اگر نقطه تعادل صفر نباشد، خطای سیستم که حال به طور مستقیم، متغیر حالت سیستم است به سمت صفر میل نمی کند. با جایگزینی مقدار خطا در معادله (4-25) داریم:

$$\begin{aligned} \dot{X}_d - (AX + bu) &= Ae + bv \\ \dot{X}_d - (AX + bu) &= A(X_d - X) + bk(X_d - X) \\ u &= b^{-1}(\dot{X}_d - AX_d - bk(X_d - X)) \end{aligned} \quad (30-4)$$

همانطور که در معادله (4-26) مشاهده می شود، ورودی سیستم به ماتریس A وابسته می شود. در مواردی که مدل سیستم پیچیده است، این وابستگی باعث افزایش محاسبات و در نتیجه ناکارآمدی کنترل کننده می شود. برای رفع این مشکل می توان مسیر مطلوب ورودی را به گونه ای پیشنهاد داد که شرط زیر برقرار باشد.

$$\dot{X}_d = AX_d \quad (31-4)$$

در نتیجه در معادله (4-26) ترم های مربوط به مدل سیستم حذف می شود و می توان نوشت:

$$u = -k(X_d - X) \quad (32-4)$$

در مسیر های نمایی امکان برقراری این شرط می باشد. این ویژگی در بسیاری از کاربردهای صنعتی فراهم می شود. بنابراین، از کنترل کننده پیشنهادی می توان به نحو مطلوب استفاده کرد.

۴-۵ شبیه سازی

به منظور ارزیابی روش ارائه شده، نتایج را برای سیستم خطی و بازوی دو محوره شبیه سازی و بررسی می نماییم. سیستم خطی زیر را در نظر بگیرید:

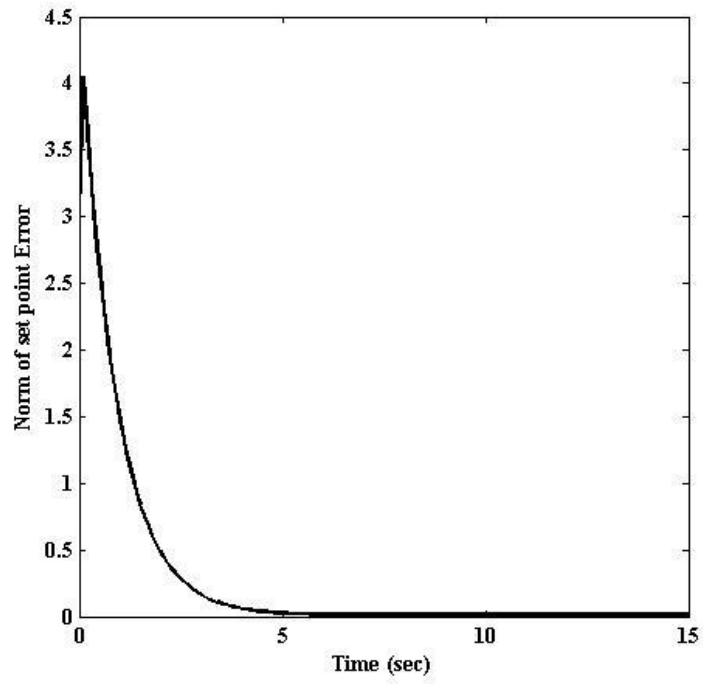
$$\dot{X} = AX + bu \quad (۴-۳۳)$$

که در آن ماتریس A و b به صورت زیر مقدار دهی شده است:

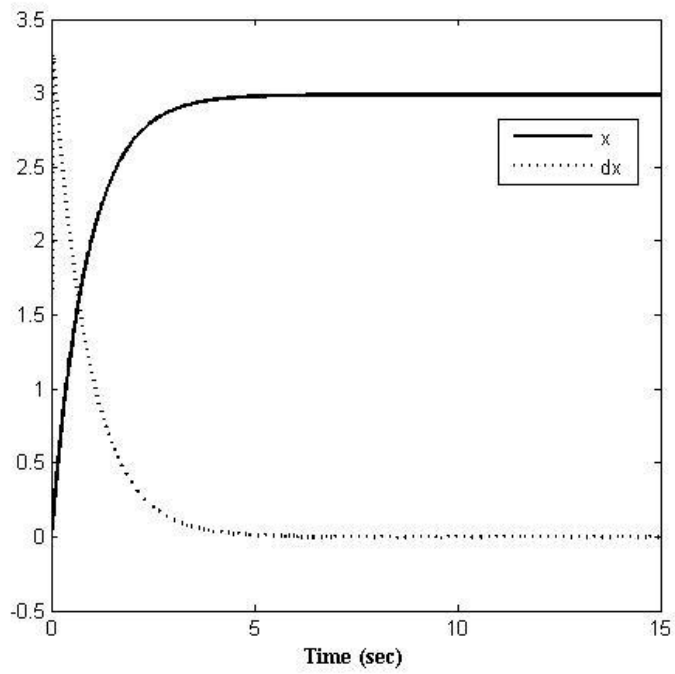
$$A = \begin{bmatrix} 0 & 1 \\ -1 & -3 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \quad X = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

روش ارائه شده در بخش (۴-۳) بر روی سیستم خطی (۴-۳۳) پیاده سازی می شود. بنابراین در ابتدا شبیه سازی برای رسیدن به نقطه مطلوب و سپس برای ردگیری مسیر طراحی شده اجرا شده است. در این طراحی نقطه مطلوب $X=(۳,۰)$ می باشد. ۳۲ گروه از پرندگان در الگوریتم بهینه سازی پرندگان در نظر گرفته شده است. الگوریتم بهینه سازی پرندگان ۵ بار تکرار شده است و در هر دوره ۳۰ تکرار^۱ به صورت خارج خط انجام شده است که بهترین نتیجه بدست آمده ارائه شده است. در شکل (۴-۱) نرم خطای عملکرد بهترین پارامتر نشان داده شده است. پرش ایجاد شده در لحظه اول ناشی از پرش مولفه دوم X می باشد. در شکل (۴-۲) نحوه تنظیم دو مولفه X را در نقطه مطلوب مشاهده می کنید. همانطور که دیده می شود مولفه اول بدون سرریز و در مدت زمان کمتر از ۳ ثانیه به نقطه مطلوب نزدیک شده است. در شکل (۴-۳) روند کاهش تابع هدف قابل مشاهده است.

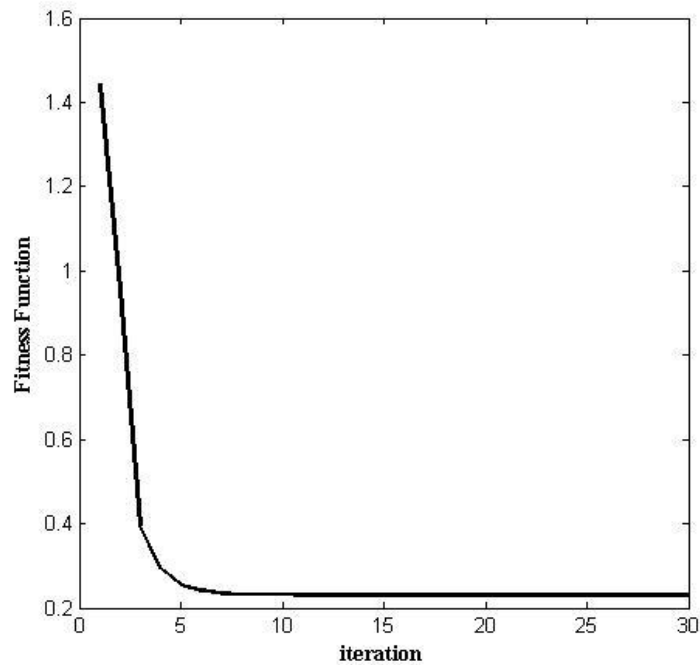
^۱ iteration



شکل (۴-۱) نرم خطای تنظیم در نقطه مطلوب



شکل (۴-۲) تنظیم در نقطه مطلوب



شکل (۴-۳) روند کاهش تابع هدف

حال به شبیه سازی ردگیری سیستم با استفاده از کنترل کننده پیشنهادی می پردازیم. طبق مطالب بیان شده در بخش (۴-۳) باید ورودی سیستم را با استفاده از رابطه (۴-۲۰) به صورت زیر بدست بیاوریم:

$$bu = (\dot{X}_d - AX_d - bk(X_d - X)) \quad (۴-۳۴)$$

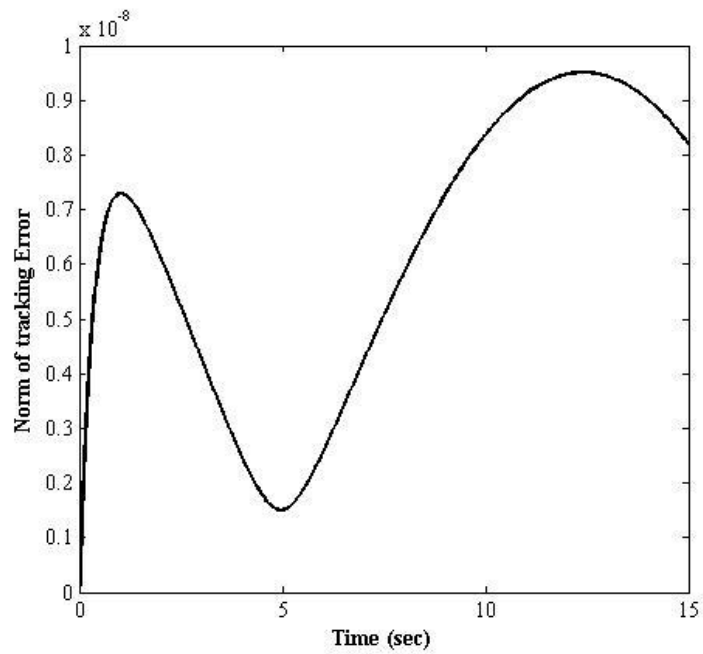
مسیر مطلوب در این شبیه سازی به صورت رابطه (۴-۳۴) تعریف شده است.

$$x_d = 0.5(1 - \cos(\pi t / 15)) \quad (۴-۳۵)$$

در نتیجه، معادله (۴-۳۵) با قرار دادن پارامترهای سیستم به صورت زیر بدست می آید.

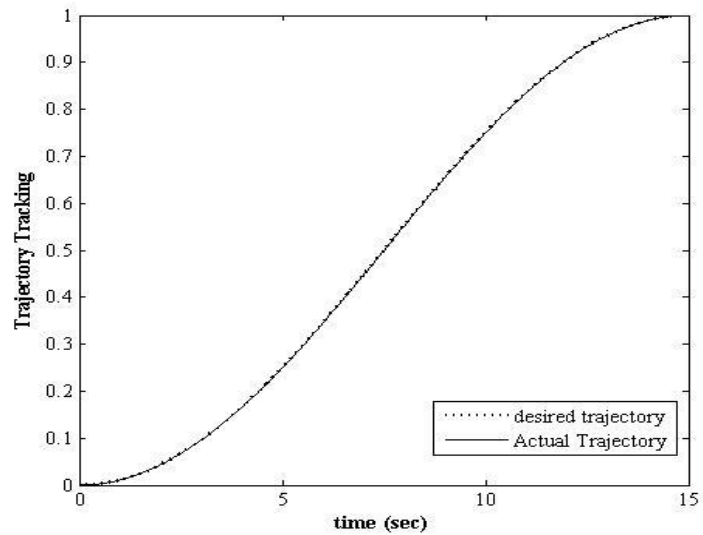
$$bu = \begin{pmatrix} 0.5 \sin(\pi t / 15) \\ \frac{\pi}{30} \cos \pi t / 15 \end{pmatrix} - \begin{bmatrix} 0 & 1 \\ -1 & -3 \end{bmatrix} \begin{bmatrix} 0.5(1 - \cos(\pi t / 15)) \\ 0.5 \sin(\pi t / 15) \end{bmatrix} - \begin{bmatrix} 0 \\ 3 \end{bmatrix} k(X_d - X) \quad (۴-۳۶)$$

در شکل (۴-۴) نتیجه عملکرد الگوریتم بهینه سازی پس از ۳۰ بار تکرار مشاهده می‌شود. شکل (۴-۴) نرم خطای ردگیری را در ۱۵ ثانیه نشان می‌دهد. نرم خطا همواره از 10^{-8} کمتر می‌باشد.



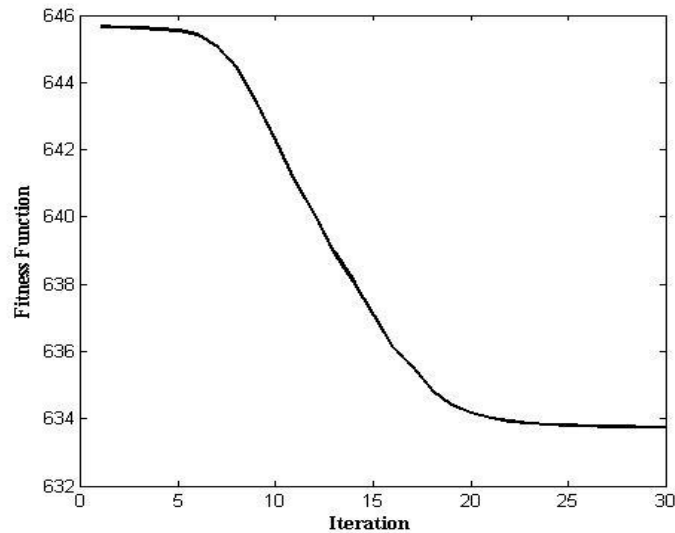
شکل (۴-۴) نرم خطای ردگیری سیستم خطی

در شکل (۵-۴) مسیر ردگیری سیستم خطی مشاهده می‌شود که بسیار نزدیک به مسیر مرجع می‌باشد.



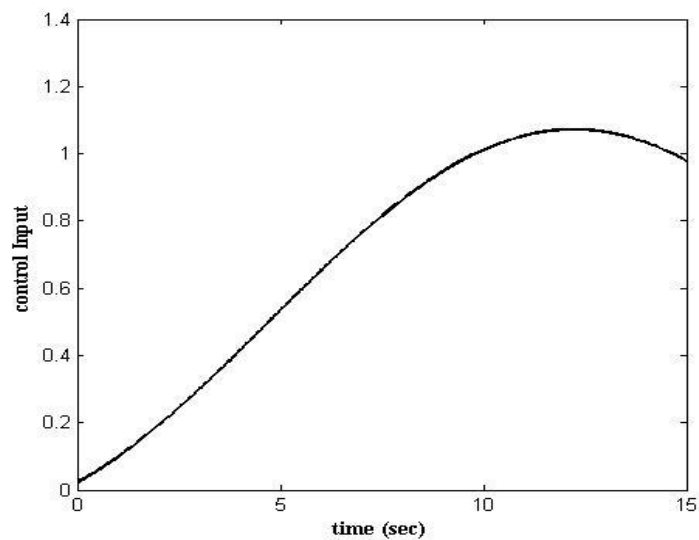
شکل (۵-۴) مسیر ردگیری سیستم خطی

در شکل (۴-۶) مشاهده می‌شود که تابع هدف پس از ۲۰ بار تکرار الگوریتم، به مقدار ثابتی همگرا شده است. با این حال بهترین پارامتر پس از ۳۰ بار تکرار در حالت خارج خط انتخاب شده است.



شکل (۴-۶) روند کاهش تابع هدف در الگوریتم PSO

برای اطمینان از قابلیت پیاده سازی کنترل کننده در سیستم های خطی باید سیگنال کنترلی مورد بررسی قرار بگیرد. همانطور که در شکل (۴-۷) دیده می‌شود سیگنال کنترلی بدون لغزش می‌باشد.



شکل (۴-۷) سیگنال کنترلی ورودی

در اینجا پس از نشان دادن قابلیت کنترل کننده پیشنهادی در کنترل سیستم های خطی، به سراغ پیاده سازی نوع تعمیم یافته آن بر روی سیستم غیرخطی ربات با مفاصل منعطف با مشخصات داده شده در جدول (۴-۱) و (۴-۲) می‌رویم. از آنجا که بعلت تزویج سنگین معادلات بازوی رباتیک، برای بدست آوردن مدل خطی سیستم در اطراف نقطه تعادل نیاز به محاسبات فراوانی می‌باشیم، بنابراین، مسیر مرجع یک مسیر نمایی به شکل زیر برای هر دو لینک در فضای مفصلی است، تا شرایط ذکر شده در معادله (۴-۳۲) تا حدودی برقرار باشد. در نتیجه مسلما به دلیل ساده سازی های انجام شده با خطا مواجه خواهیم بود.

$$\begin{cases} \theta_{d1} = -\pi/2 - 0.42e^{(-0.5t)} \\ \theta_{d2} = -e^{(-0.5t)} \end{cases} \quad (۴-۳۷)$$

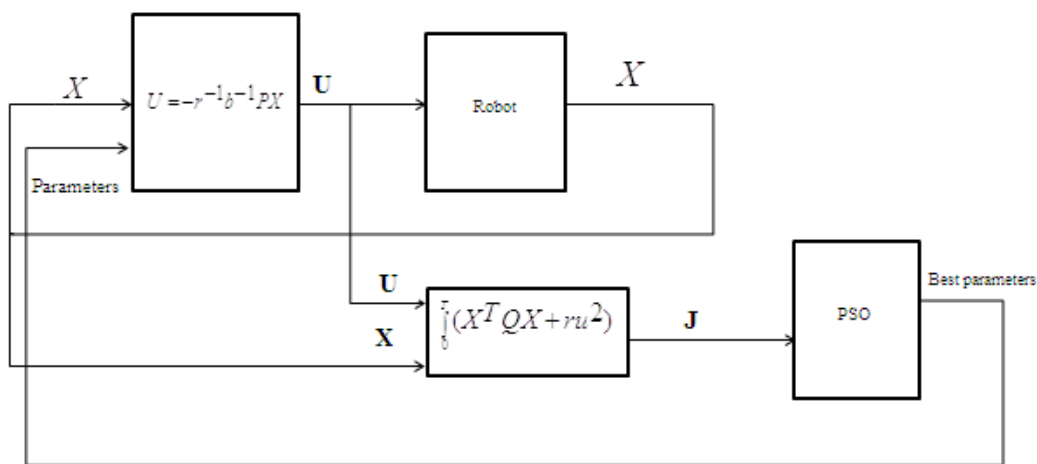
جدول (۴-۱) مشخصات موتورها

موتور	$R(\Omega)$	$K_s(Nm/rad)$	$J_m(kgm^2)$	$L(H)$	$K_b(Volt/rad/sec)$	$B(Nmsec/rad)$
۲ و ۱	۱.۶	۵۰۰	۰.۰۰۰۲	۰.۰۰۱	۰.۲۶	۰.۰۰۱

جدول (۴-۲) مشخصات ربات

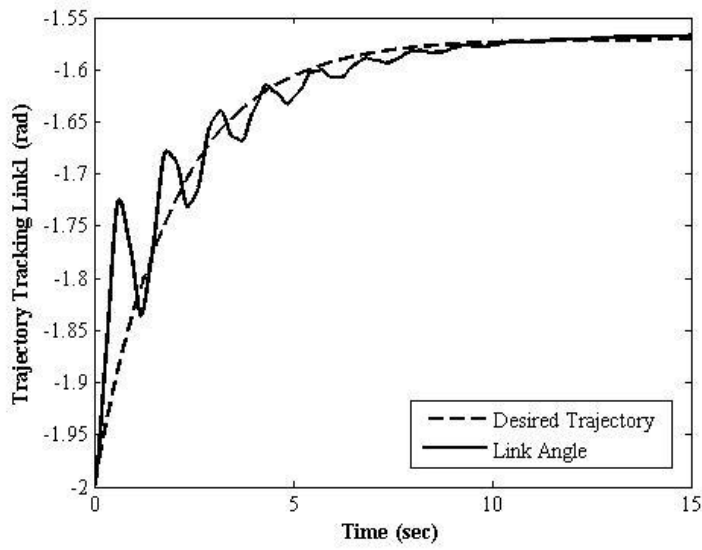
لینک	$l(m)$	$l_{ci}(m)$	$m_i(kg)$	$I_i(kgm^2)$
۱	۱	۰.۵	۱۵	۵
۲	۱	۰.۵	۶	۲

نقاط تعادل در سیستم ربات دو رابط برای رابط اول $-\frac{\pi}{2}$ و برای رابط دوم صفر می‌باشد. بلوک دیاگرام روش کنترل پیشنهادی در شکل (۸-۴) ارائه شده است. برای دو رابط به طور مستقل و همزمان دو الگوریتم بهینه سازی سعی در کمینه کردن تابع هدف دارند. در نتیجه J در شکل (۸-۴) یک بردار متشکل از توابع هدف دورابط است. پارامترهای p_1 و p_2 به ترتیب برای تشکیل ورودی های رابط اول و دوم به کار گرفته شده است. ورودی سیستم رباتیک ولتاژ موتورها در نظر گرفته شده است. الگوریتم بهینه سازی در هر بار تکرار پارامترهای p_1 و p_2 را برای بهینه کردن تابع J تغییر می دهد.

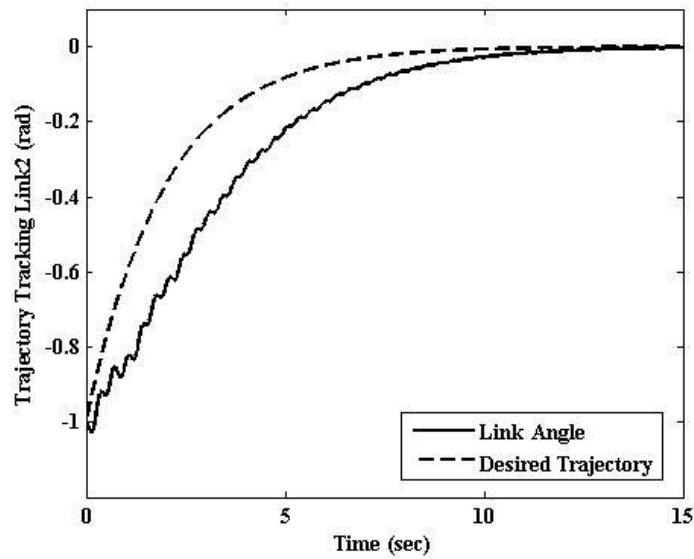


شکل (۸-۴) بلوک دیاگرام کنترل کننده پیشنهادی

همانطور که در شکل های (۹-۴) و (۱۰-۴) مشاهده می‌شود، تا زمانی که متغیرهای حالت از نقطه تعادل دور هستند، سیستم پاسخ مناسبی ندارد. ولی در زمانی که به نقاط تعادل نزدیک می‌شوند، نوسان ها کاهش می‌یابد و خطا به سمت صفر میل می کند.

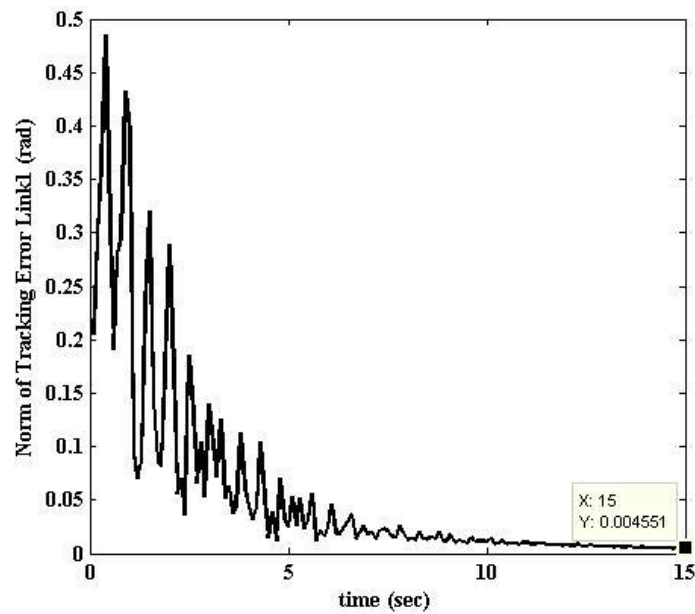


شکل (۴-۹) مسیر ردگیری رابط یک

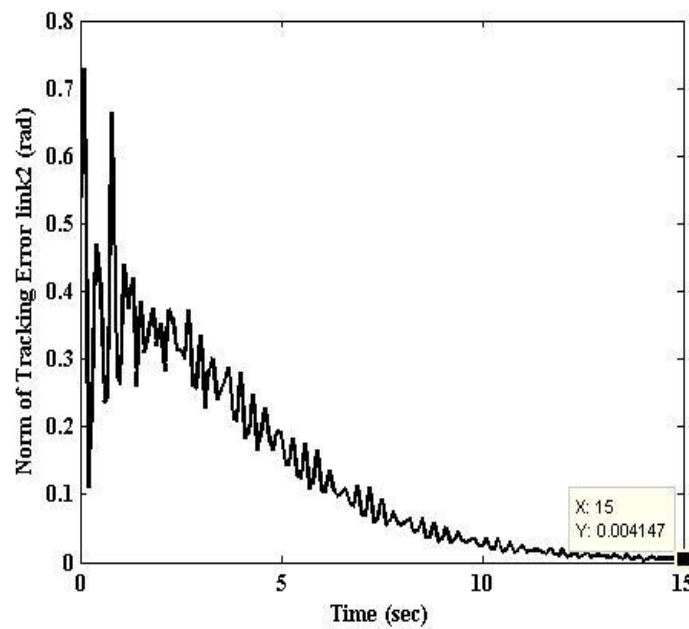


شکل (۴-۱۰) مسیر ردگیری رابط دو

نرم خطای ردگیری متغیرهای حالت رابط یک و دو در شکل (۴-۱۱) و (۴-۱۲) قابل مشاهده است.



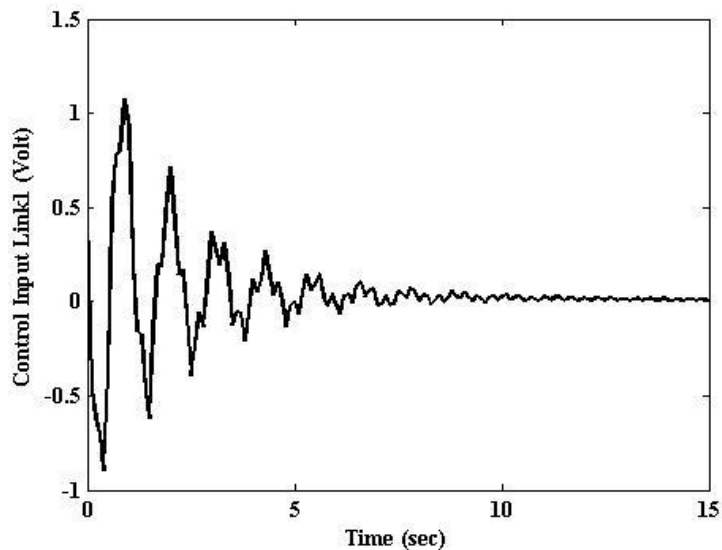
شکل (۴-۱۱) نرم خطای ردگیری رابط یک



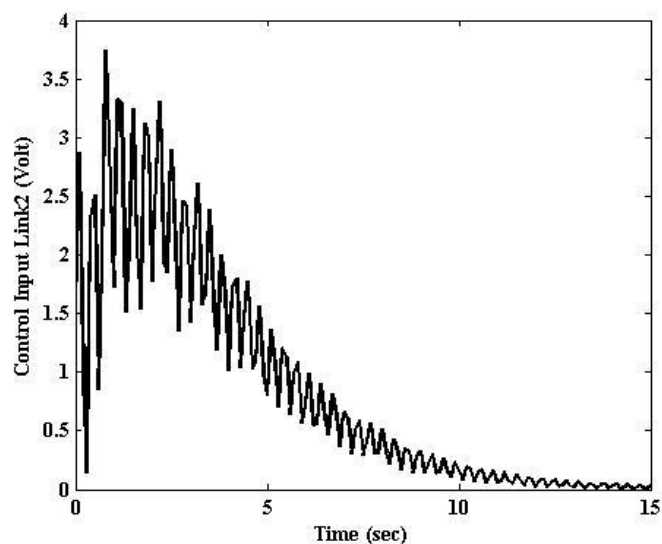
شکل (۴-۱۲) نرم خطای ردگیری رابط دو

همانطور که مشاهده می‌شود، نرم خطا در فاصله دور از نقطه تعادل دارای نوسان های بسیاری است. اما در نزدیکی نقطه تعادل این نوسان ها کاهش می‌یابد و نرم خطا بعد از ۱۵ ثانیه به مقدار 0.004 رادیان کاهش می‌یابد.

برای بررسی کاربردی بودن نتایج، سیگنال های کنترلی باید بدون لغزش و در محدوده قابل اعمال به سیستم باشند. بیشترین ولتاژ قابل اعمال به موتورهای به کار گرفته شده در این شبیه سازی 50 ولت می‌باشد. در شکل (۴-۶) و (۴-۷) ولتاژ اعمال شده به موتورها دیده می‌شود که در محدوده مجاز می‌باشند.

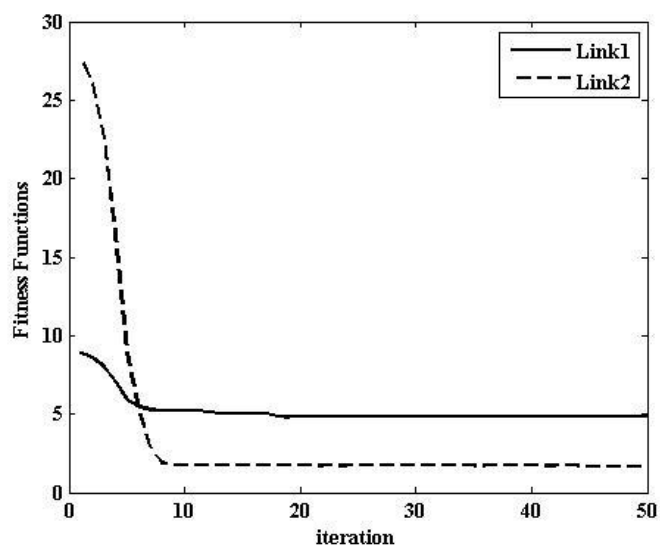


شکل (۴-۱۳) سیگنال کنترلی ورودی رابط یک



شکل (۴-۱۴) سیگنال کنترلی ورودی رابط دو

نتایج بدست آمده در این طراحی توسط الگوریتم بهینه سازی پرندگان بعد از ۵۰ بار تکرار و در نتیجه همگرایی پارامترها بدست آمده است. کمینه شدن تابع هدف برای دو رابط در شکل (۴-۸) مشاهده می شود. مقدار ضرایب c_1 و c_2 برابر ۱.۹ در نظر گرفته شده است و روند تغییر ضریب اینرسی خطی می باشد.



شکل (۴-۱۵) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان

۴-۶ خلاصه بحث و نتیجه گیری

در این فصل با ترکیب الگوریتم بهینه سازی پرندگان و تئوری کنترل مربعی خطی که پایه بسیاری از مباحث کنترلی است، روش کنترلی جدیدی ارائه شده است، که می‌تواند در هر سیستم خطی و غیر خطی در اطراف نقطه تعادل بدون پیشنهاد تابع منتخب لیاپانوف آن سیستم پایداری را تضمین نماید. این روش کنترلی در سیستم‌های غیرخطی با یک کنترل کننده خطی می‌تواند پاسخ مناسبی را برای سیستم فراهم کند. مزیت کنترل کننده خطی، پیاده سازی آسان آن نسبت به دیگر کنترل کننده‌ها می‌باشد. در سیستم‌های خطی این کنترل کننده بسیار کارآمد می‌باشد.

برای دستیابی به کنترل کننده مطلوب باید شرایطی را برای سیستم فراهم نمود. به دلیل خطی سازی در نزدیکی نقطه تعادل مسیر مطلوب همواره باید در اطراف نقطه تعادل باشد. همچنین برای تعمیم این روش به مسئله ردگیری باید نقطه تعادل سیستم صفر باشد و درنهایت برای رهایی از محاسبات پیچیده باید مسیر مطلوب نمایی باشد.

فصل پنجم

کنترل فازی بازوی ربات با مفاصل منعطف

۵-۱ مقدمه

منطق فازی برای اولین بار در سال ۱۹۶۵ توسط دکتر لطفی زاده [۲۵]، استاد علوم کامپیوتری دانشگاه برکلی کالیفرنیا ابداع شد. مقاله کلاسیک پرفسور لطفی زاده درباره مجموعه فازی که در سال ۱۹۶۵ به چاپ رسید، سرآغاز جهتی نوین در علوم و مهندسی سیستم و کامپیوتر بود. پس از آن پرفسور لطفی زاده به پژوهشهای خود در زمینه مجموعه فازی ادامه داد تا آنکه در سال ۱۹۷۳ طی یک مقاله کلاسیک دیگر تحت عنوان "شرحی بر دیدی نو در تجزیه و تحلیل سیستمهای پیچیده و فرایندهای تصمیم گیری" مفهوم استفاده از متغیرهای زبانی را در سیستم های حافظه و کنترل مطرح کرد. این مقاله اساس تکنولوژی کنترل بر مبنای منطق فازی است که در آینده اثرات عمیق در طراحی سیستم های کنترل هوشیار خواهد داشت. گرچه منطق فازی کاربردی خیلی وسیع تر از منطق متداول دارد. فازی یک فرا مجموعه از منطق بولی است که بر مفهوم درستی نسبی، دلالت می کند. منطق کلاسیک هر چیزی را بر اساس یک سیستم دوتائی نشان می دهد (درست یا غلط، ۰ یا ۱، سیاه یا سفید) ولی منطق فازی درستی هر چیزی را با یک عدد که مقدار آن بین صفر و یک است نشان می دهد. مثلاً اگر رنگ سیاه را عدد صفر و رنگ سفید را عدد ۱ نشان دهیم، آن گاه رنگ خاکستری عددی نزدیک به صفر خواهد بود. در سال ۱۹۶۵، دکتر لطفی زاده نظریه سیستم های فازی را معرفی کرد. در فضایی که دانشمندان علوم مهندسی به دنبال روش های ریاضی برای شکست دادن مسایل دشوارتر بودند، نظریه فازی به گونه ای دیگر از مدل سازی، اقدام کرد. در این فصل به پیاده سازی مرجع [۲۶] می پردازیم و ساختار یک کنترل کننده فازی را شرح می دهیم. سپس مزایا و معایب کنترل کننده ارائه شده را بررسی می کنیم و با اجرای الگوریتم بهینه سازی پرندگان برای بدست آوردن پاسخ بهینه برای کاهش خطای سیستم پارامترهای کنترل کننده را تنظیم می کنیم. پاسخ بهینه را با کنترل کننده پیشنهادی در این پایان نامه مقایسه می کنیم.

۵-۲ طراحی کنترل کننده فازی توسعه یافته

در این بخش به طراحی کنترل کننده فازی توسعه یافته می پردازیم. ابتدا طرح کلی کنترل کننده بیان می شود و سپس فازی ساز، غیر فازی ساز و قوانین کنترل کننده استفاده شده در این طراحی معرفی می شود. در نهایت شرط کافی برای غلبه بر عدم قطعیت ها و طراحی کنترل کننده مقاوم، توضیح داده می شود.

۵-۲-۱ کنترل کننده فازی PI

به طور خلاصه قواعد سیستم فازی PI را توضیح می دهیم. در این فصل فرض بر این است که خواننده آشنایی کامل با منطق فازی دارد. پس در ابتدا به معرفی نمای کلی کنترل کننده ارائه شده در مرجع [۲۶] می پردازیم تا بتوانیم محاسبات لازم برای توسعه کنترل کننده را ارائه دهیم.

کنترل کننده فازی PI نشان داده شده در شکل (۵-۱) یک کنترل کننده دیجیتال می باشد، که $T > 0$ زمان نمونه برداری، n شماره مرحله نمونه برداری و $sp(nT)$ موقعیت مطلوب می باشد. خروجی سیستم با $y(nT)$ نشان داده می شود. خروجی سیستم کنترلی به صورت کلی (خروجی قسمت خط چین در شکل (۵-۱)) $u(nT)$ می باشد. $\Delta u(nT)$ مقداری است که کنترل کننده فازی به خروجی قبلی، در هر مرحله اضافه می کند. K_p ، K_i ، K_u ضرایب ثابت تناسبی، انتگرالی و ضریب خروجی کنترل کننده فازی PI می باشد.

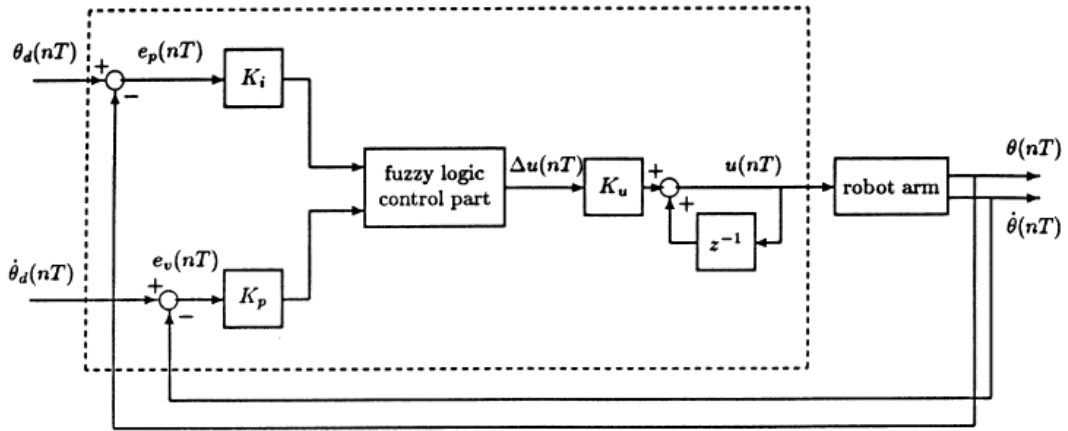
$$u(nT) = u(nT - T) + K_u \Delta u(nT) \quad (۵-۱)$$

ورودی سیستم کنترل خطای موقعیت موتور و خطای سرعت موتور است که به صورت زیر تعریف می شود.

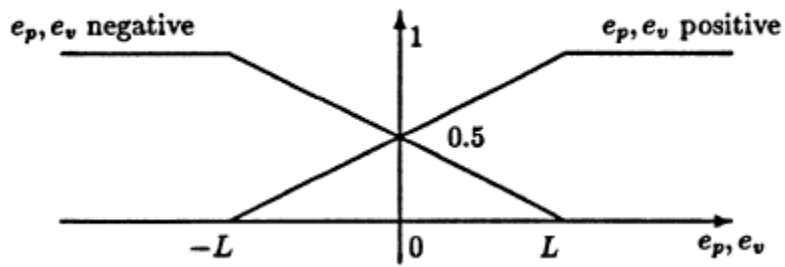
$$e_p(nT) = \theta_{dm}(nT) - \theta_m(nT) \quad (۵-۲)$$

$$e_v(nT) = \dot{\theta}_{dm}(nT) - \dot{\theta}_m(nT) \quad (۵-۳)$$

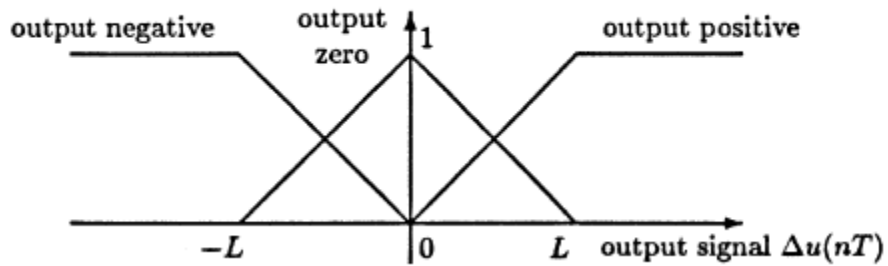
توابع تعلق ورودی های این کنترل کننده در شکل (۵-۲) و توابع تعلق خروجی در شکل (۵-۳) نشان داده شده است. ثابت $L > 0$ مقداری است که توسط طراح، برای نقطه شکست در توابع تعلق، پیشنهاد می شود و می تواند تاثیر مهمی بر روی عملکرد کنترل کننده داشته باشد.



شکل (۵-۱) بلوک دیاگرام سیستم فازی



شکل (۵-۲) توابع تعلق ورودی



شکل (۵-۳) توابع تعلق خروجی

۲-۲-۵ پایگاه استنتاج کنترل کننده فازی PI

با توجه به توابع تعلق ورودی و خروجی قوانین فازی به صورت زیر قابل تعریف است.

(R1) if $e_p = e_p.n$ AND $e_v = e_v.n$ Then $PI - output = o.n$.

(R2) if $e_p = e_p.n$ AND $e_v = e_v.p$ Then $PI - output = o.z$.

(R3) if $e_p = e_p.p$ AND $e_v = e_v.n$ Then $PI - output = o.z$.

(R4) if $e_p = e_p.p$ AND $e_v = e_v.p$ Then $PI - output = o.p$.

در اینجا "PI-output" خروجی کنترل کننده یا همان $\Delta u(nT)$ می باشد. " $e_p.p$ " به معنی تابع تعلق " e_p positive" و " $o.p$ " نشان دهنده تابع تعلق خروجی " $output$ positive" است. به همین ترتیب باقی ترکیب ها قابل تعریف می باشند. عملگر AND در اینجا طبق تعریف زاده می باشد. قوانین بالا را می توان به صورت زیر تفسیر نمود:

برای مثال در قانون یک ، منفی بودن خطای موقعیت یعنی زاویه موتور بالاتر از موقعیت مرجع می باشد. منفی بودن خطای سرعت موتور برای تنظیم نقطه ای نشان دهنده مثبت بودن سرعت موتور است. یعنی در مرحله قبل کنترل کننده با افزایش موقعیت موتور باعث شده از مقدار مرجع فراتر رود. در نتیجه در این مرحله با انتخاب تابع تعلق منفی برای خروجی کنترل کننده جهت چرخش موتور را مخالف وضعیت قبلی تنظیم می کنیم. تا موقعیت به مقدار مرجع نزدیک شود. ولی در قانون دوم با توجه به اینکه موقعیت از مقدار مرجع بیشتر است اما سرعت منفی موتور خود باعث کاهش زاویه موتور می شود، در نتیجه کنترل کننده احتیاجی به اعمال هیچ خروجی ندارد و با انتخاب تابع تعلق صفر برای خروجی این امکان فراهم می شود. به همین ترتیب قوانین دیگر قابل تفسیر هستند.

در بخش غیرفازی سازی کنترل کننده فازی به طور معمول از غیرفازی ساز میانگین مراکز استفاده می شود. بنابراین با اعمال غیر فازی ساز برای خروجی کنترل کننده داریم:

$$\Delta u(nT) = \frac{e_p.n \times o.n + e_p.n \times o.z + e_v.n \times o.z + e_v.p \times o.p}{e_p.n + e_p.n + e_v.n + e_v.p} \quad (۴-۵)$$

۵-۲-۳ کنترل کننده فازی PI اصلاح شده

در مرجع [۲۶] با استفاده از معادلات سیستم ربات با مفاصل منعطف برای غلبه بر عدم قطعیت ها کنترل کننده ارائه شده در بخش قبل گسترش داده شده است. معادلات حاکم بر سیستم ربات یک محوره در مرجع [۲۶] به صورت زیر ساده سازی شده است:

$$I \ddot{\theta} + mgl \sin(\theta) - k(r\theta_m - \theta) = 0 \quad (۵-۵)$$

$$0 = Y.P - k(r\theta_m - \theta) \quad (۶-۵)$$

که در آن

$$Y = [\ddot{\theta} \sin(\theta)] \quad P = [I \ mgl] \quad (۷-۵)$$

و با استفاده از آن برای محاسبه مقدار مرجع برای موقعیت و سرعت موتور داریم:

$$\theta_{md} = K^{-1}.Y_d.P + \theta_d \quad \dot{\theta}_{md} = K^{-1}.\dot{Y}_d.P + \dot{\theta}_d \quad (۸-۵)$$

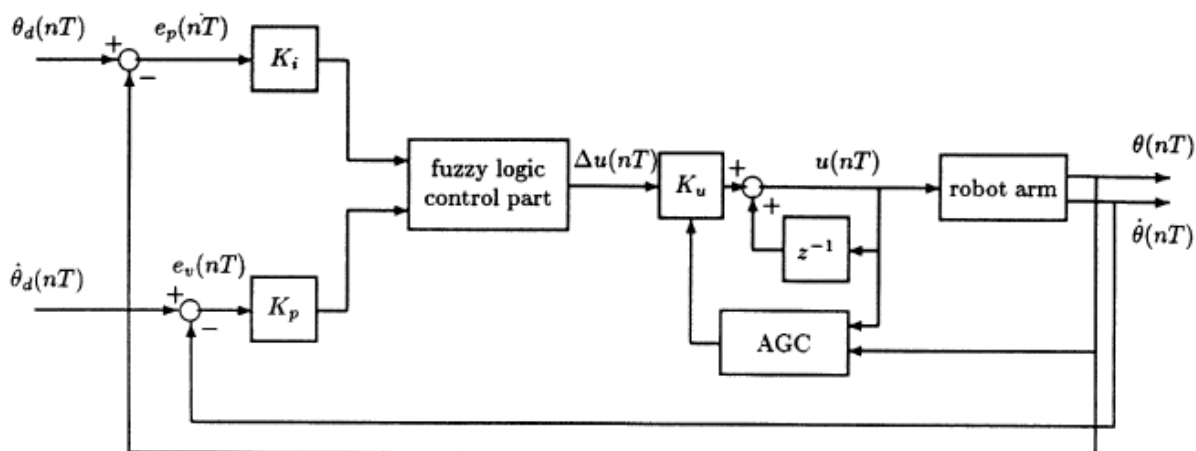
در معادله (۶-۲) با صرف نظر نمودن از مولفه سرعت موتور و انتگرال گیری از آن داریم:

$$J_m \int_0^T \ddot{\theta}_m dt + rK \int_0^T (r\theta_m - \theta) dt = \int_0^T \tau dt$$

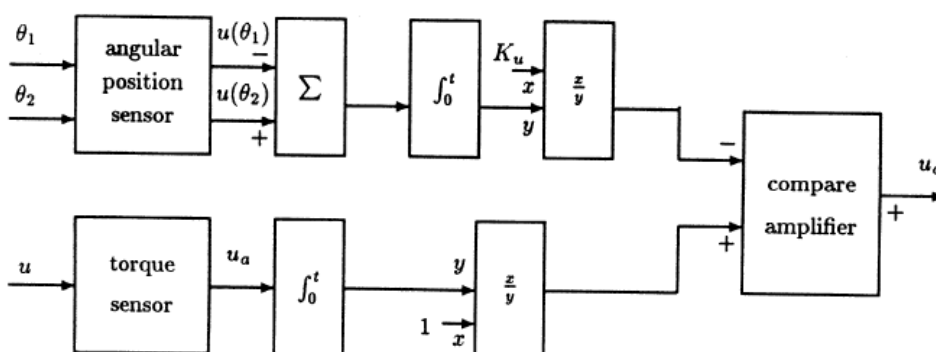
در نتیجه داریم:

$$\dot{\theta}_m = \frac{1}{J_m} \left(\int_0^T \tau dt - rK \int_0^T (r\theta_m - \theta) dt \right) \quad (۹-۵)$$

در نتیجه اگر $\int_0^T \tau dt > rK \int_0^T (r\theta_m - \theta) dt$ باشد $\dot{\theta}_m > 0$ است. پس باید ورودی سیستم را افزایش داد و برعکس اگر $\int_0^T \tau dt < rK \int_0^T (r\theta_m - \theta) dt$ سرعت موتور منفی می باشد، در نتیجه باید خروجی کنترل کننده کاهش پیدا کند. تغییر خروجی کنترل کننده با استفاده از معادله کمکی فوق بوسیله تغییر خودکار ضریب خروجی کنترل کننده ممکن می شود. این تغییر در بلوک AGC انجام می شود.



شکل (۵-۴) بلوک دیاگرام کنترل کننده فازی توسعه یافته



شکل (۵-۵) بلوک AGC

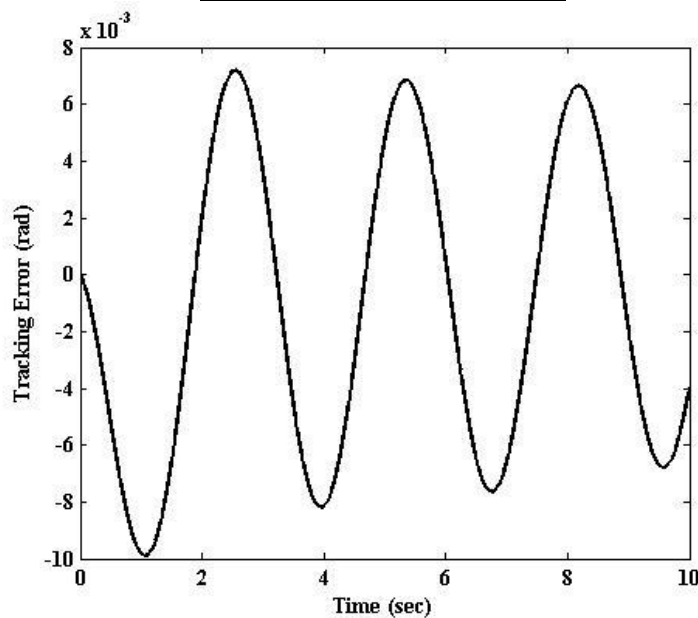
۵-۳ شبیه سازی

در مرجع [۲۶] کنترل کننده پیشنهادی روی ربات یک محوره پیاده سازی شده است. برای دستیابی به نتایج بهتر از کنترل کننده ارائه شده الگوریتم بهینه سازی پرندگان به صورت غیر متصل بر روی کنترل کننده اجرا شده است. مشخصات موتور مانند فصل چهارم می باشد. پارامترهای تنظیم شده توسط الگوریتم PSO ضرایب کنترل کننده PI و مقدار K_u می باشد. که در نهایت با مقدار $L=5$ پارامترهای جدول (۵-۱) بدست آمد. در شکل (۵-۶) خطای ردگیری قابل مشاهده است. به علت ساده سازی های انجام شده و مسئله انعطاف در

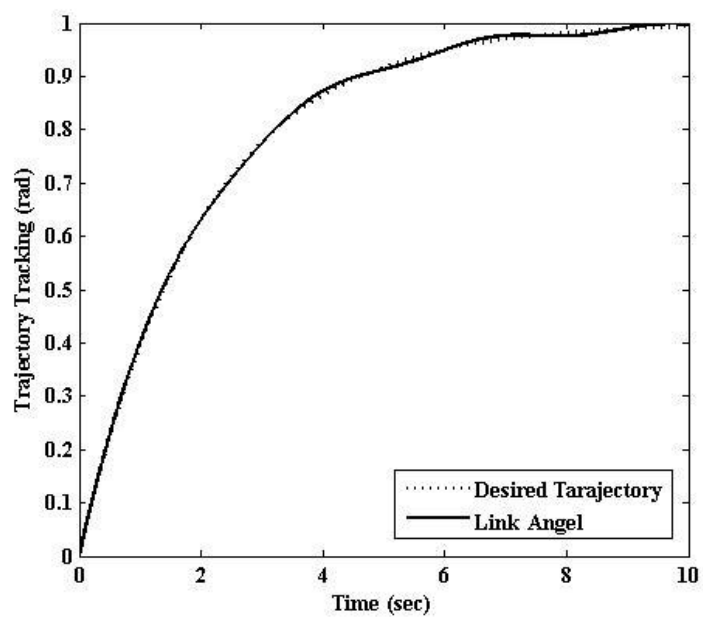
مفاصل در رفتار سیستم شاهد نوسان هستیم. در شکل (۵-۷) مسیر ردگیری نشان داده شده است، همانطور که دیده می‌شود کنترل کننده به خوبی توانسته است مسیر را ردگیری کند و خطا به کمتر از 0.007 رسیده است. ۳۲ گروه از پرندگان در الگوریتم بهینه سازی پرندگان در نظر گرفته شده است. الگوریتم بهینه سازی پرندگان ۵ بار تکرار شده است و در هر دوره ۵۰ دوره تکرار به صورت خارج خط انجام شده است که بهترین نتیجه بدست آمده ارائه شده است. روند کاهش تابع هزینه در طول الگوریتم در شکل (۵-۸) قابل مشاهده است. برای اطمینان از قابلیت پیاده سازی عملی کنترل کننده در شکل (۵-۹) سیگنال ورودی را بررسی شده است. سیگنال کنترلی در این شبیه سازی گشتاور موتورها است. نوسانات موجود می‌تواند به علت وجود انعطاف در مفاصل ایجاد شده باشد.

جدول (۵-۱) پارامترهای تنظیم شده توسط PSO

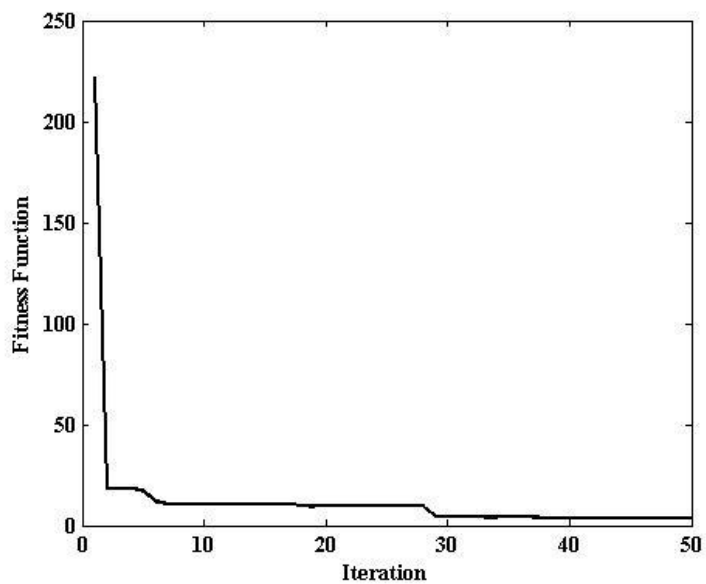
K_u	K_I	K_p
۰.۲۱	۸.۲	۹.۱۲



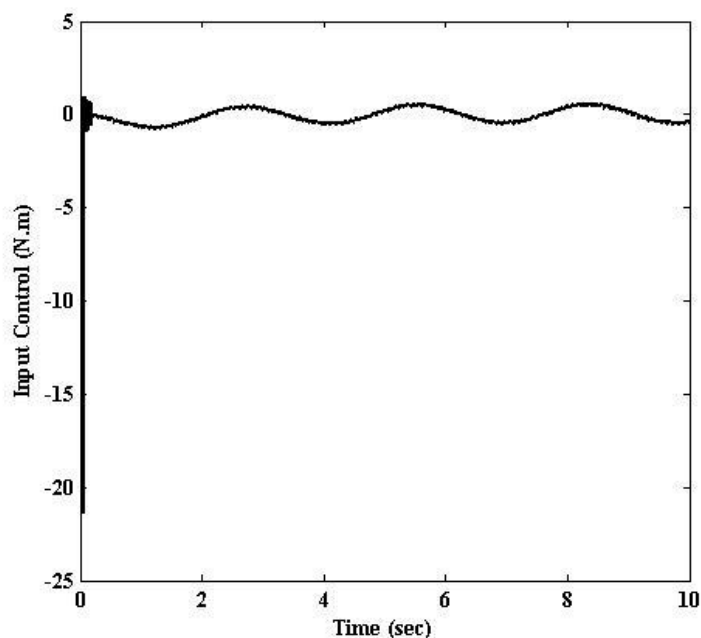
شکل (۵-۶) خطای ردگیری رابط



شکل (۵-۷) مسیر ردگیری رابط



شکل (۵-۸) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان



شکل (۵-۹) سیگنال کنترلی ورودی

۴-۵ خلاصه بحث و نتیجه گیری

در این طرح کنترلی به علت استفاده از تعداد قوانین کم محاسبات کنترلی کاهش یافته است. همچنین نیاز به اندازه‌گیری موقعیت محور و سرعت آن و یا اندازه‌گیری شتاب موتور نمی باشد و تنها با دو حالت موقعیت و سرعت موتور کار کنترل انجام می‌شود.

در اکثر موارد از منطق فازی در مواردی که مدل دقیق سیستم در اختیار نیست، استفاده می‌شود. اما در این طراحی برای بدست آوردن مقدار مرجع برای موقعیت و سرعت موتور از مدل سیستم استفاده شده است. همچنین برای بدست آوردن معادله کمکی برای توسعه سیستم کنترل فازی PI استفاده شده است. هرچند در انتهای مقاله اثبات شده است که سیستم بر عدم قطعیت پارامتری تا ۱۰٪ مقدار نامی برای پارامترهای مدل مقاوم می باشد.

همچنین تنظیم پارامترهای کنترل کننده مانند L و چگونگی تغییر ضریب خروجی در بلوک AGC توسط طراح صورت می‌گیرد، که این امر در عملکرد سیستم بسیار موثر است. برای دستیابی به عملکرد مطلوب، با پیاده سازی

کنترل کننده فازی PI اصلاح شده، پارامترهای کنترل کننده را توسط الگوریتم بهینه سازی پرندگان تعیین شد تا پاسخ بهینه برای سیستم فراهم شود.

فصل ششم

کنترل فازی بهینه بازوی ربات با مفاصل منعطف

با استفاده از الگوریتم بهینه سازی پرندگان

۶-۱ مقدمه

طراحی کنترل کننده فازی با تضمین پایداری برای سیستم‌های پیچیده دشوار است. زیرا سیستم حلقه بسته دارای معادلات بسیار پیچیده مشتمل بر کنترل کننده فازی و سیستم خواهد بود. در نتیجه، یافتن تابع لیاپانوف و تحلیل پایداری بسیار دشوار خواهد بود.

در مورد کنترل سیستم‌های غیرخطی روش‌های مختلفی پیشنهاد شده است که بتواند پایداری سیستم را تضمین کند. از آن دسته می‌توان به خطی سازی فیدبکی و تضمین پایداری سیستم با روش لیاپانوف اشاره کرد. آنگاه که مدل سیستم پیچیده باشد استفاده از روش خطی سازی فیدبکی منجر به قانون کنترلی با محاسبات زیاد و پیچیده می‌شود [۷۰]. روش مستقیم لیاپانوف هم به علت پیچیدگی سیستم دشوار می‌شود. زیرا، ممکن است نتوان تابع لیاپانوف سیستم را یافت تا بر اساس آن پایداری را تضمین نمود.

در این میان کنترل فازی به علت توانایی ارائه راه حلی موثر برای کنترل سیستم‌های پیچیده و دچار عدم قطعیت بسیار مورد توجه قرار گرفته است. دو ویژگی کنترل کننده فازی که باعث گرایش‌های زیاد به آن شده است ۱- استفاده از دانش فرد خبره ۲- عدم وابستگی به مدل سیستم می‌باشد.

با وجود مزیت‌های فوق تحلیل پایداری کنترل فازی آسان نیست و تنظیم پارامترها معمولاً فرایند زمان‌بری برای سیستم‌های غیرخطی و چند متغیره می‌باشد. همچنین کنترل فازی به علت استفاده از دانش فرد خبره پارامترهای زیادی برای تنظیم دارد که انتخاب صحیح پارامترها به نزدیک شدن سیستم به شرایط مطلوب کمک می‌کند. در این زمینه، بعد از طراحی کنترل فازی و تحلیل پایداری آن و قرار دادن پارامترها در محدوده پایداری از الگوریتم‌های بهینه سازی برای تعیین پارامترها برای قرارگرفتن سیستم در بهترین عملکرد ممکن استفاده می‌شود [۷۱].

اگر بتوان روشی را پیشنهاد داد که تنظیم پارامترهای کنترل کننده در راستای رسیدن به پایداری و عملکرد بهینه باشد بسیار مفید خواهد بود. در این طرح، تابع هزینه ای که الگوریتم بهینه سازی پرندگان سعی در کمینه کردن آن دارد، به گونه ای پیشنهاد می‌شود که پایداری و عملکرد مطلوب محقق شود. با این روش، برای تحقق پایداری نیاز به مدل سیستم نیست. روش پیشنهادی برای طراحی کنترل کننده فازی ربات با مفاصل منعطف ارائه خواهد شد.

در فصل چهارم روند گزینش پارامترها به گونه ای بود که در ضمن کمینه شدن تابع هزینه، پایداری برمیانی معیار لیاپانوف تضمین شود. برای این امر ناگزیر سیستم را در نزدیکی نقطه تعادل خطی فرض می کردیم، که این فرض باعث محدودیت هایی در طراحی مسیر مطلوب می شود. در این فصل معیار دیگری را برای تضمین پایداری در نظر گرفته ایم تا بر محدودیت های گذشته غلبه کنیم. این معیار از مفاهیم و اصول پایداری نتیجه شده است. بنابراین، برای توضیح روش پیشنهادی ابتدا مفاهیم پایداری به طور مختصر بیان می شود و سپس در بخش (۳-۶) کنترل کننده پیشنهادی معرفی می شود.

۶-۲ مفاهیم پایداری [۶۰]

نقطه تعادل x_e را به مفهوم لیاپانوف پایدار گویند، اگر برای هر t_0 و هر $\varepsilon > 0$ یک $\rho > 0$ وجود دارد که به ازای $\|x(t_0) - x_e\| < \varepsilon$ برای کلیه $t \geq t_0$ ، $\|x(t) - x_e\| \leq \rho$ باشد.

یعنی می توان با انتخاب حالت اولیه به اندازه کافی نزدیک به نقطه تعادل از دور شدن حالت از نقطه تعادل جلوگیری کرد. به عبارت دیگر حالت تعادل x_e را به مفهوم لیاپانوف پایدار گویند اگر پاسخ سیستم ناشی از حالت اولیه که نزدیک به نقطه تعادل باشد، هیچگاه از آن دور نخواهد شد.

پایداری لیاپانوفی صورت ضعیف پایداری است صورت قوی تر و عملی تر زمانی خواهد بود که پاسخ علاوه بر نزدیک ماندن به حالت تعادل، به سمت آن نیز میل کند.

نقطه تعادل پایدار مجانبی است اگر:

الف) پایدار لیاپانوفی باشد.

ب) برای هر t_0 یک $\rho(t_0) > 0$ وجود دارد که به ازای $\|x(t_0) - x_e\| < \rho$ ، برای هر $t \rightarrow \infty$ رابطه زیر برقرار است:

$$\|x(t) - x_e\| \rightarrow 0$$

بنابراین، پایداری مجانبی، علاوه بر پایداری لیاپانوفی، همگرایی سیستم به نقطه تعادل را نیز تضمین می کند.

نقطه تعادل پایدار مجانبی سراسری است اگر :

الف) پایدار به مفهوم لیاپانوف باشد.

ب) برای هر $x(t_0)$ و هر t_0 ، به ازای $t \rightarrow \infty$ داشته باشیم:

$$\|x(t) - x_e\| \rightarrow 0$$

یعنی نقطه تعادل پایدار مجانبی سراسری این خاصیت را دارد که کلیه پاسخ های دیگر نهایتاً به سمت آن میل خواهد کرد.

نقطه تعادل سیستم را ناپایدار گویند اگر به مفهوم لیاپانوف یا پایدار مجانبی نباشد. پایداری لیاپانوف و مجانبی را پایدار داخلی نیز می نامند. زیرا این تعاریف با ورودی سیستم کاری ندارند و رفتار متغیرهای حالت سیستم را توصیف می کنند. (پاسخ ورودی صفر) تعریف زیر در رابطه با رفتار ورودی خروجی سیستم است.

سیستم بدون شرایط اولیه را پایدار ورودی کران دار - خروجی کران دار^۱ گویند، اگر و فقط اگر برای هر ورودی کراندار، خروجی کراندار باشد. این تعریف پایداری حالت صفر سیستم می باشد.

یک سیستم را کاملاً پایدار یا پایدار T می نامند، اگر و فقط اگر برای هر شرایط اولیه ی حالت و هر ورودی کراندار، خروجی و کلیه متغیرهای حالت سیستم کراندار باشند.

شرط پایداری T از سایر پایداری های ارائه شده سخت تر است. در بعضی مواقع سیستمی که فقط پایدار ورودی محدود - خروجی محدود است، به خوبی عمل نمی کند، زیرا ممکن است برخی متغیرهای حالت آن ناپایدار باشد و باعث از کارافتادگی یا اشباع سیستم شوند. بنابراین، در عمل سیستم باید پایدار T باشد.

۶-۳ کنترل کننده پایدار با استفاده از الگوریتم بهینه سازی پرندگان

با توجه به تعاریف بخش قبل اگر متغیرهای حالت بعد از شروع از هر مقدار اولیه، به سمت نقطه تعادل حرکت کند پایدار مجانبی دارد. یعنی مقدار خطای نرم متغیرهای حالت در هر لحظه به سمت صفر میل کند. به عبارت دیگر با افزایش زمان در هر لحظه مقدار نرم متغیرهای حالت کاهش می یابد. حال اگر در طی اجرای الگوریتم در هر بار جایگزین پارامترها در سیستم، در تمام طول شبیه سازی نرم متغیرهای حالت را رصد شود تا مقدار آن از مقدار قبلی کمتر باشد و براساس این ویژگی و کاهش تابع هدف، پارامترها گزینش شوند. برای این منظور به معرفی تابع هزینه می پردازیم و سپس معیار پایداری را بیان می کنیم. سیستم زیر را در نظر بگیرید:

^۱ Bounded Input-Bounded Output

$$\dot{X} = f(X, u) \quad (1-6)$$

که در آن u ورودی سیستم و X بردار متغیرهای حالت سیستم می‌باشد. همچنین در این روش الگوریتم بهینه سازی سعی در کمینه کردن تابع J دارد.

$$J = \int_{t=0}^{\infty} (e^T Q e + u^T R u) \quad (2-6)$$

که در آن R و Q ماتریس های مثبت معینی هستند که توسط طراح انتخاب می شوند و e از رابطه زیر تعریف می‌شود.

$$e = X_d - X \quad (3-6)$$

انتخاب این تابع هدف همزمان با کمینه کردن بردار خطا سعی در کمینه کردن انرژی نیز دارد. انتخاب پارامترها توسط الگوریتم بهینه سازی باید به گونه ای باشد که پایداری سیستم تضمین شود. بنابراین، نرم خطای متغیرهای حالت طبق معادله (۶-۶) باید در هر لحظه از مقدار قبلی کمتر باشد. در غیر اینصورت کنار گذاشته می‌شوند.

$$\|e_{k+1}\| < \|e_k\| \quad (4-6)$$

یعنی در هر بازه زمانی خطا رو به کاهش باشد و با فرکانس نمونه برداری مشخص هر نرم، در هر نمونه زمانی از نرم بردار خطا از نمونه قبل خود کوچکتر باشد. فرکانس نمونه برداری بسیار مهم است. چراکه در بعضی مواقع ممکن است، اغتشاشات در یک نمونه زمانی باعث افزایش نرم خطا شوند اما در کل روند نرم خطای متغیرهای حالت رو به کاهش است. بدین ترتیب پایداری مجانبی سیستم تضمین می‌شود.

حال در حضور ورودی باید پایداری ورودی محدود - خروجی محدود نیز تضمین شود، تا در نهایت پایداری T فراهم شود. برای این منظور در کنار معیار معادله (۴-۶) باید کراندار بودن خروجی نیز در هر لحظه چک شود.

برای پیاده سازی روش کنترلی پیشنهادی در ربات دومحوره با مفاصل منعطف دو طرح کنترلی پیشنهاد شده است. ابتدا طرح کنترلی را بر روی کنترل کننده خطی ارائه می‌شود و سپس با معرفی کنترل کننده فازی پارامترهای این کنترل کننده با استفاده از روش پیشنهادی تنظیم می‌شود.

۴-۶ کنترل خطی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان

در این بخش تنها با استفاده از مقدار موقعیت موتور، انتگرال آن و سرعت موتور کنترل کننده خطی PID طراحی می کنیم. ضرایب آن با استفاده از طرح پیشنهادی طوری تنظیم می شود که پایداری T تضمین می شود. در سیستم ربات، مقدار مطلوب رابط ها در دسترس می باشند. برای بدست آوردن مقدار مطلوب موقعیت موتور، انتگرال آن و سرعت موتور با توجه به رابطه (۲-۵) می توان نوشت:

$$\theta_{md} = Kr^{-1}[D(\theta_d)\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d)\dot{\theta}_d + g(\theta_d) + K\theta_d] \quad (۵-۶)$$

یعنی مقدار مطلوب موقعیت موتور را با استفاده از مقادیر مطلوب رابط بدست می آوریم و با انتگرال گیری و مشتق گیری از آن، مقادیر ورودی متناسب با کنترل کننده طراحی شده را می یابیم.

اما استفاده از معادله (۵-۶) در صورتی ممکن است که مدل سیستم در دسترس باشد، حتی با وجود مدل، با توجه به تزویج سنگین ربات و پیچیدگی معادلات، استفاده از آن دشوار است. بنابراین، معادله (۵-۶) با استفاده از الگوریتم بهینه سازی پرندگان با یک معادله خطی تخمین زده می شود و پارامترهای معادله (۶-۶) همزمان با پارامترهای کنترل کننده خطی تنظیم می شود.

$$\begin{bmatrix} \theta_{m1} \\ \theta_{m2} \end{bmatrix} = \begin{bmatrix} K_{p11} & K_{p12} \\ K_{p21} & K_{p22} \end{bmatrix} \begin{bmatrix} \theta_{d1} \\ \theta_{d2} \end{bmatrix} + \begin{bmatrix} K_{d11} & K_{d12} \\ K_{d21} & K_{d22} \end{bmatrix} \begin{bmatrix} \dot{\theta}_{d1} \\ \dot{\theta}_{d2} \end{bmatrix} + \begin{bmatrix} K_{a11} & K_{a12} \\ K_{a21} & K_{a22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_{d1} \\ \ddot{\theta}_{d2} \end{bmatrix} \quad (۶-۶)$$

در هر لحظه مقدار نرم خطای موقعیت موتور و سرعت آن رصد می شود، تا از مقدار قبلی کمتر باشد. با توجه به معادله (۲-۵) می توان اثبات نمود، که اگر موقعیت و سرعت موتور به سمت مقدار مطلوب میل کنند، خطای متغیرهای حالت رابط نیز به سمت صفر میل خواهند کرد و در نتیجه پایداری مجانبی تمامی متغیرهای حالت تضمین می شود.

همزمان با کنترل مقدار نرم خطا در هر نمونه زمانی، خروجی سیستم نیز باید محدود باقی بماند. در نتیجه باید مقادیر موقعیت و سرعت رابط و موتور محدود باشند، که با توجه به استدلال فوق تنها در نظر گرفتن متغیرهای حالت موتور می تواند محدود بودن تمام خروجی ها را تضمین کند. البته تنها می توان موقعیت رابط را به عنوان خروجی در نظر گرفت که در نتیجه با کنترل موقعیت موتور در یک بازه محدود، محدود بودن موقعیت رابط نیز تضمین می شود.

۵-۶ کنترل فازی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه‌سازی پرندگان

در ابتدا ساختار فازی به کارگرفته شده در این روش پیشنهادی معرفی می‌شود. از کنترل کننده تاگاکای سوگنو در این بخش بهره گرفته شده است. قوانین کنترل در روش کنترل فازی تاگاکای سوگنو به صورت زیر می باشد :

$$Ru^l : \text{If } x_1 \text{ is } A_1^l \text{ and } x_2 \text{ is } A_2^l \text{ Then } y^l = a_0^l \quad (۷-۶)$$

که در آن Ru^l بیان کننده قوانین فازی برای $l=1, \dots, 9$ می باشد ورودی قانون l ام x_1 و x_2 می باشد و A_i^l مجموعه فازی، a_0^l ضریب ثابت می‌باشد.

در این کنترل کننده، x_1 و x_2 به ترتیب خطای موقعیت موتور و خطای سرعت موتور می‌باشد. که مقدار موقعیت مطلوب و سرعت آن از رابطه (۶-۶) بدست می‌آید. تابع $f(x)$ به صورت محاسبه می‌شود:

$$f(x) = \frac{\sum_{l=1}^9 \prod_{i=1}^2 \mu_{A_i^l}(x_i) y^l}{\sum_{l=1}^9 \prod_{i=1}^2 \mu_{A_i^l}(x_i)} \quad (۸-۶)$$

که $\mu_{A_i^l}(x_i) \in [0, 1]$ توابع عضویت می باشند با قرار دادن y^l از معادله (۷-۶) در معادله (۸-۶) می توان به معادله ساده شده زیر رسید.

$$f(x) = c_0(x) \quad (۹-۶)$$

که در آن $c_0(x)$ از رابطه زیر بدست می آید.

$$c_j(x) = \frac{\sum_{l=1}^9 \prod_{i=1}^2 \mu_{A_i^l}(x_i) a_j^l}{\sum_{l=1}^9 \prod_{i=1}^2 \mu_{A_i^l}(x_i)} \quad (۱۰-۶)$$

می توانیم قوانین فازی را بصورت زیر طراحی کنیم:

Rule ۱ If x_1 is P and x_2 is P Then $y = K_1$

Rule ۲ If x_1 is P and x_2 is Z Then $y = K_2$

Rule ۳ If x_1 is P and x_2 is N Then $y = K_3$

Rule ۴ If x_1 is Z and x_2 is P Then $y = K_4$

Rule ۵ If x_1 is Z and x_2 is Z Then $y = K_5$

Rule ۶ If x_1 is Z and x_2 is N Then $y = K_6$

Rule ۷ If x_1 is N and x_2 is P Then $y = K_7$

Rule ۸ If x_1 is N and x_2 is Z Then $y = K_8$

Rule ۹ If x_1 is N and x_2 is N Then $y = K_9$

همانطور که مشاهده می شود در این قوانین پارامترهای قابل تنظیمی وجود دارد، که در الگوریتم بهینه سازی پرندگان با سه معیار پایداری مجانبی، پایداری ورودی محدود- خروجی محدود و کمینه شدن تابع هدف در آن دسته پارامتر تنظیم می شوند.

۶-۶ شبیه سازی

۶-۶-۱ پیاده سازی کنترل خطی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم

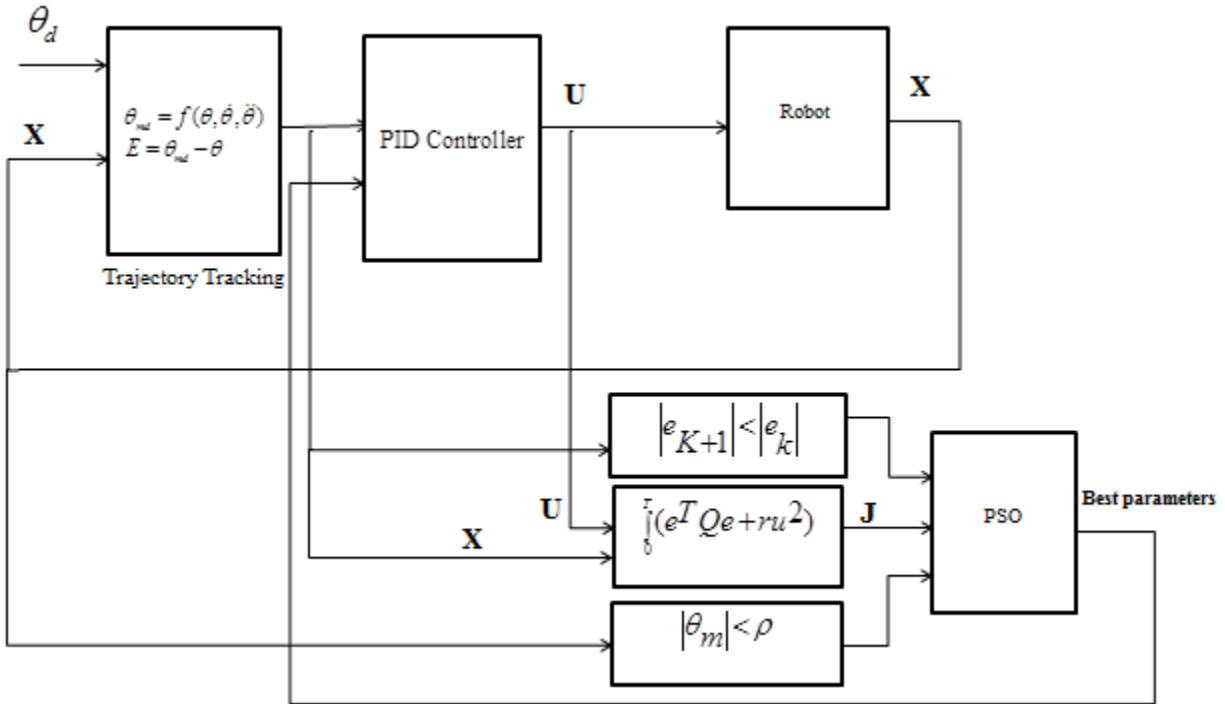
بهینه سازی پرندگان

در این بخش روش کنترلی پیشنهادی در بخش (۴-۶) بر روی ربات با مشخصات ارائه شده در جدول (۴-۱) و (۴-۲) پیاده سازی شده است. بلوک دیاگرام این کنترل کننده در شکل (۶-۱) قابل مشاهده است. در بلوک trajectory tracking با استفاده از ضرایبی که توسط PSO تنظیم می شود مسیر مطلوب موتورها بدست می آید. و سپس به عنوان ورودی به کنترل کننده PID داده می شود. در این بلوک با استفاده از مقدار موقعیت موتور، انتگرال آن و سرعت موتور در هر لحظه، سیگنال ورودی به سیستم فراهم می شود. ضرایب این بلوک نیز توسط PSO با در نظر گرفتن سه معیار ارائه شده در بخش (۴-۶) تعیین می شود.

این سه معیار را می توان به صورت زیر بیان نمود:

- ۱- کمینه شدن تابع هدف به ازای پارامتر برتر
- ۲- کاهش نرم خطا در هر نمونه زمانی نسبت به نمونه قبلی آن در طول شبیه سازی
- ۳- محدود بودن مقدار موقعیت موتور در طول شبیه سازی

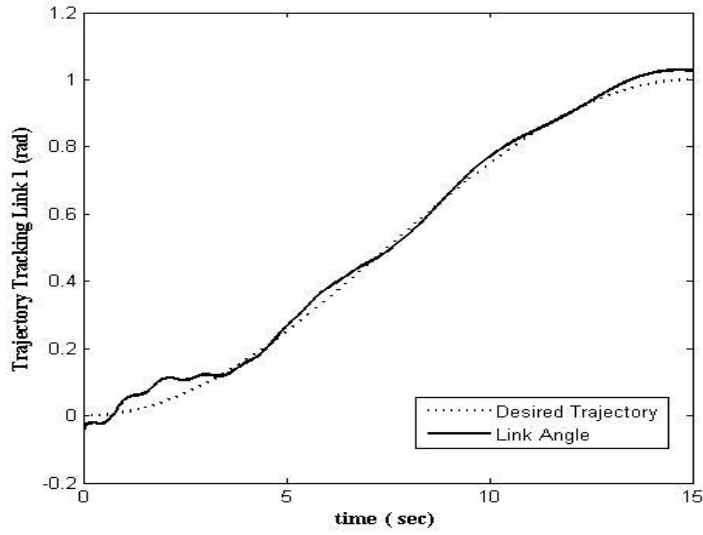
در بلوک تابع هدف، این سه معیار در هر بار تکرار الگوریتم بهینه سازی پرندگان رصد می شود و در نهایت بهترین دسته از پرندگان برای ضرایب کنترل کننده انتخاب می شود.



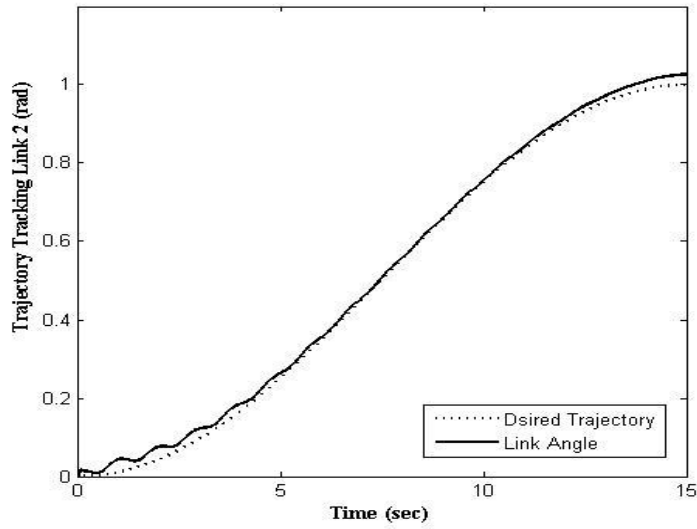
شکل (۶-۱) بلوک دیاگرام کنترل کننده خطی ربات با مفاصل منعطف

مسیر طراحی شده برای حرکت دو رابط در فضای مفصلی در معادله (۶-۱۴) بیان شده است. همانطور که در شکل (۶-۲) و (۶-۳) مشاهده می‌شود با وجود آنکه مسیر طراحی شده از نقاط تعادل فاصله دارند، اما کنترل کننده مسیر را به خوبی ردگیری کرده است.

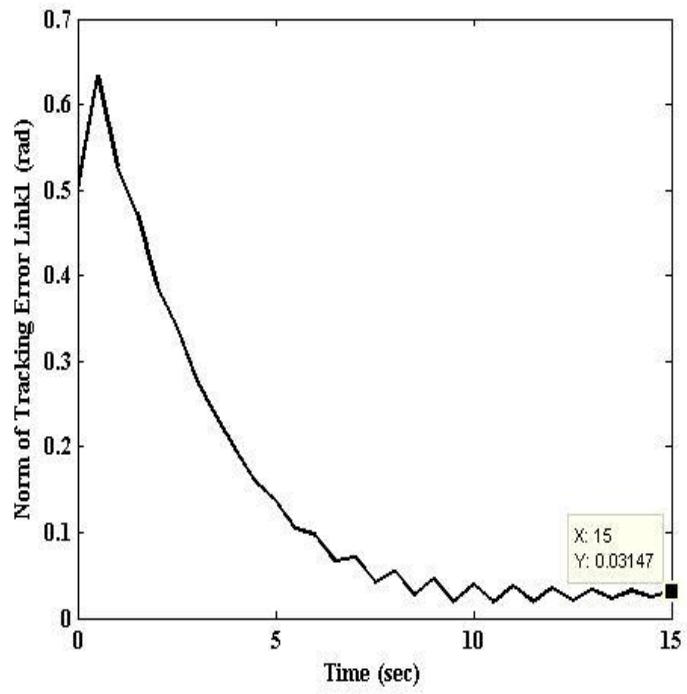
$$\begin{cases} \theta_{d1} = 0.5(1 - \cos(\frac{\pi t}{15})) \\ \theta_{d2} = 0.5(1 - \cos(\frac{\pi t}{15})) \end{cases} \quad (۶-۱۴)$$



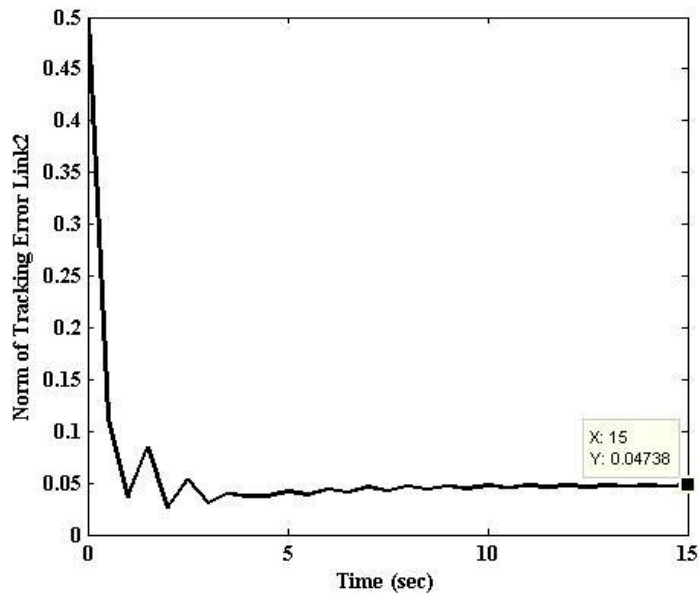
شکل (۶-۲) مسیر ردگیری رابط یک



شکل (۶-۳) مسیر ردگیری رابط دو



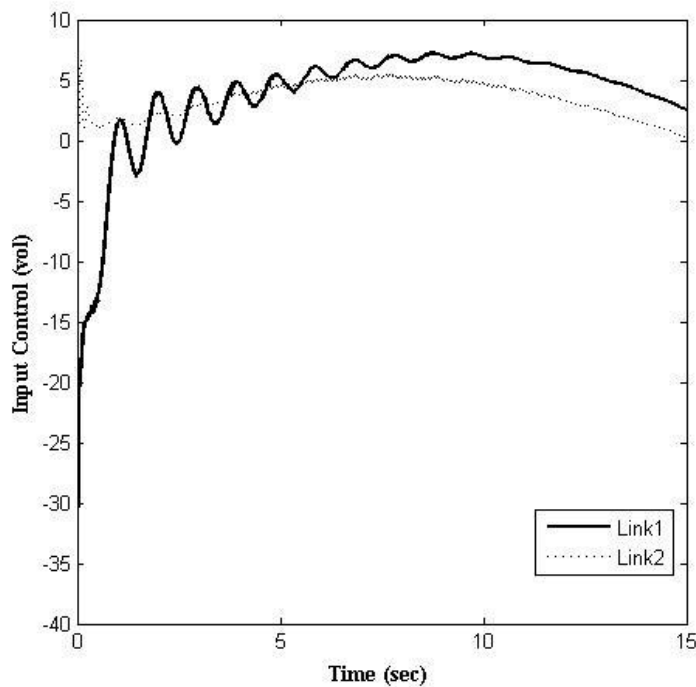
شکل (۴-۶) نرم خطای ردگیری رابط یک



شکل (۵-۶) نرم خطای ردگیری رابط دو

شکل های (۴-۶) و (۵-۶) نرم خطای ردگیری دو رابط را نشان می دهند. همانطور که مشاهده می شود در بعضی از نمونه های زمانی نرم خطا نسبت به نمونه زمانی قبلی افزایش یافته است که می تواند ناشی از عواملی چون تاثیر حرکت رابط دیگر بر حرکت رابط باشد اما در کل این روند ر به کاهش است. برای یافتن پارامتر مناسب در الگوریتم PSO پارامتری برگزیده می شود که این افزایش نرم خطا در آن کمتر اتفاق بیافتد و دو معیار دیگر در آن پارامتر فراهم شده باشد.

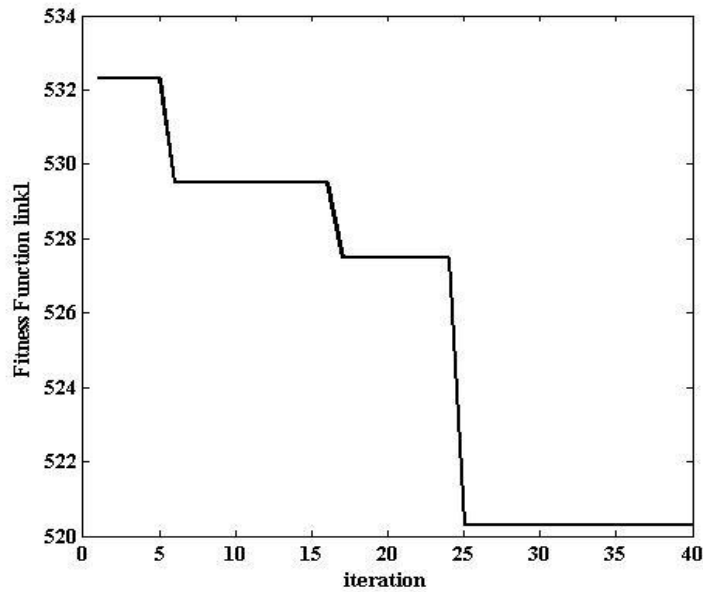
در شکل (۶-۶) سیگنال کنترلی ورودی دو رابط دیده می شود، که هر دو در محدوده معین هستند و دچار لغزش سیگنال نیستند. که این امر کارآمد بودن کنترل کننده را نشان می دهد.



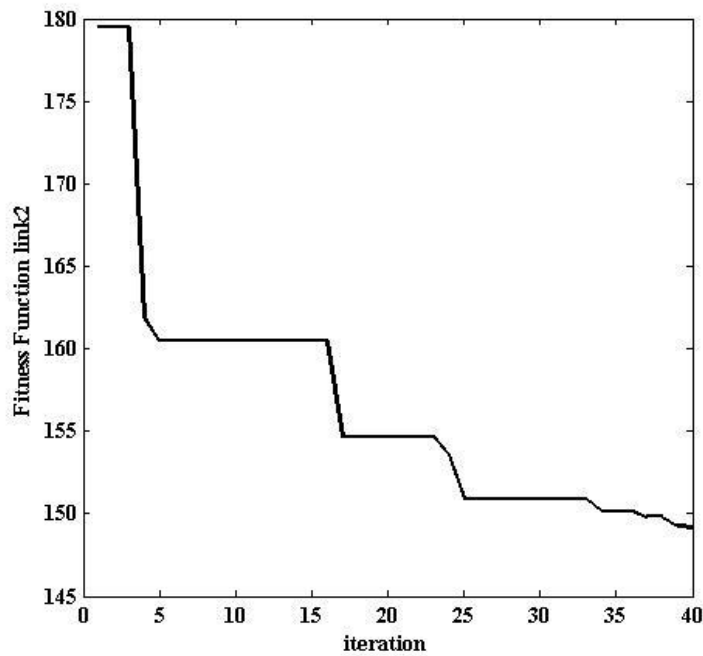
شکل (۶-۶) سیگنال کنترلی ورودی دو رابط

پارامترهای بدست آمده با استفاده از الگوریتم بهینه سازی پرندگان بعد از ۵ بار تکرار و در هر دوره ۴۰ بار تکرار به صورت خارج خط انجام شده است که بهترین نتیجه بدست آمده ارائه شده است. ۳۲ گروه از پرندگان در الگوریتم بهینه سازی پرندگان در نظر گرفته شده است. در برنامه الگوریتم برای فراهم کردن هر سه معیار برای پارامترها ضمن در نظر گرفتن مقدار تابع هدف تعداد دفعاتی که پارامتر انتخاب شده در طول شبیه سازی باعث

نقض معیار دوم شده است نیز محاسبه شده است تا دسته پرنده ای با بهترین عملکرد انتخاب شود. در شکل (۷-۶) و (۸-۶) روند کاهش تابع هدف نشان داده شده است.



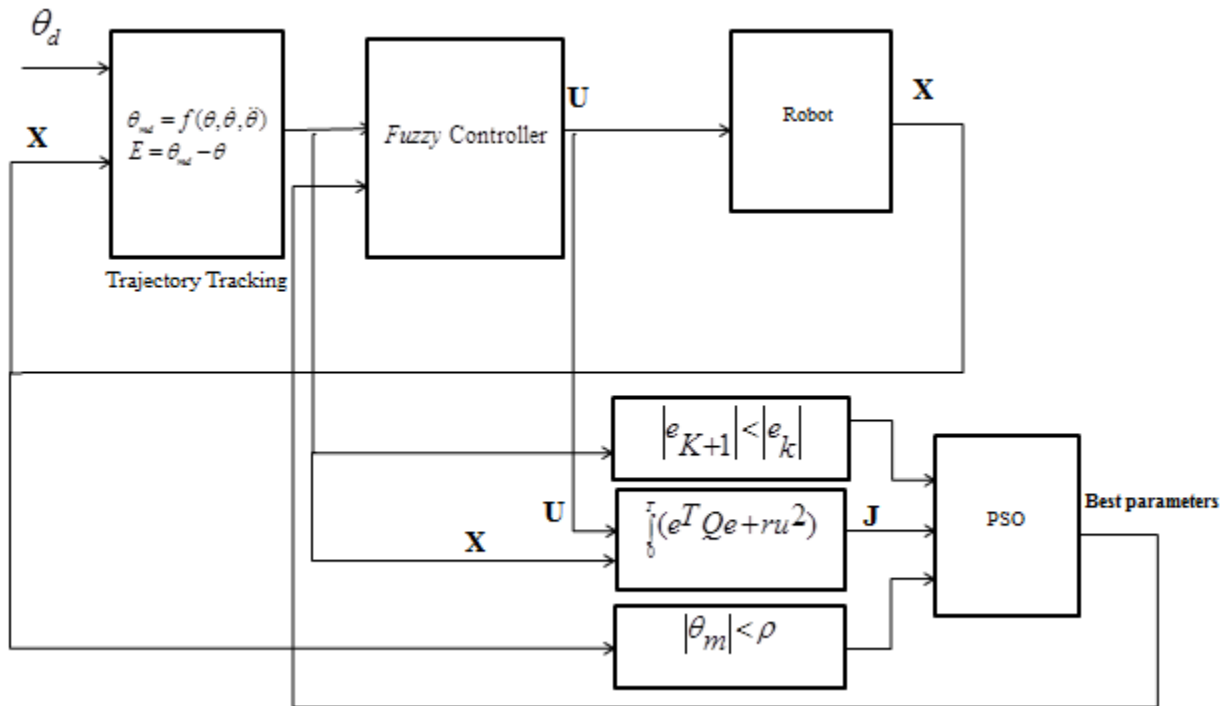
شکل (۷-۶) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط یک



شکل (۸-۶) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط دو

۶-۶-۲ پیاده سازی کنترل فازی بهینه ربات با مفاصل منعطف با استفاده از الگوریتم بهینه سازی پرندگان

ربات و مسیر طراحی شده مانند بخش (۶-۶-۱) می باشد. در اینجا کنترل کننده پیشنهادی در بخش (۶-۵) برای بررسی عملکرد آن شبیه سازی شده است. بلوک دیاگرام کنترل کننده در شکل (۶-۹) قابل مشاهده است. پارامترهای ورودی بلوک trajectory tracking با استفاده از شبیه سازی بخش قبل تنظیم شده است.

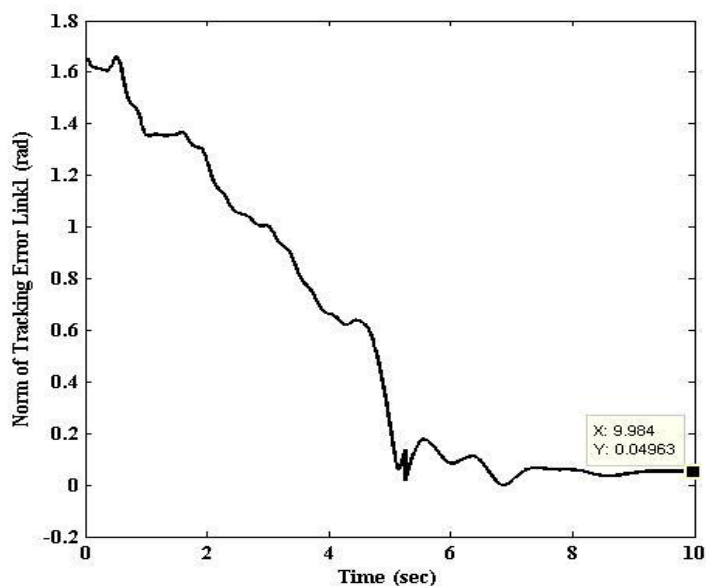


شکل (۶-۹) بلوک دیاگرام کنترل فازی بهینه ربات با مفاصل منعطف

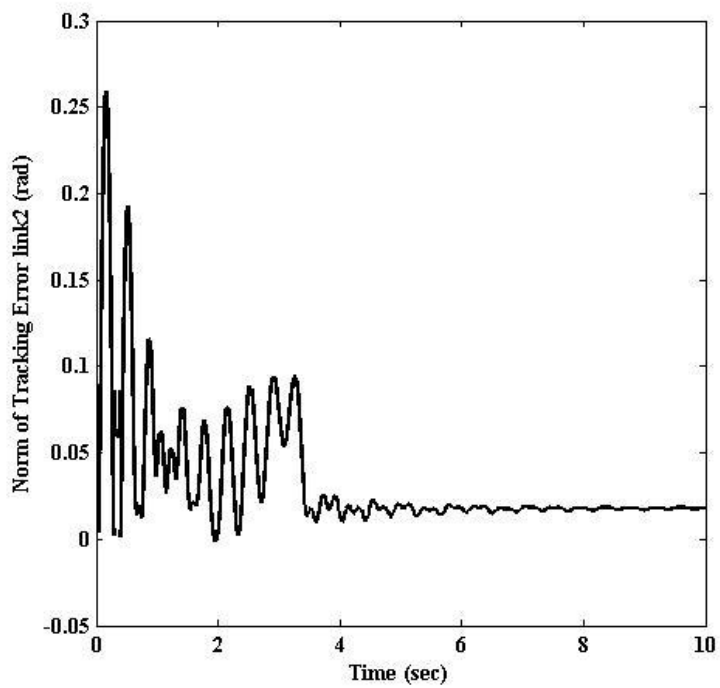
ورودی کنترل کننده فازی خطای موقعیت موتور و سرعت آن می باشد. در شکل های (۶-۱۰) و (۶-۱۱) نرم خطای رابط اول و دوم قابل مشاهده است. همانند کنترل خطی در اینجا نیز در بعضی نمونه های زمانی نرم افزایش یافته است که الگوریتم بهترین پارامتر را براساس دو معیار دیگر و البته کمتر نقض شدن معیار دوم انتخاب نموده است.

در شکل (۶-۱۲) و (۶-۱۳) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان قابل مشاهده است. ۳۲ گروه از پرندگان در الگوریتم بهینه سازی پرندگان در نظر گرفته شده است. الگوریتم بهینه سازی پرندگان ۵ بار

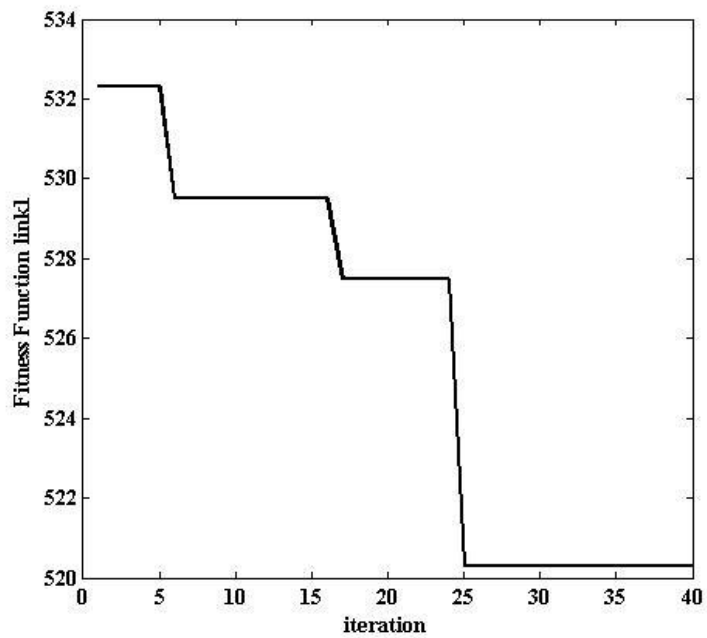
تکرار شده است و در هر دوره ۴۰ دوره تکرار به صورت خارج خط انجام شده است که بهترین نتیجه بدست آمده ارائه شده است. در آخر در شکل (۱۴-۶) و (۱۵-۶) سیگنال های کنترلی ورودی برای دو رابط به نمایش درآمده است. دو ویژگی محدود بودن سیگنال کنترلی و عدم وجود لغزش در آن، برای تضمین امکان پیاده سازی کنترل کننده بسیار مورد اهمیت است. وجود این دو ویژگی در کنترل کننده ارائه شده، در شکل (۱۴-۶) و (۱۵-۶) مشاهده می شود.



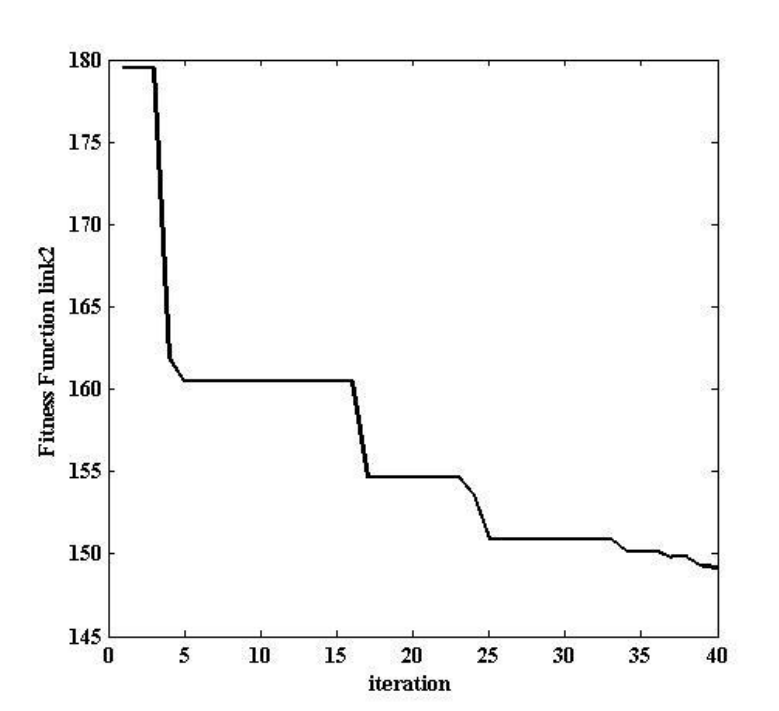
شکل (۶-۱۰) نرم خطای ردگیری رابط یک



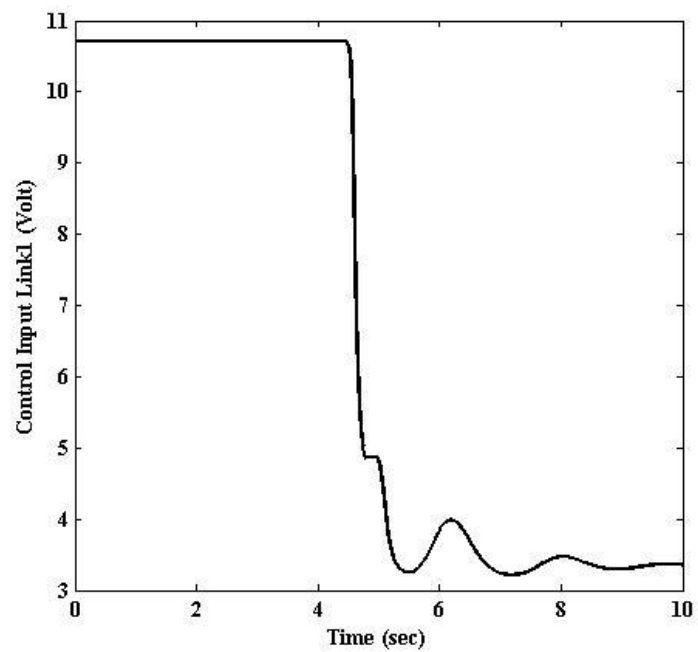
شکل (۶-۱۱) نرم خطای ردگیری رابط دو



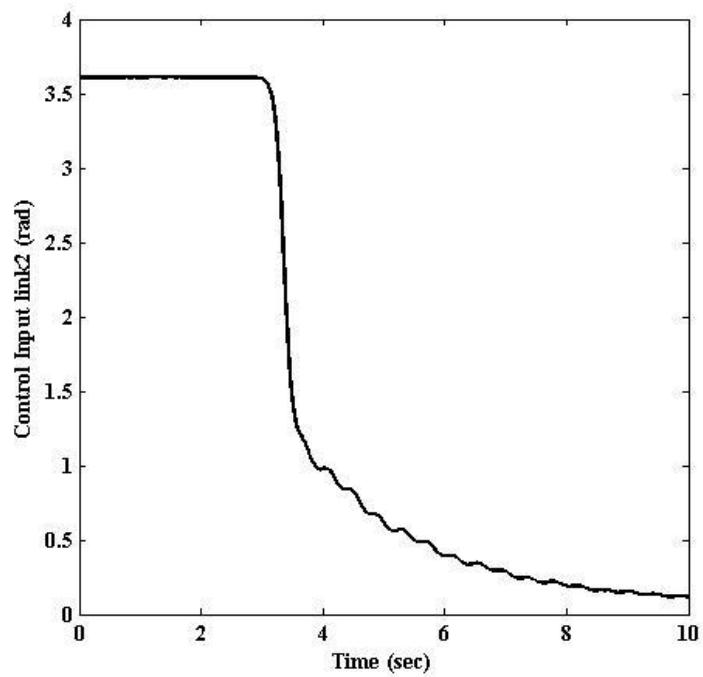
شکل (۶-۱۲) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط یک



شکل (۶-۱۳) روند کاهش تابع هدف در الگوریتم بهینه سازی پرندگان برای رابط دو



شکل (۶-۱۴) سیگنال کنترلی ورودی رابط یک



شکل (۶-۱۵) سیگنال کنترلی ورودی رابط دو

فصل هفتم

کنترل بهینه فازی بازوی ربات با مفاصل منعطف

با استفاده از الگوریتم بهینه سازی پرندگان به صورت بر خط

۷-۱ مقدمه

در دست نداشتن مدل سیستم مشکلاتی را در طراحی کنترل کننده در بردارد. از این رو محققان، با استفاده از روش های کنترلی مختلف از جمله کنترل تطبیقی یا استفاده از گرادیان نزولی مدل سیستم یا بخشی از آن را تخمین زده اند. تخمین مدل سیستم می تواند به صورت برخط یا خارج از خط صورت بگیرد. با این تفاوت که در حالت خارج خط تنها می توان مدل نامی سیستم را تخمین زد و عواملی از قبیل بار متغیر با زمان یا اغتشاشات موجود در محیط قابل تخمین نمی باشد. از طرفی هم در سیستم های پیچیده تخمین همزمان مدل سیستم فرآیند زمان بری است. با استفاده از منطق فازی می توان توابع مختلف خطی و غیر خطی را به خوبی تخمین زد. ترکیب این منطق با روش هایی چون خوشه بندی و گرادیان نزولی برای طراحی کنترل کننده ای که مدل سیستم یا بخشی از آن را تخمین می زند در گذشته به کار گرفته شده است.

همانطور که در فصل یک اشاره شد، در کارهای قبلی از روش بهینه سازی پرندگان برای بدست آوردن پارامترهای کنترل کننده به صورت برخط یا در حالت غیرمتصل استفاده شده است. در حالت غیرمتصل مدل سیستم در دسترس بوده و تنها پارامترها برای بدست آوردن پاسخ بهینه با استفاده از PSO تعیین می شوند. در حالت برخط نیز با روش کنترلی دیگری پایداری سیستم تضمین شده است و PSO تنها پارامترها را در محدوده مشخص شده برای تضمین پایداری برای رسیدن به پاسخ بهینه تنظیم می کند.

در این فصل، از قابلیت جستجو موازی و سرعت همگرایی بالای روش بهینه سازی پرندگان در تخمین مدل سیستم به صورت برخط و همزمان استفاده شده است. روش بهینه سازی پرندگان با تنظیم پارامترهای سیستم فازی مدل سیستم را به صورت برخط تخمین می زند، در نتیجه طراح مدل سیستم را برای کنترل آن در دسترس دارد.

روش ارائه شده در این فصل برای تخمین توابع در شناسایی سیستم ها می تواند بسیار کارآمد باشد. اما برای استفاده از این روش در کنترل سیستم ها باید این واقعیت را در نظر گرفت که روش های تخمین توابع به صورت برخط هرچند که سرعت بالایی در همگرایی به تابع مورد نظر داشته باشند، باز هم مقداری که برای نمونه بعدی تابع تخمین می زند از مقدار قبلی تابع پیوسته به خود تابع نزدیک تر نیست.

در نتیجه اگر خروجی تابع مورد نظر برای تخمین آن در دسترس باشد، به صرفه تر است که به جای تخمین آن، نمونه قبلی در طراحی کنترل کننده به کار گرفته شود. چراکه بدون صرف وقت برای محاسبات پیچیده، نزدیکترین مقدار در توابع پیوسته به نمونه بعدی همان نمونه قبلی است. بنابراین، استفاده از این روش در

طراحی کنترل کننده به صرفه نیست مگر نیاز به ساختار تابع مورد نظر در قانون کنترل وجود داشته باشد، نه مقدار خروجی آن. یعنی ساختار تابع برای قرار دادن ورودی های دیگری مورد نیاز باشد. در نتیجه این روش برای طراحی کنترل کننده بسیار کارآمد و مفید است.

در بخش ۷-۲ به توضیح روش تخمین تابع به صورت برخط با استفاده از منطق فازی و الگوریتم بهینه سازی پرندگان می پردازیم. در بخش ۷-۳ توابع مختلفی از مدل سیستم که تخمین آنها در طراحی کنترل کننده می-تواند موثر باشد برای انتخاب بهترین تابع مورد بررسی قرار می گیرد. در بخش ۷-۴ به چگونگی استفاده از این روش در طراحی کنترل کننده ربات با مفاصل منعطف می پردازیم. در بخش ۷-۵ شبیه سازی کنترل کننده طراحی شده ارائه می شود.

۷-۲- تخمین تابع با استفاده از روش بهینه سازی پرندگان

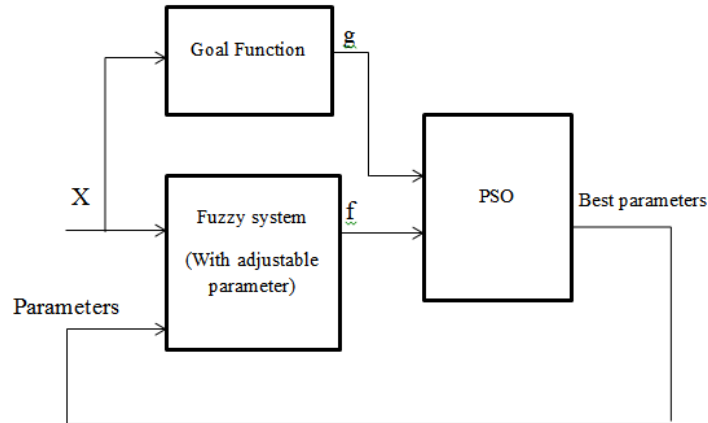
برای تخمین تابع لازم است تابع خطای زیر کاهش یابد.

$$E = (f - g)^2 \quad (1-7)$$

که در آن g نشان دهنده تابع هدف و f نمایانگر تابع تخمین زننده تابع g می باشد و هدف فرآیند تخمین، کاهش خطای E است. تابع E به عنوان تابع هزینه روش بهینه سازی پرندگان در نظر گرفته می شود.

تابع f یک سیستم فازی است که روش بهینه سازی پرندگان سعی دارد، طی فرآیند کمینه کردن تابع هزینه E با تنظیم پارامترهای های سیستم فازی در هر نمونه زمانی، تابع هدف را تقریب زند و در نتیجه بعد از چند نمونه زمانی تابع هدف به طور کامل توسط سیستم فازی تخمین زده می شود.

در شکل (۱-۷) روش کار الگوریتم تخمین نشان داده شده است. اگر ورودی تابع را X و خروجی را g در نظر بگیریم، در هر نمونه زمانی، زوج داده ورودی و خروجی به عنوان ورودی سیستم تخمین به بلوک Fuzzy داده می شود و تابع f به عنوان خروجی آن همراه با مقدار تابع g به بلوک PSO وارد می شود. الگوریتم بهینه سازی پرندگان با تشکیل خطای E سعی در کمینه کردن آن دارد. برای این منظور در هر لحظه با تغییر پارامترهای سیستم فازی مقدار f را به تابع هدف g نزدیک می کند. این روند برای هر نمونه زمانی تکرار می شود تا سیستم فازی در تمام طول بازه ورودی با خروجی منطبق شود.



شکل (۷-۱) روش کار تخمین تابع در هر نمونه زمانی

با توجه به اینکه از یک نقطه توابع زیادی را می توان عبور داد، برای تقریب بهتر تابع در هر نمونه زمانی چند زوج داده ورودی و خروجی در نظر گرفته می شود و PSO مربعات خطای این چند زوج داده را کاهش می دهد. توجه به این نکته نیز ضروری است که ممکن است با اعمال زوج داده های ورودی و خروجی در نمونه زمانی بعدی، سیستم فازی آنچه را که از زوج داده های قبلی یاد گرفته است، فراموش کند. بنابراین، در هر نمونه زمانی چند زوج داده از زوج داده نمونه قبلی نیز به الگوریتم بهینه سازی وارد می شود و در محاسبه مربعات خطا، خطای زوج داده های نمونه قبلی نیز در نظر گرفته می شود. یعنی اگر $X_1(k)$ تا $X_n(k)$ ورودی های سیستم در نمونه زمانی k ام باشند، $E_1(k)$ تا $E_n(k)$ نیز خطای مربوط به آنها می باشند.

$$E_i(k) = (f_i(k) - g_i(k))^2, \quad i = 1, \dots, n \quad (۷-۲)$$

تابع هزینه در هر نمونه زمانی برابر E_{sq} می باشد:

$$E_{sq} = \sqrt{E_1^2(k) + \dots + E_n^2(k) + E_1^2(k-1) + \dots + E_n^2(k-1)} \quad (۷-۳)$$

برای بهره گیری از این روش در کنترل ربات با مفاصل منعطف، انتخاب تابعی که تخمین آن بتواند در کنترل بازوی ربات موثر باشد بسیار مهم است. چراکه امکان تخمین کامل مدل سیستم به علت پیچیدگی و تزویج بالای معادلات دینامیکی ربات نمی باشد. هرچند که تخمین کامل مدل سیستم انتخاب هوشمندانه ای نیست و می توان تنها با تخمین قسمتی از مدل سیستم، کنترل کننده کارآمدی را پیشنهاد داد. برای این امر توابع موثر

در کنترل ربات با مفاصل منعطف را در بخش بعد معرفی می‌کنیم. سپس مزایا و معایب آنها را بررسی می‌نماییم. در نهایت تابعی مناسب را جهت تخمین توسط سیستم فازی انتخاب می‌کنیم.

۳-۷ توابع موثر در کنترل ربات با مفاصل منعطف

۱-۳-۷ تخمین تابع شتاب موتور

در دسترس بودن مدل سیستم کمک فراوانی در طراحی کنترل کننده می‌کند. با توجه به معادله (۸-۲) معادلات ربات با مفاصل منعطف داریم:

$$\dot{X} = f(X) + bu \quad (۴-۷)$$

که:

$$f(X) = \begin{bmatrix} x_2 \\ D^{-1}(x_1)(-g(x_1) - Kx_1 - C(x_1, x_2)x_2 + Krx_3) \\ x_4 \\ J^{-1}(rKx_1 - r^2Kx_3 - Bx_4 + K_mx_5) \\ -L^{-1}(K_b x_4 + Rx_5) \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ L^{-1} \end{bmatrix}, \quad X = \begin{bmatrix} \theta \\ \dot{\theta} \\ \theta_m \\ \dot{\theta}_m \\ I_a \end{bmatrix} \quad (۵-۷)$$

در این مدل تخمین معادله شتاب موتور با استفاده از معادله (۶-۲) می‌تواند به صورت زیر در طراحی قانون کنترل به کار گرفته شود.

$$J_m \ddot{\theta}_m + B\dot{\theta}_m + rK(r\theta_m - \theta) = \tau$$

$$\ddot{\theta}_m = J^{-1}(\tau - B\dot{\theta}_m - rk(r\theta_m - \theta)) \quad (۶-۷)$$

با توجه به معادلات (۷-۲) و (۸-۲) می توان معادله فوق را به صورت زیر بازنویسی نمود:

$$u = RI_a + K_b \dot{\theta}_m$$

$$K_m I_a = \tau \rightarrow \tau = k_m R^{-1}(u - k_b \dot{\theta}_m)$$

$$\ddot{\theta}_m = J^{-1}(k_m R^{-1}(u - k_b \dot{\theta}_m) - B\dot{\theta}_m - rk(r\theta_m - \theta)) \quad (7-7)$$

$$\ddot{\theta}_m = J^{-1}k_m R^{-1}(u) + g \quad (8-7)$$

که g به صورت رابطه (۹-۷) تعریف می شود:

$$g = J^{-1}(k_m R^{-1}(-k_b \dot{\theta}_m) - B\dot{\theta}_m - rk(r\theta_m - \theta)) \quad (9-7)$$

حال با فرض تخمین تابع g توسط الگوریتم بهینه سازی پرندگان، قانون کنترل به صورت زیر پیشنهاد می شود:

$$u = JRk_m^{-1}[(-g) + (\ddot{\theta}_{dm} + k_1(\dot{\theta}_{dm} - \dot{\theta}_m) + k_2(\theta_{dm} - \theta_m))] \quad (10-7)$$

با جایگزینی معادله (۱۰-۷) در معادله (۸-۷) و تشکیل معادله حلقه بسته داریم:

$$\begin{aligned} \ddot{\theta}_m &= (\ddot{\theta}_{dm} + k_1(\dot{\theta}_{dm} - \dot{\theta}_m) + k_2(\theta_{dm} - \theta_m)) \\ \ddot{e}_m + k_1 \dot{e}_m + k_2 e_m &= 0 \end{aligned} \quad (11-7)$$

اگر ضرایب معادله فوق هورویتز باشند سیستم پایدار خواهد بود و در نتیجه خطای موتور به سمت صفر میل می کند. برای تخمین تابع g نیاز به مقدار شتاب موتور در هر لحظه به عنوان خروجی و مقدار متغیرهای حالت موقعیت رابط و موقعیت و سرعت موتور در هر لحظه به عنوان ورودی داریم. در نتیجه سیستم فازی با ۳ ورودی و یک خروجی را برای تخمین تابع g ، در ربات یک رابط باید تشکیل دهیم. در ربات با دو مفصل به خاطر وجود تزویج سنگین باید دو سیستم فازی برای تخمین g_1 و g_2 ، هرکدام با ۶ ورودی تشکیل می شود. همانطور که مشاهده می شود با افزایش پارامترهای سیستم فازی، همگرایی آنها برای تخمین تابع g زمان بر می شود. حال کاملاً مشخص است که با نمونه زمانی کوچکتر می توان از مقدار قبلی تابع g استفاده کرد و تخمین تابع g مزیتی را دربر ندارد. ضمن آنکه قانون کنترل (۱۰-۷) به طور مستقیم همگرایی خطای

متغیرهای حالت رابط را دربر ندارد. همچنین مقدار مطلوب موقعیت موتور بدون در دسترس داشتن مدل سیستم مشخص نیست. در نتیجه، استفاده از این تابع به صرفه نمی‌باشد.

۷-۳-۲ تخمین تابع مشتق چهارم موقعیت رابط

با توجه معادلات فضای حالت بازوی ربات در فرم همراه داریم:

$$\dot{X} = f(X, u), \quad X = \begin{bmatrix} q \\ \dot{q} \\ \ddot{q} \\ \ddot{\ddot{q}} \end{bmatrix}, \quad q = [q_1 \quad q_2]$$

$$f(X, u) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \quad \begin{cases} f_1 = \dot{q} \\ f_2 = \ddot{q} \\ f_3 = \ddot{\ddot{q}} \\ f_4 = [q_1^{(4)} \quad q_2^{(4)}] \end{cases} \quad (12-7)$$

با تخمین تابع f_4 یا همان مشتق چهارم موقعیت رابط، همگرایی خطای همه پارامترها به سمت صفر تضمین می‌شود. برای تخمین این تابع باید مقدار خروجی یعنی مشتق چهارم و مقدار ورودی یعنی موقعیت رابط و مشتقات آن تا مرتبه سوم در هر لحظه در دسترس باشند. در ربات یک رابط، سیستم فازی باید با ۴ ورودی تشکیل شود و برای دورابط نیاز به ۸ ورودی برای هر سیستم فازی داریم. همانطور که مشاهده می‌شود، سیستم تشکیل شده بسیار پیچیده می‌شود. ضمن آنکه در دسترس نبودن مقدار مشتق سوم و چهارم موقعیت رابط در هر لحظه برای تشکیل زوج داده‌های خروجی و ورودی، نیز کار تخمین تابع را دشوار می‌کند. همچنین حتی اگر این مقادیر در دسترس باشند، استفاده از نمونه زمانی قبلی در قانون کنترل به صرفه تر خواهد بود. بنابراین، این تابع نیز برای تخمین و به کارگیری در قانون کنترل مناسب نمی‌باشد.

۷-۳-۳ تخمین تابع موقعیت موتور

مقدار مطلوب متغیرهای حالت رابط در هر لحظه معین است اما مقدار مطلوب متغیرهای حالت موتور با توجه به دینامیک ربات قابل دسترسی است. با فرض در دسترس بودن مدل سیستم و با توجه به بخش (۶-۴) داریم:

$$\theta_{md} = Kr^{-1}[D(\theta_d)\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d)\dot{\theta}_d + g(\theta_d) + K\theta_d] \quad (13-7)$$

یعنی مقدار مطلوب موقعیت موتور را با استفاده از مقادیر مطلوب موقعیت و سرعت و شتاب رابط بدست می-آوریم. حال اگر مدل سیستم در دسترس نباشد با تخمین تابع موقعیت موتور در هر لحظه و قرار دادن مقادیر مطلوب موقعیت و سرعت و شتاب رابط در آن لحظه مقدار مطلوب موتور در آن لحظه بدست می آید.

حال با داشتن مقدار مطلوب موقعیت موتور در هر لحظه می توان قانون کنترل زیر را پیشنهاد داد:

$$u(t) = RI + k_b[(\dot{\theta}_{md}) + k_1(\theta_{md} - \theta_m)] \quad (14-7)$$

با جایگذاری در معادله (7-15) و تشکیل معادله حلقه بسته داریم:

$$u = RI_a + K_b \dot{\theta}_m \quad (15-7)$$

$$RI + k_b \dot{\theta}_m = RI + k_b[(\dot{\theta}_{md}) + k_1(\theta_{md} - \theta_m)]$$

$$0 = [(\dot{\theta}_{md} - \dot{\theta}_m) + k_1(\theta_{md} - \theta_m)] \quad (16-7)$$

$$0 = \dot{e}_m + k_1 e_m$$

در نتیجه با هورویتز بودن ضرایب معادله فوق خطای موقعیت موتور به سمت صفر میل می کند. از آنجا که مقدار مطلوب موقعیت موتور از معادله (7-13) بدست آمده است، میل کردن موقعیت موتور به سمت مقدار طراحی شده، صفر شدن خطای موقعیت رابط را نیز در پی دارد.

$$\theta_{md} = Kr^{-1}[D(\theta_d)\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d)\dot{\theta}_d + g(\theta_d) + K\theta_d]$$

$$\theta_m \rightarrow \theta_{md} \Rightarrow \theta \rightarrow \theta_d$$

۷-۴ ساختار سیستم فازی

سیستم فازی از نوع تاگاکای- سوگنو انتخاب شده است و ساختار آن مانند کنترل کننده ارائه شده در فصل ششم می باشد. با این تفاوت که سیستم فازی دارای ۳ ورودی می باشد. که بردار ورودی شامل موقعیت، سرعت و شتاب رابط می باشد. در نتیجه ساختار سیستم فازی به صورت زیر قابل تعریف است: قوانین کنترل در روش کنترل فازی تاگاکای- سوگنو به صورت زیر می باشد :

$$Ru^l : \text{If } x_1 \text{ is } A_1^l \text{ and } x_2 \text{ is } A_2^l \text{ and } x_3 \text{ is } A_3^l \text{ Then } y^l = a_1^l x_1 + a_2^l x_2 + a_3^l x_3 + a_0^l \quad (17-7)$$

که در آن Ru^l بیان کننده قوانین فازی برای $l=1, \dots, 27$ می باشد ورودی قانون l ام x_1 ، x_2 و x_3 می باشد و A_i^l مجموعه فازی، a_i^l ضرایب ثابت برای $i=1, 2, 3$ می باشد. که در این کنترل کننده، x_1 ، x_2 و x_3 به ترتیب موقعیت

موتور و سرعت موتور و شتاب موتور می‌باشد. با استفاده از فازی ساز منفرد و غیر فازی ساز میانگین مراکز تابع $f(x)$ به صورت زیر پیشنهاد می‌شود:

$$f(x) = \frac{\sum_{l=1}^{27} \prod_{i=1}^3 \mu_{A_i^l}(x_i) y^l}{\sum_{l=1}^{27} \prod_{i=1}^3 \mu_{A_i^l}(x_i)} \quad (18-7)$$

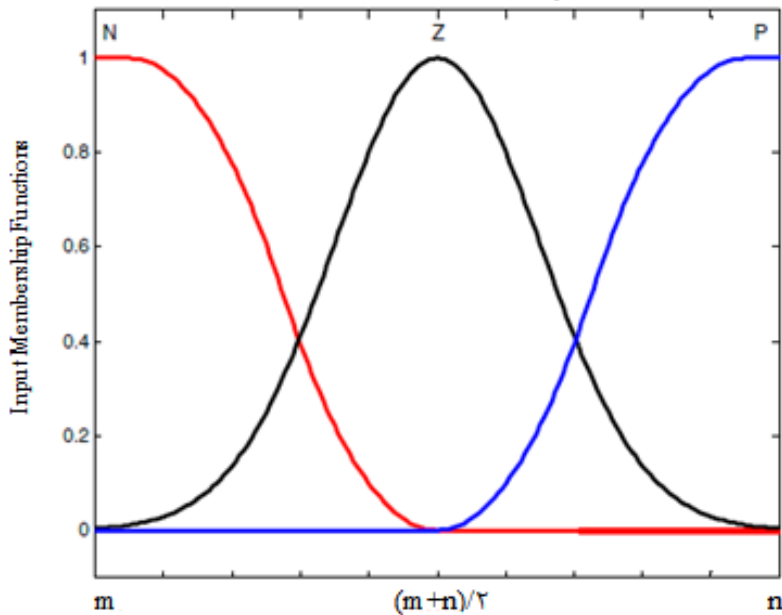
که $\mu_{A_i^l}(x_i) \in [0,1]$ توابع تعلق می‌باشند با قرار دادن y^l از معادله در معادله می‌توان به معادله ساده شده زیر رسید.

$$f(x) = c_1(x)x_1 + c_2(x)x_2 + c_3(x)x_3 + c_0(x) \quad (19-7)$$

که در آن $c_j(x)$ به ازای $j=0,1,2,3$ از رابطه زیر بدست می‌آید.

$$c_j(x) = \frac{\sum_{l=1}^{27} \prod_{i=1}^3 \mu_{A_i^l}(x_i) a_j^l}{\sum_{l=1}^{27} \prod_{i=1}^3 \mu_{A_i^l}(x_i)} \quad (20-7)$$

برای طراحی سیستم فازی به عنوان تخمین‌گر، مراکز توابع تعلق ورودی را می‌توان به دو صورت تعیین نمود. در حالت اول با دانستن محدوده تغییرات ورودی تابع هدف g می‌توان از قبل توابع تعلق ورودی را متناسب با آن طراحی نمود. در روشی دیگر، مراکز توابع تعلق را با استفاده از داده‌های ورودی تعیین نمود و در هر نمونه زمانی اگر داده‌ها در محدوده تعیین شده‌ای در مرحله قبل قرار داشتند، مراکز توابع تعلق تغییر نخواهند کرد. ولی اگر داده‌ای به سیستم وارد شد، که در این محدوده قرار نداشت، تابع تعلق جدید که مرکز آن، داده ورودی مورد نظر می‌باشد، به توابع تعلق اضافه می‌شود و بدین ترتیب توابع تعلق ورودی را گسترش می‌دهیم. لازم به ذکر است که توابع تعلق ورودی از نوع گوسین انتخاب شده‌اند. در شکل (۷-۲) توابع تعلق ورودی را مشاهده می‌کنید:



شکل (۷-۲) توابع تعلق ورودی

قوانین سیستم فازی به صورت زیر قابل تعریف می‌باشند:

Rule ۱ If x_1 is P and x_2 is P and x_3 is P Then $y = K_1$

Rule ۲ If x_1 is P and x_2 is Z and x_3 is P Then $y = K_2$

Rule ۳ If x_1 is P and x_2 is N and x_3 is P Then $y = K_3$

·
·
·

Rule ۲۵ If x_1 is N and x_2 is P and x_3 is N Then $y = K_{25}$

Rule ۲۶ If x_1 is N and x_2 is Z and x_3 is N Then $y = K_{26}$

Rule ۲۷ If x_1 is N and x_2 is N and x_3 is N Then $y = K_{27}$

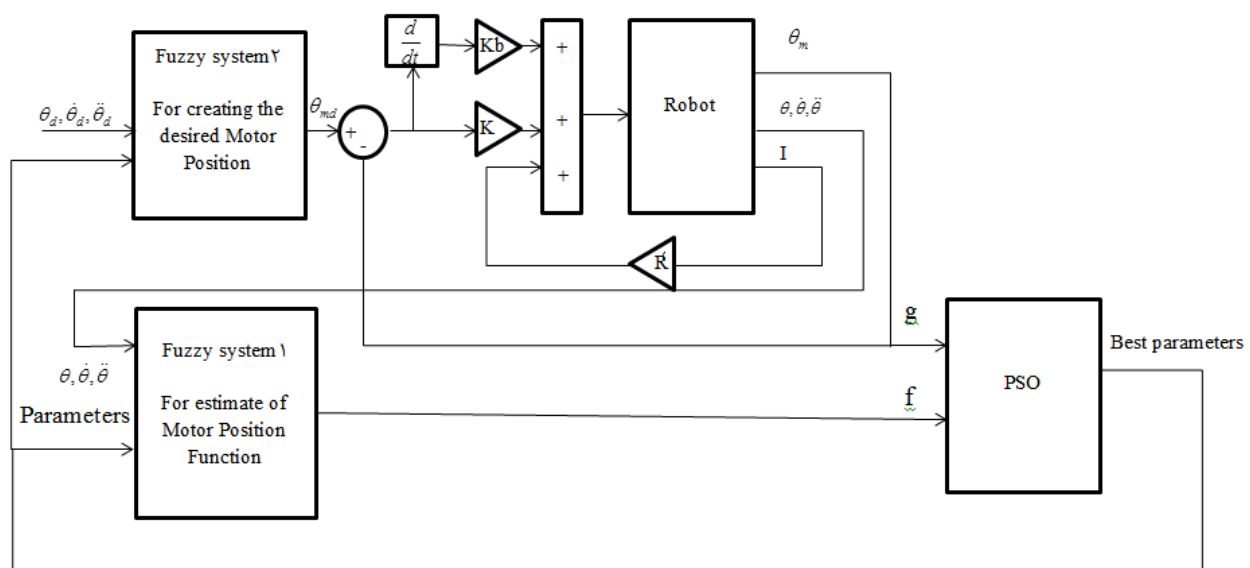
در روش پیشنهادی تنها تنظیم ضرایب y_1 با استفاده از PSO انجام می‌شود. در تنظیم این ضرایب در هر نمونه زمانی باید توجه داشت که فقط ضرایب قوانینی که مقدار تابع تعلق صفر نیست را تنظیم نمود و ضرایب قوانین دیگر را طبق مرحله قبل ثابت نگاه داشت. عدم توجه به این نکته باعث می‌شود ضرایب قوانینی که فعال نشده‌اند، به صورت کنترل نشده تغییر کند. زیرا این قوانین در این مرحله تاثیری در کاهش خطای تابع هزینه ندارند

و تنظیم آنها توسط الگوریتم بهینه سازی تأثیری بر مقدار تابع هدف ندارد. در نتیجه، در نقطه نامعلومی تنظیم می‌شوند و در مرحله بعدی که این قوانین فعال می‌شوند، باعث ناپایداری آنی سیستم می‌شوند.

۷-۵ شبیه سازی

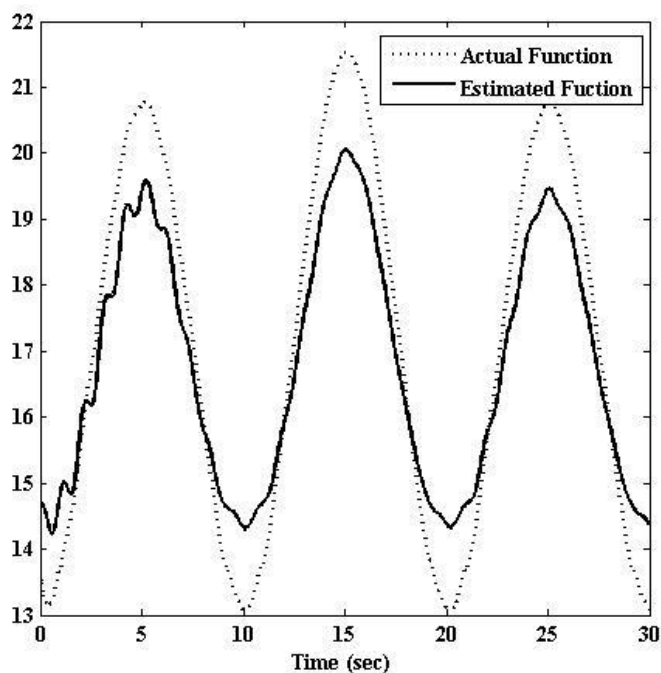
مسئله اصلی در کنترل ربات با مفاصل منعطف پیروی موقعیت رابط از موقعیت موتور است. چراکه در ربات با مفاصل منعطف موقعیت مطلوب رابط توسط طراح تعیین می‌شود اما موقعیت موتور باید با توجه به دینامیک ربات بدست آید. اما با توجه به پیچیدگی ربات با مفاصل منعطف استفاده از مدل سیستم حتی اگر در دسترس باشد به صرفه نیست. در این روش پیشنهادی تابع موقعیت مطلوب موتور توسط یک بلوک فازی به صورت برخط تخمین زده می‌شود.

بلوک دیاگرام نشان داده شده در شکل (۷-۳) بیانگر نحوه طراحی کنترل کننده می‌باشد. در این کنترل کننده با تخمین تابع موقعیت موتور در هر لحظه مقدار مطلوب موقعیت موتور توسط ورودی های موقعیت، سرعت و شتاب رابط ساخته می‌شود و به عنوان ورودی کنترل کننده برای ساخت خطای متغیرهای حالت موتور به کار گرفته می‌شود. در الگوریتم پیشنهادی اگر خطای نسبی تخمین کمتر از ۰.۰۱ باشد، روند برنامه از تغییر پارامترها خارج می‌شود و با همان پارامترها نمونه زمانی بعدی را طی می‌کند. پس لازم است در هر نمونه زمانی خطای نسبی اندازه گیری شود.



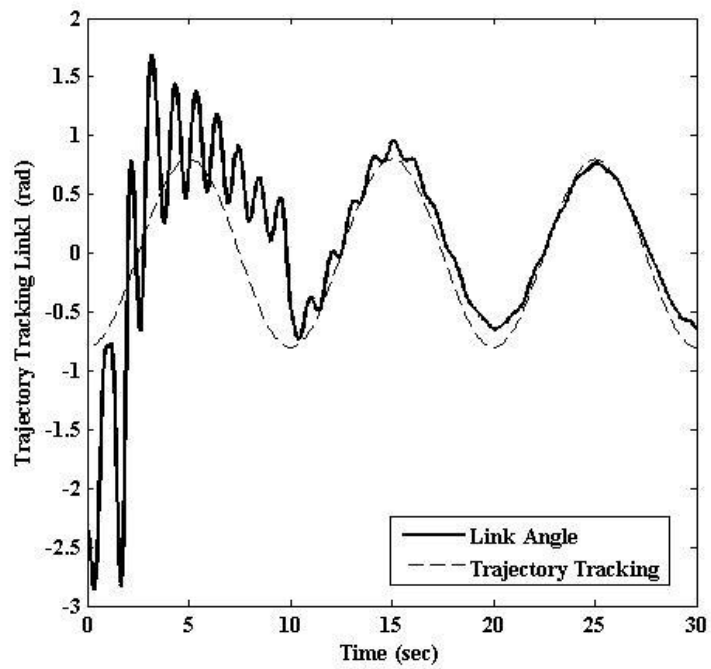
شکل (۷-۳) بلوک دیاگرام کنترل کننده پیشنهادی

شکل (۴-۷) نشان دهنده نحوه تخمین تابع توسط الگوریتم PSO به صورت برخط و همزمان با شروع به کار ربات می‌باشد. همانطور که در شکل دیده می‌شود در چند ثانیه الگوریتم بخوبی تابع را تخمین می‌زند و در اختیار کنترل کننده قرار می‌دهد.

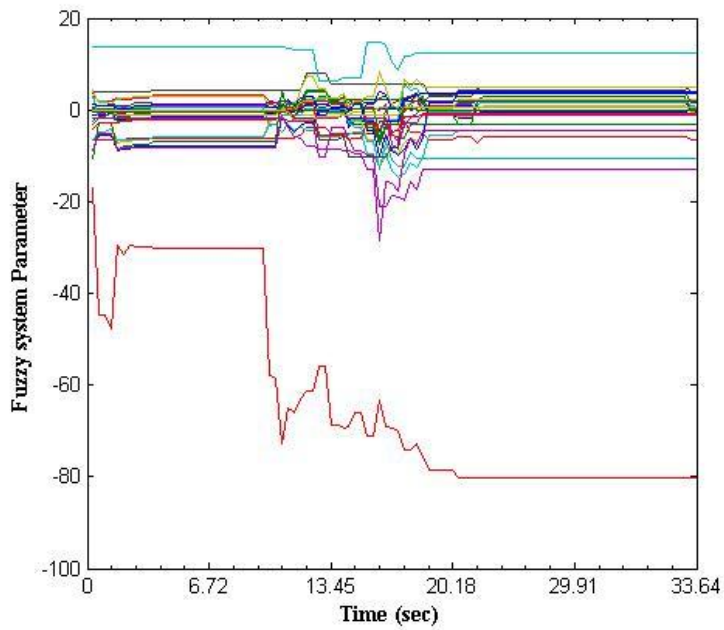


شکل (۴-۷) تخمین تابع موقعیت موتور

شکل (۵-۷) بیانگر مسیر ردگیری رابط را در طی تخمین تابع توسط الگوریتم می‌باشد. همانطور که دیده می‌شود در ابتدا روند ردگیری مناسب نیست اما بعد از تخمین تابع الگوریتم به خوبی مسیر را پیگیری می‌کند.



شکل (۷-۵) مسیر ردگیری موقعیت رابط



شکل (۷-۶) تغییر پارامترهای سیستم فازی در طول شبیه سازی

در شکل (۶-۷) نحوه تغییر پارامترها در ۳۳ ثانیه را مشاهده می‌کنید. همانطور که دیده می‌شود در زمانی که سیستم فازی نزدیک به مقدار تابع هدف است پارامترها ثابت هستند. همچنین در زمانی که قانونی فعال نشده باشد ضریب آن قانون تغییر نمی‌کند اما دیگر ضرایب برای نزدیک کردن سیستم فازی به تابع هدف تغییر می‌کند.

۶-۷ نتیجه گیری

در این فصل با استفاده از الگوریتم بهینه سازی پرندگان، قسمتی از مدل سیستم به صورت برخط تخمین زده شده است که از آن در طراحی دقیق کنترل کننده استفاده شده است. کنترل کننده پیشنهادی در مواقعی که مدل سیستم در دسترس نباشد بسیار کارآمد است. از معایب این کنترل کننده ثانیه های اولیه شروع به کار سیستم است که هنوز الگوریتم به تابع اصلی همگرا نشده است. در این الگوریتم تا حد امکان از پرس های ناگهانی جلوگیری شده است.

در فصل های گذشته برای اجرای کنترل کننده پیشنهادی نیاز به مدل سیستم است تا در حالت غیر متصل پارامترها توسط PSO تخمین زده شود. حال اگر مدل سیستم در دسترس نباشد از روش ارائه شده در تخمین کامل مدل سیستم نیز می‌توان بهره گرفت و از آن در طراحی های غیر متصل که در فصل های گذشته ارائه شده استفاده نمود.

فصل هشتم

نتیجه گیری و پیشنهادات

۸-۱ نتیجه گیری

در این پایان نامه روشی برای طراحی هوشمند کنترل کننده فازی برای بازوی رباتیک با مفاصل منعطف ارائه شد به طوری که در ضمن تنظیم پارامترها بدون استفاده از روش مرسوم تحلیل پایداری، پایداری سیستم محقق شد. در عین حال عملکرد سیستم نیز بهینه شد.

برای این منظور کنترل کننده ارائه شده در فصل چهارم به خوبی توانست یک سیستم خطی را با استفاده از الگوریتم بهینه سازی پرندگان، در ردگیری مسیر و تنظیم نقطه ای کنترل کند. اما به دلیل وابستگی روش ارائه شده به مدل سیستم محدودیت هایی را در طراحی مسیر برای ربات با مفاصل منعطف ایجاد می نمود. برای غلبه بر این محدودیت ها در فصل ششم با استفاده از تابع هدف فصل چهارم و اضافه نمودن شرایطی که از تئوری های پایداری نتیجه شده است، روشی کارآمد برای کنترل انواع سیستم های غیرخطی از جمله ربات با مفاصل منعطف ارائه داده شد. این روش به خوبی توانست در حالت خارج خط، پارامترهای کنترل کننده فازی را به گونه ای تنظیم نماید که عملکرد بهینه و پایداری به طور همزمان فراهم گردد. در مقایسه با کنترل کننده که در فصل پنجم از مرجع [۲۶] پیاده سازی شد، کنترل کننده ارائه شده در فصل ششم با کمترین استفاده از دانش فرد خبره و بدون نیاز به ساده سازی در مدل سیستم به خوبی از عهده کنترل ربات با مفاصل منعطف برآمده است. در ضمن کنترل کننده فصل پنجم برای رسیدن به عملکرد بهینه نیاز به تنظیم پارامترها توسط الگوریتم بهینه سازی پرندگان دارد و تحلیل پایداری کنترل کننده فازی ارائه شده بدون ساده سازی های انجام شده دشوار می باشد.

در فصل هفتم با استفاده از الگوریتم بهینه سازی پرندگان به صورت برخط پارامترهای سیستم فازی برای تخمین قسمتی از مدل سیستم تنظیم شد تا در مواقعی که مدل سیستم در دسترس نباشد بتوان کنترل کننده فازی را بدون استفاده از دانش فرد خبره تنظیم نمود.

۸-۲ پیشنهادات

کنترل کننده پیشنهادی در فصل ششم بدون نیاز به مدل سیستم توانست کنترل کننده پایدار با عملکرد بهینه را تنظیم نماید اما برای پیاده سازی این روش توسط الگوریتم بهینه سازی پرندگان و اجرای حلقه های تکرار الگوریتم نیاز به مدل سیستم در حالت خارج خط می باشد. یعنی در نهایت باید مدل سیستم در دسترس باشد تا بتوان کنترل کننده پیشنهادی را پیاده سازی نمود. برای رفع این نیاز می توان با استفاده از روش ارائه شده در فصل هفتم مدل سیستم را در حالت برخط تخمین زد و از آن برای پیاده سازی کنترل کننده فصل ششم بهره برد.

برای بهبود عملکرد سیستم تخمین تابع توسط الگوریتم بهینه سازی پرندگان می توان از پردازنده هایی با سرعت بالاتر استفاده نمود.

منابع و ماخذ:

۱. Kuntze H. B. , and Jacubasch A. ,(۱۹۸۴), "*On the closed loop control of an elastic industrial robot*", American Control Conf.
۲. Cesareo G. , Marino R. ,(۱۹۸۴), "*On the controllability properties of elastic robots*", Int.Conf. on Analysis and Optimization of Systems.
۳. Cesareo G., Marino R.,(۱۹۸۴), "*On the controllability of elastic robots*", INRIA Int. Conf., Lecture Notes in Control and Inf. Sci., Springer-Verlag.
۴. Khorasani K. , Spong M. W.,(۱۹۸۵), "*Invariant manifolds and their application to robot manipulators with Flexible Joints*", IEEE Int. Conf. on Robotics and Automation, ۹۷۸-۹۸۳.
۵. Nicosia S. , and Tomei P. ,(۱۹۸۸), "*On the feedback linearization of robots with elasticjoints*", IEEE Int. Conf. on Decision and Control, ۱۸۰-۱۸۵.
۶. Readman M. C. and Belanger P. R.,(۱۹۹۰), "*Analysis and Control of a flexible jointrobot*", IEEE Int. Conf. on Decision and Control, ۲۵۵۱-۲۵۵۹.
۷. Murphy S. H., Wen J. T., and Saridis G. N.,(۱۹۹۰), "*Simulation and analysis of flexiblyjointed manipulators*", IEEE Int. Conf. on Decision and Control, ۵۴۵-۵۵۰.
۸. Bridges M. M. , and Dawson D. M.,(۱۹۹۵), "*Redesign of robust controllers for rigid-link flexible-joint robotic manipulators actuated with harmonic drive gearing*", IEE Proceedings. Control Theory and Applications, ۱۴۲, ۵, ۵۰۸-۵۱۴.
۹. ازگلی، س، ۱۳۸۵، رساله دکتری، طراحی و پیاده سازی کنترلر موقعبیت برای ربات کشسان مفصل با لحاظ مسئله اشباع کار انداز، دانشکده برق، دانشگاه صنعتی خواجه نصیر طوسی.
۱۰. ایزدبخش، س، ۱۳۹۱، رسایه دکتری، کنترل مقاوم رباتهای با مفاصل انعطاف پذیر در فضای کار، دانشکده برق و رباتیک، دانشگاه شاهرود.
۱۱. Tomei P.,(۱۹۹۱), "*A simple PD controller for robots with elastic joints*"IEEE Trans. Autom. Control ۳۶, ۱۰, ۱۲۰۸-۱۲۱۳.
۱۲. Fateh M.M.,(۲۰۱۲), "*Nonlinear control of electrical flexible-joint robots*" Nonlinear Dyn, ۶۷:۲۵۴۹-۲۵۵۹.
۱۳. Wilson G.A, (۱۹۹۴), "*Robust tracking of elastic joint manipulators using sliding mode control*" Trans, Inst. Meas. Control ۱۶, ۲, ۹۹-۱۰۷.
۱۴. Spong M.W., (۱۹۹۵), "*Adaptive control of flexible joint manipulators: comments on two papers*" Automatica, ۳۱, ۴, ۵۸۵-۵۹۰.

10. Yoo Sh.J. , Park J.B., Choi Y.H. , (2008), "Adaptive output feedback control of flexible-joint robots using neural networks dynamic surface design approach" IEEE Trans. Neural Netw, 19, 10, 1712-1726.
11. Chen B., Liu X.P., Shi P., (2010), "Direct adaptive fuzzy control for nonlinear systems with time-varying delays" Inf. Sci. 180, 776-792.
12. Zhou Q., Shi P. , Lu J., Xu S.Y., (2011), "Adaptive output feedback fuzzy tracking control for a class of nonlinear systems" IEEE Trans. Fuzzy Syst , 19, 972-982.
13. Li T.S. , Tong S.C. , Feng G. , (2010), "A novel robust adaptive fuzzy tracking control for a class of nonlinear MIMO systems", IEEE Trans. Fuzzy Syst, 18, 100-110.
14. Li Y., T Sh., Li T., "Adaptive fuzzy output feedback control for a single-link flexible robot manipulator driven DC motor via backsteppin" Nonlinear Analysis, 2011.
15. Spong M. W. ,(1987), "Modeling and control of elastic joint robots" Journal of Dynamic Systems, Measurement, and Control, 109, 4,
16. Fateh M. M.,(2008), "On the voltage based control of robot manipulators", International Journal of Control, Automation, and Systems, 6, 0, 702-712.
17. Fateh M. M. ,(2012), "Robust control of flexible joint robots using voltage control strategy" Nonlinear Dyn, 67, 2, 1020-1037.
18. Fateh M. M. , (2010), "Robust Fuzzy Control of Electrical Manipulators" J Intell Robot Syst, 60, 410-434.
19. Tian L., Goldenberg A. A., (1990). "Robust adaptive control of flexible joint robots with joint torque feedback", IEEE Int. Conf. on Robotics and Automation, 1229-1234.
20. Zadeh L.A (1965), "Fuzzy sets", Information and Control, 8, 338-353.
21. Weiming T., Guanrong Ch., Rongde L., (2001) ,"A modified fuzzy PI controller for a flexible-joint robot arm with uncertainties " Fuzzy Sets and Systems , 109, 118-119.
22. Ying H.,Siler W., Buckley J.J., (1990), "Fuzzy control theory a nonlinear case" , Automatica , 26, 013-020.
23. Krause P. C.,(1986), "Analysis of electric machinery", McGraw-Hill, New York.
24. Mohd Ashraf Ahmad., Mohd Zaidi Mohd Tumari., Ahmad Nor Kasruddin Nasir., (2013) "Composite Fuzzy Logic Control Approach to a Flexible Joint Manipulator" International Journal of Advanced Robotic Systems, 10, 08.

٣٠. Rostami Kandroodi M., Mansouri M., Aliyari Shoorehdeli M., Teshnehlab M., (٢٠١٢) "*Control of Flexible Joint Manipulator via Reduced Rule-Based Fuzzy Control with Experimental Validation*" International Scholarly Research Network. ISRN Artificial Intelligence.
٣١. Ahmad M. A. , (٢٠٠٨) , "*Vibration and input tracking control of flexible manipulator using LQR with non-collocated PID controller,*" in Proceeding of ٧nd UKSIM European Symposium on Computer Modelling and Simulation, ٤٠-٤٥.
٣٢. Park C.W. and Cho Y.W., (٢٠٠٣), "*Adaptive tracking control of flexible joint manipulator based on fuzzy model reference approach*", IEE Proc.-Conna; Themy Appl., ١٥٠, ٢.
٣٣. Li L.L., Wang L., Liu L.L. , (٢٠٠٦), "*An effective hybrid PSOSA strategy for optimization and its application to parameter estimation*", Applied Mathematics and Computation, ١٧٩ , ١, ١٣٥-١٤٦.
٣٤. Noel M.M., Jannett T.C., (٢٠٠٤), "*Simulation of a new hybrid particle swarm optimization algorithm*", in: Proc ٣٦th Southeastern Symposium on System Theory, ١٥٠-١٥٣.
٣٥. Juang C.F., (٢٠٠٤), "*A hybrid of genetic algorithm and particle swarm optimization for recurrent network design*", IEEE System, Man, and Cybernetics, ٣٤ , ٢, ٩٩٧-١٠٠٦.
٣٦. Kennedy, Eberhart R.C. , (١٩٩٥) , "*Particle swarm optimization*" , Proceedings IEEE International Conference on Neural Networks, ١٩٤٢-١٩٤٨.
٣٧. Modares H., Alfi A., Fateh M.M. , (٢٠١٠), "*Parameter identification of chaotic dynamic systems through an improved particle swarm optimization*" . Expert Syst. Appl, ٣٧, ٣٧١٤-٣٧٢٠ .
٣٨. Tung S.L. , Juang Y.T. , Lee W.H. , Shieh W.Y. , Wua W.Y. , (٢٠١١) , "*Optimization of the exponential stabilization problem in active suspension system using PSO Expert Systems with Applications*" ٣٨ , ١٤٠٤٤-١٤٠٥١.
٣٩. Wai R.J. , Chuang K.L. ,(٢٠٠٩), "*Design of backstepping particle-swarm optimisation control for maglev transportation system*" IET Control Theory and Applications .
٤٠. Wai R.J. , Chuang K.L. , and Lee J.D. ,(٢٠١٠), "*On-Line Supervisory Control Design for Maglev Transportation System via Total Sliding-Mode Approach and Particle Swarm Optimization*", IEEE TRANSACTIONS ON AUTOMATIC CONTROL, ٥٥, ٧.
٤١. Wai R. J. , (٢٠٠٣), "*Supervisory genetic evolution control for indirect field-oriented induction motor drive*" Proc. Inst. Elect. Eng., ١٥٠, ٢, ٢١٥-٢٢٦.

- ξϰ. Sharma K.D., Chatterjee A., and Rakshit A. ,(2012), "A PSO-Lyapunov Hybrid Stable Adaptive Fuzzy Tracking Control Approach for Vision-Based Robot Navigation " ,IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, 61, 7.
- ξϱ. Sharma K.D., Chatterjee A., and Rakshit A.,(2009), "A hybrid approach for design of stable adaptive fuzzy controllers employing Lyapunov theory and particle swarm optimization," IEEE Trans. Fuzzy Syst., 17, 2, 329-342,.
- ξϲ. Martacuta R. , Castillo O. , and Aguilar L. T.,(2009) , "Optimization of interval type- 2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms," J. Inf. Sci., 179, 13, 2108-2114.
- ξϳ. Chiou J.S., Tsai S.H. , Liu M.T. , (2012), "A PSO-based adaptive fuzzy PID-controllers"Simulation Modelling Practice and Theory, 26.
- ξϴ. Haifa M. , Olfa B. ,(2011) , "Impedance Controller Tuned by Particle Swarm Optimization for Robotic Arms" International Journal of Advanced Robotic Systems Int J Adv Robotic System, 8, 5, 93-103.
- ξϵ. Salehi M. , Vossoughi G.R., Vajedi M., Brooshaki Impedance M., (2008), "Control and Gain Tuning of Flexible Base Moving Manipulators Using PSO Method"Proceedings" IEEE International Conference on Information and Automation , 20 -22, Zhangjiajie, China.
- ξ϶. Chiou J.S., Liu M.T. , (2009), "Numerical simulation for fuzzy-PID controllers and helping EP reproduction with PSO hybrid algorithm" , Simulation Modelling Practice and Theory , 17 , 10, 1000-1060.
- ξϷ. Eberhart R.C., Shi Y.,(2002) , "Comparing inertia weights and constriction factors in particle swarm optimization", in: Proceedings of the IEEE Congress on Evolutionary Computation, 84-88, San Diego, USA.
- ϰ0. Juang C.F. , (2004), "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design" , IEEE System, Man, and Cybernetics, 34 , 2, 997-1006.
- ϰ1. Huang T., Mohan A.S. , (2005), " A hybrid boundary condition for robust particle swarm optimization" , IEEE Antennas and Wireless Propagation Letters 4 , 112-117.
- ϰ2. Liu D., Tan K.C., Goh C.K., Ho W.K., (2007) ,"A multiobjective memetic algorithm based on particle swarm optimization", IEEE System, Man, and Cybernetics, 37 , 1, 42-50.
- ϰ3. Noel M.M., Jannett T.C.,(2004), "Simulation of a new hybrid particle swarm optimization algorithm", in: Proc 37th Southeastern Symposium on System Theory, 100-103.

٥٤. Xin J., Chen G. Hai Y.,(٢٠٠٩). "A Particle Swarm Optimizer with Multistage Linearly Decreasing Inertia Weight", In Computational Sciences and Optimization, CSO, International Joint Conference on ١, ٥٠٥-٥٠٨.
٥٥. Malik R.F.,Rahman T.A., Hashim S.Z.M. Ngah R.,(٢٠٠٧) "New Particle Swarm Optimizer with Sigmoid Increasing Inertia Weight", International Journal of Computer Science and Security (IJCSS), ١,٢,٣٥.
٥٦. Eberhart R.C., Shi Y.,(٢٠٠٢). "Tracking and optimizing dynamic systems with particle swarms", In Evolutionary Computation, IEEE, ١,٩٤-١٠٠.
٥٧. کلامی هریس م، پریز ن،(٢٠٠٧) "تاثیر میزان جمعیت و ضرایب یادگیری در عملکرد الگوریتم PSO"، سومین کنفرانس بین المللی فناوری اطلاعات و دانش_مشهد
٥٨. Das S. , Abraham A. and Konar A. , (٢٠٠٨), "Particle Swarm Optimization and Differential Evolution Algorithms Technical Analysis, pplications and Hybridization Perspectives," Springer-Verlag Berlin Heidelberg, ١١٦ ١-٣٨.
٥٩. He Z., Wei C., Yang L., Gao X., Yao S., Eberhart R. and Shi Y.,(١٩٩٨), "Extracting Rules from Fuzzy Neural Network by Particle Swarm Optimization," Proceedings of IEEE Congress on Evolutionary Computation, Anchorage, Alaska, USA,.
٦٠. Vidyasagar M.,(١٩٩٣), " Nonlinear Systems Analysis", Prentice-Hall, Englewood Cliffs, NJ,.
٦١. Wang H., Wu Z. J., Liu Y. , (٢٠٠٩), "Space Transformation Search: A New Evolutionary Technique. Genetic and Evolutionary Computation". International Journal of Computer Science and Security (IJCSS), ١,٢,٣٥.
٦٢. Li W., Slotine,J.J.E,(١٩٨٩), "Indirect Adaptive Robot control" System and Control
٦٣. Athans, M., (١٩٦٦), "The status of optimal control theory and applications for deterministic systems," IEEE Trans. Automatic Control, ٥٨٠-٥٩٦.
- ٦٤.Kirk, Donald E., (١٩٣٧), "Optimal control theory: an introduction", Carmel/ California,.
٦٥. Kalman R. E.,(١٩٦٤), "When is a linear control system optimal?," J. Basic Eng. Trans., ASME, ٨٦, ٥١-٥٦.
٦٦. Wang .Y.,(١٩٩٢), "The determination of weighting matrices in LQ optimal control system," ACTA Automatic Sinica,.
٦٧. Sung G.C., Chen G. ,(٢٠٠٦), "Optimal control systems design associated with genetic algorithm," Proceedings of ٢٦ CACS Automatic Control Conference St. John's University, ١٠-١١.Tamsui, Taiwan.

٦٨. Bottura C. P. Fonseca Neto J. V.,(٢٠٠٠), "*Rule based decision making unit for eigenstructure assignment via parallel genetic algorithm and LQR design*, " Proceeding of the American Control Conference, Chicago, Illinois, ١,٤٦٧- ٤٧١.

٦٩. Bottura C. P. and da Fonseca Neto J. V.,(١٩٩٩), "*Parallel eigenstructure assignment via LQR design and genetic algorithms*," Proceeding of the American Control Conference, San Diego,٤,٢٢٩٥-٢٢٩٩.

٧٠. Fateh M. M. , Khorashadizadeh S. , "*Optimal robust voltage control of electrically driven robot Manipulators*" Nonlinear Dyn .

٧١. Wong C.C., Wang H.Y., Li S. , (٢٠٠٨), "*A PSO-based motion fuzzy controller design for mobile robots*". Int. J. Fuzzy Syst. ١٠,١, ٢٨٤-٢٩٢.

Abstract

In this thesis, the issue of the flexibility in joints and its effects is reviewed. Since the fuzzy controller has shown a good performance in the flexible-joint robot control, we introduce and design the fuzzy controller for the flexible-joint robot manipulator. Despite of the advantages of fuzzy controller, adjustment of the fuzzy control parameters by an expert and complexity of stability analysis show always challenges. Therefore, the use of intelligent algorithms in the optimal design of the fuzzy controller has been used. Among the intelligent optimization algorithms, particle swarm optimization algorithm has an appropriate response in optimization of the complex and non-linear problems with a high convergence rate. Hence, this thesis presents the design of controllers with guaranteed stability by using a particle swarm optimization algorithm. In this regard, designing an optimal linear controller with guaranteed stability by employing particle swarm optimization algorithm is presented for linear systems. Then, the proposed method is extended for nonlinear systems such as a robotic manipulator. After that, by applying particle swarm optimization algorithm the fuzzy controller is designed somehow its stability is guaranteed and the controller provides an optimal performance. The design of optimal fuzzy controller are presented for off-line and online types. Innovation of this thesis is the use of particle swarm optimization algorithm to design the guaranteed stability fuzzy controller. Its advantage is the simplicity of design compared to conventional Lyapunov method which is difficult to analyze the fuzzy control system applied on the flexible joint robot manipulator equipped with electric motors. Also, a linear optimal controller design with guaranteed stability by applying particle swarm optimization algorithm for linear systems and its generalization for non-linear systems such as robot manipulator is novel.

Keywords: Flexible-joint robot manipulator, Particle swarm optimization algorithm, Fuzzy control, Optimal linear quadratic control



Shahrood University of Technology

Faculty of Mechanics

**Design of a Fuzzy Controller for a Robot Manipulator
with Flexible Joints Using PSO Algorithm**

Zohreh Safarcharati

Supervisor

Dr.M.M.Fateh

Sep. ۲۰۱۳