مجله کنترل
ISSN (print) 2008-8345
ISSN (online) 2538-3752
جلد ۱۱، شماره ۴، زمستان ۱۳۹۶، صفحه ۲۴-۱۳

# استخراج ویژگی از داده‌های عمق با استفاده از روش یادگیری عمیق برای کنترلِ باناظر ربات چرخ‌دار

فریناز اعلمی‌یان هرندی[۱]، ولی درهمی[۲]

[۱] دانشجوی دکتری مهندسی کامپیوتر، پردیس فنی و مهندسی، گروه مهندسی کامپیوتر، دانشگاه یزد f.alamiyan@stu.yazd.ac.ir

[۲] دانشیار، پردیس فنی و مهندسی، گروه مهندسی کامپیوتر، دانشگاه یزد vderhami@yazd.ac.ir

(تاریخ دریافت مقاله ۱۳۹۶/۱/۹، تاریخ پذیرش مقاله ۱۳۹۶/۵/۲۵)

**چکیده:** این مقاله چارچوبی از یادگیری عمیقِ با ناظر را جهت ناوبری ربات چرخ‌دار در زمین‌های هموار با محوریت وظایف پیگیری دیوار و اجتناب از موانع ارائه می‌نماید. در اینجا، فرض بر این است که ربات تنها به یک سیستم بینایی (دوربین کینکت) مجهز است. چالش اصلی در هنگام استفاده از تصاویر عمق، ابعاد بالای تصاویر و استخراج ویژگی‌های مناسب از آنها با هدف کاهش ابعاد ورودی کنترلگر می‌باشد. برای این منظور در این مقاله از یادگیری عمیق بهره‌برداری شده و ویژگی‌های مناسبی بدست می‌آیند که بازنمایی تصاویر عمق هستند. چهار معماری با استفاده از این ویژگی‌ها و سابقه‌های فرمان کنترلی برای کنترلگر ارائه می‌شود. این معماری‌ها در محیط شبیه‌ساز ویبات با یکدیگر مقایسه می‌شوند. آزمایش‌های انجام شده نشان می‌دهد معماری بهره برنده از چهار دسته ورودی شامل: ویژگی‌های بازنمایی شده از داده‌های عمق، ویژگی‌های لحظه‌ی قبل، موقعیت خط سیر در تصویر رنگی، و سابقه‌ی فرمان‌های پیشینِ کنترلگر می‌تواند به خوبی کنترل ربات را در محیط‌های هموار و با مانع به انجام برساند.

**کلمات کلیدی:** ناوبری ربات، یادگیری با ناظر، یادگیری عمیق ، داده‌های عمق.

# Feature Extraction from Depth Data using Deep Learning for Supervised Control of a Wheeled Robot

**Farinaz Alamiyan Harandi, Vali Derhami**

**Abstract:** This paper proposes a framework of Supervised Deep Learning (SDL) for wheeled robot navigation in soft terrains with a focus on wall following and obstacle avoidance tasks. Here, it is supposed the robot is only equipped with a vision system (Kinect camera). The main challenge while using depth images is high dimensionality of images and extracting proper features of them with a purpose of reducing input dimensionality of controller. To do this, the deep learning is utilized in this paper and the appropriate features which are the representation of depth images are acquired. Four architectures are created using this features and the history of steering commands. These architectures are compared in WEBOT simulator. The experiments show that the proposed architecture with four groups of features including: the represented features of depth data, previous represented features, the position of trajectory in color image, and the history of previous steering commands can control the robot in soft terrain with a variety of obstacles as well.

**Keywords:** Robot navigation, Supervised learning, Deep learning, Depth data.

[ Downloaded from joc-isice.ir on 2026-01-31 ]

[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

## ۱- مقدمه

مسئله‌ی ناوبری ربات یک مسئله‌ی چالشی در پژوهش‌های انجام شده در حوزه‌ی رباتیک است. ناوبری به معنای هدایت ربات در محیط عملیات است به گونه‌ای که ربات حرکت خود را از یک موقعیت اولیه آغاز کرده و مسیری را تا موقعیت هدف به صورت امن طی کند؛ منظور از ایمنی در مسیر حرکت این است که ربات بتواند موانع پیرامون خود را تشخیص داده و از آنها حذر کند[۱, ۲].

اگر مدل محیط و ربات، در دسترس کنترلگر ربات باشد و شرایط محیط متغیر نباشد می‌توان از روش‌های طرح‌ریزی مسیر استفاده کرد. در این روش‌ها، ابتدا با استفاده از اطلاعات مدل، مسیری بهینه برای حرکت ربات طراحی شده و سپس کنترلگر تلاش می‌کند هدایت ربات را طوری انجام دهد که خط سیر ربات منطبق با مسیر بهینه ایجاد شود. اگر مدل محیط، در دسترس نباشد (که این چالش اغلب در محیط عملیات موجود است)، ربات از اطلاعات محیطی مختلفی که توسط حسگرهای گوناگون جمع‌آوری می‌شوند، بهره‌برداری کرده و شرایط محیطی خود را از لحاظ موقعیتِ موانع احتمالی، موقعیت هدف، موقعیت فعلی ربات، یا خط‌سیر می‌سنجد؛ به این ترتیب می‌توان فاصله‌ی از موانع را محاسبه کرد و مسیر امن را برای حرکت ربات به صورت محلی اتخاذ نمود[۳, ۴].

رفتارهای گوناگونی که در ناوبری برای ربات مورد نظر است، گاه در محیط عملیات با یکدیگر تناقض دارند. برای مثال اگر وظیفه‌ی ربات در ناوبری، پیگیری هدف[۱] باشد، در این صورت پیمودن کوتاه‌ترین مسیر بین نقاط شروع و پایان، نتیجه‌ای مطلوب است. حال اگر در این مسیر موانعی موجود باشد، ربات ناچار است آن‌ها را دور زده و از مسیر بهینه دور شود. در چنین شرایطی لازم است معیارهای ارزیابی مناسبی برای رفتار ربات در محیط تعریف شود به گونه‌ای که این تناقض‌ها به خوبی مدیریت شده و کمترین تأثیر را در عملکرد ربات داشته باشند.

ربات‌ها برای درک محیط پیرامون خود به حسگرها و تجهیزات گوناگونی مجهز می‌شوند؛ نوع این حسگرها بسته به محیط عملیاتی ربات متفاوت خواهد بود. اینکه ربات، عملیاتی را در محیط‌های داخلی انجام دهد و یا در محیط‌های باز بیرونی فعالیت کند در انتخاب تجهیزات تأثیرگذار است. حسگر فراصوت[۲]، مادون قرمز، و پویشگرهای لیزری[۳] نمونه‌هایی از این حسگرها هستند که فاصله‌ی ربات را تا موانع مختلف در محیط می‌سنجند.

در محیط‌های داخلی، هدف اصلی ربات معمولاً دنبال کردن دیوارها و حرکت در کریدورهای ساختمان است. برای نمونه می‌توان به الگوریتم کنترلی طراحی شده توسط کارلی و فرایر[۴] اشاره کرد. این الگوریتم مبتنی بر اطلاعات حسگرهای مسافت‌سنجی[۵] و اندازه‌گیری‌های فاصله‌ای بوده و در آن از خروجی حسگرهای سونار برای حرکت ربات در طول کریدورها و دنبال کردن دیوارها بهره‌برداری می‌شود[۵]. شبکه‌عصبی احتمالی[۶] ساختار دیگری است که برای هدایت ربات با هدف دنبال کردن دیوار پیشنهاد شده و برای تأمین ورودی از حسگر فراصوت استفاده می‌نماید[۶]. حسگرهای فراصوت در مسائل دنبال کردن دیوار برای اندازه‌گیری و تعیین فاصله و زاویه‌ی حرکت یک ربات با استفاده از کنترلگر پلکانی فازی نیز بکار رفته‌اند[۱].

لازم به ذکر است برای استفاده ی بهتر از حسگرهایی همچون سونار و مادون قرمز، و پوشش‌دهی کامل اطراف ربات لازم است تعداد زیادی از این نوع حسگرها در اطراف ربات نصب شود. همچنین لازم است حتما دیواری موجود باشد تا بتوان از تکنولوژی این حسگرها که بر پایه‌ی برخورد و بازگشت انعکاس پرتوهای ارسالی استوار هستند استفاده نمود. استفاده از حسگرهای بینایی همچون دوربین کینکت و بهره‌برداری از اطلاعات عمق در محیط‌های هموار و با مانع، علاوه بر بهره‌مندی از حسگری ارزان قیمت و با میدان دید وسیع در هزینه‌ها صرفه‌جویی کرده و اطلاعات فراوانی را در اختیار کنترلگر قرار می‌دهد. این حسگر می‌تواند در مسئله‌های ناوبری با هدف دنبال‌کردن دیوارهای مجازی برای مثال یک خط رنگی نیز استفاده شوند.

برخی از الگوریتم‌های کنترلی از بینایی در ناوبری استفاده کرده‌اند. برای نمونه زو[۷] و همکاران، یک نقطه به نام نقطه‌ی اضمحلال[۸] را بر اساس خطوط نمایش دهنده‌ی کریدورها از تصویر دوربین استخراج کرده و از آن برای شناسایی جهت سر ربات در هدایت استفاده نموده‌اند. در این روش، محاسبات ریاضی مورد نیاز برای بدست آمدن نقاط اضمحلال، بسیار پیچیده است[۷]. یکی دیگر از کاربردهای دوربین برای کنترل موقعیت ربات نسبت به موقعیت هدف، پژوهشی است که در آن از بازخوردهای یک دوربین رنگی CCD استفاده می‌شود. ویژگی‌های بصری محیط پیرامون به ربات این امکان را می‌دهد که موقعیت و جهت حرکت خود را در محیط تشخیص داده و مسیر مناسبی را در پیش گیرد[۸].

تخمین بلادرنگ خط‌سیر ربات به صورت سه‌بُعدی و تنها بر اساس سیستم بینایی، کار دیگری است که با استفاده از یک دوربین استریوی[۹]

[۱] Goal Seeking

[۲] Ultrasonic Sensor

[۳] Laser Scanner

[۴] Carelli and Freire

[۵] Odometric Sensor

[۶] Probabilistic Neural Network (PNN)

[۷] Zhou

[۸] Vanishing Point

[۹] Stereo Camera



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

Trinocular در محیط‌های ناشناخته و بیرون از ساختمان انجام شده است. در این سیستم، ویژگی‌های مناسبی که با حرکت ربات در محیط مرتبط هستند، از تصاویر دوربین استخراج می‌شوند و تشخیص موقعیت ربات و مسیر حرکت آن با پیگیری تغییرات شکل گرفته در موقعیت این ویژگی‌ها انجام می‌گیرد[۹].

از حسگرهای بینایی برای تشخیص خط‌سیر و طراحی قواعد کنترل فرمانِ ربات با استفاده از معادلات سینماتیکی حرکت نیز استفاده شده است[۱۰, ۱۱]. در این تحقیقات از فاصله‌سنج لیزری[۱] برای پویش موقعیت ناهمواری‌های زمینِ پیشِ روی ربات استفاده شده و بررسی عبورپذیری محیط در فواصل دورتر با استفاده از دوربین استریو انجام می‌گردد. مسیر مناسب طرح‌ریزی شده و سپس قواعد کنترل برای هدایت محلی ربات تنظیم می‌شوند. این قواعد شامل کنترل سرعت طولی و عرضی ربات و نیز تنظیم زاویه‌ی سیستم بینایی برای کنترل میدانِ دید ربات است. در این سیستم‌ها، کنترلگر از اطلاعات مربوط به زمین که طرح‌ریز در اختیار آن قرار می‌دهد از جمله ضریب مقاومتِ نورد طولی[۲] و ضریب اصطکاک جانبی[۳] استفاده می‌کند و فضای خط‌سیر با تحلیل دینامیکی ربات ارائه می‌شود.

چالش اصلی در کنترل ربات به کمک داده‌های دوربین، بالا بودن ابعاد تصاویر، تعیین تعداد تصاویر عمق به عنوان حافظه‌ی دیداری کنترلگر ربات، و چگونگی استخراج ویژگی‌ها از داده‌های عمق به عنوان ورودی کنترلگر ربات است به طوری‌که محیط پیرامون ربات به کمک ترکیب خطی و وزن‌دار این ورودی‌ها شناسایی شده و فرمان کنترلی مناسب برای هدایت ربات تولید شود. در بسیاری از پژوهش‌ها، مدل محیط از پیش مشخص بوده و یا برای شناسایی محیط علاوه بر دوربین از حسگرهای دیگری نیز استفاده می‌شود. تصاویر رنگی و یا داده‌های عمق پیش از ورود به کنترلگر ربات پیش‌پردازش شده و ویژگی‌های مورد نظر طراح از آنها استخراج می‌گردد. مفهوم یادگیری برای بهره‌برداری از اطلاعات محلی به هوشمندی کنترلگر کمک می‌کند و می‌تواند مشکلات روش‌های پیشین را برطرف سازد. برای مثال روش‌هایی که از الگوریتم‌های هندسی و محاسبات فاصله‌ای استفاده می‌کنند و فرض‌هایی مانند امکان تشخیص کامل و برخط موانع را در مسئله وارد می‌نمایند؛ فرض‌هایی که در محیط‌های واقعی غیر قابل قبول و ایجاد آنها دشوار هستند.

دوربین کینکت[۴]، یکی از سیستم‌های بینایی است که برای بدست آوردن ابر نقطه‌ای سه‌بُعدی به عنوان داده‌های توصیف‌کننده‌ی محیط بکار می‌رود[۱۲]. این مدل‌های سه‌بُعدی بیشتر در الگوریتم‌های SLAM[۵] ایجاد می‌شوند. هدف این الگوریتم‌ها، ایجاد نقشه‌ی محیط از طریق تعامل ربات با محیط و استفاده‌ی همزمان از این نقشه‌ها برای هدایت ربات در محیط است. این ابرهای نقطه‌ای سه‌بُعدی می‌توانند در یک صفحه‌ی دوبُعدی تصویر شده و به این ترتیب از داده‌های کینکت در الگوریتم‌های SLAM دوبُعدی نیز استفاده نمایند. پژوهش‌هایی که در این زمینه انجام شده‌اند، نشان می‌دهند که دوربین کینکت به عنوان یک حسگر برای ناوبری ربات‌ها و در کاربردهای SLAM، یک گزینه‌ی مناسب و پایدار است. از مزایای این حسگر، قیمت پایین و فراهم‌سازی همزمان تصاویر رنگی و نقشه‌های عمق است.

نمونه‌ای دیگر از پژوهش‌هایی که در آن از دوربین کینکت استفاده شده است، یک سیستم حسی را برای ربات‌های متحرک نظارتی پیشنهاد می‌کند[۱۳]. این سیستم از دو بخش اصلی تشکیل شده و امکان ناوبری خودمختار ربات را فراهم می‌کند. بخش اول یک سیستم ناوبری واکنشی است که از داده‌های دوربین کینکت استفاده کرده و امکان حرکت ربات در محیط و پرهیز از برخورد با موانع را فراهم می‌کند. در این بخش، هر تصویر دریافتی به ۵ بلوک عمودی شکسته شده و سه بلوک که شامل بلوک‌های سمت چپ، میانی، و سمت راست است از لحاظ مقادیر پیکسلی تحلیل می‌شوند؛ سپس بیشترین و کمترین فاصله بین حسگر و موانع محاسبه می‌گردد. در این سیستم ۸ موقعیت مختلف بررسی شده و متناظر با هر یک فرمان کنترلی مناسبی برای ربات اتخاذ می‌شود. در بخش دوم، یک سیستم طبقه‌بند توسط داده‌های جمع‌آوری شده از دوربین کینکت، آموزش دیده و امکان تشخیص این موقعیت‌ها را از طریق تصاویر ایجاد می‌کند.

یادگیری با ناظر، یکی از اولین الگوریتم‌های یادگیری است. این روش می‌تواند در تنظیم پارامترهای یک کنترلگر راه‌گشا باشد[۱۴]. این یادگیری از داده‌های آموزش استفاده کرده و پارامترها را با بهره‌برداری از روش‌های مبتنی بر گرادیانت با کاهش جمع مربعات خطا در خروجی‌ها تنظیم می‌کند[۱۵]. در این نوع یادگیری، چالش‌های قابل‌توجهی وجود دارد؛ برای مثال جمع‌آوری داده‌های آموزشی در بعضی مسائل با دشواری‌هایی روبه‌رو است. ناسازگاری در داده‌ها، وجود داده‌های نویزی و خطاهای زیاد در آنها بسیار چالش‌برانگیز هستند و گاه کیفیت کنترلگر را به شدت تحت تأثیر قرار می‌دهند[۳].

در سال‌های اخیر، پژوهش‌های ارزشمندی در زمینه‌ی یادگیری عمیق انجام شده است. از جمله‌ی این پژوهش‌ها می‌توان به روش‌های مطرح‌شده در الگوریتم‌های یادگیری NFQ[۶][۱۶]، DQN[۷][۱۷]، و DDPG[۸][۱۸] اشاره کرد. در تمام این الگوریتم‌ها از ساختار شبکه‌عصبی عمیق به عنوان یک تقریب‌زننده‌ی تابع برای بازنمایی

[۱] Laser Range Finder
[۲] Coefficient of Longitudinal Rolling Resistance
[۳] Coefficient of Lateral Friction
[۴] Kinect Camera
[۵] Simultaneous Localization and Mapping
[۶] Neural Fitted Q-iteration
[۷] Deep Q-Network
[۸] Deep Deterministic Policy Gradient



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

داده‌ها به شکل‌های مختلف استفاده شده است اما مسائلی همچون زمان‌بر بودن تنظیم بردارهای پارامترها با ابعاد بالا به صورت یکجا وجود دارد و تلاش شده است روش‌های برای کاهش تعداد نمونه‌های آموزش در عین حفظ کیفیت آموزش ارائه گردد.

نمونه‌ی دیگر از کاربرد یادگیری عمیق در یک راهکار ردگیری اشیاء معرفی شده است[۱۹]. در این راهکار ورودی خام حسگر در فضای ورودی مستقیماً به ردگیری اشیاء نگاشت می‌شود بدون آن‌که هیچ عملیاتی در راستای مهندسی ویژگی‌ها و شناسایی سیستم با استفاده از مدل‌های حسگر یا بستر کار مورد نیاز باشد. سیستم پیشنهادی با دریافت داده‌های خام حسگر به عنوان ورودی به صورت بلادرنگ تخمینی از حالت کل محیط را در خروجی تولید می‌کند. برای این منظور موضوع به صورت یک مسئله‌ی یادگیری عمیق تعریف شده و مدل‌های توالی در قالب شبکه‌های عصبی بازگشتی[۱] برای یادگیری نگاشت مابین اندازه‌گیری‌های حسگرهای ورودی و ردگیری‌های اشیاء استخراج می‌شوند. . شبکه عصبی کانولوشن تحت آموزش در این مسئله ۱۱۰۰۰ پارامترها دارد. یادگیری پارامترهای این شبکه به هر دو روش با ناظر و بدون ناظر انجام شده است. برای یادگیری بدون ناظر، یک روش یادگیری به شکل حذف ورودی[۲] پیشنهاد شده است.

ترکیب انواع روش‌های یادگیری می‌تواند قابلیت‌های کنترلگر ربات را بهبود بخشد و عملکرد آن را افزایش دهد. یکی از این ترکیب‌های سودمند می‌تواند در قالب بازنمایی داده‌ها بین یادگیری با ناظر و یادگیری عمیق صورت گیرد که در موفقیت الگوریتم‌های یادگیری تأثیرگذار است؛ در واقع بازنمایی‌های مختلف با کم و زیاد کردن فاکتورهای توصیفیِ گوناگون، تنوع مستتر در داده‌ها را از ورودی‌های این الگوریتم‌ها حذف کرده یا در آنها می‌گنجانند. به همین دلیل یادگیریِ روشی برای بازنمایی داده‌ها، امکان استخراج اطلاعات مفید را تسهیل کرده و به عنوان یک عملیات خاص در ساخت طبقه‌بندها و ساختارهای پیش‌بینی‌کننده مطرح می‌شوند[۲۰].

این مقاله چارچوبی را برای ترکیب یادگیری عمیق و یادگیری با ناظر در طراحی کنترلگرها با هدف ناوبری ربات‌های چرخ‌دار در زمین‌های هموار با محوریت چالش‌های پیگیری دیوار و اجتناب از موانع، مورد مطالعه قرار می‌دهد. فرض بر این است که ربات تنها به یک سیستم بینایی (دوربین کینکت) مجهز است. استفاده از یادگیری عمیق برای بازنمایی ویژگی‌های توصیف‌کننده‌ی محیط و ترکیب آن با یادگیری با ناظر به منظور آموزش پارامترهای یک کنترلگر با معماری شبکه‌عصبی از دستاوردهای این مقاله است. استفاده از ترکیب خطی ویژگی‌های بازنمایی‌شده به همراه انتخاب ویژگی‌های خاص یک وظیفه (در این پژوهش ناوبری، اجتناب از مانع و دنبال کردن دیوار) به طوری که عملکرد کنترلگر بهبود داده شود، و کاهش تعداد پارامترهای تنظیم‌پذیر کنترلگر از دیگر اهداف این پژوهش می‌باشد. به این ترتیب مسائلی همچون زمان‌بر بودن تنظیم تمام پارامترها در کنترلگرهایی از نوع شبکه‌های عصبی به صورت یکجا، مشکل آموزش نامناسب وزن‌های لایه‌های ابتدایی شبکه‌عصبی و یا مشکلات موجود در عمل گسسته‌سازی فضاهای ورودی و از دست رفتن اطلاعات مفید که در کارهای گذشته وجود دارد برطرف می‌شود.

ساختار مقاله به شرح زیر است: ابتدا در بخش ۲ کارهای گذشته مرور می‌شوند، در بخش ۳ چارچوب پیشنهادی و ایده‌ها شرح داده می‌شوند. بخش ۴ آزمایش‌ها و مقایسه‌ی نتایج را در بر می‌گیرد و در نهایت، نتیجه‌گیری و کارهای آتی در بخش ۵ ذکر می‌شوند.

## ۲- یادگیری عمیق

یادگیری عمیق عبارت است از ساخت مدل‌های یادگیری ماشین که برای یادگیری نمایشی سلسله‌مراتبی از داده‌ها به کار می‌روند. شبکه‌های عصبی عمیق، اصطلاحی کلی برای مجموعه‌ای از شبکه‌های عصبی با معماری چند لایه است که نشان می‌دهند چگونه شبکه‌های عصبی با تعداد زیادی از لایه‌ها می‌توانند در ایجاد ساختارهای بازنمایی مورد نیاز در یادگیری عمیق موفق عمل کنند. الگوریتم‌های یادگیریِ ویژگی به صورت با ناظر و بدون ناظر می‌توانند در تنظیم وزن‌های این شبکه‌ها به کار گرفته شوند.

در واقع انواع گوناگونی از شبکه‌های عصبی عمیق موجودند از جمله: شبکه‌های کدگذار خودکار[۳]، شبکه‌های عصبی کانوولوشن[۴]، شبکه‌های عصبی عودکننده[۵] ، و شبکه‌های عصبی بازگشتی[۶]. ترکیب‌های مختلفی از این شبکه‌ها برای مثال ترکیب شبکه‌های عصبی کانوولوشن و شبکه‌های عصبی عودکننده نیز می‌توانند با توجه به نوع فضای ورودی در مسئله‌ی تحت بررسی بکار گرفته شوند[۲۱]. این شبکه‌ها با ترکیب چندین تبدیل غیرخطی شکل می‌گیرند که هدف آنها دستیابی به انتزاع بیشتر و در نهایت بازنمایی‌های سودمند از داده‌های موجود است[۲۰]. هر چه تعداد لایه‌ها در شبکه‌عصبی بیشتر شود، مسئله‌ی بهینه‌سازی پیچیده‌تر می‌شود. به همین دلیل یکی از روش‌های آموزش این شبکه‌ها با استفاده از الگوریتم‌های پیش‌آموزشی بدون ناظر و لایه‌ای حریصانه[۷] انجام می‌گیرد[۲۲-۲۵]. در این روش ابتدا هر لایه به صورت مجزا آموزش داده می‌شود و در نهایت روی کل شبکه تنظیم دقیق و یکپارچه‌ای انجام می‌گیرد[۲۳].

[۱] Recurrent Neural Network

[۲] Input Dropout

[۳] Auto-Encoder

[۴] Convolutional Neural Networks

[۵] Recurrent Neural Networks

[۶] Recursive Neural Networks

[۷] Greedy Layer-wise Unsupervised Pre-training Algorithms



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

در ساده‌ترین حالت، یک کدگذار خودکار به صورت یک شبکه‌عصبی ۳ لایه ظاهر می‌شود که در آن یک لایه‌ی ورودی، یک لایه‌ی پنهان، و یک لایه‌ی خروجی وجود دارد. اگر در لایه‌ی ورودی و خروجی $p$ نرون و در لایه‌ی پنهان $f: \mathbb{R} \longrightarrow \mathbb{R}$ تابع و باشد موجود نرون $q$ یک تابع انتقال مانند Sigmoid باشد و $x \in$ $i$ از آمده بدست ویژگی مقدار آنگاه باشد، ورودی لایه‌ی از برداری $\mathbb{R}^p$ اُمین نرون در لایه‌ی پنهان برای تمام مقادیر $i = 1, \dots, q$ $h_i(x) := f(w_i^T x + b_i)$ صورت به محاسبه می‌گردد؛ به‌طوری‌که $b_i \in$ و $w_i \in \mathbb{R}^p$ $i$ با متناظر بایاس مقدار و وزن بردار ترتیب به $\mathbb{R}$اُمین نرون هستند[۲۶].

پس از محاسبه‌ی ویژگی‌ها در لایه‌ی پنهان به صورت $h := \left[h_1(x), h_2(x), \dots, h_q(x)\right]^T \in \mathbb{R}^q$، بازنمایی ایجاد شده در این لایه، به عنوان ورودی لایه‌ی خروجی بکار گرفته می‌شود و خروجی حاصل از لایه‌ی آخر شبکه‌عصبی برای تمام مقادیر $j = 1, \dots, p$ صورت به $y_i(h) := f'(w_j'^T h + b_j')$ ایجاد می‌شود. در این محاسبه $w_j' \in \mathbb{R}^q$ و $b_j' \in$ $j$ با متناظر بایاس و وزن پارامترهای ترتیب به $\mathbb{R}$اُمین نرون در لایه‌ی خروجی هستند. در این فرآیند $W := \left[w_1, w_2, \dots, w_q\right]^T \in \mathbb{R}^{p \times q}$ و $W' := \left[w_1', w_2', \dots, w_p'\right]^T \in \mathbb{R}^{q \times p}$ ماتریس‌های وزن‌ها در کل شبکه هستند و توابع انتقال $f$ و $f'$ الزاماً یکسان نیستند. اگر ماتریس وزن $W'$ با ماتریس وزن $W$ مساوی باشد یعنی $W' = W^T$ باشد، عملکرد یک کدگذار خودکار با رفتار یک ماشین بولتزمن محدودشده[۱] قابل مقایسه می‌گردد[۲۶].

خطای موجود مابین بردار ورودی $x$ و بازنمایی انجام شده‌ی $y : = \left[y_1(h), \dots, y_p(h)\right]^T \in \mathbb{R}^p$ می‌تواند به عنوان معیاری برای کیفیت بازنمایی انجام شده در لایه‌ی پنهان $h$ در نظر گرفته شده و یک تابع هزینه برای الگوریتم انتشار رو به عقب[۲] خطا باشد[۲۷].

برای ورودی‌های تصویری مرسوم‌ترین تابع هزینه، میانگین مربعات خطا (MSE[۳]) است. به این ترتیب تابع هزینه‌ی نهایی به صورت $L(x, y, W, W') = \frac{1}{2}\|x - y\|_2^2 + \frac{\lambda_1}{2}\|W\|_F^2 + \frac{\lambda_2}{2}\|W'\|_F^2$ خواهد بود، به طوری‌که $\|.\|_F$ نُرم فوربنویس[۴] ماتریس‌هاست[۲۸]. در اینجا تابع هزینه از دو بخش تشکیل می‌شود.

[۱] Restricted Boltzmann Machine

[۲] Backpropagation Algorithm

[۳] Mean Squared Error

[۴] Frobenius Norm

یک جزء خطای باقی‌مانده/بازنمایی[۵] را نشان می‌دهد و جزء دیگر مربوط به مرحله‌ی تنظیم[۶] یا همان جزء تضعیف وزن است. پارامترهای تضعیف وزن $\lambda_i$ مقادیری مثبت بوده و اندازه‌ی وزن‌ها را در لایه‌های مختلف برای جلوگیری از بیش‌برازش[۷] کدگذار خودکار کنترل می‌کنند.

برای گسترش اصل موجود در کدگذارهای خودکار در روش‌های یادگیری عمیق، ساختار فوق بسط داده شده و بازنمایی حاصل از لایه‌ی پنهان در اولین کدگذار خودکار به عنوان ورودی دومین کدگذار خودکار به خدمت گرفته می‌شود و این روند می‌تواند برای افزایش لایه‌های شبکه‌عصبی عمیق ادامه داده شود. در آموزش شبکه‌ی حاصل نیز می‌توان از روشی که در بالا بیان شد، بهره‌برداری کرد؛ به این ترتیب که اولین لایه‌ی پنهان به روش بیان شده آموزش دیده و پس از ایجاد همگرایی، وزن‌های ورودی‌های لایه‌ی پنهان تثبیت می‌شوند و لایه‌ی پنهان بعدی به عنوان یک کدگذار خودکار به روش مشابهی آموزش می‌بیند و این روند تا تکمیل معماری کل شبکه ادامه می‌یابد. در نهایت مرحله‌ی تنظیم روی کل ساختار شبکه به صورت یکپارچه انجام می‌گیرد[۲۶].

## ۳- روش پیشنهادی

برای ایجاد کنترلگر پیشنهادی لازم است چارچوب ارائه‌شده در شکل ۱ پیگیری شود؛ برای تنظیم این کنترلگر از روش یادگیری عمیق با ناظر (SDL[۸]) پیروی می‌شود. در مرحله‌ی نخست، ساختاری برای بازنمایی حالت محیطِ پیرامون ربات به عنوان ورودی اصلی کنترلگر پیشنهاد می‌گردد. این بازنمایی به کمک یک شبکه‌عصبی پیش‌رو با ساختاری متقارن به عنوان یک تقریب‌زننده‌ی تابع[۹] انجام شده و وزن‌های شبکه با استفاده از یادگیری عمیق و داده‌های آموزش تنظیم می‌شوند.

معماری شبکه‌عصبی به دو بخش تقسیم شده و تعداد نرون‌های لایه‌های پنهان در نیمه‌ی اول، پس از طی کردن روندی کاهشی در لایه‌ی میانی[۱۰] به حداقل تعداد مورد نظر طراح تقلیل می‌یابد. عمل انتساب نرون‌ها در نیمه دوم شبکه به صورت معکوس، روندی افزایشی یافته و این افزایش تا رسیدن به تعداد اولیه‌ی نرون‌ها در آخرین لایه ادامه داده می‌شود. انتظار این است که ورودی این شبکه با گذر از لایه‌ها به گونه‌ای تغییر یابد که در لایه‌ی خروجی دوباره به حالت اولیه بازنمایی شود. این شبکه در یادگیری با ناظر با داده‌های جمع‌آوری‌شده،

[۵] Residual/Reconstruction Error

[۶] Regularization

[۷] Overfitting

[۸] Supervised Deep Learning

[۹] Function Approximator

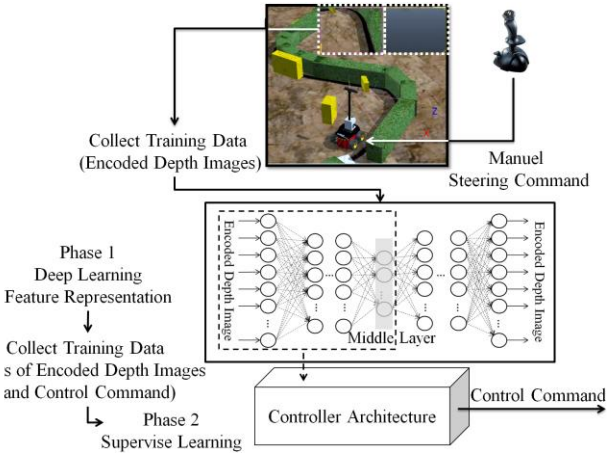
[۱۰] Middle Layer



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

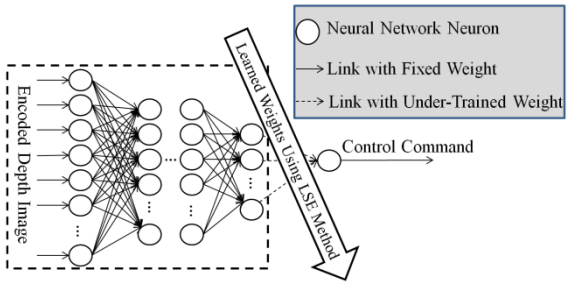
آموزش دیده و در تنظیم وزن‌های آن از قاعده‌ی انتشار رو به عقب خطا[۱] بهره‌برداری می‌شود. هر چه میزان خطا در آموزش این شبکه کمتر باشد، یادگیری بهتر انجام شده و بازنمایی داده‌های ورودی بهتر صورت پذیرفته است.

پس از پایان مرحله‌ی بازنمایی ویژگی‌های ورودی، لایه‌های پنهان موجود در نیمه‌ی ابتدایی شبکه‌عصبی و لایه‌ی میانی با حفظ مقادیر تنظیم‌شده در وزن‌ها، به عنوان بخشی از معماری کنترلگر محلی ربات مجدداً بکار گرفته می‌شوند. در ادامه، چهار معماری برای طراحی ساختار کنترلگر ربات پیشنهاد می‌شوند که این بازنمایی در آنها نقشی کلیدی داشته و می‌تواند در کنار سایر ویژگی‌ها در توصیف محیط پیرامون ربات مفید واقع شود.



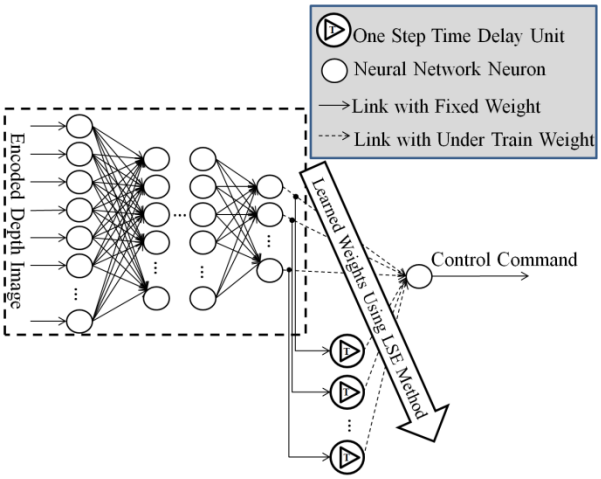

شکل ۱: چارچوب کلی روش پیشنهادی

در معماری‌های پیشنهادی، خروجی هر یک از نرون‌های موجود در لایه‌ی پنهان میانی از شبکه‌ی بازنمایی ویژگی‌ها، به عنوان ورودی به یک نرون با تابع انتقال خطی اتصال یافته و معماری شبکه با ایجاد مقداری حقیقی و پیوسته در خروجی کامل می‌گردد. مقدار وزنِ پیوندهایی که در این مرحله ایجاد شده‌اند، با استفاده از داده‌های آموزشی و به روش تک عبوره‌ی حداقل مربعات خطا[۲] خطی آموزش داده می‌شوند. در معماری اول که آن را کنترلگر تک تصویری با روش یادگیری عمیقِ با ناظر (SI-SDL[۳]) می‌نامیم، از نقشه‌ی عمق فعلی به عنوان تنها ورودی کنترلگر استفاده می‌شود. شِمایی از این کنترلگر درشکل ۲ آورده شده است.




شکل ۲: کنترلگر SI-SDL

در معماری دوم که آن را کنترلگر دو تصویری با روش یادگیری عمیقِ با ناظر (DI-SDL[۴]) می‌نامیم، از نقشه‌ی عمق فعلی و قبلی به صورت همزمان استفاده شده و فرمان خروجی بر اساس این دو ورودی تعیین می‌گردد. هر یک از تصاویر به صورت جداگانه از ساختار بازنمایی ویژگی‌ها عبور داده شده و سپس ساختار کنترلگر با تعبیه یک نرون خطی در لایه‌ی خروجی شبکه تکمیل می‌گردد. در این معماری تصویر قبلی به عنوان یک حافظه از وضعیت پیشین محیط در هدایت ربات تأثیرگذار است. شِمایی از این کنترلگر در شکل ۳ نشان داده شده است.




شکل ۳: کنترلگر DI-SDL

در معماری سوم، برای هر یک از تصویرهای ورودی، دو ویژگی که معرف موقعیت خط سیر مورد نظر ربات است به ساختار معماری دوم اضافه شده و وزن‌های آخرین لایه از شبکه مشابه معماری‌های پیشین به روش یادگیری با ناظر تنظیم می‌شوند. این معماری، کنترلگر دو تصویری به همراه ویژگی‌های مستقیم با روش یادگیری عمیقِ با ناظر (DIDF-SDL[۵]) نامیده می‌شود. شِمایی از این کنترلگر در شکل ۴ مشهود است.

[۱] Error Backpropagation Rule

[۲] Least Square Errors

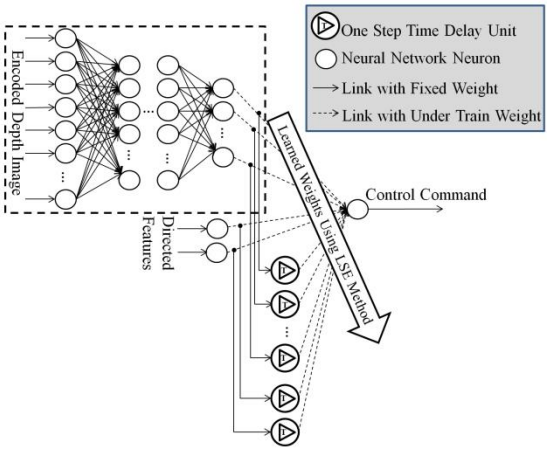
[۳] Single Image - Supervised Deep Learning

[۴] Double Image - Supervised Deep Learning

[۵] Double Image with Directed Features - Supervised Deep Learning

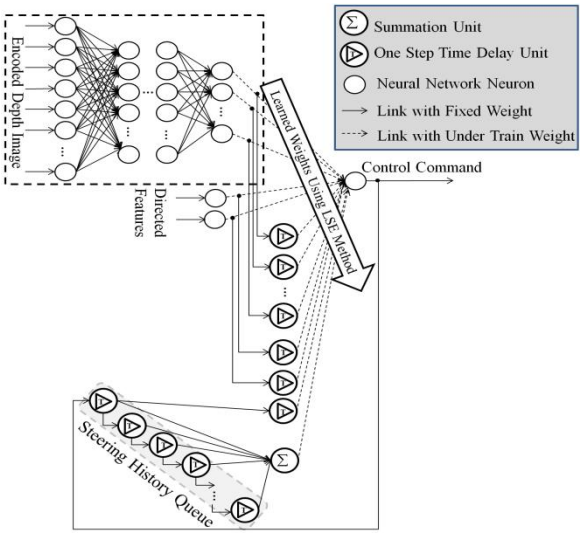


[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]




شکل ۴: کنترلگر DIDF-SDL

در معماری چهارم، دو ویژگی بر اساس فرمان‌های کنترلیِ پیشین تعریف شده و به عنوان حافظه‌ای از اَعمال ربات به ورودی‌های کنترلگر در ساختار معماری سوم اضافه می‌شوند. به این منظور از یک صف با طولی برابر با تعداد فرمان‌های اخیرِ مورد نظر بهره‌برداری می‌شود و مقادیر فرمان‌های اخیرِ کنترلگر در آن ذخیره و به مرور بروز رسانی می‌شوند.




شکل ۵: کنترلگر DIDFSH –SDL

مقدار آخرین فرمان و حاصلِ جمع فرمان‌های اخیر به عنوان ویژگی‌های رفتاری ربات به ورودی‌های لایه‌ی آخر شبکه افزوده می‌شوند. وزن‌های آخرین لایه از شبکه مشابه معماری‌های پیشین به روش یادگیری با ناظر تنظیم می‌شوند. این معماری، کنترلگر دو تصویری به همراه ویژگی‌های مستقیم و سابقه‌ی فرمان با روش یادگیری عمیقِ با ناظر (DIDFSH -SDL[1]) نامیده می‌شود. شِمایی از این کنترلگر در شکل ۵ قابل مشاهده است.

مراحل چارچوب ارائه شده را می‌توان به صورت زیر خلاصه نمود:

۱) هدایت دستی ربات در محیط آموزش با هدف جمع‌آوری داده‌ها در دو مجموعه‌داده به صورت تصویرهای ورودی کنترلگر و فرمان‌های خروجی آن.

۲) آماده‌سازی تصاویر شامل تبدیل تصاویر به نقشه‌های بلوکی و انتساب متغیرهای آماری میانگین و واریانس که در ادامه به آن پرداخته خواهد شد.

۳) استفاده از ساختار شبکه‌عصبی عمیق برای بازنمایی حالت محیطِ پیرامون ربات و آموزش این ساختار به کمک داده‌های آموزش جمع‌آوری‌شده در گام ۱.

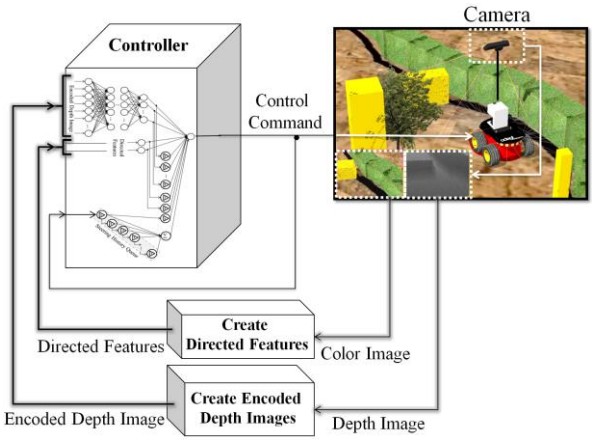
۴) استفاده از لایه‌های پنهان موجود در نیمه‌ی ابتدایی شبکه‌عصبی عمیق و لایه‌ی میانی با حفظ مقادیر تنظیم‌شده‌ی وزن‌ها در گام ۳، به عنوان ارائه دهنده قسمتی از ویژگی های حالت محیط.

۵) افزودن چند ویژگی‌های خاص و متناسب با وظیفه‌ی ربات (دنبال‌کردن دیوار و اجتناب از موانع)به ویژگی‌های قبلی بدست آمده در بند ۴.

۶) اضافه کردن یک لایه‌ی جدید بعد از لایه‌ی ویژگی‌های تولید شده در بند ۵، با تابع انتقال خطی که خروجی آن فرمان کنترلی ربات است.

۷) آموزش وزن‌های لایه‌ی آخر شبکه‌عصبی ایجاد شده در گام ۶ به کمک داده‌های جمع‌آوری شده در گام ۱ و استفاده از روش یادگیری با ناظر.

شکل ۶ بلوک دیاگرام کنترلی را با توجه به معماری چهارم برای روش پیشنهادی نشان می‌دهد:




شکل ۶: بلوک دیاگرام کنترلی روش پیشنهادی با توجه به معماری چهارم روش پیشنهادی

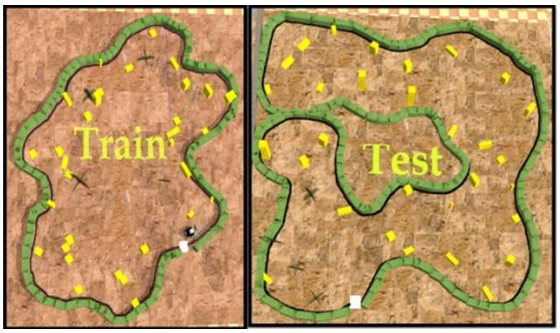
## شبیه‌سازی و نتایج آزمایش‌ها

[1] Double Image with Directed Features and Steering History - Supervised Deep Learning



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

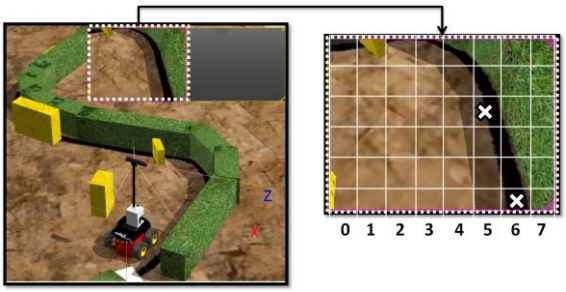
برای ارزیابی معماری‌های استفاده‌شده در مسئله‌ی پیگیری دیوار و اجتناب از موانع، از بستر نرم‌افزار متلب و شبیه‌ساز ویبات به صورت همزمان بهره‌برداری شده است. محیط‌های آموزش و تست در محیط شبیه‌ساز ویبات ایجاد شده‌اند و یک ربات چهارچرخ به نام Pioneer3_AT که تنها به یک سیستم بینایی (دوربین کینکت) مجهز شده است، تحت آزمایش قرار گرفته است. این رباتِ همه‌منظوره دارای قابلیت ناوبری در فضاهای باز است و در نمونه‌سازی[1] و تحقیقات پژوهشی رفتارهایی شامل ناوبری، نگاشت، نظارت، و شناسایی کاربرد دارد. این ربات ۵۰۸ میلی‌متر طول، ۴۹۷ میلی‌متر عرض، و ۲۷۷ میلی‌متر ارتفاع دارد و وزن آن ۱۲ کیلوگرم است. بیشینه‌ی سرعت این ربات در حرکت روبه‌جلو و روبه‌عقب برابر با ۰/۷ متر بر ثانیه و سرعت چرخشی آن ۱۴۰ درجه بر ثانیه است. اطلاعات بیشتر در مورد مشخصات و تجهیزات این ربات در وب‌سایت رسمی Adept Mobile Robots[2] موجود است. کنترلگر ربات با استفاده از نرم‌افزار متلب و به روش پیشنهادی آموزش داده شده و هدایت ربات را بر عهده دارد. محیط آموزش و آزمون مورد استفاده در آزمایش‌ها در شکل ۷ قابل مشاهده هستند. همان‌طور که در این شکل قابل مشاهده است، محیط‌های تحت آزمایش بسیار پیچیده است و موانع مقعر نیز در آنها موجود است.




شکل ۷: محیط آموزش و آزمون مورد استفاده در شبیه‌ساز ویبات

تصاویر عمق بدست آمده از دوربین کینکت به عنوان داده‌ی ورودی کنترلگر استفاده می‌شوند. این دوربین دارای قدرت تفکیک ۳۲۰ در ۲۴۰ پیکسل برای تصاویر عمق است. میدان دید دوربین کینکت در راستای افقی و عمودی به ترتیب ۵۸/۵ و ۴۶/۶ درجه می‌باشد و به این ترتیب می‌تواند برای پوشش هر درجه از میدان دید خود به طور متوسط مربعی با ابعاد ۵ در ۵ پیکسل را لحاظ کند. برای آماده‌سازی ورودی‌های کنترلگر پیشنهادی، هر تصویر عمقِ دوربین به یک نقشه‌ی بلوکی تبدیل می‌گردد. هر بلوک ۴۰ پیکسل را در هر یک از راستاهای افقی و عمودی تصویر در برمی‌گیرد؛ و میانگین و واریانسِ عمق تمام پیکسل‌های متعلق به آن بلوک به عنوان مقادیر معرف بلوک در نقشه‌ی بلوکی درج می‌شوند. این مقادیر با آرایشی خطی و با حفظ ترتیب سطری نقشه‌ی بلوکی (یعنی برداری حاوی ۹۶ مقدار عددی) به عنوان ورودی کنترلگر بکار می‌روند.

در کنترلگر SI-SDL تعداد داده‌های ورودی برای شبکه‌ی بازنمایی ویژگی‌ها ۹۶ مقدار حقیقی بوده که در کنترلگر DI-SDL این مقدار به ۱۹۲ افزایش می‌یابد. در کنترلگر DIDF-SDL، موقعیت خط‌سیر ربات در سطرهای میانی و پایانی انطباق نقشه‌ی بلوکی با هر تصویر رنگی دوربین ( با در نظر گرفتن مشکی رنگ بودن خط سیر) مشخص شده و به ورودی‌های آخرین لایه از ساختار شبکه عصبیِ کنترلگر افزوده می‌شود. نمونه‌ای از محاسبه‌ی این موقعیت‌ها در یک تصویر رنگی در شکل ۸ نمایش داده شده است. در کنترلگر DIDFSH –SDL، طول صف ۵ در نظر گرفته شده و آخرین فرمان کنترلی صادر شده به همراه حاصلِ جمع ۵ فرمان اخیر به عنوان ورودی‌های معرف سابقه‌ی فرمان‌های کنترلی به ساختار کنترلگر اضافه می‌شوند.




شکل ۸: نمونه‌ای از محاسبه‌ی موقعیت‌های خط سیر ربات در تصویر رنگی دوربین کینکت

در آزمایش‌های طراحی‌شده دو رفتار برای ربات تحت بررسی قرار گرفته است: رفتار پیگیری دیوار و رفتار اجتناب از موانع. در اینجا هدف رفتارِ پیگیری دیوار، هدایت یک ربات برای حرکت به سمت یک دیوار و دنبال کردن آن به عنوان یک خط‌سیر است به گونه‌ای که دیوار همواره در سمت راست ربات قرار داشته باشد و ربات بدون برخورد با دیوار در فاصله‌ی مناسبی از آن حرکت کند. در حین حرکت ربات نباید بیش از اندازه به دیوار نزدیک شده و یا بیش از حد از آن دور شود. در رفتار اجتناب از مانع، ربات باید به گونه‌ای هدایت شود که ربات با موانع موجود در محیط برخورد نکرده و با حاشیه‌ای امن از کنار آنها عبور کند. در ترکیب این دو رفتار انتظار می‌رود ربات با حفظِ فاصله‌ی مناسب با دیوار از موانع موجود حذر کرده و در عین حال خط‌سیر خود را گم نکند. ربات باید بتواند با در نظر گرفتن فاصله‌ی موانع با دیوار تحت پیگیری، در صورت امکان، مسیر مابین دیوار و مانع را ترجیح دهد و در صورتی که این فاصله کافی نباشد با فاصله گرفتن از دیوار مانع را دور زده و بدون گم کردن خط‌سیر خود، مجدد به سمت دیوار حرکت کرده و آن را پیگیری نماید.

[1] Prototyping

[2] http://www.mobilerobots.com/ResearchRobots/P3AT.aspx



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

برای آموزش کنترلگر ربات به روش یادگیری با ناظر، ابتدا ربات به صورت دستی در محیط آموزش حرکت داده شده و داده‌های آموزش جمع‌آوری می‌شوند. داده‌های جمع‌آوری شده در مرحله‌ی هدایت دستی ربات شامل بیش از ۱۵۰۰ تصویر است و نرخ تصویر برداری برابر با ۲/۵ تصویر در هر ثانیه است. برای چارچوب پیشنهادی دو مجموعه داده مورد نیاز است. اولین مجموعه شامل تصاویر عمقی است که به صورت نقشه‌ی بلوکی جمع‌آوری می‌شوند. این نقشه‌ها برای آموزشِ شبکه‌عصبیِ عمیق در بازنمایی ویژگی‌ها مورد نیاز هستند. هر نقشه‌ی بلوکی در این مجموعه یک زوجِ ورودی — خروجی را تشکیل می‌دهد یا به عبارت ساده‌تر به عنوان ورودی و خروجی متناظر در آموزش شبکه مشارکت داده می‌شود. در جمع‌آوری این داده‌ها ربات در وضعیت‌های متفاوتی در محیط حرکت داده می‌شود تا انواع موقعیت‌هایی را که ممکن است در طول حرکت خود در محیط با آنها مواجه شود، ملاقات کند. برای مثال موقعیتی که ربات در تصویر عمق هیچ مانعی از جمله دیوار را مشاهده نمی‌کند یا موقعیت‌هایی که دیوار کاملاً در مقابل ربات قرار می‌گیرد. این موارد به هنگام هدایت دستی ربات در محیط که هدف آن جمع‌آوری فرمان‌های کنترلی صحیح است، کمتر رخ می‌دهند و تنها در شرایط پیش‌بینی‌نشده امکان وقوع دارند.

مجموعه‌ی دوم از داده‌های آموزشی شامل زوج‌هایی مرتبی از نقشه‌های عمق بلوکی به عنوان ورودی و فرمان کنترلی متناظر به عنوان خروجی کنترلگر است. هر فرمان خروجی مقداری متعلق به مجموعه‌ی {۲۰، ۰، ۲۰-} است و زاویه‌ی چرخش چرخ‌های ربات را در حین حرکت مشخص می‌نماید. این مجموعه داده در آموزش وزن‌های آخرین لایه از شبکه‌عصبی موجود در معماری کنترلگر بکار می‌رود.

ساختار شبکه عصبی مورد استفاده در مرحله‌ی بازنمایی ویژگی‌ها یک شبکه‌عصبی ۱۱ لایه را شامل می‌شود که تعداد نرون‌ها در لایه‌های ورودی و خروجی به ترتیب برای ایجاد کنترلگر تک تصویری و دو تصویری برابر با ۹۶ و ۱۹۲ می‌باشد؛ تعداد نرون‌ها در لایه‌های پنهان به ترتیب برابر با ۸۰، ۶۰، ۵۰، ۳۰، ۲۰، ۳۰، ۵۰، ۶۰، ۸۰ است. در نرون‌های موجود در لایه‌ی پنهان وسط (لایه‌ی پنجم با بیست نرون در ساختار فعلی)، که در ساختار کلی شکل ۱ با Middle Layer مشخص شده است، از تابع انتقال Logistic Sigmoid استفاده شده و تابع انتقال برای تمام نرون‌ها در سایر لایه‌های پنهان، Tangent Sigmoid است. کنترلگر نیز یک شبکه‌عصبی ۷ لایه است که لایه‌ی ورودی آن به ترتیب برای ایجاد کنترلگر تک‌تصویری و دو تصویری دارای ۹۶ و ۱۹۲ نرون می‌باشد و لایه‌های پنهان اول تا پنجم آن از شبکه‌عصبی آموزش‌دیده در مرحله‌ی بازنمایی ویژگی‌ها با وزن‌های تثبیت‌شده گرفته شده است. خروجی پنجمین لایه‌ی پنهان به همراه ویژگی‌های دیگری که در معماری‌های دوم تا چهارم معرفی شدند، به عنوان ورودی لایه‌ی خروجی شبکه استفاده شده و ترکیب خطی آنها به کمک یک نرون با تابع انتقال خطی در لایه‌ی خروجی محاسبه می‌گردد. در این شبکه‌عصبی تنها وزن‌های ورودی‌های لایه‌ی آخر به روش باناظر آموزش داده می‌شوند. خروجی کنترلگر یک مقدار حقیقی و پیوسته است. طول صفِ سابقه‌ی فرمان‌های کنترلی در کنترلگر DIDFSH-SDL، ۵ در نظر گرفته شده است.

از آنجا که نتایج آموزش شبکه‌عصبی به مقادیر اولیه‌ی وزن‌های شبکه و مقادیر بایاس نرون‌های پنهان وابسته است و محیط تحت آزمایش دارای ویژگی عدم قطعیت است، در هر یک از آزمایش‌ها، شبکه‌عصبی چندین بار آموزش داده شده و ارزیابی نتایج آزمایش به صورت میانگین نتایج ۵ اجرای مستقل سنجیده می‌شود. هر اجرا شامل یک مرحله‌ی آموزش و یک مرحله‌ی آزمون است. مقدار ریشه‌ی میانگین مربعات خطا و بیشینه‌ی خطا در مقادیر خروجی کنترلگر برای داده‌های آموزش در جدول ۱ قابل مشاهده است.

معیارهای ارزیابی برای بررسی نتایج هر یک از آزمایش‌های انجام شده و مقایسه‌ی کارایی کنترلگرهای پیشنهادی متناسب با اهداف ربات مورد نیاز است. در مسائل ناوبری ربات با اهداف پیگیری دیوار و اجتناب از موانع، نرمی خط‌سیر ربات و تطابق آن با تغییرات موقعیت دیوار، رها نشدن روند پیگیری دیوار، رعایت فاصله‌ی مناسب با دیوار و تعداد شکست‌های ربات به عنوان معیارهای ارزیابی مطرح هستند. هر چه تعداد شکست‌ها کمتر باشد و خط سیر ربات با تغییرات مسیر انطباق بیشتری داشته باشد، هدایت ربات مطلوب‌تر بوده و کنترلگر بهتر عمل نموده است.

جدول ۱: متوسط مقادیر خطا در آموزش شبکه‌عصبی در مرحله‌ی دوم

| مقادیر خطا | روش‌های پیشنهادی | | | |
|---|---|---|---|---|
| | SI-SDL | Dl-SDL | DIDF-SDL | DIDFSH-SDL |
| ریشه‌ی میانگین مربعات خطای شبکه‌عصبی | ۹/۰۵ | ۸/۷۰ | ۸/۶۵ | **۸/۱۸** |
| بیشینه‌ی خطا در خروجی | ۲۶/۰۲۹ | ۲۴/۸۷۵ | ۲۴/۸۷۳ | **۲۳/۱۷۰** |

در اینجا ۳ معیار در نظر گرفته شده و همگی به صورت شکست تعریف شده و هر چه مقدار آنها کمتر باشد، مطلوب‌تر است. شکست " برخورد با مانع" که با استفاده از یک حسگر لمسی[۱] به نام ضربه‌گیر[۲] تشخیص داده می‌شود و تعداد برخورد بدنه‌ی ربات با موانع موجود در محیط را نشان می‌دهد. در صورت وقوع این شکست ربات کمی به عقب هدایت شده و سپس زاویه‌ی حرکت خود را کمی تغییر می‌دهد. مقدار چرخشی به صورت تصادفی در نظر گرفته می‌شود. شکست "خیلی نزدیک به دیوار" و "خیلی دور از دیوار" به ترتیب وضعیت‌هایی را نشان می‌دهند که ربات بیش از اندازه به دیوار نزدیک شده و یا از آن فاصله گرفته است که در اولی احتمال برخورد افزایش

[۱] Touch Sensor

[۲] Bumper



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

یافته و در دومی امکان گم کردن مسیر در پیگیری دیوار محتمل‌تر می‌شود. اگر ربات در فاصله‌ای از دیوار حرکت کند که موقعیت خط‌سیر ربات در انطباق نقشه‌ی بلوکی با هر تصویر رنگی دوربین (برای مثال شکل ۸) در یکی از خانه‌های ۷ یا ۶ تشخیص داده شود (یعنی فاصله با دیوار در نقشه‌ی بلوکی به ترتیب یک یا دو بلوک با عرض ۴۰ پیکسل باشد)، ربات در فاصله مطلوب با دیوار قرار دارد.

در حین حرکت ربات، گاه وضعیت "عدم مشاهده‌ی دیوار" رخ می‌دهد. در این وضعیت امکان تشخیص دیوار یا همان مسیر مورد نظر در تصاویر رنگی دوربین وجود ندارد. این وضعیت می‌تواند اجتناب‌ناپذیر باشد به خصوص زمانی که مسیر حرکت مارپیچ و دارای تحدب‌های شدید باشد. در این شرایط احتمال عدم مشاهده دیوار در تصاویر دوربین زیاد است و الزاماً ربات مسیر حرکت را گم نکرده است. به منظور تفکیک دو وضعیتِ گم شدن مسیر و پیچ مسیر در آزمایش‌ها از یک حد آستانه استفاده شده است. فرض بر این است که در مسیرهای مارپیچ ربات پس از مواجه شدن با اولین وضعیت "عدم مشاهده‌ی دیوار" می‌تواند حداکثر در ۱۱۰ گام زمانی مجدد دیوار را در تصاویر دریافتی تشخیص دهد؛ به همین دلیل تا ۱۱۰ گام زمانی وضعیت "عدم مشاهده‌ی دیوار" منجر به دخالت هدایتگر دستی در کنترل ربات نمی‌شود. در صورت افزایش تعداد دفعات این وضعیت و عبور از حد آستانه هدایتگر دستی، کنترل ربات را بر عهده گرفته و ربات را به وضعیت مناسبی بازمی‌گرداند.

نتایج عملکرد هر یک از چهار معماری پیشنهادی برای کنترل ربات در جدول ۲ و ۳ قابل مشاهده است. این نتایج نشان می‌دهند که ویژگی‌های حافظه‌ای که در قالب وضعیت پیشین محیط و فرمان‌های کنترلی قبلی به ورودی‌های کنترلگر افزوده می‌شوند، در بهبود کارایی کنترلگر تأثیرگذار هستند و تعداد شکست‌ها را کاهش می‌دهند. بهره‌برداری از یادگیری عمیق در بازنمایی تصاویر عمق با کاهش تعداد ورودی‌های شبکه سرعت اجرا را افزایش داده و در عین حال مانع از کاهش کیفیت نتایج خروجی کنترلگر می‌گردد. ایجاد نقشه‌های بلوکی و انتساب متغیرهای آماری میانگین و واریانس به همراه این بازنمایی می‌توانند تغییرات اندک موجود در وضعیت تصاویر ورودی را حذف کرده و به گونه‌ای ورودی‌ها را خوشه‌بندی نموده و تعمیم‌پذیری کنترلگر را افزایش دهند.

جدول ۲: نتایج عملکرد معماری‌های پیشنهادی برای کنترلگر ربات در محیط آموزش

| معیار ارزیابی (بر حسب تعداد) | روش‌های پیشنهادی در محیط آموزش | | | |
|---|---|---|---|---|
| | SI-SDL | Dl-SDL | DIDF-SDL | DIDFSH-SDL |
| شکست "برخورد با مانع" | ۶۰/۶ | ۴۵/۲ | ۴۱/۸ | **۳٤/٤** |
| شکست "خیلی نزدیک به دیوار" | ۲۶ | ۱۱ | ۱۴/۲ | **٥/٦** |
| شکست "خیلی دور از دیوار" | ۴۷۹/۲ | ۴۳۸/۶ | ۳۳۳ | **۲۹٤** |

جدول ۳: نتایج عملکرد معماری‌های پیشنهادی برای کنترلگر ربات در محیط آزمون

| معیار ارزیابی (بر حسب تعداد) | روش‌های پیشنهادی در محیط آزمون | | | |
|---|---|---|---|---|
| | SI-SDL | Dl-SDL | DIDF-SDL | DIDFSH-SDL |
| شکست "برخورد با مانع" | ۸۵/۶ | ۶۳ | ۶۲/۸ | **۵۳/۲** |
| شکست "خیلی نزدیک به دیوار" | ۹۱/۴ | ۵۵ | ۵۵/۸ | **۵۰/٤** |
| شکست "خیلی دور از دیوار" | ۶۷۱ | ۶۳۹/۸ | ۵۸۰/۲ | **۵۷۳/۸** |

مطابق با نتایج جدول ۳، آخرین معماری پیشنهاد شده نسبت به اولین معماری بهبود چشمگیری داشته و نرخ تعداد شکست‌ها به ترتیب برای هر یک از معیارهای ارزیابی موجود در این جدول در عملکرد ربات در محیط آزمون حدود ۱۸/۶، ۳۷/۹، ۴۴/۹، و ۱۴/۵ درصد بهبود یافته است. شکل‌های ۹، ۱۰، ۱۱ و ۱۲ مسیر طی شده توسط ربات را برای هر یک از چهار کنترلگر پیشنهادی در دو نمونه از بهترین اجراها در آزمایش‌های انجام شده در محیط آزمون نمایش می‌دهد. در این شکل‌ها با بررسی وضعیت خط‌سیر ربات می‌توان مشاهده کرد که ربات با هدایت کنترلگر چهارم که از ویژگی‌های حافظه‌ای بیشتری بهره‌برداری می‌کند کمتر مسیر را گم کرده و توانسته است مطابقت بیشتری مابین خط‌سیر خود و پیچ‌وتاب موقعیت دیوار ایجاد نماید. لازم به ذکر است که شکست "خیلی دور از دیوار" گاه به دلیل وجود موانع اجتناب‌ناپذیر است و به عنوان مثال، مقدار این معیار ارزیابی در جدول ۳ برای آخرین معماری با توجه به اینکه ربات به طور متوسط حدود ۲۶۰۰ گام زمانی در محیط آزمون فعال است، تنها ۲۲/۰۷ ٪ از زمان فعالیت ربات را تشکیل می‌دهد.

از آنجا که پدیده‌هایی همچون نویز، اغتشاشات و عدم‌قطعیت‌ها می‌توانند در محیط‌های واقعی روی پاسخ‌های سیستم تأثیرات منفی داشته باشند، عملکرد کنترلگر حاصل از معماری چهارم در حضور تصاویر نویزی بررسی شده است. برای این منظور نویزی گوسی (با میانگین صفر و واریانسی برابر با ۰/۱ بیشترین عمقِ قابل تشخیص توسط دوربین کینکت) به مقدار هر پیکسل از تصاویر عمق دوربین کینکت، اضافه شده است. هر کانال RGB از تصاویر رنگی دوربین نیز با نویز گوسی با میانگین صفر و واریانسی برابر با ۰/۱ (مقدار صفر نشان دهنده تصویر بدون نویز و مقدار ۱ بیشترین میزان نویز را نشان می‌دهد) دستخوش تغییر شده است. جدول ۴ نتایج بدست آمده از آزمایش‌ها را برای عملکرد کنترلگر DIDFSH-SDL در محیط آزمون و در مقایسه با نتایج گزارش‌شده از عملکرد این کنترلگر در جدول ۳، نشان می‌دهد. این بررسی مشخص می‌کند کنترلگر پیشنهادی در مقابل تأثیرات نویز عملکردی پایدار دارد. این پایداری به دلیل نحوه‌ی انتخاب ویژگی‌ها و استفاده از اطلاعات آماری مقادیر عمق یعنی میانگین و واریانس در نقشه‌های بلوکی بدست می‌آید.



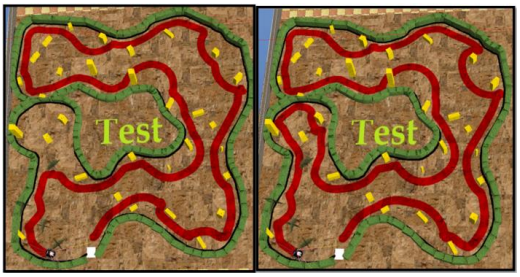
[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

جدول ۴: نتایج عملکرد معماری پیشنهادی چهارم برای کنترلگر ربات در محیط آزمون و در حضور تصاویر نویزی

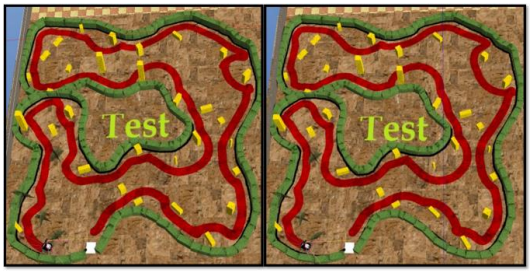
| معیار ارزیابی (بر حسب تعداد) | روش‌های پیشنهادی در محیط آزمون | |
|---|---|---|
| | DIDFSH-SDL | DIDFSH-SDL (Noisy Images) |
| شکست "برخورد با مانع" | ۵۳/۲ | **۵۲/٦** |
| شکست "خیلی نزدیک به دیوار" | **۵۰/٤** | ۵۴/۸ |
| شکست "خیلی دور از دیوار" | **۵۷۳/۸** | ۵۹۴/۶ |

لازم به ذکر است که پیش از بررسی معماری‌های منطبق بر چارچوب پیشنهادی، کنترلگری با ساختار یک شبکه‌عصبی ۷ لایه با ۹۶ نرون در لایه‌ی ورودی ایجاد شد. تعداد نرون‌ها در لایه‌های پنهان این شبکه به ترتیب برابر با ۲۰، ۳۰، ۵۰، ۶۰، ۸۰ نرون بوده و یک نرون خطی در لایه‌ی خروجی با شرایطی مشابه معماری SI-SDL تنظیم شد. این شبکه با استفاده از مجموعه‌ی دوم داده‌های آموزشی به روش با ناظر آموزش داده شد؛ اما ربات در محیط بسیار نامناسب عمل کرده و ربات قادر به تشخیص هیچ یک از موانع موجود در محیط نبود.

شکل ۹: نمایی از خط سیر ربات در هدایت توسط کنترلگر SI-SDL در محیط آزمون در ۲ اجرای مستقل




شکل ۱۰: نمایی از خط سیر ربات در هدایت توسط کنترلگر Dl-SDL در محیط آزمون در ۲ اجرای مستقل




شکل ۱۱: نمایی از خط سیر ربات در هدایت توسط کنترلگر DIDF-SDL در محیط آزمون در ۲ اجرای مستقل

شکل ۱۲: نمایی از خط سیر ربات در هدایت توسط کنترلگر DIDFSH-SDL در محیط آزمون در ۲ اجرای مستقل

## ٤- نتیجه‌گیری و کارهای آینده

در این مقاله، چارچوبی برای طراحی کنترلگر ربات بهره‌برنده از تصاویر عمق با اهداف پیگیری دیوار و اجتناب از موانع پیشنهاد شد. در طراحی کنترلگرهای پیشنهادی از ترکیب یادگیری عمیق و یادگیری با ناظر استفاده شد. چارچوب پیشنهادی از یک شبکه‌عصبی عمیقِ پیش‌رو با ساختاری متقارن برای بازنمایی وضعیت محیط پیرامون ربات در تصاویر دوربین استفاده کرده و با استفاده از یادگیری با ناظر، وزن‌های شبکه را آموزش می‌دهد؛ سپس لایه‌های موجود در نیمه‌ی ابتدایی شبکه به عنوان بخشی از معماری کنترلگر مورد بهره‌برداری قرار گرفته و یک لایه‌ی تک نرونی با تابع انتقال خطی به عنوان لایه‌ی خروجی به ساختار افزوده می‌شود. برای بهبود کیفیت کنترل، ویژگی‌های حافظه‌ای به ورودی‌های لایه‌ی خروجی کنترلگر محلق شدند. بخشی از این ویژگی‌ها موقعیت ربات را نسبت به دیوار مشخص کرده و بخشی وضعیت رفتاری ربات و تصمیمات پیشین را ارائه می‌دهند.

نتایج آزمایش‌ها نشان می‌دهند که ایجاد نقشه‌های بلوکی و بازنمایی ویژگی‌ها با استفاده از یادگیری عمیق می‌تواند تعداد داده‌های ورودی کنترلگر را کاهش داده و با خلاصه‌سازی و خوشه‌بندی اطلاعات، ویژگی‌های مناسبی را در اختیار کنترلگر قرار دهد به گونه‌ای که اختلافات جزئی تصاویر در شرایط مشابه (که در ناوبری ربات با فضای حالت ورودی گسترده بسیار متداول هستند و می‌توانند منجر به ناسازگاری داده‌ها در یادگیری با ناظر شوند)، کاهش یافته و تمایزهای اصلی مابین تصاویر برجسته‌تر شود. تأثیر افزودن هر یک از ویژگی‌ها در چهار معماری پیشنهادی، در نتایج آزمایش‌ها به خوبی قابل رؤیت است؛ به طوری‌که آخرین معماری پیشنهاد شده (کنترلگر DIDFSH-SDL) نسبت به اولین معماری (کنترلگر SI-SDL) بهبود چشمگیری داشته و نرخ تعداد شکست‌ها برای هر یک از معیارهای ارزیابی مورد بررسی بهبود یافته است.

چارچوب پیشنهادی در این مقاله تنها در محیط‌های هموار آزموده شده است؛ بررسی عملکرد معماری پیشنهادی برگزیده در محیط‌های ناهموار، تعیین ویژگی‌های مناسب برای بازنمایی شرایط این محیط‌ها، تنظیم دقیق‌تر و برخط وزن‌های شبکه‌عصبی با استفاده از روش‌های یادگیری تقویتی و تلاش برای به صفر رساندن تعداد شکست‌ها از جمله مواردی است که می‌توان در کارهای آتی به آنها پرداخت.



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]

## مراجع

[1] Hanafi, D., Abueejela, Y. M., & Zakaria, M. F., 2013, "Wall follower autonomous robot development applying fuzzy incremental controller". Intelligent Control and Automation, 4(1), 18.

[2] Ye, C., Yung, N. H., & Wang, D., 2003, "A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance". IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 33(1), 17-27.

[3] Fathinezhad, F., Derhami, V., & Rezaeian, M., 2016, "Supervised fuzzy reinforcement learning for robot navigation". Applied Soft Computing, 40, 33-41.

[4] Fathinezhad, F., & Derhami, V., 2012, "A Novel Supervised Fuzzy Reinforcement Learning for Robot Navigation", [Research]. Journal of Control, 6(3), 1-10.

[5] Carelli, R., & Freire, E. O., 2003, "Corridor navigation and wall-following stable control for sonar-based mobile robots". Robotics and Autonomous Systems, 45(3), 235-247.

[6] Karakuş, M. Ö., & Orhan, E., 2013, "Learning of robot navigation tasks by probabilistic neural network". Learning.

[7] Zhou, Z., Chen, T., Wu, D., & Yu, C., 2011, "Corridor navigation and obstacle distance estimation for monocular vision mobile robots". JDCTA: Int. J. of Digital Content Technology and its Applications, 5(3), 192-202.

[8] Jafar, F. A., Zakaria, N. A., & Yokota, K., 2014, "Visual Features Based Motion Controller for Mobile Robot Navigation". International Journal of Simulation Systems, Science & Technology, 15(1), 7-14.

[9] Saeedi, P., Lawrence, P. D., & Lowe, D. G., 2006, "Vision-based 3-D trajectory tracking for unknown environments". IEEE transactions on robotics, 22(1), 119-136.

[10] Yang, Y., Fu, M., Zhu, H., Xiong, G., & Changsheng, S., 2010, "Control methods of mobile robot rough-terrain trajectory tracking", Control and Automation (ICCA), 2010 8th IEEE International Conference on.

[11] Hoffmann, G. M., Tomlin, C. J., Montemerlo, M., & Thrun, S., 2007, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing", American Control Conference.

[12] Oliver, A., Kang, S., Wünsche, B. C., & MacDonald, B., 2012, "Using the Kinect as a navigation sensor for mobile robotics", Proceedings of the 27th Conference on Image and Vision Computing New Zealand.

[13] Correa, D. S. O., Sciotti, D. F., Prado, M. G., Sales, D. O., Wolf, D. F., & Osorio, F. S., 2012, "Mobile robots navigation in indoor environments using kinect sensor", Second Brazilian Conference on Critical Embedded Systems (CBSEC).

[14] Yung, N. H., & Ye, C., 1999, "An intelligent mobile vehicle navigator based on fuzzy logic and reinforcement learning". IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 29(2), 314-321.

[15] Jang, J., Sun, C., & Mizutani, E., 1997, "Neuro-Fuzzy and Soft Computing, Prentice-Hall, upper Sanddle River".

[16] Riedmiller, M., 2005, "Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method", European Conference on Machine Learning.

[17] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., . . . Ostrovski, G., 2015, "Human-level control through deep reinforcement learning". Nature, 518(7540), 529-533.

[18] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N. M. O., Erez, T., Tassa, Y., . . . Wierstra, D. P., 2016, "Continuous control with deep reinforcement learning": Google Patents.

[19] Ondruska, P., & Posner, I., 2016, "Deep tracking: Seeing beyond seeing using recurrent neural networks". The Thirtieth AAAI Conference on Artificial Intelligence (AAAI).

[20] Bengio, Y., Courville, A., & Vincent, P., 2013, "Representation learning: A review and new perspectives". IEEE transactions on pattern analysis and machine intelligence, 35(8), 1798-1828.

[21] Courville, I. G. a. Y. B. a. A., Deep Learning: MIT Press, 2016.

[22] Bengio, Y., 2009, "Learning deep architectures for AI". Foundations and trends® in Machine Learning, 2(1), 1-127.

[23] Hinton, G. E., & Salakhutdinov, R. R., 2006, "Reducing the dimensionality of data with neural networks". Science, 313(5786), 504-507.

[24] Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y., 2009, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations", Proceedings of the 26th annual international conference on machine learning.

[25] Liu, J. N., Hu, Y., You, J. J., & Chan, P. W., 2014, "Deep neural network based feature representation for weather forecasting", Proceedings on the International Conference on Artificial Intelligence (ICAI).

[26] Günther, J., Pilarski, P. M., Helfrich, G., Shen, H., & Diepold, K., 2016, "Intelligent laser welding through representation, prediction, and control learning: An architecture with deep neural networks and reinforcement learning". Mechatronics, 34, 1-11.

[27] Williams, D., & Hinton, G., 1986, "Learning representations by back-propagating errors". Nature, 323(6088), 533-538.

[28] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A., 2010, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion".



[ Downloaded from joc-isice.ir on 2026-01-31 ]
[ DOR: 20.1001.1.20088345.1396.11.4.6.9 ]