

دانشکده مهندسی کامپیوتر

**پروژه مقطع کارشناسی مهندسي کامپیوتر**

**ساخت و آموزش مدل حرکتي ربات در مسيرهای پرچالش به کمک يادگيری تقويتي عميق**

پرهام نوران بخت

**استاد راهنما:**

دکتر سید حسین خواسته

**[تاریخ دقیق روز، ماه و سال دفاع]**



# تأييديّه هيات داوران

اعضاي هيئت داوران، نسخه نهائي پروژه خانم / آقاي: [نام دانشجو]

را با عنوان: [عنوان پروژه]

از نظر شکل و محتوي بررسي نموده و پذيرش آن را براي تکميل درجه کارشناسی تأييد مي‏کنند.

|  |  |  |  |
| --- | --- | --- | --- |
| **اعضاي هيئت داوران** | **نام و نام خانوادگي** | **رتبه علمي** | **امضاء** |
| 1. استاد راهنما |  |  |  |
| 1. استاد داور |  |  |  |

**دانشگاه صنعتی خواجه نصیرالدین طوسی**

|  |
| --- |
| **اظهارنامه دانشجو** |
| اینجانب **پرهام نوران بخت** دانشجوی مقطع کارشناسی رشته **مهندسی کامپیوتر** گواهی می‌نمایم که مطالب ارائه شده در این پروژه با عنوان:  **ساخت و آموزش مدل حرکتي ربات در مسيرهای پرچالش به کمک يادگيری تقويتي عميق**  با راهنمایی استاد محترم **دکتر سید حسین خواسته** توسط شخص اینجانب انجام شده است. صحت و اصالت مطالب نوشته شده در این پروژه تأیید می‌شود و در تدوین متن پروژه قالب مصوب دانشگاه را به طور کامل رعایت کرده‌ام.  **امضاء دانشجو:**  **تاریخ:** |

**حق طبع، نشر و مالکیت نتایج**

1- حق چاپ و تکثیر این پروژه متعلق به نویسنده و استاد راهنمای آن است. هرگونه تصویربرداری از کل یا بخشی از پروژه تنها با موافقت نویسنده یا استاد راهنما یا کتاب‌خانه دانشکده‌های مهندسی برق و کامپیوتر دانشگاه صنعتی خواجه نصیرالدین طوسی مجاز است.

2- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی خواجه نصیرالدین طوسی است و بدون اجازه کتبی دانشگاه قابل واگذاری به شخص ثالث نیست.

3- استفاده از اطلاعات و نتایج موجود پروژه بدون ذکر مرجع مجاز نیست.

###### تقديم به:

###### تشکر و قدرداني

###### چکيده

همانطور که از عنوان پایان نامه مشخص است هدف این پروژه دستیابی به عاملی است که بتواند در مسیر های پرچالش بهترین عملکرد ممکن را داشته باشد؛ به عبارتی ربات مذکور باید بتواند در مسیر ها با چالش های گوناگون به حرکت خود آنگونه که برایش تعریف شده ادامه دهد. برای دستیابی به این هدف ابتدا باید محیط[[1]](#footnote-2) های فیزیکی موردنظر را بر روی یک موتور فیزیک پیاده کنیم، سپس با استفاده از الگوریتم[[2]](#footnote-3) های مناسب یادگیری تقویتی عمیق، عاملی را ساخته و آموزش دهیم که بتواند در ارتباط با این محیط ها به امتیاز قبل قبولی دست یابد. این عامل باید با زبان پایتون[[3]](#footnote-4) و صرفا با استفاده از کتابخانه‌ی تنسورفلو[[4]](#footnote-5) و با کمک واسط کراس[[5]](#footnote-6) طراحی شده باشد.

1.

در این پروژه یک کتابخانه جدید شامل چند محیط ساخته شده است که امکان برقراری ارتباط با آن ها از طریق واسط جیم[[6]](#footnote-7) و آموزش و آزمایش عامل ها در این محیط‌ها به کمک جیم و یا ابزار مشابه وجود دارد. با توجه به ماهیت ذاتی پروژه می توان این محیط ها را به دو دسته تقسیم کرد: 1- محیط های مخصوص آموزش 2- محیط های غیر قابل استفاده برای آموزش؛ در محیط های آموزشی ویژگی تصادفی بودن از پاداش حاصل از اعمال عامل تا جای ممکن باید حذف شود در ادامه در این باره به طور کامل توضیح داده می شود.

علاوه بر کتابخانه مذکور، یک کتابخانه شامل عامل های پیاده کننده الگوریتم های یادگیری تقویتی عمیق نیز ساخته شده است که در آن دو عامل پایه و ساده A2C و MPO و دو عامل مشتق از A2C یعنی PPO و TRPO وجود دارد. به طور کلی در مقایسه با مدل های مشابه در محیط های یکسان و یا شبیه محیط های پر چالش طراحی شده، بهترین عملکرد تاکنون مربوط به مدل پیاده شده در بستر کتابخانه پای تورچ[[7]](#footnote-8) است که الگوریتم SAC را با دو لایه میانی به ترتیب به اندازه های 400 و 300 پیاده می کند. مدل های طراحی شده در این پروژه با دو لایه میانی به اندازه های 64 برای الگوریتم های A2C و مشتق از آن و دو لایه میانی به اندازه های 256 برای الگوریتم MPO به نتایج مشابه و گاها بهتر دست می یابد. با توجه به کوتاه بودن زمان آموزش که برای مدل ها در بستر پای تورچ و با سخت افزار مناسب در حدود نیم ساعت و برای مدل های پیاده شده در بستر تنسورفلو در حدود یک الی دو ساعت است از مقایسه این پارامتر چشم پوشی می کنیم.

قابلیت نصب و استفاده بدون مشکل از هردو کتابخانه و تمام ویژگی هایشان بر روی تمام سیستم عامل هایی که قابلیت نصب پایتون و پیپ[[8]](#footnote-9) را دارند به راحتی و از طریق اجرای دستور

pip install -e {Library’s Directory} وجود دارد.

یا.......؟

2.

برای دستیابی به اهداف ذکر شده ابتدا محیط موردنظر را با استفاده از موتور فیزیک بولت[[9]](#footnote-10) طراحی می کنیم برای آموزش و آزمایش عامل از کتابخانه جیم[[10]](#footnote-11) و برای رفع مشکلات احتمالی محیط طراحی شده و اطمینان از عملکرد درست این محیط از یک مدل از پیش طراحی شده در بستر کتابخانه پای تورچ[[11]](#footnote-12) بهره می گیریم؛ این عامل را ابتدا صرفا بر روی یک محیط ساده و بدون چالش آموزش می دهیم و بعد از تنظیم عامل و میزان سازی فراپارامتر های مدل، کار را با فراپارامتر های مشابه بر روی محیط های پر چالش ادامه می دهیم. بعد از آزمایش و اطمینان از صحت عملکرد محیط های طراحی شده به پیاده سازی و آموزش مدل ها و عامل ها با استفاده از مقالات مربوطه و در بستر کتابخانه تنسورفلو می پردازیم.

درنهایت عامل ها را بر روی محیط طراحی شده در مرحله قبل آموزش می دهیم. با توجه به نتایج حاصل و مقایسه این نتایج با نتایج مقالات و مدل بی نقص آموزش داده شده قبلی به رفع ایرادات مدل های طراحی شده می پردازیم. به عبارتی عملکرد مدل بی نقص را معادل عملکرد سطح انسانی در نظر می گیریم و سعی در دستیابی به دقت بیز می کنیم. در نهایت با توجه به محدودیت تعداد گام های هر دوره پاداش متوسط حاصل آزمودن عامل در هر دوره به یک مقدار حداکثری میل می کند و در برخی مدل ها در طی تکرار های بعدی از این مقدار کم تر می شود.

###### فهرست مطالب

عنوان صفحه

[تأييديّه هيات داوران ‌ه](#_Toc112929874)

[فهرست شکل‏ها ‌و](#_Toc112929875)

[فهرست جدول‏ها ‌ز](#_Toc112929876)

[فهرست علامت‏ها و نشانه‏ها ‌ط](#_Toc112929877)

[فصل 1- مقدمه 1](#_Toc112929878)

[1-1- پيشگفتار 1](#_Toc112929879)

[1-2- تاریخچه 1](#_Toc112929880)

[1-2-1- محیط ها 1](#_Toc112929881)

[1-2-2- الگوریتم ها 4](#_Toc112929882)

[1-3- آنچه باید انجام شود 5](#_Toc112929883)

[1-3-1- کوچک تر کردن ساختار شبکه های عصبی 5](#_Toc112929884)

[1-3-2- کاهش زمان آموزش 6](#_Toc112929885)

[1-3-3- کنترل بیشتر بر روی ربات 6](#_Toc112929886)

[1-3-4- بهینه کردن مصرف توان ربات 7](#_Toc112929887)

[1-3-5- حل چالش های مشابه جهان واقعی 7](#_Toc112929888)

[1-4- نوآوری های ارائه شده 8](#_Toc112929889)

[1-4-1- ساخت کتابخانه شامل محیط های مورد نظر 8](#_Toc112929890)

[1-4-2- ساخت کتابخانه شامل چند الگوریتم ساده یادگیری تقویتی عمیق 8](#_Toc112929891)

[1-4-3- پیاده سازی، آموزش و تنظیم عامل هایی برای حل چالش های محیطی موردنظر 8](#_Toc112929892)

[1-5- ساختار پایان نامه 8](#_Toc112929893)

[فصل 2- ساختار بيان مطالب 10](#_Toc112929894)

[فصل 3- سبك نگارش 11](#_Toc112929895)

[فصل 4- مرجع دهی 12](#_Toc112929896)

[فهرست مرجع‏ها 13](#_Toc112929897)

فهرست شکل‏ها

عنوان صفحه

[شکل ‏3‑1 اندازه و حالت‏دهی چینش صفحه (الف) حاشیه‏ها، (ب) کاغذ، و (ج) طرح برای کل فایل 15](#_Toc530263777)

[شکل ‏3‑2 پنجره توليد فهرست مطالب. 17](#_Toc530263778)

[شکل ‏3‑3 پنجره به‌هنگام کردن محتواي جدول فهرست مطالب. 18](#_Toc530263779)

[شکل ‏3‑4 تنظيمات لازم در MATLAB برای انتقال شکل به MS-Word. 19](#_Toc530263780)

[شکل ‏3‑5 شکل با فضای خالی در راست و بالای آن (الف) و شکل با حذف فضای خالی (ب) 20](#_Toc530263781)

[شکل ‏3‑6 پنجره دادن خودکار شرح شکل و عنوان جدول. 21](#_Toc530263782)

[شکل ‏3‑7 پنجره Cross-reference برای ارجاع به شکل یا جدول در متن 22](#_Toc530263783)

[شکل ‏3‑8 تنظيمات لازم براي تهيه فهرست شکل‏ها و جدول‏ها. 24](#_Toc530263784)

[شکل ‏3‑9 تنظيمات پانِوِشت. 25](#_Toc530263785)

فهرست جدول‏ها

عنوان صفحه

[جدول ‏1‑1 سبك‏هاي تعريف شده و موارد استفاده آنها 33](#_Toc530720185)

فهرست علامت‏ها و نشانه‏ها

عنوان علامت اختصاري

ثابت انتشار *γ*

ثابت پلانک *h*

ثابت فاز *β*

دمای مطلق (معمولا بر حسب کلوین) 

ضریب رسانایی الکتریکی *σ*

ضریب گذردهی الکتریکی *ε*

ضریب نفوذپذیری مغناطیسی *μ*

فرکانس پلاسما *ωp*

# مقدمه

## پيشگفتار

همانطور که در بخش تاریخچه مطرح می شود، بسیاری از محیط ها و عامل های طراحی شده برای آنها بسیار ساده اند و پاسخگوی چالش های اساسی دنیای واقعی نیستند. هدف از این پروژه طراحی عاملی با کمک الگوریتم های یادگیری تقویتی عمیق است که بتواند در محیطی با چالش های محیطی دنیای واقعی عملکرد مطلوبی با توجه به ساختار فیزیکی ربات مورد استفاده داشته باشد. برای این منظور علاوه بر پیاده سازی و مقایسه الگوریتم های مذکور برای یافتن بهترین الگوریتم پاسخگوی این اهداف، نیازمند طراحی محیط هایی تا حد ممکن شبیه به محیط های موردنظر با چالش های مشابه در جهان واقعی هستیم.

## تاریخچه

در این بخش به بررسی کارهای انجام شده در زمینه طراحی محیط های مورد استفاده برای آموزش عامل ها در کنار الگوریتم های یادگیری عمیق تقویتی پیاده شده در این پروژه و پیاده سازی های مطرح و پراستفاده دیگر آنها می پردازیم.

### محیط ها

ابتدا به بررسی چند پروژه پرکاربرد در این زمینه می پردازیم.

بولت[[12]](#footnote-13): بولت یک موتور فیزیک است که اثر تصادم و دینامیک اشیا نرم و سخت را شبیه سازی می کند. این موتور برای بازی های کامپیوتری و جلوه های بصری در فیلم ها مورد استفاده قرار می گیرد. نویسنده اصلی این پروژه، اروین کومنز، برنده یک جایزه علمی و تخصصی آکادمیک برای کارهایش بر روی بولت شده است.

کتابخانه فیزیک بولت یک نرم افزار رایگان و متن باز است. منبع کد از بعد از سال 2014 بر روی گیتهاب قرار گرفته ولی پیش از آن بر روی گوگل کد قرار داشت. بخش عمده و اصلی زبان های استفاده شده در این پروژه به ترتیب سی پلاس پلاس[[13]](#footnote-14) و سی[[14]](#footnote-15) هستند.

پای بولت [1][[15]](#footnote-16): پای بولت یک ماژول ساده پایتون برای شبیه سازی فیزیکی، رباتیک و یادگیری تقویتی عمیق است که برمبنای کتابخانه فیزیک بولت پیاده سازی شده است. با کمک این کتابخانه امکان بارگذاری بدنه بند بند ربات ها از طریف فایل های یوآردی اف[[16]](#footnote-17)، اس دی اف[[17]](#footnote-18)، ایکس ام ال[[18]](#footnote-19) و دیگر فرمت های مشابه وجود دارد. از دیگر امکانات این کتابخانه می توان به شبیه سازی دینامیک[[19]](#footnote-20) رو به جلو، محاسبات دینامیک معکوس، فیزیک جنبش جلو رونده و معکوس و تشخیص برخورد اجسام اشاره کرد.

جیم [2][[20]](#footnote-21): جیم یک کتابخانه متن باز پایتون برای توسعه و مقایسه الگوریتم های یادگیری تقویتی از طریق ارائه یک واسط استاندارد برای ارتباط بین الگوریتم های یادگیری و محیط هاست.

محیط های موجوکو [2][[21]](#footnote-22): موجوکو سروند "دینامیک چند مفصله با تصادم"]معادل[ است. این کتاب خانه یک موتور فیزیکی برای تسهیل تحقیقات، توسعه رباتیکز[[22]](#footnote-23)، بیومکانیکز[[23]](#footnote-24)، گرافیک[[24]](#footnote-25) و انیمیشن[[25]](#footnote-26) است. تمام محیط های موجوکو نیازمند نصب موتور موجوکو هستند. درحال حاضر 10 محیط در کتابخانه موجوکو وجود دارد.

از سال 2021 دیپ مایند[[26]](#footnote-27)، موجوکو را به دست آورده است و از سال 2022 آن را متن باز خواهد نمود که در نتیجه آن برای همگان رایگان خواهد شد.

پای بولت جیم پریوم [3][[27]](#footnote-28): برای رفع مشکل متن باز نبودن موجوکو و تا قبل از متن باز شدن آن یک تیم از توسعه دهندگان برای آسان تر کردن کار های توسعه و تحقیق در زمینه یادگیری تقویتی، تصمیم به توسعه پای بولت جیم گرفتند که بعد ها به پای بولت جیم پریوم تغییر نام داد.

این کتابخانه یک توسعه متن باز از محیط های موجوکوی اپن ای آی جیم است. برای آسان‌تر کردن روند توسعه و نشان دادن کارکرد و استفاده محیط های این کتابخانه، چند عامل از کتابخانه یادگیری تقویتی تنسورفورس[[28]](#footnote-29) برای این محیط ها تنظیم و آموزش داده شده اند بنابراین برای هر محیط چندین عامل از پیش آموزش داده شده به عنوان تست واحد و خطوط محوری برای توسعه های بعدی وجود دارد.

رکس-جیم [4][[29]](#footnote-30): هدف این پروژه آموزش یک ربات چهارپای پرینت 3 بعدی شده با پویش در الگوریتم های یادگیری تقویتی و محیط های اپن ای آی جیم است. مقصود این است که عامل طراحی شده وظایف خانگی و عمومی را در محیط شبیه سازی یادبگیرد و سپس بدون نیاز به هیچ میزان سازی و تنظیمات دیگری این دانش یا به عبارتی سیاست آموخته شده را بدون هیچ مشکلی بر روی ربات واقعی به کار بگیرد.

در این پروژه تعدادی وظایف زیر توسط عامل طراحی شده انجام می شوند:

1. کنترل پایه
   1. ژست گرفتن ربات طبق دستورات کنترلی
   2. دویدن
   3. راه رفتن
   4. چرخیدن در یک نقطه
   5. ایستادن از حالت خوابیده روی زمین
2. مسیر یابی در زمین های نا هموار
   1. مسیر یابی بر روی تپه و زمین های ناهموار ایجاد شده به صورت تصادفی

دی ام کنترل [5][[30]](#footnote-31):

### الگوریتم ها

بیس لاینز [6]: اپن ای آی[[31]](#footnote-32) بیس لاینز یک مجموعه با کیفیت از پیاده سازی الگوریتم های یادگیری تقویتی است. این الگوریتم ها کار را برای جامعه محقق آسان تر می کنند تا به کمک این خطوط پایه به بهبود و حتی پیاده سازی ایده های جدید بپردازند.

در این کتابخانه زبان پایتون، الگوریتم های زیر با نسخه 1.14 تنسورفلو (بدون واسط کراس) پیاده شده اند:

1. ای 2 سی[[32]](#footnote-33)
2. ای سر[[33]](#footnote-34)
3. اکتر[[34]](#footnote-35)
4. دیدی پیجی[[35]](#footnote-36)
5. دیکیو ان[[36]](#footnote-37)
6. گیل[[37]](#footnote-38)
7. هر[[38]](#footnote-39)
8. پیپیو1 [[39]](#footnote-40)
9. پیپیو2
10. تی آر پیو[[40]](#footnote-41)

استیبل بیس لاینز [7][[41]](#footnote-42): استیبل بیس لاینز یک مجموعه از پیاده سازی های بهبودیافته الگوریتم های یادگیری تقویتی است که بر پایه پیاده سازی های اپن ای آی بیس لاینز است. در حال حاضر این کتابخانه در حال تعمیر است و در صورت نیاز به این الگوریتم ها باید از پیاده سازی دیگر این تیم یعنی استیبل بیس لاینز 3 استفاده شود. شایان ذکر است که این کتابخانه تنسورفلوی نسخه 1.8.0 تا نسخه 1.14.0 را پشتیبانی می کند و پیاده سازی با تنسورفلوی 2 در حال برنامه ریزی است.

از تفاوت های عمده این پیاده سازی با پیاده سازی اپن ای آی می توان به موارد زیر اشاره کرد:

* ساختار یکپارچه برای تمام الگوریتم ها
* سبک یکپارچه کد
* کلاس ها و توابع مستند نویسی شده
* تست های بیشتر و پوشش بیشتر کد با تست ها
* الگوریتم های جدید پیاده شده شامل سک[[42]](#footnote-43) و تی دی 3[[43]](#footnote-44)

استیبل بیس لاینز3[[44]](#footnote-45) [8]: استیبل بیس لاینز 3 یک مجموعه از پیاده سازی های قابل اطمینان از الگوریتم های یادگیری تقویتی در بستر پای تورچ است که دومین پیاده سازی اصلی بهره گرفته از استیبل بیس لاینز اپن ای آی محسوب می شود. این کتابخانه یک واسط تمیز و ساده برای استفاده از الگوریتم های یادگیری تقویتی ارائه می دهد.

این کتابخانه کاملا مستند نویسی شده و آزموده، امکان آموزش عامل ها را تنها در چند خط فراهم می کند.

عامل های تنسورفلو[[45]](#footnote-46) [9]: یک کتابخانه مقیاس پذیر، مطمئن و ساده برای استفاده نوشته شده در بستر تنسورفلو برای الگوریتم های بندیتز متنی[[46]](#footnote-47) و یادگیری تقویتی است.

این کتابخانه پیاده سازی، توسعه و آزمودن الگوریتم های یادگیری تقویتی را آسان تر می کند. این کتابخانه همچنین اجزا آزموده و قابل تغییری را در اختیار می گذارد که امکان دوره های سریع کد و محک آن را فراهم می آورد.

در این کتابخانه هسته اصلی الگوریتم های یادگیری تقویتی به عنوان عامل ها پیاده شده اند.هر عامل دو مسئولیت اصلی را بر عهده می گیرد: 1- تعریف یک سیاست برای ارتباط با محیط و 2- چگونگی تعلیم (بهبود) این سیاست به کمک تجربه کسب شده

این کتابخانه از تمام نسخه های تنسورفلو بعد از تنسورفلوی 1.15.0 پشتیبانی می کند.

## آنچه باید انجام شود

### کوچک تر کردن ساختار شبکه های عصبی

در بسیاری از پیاده سازی های بررسی شده در بخش تاریخچه برای دستیابی عامل به نتایج قابل قبول نیاز به آموزش شبکه های عمیق و یا پهنی است که علاوه بر مصرف حافظه برای نگه داری مدل آموزش دیده به رم[[47]](#footnote-48) با حجم زیاد برای بارگذاری و استفاده از عامل هم نیاز است. علاوه بر این موارد، آموزش اینچنین مدل هایی به طور کلی نیازمند زمان بیشتری است؛ زیرا محاسبات لازم برای چنین شبکه هایی بسیار بیشتر است. بسته به سخت افزار مورد استفاده این زمان ممکن است از چند ساعت تا چند روز متفاوت باشد؛ به عبارت دیگر در این موارد برای آموزش و استفاده مطلوب از عامل طراحی شده نیازمند حافظه و رم با حجم بیشتر و پردازنده های قوی تری هستیم که ممکن است برای تولید انبوه در دسترس نباشد یا باعث زیاد شدن هزینه ها شود.

### کاهش زمان آموزش

یکی از عوامل مهمی که می تواند در ارزیابی و مقایسه عامل های مختلف استفاده شود زمان مورد نیاز برای آموزش آنها (مشخصا در یک محیط یکسان) می باشد.

همانطور که در بخش قبل اشاره شد یکی از راه های کاهش زمان آموزش کوچکتر کردن شبکه عصبی مورد نیاز برای عامل طراحی شده می باشد. علاوه بر این همانطور که در فصول بعد مطرح می شود برخی الگوریتم های یادگیری تقویتی عمیق باوجود عملکرد مشابه سایر الگوریتم ها نیاز به شبکه های عصبی کوچک تر و زمان آموزش کمتری دارند.

### کنترل بیشتر بر روی ربات

در بیشتر محیط ها و عامل های طراحی شده هدف اصلی آموزش راه رفتن و یا دویدن به ربات های طراحی شده است و در مواردی که جهت گیری در محیط به ربات آموزش داده می شود ربات امکان تغییر مسیر و حرکت در مسیر جدید را ندارد و صرفا جهت گیری جدیدی به سمت هدف انجام می دهد به عبارتی در این موارد یک یا چندین مدل برای وظایف مختلف آموزش داده می شود و امکان استفاده یک جا از تمام قابلیت ها برای مثال برای جهت گیری و حرکت، تنها از طریق تغییر دستی مدل در حین اجرا ممکن است که عملا کاری غیر ممکن است و حتی در صورت امکان به دلیل نیاز این روش به ذخیره وزن های چندین مدل و در برخی موارد بارگذاری لحظه ای مدل ها نیازمند منابع سخت افزاری بیشتری هستیم.

در بسیاری دیگر از مواردی که امکان کنترل ربات از طریق مشخص کردن هدف و مسیر حرکت برای آن فراهم است، ربات مورد استفاده بسیار ساده است؛ به عبارتی عامل طراحی شده به دلیل کم بودن عوامل کنترلی ربات یا به عبارتی کم بودن تعداد عمل[[48]](#footnote-49) های ممکن برای عامل، کار مسیر یابی را به راحتی انجام می دهد. استفاده از اینگونه ربات ها می تواند در بسیاری از موارد پاسخگوی نیاز ما باشد اما در مواردی که نیاز به مسیر یابی در محیط های دشوار و پرچالش هست استفاده از این ربات ها و عامل ها نمی تواند مناسب باشد.

### بهینه کردن مصرف توان ربات

با توجه به موارد استفاده از اینگونه ربات ها نیازمند این هستیم که این ربات ها تا حد ممکن کم ترین مصرف توان را با وجود بهترین عملکرد ممکن داشته باشند. برای این منظور علاوه بر درنظر داشتن موارد ذکر شده در بخش های قبل مثل کوچک تر کردن ساختار شبکه عصبی، پیاده سازی بهتر الگوریتم و یا استفاده از الگوریتم های بهینه، باید تا حد امکان عملکرد ربات را به عملکرد مطلوب نزدیک کنیم؛ به عبارتی برای رسیدن به هدف موردنظر راه های زیادی وجود دارد ولی تنها یک راه بهینه است.

اختصاص زمان بیشتر به آموزش ربات طراحی شده می تواند متوسط امتیاز عامل را برای یک اپیزود تا حد مشخصی بالا ببرد و بعد از آن عملا رشد نمودار امتیاز به گام های آموزش، متوقف شده و به خط صاف میل می کند.

برای پیدا کردن راه بهینه، باید بیشترین امتیاز یا متوسط آن در یک اپیزود حاصل شود. برای این منظور باید با تنظیم فراپارامتر های هر الگوریتم بهترین عملکرد هر الگوریتم را بیابیم و از بین الگوریتم های موجود با توجه به امکانات سخت افزاری و دیگر ترجیحات یکی را انتخاب کنیم.

### حل چالش های مشابه جهان واقعی

عامل طراحی شده باید بتواند در محیط هایی که به دلایلی امکان استفاده از دیگر روش های حمل و نقل در آنها وجود ندارد مورد استفاده قرار بگیرد. برای این منظور علاوه بر درنظر داشتن موارد ذکر شده در بخش های قبل، عامل باید بتواند در محیط هایی مانند پله، زمین های نا هموار، شیب دار و یا پر پیچ و خم بدون مشکل حرکت کند.

## نوآوری های ارائه شده

نوآوری های انجام شده در این پروژه را در قالب سه بخش از پیاده سازی های انجام شده بررسی می کنیم.

### کتابخانه شامل محیط های مورد نظر

### کتابخانه شامل چند الگوریتم ساده یادگیری تقویتی عمیق

### پیاده سازی، آموزش و تنظیم عامل هایی برای حل چالش های محیطی موردنظر

## ساختار پایان نامه

# ساختار بيان مطالب

# سبك نگارش

# مرجع دهی

فهرست مرجع‏ها

|  |  |
| --- | --- |
| [1] | E. C. a. Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning," 2016--2021. [Online]. Available: http://pybullet.org. |
| [2] | G. B. a. V. C. a. L. P. a. J. S. a. J. S. a. J. T. a. W. Zaremba, "OpenAI Gym," 2016. [Online]. Available: https://github.com/openai/gym. |
| [3] | B. Ellenberger, "PyBullet Gymperium," 2018--2019. [Online]. Available: https://github.com/benelot/pybullet-gym. |
| [4] | N. Russo, "rex-gym," 2018. [Online]. Available: https://github.com/nicrusso7/rex-gym. |
| [5] | S. Tunyasuvunakool, A. Muldal , Y. Doron , S. Liu , S. Bohez , J. Merel , T. Erez , T. Lillicrap , N. Heess and Y. Tassa, "dm\_control: Software and tasks for continuous control," *Software Impacts,* p. 100022, 2020. |
| [6] | P. a. H. C. a. K. O. a. N. A. a. P. M. a. R. A. a. S. J. a. S. S. a. W. Y. a. Z. P. Dhariwal, "OpenAI Baselines," GitHub, 2017. |
| [7] | A. a. R. A. a. E. M. a. G. A. a. K. A. a. T. R. a. D. P. a. H. C. a. K. O. a. N. A. a. P. M. a. R. A. a. S. J. a. Hill, "Stable Baselines," *GitHub repository,* 2018. |

1. Environment [↑](#footnote-ref-2)
2. Algorithm [↑](#footnote-ref-3)
3. Python [↑](#footnote-ref-4)
4. Tensorflow [↑](#footnote-ref-5)
5. Keras [↑](#footnote-ref-6)
6. Gym [↑](#footnote-ref-7)
7. Pytorch [↑](#footnote-ref-8)
8. Pip [↑](#footnote-ref-9)
9. Bullet [↑](#footnote-ref-10)
10. Gym [↑](#footnote-ref-11)
11. Pytorch [↑](#footnote-ref-12)
12. Bullet [↑](#footnote-ref-13)
13. C++ [↑](#footnote-ref-14)
14. C [↑](#footnote-ref-15)
15. pybullet [↑](#footnote-ref-16)
16. URDF [↑](#footnote-ref-17)
17. SDF [↑](#footnote-ref-18)
18. XML [↑](#footnote-ref-19)
19. Dynamics [↑](#footnote-ref-20)
20. Gym [↑](#footnote-ref-21)
21. MuJoCo Environments: MuJoCo stands for Multi-joint dynamics with Contact [↑](#footnote-ref-22)
22. Robotics [↑](#footnote-ref-23)
23. Biomechanics [↑](#footnote-ref-24)
24. Graphics [↑](#footnote-ref-25)
25. Animation [↑](#footnote-ref-26)
26. DeepMind [↑](#footnote-ref-27)
27. Pybullet Gymperium [↑](#footnote-ref-28)
28. Tensorforce [↑](#footnote-ref-29)
29. Rex-Gym [↑](#footnote-ref-30)
30. DM\_Control [↑](#footnote-ref-31)
31. OpenAi [↑](#footnote-ref-32)
32. A2C: Advantage Actor Critic [↑](#footnote-ref-33)
33. ACER: Sample Efficient Actor-Critic with Experience Replay [↑](#footnote-ref-34)
34. ACKTR: Actor Critic with Kronecker-factored Trust Region [↑](#footnote-ref-35)
35. DDPG: Deep Deterministic Policy Gradient [↑](#footnote-ref-36)
36. DQN: Deep Q-Network [↑](#footnote-ref-37)
37. GAIL: Generative Adversarial Imitation Learning [↑](#footnote-ref-38)
38. HER [↑](#footnote-ref-39)
39. PPO: Proximal Policy Optimization [↑](#footnote-ref-40)
40. TRPO: Trust Region Policy Optimization [↑](#footnote-ref-41)
41. Stable Baselines [↑](#footnote-ref-42)
42. SAC: Soft Actor Critic [↑](#footnote-ref-43)
43. TD3: Twin-Delayed Deep Deterministic Policy gradient [↑](#footnote-ref-44)
44. Stable Baselines3 [↑](#footnote-ref-45)
45. Tensorflow Agents [↑](#footnote-ref-46)
46. Contextual Bandits [↑](#footnote-ref-47)
47. RAM [↑](#footnote-ref-48)
48. Action [↑](#footnote-ref-49)