



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده مهندسی برق

پروژه کارشناسی
گرایش مخابرات

ماشین های خودران -
با استفاده از یادگیری تقویتی

نگارش
محمد رضیئی فیجانی

استادان راهنما
دکتر وحید پوراحمدی و دکتر حمیدرضا امین داور

شهریور ۱۳۹۸

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

صفحه فرم ارزیابی و تصویب پایان نامه - فرم تأیید اعضاء کمیته دفاع

در این صفحه فرم دفاع یا تأیید و تصویب پایان نامه موسوم به فرم کمیته دفاع - موجود در پرونده آموزشی - را قرار دهید.

نکات مهم:

- نگارش پایان نامه/رساله باید به **زبان فارسی** و بر اساس آخرین نسخه دستورالعمل و راهنمای تدوین پایان نامه های دانشگاه صنعتی امیرکبیر باشد.(دستورالعمل و راهنمای حاضر)
- رنگ جلد پایان نامه/رساله چاپی کارشناسی، کارشناسی ارشد و دکترا باید به ترتیب مشکی، طوسی و سفید رنگ باشد.
- چاپ و صحافی پایان نامه/رساله بصورت **پشت و رو(دورو)** بلامانع است و انجام آن توصیه می شود.

به نام خدا

تاریخ: شهریور ۱۳۹۸

تعهدنامه اصالت اثر



اینجانب **محمد رضیئی فیجانی** متعهد می‌شوم که مطالب مندرج در این پایان‌نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی اساتید دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مآخذ ذکر گردیده است. این پایان‌نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان‌نامه متعلق به دانشگاه صنعتی امیرکبیر می‌باشد. هرگونه استفاده از نتایج علمی و عملی، واگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان‌نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مآخذ بلامانع است.

محمد رضیئی فیجانی

امضا

نویسنده پایان نامه، در صورت تمایل میتواند برای پاسخگویی پایان نامه خود را به شخص
یا اشخاص و یا ارگان خاصی تقدیم نماید.

سپاس‌گزاری

نویسنده پایان‌نامه می‌تواند مراتب امتنان خود را نسبت به استاد راهنما و استاد مشاور و یا دیگر افرادی که طی انجام پایان‌نامه به نحوی او را یاری و یا با او همکاری نموده‌اند ابراز دارد.

محمدرضیٰ فغانی
شهریور ۱۳۹۸

چکیده

در این قسمت چکیده پایان نامه نوشته می‌شود. چکیده باید جامع و بیان‌کننده خلاصه‌ای از اقدامات انجام‌شده باشد. در چکیده باید از ارجاع به مرجع و ذکر روابط ریاضی، بیان تاریخچه و تعریف مسئله خودداری شود.

واژه‌های کلیدی:

کلیدواژه اول، ...، کلیدواژه پنجم (نوشتن سه تا پنج واژه کلیدی ضروری است)

فهرست مطالب

عنوان

صفحه

۱	یادگیری تقویتی با استفاده از gym	۱
۱-۱	معرفی مفاهیم یادگیری تقویتی	۲
۲-۱	معرفی OpenAI gym	۲
۱-۲-۱	مقدمه	۲
۲-۲-۱	نصب	۲
۲	معرفی یادگیری تقویتی	۳
۱-۲	مقدمه	۴
۱-۱-۲	جایگاه یادگیری تقویتی در یادگیری ماشین	۴
۲-۱-۲	وجه تمایز یادگیری تقویتی از دیگر الگوهای یادگیری ماشین	۵
۳-۱-۲	عامل و محیط	۵
۴-۱-۲	حالت	۶
۵-۱-۲	مشاهده پذیری ^۱	۸
۳	پیشنیازهای نصب و معرفی قسمت های مختلف	۹
۱-۳	نرم افزارهای کلی	۱۰
۲-۳	پیشنیازهای پایتون	۱۱
۳-۳	معرفی دقیق تر اجزای کلی	۱۲
۱-۳-۳	معرفی نرم افزار پری اسکن ^۲	۱۲
۲-۳-۳	فرمت های فایل های خروجی	۱۳
۳-۳-۳	نصب موتور متلب ^۳	۱۴
۴-۳	معرفی دقیق تر پیشنیازهای پایتون	۱۵
۱-۴-۳	بسته های کمکی	۱۵
۲-۴-۳	بسته gym	۱۵

^۱Observability

^۲PreScan

^۳Matlab Engine

۴	راه اندازی و توضیح مختصری بر الگوریتم	۱۸
۵	فنی	۲۰
۶	شبیه سازی و نتایج	۲۲
۱-۶	راه اندازی	۲۳
	منابع و مراجع	۲۵
	فهرست اختصارات	۲۶
	واژه نامه انگلیسی به فارسی	۲۷
	واژه نامه فارسی به انگلیسی	۲۹

صفحه	شکل	فهرست اشکال
۴	۱-۲
۶	۲-۲
۷	۳-۲
۱۱	۱-۳	تقسیم بندی وظایف اصلی کد پایتون
۱۲	۲-۳	آیکون های اضافه شده بر روی محیط دسکتاپ پس از نصب پری اسکن
۱۲	۳-۳	پنل مدیریت نرم افزار پری اسکن
۱۳	۴-۳	صفحه گرافیکی محیط پری اسکن

فهرست جداول

صفحه

جدول

۱۴ توضیحات فرمت فایل خروجی	۱-۳
۱۶	۲-۳

فهرست نمادها

نماد	مفهوم
\mathbb{R}^n	فضای اقلیدسی با بعد n
\mathbb{S}^n	کره n یکه بعدی
M^m	خمینه m -بعدی M
$\mathfrak{X}(M)$	جبر میدان‌های برداری هموار روی M
$\mathfrak{X}^1(M)$	مجموعه میدان‌های برداری هموار یکه روی (M, g)
$\Omega^p(M)$	مجموعه p -فرمی‌های روی خمینه M
\mathcal{Q}	اپراتور ریچی
\mathcal{R}	تانسور انحنای ریمان
ric	تانسور ریچی
L	مشتق لی
Φ	۲-فرم اساسی خمینه تماسی
∇	التصاق لوی-چویتای
Δ	لاپلاسین ناهموار
∇^*	عملگر خودالحاق صوری القا شده از التصاق لوی-چویتای
g_s	متر ساساکی
∇	التصاق لوی-چویتای وابسته به متر ساساکی
Δ	عملگر لاپلاس-بلترامی روی p -فرم‌ها

فصل اول

یادگیری تقویتی با استفاده از gym

۱-۱ معرفی مفاهیم یادگیری تقویتی

۲-۱ معرفی OpenAI gym

۱-۲-۱ مقدمه

پروژه gym از قوی ترین پروژه های Open AI^۱ می باشد.

۲-۲-۱ نصب

¹<https://github.com/openai>

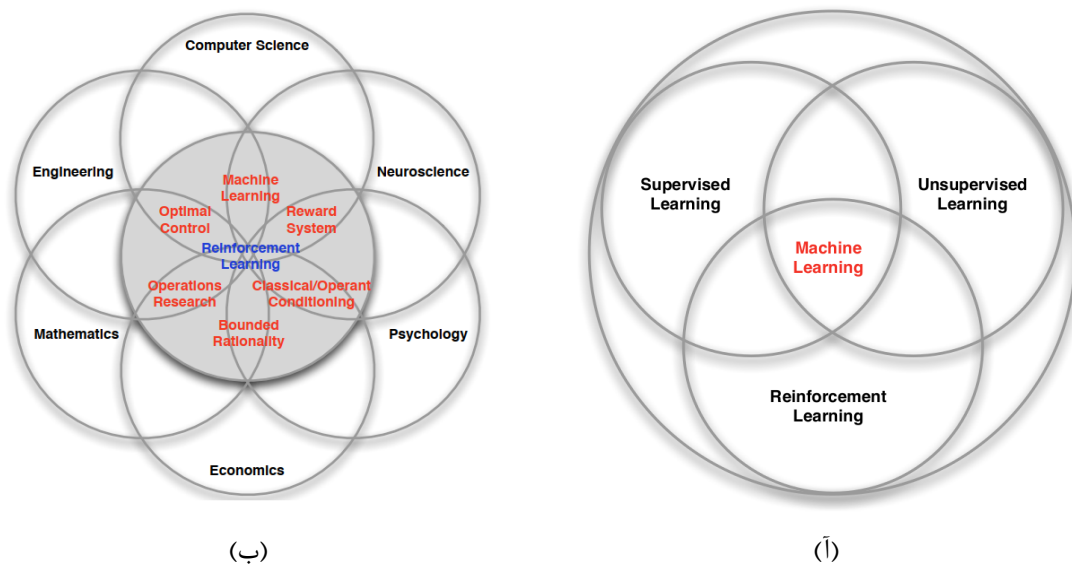
فصل دوم

معرفی یادگیری تقویتی

۱-۲ مقدمه

۱-۱-۲ جایگاه یادگیری تقویتی در یادگیری ماشین

بسیاری از صاحب نظران یادگیری ماشین^۱ را به سه دسته تقسیم می‌کنند: (۱) یادگیری با ناظر^۲ (۲) یادگیری بدون ناظر^۳ یا خوشه بندی^۴ (۳) یادگیری شبه ناظر^۵ در این میان، یادگیری تقویتی^۶ را بعضی ها دسته چهارم می‌دانند و بعضی دیگر آن را در دسته سوم قرار می‌دهند. بر اساس دسته‌بندی گروه دوم شکل ۱-۲(آ) رسم شده است. همچنین شکل ۱-۲(ب) کاربرد یادگیری تقویتی را در علوم مختلف نشان می‌دهد.



شکل ۱-۲:

¹Machine Learning²Supervised Learning³Unsupervised Learning⁴Clustering⁵Semi-Supervised Learning⁶Reinforcement Learning

۲-۱-۲ چه چیزی یادگیری تقویتی را با دیگر الگوهای یادگیری ماشین متمایز می‌کند؟

این سوال از آن جهت حایز اهمیت است که بیان می‌کند چرا ما به سراغ الگوی یادگیری تقویتی رفته‌ایم. پاسخ ملاحظات زیر است.

آ) هیچ ناظر^۷ وجود ندارد و صرفاً امتیازها^۸ وجود دارند.

ب) فیدبک^۹ همراه با تاخیر است و به صورت همزمان رخ نمی‌دهد.^{۱۰}

ج) مفهوم زمان واقعا مطرح است و یک ترتیب خاص از داده‌ها داریم. شکل ۲-۳ این توالی زمانی را نشان می‌دهد.

یادگیری تقویتی (RL^{۱۱}) بر اساس فرضیه امتیازها^{۱۲} پایه‌گذاری می‌شود.

تعریف ۲-۱-۱ (فرضیه امتیازها). همه اهداف می‌توانند براساس بیشینه کردن مقدار میانگین تجمعی امتیازها توصیف کرد.

ممکن است این عبارت کمی عجیب بنظر برسد اما در بسیاری از مسایل که به صورت برد و باخت و به نوعی دو حالت مطلوب و نامطلوب دارند، می‌توان در ساده‌ترین حالت مقدار $+1$ را برای برد و -1 را برای باخت در نظر گرفت.

۲-۱-۳ عامل و محیط

این مفهوم بسیار مفهوم مهمی می‌باشد و بارها از آن در این پروژه یاد شده است. در مسایل یادگیری تقویتی یک **عامل**^{۱۳} وجود دارد که در یک **محیط**^{۱۴} در حال تعامل است. محیط می‌تواند محیط اطراف عامل باشد و یا هر چیزی که عامل با آن در تعامل است. [۱]

⁷Supervisor

⁸Reward

⁹Feedback

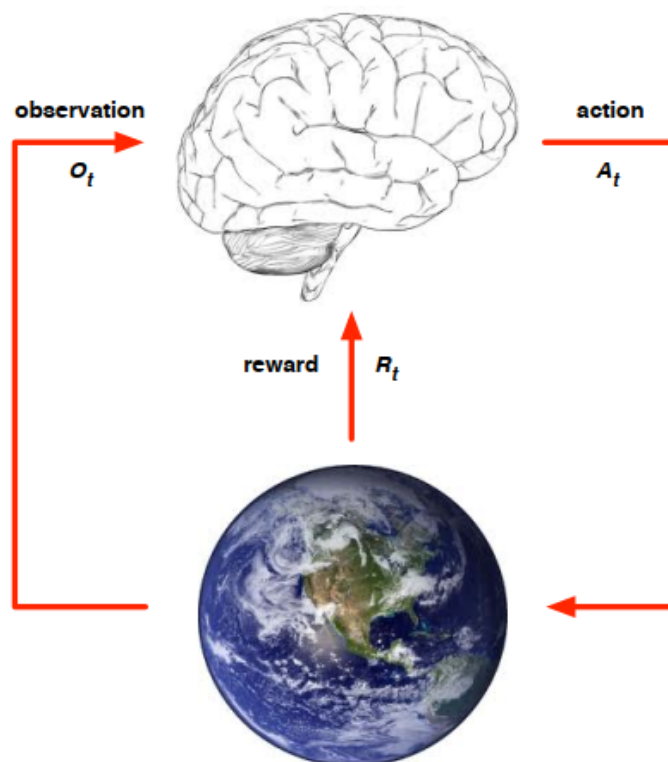
^{۱۰}در مورد علت تاخیر در ادامه توضیح داده خواهد شد.

¹¹Reinforcement Learning

¹²Reward Hypothesis

¹³Agent

¹⁴Environment



شکل ۲-۲:

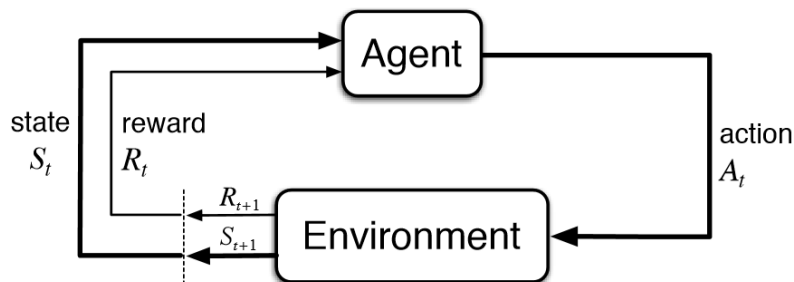
این تعامل به این صورت است که عامل که در ابتدا یک حالت^{۱۵} اولیه دارد، یک حرکت^{۱۶} بر روی محیط در زمان t انجام می‌دهد. محیط مقدار حرکت در زمان t را دریافت می‌کند و سپس محیط در زمان $t + 1$ دو اطلاعات مهم را بر می‌گرداند. (آ) مشاهده^{۱۷} (ب) امتیاز

شکل ۲-۲ و ۳-۲ این تعامل را نشان می‌دهد. لازم به ذکر است که در پایان هر مرحله^{۱۸} مقدار t یک واحد افزایش می‌یابد.

۴-۱-۲ حالت

در بخش قبل تعریف مناسبی از حالت ارایه نشد. برای این تعریف ابتدا مفهوم تاریخچه^{۱۹} ارایه می‌شود و از روی آن حالت تعریف خواهد شد.

¹⁵State¹⁶Action¹⁷Observation¹⁸Step¹⁹History



شکل ۲-۳:

تعریف ۲-۱-۲ (تاریخچه). به سری شامل مشاهده، حرکت و امتیاز می‌باشد:

$$H_t = O_1, R_1, A_1, \dots, A_{t-1}, O_{t-1}, R_t$$

با این تعریف حالت را می‌توان به شکل زیر تعریف کرد.

تعریف ۳-۱-۲. **حالت** اطلاعاتی است که در محاسبات برای آن که در بعد چه اتفاقی بیافتد، استفاده می‌شود. به عبارت دیگر حالت تابعی از تاریخچه می‌باشد.

$$S_t = f(H_t)$$

دو نوع حالت وجود دارد.

(آ) **حالت محیط**^{۲۰} که با علامت S_t^e نشان داده می‌شود. اطلاعات نهان محیط را نشان می‌دهد و معمولاً برای عامل به‌طور کامل دیده نمی‌شود. حتی اگر برای عامل مشاهده‌پذیر نیز باشد، ممکن است اطلاعات کاملاً بی‌ربطی را همراه داشته باشد.

(ب) **حالت عامل**^{۲۱} که با علامت S_t^a نشان داده می‌شود. که برابر است با هر اطلاعاتی که عامل برای رسیدن به حرکت بعدی با استفاده از الگوریتم‌های RL استفاده می‌کند.

بنابراین در تعریف ۳-۱-۲ مناسب‌تر است بجای واژه حالت از حالت عامل استفاده شود. بنابراین:

$$S_t^a = f(H_t)$$

²⁰Environment State

²¹Agent State

یادداشت ۲-۱-۴. از این پس در سراسر این پایان نامه هر جا صحبت از حالت شد منظور همان حالت عامل است.

تعریف ۲-۱-۵. یک حالت S_t مارکوف^{۲۲} است اگر و تنها اگر:

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, \dots, S_t]$$

در یک حالت مارکوف^{۲۳} آینده از گذشته مستقل است و فقط به زمان حال وابسته است. و این به این معناست که حالت از لحاظ آماری برای توصیف آینده کافی است.

نکته ۲-۱-۶. حالت محیط S_t^e مارکوف است. همچنین تاریخچه نیز مارکوف است.

۲-۱-۵ مشاهده پذیری

²²Markov

²³Markov State

فصل سوم

پیشنهاد های نصب و معرفی قسمت های مختلف

۳-۱ نرم افزارهای کلی

در این پروژه از جهت آنکه نسخه قبلی و پیشینی برای آن نبوده است، به ناچار می‌بایست که کد آن از صفر تا صد آن به صورت دستی نوشته شود. از این‌رو، پیچیدگی‌های بسیار فراوان را به طور خاص در پی داشت. ابزارهای زیادی نیز بنابه شرایط در آن استفاده شد که ارتباط بین آن ابزارها و اجزاء بر این پیچیدگی پیاده سازی طرح افزوده بود.

ابزارهای اصلی و کلی که در این پروژه استفاده شده بود، عبارتند از:

- نرم افزار پری اسکن ، نسخه 8.5.0

- نرم افزار متلب^۱ ، نسخه R2017b

- زبان برنامه نویسی پایتون ، نسخه 3.6.9

بنابراین برای راه اندازی مجدد کد این پروژه لازم است که موارد بالا روی کامپیوتر شخص به صورت کامل نصب باشد.

همچنین لازم به ذکر است که برخی ابزارات دیگر نیز در این پروژه استفاده شده است که احتمالاً با نصب موارد بالا دیگر نیازی به نصب آن‌ها به صورت جداگانه نیست. هدف این ابزارها ایجاد اتصال بین اجزای اصلی گفته شده است. این گروه شامل موارد زیر هستند:

- سیمولینک^۲ ، جهت اتصال بین متلب و پری اسکن

- شبکه UDP^۳ ، جهت اتصال داده های پویا^۴ بین پایتون و سیمولینک

- موتور متلب ، جهت اتصال داده های ساکن^۵ بین پایتون و سیمولینک

در این فصل جزئیات بیشتری در مورد لزوم و دلیل استفاده از این ابزارها بررسی می‌شود.

¹ Matlab

² Simulink

³ برای این منظور از ماژول socket در پایتون استفاده شده است.

⁴ Dynamic Data

⁵ Static Data



شکل ۳-۱: تقسیم‌بندی وظایف اصلی کد پایتون

۳-۲ پیشنیازهای پایتون

یادداشت ۳-۲-۱. کد پایتون در این پروژه شامل دو قسمت کلی زیر می‌شود. این دو دسته در شکل ۳-۱ مشخص هستند.

۱. دسته اول مربوط به آن بخش از پروژه است که وظیفه اصلی آن ارتباط پیدا کردن با محیط متلب و پری اسکن و ایجاد یک نوع واسط کاربری است. گرفتن و فرستادن اطلاعات مخصوص این قسمت است.

۲. دسته دوم با محیط و نحوه ارتباط آن کاری ندارد و تمرکز خود را بر روی الگوریتم خود که در این جا از الگوریتم‌های یادگیری تقویتی استفاده شده است، قرار داده است.

دسته اول (سمت چپ تصویر ۳-۱) به پکیج‌های زیر احتیاج دارد:

- matlab.engine
- os و time
- numpy
- gym
- pandas
- socket

اگر از آناکوندا^۶ برای پایتون استفاده می‌کنید غیراز دو بسته gym و matlab.engine به صورت پیش‌فرض نصب شده اند در صورت عدم نصب آن‌ها را با استفاده از pip^۷ می‌توان نصب کرد. بسته gym که در این فصل به تفصیل در مورد آن بحث شده است، به راحتی با همان دستور pip نصب می‌شود. اما نصب matlab.engine یا همان موتور متلب متفاوت است و نمی‌توان آن را نیز به همان روش نصب کرد.

دسته دوم شامل بسته‌های زیر است:

^۶Anaconda

^۷مثلاً بسته numpy را با استفاده از دستور pip install numpy نصب می‌توان کرد.

• gym[atari] یا gym[all]

• tensorflow

• stable-baseline

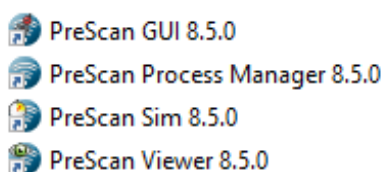
این بسته ها در لایه الگوریتم استفاده شده است. (در مورد این لایه در فصل ۴؟ بیشتر صحبت خواهد شد). هر سه تایی این بسته ها با همان دستور pip به راحتی نصب می شوند.

۳-۳ معرفی دقیق تر اجزای کلی

در این قسمت میخواهیم سه نرم افزار کلی این پروژه را از نگاهی نزدیک تر بشناسیم که عبارتند از :
(۱) نرم افزار پری اسکن (۲) متلب (۳) پایتون

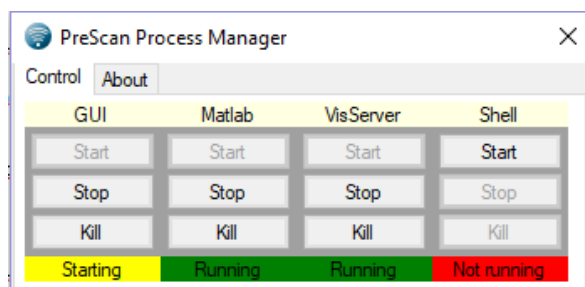
۱-۳-۳ معرفی نرم افزار پری اسکن

پس از دانلود و نصب نسخه 8.5.0 این نرم افزار چهار آیکون مانند شکل ۲-۳ به محیط دسکتاپ اضافه می کند. اصلی ترین آن ها PreScan Proccess Manager 8.5.0 نام دارد.



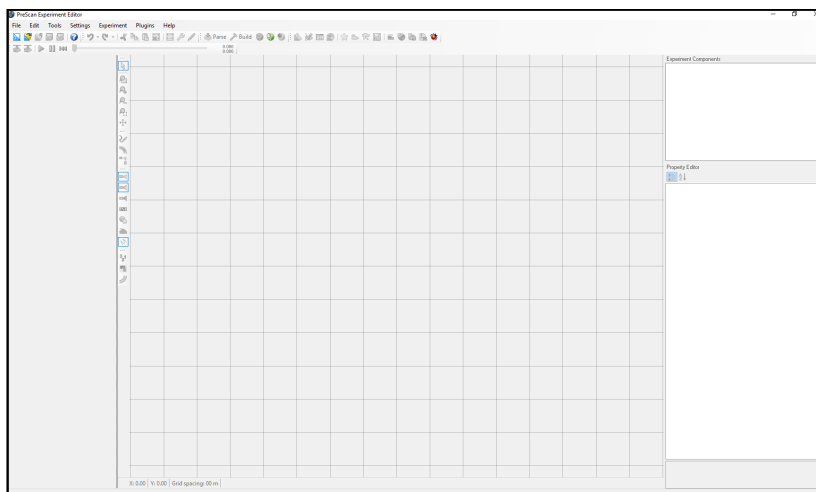
شکل ۲-۳: آیکون های اضافه شده بر روی محیط دسکتاپ پس از نصب پری اسکن

با انتخاب آن صفحه ای مانند زیر باز می شود.



شکل ۳-۳: پنل مدیریت نرم افزار پری اسکن

این پنجره شامل گزینه های زیر است:



شکل ۳-۴: صفحه گرافیکی محیط پری اسکن

• Matlab

• GUI

• Shell

• VisServer

برای ایجاد یک محیط جدید باید GUI را استارت کرد. پس از مدتی صفحه ای مانند شکل ۳-۴ باز می شود.

پس از ایجاد مدل ها و ذخیره آن، فایل های **.pex و **.pb و *_cs.slx ساخته می شود.^۸ جهت استفاده از فایل سیمولینک باید در شکل ۳-۳ متلب را استارت کنید.

نکته ۳-۳-۱. برای اجرای فایل های سیمولینک خروجی، لازم است که متلب را فقط و فقط با استفاده از نرم افزار پری اسکن و با استفاده از پنل مدیریت نرم افزار معرفی شده در شکل ۳-۳ باز شود. در صورتی که به صورت مستقیم این کار انجام شود، به مشکل منتهی می شود.

دو قسمت دیگر نیز در شکل ۳-۳ وجود دارد که نیازی به استارت کردن آن ها نیست و خودشان در صورت لزوم به صورت خودکار فراخوانی می شوند.

۳-۳-۲ فرمت های فایل های خروجی

نرم افزار پری اسکن پس از ایجاد یک محیط جدید، فایل ها و پوشه های بسیار زیادی را ایجاد می کند. اما در خارج آن پوشه ها ۳ فایل وجود دارد که پسوند آن ها **.pex و **.pb و *_cs.slx می باشد.

^۸علامت * به معنای یک اسم مشترک در این سه فایل استفاده شده است.

علامت ** همان اسم پروژه ای است که ایجاد کرده ایم. هر یک از این فایل ها به یک بلوک از شکل ۳-۳ مربوط می شود.

فرمت فایل	توضیحات
** .pex	این فایل مربوط به اولین بلوک شکل ۳-۳ است و ارتباط مستقیم با GUI دارد. برای تغییر محیط گرافیکی باید این فایل را باز کرد.
** .pb	این فایل برخی از اطلاعات فایل ** .pex را در اختیار دارد و با تغییر آن فایل این فایل نیز عوض می شود. این فایل حاوی اطلاعات استاتیک محیط ایجاد شده است و مهم ترین کاربرد آن در بلوک موتور متلب که در شکل ۳-۳ نشان داده شده است می باشد. پایتون از طریق این فایل این اطلاعات را دریافت می کند.
**_cs.slx	این فایل سیمولینک است که برای کار کردن با آن باید از پنل مدیریت شکل ۳-۳ استفاده کرد. این فایل پس از ایجاد از فایل ** .pex مستقل می شود. این فایل خود قابلیت تغییر دارد و می توان بلوک های آن را در محیط سیمولینک تغییر داد و بلوک های دیگری به آن افزود. در صورتی که فایل ** .pex تغییر کند، این امکان را نیز دارد که از داخل خود سیمولینک با فشردن دکمه ای این تغییرات جدید اعمال شود بدون آن که به تغییرات خود کاربر لطمه ای وارد شود. در این پروژه این فایل، تغییرات بسیاری را تجربه کرد.

جدول ۳-۱: توضیحات فرمت فایل خروجی

جدول ۳-۱ توضیحات لازم را جهت آشنایی با این خروجی ها آورده است. همچنین در بخش ۳-۳ در مورد فایل *_cs.slx توضیحات دقیق تری در مورد جزئیات آن گفته خواهد شد.

۳-۳-۳ نصب موتور متلب

برای نصب موتور متلب ابتدا نیاز است که به متلب به طور کامل در سیستم نصب باشد. پس از نصب متلب، محیط Command prompt (admin) را باز کنید و با توجه به نسخه و محل نصب متلب خود به آدرس زیر بروید.

```
<matlabroot>\extern\engines\python
```

مثلا برای Matlab R2017b که در محل پیش فرض خود نصب شده باشد این کار با استفاده از دستور زیر انجام می شود.

```
cd C:\Program Files\MATLAB\R2017a\extern\engines\python
```

در این پوشه یک فایل به نام `setup.py` موجود می باشد. این فایل را با استفاده از دستور

```
python setup.py install
```

در همان محیط `cmd` اجرا کنید.

یادداشت ۳-۲-۲. توجه داشته باشید که باید نسخه متلب و پایتون شما باید با یکدیگر سازگار باشند. برای بررسی این موضوع اگر فایل `setup.py` را با اسنفاده از یک ادیتور باز کنید، یک آرایه به نام `_supported_versions` در آن خواهید دید. مقادیر این آرایه، نسخه هایی از پایتون را نشان می دهد که توسط نسخه متلب شما پشتیبانی میشود مثلاً در این مورد، با توجه به خط زیر نسخه های ۲/۷، ۳/۴، ۳/۵ و ۳/۶ پایتون پشتیبانی می شود. در غیر این صورت باید نسخه سازگار متلب و یا پایتون را نصب کنید.

```
_supported_versions = ['2.7', '3.4', '3.5', '3.6']
```

۴-۳ معرفی دقیق تر پیشنهاد های پایتون

۱-۴-۳ بسته های کمکی

این بسته ها نقش حیاتی ندارند و برای برخی از موارد استفاده شده اند. این موارد در جدول ۲-۳ آمده است.

۲-۴-۳ بسته gym

معرفی

بسته `gym` که توسط `OpenAI` توسعه یافته است. این ابزار فوق العاده این امکان را برای محققین علوم کامپیوتر حرفه ای و یا آماتور فراهم می کند که انواع الگوریتم های یادگیری تقویتی (RL) را بر روی کار خود تست کنند. همچنین پتانسیل این را دارد که محققین محیط خود را بر روی این بسته توسعه دهند. هدف از ایجاد این بسته، استاندارد سازی محیط و نوعی نقطه تراز^{۱۴} برای پژوهش های RL محسوب می شود. [۲]

در حقیقت می توان این بسته را در وسط شکل ۱-۳ جای داد. جایی که لایه محیط و لایه الگوریتم^{۱۵}

¹⁴Bechmark

¹⁵Algorithm

نام بسته	دلیل استفاده	روش نصب
numpy	ایجاد ماتریس برای فضای حرکت ^۹ و فضای مشاهده ^{۱۰}	<code>pip install numpy</code>
time	جهت ایجاد تاخیر و سقف زمانی ^{۱۱}	<code>pip install time</code>
os	برای بستن پنجره های باز شده پس از اجرا	<code>pip install os</code>
pandas	برای چاپ اطلاعات آماری امتیاز های بدست آمده در پایان هر اپیزود ^{۱۲}	<code>pip install pandas</code>
socket	برقراری ارتباط با متلب و فرستان و دریافت کردن داده های پویا	<code>pip install socket</code>
tensorflow	برای لایه الگوریتم و استفاده از الگوریتم های یادگیری تقویتی عمیق ^{۱۳}	<code>pip install tensorflow</code>

جدول ۳-۲:

به یک دیگر می رسند.

این بسته محیط هایی از پیش ساخته شده دارد. نام این بسته ها در لیست زیر آمده است. ^{۱۶} اکثر این محیط ها نوعی بازی هستند که عامل سعی در یادگیری آن محیط ها دارد.

- Pong-v0
- MsPacman-v0
- SpaceInvaders-v0
- Seaquest-v0
- LunarLanderV2
- Reacher-v2
- FrozenLake-v0
- CartPole-v0
- Pendulum-v0
- MountainCar-v0
- MountainCarContinuous-v0
- BipedalWalker-v2
- Humanoid-V1
- Riverraid-v0
- Breakout-v0

نصب

برای نصب نسخه کمینه این نرم افزار با همان روش pip به راحتی می توان نرم افزار مورد نظر را نصب کرد. [۳] این نسخه کمینه برای لایه محیط کافی می باشد. اما اگر بخواهیم بخش الگوریتم را با استفاده از کتابخانه های دیگری مانند stable-baseline نوشت. نیازمند نسخه جامع تری از gym می باشد.

^{۱۶} جدول کامل در سایت <https://github.com/openai/gym/wiki/Table-of-environments> قرار دارد.

یادداشت ۳-۴-۱. پیشنهاد می شود برای نصب نسخه کامل gym و stable-baseline از لینوکس بجای ویندوز استفاده کنید. زیرا در نصب برخی بسته ها ممکن است با مشکل روبرو شوید.

برای نصب کامل این بسته از دستور `pip install gym[all]` استفاده کنید. ممکن است در نصب mujoco به مشکل برخوردید در این صورت دستور `pip install gym[atari]` استفاده کنید. اگر موفق به نصب این بسته نشدید می توانید مراجل نصب آن را با استفاده از [۲] مراجعه کنید.

فصل چهارم

راه اندازی و توضیح مختصری بر الگوریتم

فصل پنجم

فنی

سلام

فصل ششم

شبیه سازی و نتایج

در فصل‌های گذشته در خصوص ابزار هایی که در این پروژه استفاده شده‌اند، صحبت شد و توضیح مفصلی بر چستی آن ابزار ها و ضرورت استفاده از آن‌ها داده شد. اما سوالات بی جوابی نیز ماند که در این فصل به آن‌ها خواهیم پرداخت.

یکی از آن سوالات نحوه راه اندازی کد پروژه می‌باشد و سوال دیگر نتیجه حاصل از شبیه سازی نهایی چگونه است می‌باشد.

۱-۶ راه اندازی

این کد در گیت هاب در دو مخزن^۱ موجود می‌باشد. ابتدا پیش نیاز های لازم را که در بخش ۳ توضیح داده شدند، را نصب کنید. پیشنهاد می‌شود که تمامی نسخه‌های لازم توضیح داده شده در بخش ۳ را در سیستم عامل لینوکس استفاده کنید.

اگر به توصیه استفاده از لینوکس عمل نکرده باشید، می‌توانید با استفاده شبکه که کد در اختیارتان قرار می‌دهد کد را روی دو کامپیوتر ران کنید به طوری که در یک کامپیوتر ویندوز و نرم افزار های گفته شده نصب باشد و روی دیگری لینوکس و پایتون و پیش نیاز های پایتون که در بخش ۳-۴ بررسی شدند نصب باشد.

یادداشت ۱-۱-۶. بهتر است این دو کامپیوتر در یک شبکه داخلی به یک دیگر متصل شده باشند.

حال وارد کامپیوتری شوید که ویندوز بر روی آن نصب است. این کامپیوتر قرار است نقش محیط را برای ما ایجاد کند.

یادداشت ۲-۱-۶. کد های پایتون صرفاً بر روی کامپیوتری که لینوکس دارد اجرا کنید.

که دستور زیر را در ترمینال^۲ خود وارد کنید.

```
1 git clone https://github.com/MohammadRaziei/gym-Prescan.git
2 pip install -e gym-Prescan
```

خط دوم این کد، اختیاری می‌باشد و صرفاً کار را ساده می‌کند. همچنین می‌توان آن را به صورت زیر نیز نوشت:


```
1 pip install git+https://github.com/MohammadRaziei/gym-Prescan
```

¹Repository

²Terminal

سپس وارد مسیر زیر شوید.

gym-Prescan/gym_prescan/envs/PreScan

سپس با استفاده از آیکون ^۳ کلیک کنید. در این صورت بر روی نوار Toolbar این آیکون نیز ظاهر می شود. با فشردن آن، پنجره شکل ۳-۳ باز می شود. بر روی Matlab کلیک کنید تا محیط متلب باز شود. اجرای فایل startup.m برای هنگامی که از کد پایتون بر روی همان سیستم استفاده نمی کنید، اختیاری است.

فایل سیمولینک را باز کرده و

پس حال در سیستم لینکوس خود میتوانید بسته زیر را دانلود کنید.

```
1 git clone https://github.com/MohammadRaziei/gym-Prescan-minimal.git
2 cd gym-Prescan-minimal
```

در این پوشه تعدادی از الگوریتم های معروفی که در حوزه یادگیری تقویتی عمیق نوشته شده است، قرار دارد. در بین این الگوریتم ها دو الگوریتم ⁴DQN و ⁵A2C نسبت به بقیه بهتر جواب داده اند. این الگوریتم ها در بخش ؟؟ و در [۵] توضیح کامل داده شده اند.

^۳ این آیکون پس از نصب نرم افزار پری اسکن بر روی دستکتاپ تشکیل می شود.

⁴Deep Q-Learning

⁵Synchronous Actor Critics

منابع و مراجع

- [1] D. Silver, “UCL course on RL.” <http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html>, 2015.
- [2] G. Hayes, “How to install openai gym in a windows environment.” <https://medium.com/p/338969e24d30>, 2018.
- [3] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “Openai gym,” 2016.
- [4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, second ed., 2018.
- [5] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, “Stable baselines.” <https://github.com/hill-a/stable-baselines>, 2018.

فهرست اختصارات

A

A2C..... Synchronous Actor Critics

D

DQN..... Deep Q-Learning

R

RL..... Reinforcement Learning

واژه‌نامه انگلیسی به فارسی

Environmnet State حالت محیط
Episode اپیزود

F

Feedback بازخورد

H

History تاریخچه

M

Machine Learning یادگیری ماشین
Markov مارکوف
Markov State حالت مارکوف
Matlab نرم‌افزار متلب
Matlab Engine موتور متلب

O

Observability مشاهده‌پذیری
Observation مشاهده
Observation Space فضای مشاهده

A

Action حرکت
Action Space فضای حرکت
Agent عامل
Agent State حالت عامل
Algorithm الگوریتم
Anaconda نرم‌افزار آناکوندا

B

Bechmark نقطه تراز

C

Clustering خوشه بندی

D

Deep یادگیری تقویتی عمیق
Reinforcement Learning
Dynamic Data داده‌های پویا

E

Environment محیط

P

PreScan نرم افزار پری اسکن

R

Reinforcement Learning یادگیری تقویتی

Repository مخزن (گیت هاب)

Reward امتیاز

Reward Hypothesis فرضیه امتیازها

S

Semi-Supervised یادگیری شبه ناظر
Learning

Simulink سیمولینک

State حالت

Step مرحله

Supervised Learning یادگیری با ناظر

Supervisor ناظر

T

Terminal ترمینال

Timeout سقف زمانی

U

Unsupervised Learning یادگیری بدون ناظر

واژه‌نامه فارسی به انگلیسی

د	ا
Dynamic Data داده‌های پویا	Episode اپیزود
	Algorithm الگوریتم
	Reward امتیاز
س	ب
Timeout سقف زمانی	Feedback بازخورد
Simulink سیمولینک	
ع	ت
Agent عامل	History تاریخچه
	Terminal ترمینال
ف	ح
Reward Hypothesis فرضیه امتیازها	State حالت
Action Space فضای حرکت	Agent State حالت عامل
Observation Space فضای مشاهده	Markov State حالت مارکوف
م	حالت محیط
Markov مارکوف	Environmmnet State حالت محیط
Environment محیط	Action حرکت
Repository مخزن (گیت‌هاب)	
Step مرحله	خ
Observation مشاهده	Clustering خوشه بندی

مشاهده‌پذیری Observability
موتور متلب Matlab Engine

ن

ناظر Supervisor
نرم‌افزار آناکوندا Anaconda
نرم‌افزار پری‌اسکن PreScan
نرم‌افزار متلب Matlab
نقطه تراز Bechmark

ی

یادگیری با ناظر Supervised Learning
یادگیری بدون ناظر Unsupervised Learning
یادگیری تقویتی . Reinforcement Learning
یادگیری تقویتی عمیق Deep
Reinforcement Learning
یادگیری شبه ناظر Semi-Supervised
Learning
یادگیری ماشین Machine Learning