به نام حق

راهنمای بیس Agent2D

زمستان ۹۶

مقدمه

یکی از زیر شاخه‌های ربوکاپ، لیگ شبیه‌سازی ۲ بعدی است که تمرکز این لیگ بر روی نرم‌افزار، بستر مناسبی را برای پیاده‌سازی الگوریتم‌های هوش مصنوعی فراهم می‌آورد.

در این محیط شبیه‌سازی شده تلاش شده است بسیاری از اعمال و حسگرهای ربات‌های بازیکن واقعی و شرایط واقعی بازی شبیه‌سازی شود. از این رو هر بازیکن به طور مجزا از بازیکن‌های دیگر، اطلاعاتی را به صورت string و از طریق پروتکل UDP/IP برای سرور شبیه‌ساز ارسال و از آن دریافت می‌کند. اطلاعاتی که هر بازیکن دریافت می‌کند شامل اطلاعات محیط مانند فاصله بازیکن تا نقاط مختلف زمین و مکان نسبی توپ نسبت به بازیکن است که این اطلاعات همراه با خطاست. همچنین هر بازیکن اطلاعاتی مانند چگونه حرکت کردن و یا شوت کردن را برای سرور می‌فرستد.

از وظایف دیگر سرور داوری و پیش بردن زمان است. زمان به صورت گسسته و تحت عنوان cycle معرفی می‌شود. هر سایکل معادل ۱۰۰ میلی‌ثانیه است. در هر سایکل تنها یک دستور kick و یا move قابل اجراست اما دستوراتی مانند تغییر زاویه دید (turn neck) می‌تواند بارها در کنار سایر دستورات در یک سایکل اجرا شود. اگر دستورات جابه‌جایی یک شی (توپ یا بازیکن) در یک سایکل به سرور ارسال شود، شی مربوطه در ابتدای سایکل بعدی جابه‌جا خواهد شد.

نرم‌افزار سرور بازی فوتبال را شبیه‌سازی می‌کند اما آن را به تصویر نمی‌کشد. از این رو نرم‌افزار نمایشگر شبیه‌ساز فوتبال ربوکاپ (RCSS Monitor) به عنوان یک client به سرور شبیه‌ساز متصل شده و وقایع درون بازی را به صورت گرافیکی نمایش می‌دهد.

راهنمای بیس Agent2D

این بیس از دو بخش agent2d و librcsc تشکیل شده است. در هر سیکل از بازی، متد actionImpl از کلاس SamplePlayer که در بخش agent2d قرار دارد اجرا میشود و دستورات لازم برای سرور فرستاده میشود. در ذیل به توضیح کلاس ها و متدهای بخش librcsc میپردازیم که میتوانید در کد خود از آنها استفاده کنید. به عنوان مثال کلاس SamplePlayer حاوی متدی به نام world است که یک شی از نوع کلاس WorldModel برمیگرداند. این کلاس در بخش librcsc تعریف شده و بیانگر حالت کنونی جهان است و درون آن، تمامی اطلاعاتی قرار دارد که شما به آنها دسترسی دارید.

کلاس WorldModel

این کلاس شامل اطلاعاتی از شرایط حال حاضر بازی است. دقت کنید که در هر سیکل این اطلاعات تغییر میکنند و اگر شما میخواهید که به اطلاعات گذشته دسترسی داشته باشید باید خودتان آن را ذخیره کنید. در زیر به طور خلاصه به تعدادی از متدهای این کلاس اشاره میشود. لازم به ذکر است که همه متدها دارای کامنت هستند و با خواندن آنها میتوانید متوجه عملکرد متدها شوید.

متد self

این متد شی ای از نوع کلاس SelfObject برمیگرداند. این کلاس حاوی اطلاعاتی در مورد ایجنت است.

متد ball

این متد شی ای از نوع کلاس BallObject برمیگرداند. این کلاس حاوی اطلاعاتی در مورد توپ است.

متد opponentsFromBall

این متد آرایه ای از بازیکنان حریف (vector<PlayerObject*>) برمیگرداند که به ترتیب فاصله از توپ مرتب شده اند.

متد ourSide

خروجی این متد از نوع SideID است که یک enum است. اگر تیم ما سمت چپ زمین باشد مقدار LEFT یا 1 و اگر سمت راست زمین باشد مقدار RIGHT یا 1- برمیگرداند.

متد theirSide

خروجی این متد نیز از نوع SideID است. اگر تیم حریف سمت چپ زمین باشد مقدار LEFT یا 1 و اگر سمت راست زمین باشد مقدار RIGHT یا 1- برمیگرداند.

متد time

این متد شی ای از نوع کلاس GameTime برمیگرداند. با استفاده از متد cycle تعریف شده در این کلاس، میتوان سیکل جاری را دریافت کرد.

متد accurateLastKickerSide

این متد side تیمی که آخرین بار به توپ ضربه زده است را برمیگرداند.

کلاس AbstractPlayerObject

این کلاس حاوی اطلاعاتی در مورد ایجنت ها است.

متد pos

این متد بردار مکان ایجنت را برمیگرداند.

متد vel

این متد بردار سرعت ایجنت را برمیگرداند.

متد body

این متد زاویه بدن ایجنت را برمیگرداند.

متد distFromBall

این متد فاصله ایجنت از توپ را برمیگرداند.

متد distFromSelf

این متد فاصله یک ایجنت را از ایجنتی که در حال اجرای برنامه است، برمیگرداند.

متد inertiaPoint

این متد با دریافت یک عدد صحیح n ، محاسبه میکند که ایجنت با توجه به سرعت و مکان آن در سیکل جاری، بعد از گذشت n سیکل به طور تخمینی در چه مکانی قرار میگیرد.

متد inertiaFinalPoint

این متد محاسبه میکند که ایجنت با توجه به سرعت و مکان آن در سیکل جاری، به طور تخمینی در چه مکانی متوقف میشود.

کلاس PlayerObject و SelfObject

این کلاس ها هر دو از کلاس AbstractPlayerObject ارث برده اند بنابراین تمام متدهای بالا را دارند.

متد isKickable

این متد چک میکند که آیا توپ در فاصله ای هست که ایجنت بتواند شوت بزند یا نه.

کلاس BallObject

این کلاس حاوی اطلاعاتی در مورد توپ است.

متد pos

این متد بردار مکان توپ را برمیگرداند.

متد vel

این متد بردار سرعت توپ را برمیگرداند.

متد distFromSelf

این متد فاصله توپ را از ایجنتی که در حال اجرای برنامه است، برمیگرداند.

متد inertiaPoint

این متد با دریافت یک عدد صحیح n ، محاسبه میکند که توپ با توجه به سرعت و مکان آن در سیکل جاری، بعد از گذشت n سیکل به طور تخمینی در چه مکانی قرار میگیرد.

متد inertiaFinalPoint

این متد محاسبه میکند که توپ با توجه به سرعت و مکان آن در سیکل جاری، به طور تخمینی در چه مکانی متوقف میشود.

کلاس Vector2D

این کلاس برای کار با مختصات و بردارها نوشته شده است و حاوی متدهایی برای دریافت طول بردار، زاویه بردار، بردار یکه و همچنین متدهایی برای محاسبه فاصله دو نقطه، جمع و تفریق دو بردار، دوران دادن بردار و ... است.

کلاس ServerParam

این کلاس برای نگهداری و دسترسی به پارامترهای مربوط به سرور و بازی که مقدار ثابتی دارند استفاده میشود. این کلاس با پترن singleton نوشته شده است و برای دسترسی به آن باید از متد استاتیک i یا instance استفاده کنید.

متد pitchLength

این متد اندازه طول زمین را برمیگرداند.

متد pitchWidth

این متد اندازه عرض زمین را برمیگرداند.

متد goalWidth

این متد اندازه عرض دروازه را برمیگرداند.

متد ourTeamGoalPos

این متد مختصات دروازه خود را برمیگرداند.

متد theirTeamGoalPos

این متد مختصات دروازه حریف را برمیگرداند.

Action های قابل استفاده

در پوشه librcsc/action نمونه هایی از action هایی که یک ایجنت میتواند انجام دهد، تعریف شده است. همچنین نمونه هایی از استفاده از این action ها در کلاس های Bhv_BasicOffensiveKick و Bhv_BasicMove آورده شده است.

Body_GoToPoint2010

از این action برای حرکت در زمین میتوانید استفاده کنید. سازنده این کلاس سه پارامتر دریافت میکند.
پارامتر اول از نوع Vector2D است و مختصات مقصد را مشخص میکند.
پارامتر دوم از نوع double است و آستانه قابل قبول برای رسیدن به مقصد را مشخص میکند. به عنوان مثال اگر این مقدار ۱ باشد به این معنی است که اگر ایجنت در شعاع یک متری نقطه مقصد (پارامتر اول) قرار بگیرد، به مقصد رسیده است.
پارامتر سوم از نوع double است و حداکثر سرعتی را که ایجنت میتواند داشته باشد، مشخص میکند.

Body_Intercept

از این action برای تصاحب توپ میتوانید استفاده کنید. سازنده این کلاس هیچ پارامتری دریافت نمیکند. با اجرای این action بر اساس مسیر و سرعت حرکت توپ، بهترین نقطه برای سد کردن توپ محاسبه میشود و ایجنت به آن نقطه میرود.

Body_SmartKick

از این action برای شوت کردن توپ میتوانید استفاده کنید. سازنده این کلاس چهار پارامتر دریافت میکند.
پارامتر اول از نوع Vector2D است و مختصات مقصد را مشخص میکند.
پارامتر دوم از نوع double است و سرعت اولیه توپ را مشخص میکند.
پارامتر سوم از نوع double است و آستانه سرعت اولیه توپ را مشخص میکند.
پارامتر چهارم از نوع int است و ماکزیمم تعداد سیکل برای انجام شوت را مشخص میکند.

Body_Dribble2008

از این action برای حمل توپ میتوانید استفاده کنید. این اکشن به این صورت انجام میشود که ابتدا ایجنت با یک عمل kick توپ را به جلو می اندازد و سپس با تعداد مشخصی عمل dash به سمت توپ حرکت میکند.

سازنده این کلاس پنج پارامتر دریافت میکند.
پارامتر اول از نوع Vector2D است و مختصات مقصد را مشخص میکند.
پارامتر دوم از نوع double است و آستانه قابل قبول برای رسیدن به مقصد را مشخص میکند.
پارامتر سوم از نوع double است و سرعت ایجنت در حمل توپ را مشخص میکند.
پارامتر چهارم از نوع int است و تعداد dash ها را مشخص میکند.
پارامتر پنجم از نوع bool است و اگر مقدار آن true باشد ایجنت در هنگام حمل توپ سعی میکند به بازیکنان حریف نزدیک نشود.

Body_ ClearBall2009

از این action برای دور کردن توپ در موقعیت های خطرناک میتوانید استفاده کنید.