



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده مهندسی مکانیک

پایان نامه کارشناسی
مهندسی مکانیک

مدل سازی و حفظ تعادل موتورسیکلت خودران به کمک روش های هوشمند

نگارش
مهدی رحمانی

استاد راهنما
دکتر علی عظیمی

شهریور ۱۴۰۲

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ
اللّٰهُمَّ احْمَدُ عَلٰى مُحَمَّدٍ
وَسَلَّمَ وَرَأَيْتَهُ حَمَاماً

به نام خدا

تاریخ:

تعهدنامه اصالت اثر



اینجانب مهدی رحمانی متعهد می‌شوم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب تحت نظرارت و راهنمایی استادی 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 مأخذ ذکر گردیده است. این پایان نامه 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 پایان نامه 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 پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مأخذ بلامانع است.

مهدی رحمانی

امضا

تقدیم

تقدیم به پدر، مادر و برادر عزیزم که در تمامی این سال‌ها تحصیل را برایم تسهیل نمودند و همواره در سختی‌ها و دشواری‌های زندگی، یاوری دلسوز و فداکار و پشتیبانی محکم برایم بودند.

تقدیر و تشکر

بدینوسیله مراتب قدردانی و امتنان خود را خدمت،
استاد دلسوز و گرانقدر؛ جناب آقای دکتر علی عظیمی که در کمال سعه صدر، با حسن خلق و
فروتنی، از هیچ کمکی در این عرصه بر من دریغ نداشتند،
تمامی دوستان و همراهان که در این سال‌ها در کنار من بودند،
ابراز و از تمامی زحمات آنان تشکر مینمایم.

مهردادی رحمانی

۱۴۰۲ شهریور

چکیده

امروزه، افزایش وسایل نقلیه‌ی شخصی در شهرهای بزرگ، موجب بروز معصل‌های بسیاری از جمله افزایش سوانح رانندگی، ترافیک‌های شهری و مصرف سوخت، شده است. بسیاری از شرکت‌ها برای کاهش این مشکلات، به ساخت وسایل نقلیه‌ی خودران و برقی روی آورده‌اند و به خصوص در زمینه‌ی اتومبیل‌های خودران پیشرفت‌های چشمگیری داشته‌اند. در این میان، به موتورسیکلت‌های خودران به عنوان وسایل نقلیه‌ای با ابعاد و مصرف سوخت کمتر که آن‌ها را برای بسیاری از امور از جمله کارهای امدادی و ارسال بسته‌ها، به گزینه‌ای مناسب تبدیل می‌کند؛ توجه چندانی نشده است.

یک موتورسیکلت برای تبدیل شدن به نسخه‌ی کاملاً خودران، در وهله‌ی اول نیازمند حفظ تعادل می‌باشد. همین موضوع باعث افزایش پیچیدگی فرآیند خودران سازی موتورسیکلت‌ها نسبت به اتومبیل‌های خودران شده است. در این پژوهش به دنبال آن هستیم تا با به کارگیری ابزارهای مختلف مانند چرخ طیار و ژیروسکوپ و یا فرمان موتورسیکلت، و به کمک کنترل‌کننده‌هایی نظیر PID و روش‌های مبتنی بر هوش مصنوعی نظیر یادگیری تقویتی، تعادل موتورسیکلت را حفظ نماییم.

از آنجایی که آزمایش روش‌های مختلف بر روی نمونه‌های واقعی هزینه‌بر است و می‌تواند خساراتی را به همراه داشته باشد، یکی از بهترین راه‌ها استفاده از شبیه‌سازهای دارای موتور فیزیکی می‌باشد. در این پژوهش به کمک مدل‌سازی موتورسیکلت در نرم‌افزار Vortex Studio و اجرای روش‌های کنترلی مذکور، توانستیم تعادل موتورسیکلت را به کمک چرخ طیار، حتی به هنگام عبور از روی موانع و دور زدن موانع بزرگ، حفظ نماییم. پس از انجام این مرحله‌ی دشوار، می‌توان مانند اتومبیل‌های خودران سایر قابلیت‌ها از قبیل مسیریابی، تشخیص علائم راهنمایی و ... را به موتورسیکلت، اضافه نمود.

واژه‌های کلیدی:

موتورسیکلت خودران، یادگیری تقویتی، کنترل کننده PID، Vortex Studio

صفحه

فهرست مطالب

۱	چکیده
۲	فصل اول مقدمه
۳	۱- بیان مسئله و اهداف
۴	۲- تاریخچه موتورسیکلت خودران
۵	۳- ویژگی های مختلف موتورسیکلت خودران
۶	۴- سامانه های حفظ تعادل
۹	فصل دوم آشنایی با نرم افزار Vortex Studio
۱۰	آشنایی با نرم افزار Vortex Studio
۱۰	۱- موتور فیزیکی چیست؟
۱۱	۲- نحوه کار کرد موتور فیزیکی
۱۱	۱-۲-۱- حلقه های موتور فیزیکی
۱۳	۲-۲-۱- انتگرال گیری عددی
۱۴	۲-۲-۲- تشخیص برخوردها
۱۶	۴-۲-۲- سامانه پاسخ برخورد
۱۶	۳-۲- کاربردهای موتور فیزیکی
۱۹	۴-۲- محدودیت های موتور فیزیکی
۲۰	۵-۲- نرم افزار Vortex Studio
۲۰	۱-۵-۲- نحوه کار کرد نرم افزار
۲۲	فصل سوم مدل سازی موتورسیکلت
۲۳	مدل سازی موتورسیکلت
۲۳	۱-۳- مدل سازی دینامیکی
۲۴	۱-۱-۳- مدل سازی ریاضی
۲۶	۲-۱-۳- مدل سازی دوبعدی موتورسیکلت
۲۷	۳-۱-۳- مدل سازی سه بعدی موتورسیکلت
۲۹	۲-۳- مدل سازی در شبیه ساز
۳۰	۱-۲-۳- طراحی مدل سه بعدی موتورسیکلت
۳۱	۲-۲-۳- ایجاد فایل گالری گرافیکی
۳۳	۳-۲-۳- ایجاد مونتاژ

۳۸	تکمیل مکانیزم ۴-۲-۳
فصل چهارم طراحی کنترل کننده	
۴۰	طراحی کنترل کننده
۴۱	۱-۱-۴ - کنترل PID
۴۲	۱-۱-۴ - نحوه عملکرد کنترل PID
۴۳	۱-۱-۴ - نحوه تنظیم ضرایب
۴۵	۱-۱-۴ - پیاده سازی در نرم افزار Vortex Studio
۴۸	۱-۲-۴ - یادگیری تقویتی
۵۴	۱-۲-۴ - یادگیری تقویتی و فرآیند تصمیم گیری مارکوو (MDP)
۵۶	۱-۲-۴ - معماری بازیگر-منتفع
۵۷	۱-۲-۴ - الگوریتم گرادیان سیاست قطعی عمیق
۵۷	۱-۲-۴ - حفظ تعادل موتور سیکلت با یادگیری تقویتی
۶۱	۱-۲-۴ - پیاده سازی در نرم افزار Vortex Studio
۶۴	۱-۳-۴ - ژیروسکوپ
۶۵	۱-۳-۴ - اثر ژیروسکوپی
۶۶	۱-۳-۴ - پیاده سازی در نرم افزار Vortex Studio
فصل پنجم تحلیل نتایج	
۶۸	تحلیل نتایج
۶۹	۱-۱-۵ - آشنایی با موانع مورد استفاده در محیط آزمایش
۷۰	۱-۱-۵ - سطح شیبدار
۷۰	۱-۱-۵ - دست انداز
۷۱	۱-۱-۵ - جعبه بزرگ
۷۲	۱-۱-۵ - دست اندازهای متواالی با ابعاد مختلف
۷۲	۱-۱-۵ - بررسی مدت زمان حفظ تعادل
۷۳	۱-۱-۵ - بررسی زاویه و سرعت انحراف موتور سیکلت
۷۴	۱-۱-۵ - رفتار سامانه بدون افرودن کنترل کننده
۷۵	۱-۱-۵ - رفتار سامانه حین عبور از دو مانع اول در حضور کنترلرها
۷۶	۱-۱-۵ - رفتار سامانه حین چرخاندن فرمان در عبور از مانع سوم
۷۸	۱-۱-۵ - رفتار سامانه در عبور از دست اندازهای متواالی
۷۹	۱-۱-۵ - بررسی سرعت چرخش چرخ طیار
۸۰	۱-۱-۵ - بررسی جمع بندی و پیشنهادات
فصل ششم جمع بندی و پیشنهادات	
۸۲	جمع بندی و پیشنهادات

۱-۶- تلفیق روش‌های آزمایش شده.....	۸۳
۲- استفاده از یادگیری ماشین بر روی داده‌های به دست آمده.....	۸۴
۳- استفاده از کنترل کننده‌های دیگر.....	۸۴
۴- بالابردن دقیقت در شبیه‌سازی.....	۸۵
منابع و مراجع.....	۸۶
Abstract	۸۸

صفحه

فهرست اشکال

۳ شکل ۱-۱ نمایی از موتورسیکلت Lit Motors C-1
۴ شکل ۱-۲ طرح مفهومی موتور سیکلت خودران یاماها و طرح پیاده شدهی BMW
۷ شکل ۱-۳ مدلسازی موتورسیکلت خودمتعادل شده به کمک چرخ طیار
۷ شکل ۱-۴ مدلسازی موتورسیکلت خودمتعادل شده به کمک ژیروسکوپ
۷ شکل ۱-۵ مدلسازی موتورسیکلت خودمتعادل شده به کمک فرمان
۱۱ شکل ۱-۶ حلقه‌ی اجرایی موتور فیزیکی
۱۲ شکل ۲-۱ اعمال نیرو و گشتاور به جسم و خروجی حاصل از آن در موتور فیزیکی
۱۳ شکل ۳-۱ روش‌های انتگرال گیری عددی در موتور فیزیکی
۱۴ شکل ۴-۱ روش‌های سنتی مرزبندی اجسام در فرایند تشخیص برخورد
۱۵ شکل ۵-۱ فرایند تشکیل BVH و تشخیص برخورد در موتور فیزیکی
۱۷ شکل ۶-۱ تصویری از ساخت بازی در موتور فیزیکی Unity
۱۸ شکل ۷-۱ اسکلت‌بندی و تنظیم حرکات یک شخصیت آنیمیشنی
۲۱ شکل ۸-۱ مراحل انجام شده برای شبیه سازی در نرم افزار Vortex Studio
۲۴ شکل ۱-۲ دیاگرام آزاد موتورسیکلت خودمتعادل شده
۲۶ شکل ۲-۱ اعضاء و مفاصل در مدلسازی دو بعدی موتورسیکلت
۲۶ شکل ۳-۱ تعریف اعضاء و مفاصل در مدل دو بعدی
۲۸ شکل ۴-۱ درجات آزادی مدل سه بعدی موتورسیکلت
۲۹ شکل ۵-۱ مدلسازی لاستیک و تاثیر تغییرشکل آن در محل اثر نیرو
۳۰ شکل ۶-۱ مراحل تهیه‌ی مدل در شبیه‌ساز
۳۲ شکل ۷-۱ فایل سه بعدی موتورسیکلت اضافه شده در گالری گرافیکی
۳۲ شکل ۸-۱ تعریف گره‌های گرافیکی برای مشخص کردن اعضای مختلف موتورسیکلت
۳۴ شکل ۹-۱ وارد کردن فایل گرافیکی به فایل مکانیزم
۳۴ شکل ۱۰-۱ ایجاد جسم صلب برای گره‌های گرافیکی
۳۵ شکل ۱۱-۱ ایجاد هندسه‌ی برخورد برای اعضاء
۳۶ شکل ۱۲-۱ ایجاد هندسه‌ی برخورد برای اعضاء
۳۷ شکل ۱۳-۱ ایجاد هندسه‌ی برخورد برای اعضاء
۳۷ شکل ۱۴-۱ ایجاد قید برخورد
۳۸ شکل ۱۵-۱ مفاصل ایجاد شده بین اجزا
۴۱ شکل ۱-۲ دیاگرام کنترل کننده حلقه بسته
۴۲ شکل ۲-۱ اجزای مختلف کنترلر PID

..... ۴۳	شکل ۳-۴ تاثیر بلوک P در خروجی کنترلر.
..... ۴۴	شکل ۴-۴ تاثیر بلوک I در خروجی کنترلر.
..... ۴۵	شکل ۴-۵ تاثیر بلوک D در خروجی کنترلر.
..... ۴۷	شکل ۴-۶ منحنی واکنش فرآیند.
..... ۴۸	شکل ۷-۴ اضافه کردن جرم کنترلی به موتورسیکلت.
..... ۴۹	شکل ۸-۴ ارتباطات موتورسیکلت با وزنهای تعادلی و ورودی و خروجی‌های کد.
..... ۴۹	شکل ۹-۴ ارتباطات موتورسیکلت با وزنهای تعادلی و ورودی و خروجی‌های کد.
..... ۵۰	شکل ۱۰-۴ ارتباطات موتورسیکلت برای کنترل با فرمان.
..... ۵۰	شکل ۱۱-۴ ارتباطات موتورسیکلت برای کنترل با فرمان.
..... ۵۱	شکل ۱۲-۴ اضافه کردن چرخ طیار به مدل موتورسیکلت.
..... ۵۲	شکل ۱۳-۴ اضافه کردن سنسور لایدار به موتورسیکلت.
..... ۵۲	شکل ۱۴-۴ اضافه کردن چرخ طیار به مدل موتورسیکلت.
..... ۵۳	شکل ۱۵-۴ اضافه کردن مکانیزم ساخته شده به فایل صحفه.
..... ۵۳	شکل ۱۶-۴ ارتباط بین دوربین‌ها و بدنی موتورسیکلت در صحفه.
..... ۵۷	شکل ۱۷-۴ ارتباط بین دوربین‌ها و بدنی موتورسیکلت در صحفه.
..... ۵۹	شکل ۱۸-۴ نحوه ارتباط محیط و موتورسیکلت.
..... ۶۰	شکل ۱۹-۴ فرآیند انجام شده در الگوریتم DDPG.
..... ۶۱	شکل ۲۰-۴ معماری شبکه‌های عصبی بازیگر و منتقد.
..... ۶۲	شکل ۲۱-۴ بخش HUD ساخته شده در نرم افزار.
..... ۶۲	شکل ۲۲-۴ ارتباطات بین کد و HUD.
..... ۶۳	شکل ۲۳-۴ رابط ساخته شده برای ارتباط کد و محیط شبیه‌ساز.
..... ۶۳	شکل ۲۴-۴ مدل ساخته شده در نرم افزار برای یادگیری تقویتی.
..... ۶۴	شکل ۲۵-۴ صحفه‌ی تعریف شده برای آموزش موتورسیکلت.
..... ۶۵	شکل ۲۶-۴ اثر ژیروسکوپی بر روی فرفره.
..... ۶۶	شکل ۲۷-۴ حفظ تعادل موتورسیکلت به کمک ژیروسکوپ در نرم افزار.
..... ۶۷	شکل ۲۸-۴ اتصالات و تعیین ورودی‌ها و خروجی‌های کد برای مدل ژیروسکوپ.
..... ۷۰	شکل ۱-۵ نمایی از مانع سطح شیبدار.
..... ۷۱	شکل ۲-۵ نیروهای نرمال واردہ بر لاستیک‌ها در عبور از روی سطح شیبدار.
..... ۷۱	شکل ۳-۵ نمایی از مانع دست انداز و نیروهای نرمال واردہ به لاستیک‌ها.
..... ۷۲	شکل ۴-۵ نمایی از جعبه‌ی بزرگ به عنوان مانع.
..... ۷۳	شکل ۵-۵ نمایی از دستاندازهای متوالی.

صفحه

فهرست جداول

۳۰	جدول ۱-۳ پسوندهای مناسب فایل سه بعدی
۳۱	جدول ۲-۳ پسوندهای مناسب فایل CAD
۳۶	جدول ۳-۳ جرم و جنس هریک از اجسام صلب ایجاد شده
۳۸	جدول ۴-۳ جرم و جنس هریک از اجسام صلب ایجاد شده
۴۷	جدول ۱-۴ جرم و جنس هریک از اجسام صلب ایجاد شده

صفحه

فهرست نمودارها

نمودار ۱-۵ نمودار زاویه‌ی انحراف برحسب زمان در سامانه بدونه کنترل کننده	۷۵
نمودار ۲-۵ نمودار سرعت زاویه‌ای برحسب زمان در سامانه بدونه کنترل کننده	۷۶
نمودار ۳-۵ مقایسه‌ی زاویه‌ی انحراف در عبور از دو مانع اول برای کنترل کننده‌های مختلف	۷۷
نمودار ۴-۵ مقایسه‌ی سرعت زاویه‌ای در عبور از دو مانع اول برای کنترل کننده‌های مختلف	۷۷
نمودار ۵-۵ مقایسه‌ی زاویه‌ی انحراف حین چرخش فرمان در عبور از مانع سوم	۷۸
نمودار ۵-۶ مقایسه‌ی زاویه‌ی انحراف حین چرخش فرمان در عبور از مانع سوم	۷۹
نمودار ۷-۵ زاویه‌ی انحراف موتورسیکلت در روش PID در عبور از دستاندازهای متوالی	۸۰
نمودار ۸-۵ سرعت زاویه‌ای موتورسیکلت در روش PID در عبور از دستاندازهای متوالی	۸۰
نمودار ۹-۵ زاویه‌ی انحراف موتورسیکلت در روش PID در عبور از دستاندازهای متوالی	۸۱

فهرست علائم

علائم لاتین

خطای میان مقدار مطلوب و مقدار مورد انتظار	e
پاداش کسب شده توسط عامل هوشمند	r
تابع زیان	L
حالت	s
تعداد کل گام‌های اجرا	N

علائم یونانی

گشتاور خروجی به چرخ طیار	τ
زاویه‌ی انحراف موتورسیکلت یا همان زاویه‌ی رول	θ
سرعت زاویه‌ای انحراف	$\dot{\theta}$
سرعت زاویه‌ای چرخ طیار	φ
سیاست	π

زیرنویس‌ها

بیانگر ضریب تنااسبی	p
بیانگر ضریب انتگرالی	i
بیانگر ضریب مشتقی	d

فصل اول

مقدمه

مقدمه

۱-۱- بیان مسئله و اهداف

با افزایش جمعیت و وسعت شهرها نیاز به استفاده از وسائل نقلیه روزبه روز افزایش می‌یابد. این موضوع باعث شده است که ترافیک در شهرها به معضلی تبدیل شود که به موجب آن مردم به دنبال استفاده از وسائل نقلیه‌ای با ابعاد کمتر برای عبور راحتتر از ترافیک و رسیدن سریعتر به مقصد، باشند. در بسیاری از موارد، مسئله صرفا جابه‌جایی اشخاص برای کارهای روزانه‌ی زندگی نمی‌باشد و بلکه خواست اصلی آنها ارسال یک بسته از نقطه‌ای به نقطه‌ی دیگر و یا همچنین انتقال وسائل و خدمات امدادی توسط سازمان‌های مربوطه، در زمان کم، به مقصد می‌باشد. موتور سیکلت‌ها به عنوان وسائل نقلیه‌ی دارای دو چرخ و ابعاد نسبتاً کم، میتوانند این خلاصه را پر نمایند.

از طرفی محبوبیت استفاده از موتور سیکلت و بالا رفتن تعداد این وسائل نقلیه در شهرهای بزرگ، افزایش آلودگی هوا را در پی داشته است. به عنوان مثال در شهر تهران بیش از سه میلیون دستگاه موتورسیکلت وجود دارد که ده درصد از آلودگی هوا در این شهر به دلیل آلایندگی این وسائل نقلیه می‌باشد. با توجه به افزایش سطح آلاینده‌ها و همچنین روبه اتمام بودن منابع سوخت‌های فسیلی و روی‌آوردن به انرژی‌های تجدیدپذیر، در صنعت ساخت موتورسیکلت نیز شاهد توسعه موتورسیکلت‌های برقی و الکتریکی هستیم که ضمن کاهش آلایندگی، موجب صرفه‌جویی در هزینه‌های حمل و نقل نیز می‌گردند.

با رواج یافتن تولید موتورسیکلت‌های برقی، امکان توسعه موتورسیکلت‌ها نیز به مراتب بیشتر شده و شرکت‌های بزرگ خودرو و موتورسیکلت سازی به سوی هوشمندسازی این وسائل نقلیه روی آوردند. همچنین در مواردی مانند آنچه پیشتر اشاره شد، نظیر انتقال یک بسته یا خدمات امدادی، لزومی به وجود راننده نمی‌باشد که این امر نیاز به موتور سیکلت خودران را پیش می‌آورد. موتورسیکلت‌های خودران آخرین تکنولوژی موتورسیکلت‌ها هستند. این موتورسیکلت‌ها میتوانند تعادل خود را حتی زمانی که حرکتی ندارند و ساکن هستند، نگه می‌دارند و نیازی به راننده ندارند. در حقیقت مهمترین مرحله

برای تبدیل یک موتورسیکلت به نسخهٔ خودران خود، حفظ تعادل آن می‌باشد. در گام‌های بعدی می‌توان سایر ویژگی‌های یک وسیلهٔ نقلیهٔ خودران را به آن اضافه کرد.

۲-۱- تاریخچه موتورسیکلت خودران

فن آوری موتور سیکلت خود متعادل‌کننده^۱ از نمونه‌های اولیه به سیستم‌های پیشرفته‌ی امروزه تکامل یافته است. اولین موتورسیکلت خود متعادل‌کننده در سال ۱۹۹۹ توسط محققان موسسه فناوری ماساچوست^۲ ساخته شد و به نام Gyrocycle شناخته شد. توسعه دهندگان Gyrocycle از ژیروسکوپ‌ها برای حفظ تعادل در هنگام سواری استفاده کردند و سیستم را از طریق کنترل‌های الکترونیکی پیچیده کنترل نمودند.

در سال ۲۰۰۹ نمونه اولیه جدیدی به نام Lit Motors C-1 معرفی شد که از موتور الکتریکی و ژیروسکوپ برای حفظ تعادل در حالت ایستاده یا در سرعت‌های پایین استفاده می‌کرد. این سیستم از یک سیستم کنترل کامپیوتری برای تنظیم ثبات و تعادل استفاده می‌کرد. در شکل ۱-۱ نمایی از آن مشخص است.



شکل ۱-۱ نمایی از موتورسیکلت Lit Motors C-1

از آن زمان، چندین شرکت دیگر نیز موتورسیکلت‌های خود متعادل کننده، از جمله Honda's Riding Assist-e و BMW's Motorrad Vision Next 100 از ترکیبی

¹ Self-balancing Motorcycle

² MIT

از سنسورها، محرک‌ها و سیستم‌های کنترل پیشرفته برای دستیابی به سطوح بالایی از ثبات و تعادل استفاده می‌کنند. تصاویری از این دو موتورسیکلت در شکل ۲-۱ قابل مشاهده است.

امروزه، فناوری موتورسیکلت خود متعادل کننده به سرعت در حال پیشرفت است و می‌توان انتظار داشت که در آینده شاهد نوآوری‌های بیشتری باشیم. با استفاده از هوش مصنوعی و یادگیری ماشینی، موتورسیکلت‌های خودمتعادل حتی کارآمدتر و شهودی‌تر می‌شوند و تجربه رانندگی ایمن‌تر و راحت‌تری را برای همه سطوح رانندگی ارائه می‌کنند [۱].



شکل ۲-۱ طرح مفهومی موتورسیکلت خودران یاماها و طرح پیاده شدهی BMW

۱-۳-۱- ویژگی‌های مختلف موتورسیکلت خودران

یک وسیله نقلیه در حالت کلی لازم است ویژگی‌های مختلفی داشته باشد تا در دسته وسائل نقلیه خودران جای گیرد:

- ۱- درک محیط: اولین مرحله درک محیط است که به‌واسطه رادار، لایدار، موقعیت‌یاب، بینایی ماشین و تجهیزات دیگر انجام می‌شود. با استفاده از این تجهیزات اتومبیل‌ها می‌توانند با محیط بیرون ارتباط بگیرند که این فرایند با الگوبرداری از سیستم بینایی انسان در رایانه شبیه‌سازی می‌شود.
- ۲- سیستم تصمیم‌گیری: این سیستم با ارائه الگوریتم و براساس اطلاعات و داده‌های دریافتی از سیستم درک محیط، تصمیم‌می‌گیرد که وسیله نقلیه در کدام مسیر و با چه سرعتی حرکت کند. تصمیم‌گیری حرکت وسیله نقلیه با شبیه‌سازی بهینه کنترل سیگنال‌های طولی و عرضی آن امکان‌پذیر است.

۳- سیستم کنترل و فرمان: این قسمت با داده‌های سیستم تصمیم‌گیری، تصمیم می‌گیرد چه اتفاقاتی در بخش‌های مکانیکی و دینامیکی وسیله نقلیه انجام شود. درنهایت این تصمیمات به بخش بعدی ارسال می‌شود.

۴- عملگرها: عملگرها که نقش عملی راننده را ایفا می‌کنند و فرمان نهایی را در وسیله نقلیه اجرا می‌کنند؛ باید وسیله نقلیه را در شرایط مختلف هدایت کنند. این شرایط شامل جاده‌های بارانی، یخزده و... می‌شود. از این رو استفاده از قوانین کنترلی تطبیقی یا مقاوم و هوش مصنوعی الزامی است. به عبارت دیگر بخش اول چشم راننده، بخش‌های دوم و سوم ذهن راننده و بخش آخر دست‌ها و پاهای راننده محسوب می‌شوند که اتومبیل را هدایت می‌کنند.

تعاریف فوق، تعاریف جامعی هستند و اگر بخواهیم با دقت بیشتری به قسمت‌های مختلف یک وسیله نقلیه خودران اشاره کنیم، موارد زیادی وجود دارد، با این حال؛ در ادامه به صورت مختصر به مواردی که حتی امروزه بعضاً در خودروها دیده می‌شود، اشاره می‌کنیم:

- **سامانه ناویگری و مسیر یابی:** دانستن مکان دقیق و همچنین تعیین مسیر بهینه بین مبدأ و مقصد از جمله ویژگی‌هایی می‌باشد نه تنها در وسایل نقلیه خودران بلکه در رباتیک نیز مورد توجه می‌باشد. در وسایل نقلیه خودران به علت سرعت بالایی که می‌توانند داشته باشند دانستن موقعیت خودرو در GPS فریم جهانی با چالش‌هایی مواجه می‌باشد. اولین سنسوری که برای این منظور به ذهن می‌رسد می‌باشد ولی این سنسور به علت فرکانس نسبتاً کمی که در تعیین مکان خودرو دارد و همچنین محدودیت استفاده از آن در تونل‌ها به تنها‌ی پاسخگوی نیاز ما نمی‌باشد. لذا اطلاعات به دست آمده از آن با سنسورهای دیگر مانند IMU و Lidar ترکیب می‌شود تا مکان دقیق‌تری با نرخ به روز رسانی بالاتری را در اختیار ما قرار دهد.

- **تشخیص موائع و علائم راهنمایی:** امروزه به لطف الگوریتم‌های مختلف در هوش مصنوعی به راحتی می‌توان موائع را به صورت بلادرنگ تشخیص داد و حتی نوع آن‌ها را نیز با توجه مجموعه داده‌هایی که مدل با آن‌ها آموزش دیده است، مشخص کرد. از جمله الگوریتم‌های معروف در این حوزه الگوریتم YOLO می‌باشد که ورژن‌های متعددی دارد و نکته قابل توجه در آن، تشخیص همه‌ی اشیاء موجود در یک تصویر و همچنین مشخص کردن مرزی مستطیلی دور هریک از آن‌ها با سرعت بالا می‌باشد و استفاده از آن را برای کاربردهای بلادرنگ تسهیل می‌کند.

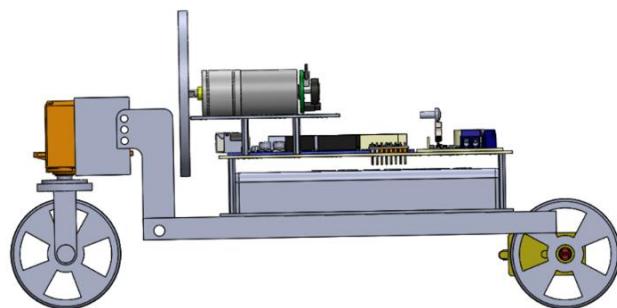
- **رانندگی بین خطوط به صورت اتوماتیک:** با توجه به مدل‌های آموزش دیده هوش مصنوعی و همچنین علم پردازش تصویر و تضاد رنگی که بین خطوط خیابان و آسفالت وجود دارد، می‌توان خطوط را تشخیص داد و بین آن‌ها حرکت کرد.
- **سیستم محركه خودرو:** با توجه به شرایط محیطی از قبیل لغزنده، بارانی بودن و ... و همچنین اینکه در سربالایی قرار داریم و ... نیروی محركه و در حقیقت گشتاور مناسبی با آن شرایط لازم است تا توسط موتور و گیربکس وسیله نقلیه اعمال شود. در وسایل نقلیه کاملاً خودران به واسطه‌ی سنسورهایشان و با توجه به درکی که از محیط پیدا می‌کنند باید خودشان حالت مناسب رانندگی را انتخاب کنند.

در موتورسیکلت خودران علاوه بر همه‌ی این موارد نیاز به یک سامانه‌ی حفظ تعادل نیاز داریم که با توجه به اهمیت آن، در ادامه جداگانه به آن خواهیم پرداخت.

۱-۴- سامانه‌ی حفظ تعادل

ویژگی‌های نام بردۀ برای خودران شدن یک وسیله نقلیه لازم است ولی در موتورسیکلت لازم است یک گام ابتدایی و مهم برداشته شود و آن هم حفظ تعادل موتورسیکلت می‌باشد. چنانچه در ابتدای امر به این مهم دست بیابیم، در ادامه می‌توان مانند یک اتومبیل خودران اجزا و تکنولوژی‌های مختلف را به آن اضافه نمود. در این پژوهش نیز ما بر روی حفظ تعادل موتورسیکلت متمرکز می‌شویم.

مقالات متعددی در زمینه حفظ تعادل وسایل نقلیه‌ی دو چرخ نگارش شده است. در بسیاری از روش‌ها از یک چرخ طیار و به کارگیری قوانین موجود در مسئله پاندول معکوس کمک گرفته شده است. به این صورت می‌باشد که بدون اثرات گشتاور خارجی بر روی یک جسم، تکانه زاویه‌ای کل آن جسم ثابت می‌ماند. زمانی که موتورسیکلت از حالت تعادل منحرف شود، گرانش موتورسیکلت باعث ایجاد گشتاوری می‌شود که ان را به سمت پایین می‌کشد. در این زمان چرخ طیار حول محور خود با شتاب زاویه‌ای α می‌چرخد و گشتاوری ایجاد می‌کند که با گشتاور ایجاد شده‌ی ناشی از گرانش برابر شود و سیستم تعادل شود. برای این منظور باید ولتاژ ورودی به موتور متصل به چرخ طیار را با روش‌های مختلف مثل کنترل مقاوم یا تطبیقی و ... کنترل کرد [۳، ۲]. یک مدل از حفظ تعادل به لیت روش در شکل ۱-۳ قابل مشاهده است.



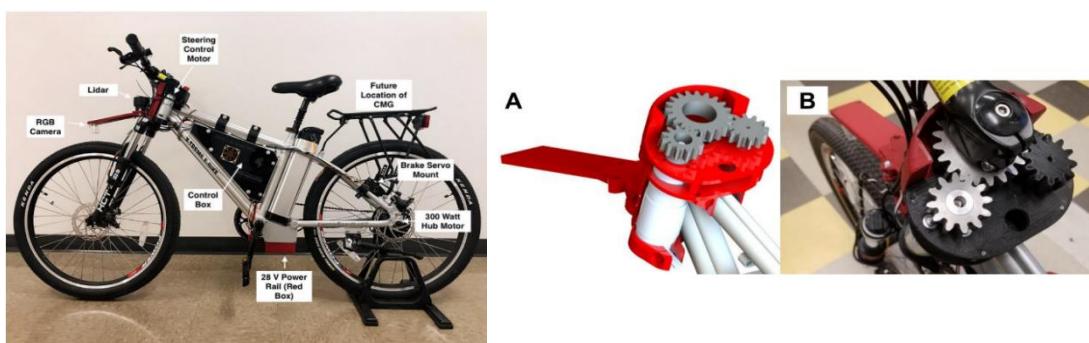
شکل ۱-۳ مدلسازی موتورسیکلت خودمتعادل شده به کمک چرخ طیار

همچنین در تعدادی از پژوهش‌ها به حفظ تعادل موتورسیکلت به کمک ژایروسکوپ پرداختند. در این حالت یک دیسک با سرعت بالایی می‌چرخد و با توجه به خاصیت ژایروسکوپیک چنانچه نیروی خارجی وارد شود، مجدداً موتور سیکلت را به موقعیت اولیه بر می‌گرداند [۴]. در شکل ۱-۴ نمونه‌ی آن آمده است.



شکل ۱-۴ مدلسازی موتورسیکلت خودمتعادل شده به کمک ژایروسکوپ

روش دیگری که در سرعت‌های پایین برای حفظ تعادل موتورسیکلت مورد مطالعه قرار گرفته است، حفظ تعادل به کمک فرمان دوچرخه می‌باشد. کنترل کننده‌های پایداری برای فرمان دادن به دوچرخه برای دستیابی به تعادل خود در سرعت‌های مختلف رو به جلو طراحی و اجرا شده‌اند.



شکل ۱-۵ مدلسازی موتورسیکلت خودمتعادل شده به کمک فرمان

در ادامه ابتدا به مدلسازی دینامیکی دوچرخه پرداخته و سپس با کمک موتور فیزیکی به مدلسازی موتور سیکلت و حفظ تعادل موتور سیکلت خود در این نرم افزار می‌پردازیم. درنهایت هم به بحث درباره نتایج به دست آمده خواهیم رسید.

فصل دوم

آشنایی با نرم افزار Vortex Studio

آشنایی با نرم افزار Vortex Studio

Vortex Studio یک نرم افزار شبیه سازی و توسعه‌ی سیستم‌های فیزیکی است و توسعه یافته در شرکت CM Labs می‌باشد، که به مهندسان اجازه می‌دهد تا مدل‌های دقیق واقعی‌گرایانه از اشیاء و پدیده‌های فیزیکی ایجاد کنند. این نرم افزار از موتور فیزیکی^۱ استفاده می‌کند تا تاثیرات و تعاملات مختلفی که در شبیه سازی‌ها نمایش داده می‌شوند، اعم از جاذبه، اصطکاک و ارتباطات جسمانی، را به خوبی شبیه سازی کند.

در این فصل ابتدا با مفهوم موتور فیزیکی آشنا می‌شویم. همچنین به صورت مختصر به نحوه کار کرد یک موتور فیزیکی پرداخته می‌شود و به کاربردها و محدودیت‌های آن پی می‌بریم. در نهایت به معرفی نرم افزار Vortex Studio به عنوان یک نمونه از موتورهای فیزیکی خواهیم پرداخت.

۱-۲ - موتور فیزیکی چیست؟

موتور فیزیکی نرم افزاری است که برای شبیه سازی رفتار فیزیکی اشیاء در محیط مجازی استفاده می‌شود. این ابزار به توسعه دهندگان این امکان را می‌دهد تا تعاملات فیزیکی واقع گرایانه‌ای را به اپلیکیشن‌های مختلفی نظریه بازی‌های ویدیویی، انیمیشن‌ها و شبیه سازی‌ها اضافه کنند. با استفاده از موتورهای فیزیکی، تعاملات فیزیکی پیچیده از جمله برخوردها، گرانش، اصطکاک و دینامیک سیالات به طور دقیق در دنیای دیجیتالی نمایش داده می‌شوند.

به صورت کلی دو دسته موتور فیزیکی وجود دارد:

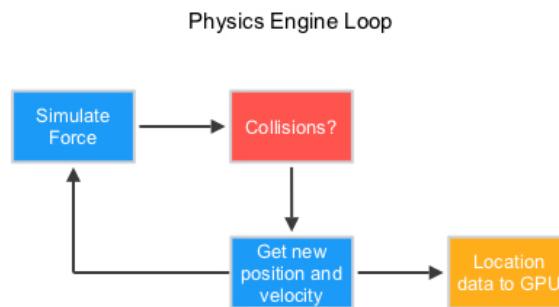
- **مotaورهای فیزیکی با دقت بالا:** این دسته از موتورهای فیزیکی به توان پردازشی بالایی برای محاسبه‌ی دقیق فیزیک یک مسئله، نیاز دارند و غالباً توسط دانشمندان و مهندسین در حوزه‌های مختلف و همچنین انیمیشن سازهای کامپیوتری استفاده می‌شوند.
- **مotaورهای فیزیکی بلاذرنگ:** این مورد در بازی‌های ویدئویی و یا کارهای دیگری که به صورت تعاملی با کاربر یا اجزای مختلف همان پلت فرم، استفاده می‌شود. در حقیقت در این حالت دقت کاهش

¹ Physics Engine

می‌یابد و از محاسبات ساده شده استفاده می‌شود تا بازی یا شبیه ساز در یک نرخ تعیین شده، پاسخگو باشد.

۲-۲ - نحوه کار کرد موتور فیزیکی

به صورت کلی یک موتور فیزیکی وظیفه‌ی حل معادلات حرکت و تشخیص برخوردها را بر عهده دارد. شما می‌توانید یک موتور فیزیکی را به عنوان یک حلقه‌ای که به طور مداوم در حال اجرا می‌باشد که با شبیه سازی نیروهای خارجی وارد بر اجسام مانند جاذبه، شروع می‌شود. در هر مرحله‌ی زمانی برخوردها را شناسایی می‌کند و سپس سرعت و موقعیت اجسام را محاسبه می‌کند و در آخر داده‌های به دست آمده را به واحد پردازش گرافیکی^۱ سیستم ارسال می‌کند. این مراحل کار در شکل ۲-۱ نمایش داده شده است.



شکل ۲-۱ حلقه‌ی اجرایی موتور فیزیکی

۲-۱ - حلقه‌ی موتور فیزیکی

همانطور که پیشتر اشاره شد، یک موتور فیزیکی با توجه به نیروها و گشتاورهای اعمالی به بدنه‌ی اجسام و اشیاء تعریف شده در آن، به محاسبه‌ی شتاب، سرعت و جایه‌جایی ناشی از آن‌ها می‌پردازد. براین اساس می‌توان گفت که پایه‌ای ترین و اساسی‌ترین معادلات در موتور فیزیکی، قانون دوم نیوتون و معادله اویلر برای دینامیک اجسام صلب می‌باشند که در معادلات (۲-۱) و (۲-۲) به آن‌ها اشاره شده است.

$$\vec{F} = m\vec{a} \quad (2-2)$$

^۱ GPU

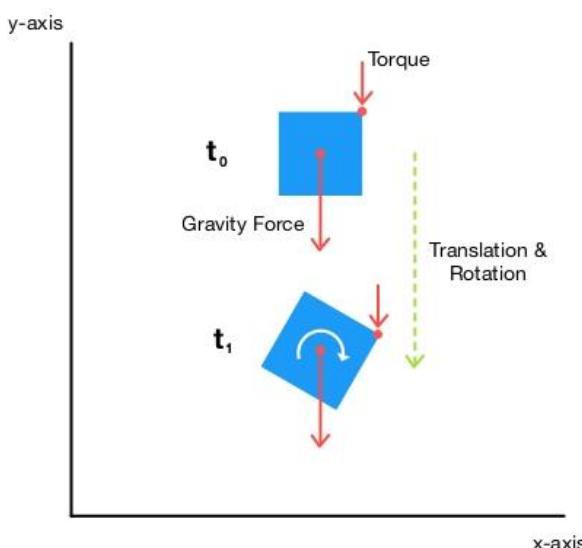
$$\overrightarrow{M} = I \vec{\alpha} \quad (2-2)$$

به این معادلات، معادلات حرکت گفته می‌شود.

می‌دانیم، چنانچه از شتاب نسبت به زمان انتگرال بگیریم، به سرعت و چنانچه از سرعت نسبت به زمان انتگرال بگیریم به جابه‌جایی می‌رسیم. موتور فیزیکی از معادلات حرکت انتگرال می‌گیرد تا به سرعت و جابه‌جایی اجسام برسد و این کار را در یک حلقه به صورت پیوسته انجام می‌دهد و شامل مراحل زیر می‌باشد:

- ۱ تمامی نیروها و گشتاورهای وارد بر جسم را شناسایی می‌کند.
- ۲ بردار برآیند نیروها و گشتاورها را محاسبه می‌نماید.
- ۳ معادلات حرکت را برای محاسبه شتاب خطی و زاویه‌ای حل می‌کند.
- ۴ از شتاب برحسب زمان برای یافتن سرعت خطی و زاویه‌ای انتگرال می‌گیرد.
- ۵ از سرعت برحسب زمان انتگرال می‌گیرد تا جابه‌جایی خطی و زاویه‌ای را بیابد.

اگر نیروی جاذبه و ممانی به یک جسم اعمال شود، آنگاه حلقه‌ی موتور فیزیکی تصویر افتادن و چرخیدن این جسم را به عنوان نتیجه، ایجاد می‌کند و نتیجه‌ای مانند شکل ۲-۲ را خواهیم دید.



شکل ۲-۲ اعمال نیرو و گشتاور به جسم و خروجی حاصل از آن در موتور فیزیکی

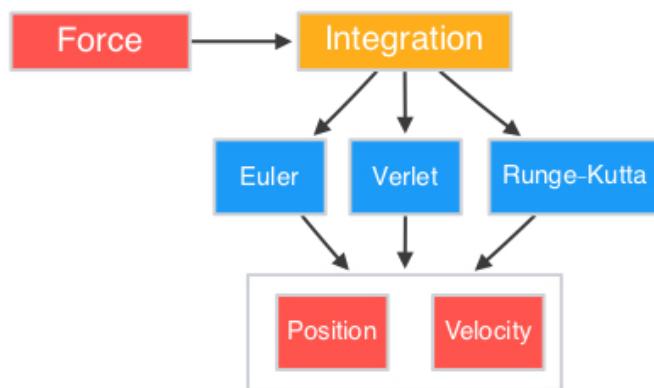
۲-۲-۲- انتگرال‌گیری عددی

در رایانه‌ها برای محاسبه‌ی انتگرال از روش‌های عددی یا به اصطلاح انتگرال‌گیری عددی استفاده می‌شود. انتگرال‌گیری عموماً در یک بازه‌ی پیوسته انجام می‌شود ولی می‌دانیم در کامپیوتر کوچک‌ترین واحد شمارش بیت‌ها هستند و بنابراین رایانه‌ها از تکنیک‌های حل عددی برای تخمین مقدار انتگرال استفاده می‌کنند.

روش‌های انتگرال‌گیری عددی مختلفی برای حل معادلات حرکت استفاده می‌شود که معروف‌ترین آن‌ها عبارتند از:

- روش اویلر
- روش ورلت
- روش رانگ-کوتا

دربین روش‌های نام برد، روش اویلر آسان‌ترین پیاده‌سازی و در عین حال کم دقیق‌ترین نتیجه را دارد. در مقابل، روش رانگ-کوتا پیاده‌سازی پیچیده‌تری دارد اما نتیجه‌ی آن از دقت بیشتری برخوردار می‌باشد [۵]. دسته‌بندی روش‌ها در شکل ۳-۲ قابل مشاهده است.



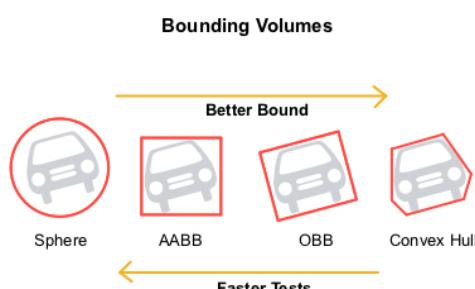
شکل ۳-۲ روش‌های انتگرال‌گیری عددی در موتور فیزیکی

۳-۲-۳- تشخیص برخوردها

تشخیص برخورد به طور کلی از وظایف موتور فیزیکی به شمار می‌آید ولی در واقعیت مسئولیت آن را یک زیرسامانه به نام «سامانه تشخیص برخورد» بر عهده دارد و این دو سیستم به کمک یک دیگر و با ارتباط هماهنگ با یک دیگر به تشخیص برخوردها می‌پردازند. برای سادگی، در اینجا بین این دو سامانه تفاوتی قائل نمی‌شویم.

وظیفه‌ی سامانه‌ی تشخیص برخورد، این است که چنانچه دو شیء با یک دیگر برخورد کردند، آن را اعلام کند و دراقع یک جواب ساده‌ی بله یا خیر را در هر لحظه گزارش می‌دهد، هرچند که این پردازش بسیار زمان بر می‌باشد. موتور فیزیکی برای آنکه به فرآیند تشخیص برخوردها سرعت بخشد، این فرآیند را به دو فاز با نام‌های فاز گسترده^۱ و فاز باریک^۲ تقسیم کرده است.

فاز گسترده: اشیاء و اجسامی که موتور فیزیکی به زیرسامانه‌ی تشخیص برخورد می‌دهد، در حجم‌های مشخصی محصور شده‌اند؛ مانند بسته بندی یک شیء و به این طریق حدود این اجسام مشخص می‌شود. قدیمی‌ترین و مرسوم‌ترین هندسه‌ها برای ایجاد این مرزبندی‌ها عبارتند از: مرزبندی کروی^۳، مرزبندی با محورهای تراز شده (AABB)^۴، مرزبندی با محورهای جهت‌دار (OBB)^۵ و مرزبندی با پوسته محدب^۶.



شکل ۴-۲ روش‌های سنتی مرزبندی اجسام در فرآیند تشخیص برخورد

¹ Broad-Phase

² Narrow-Phase

³ Sphere Bounding Box

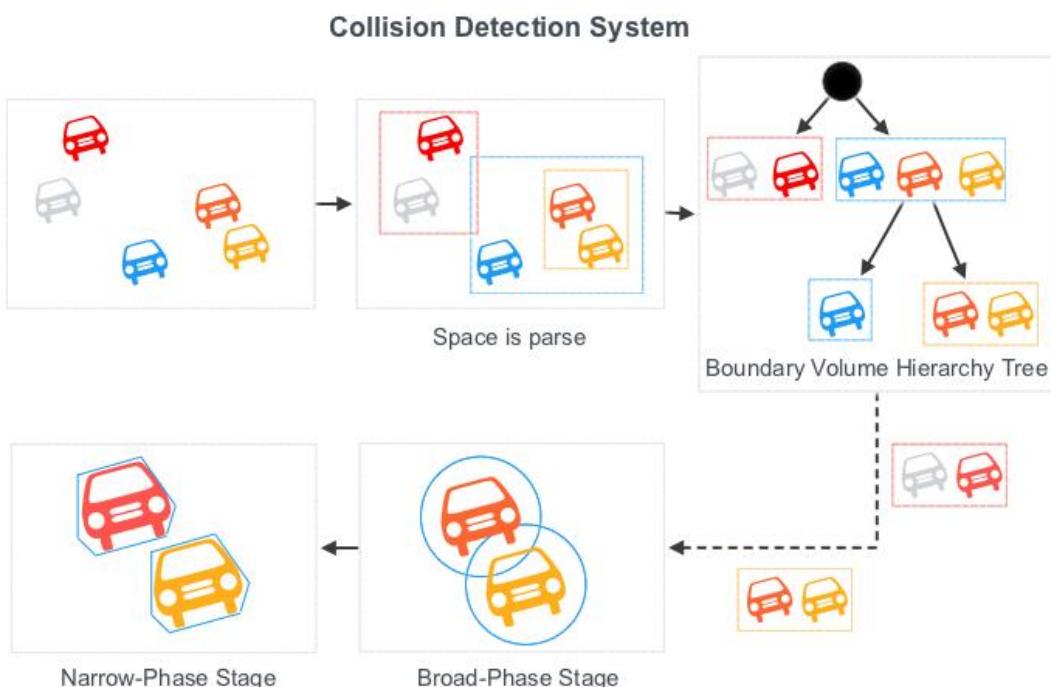
⁴ Axis-Aligned Bounding Box

⁵ Oriented Bounding Box

⁶ Convex Hull Bounding Box

در پایان فرآیند تشخیص برخورد در فاز گستردگی، هر شیء با یک مرزبندی کروی محصور و بسته بندی می‌شود. این فرآیند به این صورت انجام می‌گیرد که موتور فیزیکی فضای اطراف اشیاء را تجربیه می‌کند تا به یک مرزبندی سلسله مراتبی (BVH)^۱ برای اشیاء برسد، به عبارتی در نهایت BVH به ما یک ساختار درختی می‌دهد که هر نواد آن اشیائی را در بردارد، که احتمال بالایی برای برخورد با یک دیگر دارند و درواقع به هم نزدیک‌تر می‌باشند. نمونه‌ای از آن را در شکل ۲-۵ می‌توانید مشاهده نمایید.

موتور فیزیکی در حقیقت هر نواد را آزمایش می‌کند و لیستی از جفت برخوردها ایجاد می‌کند. فاز گستردگی سریع است و ممکن است به اشتباه تشخیص دهد که برخودهایی رخ داده است. این امر به دلیل آن است که در این فاز از مرزبندی ساده‌ی کروی استفاده شده است که توان پردازشی زیادی برای محاسبات لازم ندارد و از طرفی موجب تولید مثبت‌های کاذب می‌شود. نکته‌ی مثبت این فاز آن است که اشیائی را که سامانه تشخیص می‌دهد با یک دیگر برخورد ندارند را با اطمینان از لیست بررسی حذف کرد.



شکل ۲-۵ فرآیند تشکیل BVH و تشخیص برخورد در موتور فیزیکی

¹ Boundary Volume Hierarchy

فاز باریک: لیست جفت-برخوردهای تولید شده در مرحله‌ی قبل به فاز باریک سامانه‌ی تشخیص برخورد منتقل می‌شود. در این مرحله هریک از آن اشیاء، به کمک پوسته‌ی محدب مرزبندی می‌شود تا شکل تقریبی آن تخمین زده شود. سپس سامانه یک آزمایش تشخیص برخورد بین پوسته‌های محدب حاصل شده انجام می‌دهد. لازم به ذکر است این آزمایش از الگوریتم GJK^۱ استفاده می‌کند که الگوریتمی دقیق است ولی از نظر زمانی هزینه‌بر می‌باشد.

۴-۲-۲- سامانه پاسخ برخورد

بخشی که مسئولیت پاسخ برخورد را بر عهده دارد، در حقیقت پس از تشخیص برخورد اشیاء، تغییراتی که در سرعت‌های خطی و زاویه‌ای آن‌ها ایجاد می‌شود را محاسبه می‌کند. این محاسبات به اطلاعاتی از قبیل جرم، ضریب بازگشت، نقاط برخورد و بردارهای نرمال برخورد، نیاز دارد.

در لحظه‌ی برخورد، بیشترین نیرویی که روی اشیاء اثر می‌کند، نیروی ضربه می‌باشد و بنابراین تمامی نیروهای دیگر در آن لحظه‌ی برخورد درنظر گرفته نمی‌شوند. همانطور که می‌دانیم اندازه‌ی نیروی ضربه بسیار بزرگ است ولی در زمان کوتاهی اعمال می‌شود.

بعد از رخدادن برخورد، اثر نیروی ضربه از بین می‌رود ولی سایر نیروهای خارجی مجدداً روی جسم اعمال می‌شوند. معادله‌ی حرکت حل می‌شود و مقادیر جدیدی برای مکان و سرعت در اختیار ما قرار می‌دهد. موتور فیزیکی پیوسته این فرآیند را تکرار می‌کند و این خروجی و تصویر را برای ما ایجاد می‌کند که برای مثال جسمی در اثر جاذبه سقوط می‌کند.

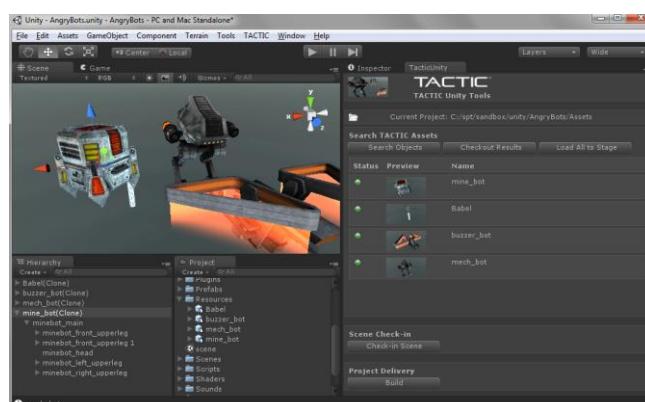
۳-۲- کاربردهای موتور فیزیکی

باتوجه به توضیحاتی که در قسمت قبل پیرامون نحوه عملکرد موتور فیزیکی داده شد، متوجه می‌شویم که هرچه توان پردازشی سخت‌افزارهای ما بالاتر باشد، باعث می‌شود که بتوانیم مرزبندی‌های مشخص کننده‌ی شکل کلی اجسام را، دقیق‌تر انجام دهیم و درنهایت نتیجه‌های نزدیکتر به واقعیت را بدون هیچ‌گونه تاخیری در خروجی مشاهده کنیم. امروزه با پیشرفت‌هایی که در زمینه‌ی ساخت‌افزارها

^۱ Gilbert-Johnson-Keerthi

داشته‌ایم، عرصه جهت پیشرفت و دقیق‌تر شدن محاسبات موتورهای فیزیکی نیز گسترش یافته است و این امر باعث شده است تا در بسیاری از کاربردها و صنایع ابتدا آزمایش‌ها را به کمک این موتورهای فیزیکی شبیه سازی کنند تا تخمین خوبی از خروجی آن آزمایش در واقعیت نیز به دست آورند و از صرف هزینه‌های بسیار جهت مشاهدات تجربی، پرهیز کنند. همچنین از جمله کاربردهای محبوب موتورهای فیزیکی در زمینه‌های بازی‌سازی و انیمیشن سازی می‌باشد که هرچه بتوانند نتایج با کیفیت‌تر و واقعی‌تری را به کابر ارائه دهند، موفقیت و فروش بیشتری را کسب خواهند کرد. در ادامه نمونه‌هایی از کاربرد موتورهای فیزیکی در صنایع و زمینه‌های مختلف آمده است:

- **صنعت بازی سازی:** در بسیاری از بازی‌های کامپیوتری، سرعت شبیه سازی نسبت به دقت شبیه سازی‌های فیزیکی، از اهمیت بیشتری برخوردار است. این موضوع باعث شد که موتورهایی طراحی بشوند که شبیه سازی‌ها را به صورت بلادرنگ انجام دهند و تنها شبیه سازی فیزیکی برخی از موارد خاص و ساده همانند دنیای واقعی انجام شود. در اصطلاح به این موتورها، موتورهای بازی می‌گویند. در بازی‌های کامپیوتری از تکنیک‌های مختلفی برای شبیه سازی قوانین فیزیکی حاکم بر دنیای واقعی استفاده می‌شود ولی به صورت کلی تمام شبیه سازی‌های فیزیکی مورد استفاده، جزء دو دسته‌ی اصلی شبیه سازی‌های جسم صلب^۱ و جسم انعطاف‌پذیر^۲ است. از معروف‌ترین موتورهای بازی می‌توان به شبیه سازی‌های Godot، GameMaker، Unity، Unreal Engine و ... اشاره کرد. در شکل ۶-۲ نمایی از محیط Unity نمایش داده شده است.



شکل ۶-۲ تصویری از ساخت بازی در موتور فیزیکی Unity

¹ Rigid Body

² Soft Body

- صنعت انیمیشن سازی: همانطور که اشاره کردیم، ابتدا موتورهای فیزیکی برای ساخت بازی‌های ویدیوئی توسعه یافته‌ند و می‌توانستیم به کمک آن‌ها تعاملات شخصیت‌ها و حرکات آن‌ها را توسعه دهیم. به کمک موتورهای فیزیکی می‌توان حتی جنس و بفت لباس‌ها را باتوجه به ماده‌ای که برای آن‌ها تعریف می‌شود به صورت طبیعی جلوه داد. همچنین به لطف هوش مصنوعی به کاررفته داخل موتورهای فیزیکی می‌توان فقط ۴ یا ۵ فریم را به جای ۵۰ فریم برای یک حرکت خاص یک کاراکتر طراحی کرد؛ چراکه هوش مصنوعی بقیه‌ی حرکت را به صورت کاملاً روان و واقعی ایجاد می‌کند. این سبب می‌شود که زمان موردنیاز برای طراحی به شدت کمتر شود. همچنین می‌توان طراحی لایه‌بندی شده برای عمق دادن به تصاویر داشت و لایه‌ها را پشت یک دیگر اضافه کرد. برای مثال Spine یکی از موتورهای فیزیکی برای انیمیشن‌های دو بعدی می‌باشد. نحوه‌ی طراحی شخصیت کارتونی در شکل ۷-۲ قابل مشاهده است.



شکل ۷-۲ اسکلت‌بندی و تنظیم حرکات یک شخصیت انیمیشنی

- شبیه‌سازی‌های علمی و فیزیکی: موتورهای فیزیکی در شبیه‌سازی‌های علمی مورد استفاده قرار می‌گیرند تا واقع‌گرایی بالا در مدل‌سازی پدیده‌های فیزیکی داشته باشیم. این موتورها در مطالعه و تحلیل اثرات مختلف متغیرهای فیزیکی مثل جاذبه، اصطکاک و برخوردها، در پژوهش‌های علمی و تحقیقاتی مورد استفاده قرار می‌گیرند. در این نوع شبیه‌سازی‌ها، موتورهای فیزیکی برای مدل‌سازی و پیش‌بینی رفتارهای مختلف در شرایط مشخص به کار می‌روند. از جمله مثال‌ها به مدل‌سازی الگوهای آب و هوا، شبیه‌سازی تونل باد برای تست وسایل نقلیه پرنده یا آیروдинامیک وسایل نقلیه زمینی و ... اشاره کرد. نرم افزار Vortex Studio از جمله نمونه‌هایی در این زمینه می‌باشد.

- **خودروهای خودران و هوش مصنوعی:** در زمینه تعلیم خودروهای خودران و مدل‌های هوش مصنوعی، موتورهای فیزیکی برای ترکیب بازی‌ها و تست‌های تعاملی مورد استفاده قرار می‌گیرند. این ابزارها می‌توانند به ماشین‌ها یاد دهند که چگونه در محیط‌های مختلف با اشیاء و عوامل فیزیکی تعامل داشته باشند. برای مثال شبیه ساز CARLA نمونه‌ای است که برای توسعه‌ی خودروهای خودران پدید آمده است. به جای آنکه آزمایش را بر روی یک خودرو با تعداد زیادی سنسور به صورت واقعی در سطح شهر انجام دهند، در شبیه‌ساز مجهز به موتور فیزیکی این کار را انجام می‌دهند تا هزینه‌ی کمتری نیز به همراه داشته باشد.
- **طراحی سیستم‌های صنعتی و ماشینی:** در طراحی و توسعه سیستم‌های مختلف، از جمله ربات‌های صنعتی، موتورهای فیزیکی به منظور مدل‌سازی و پیش‌بینی عملکرد ماشین‌ها و سیستم‌ها با در نظر گرفتن تاثیرات فیزیکی استفاده می‌شوند [۶].

۴-۴- محدودیت‌های موتور فیزیکی

با توجه به محدودیت‌های سخت‌افزارهای اجراکننده‌ی موتورهای فیزیکی همچنان نقاط ضعفی در موتورهای فیزیکی وجود دارد. این محدودیت ناشی از وجود تقریب در مش بندی اشیاء و پایین بودن سرعت همگرایی الگوریتم‌های تشخیص دهنده‌ی برخورد، می‌باشد و در نتیجه بسیاری از نتایج به صورت تقریبی می‌باشند.

در حقیقت چنانچه الگوریتم تشخیص برخورد، به آرامی همگرا شود، سبب می‌شود که اشیاء کمی از یک دیگر عبور کنند، و یک نیروی دافعه‌ی غیرعادی و بزرگی در اثر این برخورد به دست آید که در نتیجه‌ی آن اجزای سامانه از هم گسیخته می‌شوند. همچنین به علت همگرایی کند حل‌کننده‌های گاووس-سایدل، نیروهای عکس‌العمل تخمینی محاسبه شده معمولاً منجر به جهش‌های غیرطبیعی در سامانه می‌شوند. این قبیل محدودیت‌ها به صورت عمدی در سامانه‌های دینامیکی چندجسمی، سامانه‌هایی که در آن اعضایی مانند طناب و زنجیر تحت کشش زیاد قرار می‌گیرند و یا سامانه‌هایی که در آن چند جسم متصل به یک دیگر روی یک یا چند چرخ در حال حرکت می‌باشند نمود بیشتری پیدا می‌کند.

همانطور که در ابتدای این بخش ذکر شد، بخش زیادی از این خطاهای به کمک سیستم‌هایی با توان پردازشی بالا قابل جبران است که این نیز مستلزم صرف هزینه‌ی زیادی برای تهیه‌ی این سخت‌افزارها و پردازنده‌ها می‌باشد.

۴-۵-۱- نرم افزار Vortex Studio

نرم افزار Vortex Studio یک پلتفرم برای ایجاد شبیه‌سازی‌های مکانیکی به صورت بلادرنگ و تعاملی می‌باشد که در شرکت CM Labs توسعه داده شده است. این نرم افزار برای شبیه‌سازی‌های خود از موتور فیزیکی، که پیش‌تر به صورت کامل به آن پرداختیم، استفاده می‌کند.

این نرم افزار، یک بستر مناسب با صحت بالا برای نمونه‌سازی مکانیکی سامانه‌های سریع و کاربرمحور، طراحی مفهومی و ساده محصولات ایجاد می‌کند و امکان تجربه مجازی برای آزمایش‌های انسان در حلقه، آموزش اپراتورهای ماشین آلات صنعتی و ... را فراهم می‌آورد.

باتوجه به آنکه سامانه‌های مکانیکی و ربات‌های متحرک در حال آمیخته شدن با سامانه‌های هوشمند مانند حسگرهای با وضوح بالا و الگوریتم‌های یادگیری ماشین می‌باشند، ساخت نمونه‌های آزمایشی و آزمایش آن‌ها به طور فرایندهای پیچیده‌تر، گرانتر و دارای چالش‌های بسیار بیشتری می‌شود. ازین رو می‌توان با استفاده از این نرم‌افزار، ضمن طراحی مفهومی محصولات به صورت مجازی و طراحی فرایند عملکرد و برنامه‌ریزی عملیات برای دستگاه‌ها، بستری جهت آموزش اپراتورهای دستگاه و آزمایش مجازی عملکرد دستگاه‌ها و سامانه‌های مکانیکی و مکاترونیکی استفاده نمود [۷].

۴-۵-۲- نحوه کارکرد نرم افزار

از آنجایی که این نرم افزار نمونه‌ای از موتورهای فیزیکی می‌باشد، روند کلی شبیه سازی در آن مانند سایر موتورهای فیزیکی می‌باشد که به صورت کامل در بخش قبلی به آن پرداختیم.

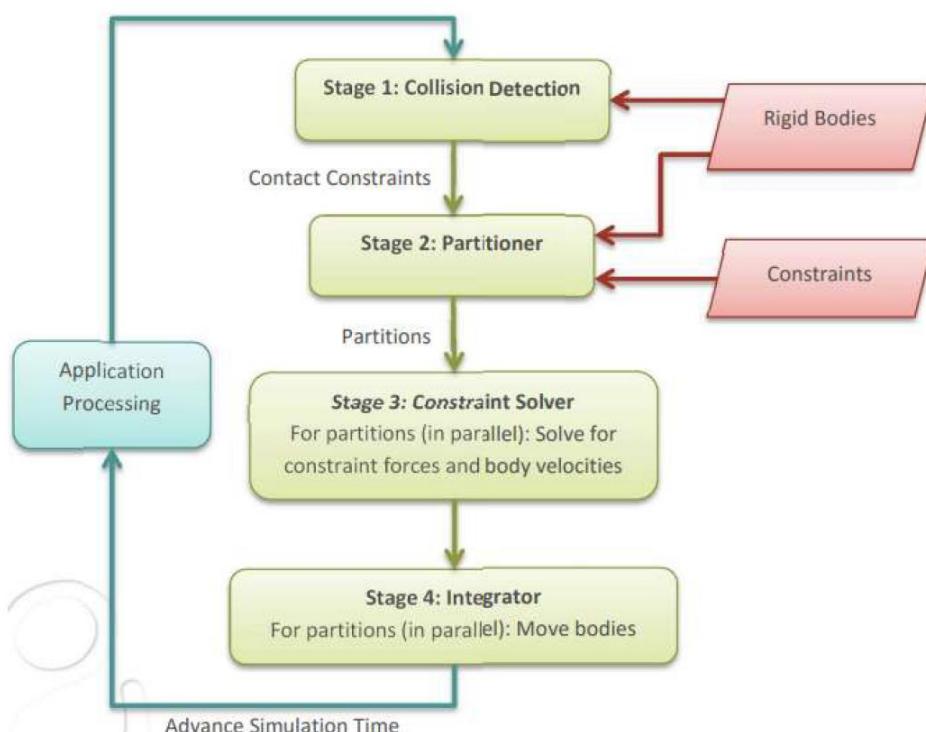
در نرم افزار Vortex Studio شبیه سازی یک سامانه مکانیکی در ۴ مرحله انجام می‌گیرد:
مرحله اول: نرم افزار در این گام به تشخیص برخوردها می‌پردازد. به کمک هندسه‌ی برخورد تعریف شده برای اجسام و روابط ریاضی مربوط به آن‌ها فرآیند تشخیص را انجام می‌دهد.

مرحله دوم: در مرحله بعدی، نرم افزار با دریافت قیود محاسبه شده در مرحله قبل و همچنین جسم صلب تعریف شده، سامانه را به بخش‌های کوچک‌تری تقسیم می‌کند که هریک از این بخش‌ها تعدادی از اجزای سامانه‌ی اصلی را شامل می‌شود. در حقیقت مانند ساختار درختی BVH که پیش‌تر بحث شد، این امر برای آن است که فقط اجسامی که به یک دیگر بسیار نزدیک هستند و برخورد دارند، در مراحل بعدی بررسی شوند و حجم محاسبات کمتر شود.

مرحله سوم: در این مرحله، نرم افزار قیود فیزیکی مانند لغزش، عدم گسیخته شدن یک جسم و همچنین قیود تعریف شده توسط کاربر را اعمال نموده و نیوهای داخلی و خارجی وارد بر هر جسم را محاسبه می‌نماید.

مرحله چهارم: در این گام، بخش انتگرال‌گیر از شتاب به دست آمده برای اجسام، انتگرال گیری می‌کند و سرعت و مکان آن را محاسبه می‌کند و با اعمال شرایط جدید بر روی اجسام در شبیه سازی، آن را به کاربر نمایش می‌دهد.^[۸]

در شکل ۸-۲ مراحل ذکر شده، نمایش داده شده است.



شکل ۸-۲ مراحل انجام شده برای شبیه سازی در نرم افزار Vortex Studio

فصل سوم

مدل‌سازی موتورسیکلت

مدل‌سازی موتورسیکلت

در بسیاری از پروژه‌ها و مسائل، گام اول و بسیار مهمی که وجود دارد مدل‌سازی می‌باشد. چنانچه مرحله‌ی مدل‌سازی به خوبی انجام گیرد، فهم مسئله در ادامه آسان‌تر خواهد بود و موجب می‌شود نتایجی که می‌گیریم بیشتر به واقعیت نزدیک باشد.

در این پروژه ما با یک مسئله‌ی کنترلی روبرو هستیم و میخواهیم تعادل موتور سیکلت را برقرار کنیم. در مسائل کنترلی نیز یک گام اولیه و مهم مدل‌سازی سیستم مورد نظر می‌باشد. با توجه به اینکه در اینجا با یک سامانه‌ی دینامیکی مواجه هستیم، بنابراین باید به مدل‌سازی دینامیکی آن بپردازیم. البته در این فصل به دنبال مدل‌سازی ریاضی نیستیم و به سعی برآن است تا براساس پژوهش‌های مشابه انجام شده، مدلی ساده‌تر از موتورسیکلت که دارای جزئیات زیادی هست، به دست آوریم به نحوی که در نتایج ما نیز تاثیر نامطلوبی نداشته باشد.

پس از آنکه در تئوری و با توجه روابط به مدلی ساده دست یافتیم، سعی می‌کنیم تا همان را در نرم افزار مدل‌سازی کنیم و پردازنده را در گیر محاسبات اضافی نکنیم. در این فصل نیز پس از بررسی مدل‌سازی دینامیکی به مدل‌سازی در نرم افزار می‌پردازیم و موتورسیکلت را آماده‌ی پیاده سازی کنترل بر روی آن می‌کنیم.

۱-۳ - مدل‌سازی دینامیکی

روش‌های بسیاری برای مدل‌سازی یک سامانه‌ی دینامیکی وجود دارد. از جمله روش‌های معروف برای مدل‌سازی ریاضی این سامانه‌ها عبارتند از:

- روش نیوتن-اویلر
- روش لاگرانژ

اگر بخواهیم روش لاگرانژ را به صورت مختصر توضیح دهیم، می‌توان گفت که در این روش ابتدا باید روابط مربوط به انرژی جنبشی و پتانسیل کل سیستم را بر حسب مختصات تعمیم یافته و مشتقه‌ات آن به دست آورید. چنانچه در گام بعدی اختلاف انرژی جنبشی از انرژی پتانسیل را بیابیم، لاگرانژین به دست

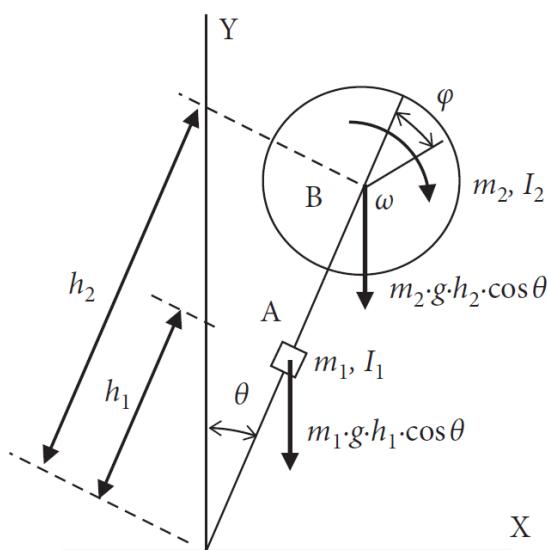
می‌آید که با L نمایش می‌دهند. سپس L را در معادله لاغرانژ قرار می‌دهیم که در زیر آمده است. لازم به ذکر است که مختصات تعمیم یافته را با q_k و سرعت‌های تعمیم یافته را با \dot{q}_k نمایش می‌دهیم.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_k}\right) - \frac{\partial L}{\partial q_k} = \tau_k \xrightarrow{L=T-V} \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_k}\right) - \frac{\partial T}{\partial q_k} + \frac{\partial V}{\partial q_k} = \tau_k \quad (1-3)$$

همچنین τ_k معرف نیروهای ناپایستار مانند اصطکاک و نیروهای خارجی می‌باشد که چنانچه، نیرویی از این جنس وجود نداشته باشد، طرف راست معادله برابر با صفر قرار داده می‌شود.

۱-۱-۳- مدل‌سازی ریاضی

در این پژوهه، قصد داریم تا به کمک یک چرخ طیار تعادل موتور سیکلت را حفظ نماییم. با توجه به این موضوع، لازم است تا چرخ طیار را نیز در مدل‌سازی خود لحاظ نماییم. مطابق توضیحاتی که در رابطه با معادله لاغرانژ داده شد، ابتدا از نیروهای اصطکاک و نیروهای خارجی و به صورت کلی نیروهای ناپایستار صرف نظر می‌نماییم و سعی می‌کنیم تا معادلات را به دست آوریم. برای این منظور ابتدا لازم است تا دیاگرام آزاد جسم را با توجه به مفروضات رسم نماییم.



شکل ۱-۳ دیاگرام آزاد موتورسیکلت خودمتعادل شده

با توجه به شکل ۱-۳ می‌توان گفت که انرژی جنبشی کلی جسم از دو قسمت تشکیل شده است، انرژی جنبشی موتورسیکلت و انرژی جنبشی چرخ طیار. در حقیقت موتور سیکلت به عنوان یک جسم صلب در نظر گرفته شده است که فرض ساده کننده‌ای می‌باشد [۲].

$$T_1 = \frac{1}{2} m_1 |v_A|^2 + \frac{1}{2} I_1 \dot{\theta}^2 \quad (2-3)$$

$$T_2 = \frac{1}{2} m_2 |v_B|^2 + \frac{1}{2} I_2 (\dot{\theta} + \dot{\phi})^2 \quad (3-3)$$

$$T = T_1 + T_2 \rightarrow T = \frac{1}{2} (m_1 h_1^2 + m_2 h_2^2 + I_1 + I_2) \dot{\theta}^2 + \frac{1}{2} I_2 \dot{\phi}^2 + I_2 \dot{\theta} \dot{\phi} \quad (4-3)$$

$$V = g \cdot \cos \theta \cdot (m_1 h_1 + m_2 h_2) \quad (5-3)$$

در نهایت با توجه به معادله لاغرانژ و جاگذاری در آن خواهیم داشت:

$$\begin{cases} q_i = \theta \rightarrow (m_1 h_1^2 + m_2 h_2^2 + I_1 + I_2) \ddot{\theta} + I_2 \dot{\phi} - g \cdot \sin \theta \cdot (m_1 h_1 + m_2 h_2) = 0 \\ q_i = \phi \rightarrow I_2 \ddot{\phi} + I_2 \dot{\theta} = \tau_m \end{cases} \quad (6-3)$$

در رابطه بالا، τ_m گشتاور اعمالی موتور به چرخ طیار می‌باشد. اگر با ترکیب دو معادله فوق، سعی کنیم تا گشتاور را براساس زاویه‌ی انحراف موتورسیکلت از تعادل و مشتقات آن به دست آوریم، آنگاه می‌توان به راحتی با ورودی گرفتن این پارامترها، خروجی مورد نظر را به دست آورد؛ با این حال، این رابطه نتیجه‌ی فرض‌های ساده کننده‌ی بسیاری است. البته دیدگاه خوبی درباره‌ی مقدار حدودی نیروی لازم برای ایجاد گشتاور در چرخ طیار به ما می‌دهد. همچنین به ما نشان می‌دهد چه پارامترهایی از موتور سیکلت برای حفظ تعادل حائز اهمیت می‌باشند.

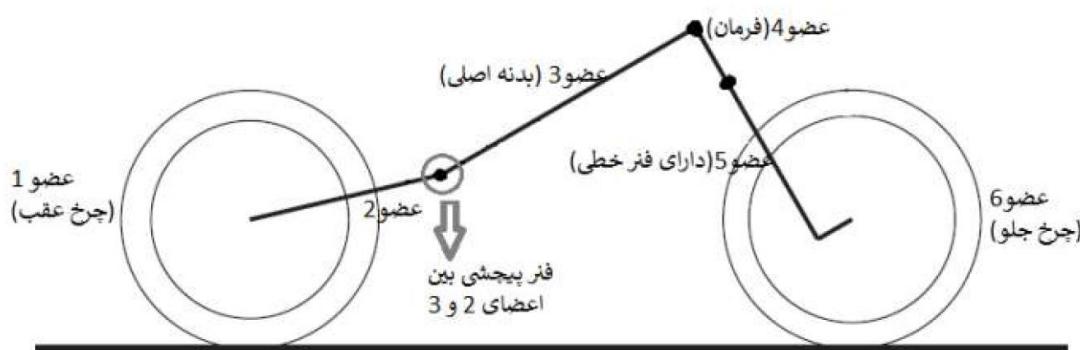
از آنجایی که ما در این شبیه‌ساز می‌توانیم جنس قسمت‌های مختلف را تعیین نماییم و برای مثال به چرخ‌ها و بدنه جنس‌هایی متناسب با واقعیت دهیم و مقادیر سختی هریک را تعیین کنیم و همچنین در بعضی موارد لازم است تا چرخش فرمان را هم داشته باشیم، پس به کمک این روابط ساده شده نمی‌توانیم موتورسیکلت را کنترل کنیم.

در این پژوهش ما به دنبال حفظ تعادل موتور سیکلت با روش‌های هوشمند هستیم و درواقع موتورسیکلت را به عنوان یک بلوك با تعدادی ورودی و خروجی در نظر می‌گیریم و به مدل‌سازی ریاضی موتور سیکلت نمی‌پردازیم چرا که اگر بخواهیم تمام جزئیاتی که در تعادل یک موتور سیکلت در دنیای واقعی موثر است را در نظر بگیریم معادلات پیچیده‌ای به دست می‌آید که ساده‌سازی و حل آن‌ها بسیار دشوار خواهد بود. در ادامه به صورت مختصر به مدل‌سازی دو بعدی و سه بعدی موتورسیکلت می‌پردازیم

و با توجه به آن‌ها مدلی از موتورسیکلت که صرفا شامل اجزای تاثیرگذار در تعادل آن هستند را برای ادامه‌ی کار، ارائه می‌دهیم.

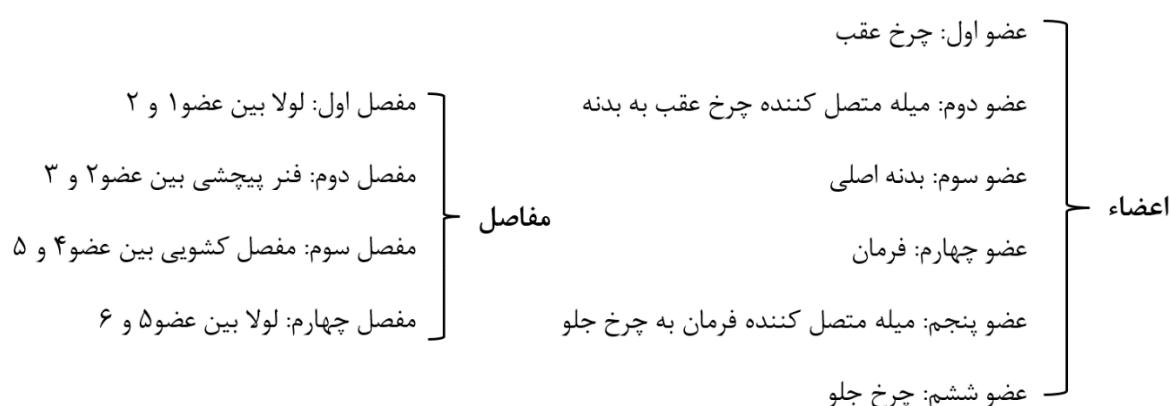
۲-۱-۳- مدل‌سازی دوبعدی موتورسیکلت

در مدل‌سازی دوبعدی، فرض می‌کنیم که موتورسیکلت در یک صفحه‌ی دوبعدی در حال حرکت می‌باشد و چرخش فرمان را نخواهیم داشت. در حقیقت این مدل در شکل ۲-۳ قابل مشاهده است.



شکل ۲-۳ اعضاء و مفاصل در مدل‌سازی دوبعدی موتورسیکلت

در این حالت سامانه از ۶ عضو و ۴ مفصل تشکیل می‌شود که هریک از این اعضاء و مفاصل به صورت دقیق در شکل ۳-۳ مشخص شده‌اند.



شکل ۳-۳ تعریف اعضاء و مفاصل در مدل دوبعدی

در حقیقت در این حالت اتصال فرمان به بدنه یک اتصال صلب می‌باشد و فرض شده موتور سیکلت صرفاً روی یک مسیر مستقیم حرکت می‌کند. همچنین به کمک فنر پیچشی و مفصل کشویی، جابه‌جایی در راستای عمودی امکان پذیر است. با توجه به توضیحات، می‌توان سامانه را با ۶ مولفه تعمیم یافته مشخص نمود و قید عدم حرکت نسبی چرخ‌ها در محور عمودی، سامانه را به ۵ درجه آزادی می‌رساند^[۹].

۳-۱-۳- مدل‌سازی سه بعدی موتورسیکلت

در مدل‌سازی سه بعدی، فرض می‌شود که فرمان نیز قابلیت چرخش دارد و همچنین در راستای عمود بر صفحه نیز به بعد اجسام اضافه می‌شود. شاید در نگاه اول، به نظر برسد که بعد سومی که در تمامی لینک‌ها اضافه می‌شود تفاوتی را در مدل‌سازی ایجاد نمی‌کند ولی در ادامه به تفاوت ایجاد شده بیشتر می‌پردازیم. از آنجایی که ما نیز به مدل‌سازی سه بعدی موتور سیکلت نیازمندیم، در این بخش توجه ویژه‌ای به نکات مطرح شده خواهیم کرد تا از آن‌ها برای مدل‌سازی نهایی خود استفاده کنیم.

در مدل سه بعدی، تعریف اعضاء و مفاصل مانند حالت دو بعدی می‌باشد ولی در درجات آزادی فضای کاری یا به اصطلاح مختصات‌های تعمیم‌یافته‌ی موتورسیکلت عبارتند از:

- مختصات مرکز جرم : شامل سه مختصه‌ی X و Y و Z

- زاویه انحراف موتور سیکلت نسبت به راستای عمودی خود (رول^۱)

- زاویه انحراف موتورسیکلت نسبت به محور X در صفحه افقی (یاو^۲)

- زاویه بین عضو شماره ۳ با افق (پیچ^۳)

- زاویه‌ی چرخش فرمان

- تغییر طول فنر به کار رفته بین فرمان و چرخ جلو

- میزان پیچش فنر پیچشی بین عضو ۲ و ۳

- میزان چرخش و موقعیت دورانی چرخ‌های جلو عقب

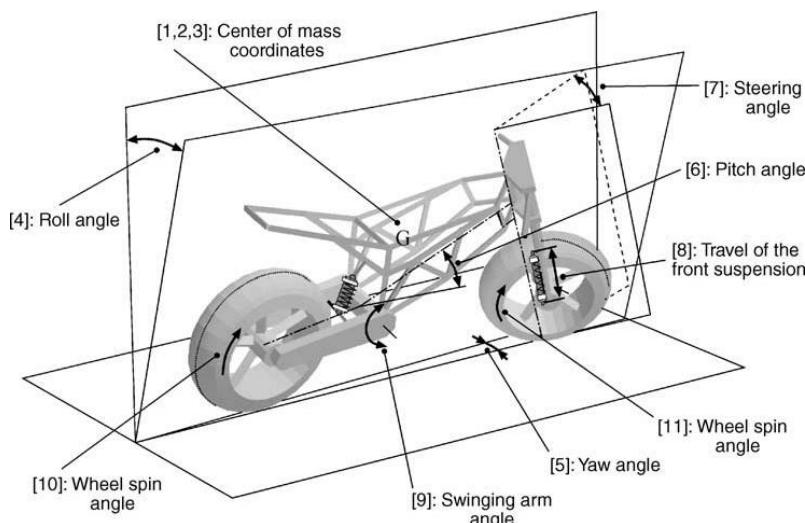
¹ Roll

² Yaw

³ Pitch

بنابراین در این سامانه در مجموع ۱۱ درجه آزادی وجود دارد. در حقیقت ۳ مختصه‌ی مکانی و ۳ زاویه‌ی رول و پیچ و یا برای تعیین مکان و چرخش دورانی بدن‌ی اصلی موتورسیکلت استفاده می‌شود. ۵ درجه‌ی آزادی دیگر برای تعیین موقعیت نسبی اعضاء نسبت به یکدیگر استفاده می‌شوند. این مدل در شکل ۴-۳ به خوبی نمایش داده شده است.

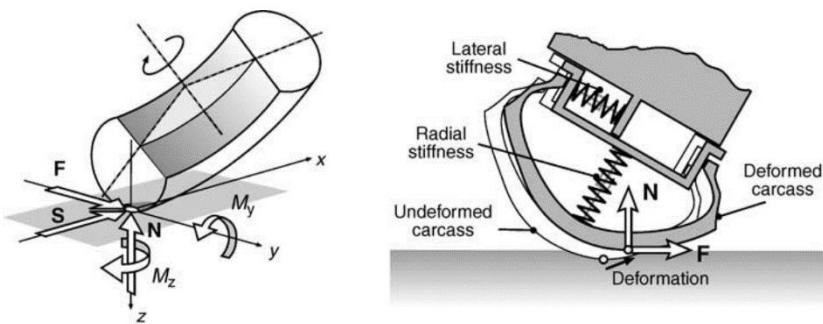
لازم به ذکر است، چنانچه قید عدم وجود حرکت نسبی بین چرخ جلو و عقب را تعریف نماییم؛ آنگاه می‌توان گفت که مدل ۱۰ درجه آزادی دارد [۱۰].



شکل ۴-۳ درجات آزادی مدل سه بعدی موتورسیکلت

در این مدل، اجزای بدن به عنوان جسم صلب در نظر گرفته شدند، چراکه موتورسیکلت‌های اسپورت، سختی یا سفتی بالایی دارند و انعطاف پذیری اندک سامانه، تاثیر ناچیزی بر روی تعادل خواهد داشت.

از جمله تفاوت‌هایی که به علت اضافه شدن بعد سوم ایجاد می‌شود، تفاوت در مدل‌سازی لاستیک‌ها می‌باشد. چراکه با توجه به رفتار غیرصلب لاستیک، با تغییر شکل لاستیک تحت اثر فشار ناشی از وزن موتورسیکلت، نیروی موثر دیگر به صورت نقطه‌ای و به شکل یک نیرو در جهت عمودی بر آن نقطه؛ وارد نمی‌شود. این موضوع به خوبی در شکل ۴-۳ نمایش داده شده است. با توجه به تغییر شکل لاستیک و تغییرات جهت و محل اثر نیروی سطح، چنانچه بخواهیم مدل‌سازی ریاضی انجام دهیم، کار بسیار پیچیده‌ای خواهد بود. در شبیه‌سازی انجام شده، ما برای لاستیک‌ها جنس را wheel در نظر می‌گیریم و طبیعتاً برخی از نکات مانند انعطاف پذیر بودن و خاصیت ارتجاعی اعمال می‌شود ولی به دلیل هوشمند بودن روش‌های کنترلی نیازی به مدل‌سازی ریاضی نمی‌باشد.



شکل ۵-۳ مدل‌سازی لاستیک و تاثیر تغییر شکل آن در محل اثر نیرو

۲-۳- مدل‌سازی در شبیه‌ساز

مدل‌سازی درست و اصولی از مهم‌ترین نکاتی است که در ابتدای فرآیند شبیه‌سازی در یک نرم افزار باید انجام شود. با توجه به توضیحات بخش قبل به خصوص در رابطه با مدل‌سازی سه بعدی، در این بخش نیز سعی می‌کنیم تا اعضا و مفاصل مختلفی از موتورسیکلت را برای نرم افزار تعریف نماییم. لازم به ذکر است که برخی از موارد مانند میزان سختی و صلابت اجسام با تعریف جنس آن‌ها به صورت خودکار اعمال می‌شود و همچنین می‌توان در نرم افزار توزیع جرمی و ماتریس اینرسی را برای هر جسم تعریف کرد. بنابراین، مدلی که در انتهای به دست می‌آید، مدلی به نسبت کامل‌تر از آنچه که در بخش‌های قبلی به آن‌ها پرداختیم، خواهد بود.

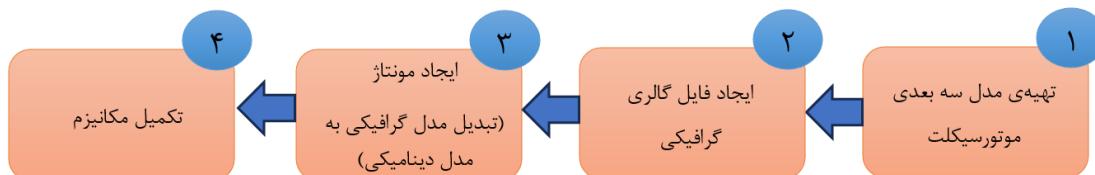
فرآیند مدل‌سازی در نرم افزار Vortex Studio شامل چهار مرحله می‌باشد که در تصویر زیر می‌توانید آن را مشاهده نمایید. پیش از ورود به نرم افزار باید مدل سه بعدی مورد نظر در یک نرم افزار جانی طراحی شود و سپس با وارد کردن آن در قسمت گالری گرافیکی^۱ نرم افزار آن را Vortex Studio آماده تبدیل به مونتاژ^۲ می‌کنیم. سپس با وارد شدن در بخش مکانیزم^۳ نرم افزار می‌توانیم ابتدا تبدیل گالری گرافیکی به مدل دینامیکی را انجام داده و کارهایی از قبیل مش زدن، ایجاد قیود و ... را انجام دهیم. انجام هریک از این مراحل، شامل جزئیات بسیاری می‌باشد که در این گزارش صرفاً به بیان

¹ Graphic Gallery

² Assembly

³ Mechanism

کلیاتی از هریک از مراحل می‌پردازیم و جهت اطلاعات بیشتر می‌توانید به آموزش قرار داده شده در سایت نرم افزار مراجعه نمایید [11]. مراحل تهیه‌ی مدل در شبیه‌ساز در شکل ۳-۶ نمایش داده شده است.



شکل ۳-۶ مراحل تهیه‌ی مدل در شبیه‌ساز

۱-۲-۳- طراحی مدل سه بعدی موتورسیکلت

این بخش می‌بایست در خارج از نرم افزار Vortex Studio انجام گیرد. به کمک نرم افزارهای طراحی سه بعدی مانند 3Dmax می‌توان این کار را انجام داد. برای وارد کردن فایل سه بعدی به قسمت گالری گرافیکی، مدل موردنظر باید با یکی از پسوندهایی که در جدول ۱-۳ آمده است، ذخیره شده باشد تا توسط نرم افزار Vortex Studio قابل شناسایی باشد.

جدول ۱-۳ پسوندهای مناسب فایل سه بعدی

*.cive	*.osg2
*.dae	*.osga
*.fbx	*.osgb
*.flt	*.osgs
*.ive	*.osgt
*.obj	*.osgx
*.osg	*.shp

البته در این نرم افزار امکان وارد کردن فایل‌های CAD نیز وجود دارد ولی نیازمند آن است که یک لایسنس مرتبط با ماژول تشخیص دهنده‌ی فایل‌های CAD را نصب نمایید. در این صورت به راحتی می‌توانید طراحی مدل را در نرم افزارهایی مانند Solid Works یا CATIA انجام دهید و فایل خروجی را در قالب مناسب برای اضافه کردن در نرم افزار Vortex Studio، ذخیره نمایید. پسوندهای قابل قبول برای فایل CAD در جدول ۲-۳ مشخص شده است.

جدول ۲-۳ پسوندهای مناسب فایل CAD

*.stl	*.3dxml
*.wrl	*.vrml
*.stp	*.step
*.sldasm	*.sldprt
*.catpart	*.catproduct

در این پروژه، از فایل vae. که طراحی سه بعدی موتورسیکلت در آن انجام شده بود، استفاده شده است. همچنین برای سایر قسمت‌هایی که برای کنترل موتور سیکلت اضافه شدند، مانند چرخ طیار، کره‌ی متعادل کننده و ...، ابتدا در نرم افزار CATIA مدل سه بعدی آن‌ها طراحی شده و سپس خروجی را قالب stl. ذخیره کرده و به کمک برخی سایتها تبدیل آن به فایل vae. انجام گرفته است.

۲-۴-۱- ایجاد فایل گالری گرافیکی

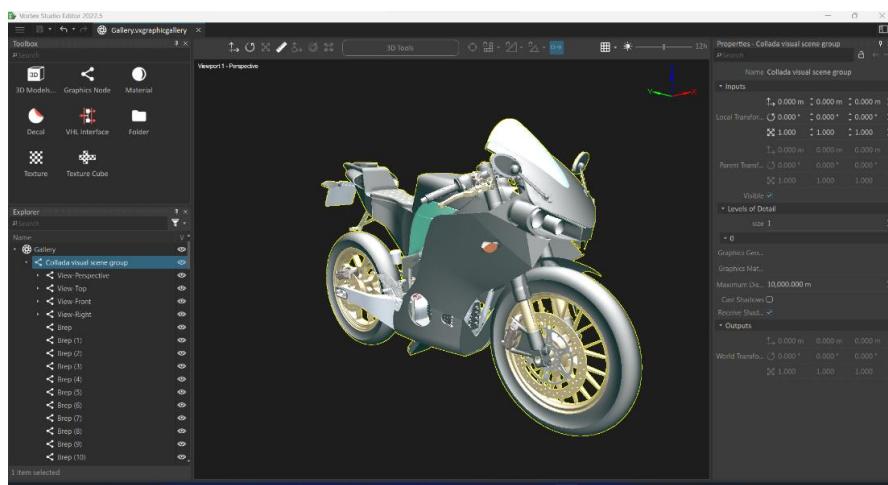
در این بخش و همچنین بخش‌های بعدی به صورت مستقیم وارد نرم افزار Vortex Studio و کار با آن می‌شویم. لازم است پس از باز کردن ویرایشگر^۱ نرم افزار، وارد گالری گرافیکی شده و سپس فایل سه بعدی vae. حاوی مدل موتورسیکلت را اضافه نماییم.

در گام بعدی، باید مدل وارد شده کمی ساده‌سازی و مناسب سازی شود، به‌طوری که جزئیات بیش از حد اجسام، ساده‌تر شده و برخی اجزای کوچک مثل پیچ‌ها و پین‌ها که تأثیر چندانی در رفتار اصلی سامانه ندارند، از مدل سه بعدی حذف شوند تا ضمن کاهش حجم فایل، از کاهش سرعت عملکرد نرم‌افزار نیز جلوگیری نمود.

در گالری گرافیکی، مفهومی تحت عنوان گره^۲ وجود دارد، که در مراحل بعدی از گره‌ها به عنوان بخش‌های سازنده موتورسیکلت استفاده می‌شود و در حقیقت هر گره که یک یا تعدادی جزء گرافیکی در داخل خود دارد، در نهایت به یک بخش از موتورسیکلت تبدیل می‌شود. در ابتدای وارد کردن فلیل همهی بخش‌های گرافیکی در داخل یک گره قرار دارند که در شکل ۷-۳ نمایش داده شده است، پس لازم است تا تعدادی گره تعریف نماییم و اجزای مختلف موتور سیکلت را به کمک آن‌ها دسته‌بندی کنیم.

¹ Editor

² Node

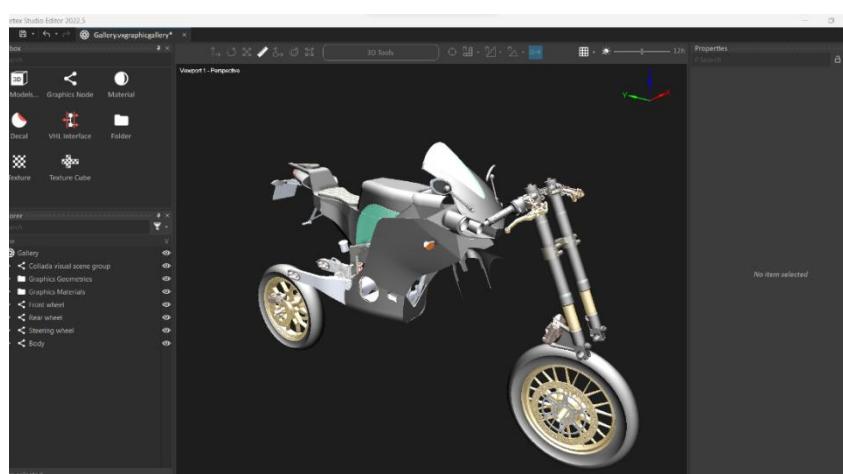


شکل ۳-۷ فایل سه بعدی موتورسیکلت اضافه شده در گالری گرافیکی

باتوجه به توضیحات بخش مدلسازی دینامیکی، سامانه‌ی موتورسیکلت با یک سامانه چهار جسمی که نسبت به یکدیگر متحرک میباشند، مدلسازی شد [۱۲] و اعضای آن به شرح زیر می‌باشند:

- چرخ جلو
- چرخ عقب
- دسته فرمان
- بدنی اصلی

براین اساس چهار نود تعریف می‌نماییم و قسمت‌های مختلف گرافیکی را در آن اضافه می‌نماییم. نتیجه در شکل ۸-۳ آمده است (صرفا برای نمایش بهتر، اجزا از یکدیگر فاصله گرفتند). به این ترتیب کار آماده سازی فایل گرافیکی به اتمام می‌رسد و لازم است آن را با پسوند .vxgraphicgallery ذخیره نماییم.



شکل ۸-۳ تعریف گره‌های گرافیکی برای مشخص کردن اعضای مختلف موتورسیکلت

لازم به ذکر است که می‌توان سیستم را به شکل دقیق‌تری با مدل‌سازی شش جسمی نیز نمایش داد تا تأثیر کمک فنر و میراگرها را نیز در شبیه‌سازی درنظر گرفت. برای این کار، دسته فرمان به بخش‌های بالا و پایین تقسیم شده و بدنه اصلی موتورسیکلت نیز به دو بخش اصلی و ناحیه‌ی فنربندی تقسیم شده و با محدودیت‌های مرتبط به هم متصل می‌شوند. به هرحال، این دقت بیشتر باعث مدل‌سازی واقع‌گرایانه‌تر و رفتار صحیح‌تر موتورسیکلت در عبور از موانع به دلیل وجود فنر و میراگرها می‌شود، ولی با افروختن این موارد به محاسبات نرم افزار نیز افزوده شده و باعث کاهش سرعت فریم شبیه‌سازی و کاستن از شرایط شبیه‌سازی بلادرنگ می‌شود.

۳-۲-۳- ایجاد مونتاژ

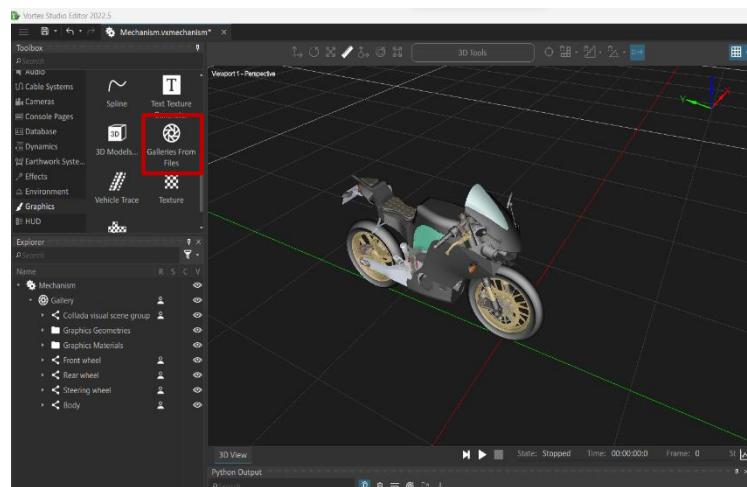
در این مرحله باید فایل گرافیکی ایجاد شده را به عنوان ورودی استفاده کرده و سپس با توجه به گره‌های گرافیکی تعریف شده در آن، هریک را به جسم صلبی تبدیل کرده و قیدهای لازم را بین این اعضاء تعریف نماییم.

برای شروع لازم است تا از طریق ویرایشگر نرم‌افزار Vortex Studio یک فایل مکانیزم ایجاد کنیم و سپس مراحل زیر را در آن انجام دهیم:

- ۱- وارد کردن گالری گرافیکی
- ۲- ساختن یک جسم صلب برای هریک از گره‌های گرافیکی
- ۳- تعریف هندسه‌ی برخورد^۱ هریک از اعضاء
- ۴- مشخص کردن جنس و جرم اعضاء
- ۵- ایجاد قید برخورد برای مونتاژ
- ۶- تعریف قیود و مفاصل برای اتصال اعضاء

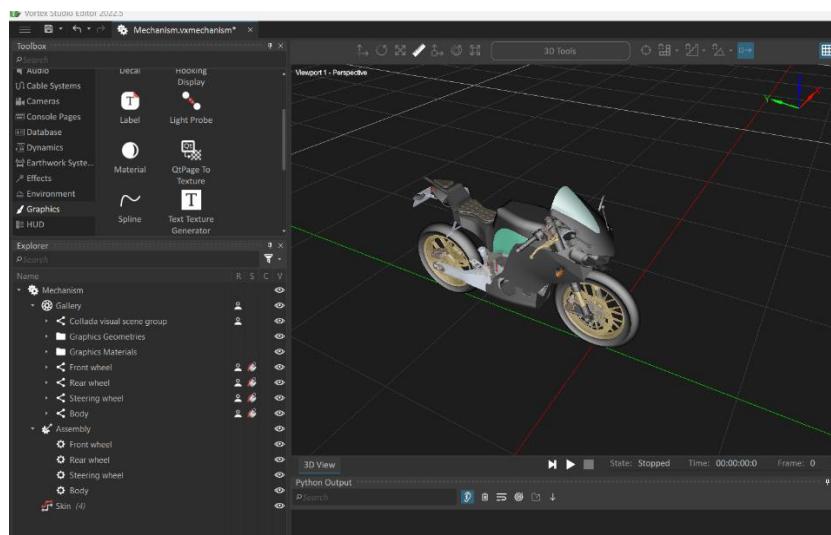
برای وارد کردن گالری گرافیکی ساخته شده، لازم است تا از قسمت جعبه ابزار وارد بخش گرافیک شده و گزینه‌ی مشخص شده در شکل ۹-۳ را انتخاب کرده و فایل vxgraphicsgallery. را انتخاب کنیم. در نهایت فایل موتور سیکلت در این قسمت اضافه می‌شود.

^۱ Collision Geometry



شکل ۹-۳ وارد کردن فایل گرافیکی به فایل مکانیزم

سپس برای انجام مرحله‌ی دوم لازم است تا روی هریک از گره‌های گالری گرافیکی کلیک راست کرده و گزینه‌ی «ایجاد یک جسم صلب» را برای گره اول و گزینه «ایجاد یک جسم صلب در قسمت مونتاژ» را برای گره‌های دیگر انتخاب می‌نماییم. در نهایت نتیجه مانند شکل ۱۰-۳ خواهد شد. لازم به ذکر است که همزمان با ایجاد شدن قسمت مونتاژ در فایل مکانیزم، یک فایل vxassembly در کنار سایر فایل‌ها ایجاد خواهد شد.



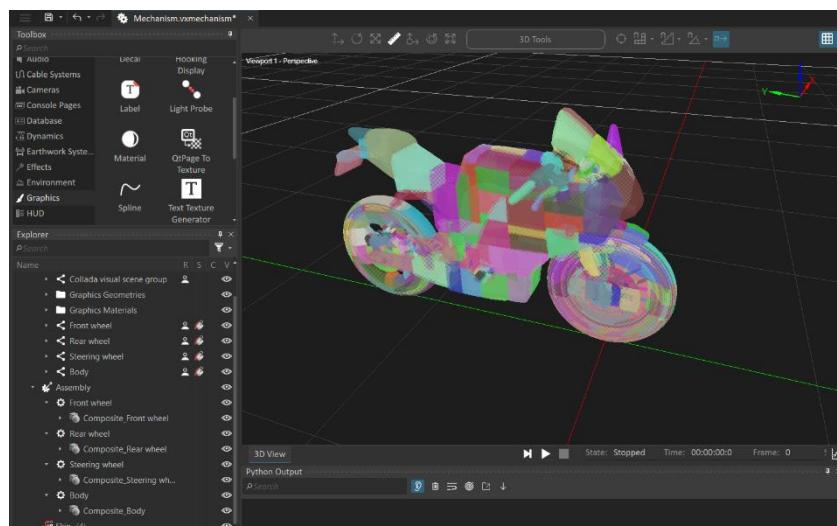
شکل ۱۰-۳ ایجاد جسم صلب برای گره‌های گرافیکی

در مرحله‌ی سوم، لازم است تا هندسه بروخورد را برای هریک اجسام ایجاد شده، تعریف نماییم. برای انجام آن لازم است تا روی هریک کلیک راست کرده و گزینه‌ی «ایجاد هندسه متناسب» را انتخا

میکنیم و سپس از بین انواع هندسه‌ها می‌توانیم یکی را انتخاب نماییم. در نرم افزار Vortex Studio، انواع مختلفی از هندسه برخوردها وجود دارد. نوع اول این هندسه‌ها، احجام هندسی از پیش تعریف شده؛ مانند مکعب، کره، استوانه، کپسول میباشد که نرم افزار به طور خودکار سعی می‌کند با نوع هندسه انتخاب شده از سوی کاربر، به بهترین نحو هندسه جسم را پوشش دهد. نوع دیگری از هندسه برخورد در این نرم افزار، پوشش اجسام به وسیله مش زنی و همچنین ایجاد یک هندسه برخورد مرکب از اشکال ساده میباشد تا بتواند به بهترین نحو این پوشش را ایجاد کند.

لازم به ذکر است، دقت مش زنی و ایجاد هندسه‌های برخورد، به طور مستقیم روی محاسبات برخوردها تأثیرگذار بوده و همچنین روی سرعت اجرای شبیه‌سازی نیز تأثیرگذار می‌باشد. با این حال با توجه به سخت افزار مناسبی که در اختیار داریم برای بالا رفتن دقت شبیه سازی، هندسه‌ها را از نوع مش محدب تجزیه‌ای انتخاب می‌کنیم. برای فرمان و بدنه سطح جزئیات را در تنظیمات آن روی مقدار کم و برای چرخ‌ها روی سطح متوسط تنظیم می‌نماییم.

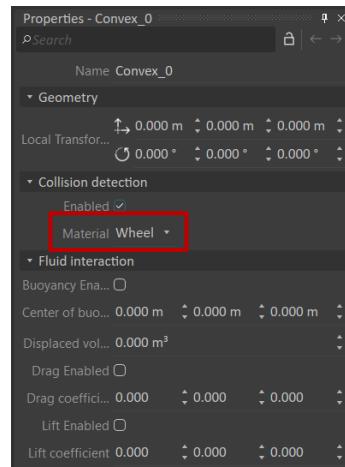
در نهایت پس از آنکه هندسه‌ی برخورد تمامی اعضاء را تعیین کردیم، نتیجه مانند شکل ۱۱-۳ خواهد شد.



شکل ۱۱-۳ ایجاد هندسه‌ی برخورد برای اعضاء

در مرحله چهارم، باید جنس اعضاء را مشخص نماییم. در نرم افزار باید جنس را به هندسه‌های برخورد ایجاد شده اعمال نماییم. برای این منظور ابتدا روی مونتاژ ایجاد شده در مکانیزم دوبار کلیک کرده و تغییرات را از این مرحله به بعد در آن فایل اعمال می‌نماییم. برای چرخ‌های جلو و عقب جنس را از نوع

چرخ انتخاب می‌کنیم و برای بدنه و فرمان جنس را شاسی قرار می‌دهیم که گزینه‌ی انتخابی برای چرخ را در شکل ۱۲-۳ مشاهده می‌کنید.



شکل ۱۲-۳ ایجاد هندسه‌ی برخورد برای اعضاء

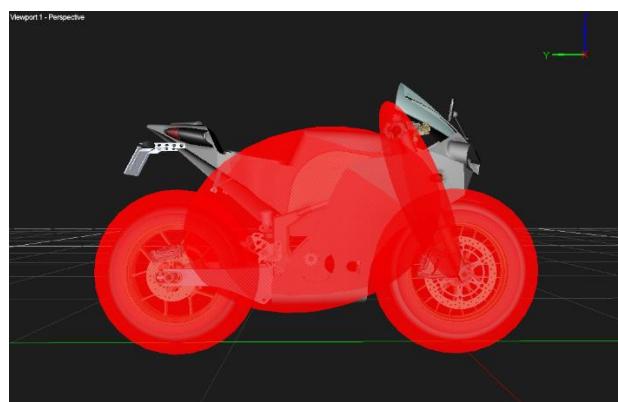
برای تعیین مقدار جرم و محاسبه‌ی ماتریس اینرسی، روی هریک از اعضاء کلیک کرده و از قسمت پارامترها در صفحه‌ی کوچک ویژگی‌ها^۱ در سمت راست، می‌توان مقدار جرم را تعیین کرد و سپس گزینه‌ی «محاسبه‌ی اینرسی و مرکز جرم» را انتخاب کرد. در این پژوهش سعی شده است تا با توجه به جرم اعضای موتورسیکلت‌های واقعی، جرم اعضای ایجاد شده، تعیین شوند. در جدول ۳-۳ می‌توانید جنس و جرم در نظر گرفته شده برای هریک از اعضاء را مشاهده نمایید. این موتورسیکلت در مجموع حدود ۱۸۰ کیلوگرم وزن دارد.

جدول ۳-۳ جرم و جنس هریک از اجسام صلب ایجاد شده.

نام عضو	جرم (kg)	جنس
Front wheel	۲۰	wheel
Rear wheel	۲۵	wheel
Steering wheel	۲۰	chassis
Body	۱۱۵	chassis

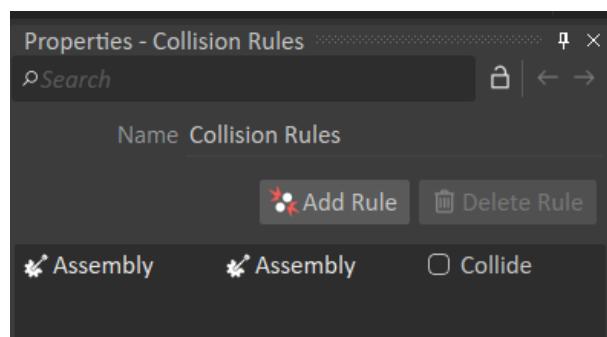
^۱ Properties

همچنین لازم به ذکر است که می‌توان توزیع اینرسی اجسام را نیز مشاهده کرد و حتی به صورت دستی شکل این توزیع را تغییر داد. برای مثال موتور یک موتورسیکلت برقی از جمله سنگین‌ترین قسمت‌ها می‌باشد پس می‌توان به صورت دستی بیشترین جرم را برای جلو و پایین بدن در نظر گرفت که موتور در آنجا قرار می‌گیرد. در نهایت توزیع اینرسی را در شکل ۱۳-۳ می‌توانید مشاهده نمایید.



شکل ۱۳-۳ ایجاد هندسه‌ی برخورد برای اعضاء

در مرحله‌ی پنجم، قید برخورد را برای مونتاژ ایجاد کنیم. لازم است تا از قسمت Basics در جعبه ابزار بر روی گزینه‌ی قیود برخورد، دوبار کلیک نماییم و سپس در صفحه‌ی ویژگی‌ها بر روی گزینه‌ی ایجاد قانون بزنیم و در هردو سلول خالی، کل مونتاژ را قرار دهیم. نتیجه را در شکل ۱۴-۳ مشاهده می‌کنید.



شکل ۱۴-۳ ایجاد قید برخورد

در مرحله‌ی پایانی لازم است تا بین اجزا قیدها یا درواقع مفاصلی را تعریف نماییم. تمامی قیدها از نوع لولا^۱ می‌باشند. جزئیات دقیق‌تر را می‌توانید در جدول ۴-۳ مشاهده نمایید. از قسمت محورهای کنترلی،

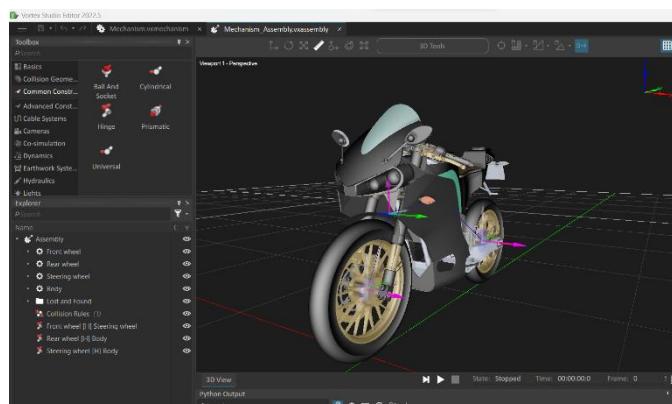
^۱ Hinge

باید مشخص نماییم که مفصل ما دارای موتور باشد، قفل باشد و یا به صورت آزادانه با توجه به نیروهای خارجی که ممان ایجاد می‌کنند به گردش در بیاید که فرمان را قفل و چرخ جلو را آزاد و چرخ عقب را متصل به موتور می‌کنیم.

جدول ۳-۴ جرم و جنس هریک از اجسام صلب ایجاد شده.

نام قید	نوع	عضو اول	عضو دوم	مکان	محور اولیه (جسم اول)	محور ثانویه (جسم دوم)
Front wheel [H] Steering wheel	لولا	چرخ جلو	فرمان	0.0m; 0.0m; 0.0m	+X: 1m; 0m; 0m	+Y: 0m; 1m; 0m
Rear wheel [H] Body	لولا	چرخ عقب	بدنه	0.0m; 0.0m; 0.0m	+X: 1m; 0m; 0m	+Y: 0m; 1m; 0m
Steering wheel [H] Body	لولا	فرمان	بدنه	0.0m; 0.0m; 0.0m	+Z: 0m; 0m; 1m	+X: 1m; 0m; 0m

در نهایت چنانچه قیدهای ایجاد شده در شکل ۳-۱۵ قابل مشاهده می‌باشد.



شکل ۳-۱۵ مفاصل ایجاد شده بین اجزا

۳-۲-۴- تکمیل مکانیزم

در این پروژه در حقیقت مدل‌سازی موتورسیکلت، بعد از مرحله‌ی قبل تمام شده است. فابل مونتاژ را می‌بندیم و به فایل مکانیزم برمی‌گردیم. می‌توان برای ساماندهی و نظم بیشتر پوشش‌هایی را ایجاد کرد.

همچنین در این قسمت می‌نوانیم مونتاژهای دیگری را در صورت نیاز اضافه نماییم. در نهایت، محتویات این قسمت نیز در قالب یک فایل vxmechanism ذخیره خواهد شد.

در حقیقت این مرحله، به منظور تخصیص شرایط اولیه و شرایط عملکردی و اعمال ورودی‌ها، تعریف کنترل‌کننده و ایجاد بلوک‌های کنترلی و پردازش خروجی‌های سامانه می‌باشد. با ایجاد یک فضای اتصال^۱ و ارسال داده‌ها میان بخش‌های مختلف سامانه و ارسال شرایط بخش‌های مختلف سامانه در هر لحظه به کنترل کننده به عنوان ورودی و ارسال خروجی کنترل کننده به عضو برقرار کننده تعادل، برای حفظ تعادل موتورسیکلت انجام می‌شود.

باتوجه به نوع کنترل‌راتخابی و جسم برقرار کننده تعادل، در انجام این مرحله کمی تفاوت ایجاد می‌شود و در بخش‌های آتی در رابطه با آن توضیحات بیشتری خواهیم داد.

^۱ Connection container

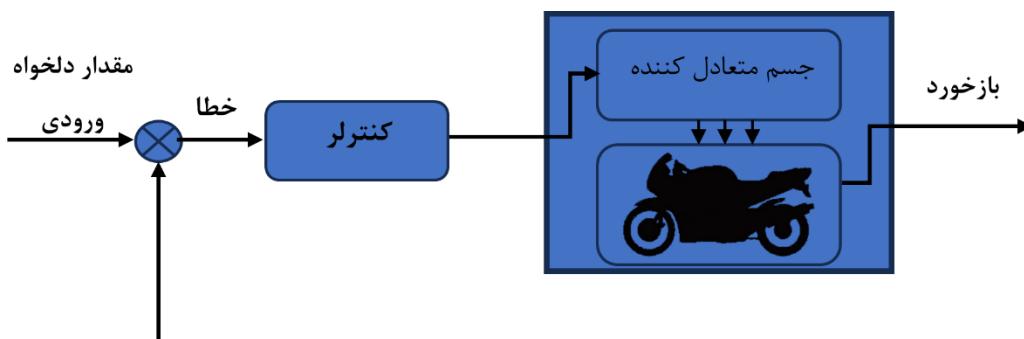
فصل چهارم

طراحی کنترل کننده

طراحی کنترل کننده

هدف اصلی در این پژوهش حفظ تعادل موتور سیکلت به کمک روش‌های هوشمند می‌باشد. در حقیقت، هدف طراحی کلاسیک کنترلر نمی‌باشد که برای مثال ابتدا مدل ریاضی را به دست آوریم و سپس آن را ساده کنیم و در آخر تابع تبدیل را به دست آورده و برای آن کنترلر طراحی نماییم. البته مدل‌های به دست آمده و کنترلرهای طراحی شده در این روش دید خوبی برای کنترل موتور سیکلت به روش‌های هوشمند می‌دهند.

در این پژوهش، موتور سیکلت و جسمی که به عنوان برقرار کننده تعادل استفاده می‌شود، داخل یک بلوک دیاگرام در نظر گرفته می‌شوند و ما با ورودی دادن و فیدبک گرفتن سعی می‌کنیم تعادل موتور سیکلت را حفظ نماییم. برای درک بهتر می‌توانید، شکل ۱-۴ را مشاهده کنید.



شکل ۱-۴ دیاگرام کنترل کننده حلقه بسته

البته این یک طراحی حلقه بسته می‌باشد ولی برای مثال چنانچه از ژایروسکوپ برای حفظ تعادل استفاده کنیم، حلقه‌ی بازخورد نداریم و کنترل حلقه باز می‌باشد.

در این فصل ابتدا به معرفی کنترلر تناسبی-انتگرالی-مشتقی (PID)^۱ و نحوه پیاده سازی آن در نرم افزار Vortex Studio می‌پردازیم. سپس یادگیری تقویتی^۲ که یکی از روش‌های یادگیری در هوش مصنوعی می‌باشد را معرفی می‌کنیم که در این پژوهش به عنوان یک کنترلر به ما کمک می‌کند تا تعادل

¹ Proportional–Integral–Derivative

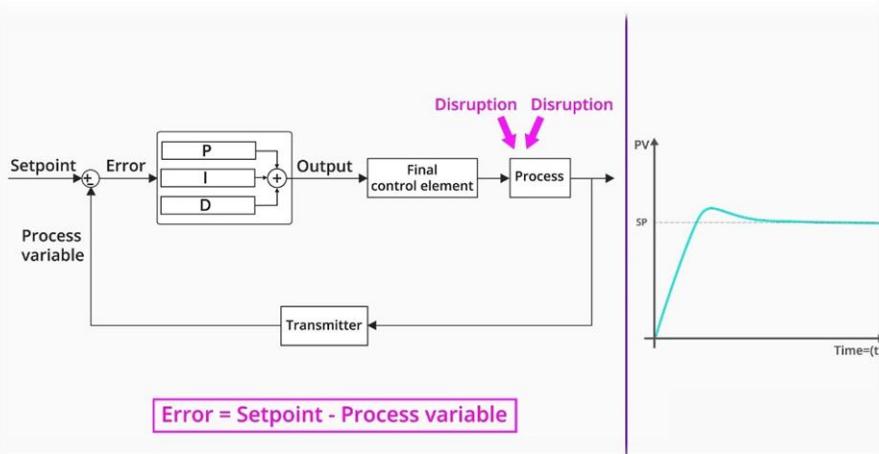
² Reinforcement learning

موتور سیکلت را نگه داریم. همچنین در ادامه نیز به نحوه پیاده سازی و آموزش مدل یادگیری در Vortex Studio می پردازیم.

۴-۱- کنترل PID

کنترل کننده PID یا تناسبی-انتگرالی-مشتقی، یک الگوریتم و روش کنترل حلقه بسته با بهره گیری از مفهوم بازخورد است که در بسیاری از فرایندهای صنعتی برای کنترل سرعت موتورهای DC، کنترل فشار، کنترل دما و... به کار می رود.

این کنترلرها در حقیقت مقدار کنترلی که باید به سامانه اعمال شود را از مجموع وزن دار سه ترم محاسبه می کنند و هریک از این ترمها با توجه به خطای بین مقدار حالت دلخواه و مقدار حالت فعلی، به نحوی به دست می آیند و واکنش متفاوتی نسبت به خطا دارند. این سه بخش به عنوان سه بلاک مجزا در شکل ۲-۴ نمایش داده شده اند [۱۳].



شکل ۲-۴ اجزای مختلف کنترل PID

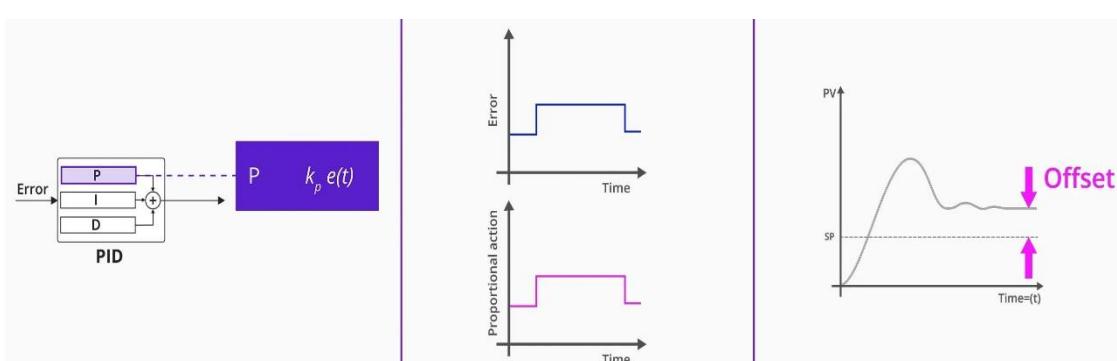
مقدار پاسخ تولیدی هر بخش کنترلی را می توان با تغییر تنظیمات کنترل کننده سامانه، تغییر داد. در ادامه، هر یک از این بخش ها و نحوه تنظیم آن ها را توضیح می دهیم.

۴-۱-۱- نحوه عملکرد کنترلر PID

سه بخش کنترلی که در مقدمه‌ی این بخش به صورت مختصر به آن‌ها اشاره شد، عبارتند از: کنترل تناسبی، کنترل انتگرالی و کنترل مشتقی. حال به صورت دقیق‌تر در ادامه هریک را توضیح می‌دهیم.

۱- کنترل تناسبی: ترم کنترل تناسبی، در اغلب موارد نیروی محرک کنترل کننده است. این ترم، خروجی کنترل کننده را متناسب با مقدار خطا تغییر می‌دهد. اگر خطا بزرگ شود، عمل یا کنش کنترلی نیز بزرگ‌تر می‌شود. پارامتر قابل تنظیم کنترل تناسبی، بهره کنترل کننده یا K_p نامیده می‌شود. هرچه بهره کنترل کننده بزرگ‌تر باشد، تاثیر عمل کنترل تناسبی خطا، افزایش می‌یابد. اگر بهره کنترل کننده در مقدار بسیار بالایی تنظیم شود، حلقه کنترل شروع به نوسان می‌کند و ناپایدار می‌شود. از سوی دیگر، اگر بهره بسیار کم باشد، پاسخ به اغتشاشات و خطای ایجاد شده، به اندازه کافی کارساز نخواهد بود. همچنین با افزایش مقدار K_p سرعت پاسخ دادن کنترلر نیز افزایش می‌یابد. تنظیم بهره کنترل کننده، بر ترم‌های انتگرالی و مشتقی نیز تأثیر می‌گذارد. به همین دلیل است که این پارامتر را به جای بهره تناسبی، بهره کنترل کننده می‌نامیم.

همانطور که از رابطه موجود در شکل ۳-۴ مشخص است، چنانچه خطا صفر شود مقدار خروجی این ترم کنترلی نیز صفر خواهد شد. در برخی از کاربردها، چنانچه این ترم به تنها بی استفاده شود لازم است تا به کمک یک مقدار بایاس یا به صورت دستی، خطا را جبران کرد، چراکه در این کاربردها هیچ گاه سامانه به حالت پایا نمی‌رسد و دائمًا حول نقطه تعادل نوسان می‌کند.

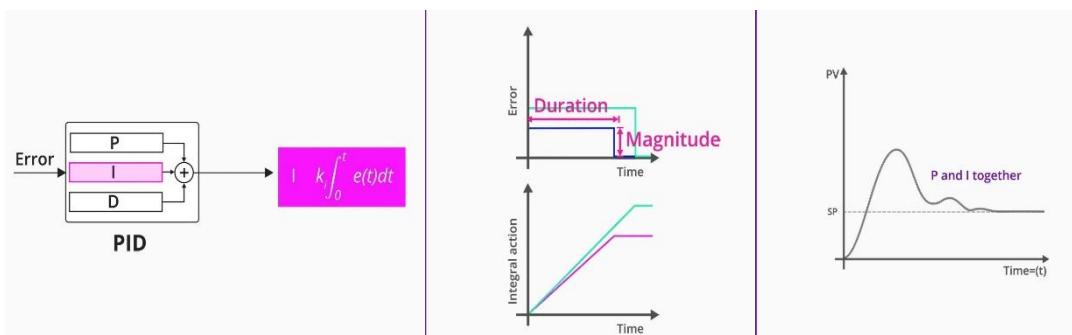


شکل ۳-۴ تاثیر بلوک P در خروجی کنترلر

۲- کنترل انتگرالی: ترم انتگرالی، یک خروجی متناسب با مدت زمانی سیگنال خطا و اندازه‌ی آن ایجاد می‌کند. هرچه خطا برای مدت طولانی‌تری در سیستم باقی بماند و هرچه مقدار آن هم بیش‌تر باشد،

خروجی ترم انتگرالی نیز بزرگ‌تر خواهد بود. بنابراین تا زمانی که خطای وجود داشته باشد، عمل انتگرال نیز ادامه خواهد یافت.

ترم کنترلی تناسبی در برخی کاربردها محدودیتی دارد که به موجب آن یک خطای دائم در سیستم باقی خواهد ماند؛ در این حالت لازم است تا از کنترلر انتگرالی برای جبران خطای پایا استفاده کرد. این موضوع در شکل ۴-۴ قابل مشاهده می‌باشد.



شکل ۴-۴ تاثیر بلوک I در خروجی کنترلر

زمانی که مقدار خطای منفی شود، مقدار ترم انتگرالی نیز کاهش می‌یابد. همچنین سرعت پاسخ‌دهی سیستم را محدود می‌کند و روی پایداری سیستم اثر می‌گذارد. هرچه مقدار ضریب K_I کاهش یابد، سرعت پاسخ‌دهی سیستم نیز افزایش می‌یابد. همچنین برای جلوگیری از شرایط رشد بیش از حد ترم انتگرالی^۱ که به دلیل وجود برخی پدیده‌های غیرخطی سامانه است، لازم است خروجی ترم انتگرالی حتماً محدود شود و سامانه ناپایدار نشود.

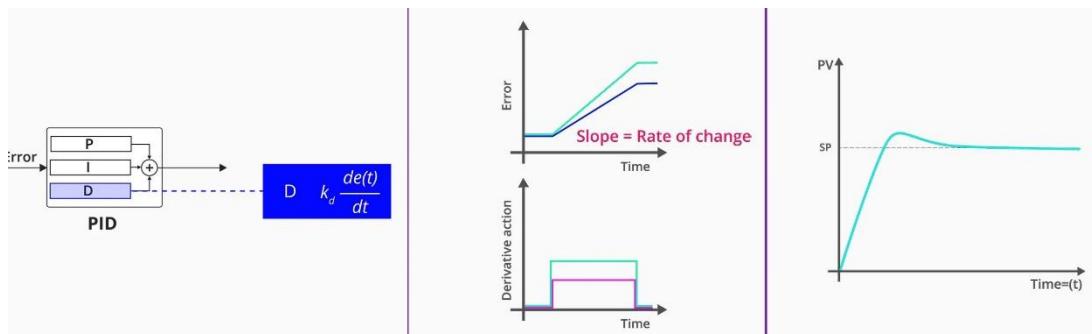
۳- کنترل مشتقی: ترم کنترل مشتقی، یک خروجی را بر اساس میزان تغییرات خطای تولید می‌کند. وقتی تغییرات خطای زیاد باشد یا در حقیقت تغییرات سریع تر رخ دهن، ترم مشتقی، مقدار کنترلی بیشتری تولید خواهد کرد. وقتی خطای تغییر نکند، مقدار ترم مشتقی صفر خواهد بود.

همانطور که در شکل ۴-۵ مشاهده می‌شود، کنترل مشتقی یک ترم پیش‌بینی کننده است و با توجه به خطایی که در آینده قرار است تولید شود، مقداری را به عنوان خروجی می‌دهد تا آن را جبران کند.

^۱ Integral wind up conditions

همچنین زیاد کردن مقدار K_d باعث می‌شود سرعت پاسخگویی سیستم افزایش یابد و همچنین ترم

مشتقی باعث می‌شود مقدار فراجهش^۱ نیز کاهش یابد.



شکل ۴-۵ تاثیر بلوک D در خروجی کنترلر

در نهایت با کنار هم قرار دادن این سه بلوک در کنار یکدیگر به رابطه‌ی زیر خواهیم رسید.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t) \quad (4-4)$$

همانطور که از رابطه‌ی (4-4) بر می‌آید، لازم است تا هر سه ضریب K_p , K_i و K_d را تنظیم نماییم که در ادامه به نحوه‌ی تنظیم این ضرایب می‌پردازیم.

۴-۱-۲- نحوه‌ی تنظیم ضرایب

ضرایب موجود در رابطه‌ی (4-4) بسته به سیستمی که می‌خواهیم کنترل کنیم، می‌توانند مقادیر متفاوتی اختیار کنند. بنابراین بازه‌ی انتخابی بسیار گسترده است و باید به صورت هوشمندانه‌ای آن‌ها را تنظیم کرد.

از جمله مقادیری که هریک از این ضرایب می‌توانند داشته باشند، مقدار صفر می‌باشد. بسته به اینکه کدام ضریب صفر شود، می‌توانیم کنترلر متفاوتی داشته باشیم. از جمله کنترلرهای مرسوم دیگر همانند PID که برای کاربردهای مختلف، استفاده می‌شوند کنترلرهای P, PI و PD می‌باشند. در اغلب کاربردها کنترلر PD گرینه‌ی مناسبی می‌باشد اگر قرار باشد از ترم انتگرالی نیز استفاده شود، معمولاً مقدار کوچکی برای آن در نظر گرفته می‌شود تا افزایش ترم انتگرالی موجب ناپایداری سیستم مورد نظر نشود.

^۱ Overshoot

روش‌های متفاوتی برای تنظیم ضرایب وجود دارد تا بتوان مقدار دلخواه را در خروجی کنترلر دریافت کرد. در زیر به صورت مختصر به سه تا از معروف‌ترین روش‌ها اشاره خواهیم کرد.

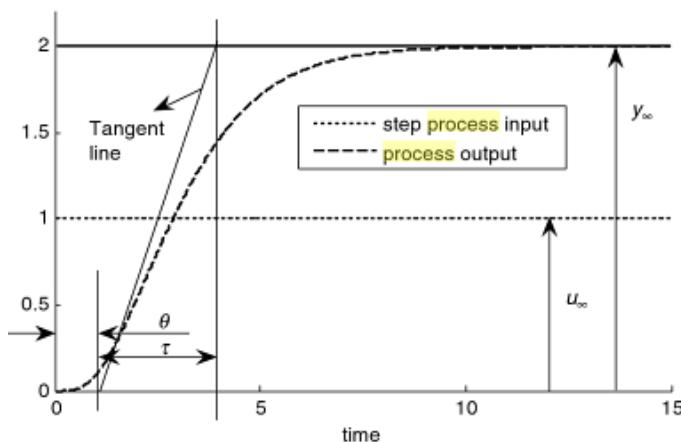
۱- روش سعی و خطأ: ساده‌ترین روش تنظیم ضرایب PID می‌باشد. البته در این روش نیز نکاتی را باید در نظر گرفت تا تنظیم ضرایب به درستی و در زمانی مناسب انجام گیرد. زمانی که سیستم و کنترلر در حال کار می‌باشند می‌توان ضرایب را تنظیم کرد.

- **راه حل اول:** ابتدا مقدار ضرایب K_i و K_d برابر با صفر در نظر گرفته می‌شوند. سپس مقدار ضریب تناسبی (K_p) را به آرامی افزایش می‌دهیم تا سیستم به رفتار نوسانی برسد. سپس در این حالت لازم است ضریب انتگرالی را تنظیم کرد سیستم از نوسان خارج شود. در آخر نیز ضریب مشتقی به نحوی تعیین می‌شود که زودتر به پاسخ برسیم و همچنین مقدار فراجهش نیز کاهش یابد.

- **راه حل دوم:** همچنین راه حل پیشنهادی دیگر در این روش آن است که ابتدا مقدار ضریب تناسبی به نحوی انتخاب شود که نوسان زیادی در سیستم دیده نشود. سپس ضریب مشتقی باید انتخاب شود. مقدار آن برای شروع حدود ۰.۱ مقدار تناسبی باشد. چنانچه نوسان در سیستم مشاهده نشد آن را زیاد می‌کنیم تا نوسان مشاهده شود و اگر با همان مقدار ۰.۱ ضریب تناسبی، شاهد نوسان بودیم آن را کاهش می‌دهیم تا نوسان ازین برود. سپس مقدار ضریب تناسبی را مجدد به اندازه ۲ تا ۳ برابر زیاد می‌کنیم به نحوی که شاهد نوسان باشیم. سپس به سراغ ضریب انتگرالی رفته و با مقدار اولیه‌ی ۰.۰۰۰ ضریب تناسبی شروع می‌کنیم و اگر نوسان داشتیم باید K_i کاهش یابد و چنانچه نوسانی وجود نداشت، می‌بایست افزایش یابد.

۲- روش منحنی واکنش فرآیند^۱: یک روش حلقه باز است. در این روش به سامانه یک ورودی پله واحد داده می‌شود و یک پاسخی از آن دریافت می‌شود. در گام اول باید یک سری مقادیر کنترلی به صورت دستی به سیستم بدهیم و منحنی پاسخ را ذخیره نماییم. سپس لازم است تا شیب، زمان مرگ، زمان تغییر شیب منحنی را به دست آورده و آن‌ها را در معادلات ترم‌های تناسبی، انتگرالی و مشتقی وارد کرده تا مقدار کنترلی حساب شود. در شکل ۶-۴ نمونه‌ای ازین نمودارها قرار گرفته است.

^۱ Process Reaction Curve Technique



شکل ۶-۴ منحنی واکنش فرآیند

۳- روش زیگلر-نیکولز^۱: یک روش حلقه بسته برای تنظیم ضرایب PID می‌باشد. در این روش نیز ابتدا باید مقدار ضریب تناسبی روی یک مقدار ثابت تنظیم شود درحالی که مقدار K_i و K_d صفر خواهند بود. مقدار ضریب تناسبی افزایش می‌یابد تا زمانی که سیستم در یک دامنه ثابت نوسان کند. به ضریبی که باعث تولید نوسان با دامنه ثابت شود، بهره نهایی^۲ (K_u) و به دوره زمانی نوسان، دوره‌ی نهایی (P_c) گویند. زمانی که به این بهره دست یافتیم، می‌توان مقادیر تناسبی و انتگرالی و مشتقی را براساس نوع کنترلر از جدول ۱-۴ که به جدول زیگلر-نیکولز معروف است، تعیین نمود.

جدول ۱-۴ جرم و جنس هریک از اجسام صلب ایجاد شده.

	K_c	T_I	T_D
P	$K_u/2$	-	-
PI	$K_u/2.2$	$P_u/1.2$	-
PID	$K_u/1.7$	$P_u/2$	$P_u/8$

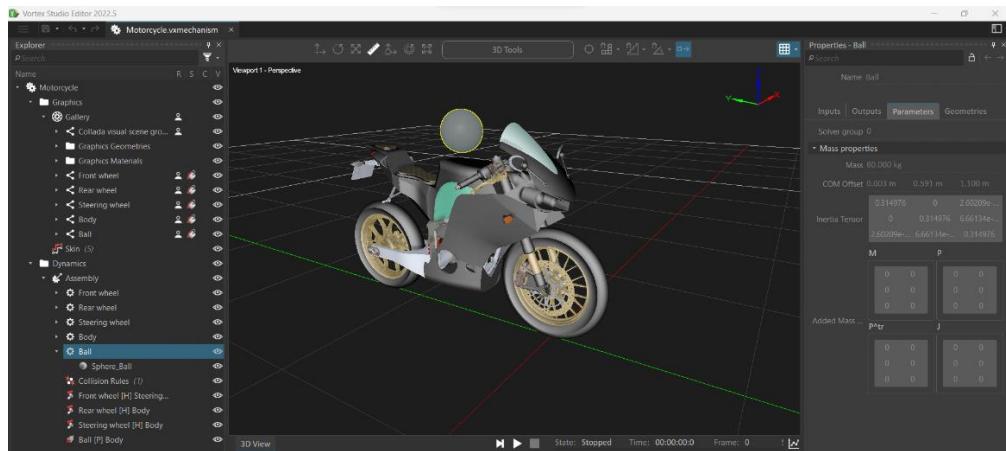
ما در این پژوهش برای تنظیم ضرایب از راه حل دوم مربوط به روش اول استفاده کردیم و برای جلوگیری از مشکل تجمع خطای انتگرالی و ناپایدار شدن سیستم از کنترلر PD استفاده کردیم. البته راه حل دیگر استفاده از ترم ضدجمع شوندگی^۳ می‌باشد که از این مشکل جلوگیری به عمل می‌آورد.

¹ Zeigler-Nichols method² Ultimate gain³ Anti-windup

۴-۱-۳- پیاده‌سازی در نرم افزار Vortex Studio

در این قسمت به روش‌های پیاده سازی شده برای کنترل موتورسیکلت در نرم افزار Vortex Studio با تمرکز بر کنترل به کمک PID پرداخته می‌شود. در ابتدا سعی بر آن بود تا به کمک یک گوی که در واقع مدل ساده شده‌ای از جرم انسان است، تعادل موتورسیکلت حفظ شود. پس از عدم موفقیت در آن، به حفظ تعادل به کمک فرمان موتورسیکلت پرداختیم. در هردوی این روش‌ها به نظر می‌رسید کنترلر PID، کنترلر مناسبی برای کنترل موتورسیکلت نیستند و روش‌هایی مانند کنترلر فازی یا کنترلر پیش‌بینی‌کننده مدل و یا ... شاید برای این امر مناسب باشند. در نهایت به کمک یک چرخ طیار و کنترلر PID توانستیم تعادل موتورسیکلت را برقرار نماییم. در ادامه به صورت مختصر به هریک از روش‌های گفته شده می‌پردازیم.

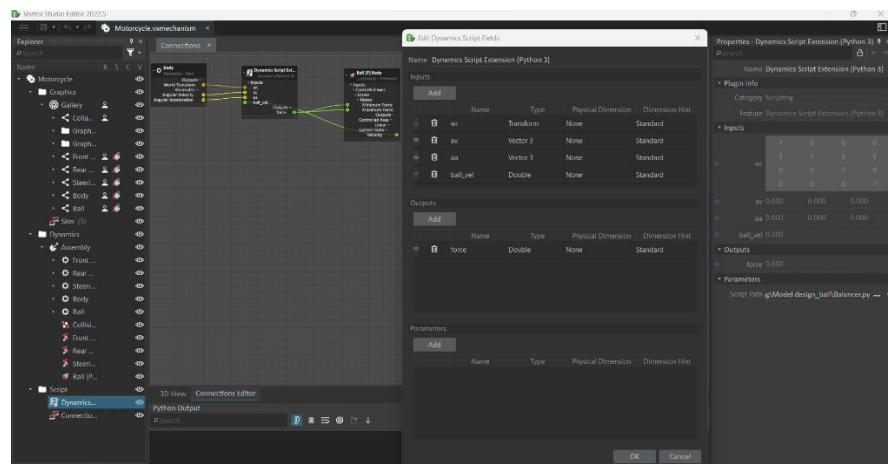
روش استفاده از جرم کنترلی: برای این منظور لازم است تا تقریباً در بالای مرکز ثقل موتورسیکلت یک گوی به عنوان جرم متمرکز که مدلی ساده شده از راننده می‌باشد اضافه نماییم. برای انجام آن لازم است مانند مباحث فصل قبل، یک گالری گرافیکی از گوی اضافه نموده و سپس در قسمت مونتاژ، جسمی صلب از آن بسازیم و جرم و سایر مشخصات را تعیین نماییم. برای جرم مقدار ۶۰ کیلوگرم که تقریباً برابر با جرم یک انسان معمولی می‌باشد را در نظر می‌گیریم. همچنین یک قید پریزماتیک نیز برای جابه‌جایی گوی در نظر می‌گیریم. در نهایت مدل در شکل ۷-۴ قابل مشاهده می‌باشد.



شکل ۷-۴ اضافه کردن جرم کنترلی به موتورسیکلت

سپس برای طراحی کنترلر از برنامه نویسی به زبان پایتون کمک می‌گیریم. برای این امر لازم است تا از قسمت جعبه ابزار اسکریپت را به مدلمن اضافه نماییم. همچنین در نهایت باید ورودی‌های نرم افزار به کد

و خروجی‌های کد به نرم افزار را مشخص نماییم و از طریق ارتباطات^۱ مقادیر خروجی مورد نظر نرم افزار را به ورودی‌های کد متصل می‌کنیم. موارد توضیح داده شده را در شکل ۴-۸ می‌توانید مشاهده نمایید.



شکل ۴-۸ ارتباطات موتورسیکلت با وزنه‌ی تعادلی و ورودی و خروجی‌های کد

در نهایت لازم است تا در پایتون کنترلر PID مورد نظر را طراحی کنیم. باید کتابخانه‌ی Vortex را به آن اضافه نماییم و به کمک رابطه‌های تعریف شده می‌توان به ورودی‌های تعریف شده داخل نرم‌افزار دسترسی یافت. همچنین لازم به ذکر است که هریک از مقادیر زاویه‌ی انحراف، سرعت زاویه‌ای و شتاب زاویه‌ای موتورسیکلت و همچنین سرعت گوی حائز اهمیت می‌باشد. پس کنترلر طراحی شده باید خطای حاصل از هریک را برای تولید مقدار کنترلی در نظر بگیرد و می‌توان برای مثال چهار کنترلر PID را به صورت موازی در نظر گرفت و رخروجی آن‌ها را با هم جمع کرد و نیروی لازم برای به حرکت درآوردن گویی به عنوان خروجی کد را ایجاد کرد. نحوه‌ی گرفتن ورودی و دادن خروجی در کد در شکل ۹-۴ آمده است.

```

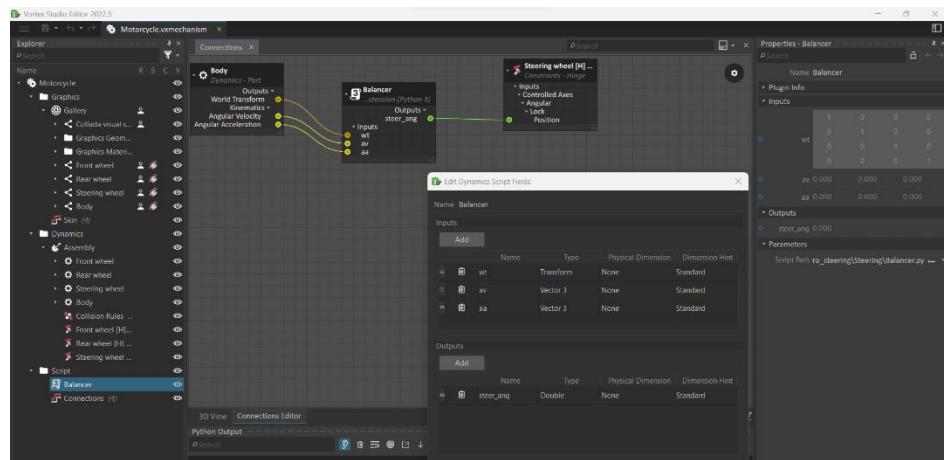
102 def post_step(extension):
103     """ Called after the collision detection and after the dynamic solver.
104     Use this method to set outputs or get values from dynamics objects.
105     Parameters
106     -----
107     extension : object
108         | The DynamicsScript extension referring to this script.
109         |
110     theta = roll_finder(extension.inputs.wt.value)
111     thetaDot = extension.inputs.av.value[1]
112     thetaDotDot = extension.inputs.aa.value[1]
113     ball_vel = extension.inputs.ball_vel.value
114
115     force = get_torque(theta,thetaDot, thetaDotDot, ball_vel)
116
117     extension.outputs.force.value = force

```

شکل ۹-۴ ارتباطات موتورسیکلت با وزنه‌ی تعادلی و ورودی و خروجی‌های کد

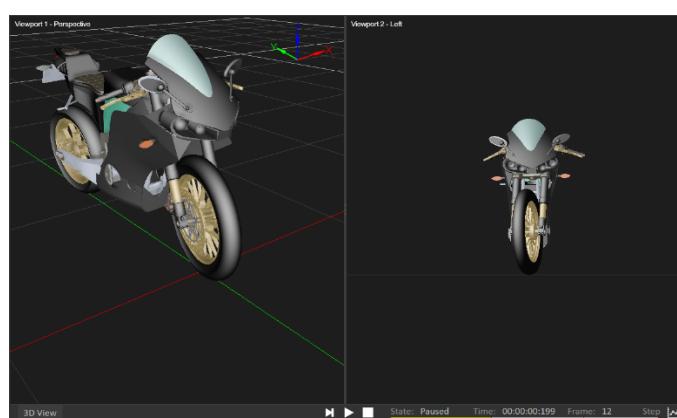
^۱ Connections

حفظ تعادل به کمک فرمان: در این روش هدف آن است تا به کمک چرخش فرمان حول محور عمودی بتوان تعادل موتورسیکلت را حفظ کرد. پیاده سازی این بخش نسبت به بخش قبل ساده‌تر می‌باشد و نیازی به اضافه کردن جسم صلب جدیدی نیست و فقط لازم است تا کد پایتون و کنترلر PID را مانند قسمت قبل اضافه نماییم. همچنین لازم است زاویه انحراف، سرعت زاویه‌ای و شتاب زاویه‌ای به عنوان ورودی کد و زاویه‌ی چرخش فرمان به عنوان خروجی تعریف شود. در قسمت ارتباطات نیز لازم است قسمت‌های مرتبط را به یک دیگر متصل نماییم. در شکل ۱۰-۴ جزئیات لازم فراهم شده است.



شکل ۱۰-۴ ارتباطات موتورسیکلت برای کنترل با فرمان

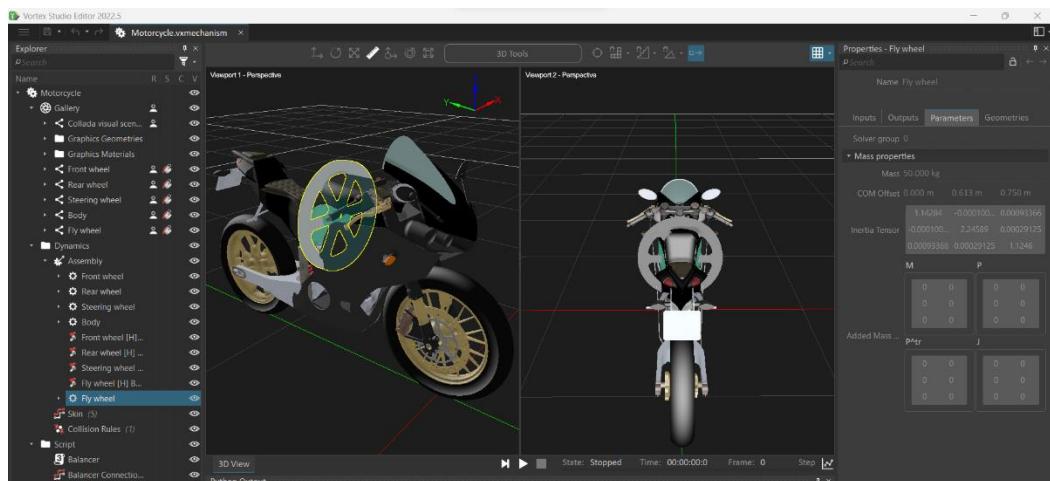
در این بخش به علت محدودیتی که در تعریف قید لولا داشتیم و نمی‌توانستیم آن را به صورت مایل تعریف نماییم و همچنین ساده‌بودن کنترلر PID برای این امر، نتوانستیم تعادل موتورسیکلت را تضمین نماییم و پس از مدتی تعادل موتورسیکلت از بین می‌رود. در شکل ۱۱-۴ قسمتی از نحوه چرخش فرمان نمایش داده شده است.



شکل ۱۱-۴ ارتباطات موتورسیکلت برای کنترل با فرمان

حفظ تعادل به کمک چرخ طیار: توضیحات مربوط به ریاضیات این بخش در قسمت مدلسازی داده شده است. در حقیقت در اینجا به دنبال ایجاد ممانی برای جبران ممان ایجاد شده توسط انحراف موتورسیکلت از حالت ایستاده می‌باشیم. بنابراین لازم است محور چرخش چرخ طیار هم راستا با محور Roll موتور سیکلت باشد اما جهت اعمال ممان به چرخ طیار خلاف جهت ممان ایجاد شده در موتورسیکلت باشد.^[۱۴]

برای این بخش نیز لازم است ابتدا گالری گرافیکی را اضافه کرده و سپس آن را تبدیل به یک جسم صلب در قسمت مونتاژ نماییم. همچنین لازم است یک قید لولا بین بدنه موتور سیکلت و همچنین چرخ طیار تعریف نماییم. جرم چرخ طیار برابر با ۵۰ کیلوگرم در نظر میگیریم و همچنین قید لولا را به صورت موتور در نظر می‌گیریم. در نهایت مدل مورد نظر در شکل ۱۲-۴ قابل مشاهده می‌باشد. همانطور که مشاهده می‌کنید سعی شده است تا چرخ طیار تا جای ممکن به مرکز ثقل نزدیک‌تر باشد تا سامانه پایداری بیشتری داشته باشد.



شکل ۱۲-۴ اضافه کردن چرخ طیار به مدل موتورسیکلت

در این مدل یک سنسور لایدار^۱ جهت تشخیص مانع نیز اضافه شده است. مشخصات لایدار در قسمت ویژگی‌ها در شکل ۱۳-۴ قابل مشاهده است. در حقیقت به کمک این سنسور می‌توان فاصله‌ی موانع بزرگ موجود بر سر راه موتور را دریافت و چنانچه موتورسیکلت به آن مانع نزدیک بود سرعت خود را کم کرده و فرمان را بچرخاند تا به آن برخورد نکند. البته این سنسور علاوه بر کاربرد تشخیص مانع کاربردهای دیگری

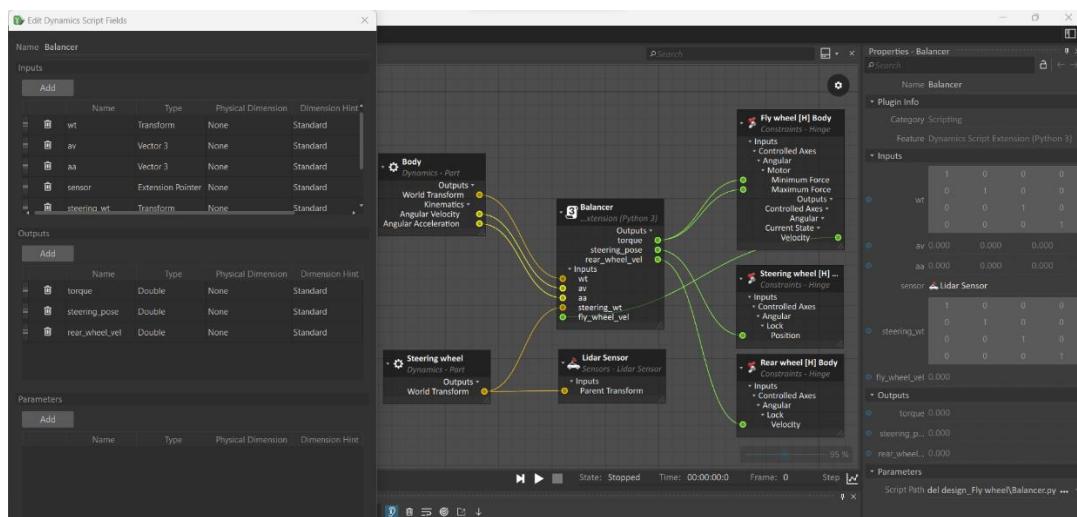
¹ Lidar

از قبیل کمک در کارهای مکانیابی، نقشه برداری و دید در تاریکی را می‌تواند نیز انجام دهد که در این پژوهش به این موارد نمی‌پردازیم.



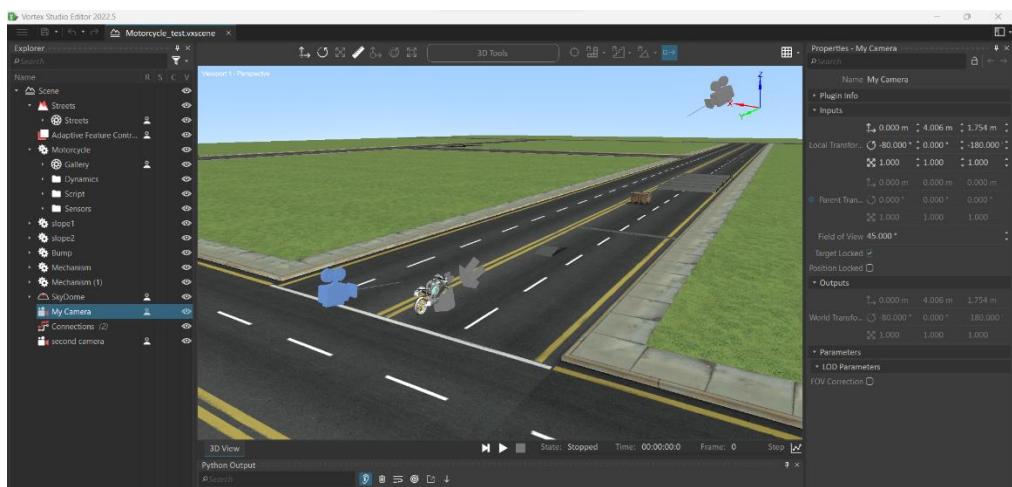
شکل ۱۳-۴ اضافه کردن سنسور لایدار به موتورسیکلت

همچنین لازم است تا یک کد پایتون جهت ایجاد کنترل PID اضافه شود. در اینجا از سه عدد کنترلر PD استفاده شده است. یکی برای زاویه، یکی برای سرعت زاویه‌ای و دیگری برای سرعت چرخش چرخ طیار. در نهایت خروجی حاصل از این سه کنترلر با یک دیگر جمع شده و به عنوان نیروی لازم برای ایجاد ممان در چرخ طیار به کار گرفته می‌شود. در شکل ۱۴-۴ ارتباطات و ورودی و خروجی‌ها مشخص شده است. همچنین در این کد پایتون لازم است توابعی برای یافتن فاصله مانع بزرگ تا موتورسیکلت به کمک لایدار و همچنین اتخاذ تصمیم درست براساس آن، تعریف نماییم.



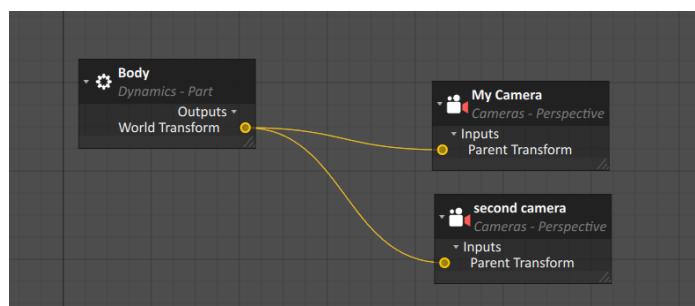
شکل ۱۴-۴ اضافه کردن چرخ طیار به مدل موتورسیکلت

از آنجایی که در این روش موفق به حفظ تعادل موتورسیکلت شدیم، لازم است یک گام پیش برویم و چک کنیم که آیا موتورسیکلت با عبور از روی موانع هم آیا تعادل خود را حفظ می‌کند یا خیر. برای این امر لازم است تا یک صحنه^۱ در نرم افزار ایجاد کنیم. سپس با توجه به فایل‌های آماده‌ای که در پوشش‌های نسبی نرم افزار وجود دارد، محیط را برای آزمایش آماده نماییم. در این بخش لازم است یک مانع مثلثی، یک دست انداز، یک مانع بزرگ برای تشخیص به کمک لایدار و تعدادی دست انداز متوالی پشت سرهم را تعریف نماییم. همچنین دو عدد دوربین برای گرفتن فیلم از موتورسیکلت در حال حرکت تعریف می‌نماییم. مدل نهایی در شکل ۱۵-۴ قابل مشاهده می‌باشد.



شکل ۱۵-۴ اضافه کردن مکانیزم ساخته شده به فایل صحنه

همچنین برای آنکه دوربین‌ها به همراه موتورسیکلت حرکت کنند لازم است تا ارتباطات موجود در شکل ۱۶-۴ را تعریف نماییم. همچنین باید تیک مربوط به قفل شدن روی هدف را هم فعال نماییم.



شکل ۱۶-۴ ارتباط بین دوربین‌ها و بدنه‌ی موتورسیکلت در صحنه

¹ Scene

این روش با موفقیت تمام موانع را پشت سر می‌گذارد. در قسمت بعدی سعی می‌کنیم تا روش کنترلی را تغییر دهیم اما جسم کنترلی چرخ طیار باقی می‌ماند و در نهایت در فصل بعد به مقایسه نتایج می‌پردازیم.

۴-۲- یادگیری تقویتی

باتوجه به اینکه طراحی کنترل برای موتورسیکلت کاری سخت می‌باشد و به دانش افراد متخصص در حوزه‌ی مکانیک و کنترل نیازمند است، امروزه از روش‌های مبتنی بر هوش مصنوعی استفاده‌های فراوانی می‌شود. یادگیری تقویتی روشی برای یادگیری کنترل بدون داشتن دانشی از مفاهیم پیچیده‌ی کنترلی و مکانیکی می‌باشد.

در ادامه با یک مثال ساده یادگیری تقویتی را توضیح می‌دهیم. یک کودک که در حال یادگیری راه رفتن می‌باشد بارها می‌افتد و بلند می‌شود و به مرور یاد می‌گیرد که چگونه راه برود و تعادل خود را حفظ نماید. هریار که می‌افتد، دردی که در اثر افتادن حس می‌کند، به عنوان تنبیه‌ی برای او تلقی می‌شود. هریار هم که موفق می‌شود مسیر بیشتری را برود افراد خانواده او را تشویق می‌کنند و او خوشحال می‌شود. از آنجایی که کودک به طور غریزی از درد فرار می‌کند و به دنبال تشویق است، تلاش می‌کند که به درستی راه برود و تعادل خود را حفظ نماید. به کمک یادگیری تقویتی می‌توان به ربات‌ها و سایر اجسام بی‌جان دارای پردازنده، یاد داد که چگونه یک کاری را انجام دهنند.

اولین مطالعه بر روی کنترل تعادل موتورسیکلت به کمک یادگیری تقویتی، استفاده از الگوریتم SARSA¹ می‌باشد. در این الگوریتم فرض شده است که فضای حالت² گستته می‌باشد و واکنش سامانه هم به صورت گستته اعمال می‌شود و درواقع به صورت عملی کاربردی نداشت چرا که نمی‌توانست فضای پیوسته را مدل کند.

¹ SARSA

² State Space

در مطالعات بعدی از الگوریتم تکرار سیاست کمترین مربعات^۱ استفاده شد که در آن فضای حالت به صورت پیوسته ولی فضای واکنش^۲ سامانه به صورت گسسته می‌باشد و درواقع خروجی‌های آن مقادیری گسسته می‌باشد. البته این کنترلر نسبت به نسخه‌ی قبلی عملکرد بهتری داشت ولی باز هم به علت خروجی‌های گسسته سازی شده، محدودیت داشت.

برای آنکه بتوانیم با یک فضای حالت و واکنش پیوسته کار کنیم، به یک الگوریتم یادگیری تقویتی نیاز داریم که بتواند فضای حالت و فضای واکنش با ابعاد بالا و پیوسته کار کند. امروزه با توجه به نوآوری‌ها در زمینه‌ی سیستم‌های کامپیوتری، محققان روی نوعی از الگوریتم یادگیری تقویتی به نام یادگیری تقویتی عمیق^۳ متمرکز شده‌اند که در آن از سیاست شبکه‌های عصبی عمیق استفاده می‌نمایند. یکی از الگوریتم‌های یادگیری تقویتی عمیق، گرادیان سیاست قطعی عمیق (DDPG)^۴ می‌باشد که می‌تواند با فضای واکنش پیوسته با ابعاد بالا کار کند و همچنین از شبکه‌های عصبی عمیق برای بیان سیاست‌ها استفاده کند. علاوه بر آن، این الگوریتم رویکردهای سنتی یادگیری تقویتی مانند گرادیان سیاست و بازیگر-منتقد^۵ را به ارت برده است.

در این پژوهش قصد داریم تا الگوریتم DDPG را برای حفظ تعادل موتورسیکلت اعمال کنیم و درحقیقت این الگوریتم به عنوان یک کنترلر برای ما عمل خواهد کرد. با توجه به آنکه در قسمت قبل، چرخ طیار به عنوان یک جسم کنشگر برای جبران ممان ایجاد شده در اثر انحراف موتورسیکلت، عملکرد خوبی داشت؛ در این قسمت نیز چرخ طیار را به عنوان جسم کنشگر در نظر می‌گیریم. در ادامه به صورت دقیق‌تری به توضیح یادگیری تقویتی پرداخته و سپس به پیاده‌سازی در نرم افزار می‌پردازیم.

¹ Least-Squares Policy Iteration

² Action Space

³ Deep RL

⁴ Deep Deterministic Policy Gradient

⁵ Actor-Critic

۴-۱-۲- یادگیری تقویتی و فرآیند تصمیم‌گیری مارکوو^۱ (MDP)

یادگیری تقویتی بخشی از یادگیری ماشین می‌باشد که بر روی تعامل با محیط، دریافت پاداش از محیط و یادگیری از طریق پاداش دریافت شده، متمرکز می‌باشد. در اغلب موقع مسائل یادگیری تقویتی به صورت مسئله‌ی فرآیند تصمیم‌گیری مارکوو گستته در زمان و تعریف متغیرهای $\langle S, A, P, r, \gamma \rangle$ مدل می‌شود. متغیر S بیانگر فضای حالت، متغیر A بیانگر فضای عمل، نماد P بیانگرتابع انتقال $(P(s_{t+1}|s_t, a_t))$ که مقدار احتمال رفتن به حالت بعدی s_{t+1} با توجه به حالت فعلی و عمل تصمیم‌گیری شده (s_t, a_t) ، تابع $r(s_t, a_t)$ بیانگر مقدار پاداش دریافت شده براساس پاداش به دست آمده در حالت و عمل فعلی و در آخر متغیر γ که عددی حقیقی بین ۰ تا ۱ هست، بیانگر نرخ کاهش؛ می‌باشند. یک ترتیب از جفت حالت-عمل‌ها، یک خط سیر ξ که به اپیزود معروف است را می‌سازد. همچنین مجموع پاداش کاهیده شده‌ی آن اپیزود از رابطه‌ی (۳-۴) به دست می‌آید.

$$R(\xi) = \sum_{t=0}^T \gamma^t r(s_t, a_t) \quad (3-4)$$

یک الگوریتم یادگیری تقویتی سعی می‌کند تا یک سیاست بهینه به نام π^* را بیابد، به نحوی که امید ریاضی مجموع پاداش کاهیده شده‌ی یک اپیزود که در رابطه‌ی (۳-۴) آمده است را، بیشینه نماید.

$$J(\pi) = E[R(\xi)] = \int p(\xi|\pi) R(\xi) d\xi \quad (3-4)$$

روش‌های مبتنی بر گرادیان سیاست، یکی از رویکردهای یافتن سیاست بهینه به شمار می‌آیند. در این روش‌ها، سیاست با بردار پارامتر θ ، پارامتری می‌شود و در امتداد جهت گرادیان مجموع پاداش کاهیده شده‌ی مورد انتظار (گرادیان $E[R(\xi)|\pi]$) باتوجه به رابطه‌ی (۴-۴) آپدیت می‌شود.

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi(\theta_k)) \quad (4-4)$$

در رابطه‌ی (۴-۵)، ضریب α ، نرخ یادگیری و k شماره‌ی آپیت فعلی می‌باشد. روش‌های مبتنی بر گرادیان سیاست، همگرا شدن به یک نقطه‌ی بهینه و یا حداقل یک بهینه‌ی محلی را تضمین می‌کنند. قسمت اصلی محاسبات در روش گرادیان سیاست، آن است که چگونه با توجه به خط سیر داده‌شده، مقدار

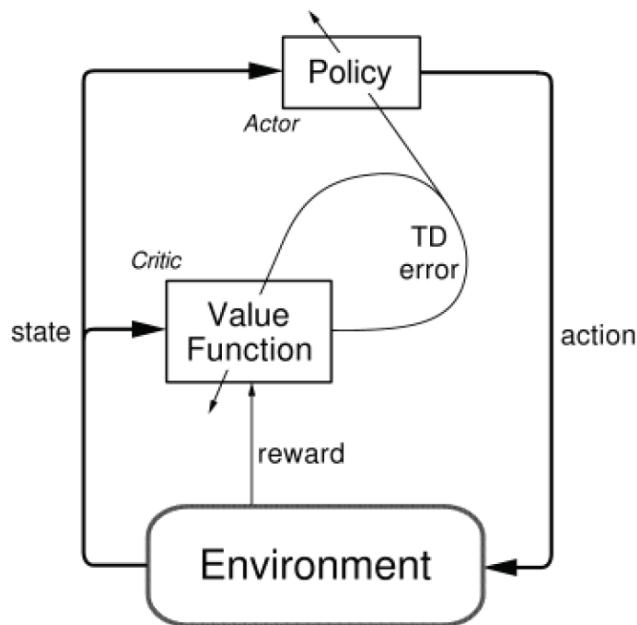
¹ Markov Decision Process

گرadiان را تخمین بزنیم به طوری که واریانس آن کوچک باشد. بر همین اساس روش‌های گرadiان سیاست بسیاری مانند تفاضل محدود، تقویتی، بازیگر-منتقد و گرadiان سیاسی طبیعی وجود دارند[۱۵].

۴-۲-۴- معماری بازیگر-منتقد

این معماری به عنوان مدلی برای توسعه‌ی الگوریتم‌ها به کارگرفته شده است. این معماری از دو بخش تشکیل شده است: بخش بازیگر و بخش منتقد.

بخش بازیگر، یک حالت را از محیط دریافت می‌کند و یک سیاست را برای پیش‌بینی عمل استفاده می‌کند. بخش منتقد، همان حالت به همراه پاداش دریافت شده از محیط را دریافت می‌کند و ارزیابی می‌کند که چه قدر در آن حالت خوب عمل کرده است. سیاست بخش بازیگر براساس ارزیابی بخش منتقد در فرآیند یادگیری، آپدیت می‌شود. معماری بازیگر-منتقد به خوبی در شکل ۱۷-۴ نمایش داده شده است.



شکل ۱۷-۴ ارتباط بین دوربین‌ها و بدنی موتورسیکلت در صحنه

۴-۲-۳- الگوریتم گرadiان سیاست قطعی عمیق

الگوریتم DDPG یک الگوریتم به روز در حوزه‌ی الگوریتم‌های یادگیری تقویتی عمیق می‌باشد. این الگوریتم مشخصات روش گرadiان سیاست و روش بازیگر-منتقد را به ارث برده است. در حقیقت از دو

شبکه‌ی Q عمیق^۱ برای بیان سیاست‌ها استفاده می‌کند؛ یکی برای بخش بازیگر و دیگری برای بخش منتقد می‌باشد. بنابراین سیاست‌ها می‌توانند مسائل با پیچیدگی بالا را نیز بازنمایی نمایند.

یکی از مزیت‌های بزرگ DDPG آن است که می‌تواند مشکل فضای حالت و عمل پیوسته و با ابعاد بالا را حل کند که موضوعی مهم برای مسائل کنترلی و رباتیک می‌باشد. برای داشتن این ویژگی، سیاست بخش بازیگر که از یک شبکه عصبی عمیق تشکیل شده است، یک حالت را از محیط دریافت می‌کند و یک واکنش که مقدار پیوسته‌ای دارد را به عنوان خروجی تولید می‌کند. پارامترهای شبکه در سیاست بازیگر براساس روش گرادیان سیاست قطعی آپدیت می‌شوند، که خود روشی جدید در گرادیان سیاست می‌باشد.

۴-۲-۴- حفظ تعادل موتورسیکلت با یادگیری تقویتی

همانطور که پیشتر اشاره شد، تعادل موتورسیکلت مورد پژوهش، به کمک چرخ طیار حفظ خواهد شد. عامل هوشمند، اطلاعاتی را که از محیط به عنوان مشاهدات خود دریافت می‌کند عبارتند از:

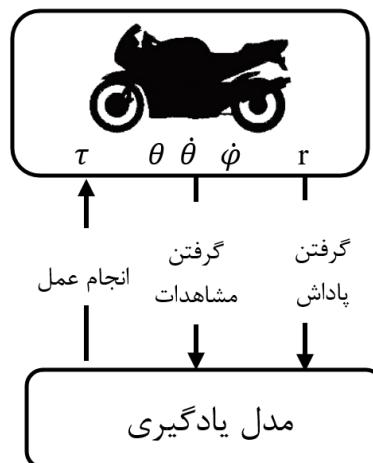
- سینوس و کسینوس زاویه‌ی انحراف موتورسیکلت از حالت تعادل (θ)
- سرعت زاویه‌ای انحراف موتورسیکلت از حالت تعادل ($\dot{\theta}$)
- سرعت زاویه‌ای چرخ طیار ($\dot{\phi}$)

باتوجه به پارامترهای مورد بررسی در طراحی کنترلر PID در بخش قبل، در این بخش نیز با کمی سعی و خطابه موارد فوق برای پارامترهای ورودی و یا درواقع مشاهدات عامل هوشمندمان رسیدیم. در هر واحد زمانی که برابر با ۰.۰۱۵ ثانیه می‌باشد، عامل اطلاعات ذکر شده را در باره‌ی حالت فعلی خود دریافت می‌کند و سپس یک خروجی پیوسته در بازه‌ی ۱-۱ تا ۱ تولید می‌کند که در مقدار ۵۰۰ ضرب شده و در حقیقت ممان لازم در چرخ طیار برحسب نیوتن متر، خواهد بود.

همچنین لازم به ذکر است که فرآیند یادگیری را برای حالتی که موتورسیکلت ساکن هست انجام می‌دهیم ولی همین مدل را برای موتورسیکلت در سرعت پایین تست خواهیم کرد. در حالت تست سرعت چرخ عقب موتورسیکلت بر روی مقدار ۴۰۰ درجه برثانیه تنظیم می‌شود.

^۱ Deep-Q Network

در شکل ۱۸-۴ موتورسیکلت در نقش محیط می‌باشد و که حالت‌ها را در اختیار الگوریتم یادگیری عمیق قرار می‌دهد و با توجه به مقدار خروجی تولید شده توسط الگوریتم، واکنش نشان می‌دهد.



شکل ۱۸-۴ نحوه ارتباط محیط و موتورسیکلت

نمایش کلی الگوریتم DDPG در شکل ۱۹-۴ آمده است. مطابق با این عکس شبکه‌ی مربوط به بازیگر یک شبکه‌ی عصبی با پارامترهای θ^{μ} و θ^{π} می‌باشد. همچنین منتقد نیز یک شبکه عصبی دیگر با پارامترهای θ^Q و θ'^Q است. مراحل الگوریتم DDPG در ادامه آمده است:

- ۱ - موتورسیکلت حالت s_t را مشاهده می‌کند و به شبکه‌ی بازیگر منتقل می‌کند.
- ۲ - شبکه‌ی بازیگر حالت s_t را به عنوان ورودی دریافت می‌کند و عمل a_t را به عنوان خروجی آمده می‌کند. سپس یک نویز کوچک به این عمل اضافه می‌نماید و به موتورسیکلت برمی‌گردد.

$$a_t = \mu(s_t | \theta_\mu + N_t) \quad (5-4)$$

- ۳ - موتورسیکلت پاداش r_t و حالت بعدی یعنی s_{t+1} را مشاهده می‌کند. همچنین مقادیر $\langle s_t, a_t, r_t, s_{t+1} \rangle$ در مخزن تجربیات^۱ ذخیره می‌شود.
- ۴ - در مخزن تجربیات، ما به صورت رندوم یک دسته‌ی N تایی را انتخاب می‌کنیم و از آن‌ها برای یادگیری سیاست استفاده می‌نماییم.

¹ Experience Pool

۵- تابع زیان^۱ را با توجه به رابطه‌ی (۶-۴) محاسبه می‌کنیم:

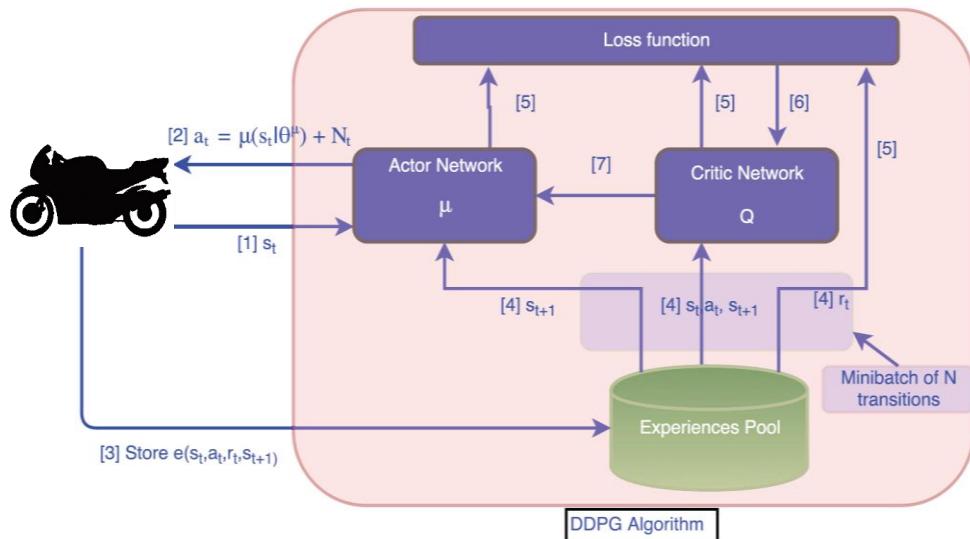
$$L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i | \theta^Q))^2 \quad (6-4)$$

$$y_i = r_i + \gamma Q'(s_{t+1}, \mu'(s_{t+1} | \theta^\mu) | \theta^Q') \quad (7-4)$$

۶- شبکه‌ی منتقد را با کمینه کردن تابع زیان یعنی L آپدیت می‌کنیم.

۷- شبکه‌ی بازیگر با توجه به تئوری گرادیان سیاست قطعی آپدیت می‌شود.

$$\nabla_{\theta^\mu} \mu|_{s_i} \approx \frac{1}{N} \sum_i \nabla_a Q(s, a | \theta^Q) |_{s=s_i, a=\mu(s_i)} \times \nabla_{\theta^\mu} \mu(s | \theta^\mu) |_{s_i} \quad (8-4)$$



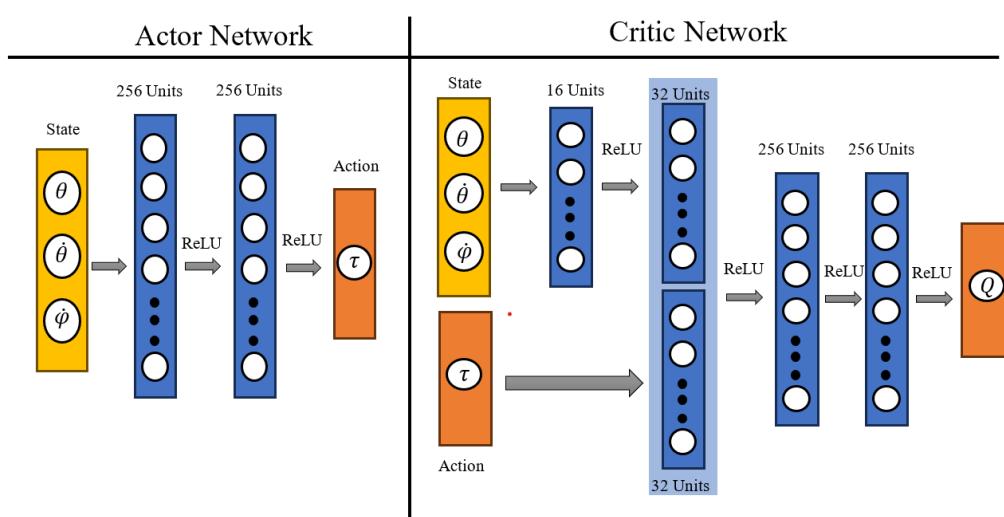
شکل ۴-۱۹- فرآیند انجام شده در الگوریتم DDPG

همانطور که اشاره شد برای بخش‌های بازیگر و منتقد از شبکه‌ی عصبی استفاده شده است. معماری این شبکه‌ها و پارامترهای آن‌ها حائز اهمیت می‌باشند. به صورت کلی راه حل خاصی برای تعیین تعداد لایه‌ها و تعداد نورون‌های هر لایه و همچنین تعیین هایپرپارامترها وجود ندارد او به صورت تجربی می‌توان تاحدودی درک کرد که چه شبکه‌ای برای چه مسئله‌ای مناسب است و سپس با سعی و خطا و درک مناسب از کاربرد هریک از پارامترها مقدار آن‌ها را تعیین کرد.

^۱ Loss function

برای شبکه‌ی مربوط به بازیگر لازم است ابتدا یک لایه‌ی ورودی داریم. ورودی‌های شبکه که حالت فعلی و لیست مشاهدات آن می‌باشد را دریافت می‌کنیم. سپس از دو لایه‌ی متراکم و کاملاً متصل استفاده می‌کنیم که تابع فعال‌ساز آن‌ها نیز ReLU می‌باشد. در پایان هم یک لایه‌ی متراکم با تعداد نورون‌هایی به اندازه تعداد خروجی‌ها که در اینجا یکی است ایجاد می‌کنیم. تابع فعال‌ساز آن tanh می‌باشد[۱۶].

برای شبکه‌ی مربوط به منتقد لازم است هم حالت و هم عمل انجام شده را به عنوان ورودی بگیریم. حالات‌ها ابتدا باید سه لایه یعنی یک لایه ورودی، یک لایه‌ی متراکم ۱۶ نورونه و یک لایه‌ی متراکم ۳۲ نورونه با تابع فعال‌ساز ReLU را بگذرانند و عمل باید پس از گذر از لایه‌ی ورودی به یک لایه‌ی متراکم ۳۲ نورونه وارد شود. لایه‌ی ۳۲ نورونه‌ی هردو را به یک دیگر متصل می‌نماییم و سپس خروجی از دو لایه‌ی ۲۵۶ نورونه‌ی متراکم با تابع فعالیت ReLU می‌گذرد تا به لایه‌ی خروجی برسد که شامل یک نورون برای تولید گشتاور می‌باشد. این معماری در شکل ۲۰-۴ قابل مشاهده می‌باشد.



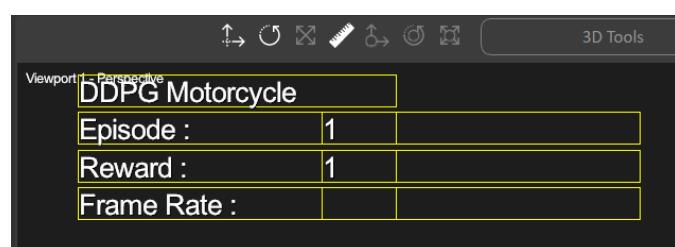
شکل ۲۰-۴ معماری شبکه‌های عصبی بازیگر و منتقد

۴-۲-۵- پیاده سازی در نرم افزار Vortex Studio

برای الگوریتم یادگیری تقویتی لازم است تا برنامه‌ای بنویسیم که شبیه‌ساز را چندین بار اجرا کند و فرآیند یادگیری انجام شود. در نرم افزار Vortex Studio قابلیتی فراهم شده است که می‌توان یک اپلیکیشن را طراحی کرد و یک واسط VHL در آن تعریف کرد که بتوان به پارامترهای مورد نیاز از شبیه‌ساز و موتورسیکلت دسترسی داشت. روال ساخت مدل موتورسیکلت و چرخ طیار مانند قبل می‌باشد.

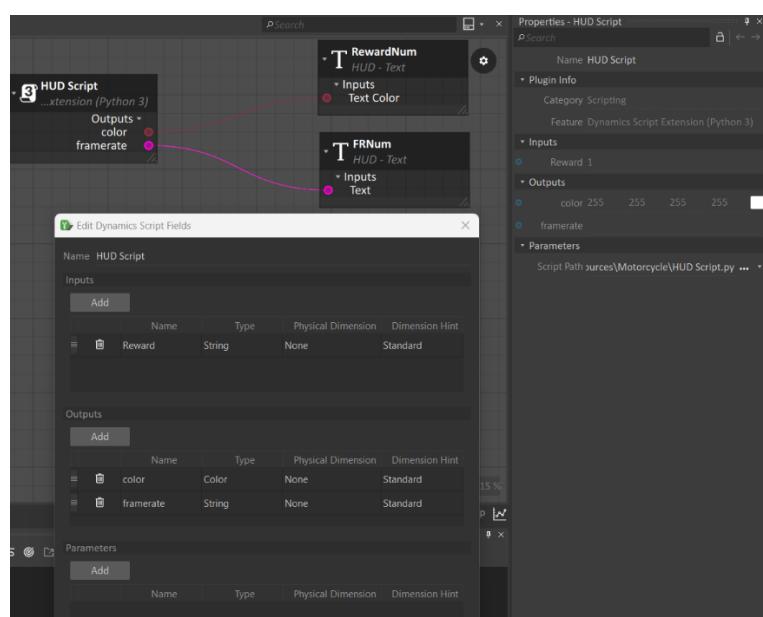
برای قسمت گرافیکی لازم است از قسمت چراغ مربوط به جعبه ابزار، نورپردازی محیط را انتخاب نماییم و محل آن را نیز در وسط موتورسیکلت قرار دهیم. همچنین یک دوربین هم در روپروری موتورسیکلت قرار دهیم تا هنگام آموزش که موتورسیکلت ساکن است نمای آن را از روپرور داشته باشیم تا انحرافات آن را به خوبی متوجه شویم.

لازم است تا بخشی برای نمایش شماره اپیزود و نمایش پاداش کسب شده و همچنین نرخ نمایش فریم داشته باشیم. برای این منظور می‌توان به بخش **HUD** از جعبه ابزار مراجعه کرد و به کمک ابزار متن، موارد گفته شده را ایجاد کرد. در شکل ۲۱-۴ **HUD** ایجاد شده نمایش داده شده است.



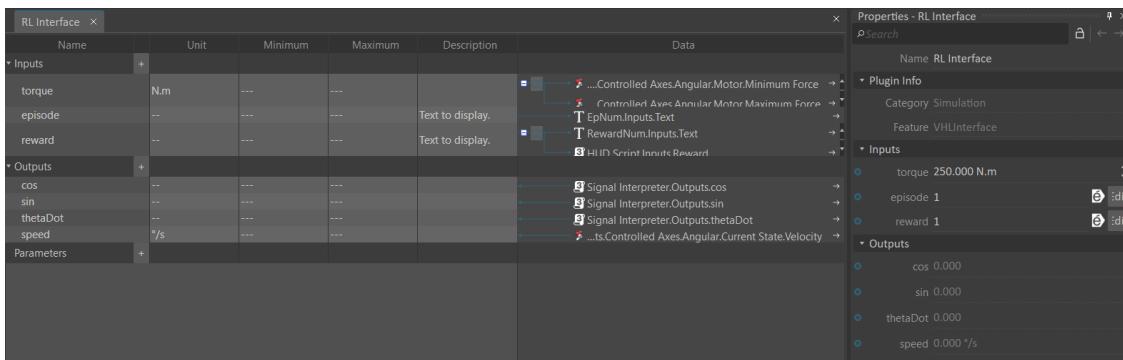
شکل ۲۱-۴ بخش **HUD** ساخته شده در نرم افزار

همچنین لازم است کدی نوشته شود که هم نرخ نمایش فریم‌ها و هم رنگ مربوط به پاداش دریافت شده با توجه به مثبت یا منفی بودن را محاسبه نماید. سپس باید یک قسمت برای ارتباطات فراهم شود و ارتباطات این کد و قسمت‌های موردنظر از **HUD** برقرار شود.



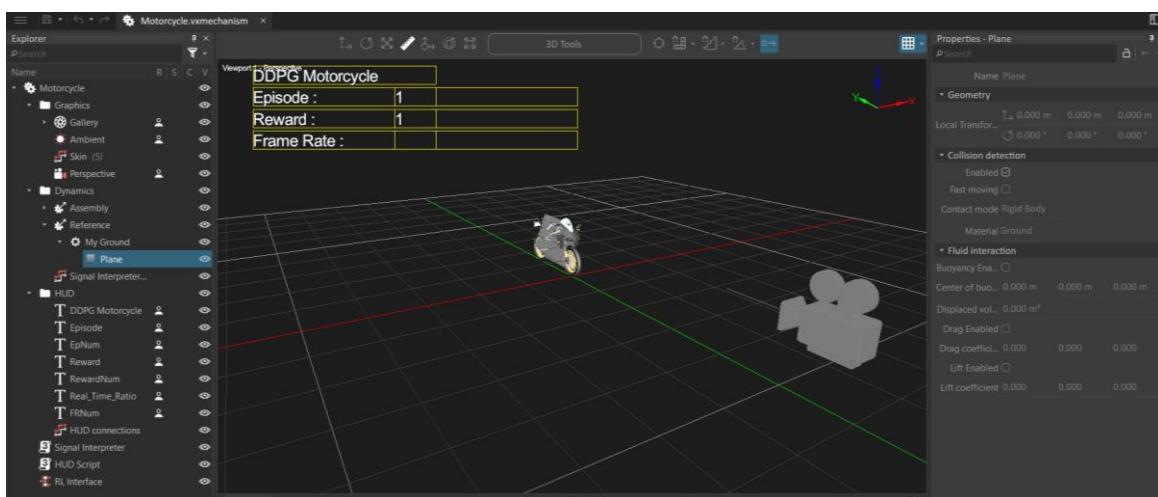
شکل ۲۲-۴ ارتباطات بین کد و **HUD**

از جمله مهمترین تغییرات ایجاد یک رابط پارامترهای لازم از شبیه سازی را به الگوریتم یادگیری ماشین می‌دهد و در حقیقت به ما این اجازه را می‌دهد که کد پایتون اجرا کننده اپلیکیشن به مقادیر مورد نظر دسترسی داشته باشد. رابط ساخته شده را می‌توانید در شکل ۲۳-۴ مشاهده کنید.



شکل ۲۳-۴ رابط ساخته شده برای ارتباط کد و محیط شبیه ساز

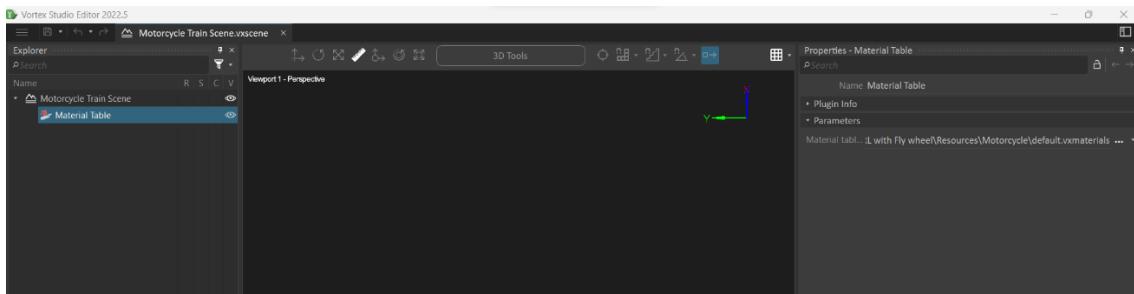
لازم است یک مونتاژ جدید به مکانیزم اضافه شود. از آنجایی که دیگر موتورسیکلت در محیط شبیه ساز اجرا نمی‌شود و خودمان یک اپلیکیشن جداگانه می‌سازیم، ضروری است که زمین را نیز خودمان تعریف نماییم. برای این کار باید یک صفحه تعریف نماییم و هندسه‌ی مورد نظر را هم به آن بدهیم. اگر این بخش را به درستی انجام ندهیم هنگام اجرای شبیه سازی، موتورسیکلت سقوط خواهد کرد. در نهایت مدل تعریف شده در شکل ۲۴-۴ نمایش داده شده است.



شکل ۲۴-۴ مدل ساخته شده در نرم افزار برای یادگیری تقویتی

سپس باید برای آن یک صحنه ایجاد کنیم. در این صحنه فقط لازم است که جدول مواد را اضافه نماییم. چرا که در اپلیکیشن ساخته شده لازم است که جنس قسمت‌های مختلف مشخص باشد و بدون ایجاد این

جدول تماس قسمت‌ها با یک دیگر و جنس آن‌ها مشخص نخواهد بود و رفتار غیرمنتظره‌ای را می‌توان از اپلیکیشن انتظار داشت.



شکل ۴-۲۵ صحنه‌ی تعریف شده برای آموزش موتورسیکلت

در نهایت با کمک از کد نوشته شده برای پاندول معکوس توسط شرکت، می‌توان مکانیزم و صحنه ساخته شده را در کد اضافه کرد و به کمک واسطه‌ی که تعریف کردیم به پارامترهای مورد نظر دسترسی داشته باشیم. در نهایت کدی برای آموزش مدل نوشته‌ایم که در حدود ۱۰۰۰ بار این شبیه‌ساز را اجرا کرده و فرآیند آموزش را انجام می‌دهد. سپس با افزایش پاداش کسب شده، کد باید مدل آموزش دیده را ذخیره نماید.

سپس یک صحنه مانند آنچه که برای کنترلر PID تعریف کردیم، مجدداً تعریف می‌نماییم و موتورسیکلت را در آن صحنه به کمک مدل آموزش دیده آزمایش می‌کنیم. خواهیم دید که مدل آموزش دیده به خوبی می‌تواند به عنوان کنترلر عمل نماید و تعادل موتورسیکلت را حفظ نماید.

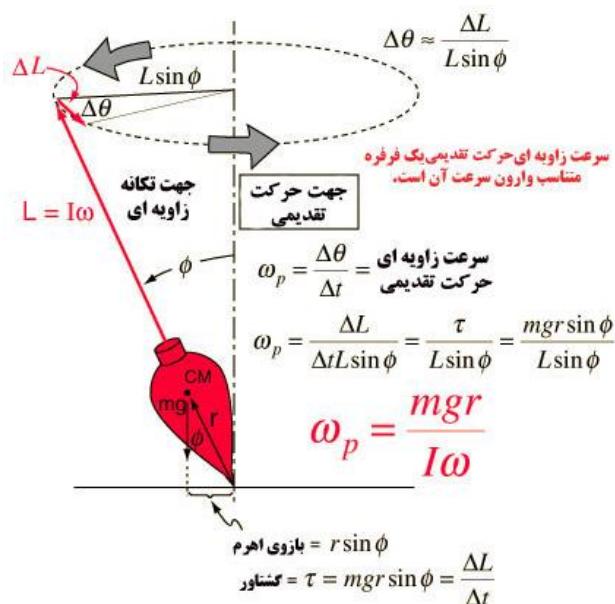
۴-۳-۴- ژیروسکوپ

در قسمت‌های قبل نیاز داشتیم که کدنویسی کنیم و یک بلوکی تحت عنوان کنترلر را طراحی نماییم. نحوه‌ی عملکرد به این صورت بود که پارامترهایی نظیر زاویه‌ی انحراف و سرعت زاویه‌ای و ... را به عنوان ورودی دریافت می‌شد و با محاسبه‌ی خطای توانستیم واکنش مناسب را به عنوان خروجی بدھیم. در این بخش سعی می‌کنیم که با کمک ژیروسکوپ و خاصیت ژیروسکوپی تعادل موتور سیکلت را حفظ نماییم. بنابراین نیازی به کد نویسی نداریم و باتوجه به قوانین فیزیکی می‌خواهیم به صورت خودکار تعادل موتورسیکلت نگه داشته شود. در ادامه به صورت مختصر به پدیده‌ی ژیروسکوپی پرداخته و در نهایت به پیاده‌سازی آن در نرم‌افزار می‌پردازیم.

۴-۳-۱- اثر ژیروسکوپی

اثر ژیروسکوپی که به پایداری ژیروسکوپی و حرکت ژیروسکوپی هم معروف است، یک پدیده است که در اجسام در حال چرخش مشاهده می شود به خصوص آن دست از اجسامی که مومنتوم زاویه‌ای بالایی دارند. در حقیقت یک قانون اساسی در فیزیک است که سیستم‌های مکانیکی مختلفی مانند ژیروکوپ‌ها، دوچرخه‌ها و فضاییمهای در حال چرخش دیده می شود.

اثر ژیروسکوپی، تمايل یک جسم چرخان برای ثابت نگه داشتن جهت محور دوران خود می باشد. در حقیقت، اصل بقای مومنتوم زاویه‌ای پشتونه‌ی اصلی این پدیده، می باشد. مومنتوم زاویه‌ای یک جسم، کمیتی برداری است که به جرم، هندسه و سرعت زاویه‌ای جسم بستگی دارد. براساس این اصل، وقتی یک جسم در حال دوران داریم؛ در برابر تغییرات راستای محور دوران خود مقاومت می کند. برای مثال اگر یک فرفه را در نظر بگیرید، وقتی آن را می چرخانید، سعی میکند تعادل خود را حفظ نماید و با آن که نیروی گرانش سعی می کند آن را بر روی زمین بیندازد ولی گویی گشتاوری ایجاد می شود که با این انحراف در محور چرخش فرفه مخالفت می کند و در نهایت با کمی انحراف از حالت ایستاده علاوه بر حرکت حول محور چرخش، به حرکت تقدیمی حول محور قائم می پردازد و پایدار می ماند. این پدیده به علت اثر ژیروسکوپی می باشد و چنانچه فرفه را بدون چرخاندن از حالت ایستاده رها کنید بلا فاصله بر روی زمین می افتد.

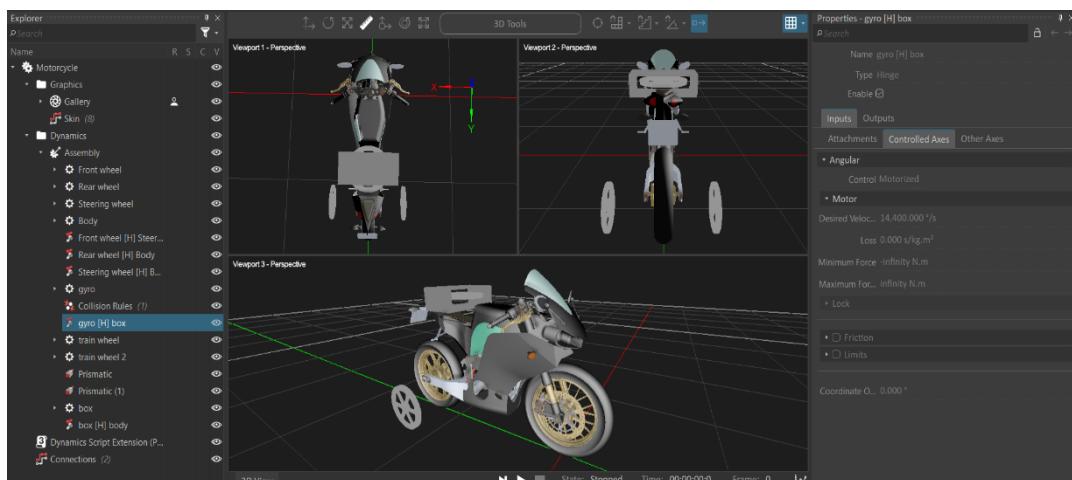


شکل ۴-۲۶ اثر ژیروسکوپی بر روی فرفه

۴-۳-۲- پیاده سازی در نرم افزار Vortex Studio

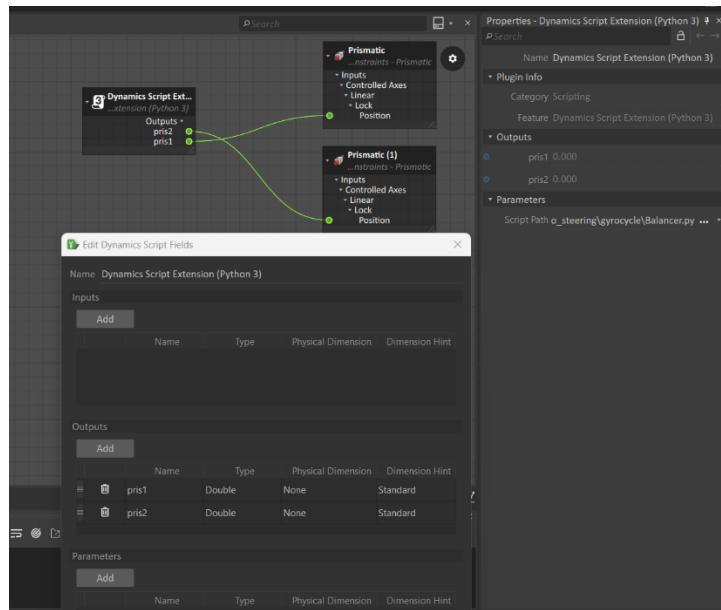
جهت پیاده سازی در نرم افزار Vortex Studio لازم است که ابتدا در قسمت گالری گرافیکی اضافه نماییم. با توجه به مدل سه بعدی چرخ طیار که از قبل داشتیم لازم است ۳ مرتبه آن را اضافه کنیم. یکی به عنوان دیسک ژیروسکوپ برای ما عمل می کند. دو چرخ دیگر به عنوان چرخ های کمکی برای حفظ تعادل موتورسیکلت در هنگام شروع چرخش دیسک عمل می کنند و کمک می کنند تا تعادل موتورسیکلت تا رسیدن دیسک ژیروسکوپ به سرعت موردنظر حفظ شود. همچنین لازم است یک واسطه که شبیه یک کانال مستطیلی کوچک می باشد، طراحی نماییم. برای این منظور در نرم افزار کتیا طراحی آن را انجام می دهیم و سپس خروجی را به سایتها می توان به راحتی به فایل vae تبدیل نمود. این جسم به عنوان محفظه‌ی نگه دارنده دیسک ژیروسکوپ عمل می نماید که خود نیز توسط یک لولا به بدنی موتورسیکلت متصل است و می تواند دوران pitch داشته باشد [۴].

سپس لازم است جسم صلب را در فایل مونتاژ ایجاد کرده و در نهایت هندسه و جنس مناسب را برای آنها تعیین نماییم. سپس به سراغ تعیین قیدهای مفصلی مناسب می رویم. یک قید لولا که به صورت موتوردار می باشد برای چرخاندن دیسک نسبت به کانال نیاز داریم. سرعت چرخش را ۱۴۰۰۰ درجه بر ثانیه که حدودا معادل ۲۳۰۰ دور بر دقیقه می باشد، تنظیم می نماییم. یک قید لولا که حرکت آزادله دارد، بین کلنال و بدنه تعريف می نماییم. دو قید پریزماتیک بین چرخها و بدنه تعريف می نماییم که پس از رسیدن ژیروسکوپ به سرعت مناسب، چرخ های کمکی را به سمت بالا به حرکت در می آورد و تعادل موتورسیکلت به کمک ژیروسکوپ حفظ می شود. مدل نهایی در شکل ۲۷-۴ قابل مشاهده است.



شکل ۲۷-۴ حفظ تعادل موتورسیکلت به کمک ژیروسکوپ در نرم افزار

همچنین لازم است یک کد پایتون اضافه شود که برای مثال بعد از حدود ۱۰۰ واحد زمانی که از شروع شبیه سازی گذشت، چرخ های کمکی را تا ارتفاع مناسبی بالا ببرد و همچنین سرعتی حدود ۴۰۰ درجه بر ثانیه نیز به چرخ عقب موتورسیکلت اعمال نماید. اتصالات مربوط به کد با بخش های مختلف و همچنین فیلدهای ورودی و خروجی کد در شکل ۲۸-۴ قابل مشاهده است.



شکل ۲۸-۴ اتصالات و تعیین ورودی ها و خروجی های کد برای مدل ژیروسکوپ

در نهایت یک صحنه مانند قسمت های قبل آماده می کنیم و مکانیزم ساخته شده را در آن اضافه می نماییم و نحوه عملکرد موتورسیکلت را آزمایش می کنیم. خواهیم دید که تا حد خوبی تعادل موتورسیکلت حفظ می شود ولی متاسفانه با گذشت مدت زمانی موتورسیکلت تعادل خود را از دست می دهد. شاید به نظر برسد که اگر سرعت یا جرم دیسک چرخان را بیشتر نماییم باعث می شود که موتورسیکلت پایداری بهتری داشته باشد ولی در نرم افزار با زیاد کردن هریک از این مقادیر نسبت به مقادیر گفته شده، شاهد ناپایدار شدن موتورسیکلت و برهم خوردن سریع تر تعادل آن خواهیم بود؛ این امر هم به دلیل محدودیت های سبیه سازی می باشد و هم آنکه سیستم به نویز حساس شده و گشتاور زیادی برای حفظ تعادل تولید می کند که موجب ناپایداری می شود. در قسمت نتایج به صورت مفصل تری به نتایج به دست آمده می پردازیم.

فصل پنجم

تحلیل نتایج

تحلیل نتایج

همانطور که در قسمت قبل نشان داده شد، توانستیم تعادل موتورسیکلت را در نرم افزار Vortex Studio به سه روش برقرار کنیم. در دو روش اول جسم برقرار کننده‌ی تعادل چرخ طیار بود و در روش سوم، از ژیروسکوپ برای این امر استفاده کردیم.

در روش اول به دنبال آن بودیم که به کمک کنترلر PID که روشی سنتی‌تر می‌باشد، تعادل را حفظ نماییم. تعیین مقداره بهره‌های این کنترلر کاری زمان بر است و ممکن است نتوانیم به صورت کاملاً بهینه‌ای برسیم. از آنجایی که فقط یک کنترلر PID تنها، کافی نبود مجبور شدیم برای کنترل هر یک از متغیرهای تاثیرگذار مانند زاویه و سرعت زاویه‌ای مربوط به انحراف موتورسیکلت و همچنین سرعت زاویه‌ای چرخ طیار، یک کنترلر تعریف نماییم. این موضوع بر سختی فرآیند تنظیم بهره‌ها نیز افروز.

در روش دوم سعی کردیم از یادگیری تقویتی که از روش‌های نوین یادگیری در هوش مصنوعی می‌باشد، برای حفظ تعادل استفاده نماییم. در این مسئله دیگر نیازی به تعیین بهره‌ها و شناسایی متغیر تاثیرگذار، نداریم. البته هایپرپارامترهایی در شبکه باید تعیین شوند که با کمی سعی و خطاب قابل تنظیم هستند و فرآیند آماده سازی مدل به جای کنترلر قبلی آسان‌تر و سریع‌تر می‌باشد.

در روش سوم، به دنبال آن بودیم که دیگر از موتورسیکلت بازخورد نگیریم و قسمتی تحت عنوان کنترلر نداشته باشیم. برای این موضوع از وسیله‌ای به نام ژیروسکوپ و خاصیت فیزیکی مهم آن یعنی اثر ژیروسکوپی برای حفظ تعادل بهره بردیم. البته پس از مدتی و بعد از عبور از روی دو مانع اول تعادل موتورسیکلت از دست می‌رفت و به زمین می‌خورد ولی به دلیل محدودیت‌های شبیه ساز، نمی‌توانستیم سرعت ژیروسکوپ را بیشتر نماییم.

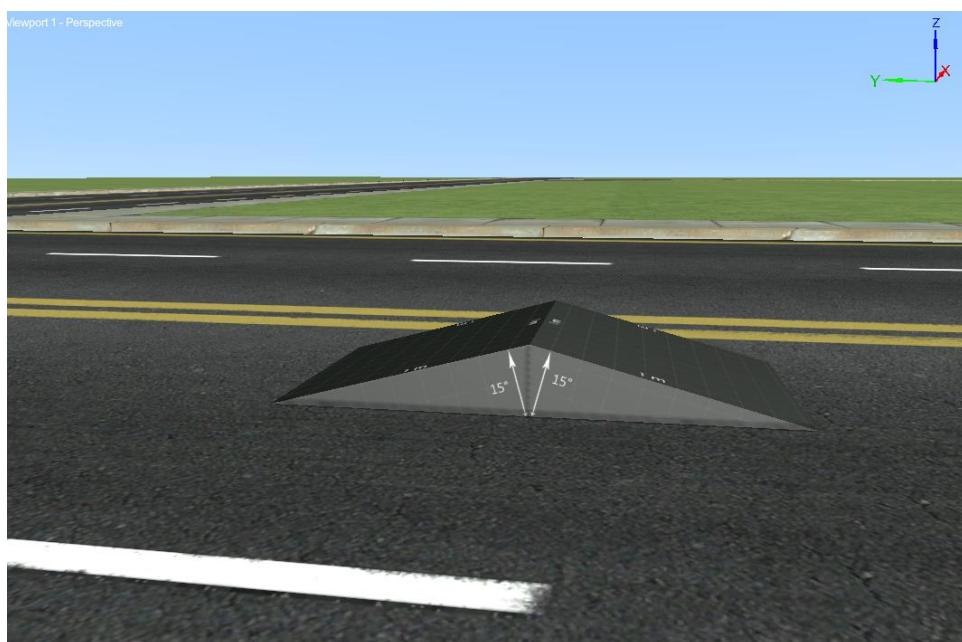
حال در ادامه به توضیح بیشتر در رابطه با عملکرد هریک از روش‌ها در عبور از روی موانع می‌پردازیم و همچنین به بررسی پارامترهایی نظیر مقدار زایه‌ی انحراف، سرعت زاویه‌ای انحراف و سرعت زاویه‌ای چرخ طیار در زمان شبیه‌سازی می‌پردازیم. البته در ابتدای کار لازم است با موانع محیطی تاثیرگذار در آزمایش آشنا شویم.

۱-۵- آشنایی با موانع مورد استفاده در محیط آزمایش

در این آزمایش از موانع مختلفی استفاده کردیم تا نحوهی عملکرد موتورسیکلت در مواجهه با هریک از آن‌ها بسنجدیم. در این بخش لازم است با این موانع و علت استفاده از هریک آشنا شویم تا در قسمت‌های بعد درک نسبتاً خوبی برای تحلیل داده‌های به دست آمده داشته باشیم.

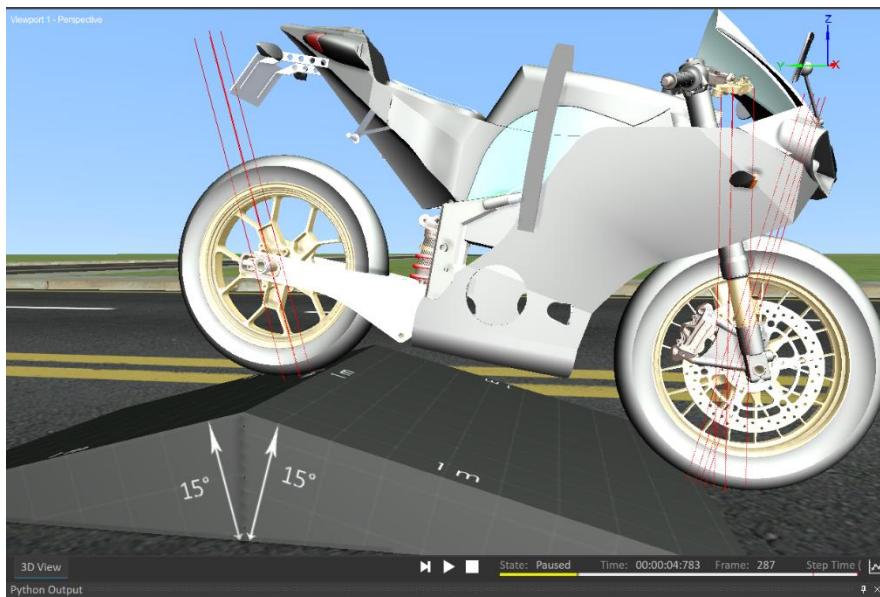
۱-۱-۵- سطح شیبدار

این مانع نمای جانبی مثلثی دارد و از اتصال دو سطح شیبدار با شیب یکسان ولی خلاف جهت تشکیل شده است. زاویه‌ی سطوح شیب دار ۱۵ درجه می‌باشد. موتورسیکلت باید از روی این مانع با حفظ تعادل عبور نماید. نمایی از این مانع را می‌توانید در شکل ۱-۵ مشاهده نمایید.



شکل ۱-۵ نمایی از مانع سطح شیبدار

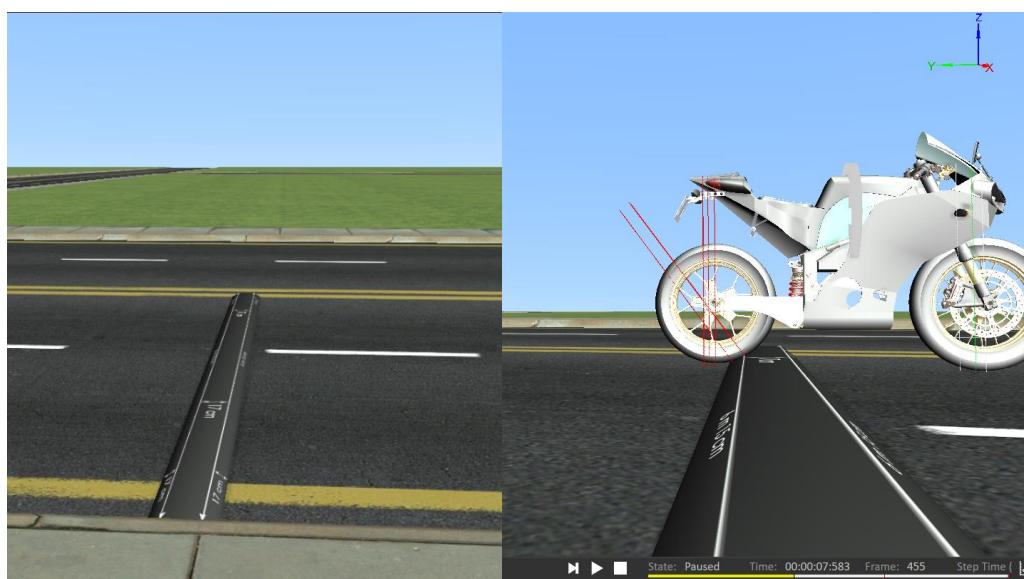
در این حالت سطح تماس لاستیک‌ها در هنگام رفتن از سطح زمین به سطح سیبدار یا بالعکس تغییر می‌کند و در نتیجه جهت نیروهای واردہ تغییر می‌نماید. در شکل ۲-۵ می‌توانید مثالی از نیروهای نرمال واردہ را مشاهده نمایید.



شکل ۲-۵ نیروهای نرمال واردہ بر لاستیک‌ها در عبور از روی سطح شیبدار

۲-۱-۵ دست انداز

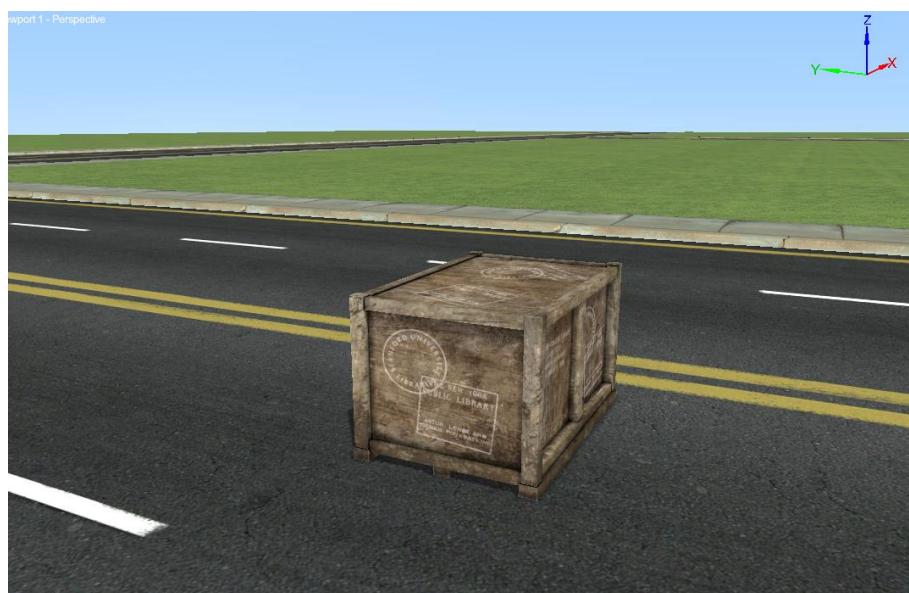
این مانع برای آن است که اثر وارد شدن نیرویی شبیه پله واحد را در لاستیک‌های جلو و عقب مشاهده نماییم. ارتفاع این دست انداز حدود ۱۰ سانتی‌متر می‌باشد. موتورسیکلت باید با حفظ تعادل از روی آن عبور نماید. نیروهای نرمال وارد شده به لاستیک‌ها نیز در این حالت متفاوت است. در شکل ۳-۵ مانع و نیروهای مذکور قابل مشاهده می‌باشند.



شکل ۳-۵ نمایی از مانع دست انداز و نیروهای نرمال واردہ به لاستیک‌ها

۳-۱-۵- جعبه‌ی بزرگ

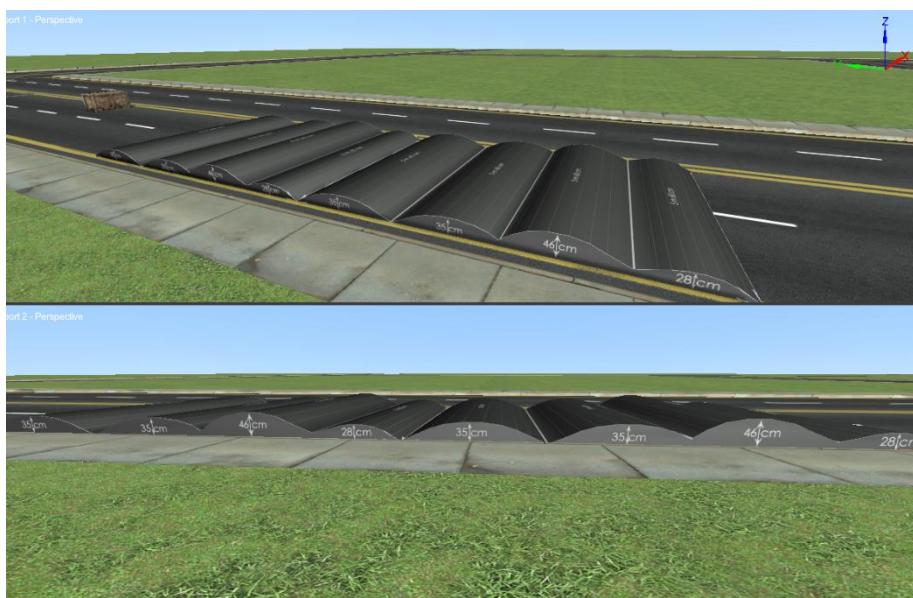
در میله‌ی مسیر موتورسیکلت به یک جعبه‌ی حدوداً ۱.۸ متری می‌رسد که برخلاف بقیه‌ی موانع نمی‌تواند از روی آن رد شود. در این مرحله باید موتورسیکلت مانع بزرگ را به کمک سنسور لایدار نصب شده در جلوی آن، تشخیص دهد و هرچه فاصله‌اش با مانع کمتر شود لازم است تا سرعت چرخ عقب نیز کاهیده شود و همچنین فرمان را به آرامی بپیچد تا مانع را رد نماید. برای انجام آن نیز از یک کنترلر PID استفاده نمودیم. در این بخش هدف آن است تا بینیم چرخاندن فرمان چه تاثیری بر روی تعادل موتورسیکلت دارد. در شکل ۴-۵ می‌توانید مانع را مشاهده نمایید.



شکل ۴-۵ نمایی از جعبه‌ی بزرگ به عنوان مانع

۴-۱-۵- دست اندازه‌های متواالی با ابعاد مختلف

در مرحله‌ی آخر موتورسیکلت از روی دست اندازه‌های متواالی که به یکدیگر چسبیده‌اند عبور می‌کند. در حقیقت به آن نیروی سینوسی با دامنه‌های مختلفی وارد می‌شود چراکه ارتفاع این دست اندازها متفاوت است. این مرحله به نظر می‌رسد سخت ترین گام می‌باشد و چنانچه موتورسیکلت این مرحله را نیز به موفقیت بگذراند، می‌توان گفت تعادل موتورسیکلت در عبور از مانع مختلف و نیروهای خارجی مختلف وارد شده تضمین می‌شود. در شکل ۵-۵ نمای جانبی و همچنین پریزماتیک این مانع قابل مشاهده می‌باشد.



شکل ۵-۵ نمایی از دستاندازهای متوالی

۲-۵- بررسی مدت زمان حفظ تعادل

در این قسمت به بررسی زمان متعادل بودن سامانه می‌پردازیم. زمان متعادل بودن بستگی به عوامل محیطی و آزمایشی نیز دارد. برای مثال باید این زمان در صورت وجود و همچنین عدم وجود موائع بررسی نماییم. لازم به ذکر است در تمامی روش‌ها، قبل از آزمایش، ابتدا برای آموزش و یا تنظیم بهره‌ها، موتورسیکلت را در حالت سکون قرار می‌دادیم و چرخ عقب حرکتی نداشت. سپس همان مدل را برای حالتی که موتورسیکلت در حال حرکت با سرعت کم بود، امتحان کردیم.

مدت زمان پایداری با PID: در بین روش‌های انجام گرفته، این روش از نظر زمان پایداری بیشترین زمان را در حالتی که موائع مختلف موجود باشد، داشت. درواقع موائع اول و دوم رو به خوبی پشت سر می‌گذارد و سپس در رسیدن به مانع سوم به خوبی فرمان را می‌پیچد سپس به مسیر قبلی خود ادامه می‌دهد. در نهایت گام مهمی که داریم دستاندازهای متوالی می‌باشد که تنها در این روش موتورسیکلت توانست این مانع را هم پشت سر بگذارد. در آخر هم مدتی مسیر را به راحتی ادامه می‌دهد بدون آنکه تعادل آن برهم بخورد. بنابراین این روش چه در وجود موائع و چه در غیبت آن‌ها بیشترین زمان پایداری را دارد.

مدت زمان پایداری در روش یادگیری عمیق: این روش نیز سه مانع اول را به خوبی پشت سر می‌گذارد ولی در مواجهه با مانع سوم تعادل خود را از دست می‌دهد. چنانچه مانع چهارم را حذف نماییم به راحتی پس از سه مانع اول، به مسیر خود ادامه می‌دهد و تعادل آن تضمین می‌شود. در این روش شاید نیاز باشد کمی مدل آموزش دیده را تغییر دهیم یا اینکه مدل را در هنگام حرکت موتورسیکلت آموزش دهیم تا بتواند برای همهی حالت‌ها آمادگی داشته باشد. یکی از کارهای مهم که می‌توان انجام داد؛ استفاده از یادگیری انتقالی^۱ می‌باشد. به این صورت که با توجه به اینکه مدل فعلی برای حالت سکون به خوبی آموزش دیده است؛ سپس آن را به عنوان وزن اولیه در فرآیند آموزش دیگری قرار دهیم و مجدداً موتورسیکلت را در حالت حرکت آموزش دهیم. در قسمت پیشنهادات بیشتر به آن می‌پردازیم. به هرجهت مدل فعلی در صورتی که مانعی در مسیر نباشد تعادل موتورسیکلت را تضمین می‌کند و مانند کنترلر PID عمل می‌کند. همچنین در عبور از مانعی همانند سه مانع اول موفق عمل می‌کند ولی به علت عدم موفقیت در مانع آخر می‌توان گفت در حضور موانع نسبت به کنترلر PID عملکرد بدتری دارد.

مدت زمان پایداری در حضور ژیروسکوپ: این روش ضعیفترین عملکرد را در مدت زمان پایداری چه در حضور موانع و چه بدون آن‌ها دارد. به این صورت است که در وجود موانع، مدت کمی پس از عبور از مانع دوم، تعادلش را از دست می‌دهد. در صورتی که مانعی هم در مسیر نباشد پس از مدتی مجدداً تعادل موتورسیکلت ازبین می‌رود. بنابراین در این روش تضمینی در حفظ تعادل وجود ندارد. البته ممکن است اگر تغییراتی در پیاده سازی و یا مکان ژیروسکوپ دهیم نتایج متفاوت شود؛ اما آنچه مهم است مقایسه‌ای است که در قسمت‌های بعدی انجام می‌شود و در مدت زمان پایداری به بررسی عملکرد می‌پردازیم. می‌توان در کارهای بعدی روی هریک از روش‌ها متمرکز شد و آن‌ها را بهبود داد.

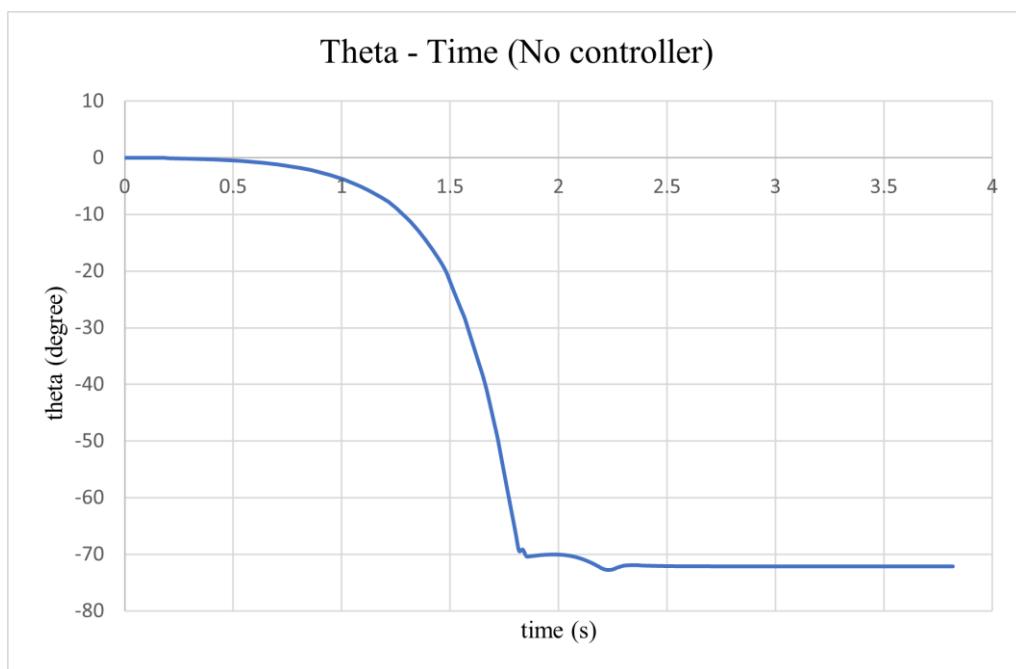
۳-۵- بررسی زاویه و سرعت انحراف موتورسیکلت

در این بخش ابتدا بررسی می‌کنیم که در صورت عدم وجود کنترلر، رفتار موتورسیکلت به چه صورت خواهد بود. سپس به مقایسه‌ی عملکرد کنترلرهای مختلف بر روی موانع مختلف می‌پردازیم. در این بخش منظور از عملکرد موتورسیکلت بررسی مقدار زاویه‌ی انحراف و سرعت تغییرات این زاویه می‌باشد.

^۱ Transfer Learning

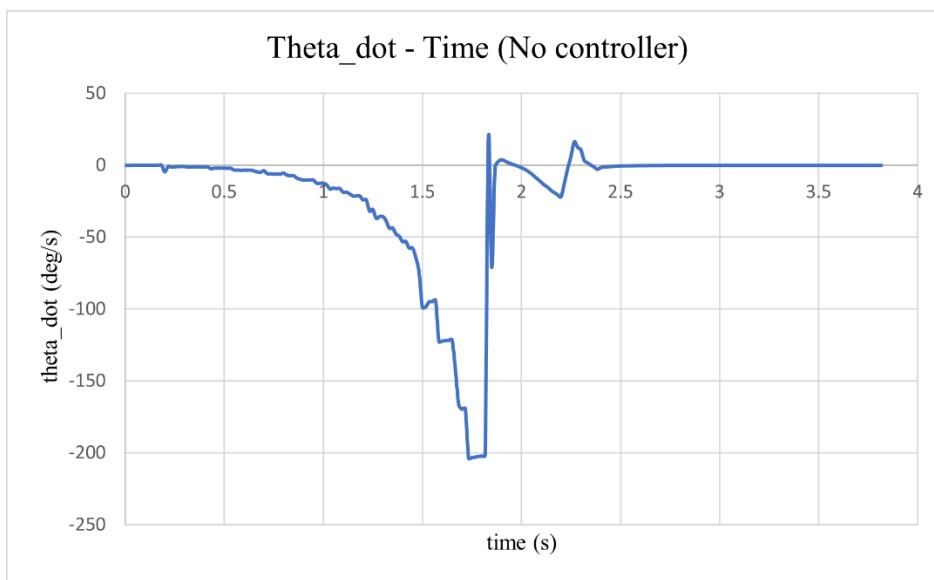
۱-۳-۵- رفتار سامانه بدون افزودن کنترل کننده

یک موتورسیکلت همانند یک پاندول معکوس، سامانه‌ای ناپایدار می‌باشد. در این سامانه به دلیل آنکه مرکز جرم در صفحه گذرنده از چرخ‌ها که تکیه‌گاه سامانه می‌باشند وجود ندارد، پس از آغاز شبیه‌سازی شروع به سقوط از یک سمت نموده و همانطور که انتظار میرفت، بدون داشتن کنترل کننده و نیروی مقاوم در مقابل سقوط، تعادل خود را از دست می‌دهد. همانطور که در نمودار ۱-۵ مشخص است، زاویه‌ی انحراف موتورسیکلت افزایش یافته تا سرانجام کاملاً بر روی زمین افتاده که در این حالت زاویه‌ی ۷۰ درجه دارد.



نمودار ۱-۵ نمودار زاویه‌ی انحراف برحسب زمان در سامانه بدونه کنترل کننده

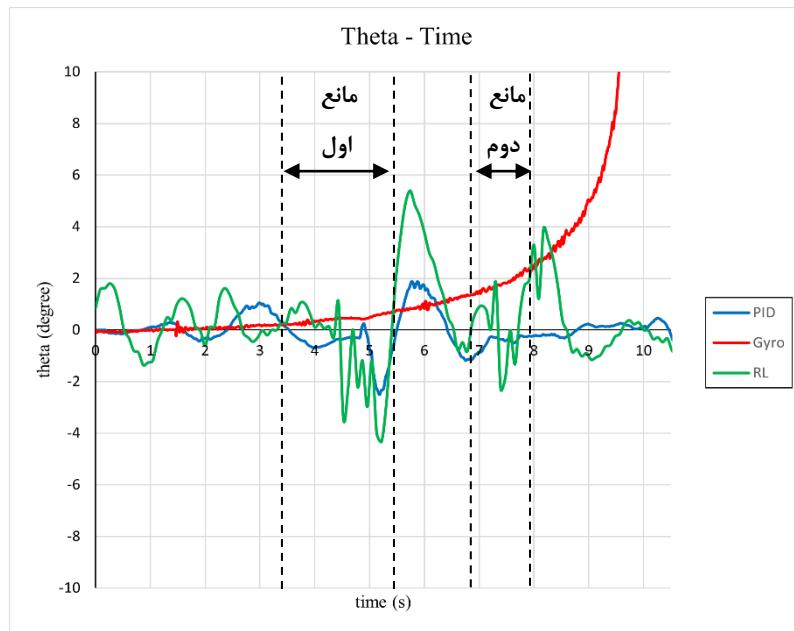
همچنین می‌توان مشاهده کرد که سرعت انحراف از وضعیت تعادلی نیز رفته افزایش می‌یابد و زمانی که کاملاً بر روی زمین قرار گرفت سرعت تغییرات زاویه تقریباً نزدیک به صفر می‌شود. این موضوع در نمودار ۲-۵ قابل مشاهده است.



نمودار ۲-۵ نمودار سرعت زاویه‌ای بر حسب زمان در سامانه بدون کنترل کننده

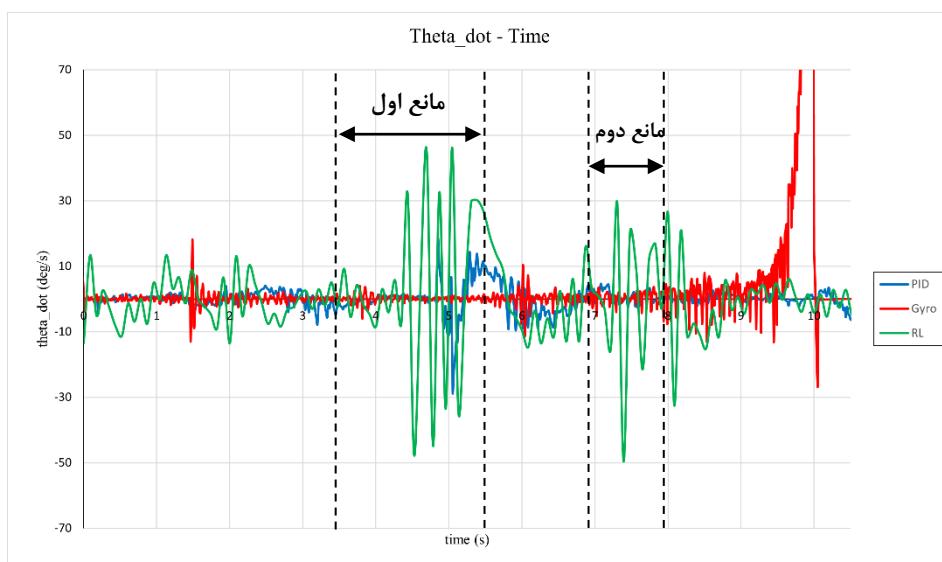
۵-۳-۲- رفتار سامانه حین عبور از دو مانع اول در حضور کنترلرها

با عبور از روی موانع نیرویی خارجی به سامانه وارد می‌شود که در مقدار زاویه‌ی انحراف موتورسیکلت و سرعت آن موثر است. از طرفی در هر لحظه با توجه به متغیرهای ورودی از جمله زاویه و سرعت زاویه‌ای کنترل کننده گشتاوری جهت جبران، تولید می‌کند. همانطور که از نمودار ۳-۵ مشخص است، پایداری ژیروسکوپ در عبور از موانع بهتر می‌باشد ولی با گذشت زمان زاویه‌ی انحراف بیشتر شده است و در نهایت تعادل آن به صورت کامل از دست می‌رود. در حقیقت مانند یک فرفه فقط زمان زمین خوردن وسیله را به تاخیر میندازد. اگر بتوانیم این مشکل را رفع نماییم و به نحوی عمل کند که انحراف را سعی کند از همان ابتدا کامل جبران نماید و نگذارد رفته زیاد شود، به نظر می‌رسد این روش باعث پایداری خوبی دارد و از تکان‌های شدید در موتورسیکلت جلوگیری می‌کند. در بین دو روش یادگیری تقویتی و کنترلر PID به نظر می‌رسد که کنترلر PID بهتر عمل کرده است. از این نظر که هم سعی کرده است دامنه‌ی نوسانات کم باشد و هم آنکه پس از وارد شدن نیروهای خارجی سریع‌تر سعی می‌کند آن‌ها را دفع نماید و به حالت پایدار برسد.



نمودار ۳-۵ مقایسه‌ی زاویه‌ی انحراف در عبور از دو مانع اول برای کنترل‌کننده‌های مختلف

اگر به بررسی سرعت زاویه‌ای هم بپردازیم مشخص می‌شود که به کمک ژیروسکوپ سرعت زاویه‌ای موتورسیکلت تا قبل از ازدست رفتن کامل تعادل حول مقدار صفر با دامنه‌ی نسبتاً کمی نوسان می‌کرده است اما در روش یادگیری تقویتی دامنه‌ی نوسانات بیشتر است و تا حدود ۵۰ درجه بر ثانیه نیز می‌رسد. همانطور که مشخص است، کنترلر PID بعد از ژیروسکوپ از نوسانات کمتری در سرعت زاویه‌ای برخوردار است. نکات گفته شده در نمودار ۴-۵ قابل مشاهده است.

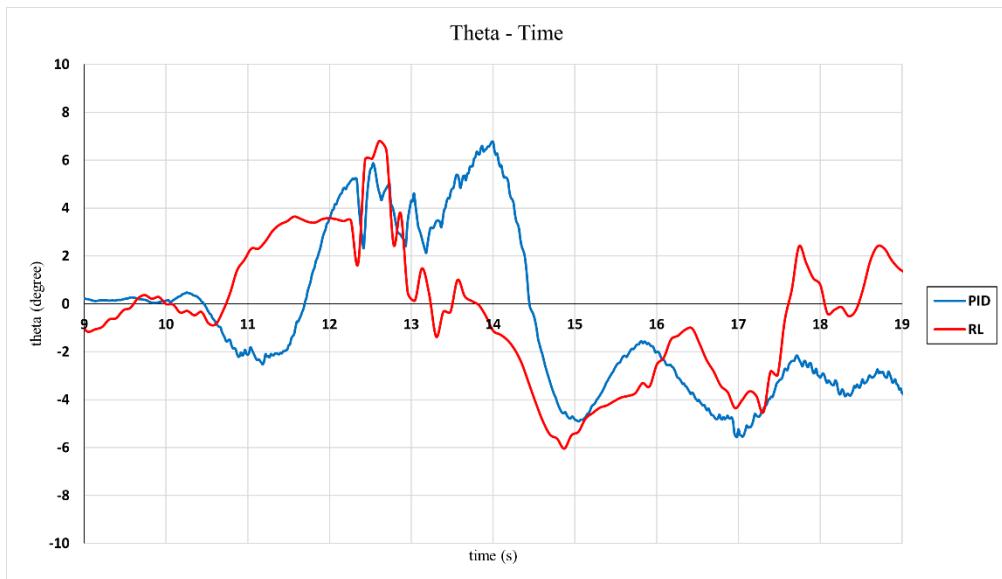


نمودار ۴-۵ مقایسه‌ی سرعت زاویه‌ای در عبور از دو مانع اول برای کنترل‌کننده‌های مختلف

۵-۳-۳- رفتار سامانه حین چرخاندن فرمان در عبور از مانع سوم

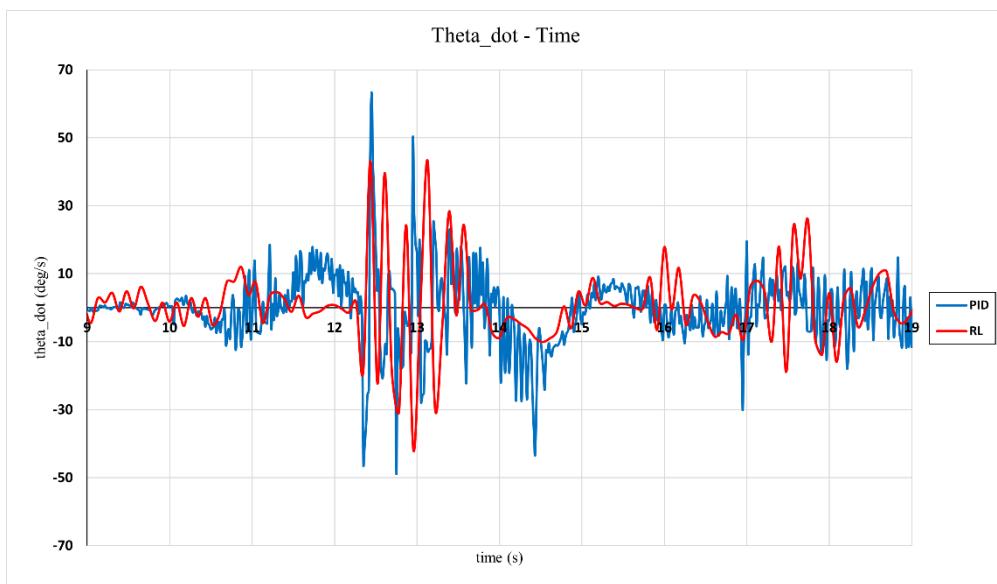
در این بخش، داده‌های مربوط به ژیروسکوپ را نداریم چراکه تعادل موتورسیکلت قبل از رسیدن به مانع سوم، از دست رفت. موتورسیکلت با نزدیک شدن به مانع فرمان را می‌چرخاند تا آن مانع را پشت سر بگذارد. این عمل موجب برهم زدن تعادل موتورسیکلت می‌شود که وظیفه‌ی کنترلر آن است تا تعادل را در این مرحله نیز حفظ نماید تا موتور با وجود چرخش فرمان، کاملاً منحرف نشود.

در بازه‌ی زمانی ۱۰.۵ تا ۱۸ ثانیه، موتورسیکلت در حال پشت سر گذاشتن مانع می‌باشد. همانطور که در نمودار ۵-۵ مشاهده می‌شود زاویه‌ی انحراف موتورسیکلت در این بازه نوسانات زیادی دارد. آنچه که قابل درک است این است که در شیوه‌ی یادگیری تقویتی موتورسیکلت میانگین دامنه‌ی نوسانات کمتری دارد و همچنین سریع‌تر به حالت پایدار می‌رسد.



نمودار ۵-۵ مقایسه‌ی زاویه‌ی انحراف حین چرخش فرمان در عبور از مانع سوم

همچنین اگر برای درک بهتر عملکرد این دو روش در برخورد با این مانع، به سراغ نمودار سرعت زاویه‌ای در نمودار ۵-۶ برویم؛ خواهیم دید که فرکانس و همچنین دامنه‌ی نوسانات در روش یادگیری تقویتی نسبت به استفاده از PID کم‌تر است. همچنین در یادگیری تقویتی، نوسانات سرعت نیز همانند نوسانات زاویه، سریع‌تر به حالت پایدار می‌رسند و برطرف می‌شوند.



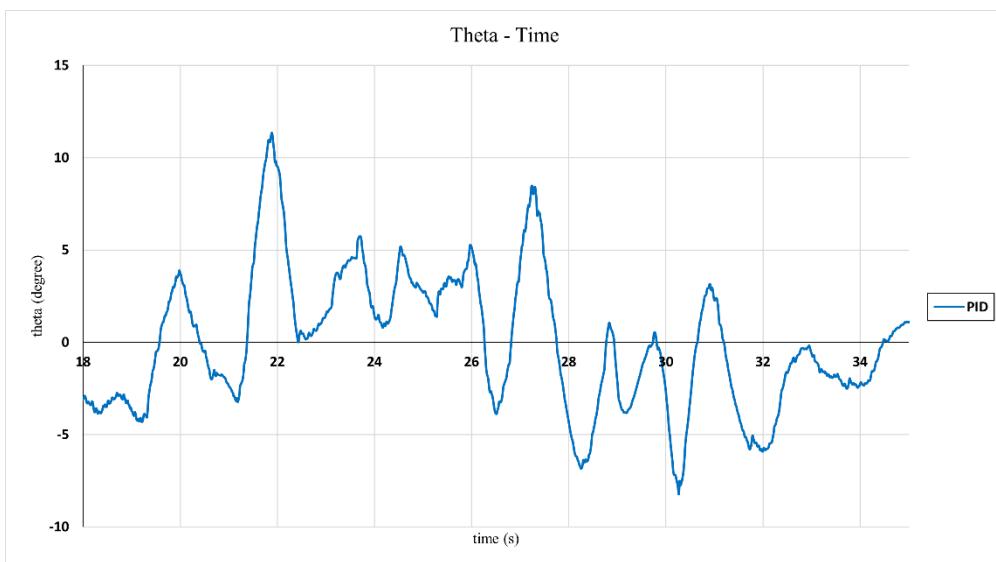
نمودار ۵-۶ مقایسه‌ی زاویه‌ی انحراف حین چرخش فرمان در عبور از مانع سوم

بنابراین می‌توان پی‌برد مدل آموزش دیده در یادگیری تقویتی به هنگام چرخش فرمان و پیچیدن در پیچ‌ها عملکرد بهتری در حفظ تعادل دارد.

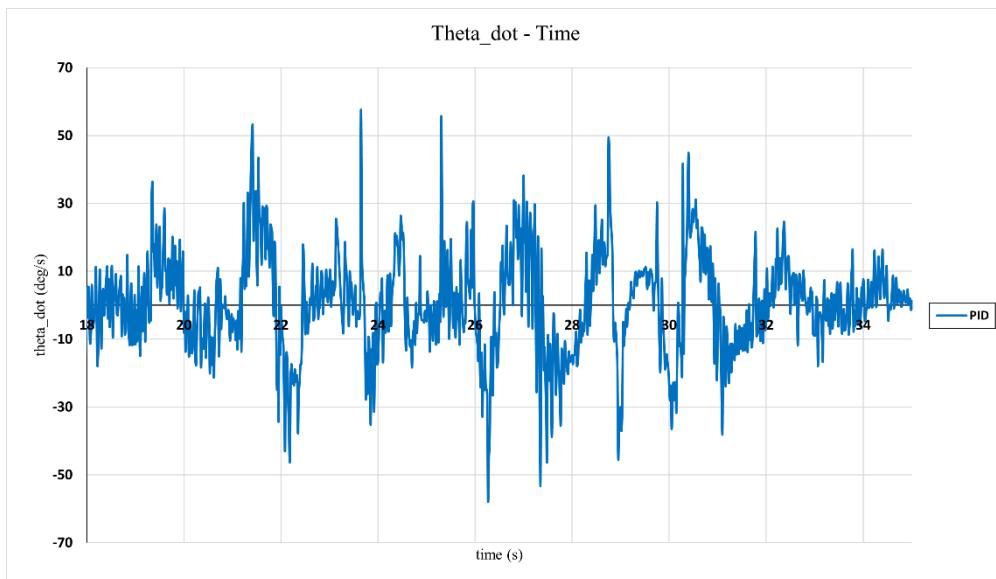
۵-۳-۴- رفتار سامانه در عبور از دستاندازهای متوالی

در این بخش، فقط داده‌های روش استفاده از کنترل کننده PID در دسترس است چراکه دو روش دیگر در عبور از این قسمت تعادل‌شان را از دست می‌دادند. همانطور که در نمودار ۷-۵ مشخص است، فرکانس نوسانات در عبور از این مانع بسیار زیاد است و حتی دامنه‌ی نوسان به ۱۱ درجه می‌رسد که نسبت به تمامی موانع گذشته عددی بزرگ‌تر می‌باشد ولی با این حال کنترل‌کننده PID توانست با موفقیت تعادل موتورسیکلت را حفظ نماید. البته لازم به ذکر است که در این بخش علاوه بر عبور از دستاندازهای متوالی کمی چرخش فرمان هم برای دنبال کردن مسیر اصلی و دور نشدن از آن داشتیم که آن نیز در نوسانات بی تاثیر نبوده است.

اگر به نمودار ۵-۸ توجه کنیم متوجه می‌شویم که فرکانس و دامنه‌ی سرعت زاویه‌ای نیز در این بخش به شدت بالا می‌باشد ولی دامنه‌ی آن در این بخش از ۵۷ درجه برثانیه فرارتر نرفته است.



نمودار ۷-۵ زاویه‌ی انحراف موتورسیکلت در عبور از دستاندازهای متوالی



نمودار ۸-۵ سرعت زاویه‌ای موتورسیکلت در عبور از دستاندازهای متوالی

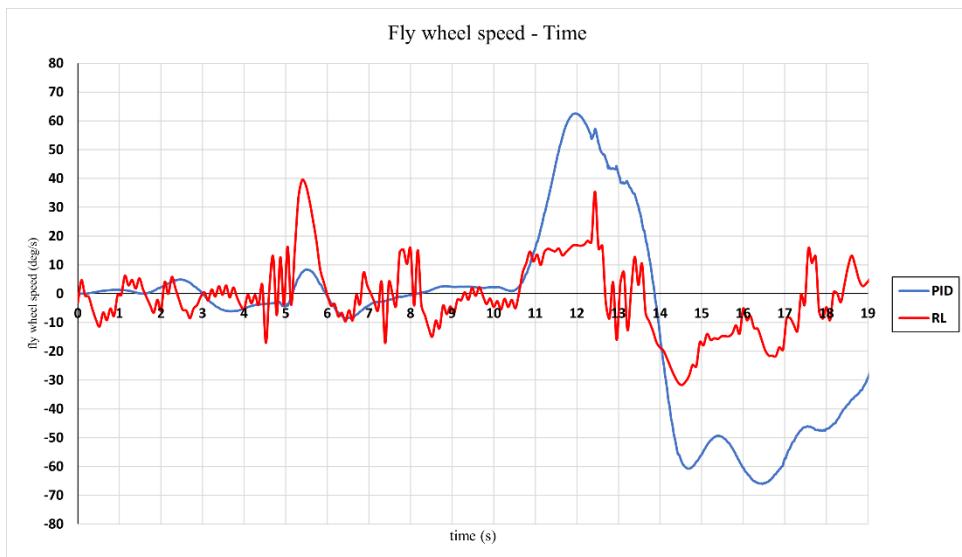
۶-۴- برسی سرعت چرخش چرخ طیار

یکی از عوامل ناپایداری زیاد شدن بیش از اندازه‌ی سرعت چرخ طیار می‌باشد. همین امر سبب شد که هم در روش یادگیری تقویتی و هم در روش کنترل به کمک PID، این متغیر به عنوان یکی از ورودی‌ها در نظر گرفته شود و سعی کنیم تا جای ممکن آن را نزدیک به صفر نگه داریم.

علت زیاد شدن آن، این است که ما ورودی‌ای که به چرخ طیار می‌دهیم، مقدار گشتاور می‌باشد. اگر این گشتاور فارغ از توجه به سرعت چرخ طیار و صرفا براساس زاویه و سرعت زاویه‌ای موتورسیکلت، به آن اعمال شود؛ طبق رابطه‌ی (۱-۵) سبب می‌شود که باگذر زمان سرعت زاویه‌ای، پیوسته زیاد شود و به حدی برسد که شبیه ساز دچار مشکل شود و در واقعیت نیز موتور جریان زیادی بکشد و بسوزد.

$$\omega = \int \alpha dt \approx \int \tau dt \quad (1-5)$$

حال با توجه به نمودار ۹-۵ متوجه می‌شویم با در نظر گرفتن این موضوع، توانستیم سرعت زاویه‌ای چرخ طیار را کنترل نماییم. در بین دو کنترل کننده‌ی طراحی شده براساس PID و همچنین یادگیری تقویتی، مشاهده می‌شود که مقدار فرکانس نوسانات سرعت زاویه‌ای در روش یادگیری عمیق عموماً مقدار کمتر و کنترل شده‌تری دارد. البته همانطور که در قسمت‌های قبل دیدیم نتیجه‌ی این موضوع آن است که پاسخ کنترلی به می‌دهد و زاویه‌ی انحراف در عبور از موانع به جز درحال پیچیدن بیشتر می‌شود. در زمان چرخ‌لندن فرمان در عبور از جعبه، کنترلر PID مقدار نیروی زیادی اعمال نموده که در نتیجه‌ی آن سرعت زاویه‌ای چرخ طیار افزایش یافته و نوسانات شدیدی را ایجاد می‌کند، حال آنکه با مقدار ممکن کمتری هم می‌شد تعادل را حفظ نمود که در نتیجه‌ی آن فراجهش کمتری هم در زاویه‌ی انحراف خواهیم داشت.



نمودار ۹-۵ زاویه‌ی انحراف موتورسیکلت در روش PID در عبور از دستاندازهای متوالی

فصل ششم

جمع‌بندی و پیشنهادات

جمع‌بندی و پیشنهادات

همان ظور که در فصل قبل مشاهده شد، کنترل کننده‌های طراحی شده به کمک روش PID و همچنین یادگیری تقویتی، توانستند تعادل موتورسیکلت را تا حد مطلوبی تضمین نمایند. همچنین ژیروسکوپ به کمک اثر ژیروسکوپی خود قابلیت خوبی در حفظ تعادل و پایداری سیستم دارد که البته محدودیت‌هایی نیز دارد که پیش‌تر به آن‌ها اشاره شد. علاوه بر آن، در آزمایش انجام شده، ژیروسکوپ توانایی حفظ تعادل موتورسیکلت را تا انتهای مسیر موردنظر نداشت.

هریک از روش‌ها نقاط قوت و نقاط ضعفی دارند که با توجه به بررسی نمودارها در فصل قبل به آن‌ها پی بردیم. از اثرات مطلوب ژیروسکوپ، کاهش نوسانات موتورسیکلت می‌باشد در حالی که دو روش دیگر با وجود نوسانات، پایداری و حفظ تعادل سیستم را تا انتهای مسیر تضمین می‌نمایند. همچنین در روش یادگیری تقویتی، نوسانات در هنگام چرخاندن فرمان برای دور زدن مانع از روش PID کمتر بود در حالی که در عبور از موانع سطح شیبدار و دست انداز؛ PID عملکرد بهتری داشت.

در خصوص شبیه‌سازی انجام شده نیز واضح است که موتورسیکلت از حالت سکون شروع به حرکت نموده و رفتہ رفتہ سرعت خود را افزایش می‌دهد تا به سرعت هدف برسد و یا حتی در رسیدن به موانع سرعت خود را کاهش می‌دهد. همین امر باعث می‌شود تا یک کنترل کننده ساده از خانواده PID نتواند در تمام مدت شبیه‌سازی عملکرد کاملاً مطلوبی داشته باشد.

در ادامه پیشنهاداتی جهت بهبود عملکرد این سامانه ارائه شده است. با مرکز شدن روی هریک از روش‌های انجام شده در این پژوهش نیز، می‌توان نوآوری‌هایی داشت که باعث تعادل هرچه بهتر موتورسیکلت شوند.

۱-۶ - تلفیق روش‌های آزمایش شده

هریک از روش‌های آزمایش شده مزایا و معایبی داشتند که به آن‌ها اشاره شد. می‌توان سیستمی طراحی کرد که به صورت بهینه عمل نماید و از مزایای هریک از روش‌های فوق استفاده نماید. برای مثال کنترلی طراحی نماییم که در عبور از موانع از مقادیر خروجی کنترل کننده PID استفاده نماید و در هنگام چرخش فرمان و دور زدن موانع از خروجی مدل آموزش دیده توسط یادگیری عمیق استفاده نماییم.

همچنین می‌توان از یک ژیروسکوپ نیز در سیستم جهت پایداری بیشتر استفاده کرد که البته این موضوع هزینه‌ها را نیز افزایش می‌دهد و یک ژیروسکوپ در حال چرخش مداوم انرژی زیادی مصرف می‌نماید.

۶-۲- استفاده از یادگیری ماشین بر روی داده‌های به دست آمده

باتوجه به داده‌های جمع آوری شده در روش‌های مختلف می‌توان به کمک الگوریتم‌های مختلف یادگیری ماشین، مدلی را توسعه داد که با گرفتن برخی ورودی‌ها مانند زاویه‌ی انحراف موتورسیکلت و سرعت زاویه‌ای و همچنین سرعت زاویه‌ای چرخ طیار، مقدار مناسبی را به عنوان گشتاور لازم برای چرخش چرخ طیار تولید نماید.

در حقیقت براساس پژوهش انجام گرفته، می‌دانیم که باتوجه به ورودی‌ها چه مقدار گشتاور جهت تعادل سیستم موردنیاز است. حال می‌توان داده‌های مطلوب روش‌های مختلف را با یک دیگر ترکیب کرد و سپس به عنوان داده‌ی آموزش در اختیار الگوریتم یادگیری ماشین قرار دهیم. این الگوریتم‌ها می‌توانند به صورت بهینه‌تری خروجی را برای ما تولید نمایند و حتی از نوسانات سیستم نیز به کمک این روش بکاهیم.

همچنین کار دیگری که می‌توان انجام داد، آن است که باتوجه به داده‌ها بهره‌های کنترل‌کننده PID را نیز بهینه نماییم و برای تنظیم این ضرایب از یادگیری ماشین استفاده نماییم. به این شکل دیگر نیازی نیست زمان زیادی جهت تنظیم ضرایب صرف کنیم و همچنین این روش ضرایب دقیق‌تری را برای مدل کردن سیستم در اختیار می‌دهد.

۶-۳- استفاده از کنترل‌کننده‌های دیگر

از سویی دیگر می‌توان از دیگر انواع کنترل‌کننده نظری منطق فازی، کنترل تطبیقی، کنترل مقاوم و ترکیب این نوع کنترل‌کننده‌های پیشرفت‌های تر استفاده نمود تا شاهد رفتار مطلوب‌تر سامانه تحت اثر کنترل‌کننده‌ها باشیم. به طور مثال در یک پژوهش انجام شده در خصوص طراحی کنترل‌کننده‌ای برای حفظ پایداری دو پاندول معکوس متصل به یکدیگر با فنر که سامانه‌ای ناپایدار و غیرخطی بوده و از این‌رو

مشابه سامانه موردبحث این پژوهش است، از کنترل‌کننده‌هایی با استفاده از منطق فازی و کنترل‌کننده PID استفاده شده است.

۶-۴- بالا بردن دقیق در شبیه‌سازی

از جمله محدودیت‌هایی که داریم محدودیت سخت افزار برای شبیه‌سازی می‌باشد. با توجه به توضیح مطرح شده در فصل‌های قبل در رابطه با عوامل خطا و اثر تاخیر زمانی در رفتار کنترل کننده، می‌توان نرخ شبیه‌سازی نرم‌افزار را بالاتر برده تا میزان تاخیر زمانی در رفتار کنترل کننده کاهش یافته و اثرباری مطلوب‌تری از سوی کنترل کننده مشاهده گردد.

همچنین مدل‌سازی این سامانه به صورت یک سامانه چهار جسمی انجام شده است. به منظور شبیه‌سازی دقیق‌تر این سامانه، می‌توان مدلی با تعداد اعضا و درجات آزادی بیشتری تولید نمود. به عنوان مثال، مطابق مدل ارائه شده در فصل اول، می‌توان هر یک از اعضای بدن و فرمان را به دو قسمت مجزا تقسیم نموده و با اعمال شرایط مربوط به میرایی و فنریت موجود در اتصالات آنها، مدل فنر و میراگر موجود در موتورسیکلت را نیز به خوبی شبیه‌سازی نمود تا در صورت نیاز به شبیه‌سازی عبور موتورسیکلت از روی موانع، رفتار سامانه شبیه‌سازی شده به واقعیت نزدیکتر باشد.

منابع و مراجع

- [1] Martínez-Díaz, Margarita & Soriguera, Francesc. (2018). Autonomous vehicles: theoretical and practical challenges. *Transportation Research Procedia*. 33. 275-282. doi: 10.1016/j.trpro.2018.10.103.
- [2] N. Vu and H. Nguyen, "Balancing Control of Two-Wheel Bicycle Problems," *Mathematical Problems in Engineering*, vol. 2020, pp. 1-12, 07/22 2020, doi: 10.1155/2020/6724382.
- [3] N. K. Vu and H. Q. Nguyen, "Design Low-Order Robust Controller for Self-Balancing Two-Wheel Vehicle," *Mathematical Problems in Engineering*, vol. 2021, p. 6693807, 2021/05/24 2021, doi: 10.1155/2021/6693807.
- [4] P. Gogoi, M. Nath, B. Doley, A. Boruah, and H. Barman, "Design and Fabrication of Self Balancing Two Wheeler Vehicle Using Gyroscope," *International Journal of Engineering and Technology*, vol. 9, pp. 2051-2058, 06/30 2017, doi: 10.21817/ijet/2017/v9i3/1709030206.
- [5] Serrano, H. How a physics engine works an overview. 2016; Available from: <https://www.haroldsserrano.com/>
- [6] Mehroz, K. Top 5 Applications of Physics Engine in the Animation Industry. 2023; Available from: <https://www.goodfirms.co/>
- [7] CM-LABS. Vortex Studio Introduction. Available from: <https://www.cmlabs.com/vortex-studio/>
- [8] CM-LABS, Theory Guide: Vortex Software's Multibody Dynamics Engine
- [۹] اسدزاده, م. مدلسازی دینامیکی و کنترل پایداری موتورسیکلت. ۱۳۹۹، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران).
- [۱۰] آل علی, ا. مدلسازی دینامیکی و طراحی کنترلر برای پایدار سازی موتورسیکلت در فضای سه بعدی. م. عسکری, ۱۴۰۰، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران).

[11] CM-LABS. Vortex Studio User Guide. 2022; Available from: <https://vortexstudio.atlassian.net/wiki/spaces/VSD223/overview>.

[۱۲] آرانی، ع. مدلسازی موتورسیکلت خودران و طراحی کنترلر پایدارکننده برای آن با استفاده از نرم افزار موتور فیزیکی ۱۴۰۱، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران).

[13] REALPARS. PID Controller Explained. Available from: <https://realpars.com/pid-controller/>

[14] Kim, Hyun-Woo & An, Jae-Won & Yoo, Han & Lee, Jang-Myung. (2013). Balancing control of bicycle robot using PID control. 145-147. 10.1109/ICCAS.2013.6703879.

[15] Le, Tuyen & Chung, TaeChoong. (2017). Controlling bicycle using deep deterministic policy gradient algorithm. 413-417. 10.1109/URAI.2017.7992765.

[16] Choi, SeungYoon & Le, Tuyen & Nguyen, Quang & Layek, Abu & Lee, SeungGwan & Chung, TaeChoong. (2019). Toward Self-Driving Bicycles Using State-of-the-Art Deep Reinforcement Learning Algorithms. Symmetry. 11. 290. 10.3390/sym11020290.

Abstract

Abstract

Nowadays, the proliferation of personal vehicles in large cities has led to numerous problems, including an increase in traffic accidents, urban congestion, and fuel consumption. Many companies have turned to the development of self-driving electric vehicles to mitigate these issues, particularly making significant advancements in self-driving cars. However, autonomous motorcycles, which offer smaller dimensions and lower fuel consumption, have received less attention, despite their suitability for various tasks such as emergency response and package delivery.

To transform a motorcycle into a fully autonomous version, the primary challenge lies in maintaining balance. This complexity has made the process of converting motorcycles into self-driving vehicles more intricate compared to self-driving cars. In this research, we aim to preserve motorcycle balance by employing various tools like gyroscopes, fly-wheel mechanisms, or motorcycle steering, and utilizing controllers such as PID and artificial intelligence-based methods like reinforcement learning.

Since experimenting with various methods on real-world samples can be costly and potentially risky, one of the best approaches is the use of physics engine-based simulators. In this study, by modeling motorcycles in software such as Vortex Studio and implementing the mentioned control methods, we were able to maintain motorcycle balance, even while navigating obstacles and avoiding significant obstacles. Following this challenging phase, it becomes possible to equip motorcycles with additional features like route planning, traffic sign recognition, and more, like self-driving cars.

Key Words:

Self-driving motorcycle, Reinforcement learning, PID controller, Vortex Studio



**Amirkabir University of Technology
(Tehran Polytechnic)**

Mechanical Engineering Department

BSc Thesis

**Maintaining the balance of the self-driving
motorcycle with the help of intelligent methods**

**By
Mahdi Rahmani**

**Supervisor
Dr. Ali Azimi**

September 2023