# 

Escuela Internacional de Posgrado

Master Universitario en Ciencia de Datos e Ingeniería de Computadores

Trabajo de Fin de Master

**Análisis y estudio en frameworks de simulación sobre la generación de datos en problemas robótica industrial y su aplicación en tareas de aprendizaje automático.**

Presentado por: Hodei Zia López

Tutores: Alberto Luis Fernández Hilario y Mikel Galar Idoate

15/09/2022

Resumen

Este proyecto pretende abordar la problemática que se genera a la hora de buscar con qué datos trabajar en un problema de predicción de trayectorias para la realización de tareas en robótica industrial. Para ello, comenzaremos analizando algunos de los frameworks de simulación más utilizados para este tipo de proyectos. Es por este motivo, que además de analizar diferentes frameworks, realizaremos también un minucioso estudio sobre otros trabajos realizados dentro de este campo con el objetivo de obtener un mayor conocimiento con el que abordar el problema. A continuación, tras haber realizado este estudio inicial, comenzaremos con el apartado más importante del proyecto, la generación de datos. En este apartado, intentaremos extraer la mayor cantidad posible de datos de la tarea a analizar para poder evaluar sus posibles configuraciones en la fase de entrenamiento del modelo mediante una red neuronal. Una vez obtengamos el modelo, volveremos al simulador para probar la calidad de los datos utilizados así como de la red en cuestión. Todo esto lo haremos sobre diferentes entornos de simulación para evitar un sesgo en los datos y con el fin de generalizar qué tipo de datos se han de utilizar para futuros problemas relacionados con la robótica dentro de este ámbito.

Palabras clave

* Robótica industrial
* Generación de datos
* Entorno de simulación
* Aprendizaje automático
* RLBench
* Metaworld

Índice

[1](#_Toc108339828)

[Resumen 3](#_Toc108339829)

[Palabras clave 3](#_Toc108339830)

[Índice 4](#_Toc108339831)

[Capítulo 1 5](#_Toc108339832)

[Introducción 5](#_Toc108339833)

[1.1. Objetivo 6](#_Toc108339834)

[Capítulo 2 7](#_Toc108339835)

[Marco teórico 7](#_Toc108339836)

[2.1. Estado del arte 7](#_Toc108339837)

Capítulo 1

Introducción

La robótica industrial es una rama de la ingeniería que pretende realizar múltiples procesos industriales tales como la manipulación de objetos haciendo uso de robots con el objetivo de completar diversas tareas en cadena de forma automática de modo que necesitemos la mínima supervisión humana posible. Partiendo de esta definición, podemos encontrarnos tanto con tareas que no dependan del entorno y que por tanto, se realicen aplicando los mismos movimientos sobre el robot como con otras en las que tanto el entorno como los objetos u obstáculos que se encuentren en él sean cambiantes, por lo que necesitemos hacer que nuestro robot adapte sus movimientos, decisiones o incluso fuerza en base a lo que se encuentre en cada momento. Para ello, debemos hacer que el robot consiga manejarse en distintas situaciones, haciendo que aprenda de una cantidad de escenarios lo suficientemente amplia como para que sepa cómo actuar frente a un obstáculo inesperado nunca antes visto o aprendido en la fase de entrenamiento. Es aquí donde cobra una importancia real la realización de un estudio minucioso sobre la generación de datos como pueden ser distintos escenarios, la clase de imágenes sobre el entorno que van a ser capaces de aportar una mayor información, el tipo de acciones con las que el robot va a ser capaz de generalizar mejor el problema…

En este contexto, es común simular distintas tareas que pueda realizar un brazo robótico tales como pueden ser coger una pelota, apilar unos cubos, entre otros, y evaluar los datos devueltos en busca de los casos de estudio más adecuados. A partir de los mismos, se realizará un proceso de entrenamiento para generar un modelo que sea capaz de replicar estas tareas en sus simuladores correspondientes. Adicionalmente, se estudiará la mejor sinergia entre caso de estudio y simulador con vistas a formular una generalización de los tipos de datos que se han de utilizar para futuros problemas relacionados con la robótica en el ámbito de la inteligencia artificial.

# Objetivo

El objetivo principal será obtener una configuración general de datos con el que se caractericen distintos casos de uso en robótica industrial. Para ello, desglosamos entre los siguientes subobjetivos:

1. Estudiar y analizar en profundidad tanto los distintos estudios y proyectos realizados en este ámbito como los diferentes frameworks de simulación utilizados, dejando claras tanto sus diferencias como sus similitudes.
2. Generar datos para un conjunto de tareas con cada uno de los frameworks que sean los que utilicemos para generar los distintos modelos en la fase de entrenamiento.
3. Analizar las mejores configuraciones posibles de datos en busca de un patrón capaz de generar una generalización para futuros problemas de modo que sepamos de donde partir inicialmente sea cual sea el problema a evaluar. Para esto trataremos de encontrar las configuraciones que maximicen el RMSE como métrica estándar.
4. Simular en los robots los modelos generados y visualiza los resultados obtenidos mediante diferentes aproximaciones.

Capítulo 2

Marco teórico

En este capítulo vamos a exponer toda la parte teórica recopilada tras realizar una investigación documental sobre distintos trabajos centrados en el mismo tema. De igual forma, una vez finalizado dicho apartado, pasaremos a hablar sobre los diferentes frameworks de simulación que hemos utilizado.

# 2.1. Estado del arte

# 2.2. Frameworks de simulación utilizados

Como ya hemos ido mencionado a lo largo del inicio de este documento, un entorno de simulación es una aplicación software que permite al usuario trabajar tanto con una amplia variedad de robots como de tareas y entornos virtuales totalmente personalizables a los gustos y necesidades de cada uno. Al ser un entorno virtual, no requiere de la disposición de ninguna clase de herramienta física, haciendo así que ahorremos tiempo y dinero en la compra y adjudicación de la puesta en marcha de todo lo necesario para simular cualquier tipo de tarea.

El objetivo principal de cualquier entorno de simulación es el de emular y validar tareas dentro de un mundo virtual antes de hacerlo en el mundo físico para así, tal y como hemos comentado antes, evitar sobrecostes y ahorrar tiempo, ya que es mucho más rápido y menos costoso simular una tarea dentro de un software que podemos modificar, iniciar o detener a nuestro antojo antes que hacerlo sobre un robot físico el cual muy posiblemente requiera de unos cuidados y puesta en marcha muy específicos. De esta manera, podemos probar distintas configuraciones de un modo muy sencillo para posteriormente evaluar cual es la que finalmente debemos llevar a cabo sobre nuestro robot o entorno físico.

Para ello, dentro de casi cualquier entorno, tenemos la libertad para crear y modificar materiales, objetos e incluso comportamientos de una manera rápida que en el mundo real no seriamos capaces de realizar tan fácilmente.

Para la realización de este proyecto, y con el objetivo principal de evitar un sesgo en los datos por culpa de las características específicas de cada framework, hemos decidido utilizar dos de los entornos de simulación de datos más conocidos en problemas de robótica industrial: MuJoCo y CoppeliaSim. Cada uno de estos entornos de simulación es capaz de utilizar distintas librerías que son las encargadas de especificar al sistema como generar, simular y trabajar con sus propias herramientas. Nosotros hemos seleccionado las dos más utilizadas en este tipo de proyectos: RLBench y Metaworld.

## 2.2.1. CoppeliaSim

Este entorno de simulación, antes conocido como V-Rep, es uno de los mas utilizados para la simulación de cualquier tipo de tarea dentro de la comunidad robótica. Fue desarrollado por Coppelia Robotics en Marzo de 2010 y a día de hoy, según el servicio de búsqueda de patentes *The Lens [x]*, cuenta con un total de 1.282.860 citas distribuidas entre 73.775 patentes, por lo que podemos hacernos a la idea del prestigio que tiene.

Diagrama

Descripción generada automáticamente Cuenta con una interfaz gráfica muy bien estructurada en donde podemos ver toda la información sobre la escena que tenemos delante así como modificar parámetros o insertar nuevos objetos a ella. A continuacion se muestra un pantallazo de lo que seria esta interfaz grafica con una escena cargada en ella:

Figura x. Interfaz gráfica de CoppeliaSim

Como podemos apreciar en la figura anterior, al lado izquierdo tenemos la parte que mas ocupa en pantalla, la visualización de la escena en tiempo real. Dentro de ella podemos ver tanto los objetos que participan activamente en la escena (mesas, cubos, robots…) como los objetos pasivos (cámara, dirección de la luz…). Todos estos objetos, sean del tipo que sean, pueden ser añadidos desde la ventana llamada *Model browser* que se encuentra en el lado derecho de la pantalla, en donde tenemos una estructura jerárquica de directorios en la cual podemos encontrar todo tipo de objetos además de visualizarlos debajo tras hacer click en cualquiera de ellas. Entre medias de las dos pantallas comentadas, tenemos la denominada como *Scene hierarchy*. Esta pantalla engloba la jerarquía de todos los objetos escogidos y puestos a escena por partes, es decir, teniendo la opción de modificar cada objeto por trozos, lo cual resulta muy útil, ya que amplia la capa de personalización disponible. Finalmente, arriba del todo, cogiendo toda la pantalla en el eje horizontal, tenemos la barra de simulación en donde podemos ajustar la tasa de refresco, modificar la posición de los objetos, movernos por la escena, acelerar o decelerar la simulación y muchas mas acciones relacionadas con las ahora mencionadas.

Al estar basado en una arquitectura de control distribuida, podemos programar escenas mediante seis métodos distintos:

* Scripts embebidos
* API remota de cliente
* Add-ones
* Plugins
* Nodo ROS/ROS2
* Nodo ZeroMQ

Cada cual presenta ventajas y desventajas respecto al resto de métodos tal y como vamos a mostrar a continuación:

Tabla

Descripción generada automáticamente

Figura x. Interfaz gráfica de CoppeliaSim

De entre todos estos métodos de programación, nosotros vamos a trabajar en Python con scripts embebidos provenientes de la librería RLBench [x] y modificaciones que haremos sobre ellos, de modo que tan solo usemos la interfaz gráfica para extraer información sobre la escena en todo momento tal y como explicaremos más adelante.