

# Índice general

<b>1. Plataforma de Desarrollo</b>	<b>7</b>
1.1. Blender . . . . .	7
1.2. Simulador Gazebo . . . . .	8
1.3. JdeRobot . . . . .	8
1.4. ROS . . . . .	9
1.4.1. MoveIt! . . . . .	11
1.4.2. rviz . . . . .	11
1.4.3. rqt . . . . .	11
1.5. ARIAC . . . . .	12
<b>2. Practica Mónaco</b>	<b>13</b>
2.1. Blender . . . . .	13
<b>Bibliografía</b>	<b>15</b>



# Índice de figuras



# Índice de cuadros



# Capítulo 1

## Plataforma de Desarrollo

### 1.1. Blender

Blender[1] es un software libre y gratuito de creación en 3D. Está diseñado para realizar tareas como modelado, iluminación, renderizado, animación y creación de gráficos tridimensionales, edición de vídeo y escultura y pintura digital. En Blender, además, se pueden desarrollar vídeo juegos ya que posee un motor de juegos interno. Actualmente es compatible con todas las versiones de Windows, Mac OS X, GNU/Linux (Incluyendo Android), Solaris, FreeBSD e IRIX. Su interfaz utiliza OpenGL<sup>1</sup> (*Open Graphics Library*) para proporcionar una experiencia consistente y de calidad.

Blender fue liberado al mundo bajo los términos de la Licencia Pública General de GNU v2 (GPL)<sup>2</sup>, y su desarrollo continúa conducido por un equipo de voluntarios procedentes de diversas partes del mundo y liderados por el creador de Blender, Ton Roosendaal.

Se eligió este programa para realizar la edición 3D de mundos para Gazebo frente a alternativas como 3DSMax, Maya o XSI por diversos motivos: es más ligero que sus competidores; posee más herramientas de escultura 3D; la comunidad es muy activa y hay gran cantidad de información, tutoriales y soluciones disponibles; y es de distribución comercial libre y gratuita.

---

<sup>1</sup><https://www.opengl.org/>

<sup>2</sup><https://www.gnu.org/licenses/old-licenses/gpl-2.0.html>

## 1.2. Simulador Gazebo

Gazebo[2] es un simulador 3D de robots para interiores y exteriores, con un motor de físicas y cinemáticas muy potente. Dispone de un conjunto de plugins que facilita la integración con ROS, lo cual agiliza el desarrollo de código y permite la simulación de algoritmos antes de implementarlos en el robot físico, lo cual se puede lograr sin realizar apenas cambios en el código. Además está mantenido por una comunidad activa y la OSRF<sup>3</sup> (*Open Source Robotics Foundation*), la cual también da soporte a ROS, y fué elegido para realizar el DARPA Robotics Challenge<sup>4</sup> entre 2012 y 2015.

Cabe destacar que Gazebo se compone principalmente de un cliente y un servidor. El servidor es el encargado de realizar los calculos y la generación de los datos de los sensores, y puede ser usado sin necesidad de una interfaz gráfica, por ejemplo en un servidor remoto. El cliente proporciona una interfaz gráfica basada en QT que incluye la visualización de la simulación y una serie de controles de multitud de propiedades. Esta configuración permite lanzar multiples clientes sobre un servidor, consiguiendo multiples interfaces de la misma simulación

## 1.3. JdeRobot

JdeRobot[3] es una suite de desarrollo de software de robótica, domótica y sistemas de visión computerizados cuya última versión, la 5.5, es la usada en este proyecto y permite la integración con ROS Kinetic. Proporciona un entorno distribuido donde las aplicaciones se forman mediante una colección de componentes asíncronos. Estos componentes utilizan interfaces ICE<sup>5</sup> para comunicarse, lo que permite lanzarlos desde distintos equipos y que estén escritos en diferentes lenguajes, como C++, Python o Java.

JdeRobot simplifica el acceso a elementos de hardware, siendo tan simple como realizar una llamada a una función. También permite que los sensores o actuadores con los que se comunica sean reales o simulados, conectados mediante la red tanto dentro de la misma máquina como en una red local o de forma remota mediante internet. Actualmente se han desarrollado drivers para una multitud de dispositivos, como por ejemplo sensores RGB como Kinect o cámaras USB o IP, vehículos como Kobuki o Pioneer, drones como el ArDrone de Parrot, etc.

---

<sup>3</sup><http://www.osrfoundation.org/>

<sup>4</sup><http://www.theroboticschallenge.org/>

<sup>5</sup><http://www.zeroc.com/>



Es un software libre, licenciado como GPL y LGPL que se sirve de software como Gazebo, ROS, OpenGL, QT... que se usa tanto para docencia como para investigación en la URJC y ha formado parte del *Google Summer of Code 2015*<sup>6</sup>, programa donde Google premia a los estudiantes al completar un proyecto de programación de software libre durante un verano.

## 1.4. ROS

El Sistema Operativo de Robots, ROS[5] (*Robot Operating System*) es un entorno de trabajo flexible donde desarrollar software para robots. Es una colección de herramientas y librerías que buscan simplificar la tarea de crear comportamientos robustos y complejos en una amplia variedad de plataformas robóticas.

Desde mediados de la década de los 2000 se han realizado esfuerzos para aunar los entornos y herramientas existentes en sistemas de software dinámicos y flexibles para su uso en robots. Estos esfuerzos culminaron, gracias al interés y la ayuda de innumerables desarrolladores, en las ideas que forman el núcleo de ROS y sus paquetes de software fundamentales. Desde entonces se ha expandido su uso bajo la licencia de software libre BSD<sup>7</sup> y se ha convertido en una plataforma muy usada en la comunidad de investigadores.

Desde el inicio ROS se desarrolló en multitud de instituciones para multitud de robots, permitiendo a una gran variedad de investigaciones tener éxito bajo esta plataforma. Sólo el núcleo de ROS se mantiene en los servidores centrales, cualquier desarrollador es libre de crear, desarrollar y compartir sus propias ideas y proyectos de forma que, si así lo desea, estén disponibles para toda la comunidad desde sus propios repositorios. De esta forma se consigue mantener un ecosistema formado por decenas de miles de usuarios a nivel global, desde proyectos como hobby hasta sistemas industriales automatizados.

ROS se diseñó para ser modular y fragmentado, de modo que los usuarios pueden usar sólo las partes que necesiten. En bajo nivel ofrece una interfaz de comunicación por mensajes que permite ahorrar tiempo manejando los detalles de la comunicación entre nodos mediante un mecanismo anónimo de publicación/subscripción de mensajes estructurados. Este sistema fuerza al usuario a implementar interfaces limpias entre los nodos del sistema, mejorando la encapsulación y promoviendo la reutilización de código.

Adicionalmente ROS proporciona librerías y herramientas para agilizar el trabajo de sus

---

<sup>6</sup><https://summerofcode.withgoogle.com/organizations/6493465572540416/>

<sup>7</sup>[https://es.wikipedia.org/wiki/Licencia\\_BSD](https://es.wikipedia.org/wiki/Licencia_BSD)

usuarios. Dado el carácter colaborativo y comunitario del proyecto, se han unificado una gran variedad de formatos mensajes estándar que cubren la mayoría de las necesidades básicas en robótica, tales como posiciones, transformaciones, vectores, sensores como cámaras o lasers, datos de navegación como caminos o mapas, etc.

Un problema común que aborda ROS es la descripción de un robot de forma que sea comprensible para un ordenador, consiguiendo un Formato Unificado de Descripción del Robot o URDF (*Unified Robot Description Format*). Consiste en un fichero XML en que se describen las propiedades físicas del robot, partiendo del cual el robot se puede utilizar con librerías, simuladores y planificadores de movimientos.

También proporciona herramientas de diagnóstico, estimación, localización, navegación, así como una colección de herramientas gráficas y de línea de comandos para facilitar el desarrollo y la depuración. Las herramientas de línea de comandos permiten la utilización de ROS desde cualquier terminal, incluso con conexión remota. Las herramientas gráficas incluyen rviz y rqt, muy potentes tanto para planificar como para desarrollar proyectos en ROS.

Para entender mejor cómo funciona ROS podemos pensar en un sistema de grafos en el que situamos los siguientes elementos:

- **Nodos:** Son las partes de código que se ejecutan. Escritos en C++ o Python permiten realizar tareas en el robot, subscribirse y publicar en topics o proporcionar y usar servicios. De esta manera se facilita el diseño modular de los proyectos.
- **Topics:** Son las vías de comunicación usadas por los nodos. Cada topic utiliza un único tipo de mensaje, de esta manera un nodo puede utilizar varios topics para comunicarse.
- **Mensajes:** Son los datos estructurados que se envían entre topics. En ROS existen varios tipos definidos para los mensajes más utilizados, pero se pueden definir nuevos tipos de mensajes de acuerdo con las necesidades particulares de cada proyecto
- **Servicios:** A diferencia de los topics, son vías de comunicación síncronas entre nodos compuestas por dos mensajes: uno de petición y otro de respuesta. De esta forma el nodo que envía la petición espera hasta recibir la respuesta.

### 1.4.1. MoveIt!

MoveIt![6] es un software de código abierto para ROS (Robot Operating System) que es el estado de la técnica de software para la manipulación móvil. De hecho, podríamos afirmar que se está convirtiendo en un estándar de facto en el campo de la robótica móvil, ya que hoy en día más de 65 robots utilizan este software.

Incluye diversas utilidades que aceleran el trabajo con brazos robóticos, y sigue la filosofía de ROS de reutilización de código. Este software permite llevar a cabo tareas de planificación de trayectorias complejas, percepción 3D, cálculos cinemáticos, control de colisión, control y navegación de forma sencilla, accediendo por la API o mediante las herramientas de la consola.

### 1.4.2. rviz

Rviz[7] es una herramienta de visualización en 3D llamada que posibilita que prácticamente cualquier plataforma robótica pueda ser representada en imagen 3D, respondiendo en tiempo real a lo que le ocurre en el mundo real. Se puede usar para mostrar lecturas de sensores y obtener información de estado de ROS.

Usado en conjunto con MoveIt! permite mostrar el brazo en su estado actual, la colocación del brazo en una posición objetivo, y la visualización del camino pensado por MoveIt! y del movimiento real del brazo siguiendo dicho camino.

### 1.4.3. rqt

Rqt[8] es un software de ROS que implementa varias herramientas de GUI (*Graphical User Interface*) en forma de plugins, permitiendo cargarlas unificadas como una ventana en la pantalla facilitando trabajo al usuario. Simplemente con un comando en la consola, *rqt* muestra una ventana donde elegir cualquier plugin disponible en el sistema en ese momento.

Contiene una herramienta que ha resultado muy útil para la realización de este proyecto: *rqt\_graph*. Al introducir en la consola *roslaunch rqt\_graph rqt\_graph* crea un grafo dinámico que muestra qué nodos y qué topics están activos en ese momento y cuál es su relación. Al situar el ratón encima de cada elemento marcará con un código de color cuál es el elemento activo, de qué tipo es y cuál es su relación con los demás elementos del grafo.

## 1.5. ARIAC

ARIAC[4] (*Agile Robotics for Industrial Automation Competition*) es una competición pionera cuyo objetivo es probar la agilidad de los sistemas robóticos industriales. Realizada por primera vez en Junio de 2017, nace de un esfuerzo conjunto entre la Conferencia de Automatización en Ciencia e Ingeniería del IEEE o CASE (*Conference on Automation Science and Engineering*) y el NIST (*National Institute of Standards and Technology*). Se sirve de Gazebo como plataforma de simulación y de un conjunto propio de modelos, plugins y scripts para simular un brazo robótico en un entorno dinámico, todo ello elaborado con la ayuda de la OSRF (*Open Source Robotics Foundation*). Tiene el objetivo de aumentar la productividad y la autonomía de los robots industriales, entendiendo como agilidad la cosecución de manera automática de identificación de fallos y recuperación de los mismos, automatización para disminuir la reprogramación ante cambios en la producción, interacción con el entorno incluso en áreas no previstas inicialmente, y la capacidad de *plug and play*, es decir, de introducir robots de otros fabricantes sin la necesidad de reprogramarlos.

En este proyecto nos servimos del entorno y del propio brazo robótico para desarrollar nuestros propios plugins para entender y controlar el brazo por medio de ROS.

# Capítulo 2

## Practica Mónaco

Una vez explicado el contexto, los objetivos y las herramientas usadas en el proyecto, en este capítulo vamos ver en detalle la construcción de un mundo para JdeRobot, en concreto el circuito de Fórmula 1 de Mónaco.

### 2.1. Manejo de Blender

### 2.2. Circuito plano

### 2.3. Circuito con elevaciones



# Bibliografía

- [1] Página oficial de Blender <https://www.blender.org/>
- [2] Página oficial de Gazebo <http://gazebosim.org/>
- [3] Página oficial de JdeRobot [http://jderobot.org/Main\\_Page](http://jderobot.org/Main_Page)
- [4] Página oficial de ARIAC <http://gazebosim.org/ariac>
- [5] Página oficial de ROS <http://www.ros.org/>
- [6] Página oficial de MoveIt <http://moveit.ros.org/>
- [7] Página oficial de rViz <http://wiki.ros.org/rviz>
- [8] Página oficial de rqt <http://wiki.ros.org/rqt>