Capítulo 2

Objetivos

Una vez explicado el contexto de este proyecto, se describirán los objetivos, los requisitos y la metodología que se han empleado.

El propósito principal de este proyecto es la creación de diferentes prácticas para el entorno docente. En concreto una práctica de navegación global, una práctica de una aspiradora robótica y, por último, una práctica acerca del aparcamiento de coches autónomos. Para ello se empleará la plataforma de software libre JdeRobot, que se describirá en el capítulo 3.

* 1. Objetivos

El objetivo principal de este proyecto es la creación de prácticas para el entorno JdeRobot/Academy, que emplearán los alumnos. En cada una de estas prácticas se elaborará toda la infraestructura que se comunica con el Simulador Gazebo, donde el alumno podrá ver el resultado de la ejecución de su algoritmo. Además, se creará el entorno gráfico que facilitará al alumno la resolución de las prácticas; así como un árbitro que mide diferentes parámetros de cada práctica y permite la corrección de cada una de ellas en función de dichos parámetros. Para cada una de las prácticas se ofrece una posible solución. Se ofrecerá, en cada práctica, un fichero MyAlgorithm.py donde el alumno podrá programar su solución.

En este proyecto, como hemos mencionado antes, se llevarán a cabo tres prácticas. Para cada una de estas prácticas se han creado todos los elementos mencionados en el párrafo anterior. Lo que difiere en estas prácticas es el escenario de cada una de ellas, los diferentes elementos que se mostrarán en el interfaz gráfico, los elementos que tendrá en cuenta el árbitro a la hora de calcular la nota de cada alumno, y por último el algoritmo de la solución.

A continuación, comentaremos cual es el objetivo de cada práctica desarrollada en este proyecto. Tal y como se ha dicho se han realizado tres prácticas. Dichas prácticas son:

* Global Navigation
* Vacuum cleaner
* Autopark

En la práctica “Global Navigation” el objetivo es que el alumno pueda programar una solución para el algoritmo de navegación global. Se ofrecerá una solución basada en la técnica Gradient Path Planning para resolver el problema de navegación global.

En la práctica “Vacuum cleaner” el principal objetivo es que el alumno sea capaz de proporcionar una solución para la limpieza de una casa sin autolocalización. En concreto, este algoritmo se basará en el algoritmo que llevan a cabo los modelos de la serie 500 de Roomba de iRobot. Se realizará una solución que limpia en función de tres modos de limpieza: giro en espiral, seguimientos de paredes y cruce de habitación.

En último lugar, en la práctica “Autopark”, el propósito es la realización de una solución que sea capaz de aparcar un coche de forma autónoma. La solución propuesta es una solución “ad hoc” mediante un algoritmo que se propone basado en las medidas sensoriales que se obtienen de los lásers.

La plataforma de desarrollo consta de dos componentes que le permitirán al alumno realizar la solución de las prácticas. Estos componentes están compuestos por una interfaz gráfica (GUI) que proporciona elementos de ayuda a la resolución de las prácticas, y un árbitro, que permite llevar a cabo la corrección de las prácticas. En estas prácticas se harán las conexiones necesarias de los robots con los sensores y actuadores que se empleen. El propósito de los componentes es permitir la abstracción por parte de los alumnos de los elementos complejos que no son parte de la resolución de las prácticas. De esta forma, el alumno lo único que tendrá que hacer es programar su solución de cada algoritmo que se le propone.

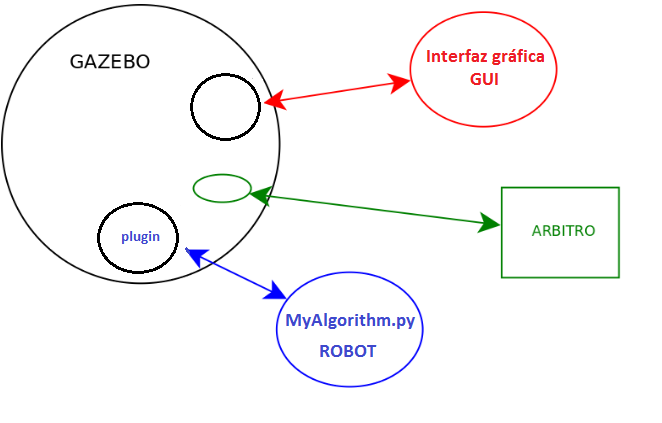


Figure 2.1: Estructura

El interfaz gráfico que se propone en cada práctica, muestra un visor del mapa del escenario donde estará situado cada robot. En este visor, además, se mostrará la situación del robot en este mapa. En este interfaz gráfico, también, se dispone de un teleoperador que puede emplear el alumno cuando lo desee para mover el robot. En la interfaz de la práctica “Autopark” también se mostrará de forma gráfica las lecturas que realizan cada uno de los lásers.

Se crearán los componentes siguiendo una arquitectura software que permite facilitar el desarrollo de las prácticas a los alumnos, los cuales únicamente deberán realizar la solución, ya sea el pilotaje en función de los datos que proporcionan los sensores (es el caso de las prácticas “Vacuum cleaner” y “Autopark”) o la realización de la planificación y el pilotaje (es el caso de “Global navigation”). Los componentes podrán interactuar con el fichero MyAlgorithm.py (donde se lleva a cabo la resolución), mostrando en la interfaz las pruebas o soluciones que realicen los alumnos. En la figura 2.1 podemos ver la estructura que tendrá cada una de las prácticas.

* 1. Requisitos

El desarrollo del proyecto estará guiado por los objetivos mencionados anteriormente y deberá ajustarse a los requisitos de partida del proyecto, los cuales hacen que la solución esté condicionada. Estos requisitos son:

1. En este proyecto todas las simulaciones se realizarán en el simulador Gazebo, en concreto en la versión 7. Los modelos de robots que se emplearán serán creados. En este caso se utilizará un modelo de taxi, otro taxi con otros sensores y el modelo Roomba.
2. Se hará uso de la plataforma JdeRobot en su versión 5.5.2, que se explicará con detalle en el siguiente capítulo. El uso de esta plataforma simplifica el desarrollo del comportamiento del robot.
3. El sistema operativo que se empleará para este proyecto será Ubuntu 16.04.
4. El lenguaje de desarrollo empleado para crear los plugins necesarios será C++. Sin embargo, en el resto de componentes se utilizará el lenguaje Python. Este requisito restringe la solución a que se programe en Python.
5. Las soluciones han de ser vivaces. Los algoritmos propuestos no pueden detenerse mucho tiempo a pensar cuál será el próximo movimiento del robot, porque han de reaccionar rápido, en tiempo real y con movimientos suaves.
   1. Metodología

En este apartado se describirá la metodología empleada para la realización de este proyecto. Este proyecto se ha basado en una metodología iterativa, donde cada iteración se compone de varias fases: determinar objetivos, planificación, diseño e implementación, análisis de riesgos, así como reuniones periódicas con el tutor.

Para desarrollar este proyecto se ha optado por seguir el modelo de desarrollo en espiral, creado por Barry Boehm. Este modelo se adapta perfectamente a este tipo de proyectos, ya que permite separar el comportamiento final en varias subtareas más sencillas para después juntarlas. Este modelo permite una gran flexibilidad ante cambios en los requisitos, algo bastante común en el desarrollo de proyectos.

Este modelo de ciclo de vida nos permite ir obteniendo prototipos funcionales, a la vez que se realiza el desarrollo del producto de forma incremental. Este modelo consta de iteraciones, que se pueden llamar ciclos. En cada ciclo existen cuatro fases bien diferenciadas:

* Determinar o fijar objetivos: En esta fase se definen los objetivos específicos que deben cumplirse para que el ciclo actual pueda considerarse finalizado en base a los objetivos finales. Conforme vayan sucediéndose más iteraciones, los objetivos serán más complejos.
* Análisis del riesgo: En esta fase se efectúa un análisis detallado para cada uno de los riesgos identificado del objetivo fijado en la fase anterior. Se definen los pasos a seguir para minimizar los riesgos y después del análisis se planean estrategias alternativas.
* Desarrollar y probar: En la tercera fase, se desarrolla el producto o las partes del producto que se han acordado en las fases anteriores. Además, se llevarán a cabo las pruebas oportunas que nos permitan asegurar la calidad de la implementación, y que pueda seguir sirviendo en iteraciones futuras.
* Planificación: En esta última fase, es donde se revisan los resultados obtenidos mediante las pruebas de la fase anterior, y es donde se planifica la iteración siguiente teniendo en cuenta los posibles errores que se han cometido.

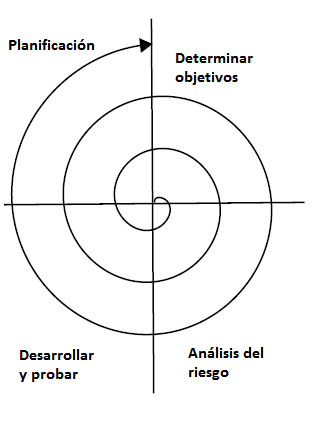


Figure 2.2: Modelo en espiral

Para poder llevar a cabo esta metodología en el proyecto se han mantenido reuniones semanales con el tutor. En estas reuniones se analizaban los resultados de cada iteración, y en función de los resultados se fijaban nuevos objetivos y se planteaban posibles vías para resolver dichos objetivos. El código que se ha ido desarrollando semanalmente se ha subido al repositorio propio de Github [2], que emplea el sistema de control de versiones. Además, las tareas realizadas se han ido mostrando semanalmente mediante explicaciones, vídeos o imágenes en la wiki de la página de JdeRobot [1].

* 1. Plan de trabajo

En esta sección se exponen las etapas en las que se ha dividido el proyecto, que además se corresponden con el modelo en espiral:

* Familiarización con el entorno JdeRobot y OpenCV. En esta etapa se ha descargado e instalado la plataforma JdeRobot, el entorno docente JdeRobot/Academy, y todo el software necesario para el desarrollo del proyecto. En esta fase se engloba el aprendizaje del uso de Github, para el control de versiones, y el aprendizaje básico de la librería OpenCV. Para poder cerrar esta fase se ha realizado algunas soluciones de prácticas del entorno JdeRobot/Academy.
* Familiarización con el simulador Gazebo. En esta etapa se ha estudiado código de la plataforma JdeRobot, así como el material disponible en Gazebo en la página oficial [3]. Además, se han realizado pruebas creando mundos simples en Gazebo mediante modelos ya disponibles.
* Familiarización con los plugins de Gazebo. Esta etapa conlleva el estudio de los plugins creados en JdeRobot (compilación e instalación), aprendizaje básico de C++, y desarrollo de algún plugin necesario para el desarrollo de las prácticas.
* Desarrollo de mundos y modelos necesarios para las prácticas. Se han creado los modelos necesarios para cada práctica, y con dichos modelos se ha podido crear los mundos que serán empleados en las prácticas.
* Desarrollo de la interfaz gráfica. Ha sido necesario familiarizarse con la herramienta PyQt5. En esta etapa se ha programado la aplicación gráfica que facilita la resolución de las prácticas a los alumnos.
* Desarrollo del árbitro. Al igual que en la fase anterior se ha empleado PyQt5. Se ha desarrollado, en esta etapa, el árbitro que permitirá calificar las prácticas de los alumnos.
* Desarrollo de la solución. En esta etapa, se ha programado el algoritmo pertinente para cada práctica, ya sea una solución que conlleva planificación más pilotaje, o una solución que realiza el pilotaje en base a los datos proporcionados por los sensores.

Bibliografía

<http://www.ojovisual.net/galofarino/modeloespiral.pdf>

<http://modeloespiral.blogspot.com.es/>

<https://prezi.com/y7slahvparel/modelo-en-espiral-y-modelo-basado-en-prototipos-para-el-desarrollo-de-software/>

<https://es.slideshare.net/soniaposligua/modelo-enespiral>

[1] Wiki JdeRobot. <http://jderobot.org/Vmartinezf-tfg>

[2] Repositorio Github. <https://github.com/RoboticsURJC-students/2016-tfg-vanessa-fernandez>

[3] Gazebo. <http://gazebosim.org/>