ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA EN INFORMÁTICA

Área de Ciencia de la Computación e Inteligencia Artificial

IMPLEMENTACIÓN DE ARQUITECTURA DE CONTROL 3T EN LA ROBOCUP SMALL SIZE LEAGUE: FORMACIÓN DE ATAQUE

Alumna: Verónica Raspeño Gutiérrez

Tutor: José Antonio Iglesias Martínez

Director: Agapito Ismael Ledezma Espino

Leganés, julio de 2012

AGRADECIMIENTOS

Este proyecto ha sido resultado del trabajo y esfuerzo a lo largo de estos meses, logrado también gracias a aquellas personas que han estado a mi lado.

Lo primero, quiero agradecer este proyecto a mis padres, que confiaron en mí cuando yo misma no lo hacía, que no han dudado nunca de mí y que siempre me recordaron lo orgullosos que estaban.

A mi hermana, Noemí, y a mi cuñado, Hilario, por ser mis compañeros de viaje en todas nuestras aventuras y darme la posibilidad de tener en mi vida al pequeño Izan.

Gracias a mi director y mi tutor de proyecto, Agapito y José Antonio, por permitirme poder realizar un trabajo interesante que fuera rico para mi educación y mi carrera. También al equipo Parsian Robotic de Irán, por dejarme participar en su proyecto. Asimismo, a Isaac, por ayudarme en todo lo relacionado con los robots físicos y por tener buen equilibrio para sujetar la cámara, aun a riesgo de caerse.

No quiero olvidarme de mis compañeros y amigos, sobre todo de mi grupo de Sistemas Informáticos, Álex, Fran, Javi y Luis, por todas las actividades que hemos hecho juntos, por dejarme ser "la jefa" y por tener mucha paciencia conmigo. A Roberto, por escucharme y, lo más difícil, comprenderme. A Samu, por tener la habilidad de hacerme reír. También al resto de mis compañeros, por todas las conversaciones, comidas y risas compartidas.

Por último, a mi compañero de proyecto, Álex, no solo eso, ya parte de mi vida y de mi familia, cuando se me ocurría una idea, por muy descabellada que fuera, siempre me preguntaba dónde y cuándo sin dudar. Gracias por hacer las horas, minutos y por ayudarme a sacar adelante este proyecto.

RESUMEN

La robótica es un campo en pleno auge, de actualidad y de interés general, además de ser un tema en desarrollo e investigación. En concreto, la aplicación de robots en el fútbol es una actividad completa, ya que engloba diferentes destrezas positivas para aplicar en un robot útil para las personas, como, por ejemplo, trabajo en equipo, estrategia, decisiones, etc.

Este proyecto aborda el tema de la implementación de una arquitectura de control 3T para la Small Size League en la competición RoboCup. Se trata de diseñar e implementar una arquitectura útil para esta competición y, tras ello, conectarlo a un entorno real, como es un simulador o, incluso, robots físicos que respeten las condiciones de esta liga.

El diseño de esta arquitectura, así como la implementación práctica, tiene como objetivo establecer los cimientos para formar un equipo completo y habilitado para participar en la Small Size League de la competición RoboCup. Así, se podrá ampliar con facilidad, con la incorporación de nuevas habilidades y desarrollando la parte de estrategia y planificación, dotando a los robots de mayor inteligencia.

ABSTRACT

The robotics is an important and interesting field in technological sciences; moreover, it's a topic in developing and research. In specific, the robotics applied in football is an interesting and complete activity, because that includes different positive skills to apply in robots for people, for example, teamwork, decisions, strategies, etc.

This project approaches the implementation of control architecture 3T for the Small Size League in the RoboCup's competition. This architecture must be designed and implemented in a useful way for this competition and, then, it's connected to a real environment, such as a simulator or physical robots that follow the rules in this league.

The design of the architecture and the practical implementation have the objective to lay the foundations to make up a complete and enabled team to participate in the Small Size League of RoboCup competition. So, this could extend easily, with incorporation of new skills and developing the part of strategy and making robots more *intelligent*.

Índice de Contenido

Índice de Ilustraciones

Índice de Tablas

INTRODUCCIÓN

1. Introducción y motivación

A lo largo de los años la robótica ha ido evolucionando y transformándose, siendo un campo que en la actualidad es de interés general. Esta rama de la ingeniería y de la informática es un campo joven y se encuentra en la actualidad en pleno desarrollo e investigación. Por eso mismo, existen distintas competiciones y concursos que intentan realizar un progreso en esta área.

En los últimos veinte años han surgido nuevos descubrimientos y avances que plasman un punto de inflexión en lo que se entendía hasta el momento por robótica. Uno de los sucesos más destacables en la década de los 90 es en mayo de 1997, cuando el ordenador *Deep Blue* de IBM consigue derrotar al campeón del mundo de ajedrez, Gary Kaspárov [30].

Este mismo año surge la competición de robots que juegan al fútbol, la RoboCup [17], con la intención de conseguir que en el año 2050 un equipo de robots sean capaces de ganar al equipo humano campeón del Mundial. Actualmente, la competición se ha ampliado a otros ámbitos, como robots de asistencia para emergencias o para ayuda en el hogar. Dentro del ámbito del fútbol, existen distintas ligas: humanoides, tamaño pequeño (SSL [9]), tamaño mediano plataforma estándar y simulación.

Este es el contexto del presente PFC, que se enfoca en la creación y desarrollo de la arquitectura de un equipo de fútbol completo de la liga *Small Size League* [9] (a partir de aquí SSL, por su acrónimo en inglés) de la competición mundial RoboCup [17].

Este PFC comienza con las bases anteriores de la arquitectura híbrida 3T [36] programada en lenguaje de programación Java [18], elaborada por un compañero de la Universidad Carlos III de Madrid, José Luis Díaz Cebrián, en su PFC "Arquitectura de control híbrida para robots autónomos" [31]. Este trabajo detalla el diseño y la implementación de esta arquitectura híbrida, en concreto para el robot Pioneer 3-DX. Por eso mismo, parte de este PFC incluye la realización de la adaptación e integración de esta arquitectura a las exigencias del equipo de fútbol elaborado.

Tras la adaptación, se ha continuado con el desarrollo de las habilidades básicas del comportamiento de los robots de fútbol. Estas habilidades son complementarias con otras elaboradas en otro PFC existente que será publicado próximamente, perteneciente a un compañero de la Universidad Carlos III de Madrid, Alejandro Caparrós Hernando [32].

La creación de las pruebas del comportamiento de los robots y del pleno funcionamiento de la arquitectura se ha plasmado a través de un simulador de SSL [9], desarrollado por el equipo Parsian Robotic [7].

Todo el trabajo realizado con el simulador grSim [7] [8] ha sido compartido con el equipo creador Parsian Robotic [7], pudiendo colaborar con un proyecto real y en desarrollo, ayudando asimismo a aportar documentación, vídeos demostrativos y los clientes en Java [18], tanto de recepción como de envío. Previamente a la colaboración realizada, no disponían de documentación, ni de clientes en el lenguaje de programación Java [18] y se ha realizado como parte de este PFC.

Por último, para completar el desarrollo de este PFC y obtener el ciclo completo de producción de un equipo de fútbol de esta liga, se han realizado pruebas con robots físicos. Este robot ha sido llevado a cabo, en su primer prototipo, por el compañero de la Universidad Carlos III de Madrid Álvaro García López en su PFC "Desarrollo de Sistema de Locomoción de una plataforma hardware par RoboCup Small Soccer League (SSL)" [33] y, en su segundo prototipo, por dos PFCs de dos compañeros de la Universidad Carlos III de Madrid por el Departamento de Ingeniería de Sistemas y Automática, por un lado Lidia Escudero Jiménez en "Sistema de Procesamiento, alimentación y estructura de un microrobot" [34] y, por otro lado, José Luis Martín Gómez en "Diseño del Sistema de Locomoción y *dribbler* de un microrobot" [35].

1.1. Objetivos

El objetivo principal de este proyecto es elaborar la arquitectura de un equipo de fútbol de la liga SSL [9] de la RoboCup [17], definiendo las bases para poder formar un grupo de robots que se presente a la competición oficial.

La creación de esta arquitectura y las habilidades básicas de los robots tiene como objetivo establecer las bases para, en el futuro, poder desarrollar una ampliación para que el equipo al completo se comporte como tal, siendo capaz de hacer estrategias, toma decisiones y actividades colaborativas.

La arquitectura que se propone debe ser capaz de permitir a un robot realizar determinadas habilidades básicas. De esta manera los robots en el simulador utilizado podrán realizar movimientos lógicos propios de un jugador en un partido de fútbol.

Para alcanzar este objetivo general, es necesario realizar una serie de objetivos específicos: el primero de ellos consiste en realizar una integración plena de la arquitectura ya existente en el PFC "Arquitectura de control híbrida para robots autónomos" [31], para poder unificar el trabajo anteriormente publicado con lo creado en este trabajo.

Tras ello, el segundo objetivo específico es lograr una plena combinación del simulador utilizado, grSim [7] [8], y la arquitectura elaborada, para que sean capaces de comunicarse entre ellos. Además, se deben realizar las modificaciones necesarias en el simulador [7] [8] para que pueda adaptarse a los robots físicos existentes y sea sencillo, posteriormente, la fase de pruebas.

También es importante destacar que este trabajo consigue ser ampliado gracias al PFC de Alejandro Caparrós Hernando, y que ha sido elaborado de forma simultánea a este. Esto permite que se tenga un mayor número de habilidades con las que realizar pruebas y combinaciones para la simulación.

Asimismo, un objetivo adicional que logra ampliar el presente PFC es utilizar la arquitectura elaborada en robots físicos que demuestran el funcionamiento y futura

aplicación en un equipo real de robots. Así, la arquitectura debe ser adaptada a los robots físicos.

Finalmente, se puede realizar un resumen esquemático de los objetivos principales de este PFC:

- **Objetivo principal:** creación de una arquitectura para la liga SSL [9] de la competición RoboCup [17] con ciertas habilidades básicas que plasmen las bases para la elaboración de un equipo de esta liga.
    - **Subjetivo 1:** adaptación de arquitectura 3T [36] del PFC "Arquitectura de control híbrida para robots autónomos" [31].
    - **Subjetivo 2:** adaptación del simulador grSim [7] [8] para realizar las comunicaciones pertinentes entre el simulador y la arquitectura.
    - **Subjetivo 3:** modificaciones del simulador grSim [7] [8], adaptándolo a las exigencias del robot físico elaborados por los PFCs citados anteriormente [33] [34] [35].
    - **Subjetivo 4:** funcionamiento pleno de habilidades básicas en robots simulados por el simulador grSim [7] [8].
    - **Subjetivo 5:** funcionamiento de habilidades básicas en robots físicos.

1.2. Fases del Desarrollo

En el presente trabajo existen cinco fases del desarrollo diferenciadas que se van a detallar a continuación.

La primera fase corresponde a la documentación e investigación sobre el tema a tratar. Este bloque ha consistido meramente en el asentamiento de las bases del conocimiento para, posteriormente, poder realizar las siguientes fases. La documentación se ha basado en adquirir información global sobre la competición RoboCup [17], así como ahondar en la liga en la que se enmarca este PFC: SSL [9]. Por

otro lado, se han buscado los posibles simuladores para realizar la simulación del comportamiento de los robots, que plasmará exactamente las exigencias de la SSL [9]. Además, se ha indagado sobre las distintas arquitecturas y paradigmas de la robótica para conocer las posibles alternativas que existen.

La segunda fase corresponde a la fase de análisis, en donde se ha realizado un estudio del trabajo a realizar, definiendo qué había que hacer. Debido a que este PFC abarca muchas materias y van apareciendo nuevas tareas a realizar una vez que se está haciendo, es necesario analizar con minuciosidad todo el proceso.

La tercera fase equivale al diseño de la arquitectura, detallando el nivel arquitectónico y el detallado. Este trabajo tiene distintas vertientes que se unifican posteriormente, por lo que el diseño se ha hecho de todas ellas. Esto se debe a que, por un lado, existe el trabajo desarrollado de cara a adaptar la arquitectura 3T [36] ya existente y, por otro lado, todo el trabajo de adaptación del simulador grSim [7] [8] a los robots físicos y a la arquitectura. La unificación de estas dos tareas da como resultado las bases para poder elaborar las habilidades básicas propias de un robot de la SSL [9].

La cuarta fase equivale a la implementación del sistema. Como se ha descrito en los párrafos anteriores, tras realizar un análisis y diseño del trabajo a realizar, se procedió a llevarlo a cabo. Para eso, de nuevo aparecen dos áreas diferenciadas, por un lado la integración y modificación de la arquitectura 3T [36] para la adaptación a los robots de la SSL [9] y, por otro lado, la adaptación del simulador grSim [7] [8]. Una vez que estas dos áreas se unifican, siendo la arquitectura capaz de comunicarse con el simulador grSim [7] [8], se procede a realizar las habilidades básicas del robot.

La quinta fase corresponde a la fase de pruebas, en donde se ha corroborado que los robots realizan las habilidades programadas. Para ello, se ha probado con el simulador grSim [7] [8] y robots simulados. Además, y a modo de ampliación de este PFC, se ha probado con robots físicos.

Por último, a pesar de que se ha ido realizando en cada una de las fases de forma incremental, la sexta fase consiste en la creación de la presente memoria, que incluye la recopilación de toda la documentación obtenida.

1.3. Medios Empleados

En este punto se van a desarrollar los medios empleados a lo largo de este proyecto, detallando el software y el hardware utilizado.

1.3.1. Hardware

Se detallará todo el hardware que se ha utilizado a lo largo de la creación de este PFC, detallando el ordenador portátil, los robots físicos y la cámara para el sistema de visión:

- **Ordenador portátil:** se ha utilizado un ordenador portátil para el desarrollo de la presente memoria, así como para la elaboración de todas las fases del desarrollo, incluida la documentación, el análisis, el diseño, la implementación y las pruebas. El ordenador que se ha usado ha sido un Sony Vaio VPCS13C5E de 13'3 pulgadas, 8 GB de RAM y procesador Intel® Core™ i7 [14].

Ilustración 1: Sony Vaio VPCS13C5E [14]

- **Robots físicos:** robots creados en su primer y segundo prototipo por el departamento de Ingeniería de Sistemas y Automática, en el Laboratorio de Sistemas Inteligentes de la Universidad Carlos III de Madrid [33] [34] [35]. Las especificaciones se detallarán con mayor exactitud en el apartado "3.6.1. Especificaciones del robot físico", aunque coinciden con las normas de la liga SSL [9].

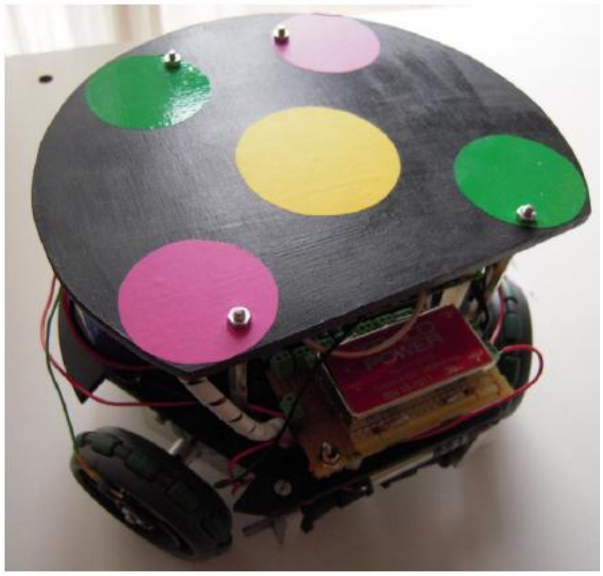


Ilustración 2: robot físico perteneciente a la Universidad Carlos III de Madrid

- **Cámara AVT Stingray**: para la realización del sistema de visión de la SSL [9] es necesario disponer de una cámara. Esta cámara se encarga de captar todo el movimiento que se produce en el terreno de juego y almacenar las posiciones de todos los robots y de la pelota. Periódicamente, envía paquetes con esta información al ordenador que ejecuta la arquitectura de los robots, para que esta conozca la posición de todos los robots, así como de la pelota. [16]

Ilustración 3: cámara Stingray [16] usada para el sistema de visión de la SSL [9]

1.3.2. Software

Se detallará todo el software que se ha utilizado a lo largo de este PFC:

- **GIMP:** es un programa de edición de imágenes que se ha utilizado para realizar todas las ilustraciones y representaciones que se ofrecen a lo largo del presente documento. Este programa permite realizar el tratamiento de imágenes para representar la información relacionada con el simulador grSim [7] [8] y así mostrar de forma gráfica lo que se cita textualmente.

  Se ha usado este programa de edición de imágenes porque es de software libre, gratuito y se puede utilizar en distintos sistemas operativos, como Windows y Ubuntu.

Ilustración 4: logotipo de GIMP [23]

- **GrSim:** simulador que simula el terreno de juego de la competición SSL [9], así como los robots y su comportamiento. Permite plasmar los movimientos producidos por los robots en la competición SSL [9]. [7] [8]

  Se puede consultar un manual de usuario, con toda la información de instalación de este simulador en el "Anexo III: Manual de Usuario".

Ilustración 5: logotipo del equipo desarrollador del simulador grSim [7] [8]

- **Máquina Virtual de Java:** permite ejecutar el código Java [18] creado para representar el comportamiento y habilidades de los robots. [18] Se ha utilizado este lenguaje de programación debido a que se partía de una arquitectura híbrida ya existente y programada en este lenguaje [31] y, además, es posible usarlo independientemente del sistema operativo que se utilice.

Ilustración 6: logotipo de Java [18]

- **Microsoft Office 2010:** suite de oficina utilizada para realizar la documentación de este proyecto. Dentro de esta suite se ha usado:

- **Microsoft Excel 2010:** programa utilizado para realizar las gráficas que aparecen a lo largo de este documento. Permiten mostrar un conjunto de datos de forma clara, ayudando a su fácil lectura.
- **Microsoft PowerPoint 2010:** programa utilizado para la presentación realizada en la presentación pública de este PFC.
- **Microsoft Word 2010:** programa usado para la creación del presente documento.
- **Microsoft Project 2010:** programa utilizado para la creación de los diagramas de Gantt incluidos en la presente memoria.

Se ha escogido esta suite de ofimática debido a su completitud, sus posibilidades y su sencillo e intuitivo uso.

Ilustración 7: logotipo Microsoft Office 2010 [24]

- **Netbeans:** entorno de desarrollo que se ha utilizado para ejecutar el código desarrollado durante este proyecto en el lenguaje de programación Java [18].

Este programa ha permitido probar y conocer los resultados del código desarrollado, así como enviar la información al simulador y a los robots físicos para ver la respuesta que estos ofrecían. [25]

La decisión del uso de este entorno de desarrollo ha sido porque es gratuito, de software libre y está hecho para la programación en lenguaje de programación Java [18].

Ilustración 8: logotipo Netbeans [25]

- **SSL-Vision:** sistema de visión usado para las pruebas con robots físicos. Permite enviar la información sobre los robots al ordenador remoto y así poder recibir el paquete correspondiente con la posición de los robots y la pelota. Es utilizado por toda la comunidad participante en la SSL. [11]

Ilustración 9: logotipo de SSL-Vision [11]

- **Ubuntu:** sistema operativo utilizado para la ejecución del simulador grSim [7] [8], ya que este solo funciona sobre sistemas operativos Unix.

Este sistema operativo se ha usado para ejecutar el código realizado durante este PFC, así como para todo lo realizado con el simulador grSim [7] [8].

El simulador grSim [7] [8] requiere una distribución Linux, se ha utilizado Ubuntu por la recomendación directa de los creadores de este simulador, así como su simplicidad y facilidad de uso. Se podría haber usado cualquier otra distribución de Linux, ya que grSim [7] [8] es compatible.

Ilustración 10: logotipo Ubuntu [26]

- **Windows 7:** sistema operativo utilizado para la realización del trabajo relacionado con la documentación de este PFC. Se ha usado meramente para la documentación, ya que esta ha sido realizada por una suite propia de este sistema operativo. [27]

  La elección de este sistema operativo para la documentación es debido a que el software de ofimática que ofrece es completo e incluye herramientas útiles que facilitan la elaboración de esta. Por ejemplo, la creación de diagramas de Gantt o las herramientas de diseño.

Ilustración 11: logotipo Windows 7 [27]

- **Wireshark:** analizador de protocolos utilizado para captar los paquetes enviados al simulador grSim [7] [8] y recibidos por este. Se ha utilizado este software por ser un programa gratuito y fácil de utilizar. [61]

Ilustración 12: logotipo Wireshark [61]

1.4. Estructura del documento

En esta sección se incluirá una descripción genérica de la estructura del documento, detallando cada capítulo que se definirá en el presente documento.

En el primer capítulo se ha introducido una visión global de este documento, su estructura, las fases de desarrollo, los medios empleados, así como los objetivos a cumplir en este trabajo y la estructura.

Seguidamente, en el segundo capítulo se va a proceder a definir el estado del arte, en el que se realizará una visión general de toda la base teórica establecida para la creación de este PFC. Para ello se comentará el concepto de robótica y su evolución, los paradigmas de la robótica existentes en la actualidad, qué es la RoboCup [17] y sus distintas competiciones y ligas, además de los simuladores más conocidos de la liga SSL [9].

En el siguiente capítulo se describirá el desarrollo técnico de este trabajo, comentando las distintas ramas que incluye. Para ello, primero se introducirán los requisitos de usuario del trabajo elaborado, para después comentar las características del simulador grSim [7] [8], el cliente Java [18] de envío y recepción del simulador, la utilización de la arquitectura 3T [36] para el desarrollo de las habilidades para la SSL [9], las habilidades desarrolladas para los robots y, finalmente, la aplicación realizada en los robots físicos.

A continuación se incluirá un capítulo detallando los experimentos realizados, marcando la diferencia entre los experimentos con el simulador [7] [8] y los experimentos con los robots físicos, comentando al final la comparativa entre las dos pruebas y los resultados aportados.

Posteriormente, se aportarán las conclusiones y futuras líneas de trabajo añadidas por el presente PFC, resaltando las diferencias que puede haber entre los robots físicos y los simulados y cómo se podría ampliar este trabajo.

Tras ello, se añadirán todas las referencias consultadas, que amplían la información citada y que permiten revisar todo lo consultado para la realización de este trabajo.

Por último, se ofrecen distintos anexos que ayudan a explicar el contenido de este documento. El primer anexo corresponde a los acrónimos y definiciones utilizados en toda la memoria, que ayudan a explicar y aclarar el contenido. El segundo anexo corresponde a la gestión del proyecto, que incluye las fases que se han desarrollado, el ciclo de vida, un diagrama de Gantt con toda esta información y el presupuesto. El tercer

y cuarto anexos equivalen al manual de usuario y manual de referencia que explican el pleno funcionamiento del trabajo desarrollado. El quinto anexo detalla el reglamento del año 2011 para la liga SSL [9] para la competición RoboCup [17], en el cual se ha basado este PFC y, por último, el sexo anexo describe los cambios que ha sufrido el reglamento en el año 2012.

ESTADO DEL ARTE

2. Estado del Arte

A lo largo de este capítulo se va a mostrar la base teórica existente de la que parte este proyecto, asimismo, se realizará un repaso de las técnicas relacionadas y las aportaciones que han servido como guía y ayuda.

Para ello, es necesario hacer una breve introducción de la robótica para conocer el estado actual de este campo, siguiendo con la arquitectura implementada y, por último, toda la información relacionada con la RoboCup [17].

2.1. Robótica

Basándose en el diccionario de la RAE, se define robótica como la técnica que aplica la informática al diseño y empleo de aparatos que, en sustitución de personas, realizan operaciones o trabajos, por lo general en instalaciones industriales [1].

Es en 1920 la primera vez que se utiliza la palabra "robot", por el autor Karel Capek en una de sus obras de teatro titulada R.U.R. [2]. Asegura que el término proviene de una palabra checa, *robota*, cuyo significado en español es trabajo. Indica que el robot es similar a un esclavo o un trabajador forzado a hacer una actividad para los seres humanos. Los robots que se describen en esta obra son imitaciones de humanos que trabajan en una fábrica.

En la actualidad, se utiliza el vocablo androide para referirse a humanos artificiales, mientras que robot, según la RAE, se define como aquella máquina o ingenio electrónico programable, capaz de manipular objetos y realizar operaciones antes reservadas solo para las personas [1].

2.2. Paradigmas en la Robótica

Existen tres paradigmas diferenciados que se encargan de organizar la inteligencia y el comportamiento de los robots. Estos son: paradigma jerárquico, paradigma reactivo y paradigma híbrido deliberativo/reactivo.

2.2.1. Paradigma Jerárquico

El paradigma jerárquico es el paradigma más antiguo, naciendo en 1967. A lo largo de esta década se desarrolla y es en los años 70 cuando tiene mayor importancia en la robótica.

Su nombre viene dado porque está formado por tres actividades (percibir, planificar y actuar) que realiza de forma secuencial o jerárquica.

Su esquema básico es el siguiente:

Ilustración 13: diagrama de Paradigma Jerárquica

Por lo tanto, la forma de actuar de este paradigma es la siguiente: en PERCIBIR, el robot recibe los datos de los sensores y devuelve la información percibida, que le llega a PLANIFICAR, que ejecuta las directivas y se las pasa a ACTUAR, allí donde se definen los comandos de los actuadores. [10]

2.2.2. Paradigma Reactivo

Este paradigma surge en 1988 a partir del paradigma jerárquico y fue muy usado en los años 90, proponiendo soluciones al paradigma jerárquico.

Está basado en las ciencias cognitivas y en la biología, basándose en la planificación y en sus correspondientes respuestas. Su esquema básico se muestra a continuación:

Ilustración 14: diagrama de Paradigma Reactivo

La información entra a través de PERCIBIR por los datos de los sensores y da como salida la información percibida, en ACTUAR se elaboran los comandos de los actuadores. [10]

2.2.3. Paradigma Híbrido

Este paradigma surge al intentar combinar los anteriores paradigmas, el jerárquico y el reactivo, en 1990. El objetivo es ofrecer las mejores partes de cada uno, constando de una parte deliberativa, con un planificador, pero ofrecer una parte reactiva a bajo nivel:

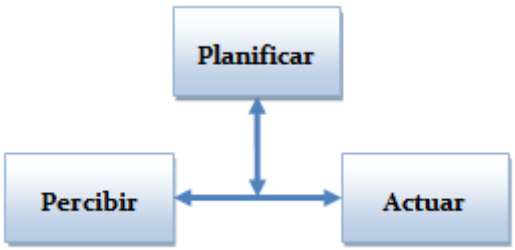


Ilustración 15: diagrama de Paradigma Híbrido

La entrada por PLANIFICAR viene dada por la información percibida y/o cognitiva, sacando como salida las directivas, que las recibe PERCIBIR-ACTUAR y con los datos de los sensores, saca la salida con los comandos de los actuadores. [10]

2.3. Arquitectura 3T

La arquitectura 3T [36] nace en el año 1997 de la mano de Bonasso, con el objetivo de funcionar en diferentes robots, independientemente del hardware que tuvieran y en entornos dinámicos.

El esquema básico que describe esta arquitectura es el siguiente:

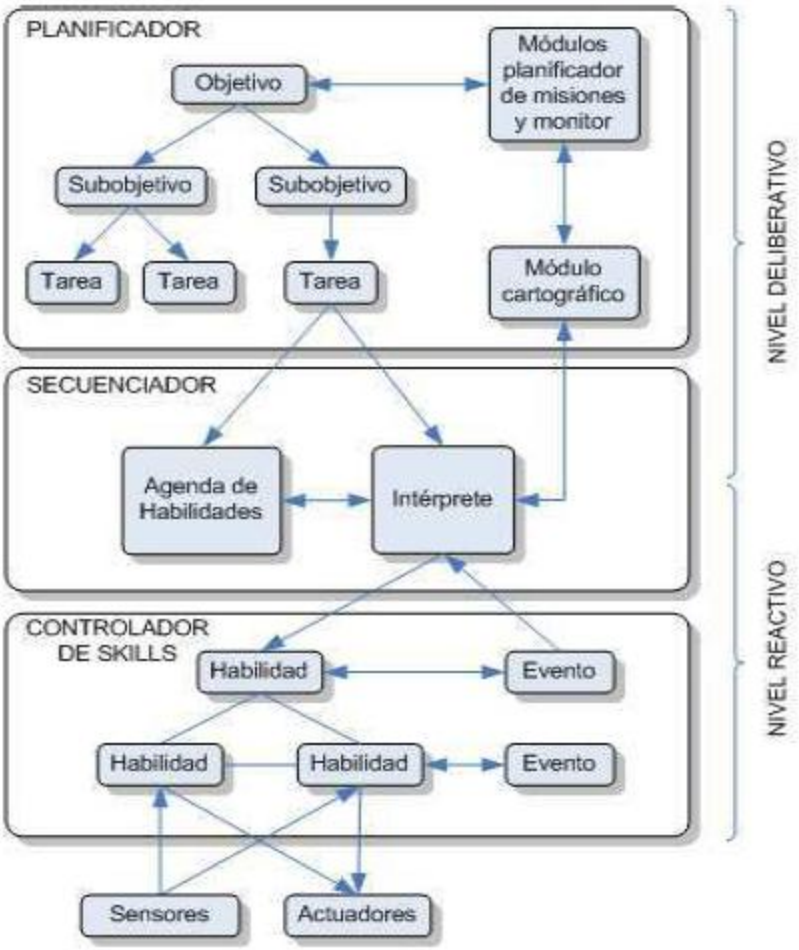


Ilustración 16: arquitectura 3T [31] [36] [38]

La parte superior pertenece al planificador, este se encarga de elegir los objetivos que luego se procederán a descomponer en el siguiente nivel. El secuenciador se basa en la división de actividades en más pequeñas de bajo nivel, llamadas acciones. Finalmente, la última cala, la del controlador, descompone las acciones en habilidades.

Esta arquitectura sigue el paradigma híbrido, en donde el planificador ejerce de capa deliberativa y el controlador equivale a la capa reactiva. La capa intermedia o secuenciador se comunica con ambas. [38]

2.4. RoboCup

La RoboCup [17] es una competición sobre robots que juegan al fútbol que nace en 1997. El objetivo de este proyecto mundial es promover, a través de las competencias integradas por robots autónomos, la investigación y educación sobre inteligencia artificial:

Ilustración 17: logotipo de la competición RoboCup [17]

Aunque esta competición comienza en 1997, ya anteriormente la idea empieza a tomar forma. En el año 1992, en el artículo *On Seeing Robots* [28] escrito por Alan Mackworth de la UBC de Vancouver (Canadá) se sugiere la posibilidad de crear un equipo de fútbol de robots que sean capaces de jugar un partido completo. En este mismo año, en el mes de octubre, unos investigadores japoneses organizaron un taller sobre los grandes retos que tenía que afrontar la inteligencia artificial, fue aquí donde se expuso la posibilidad de crear un equipo de fútbol de robots, por su completitud de

actividades: motricidad, movimientos, trabajo en equipo, colaboración entre robots, etc. [29]

A partir de este momento, este grupo de investigadores japoneses comienzan un estudio de viabilidad ante la posibilidad de realizar un equipo de fútbol de robots. Se incluye un estudio de viabilidad de la tecnología, el impacto social y un estudio financiero. Se realizaron prototipos de robots que pudieran jugar al fútbol y simulaciones. En junio de 1993, se concluye que el proyecto puede llegar a realizarse y un grupo de investigadores deciden comenzar con esta competición. El nombre que recibe provisionalmente es J-League.

Tras esto, los organizadores de la J-League comienzan a difundir esta competición de forma mundial. Tras el éxito y la acogida de distintos sectores de todos los continentes, se renombra el proyecto y pasa a llamarse RoboCup [17], por sus siglas en inglés "La Iniciativa de la Copa Robótica Mundial de Fútbol". El comienzo de esta competición consiguió unificar estudios paralelos que se habían dado, como los estudios de la profesora Manuela Veloso y su estudiante Peter Stone de la Universidad Carnegie Mellon o el profesor Minoru Asada de la Universidad de Osaka.

A partir de este momento, con la idea hecha pública y ciertos investigadores interesados, se comienzan a realizar conferencias y talleres a lo largo de todo el mundo, discutiendo las opciones y desarrollando todas las posibilidades. Después de esto, en 1995 se decide conceder dos años de preparación a los equipos interesados para que los posibles participantes pudieran crear y programar los robots.

El primer año de su celebración, en 1997, tuvo una excepcional acogida, participaron 40 equipos entre robots simulados y físicos y asistieron 5000 espectadores. [17]

Esta iniciativa científica internacional pretende avanzar en el desarrollo de los robots inteligentes. Cuando comienza, el objetivo es formar un equipo de robots capaces de ganar contra el equipo campeón en el Mundial de 2050. Aunque la misión sigue

siendo la misma, los dominios de esta competición han aumentado, desarrollando nuevas competiciones con otras habilidades [3].

Actualmente, y tras 14 años celebrándose, cada vez hay más países y equipos interesados en participar y cada año se presentan más candidaturas. Han surgido más competiciones y otras han sido modificadas, quedando finalmente cuatro: RoboCupSoccer, RoboCupRescue, RoboCup@Home y RoboCupJunior. El objetivo actualmente de las cuatro competiciones es desarrollar la inteligencia artificial, ahondando en el comportamiento inteligente, las estrategias independientes y en grupo, el aprendizaje y la toma de decisiones.

2.4.1. Competiciones

Dentro de la RoboCup [17] han surgido distintas competiciones que se desarrollan a continuación:

- **RoboCupSoccer:** consiste en la creación de equipos completamente autónomos, siendo robots que cooperen entre ellos y muestren comportamientos avanzados, como, por ejemplo, competitividad y estrategia para ganar el partido.

Ilustración 18: ejemplo de competición RoboCupSoccer [17]

- **RoboCupRescue:** es la competición orientada a la asistencia de una situación de emergencia. Pretende definir robots capaces de salvar a personas y realizar tareas peligrosas para el rescate o el análisis de terrenos complejos.

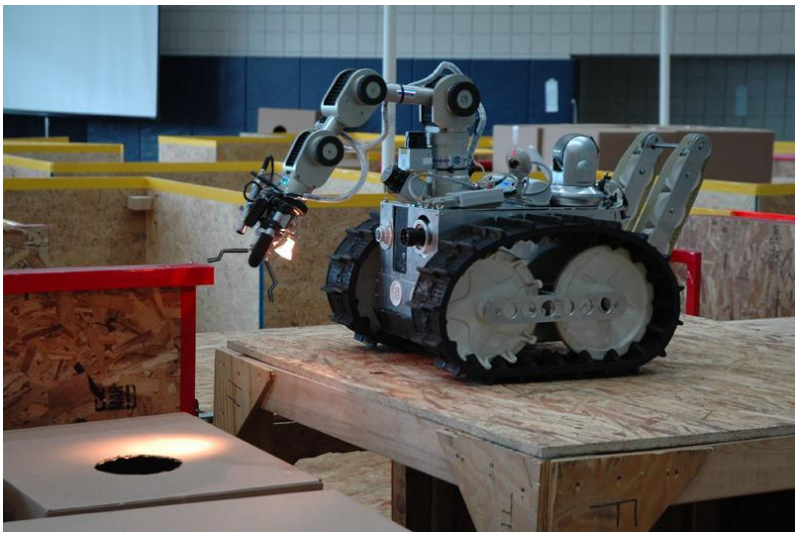


Ilustración 19: ejemplo de competición RoboCupRescue [17]

- **RoboCup@Home:** participan robots que intentan ayudar a las personas en su vida cotidiana en el hogar. Estos robots tienen que demostrar habilidades interactivas y comunicativas.

Ilustración 20: ejemplo de competición RoboCup@Home [17]

- **RoboCupJunior:** pretende motivar a jóvenes y adolescentes a la creación de robots sencillos. Se usan conocimientos relacionados con la ciencia, tecnología, ingeniería y matemáticas para resolver un problema. [3]

Ilustración 21: ejemplo de competición RoboCupJunior [17]

2.4.2. Ligas Existentes

Las ligas existentes se describen en este punto:

- **Humanoides:** es la única liga con robots con aspecto humano. Se desarrolla tanto el aspecto físico (caminar, correr, golpear la pelota, equilibrio, etc.) como el comportamiento (auto-localización, estrategia, etc.). Existen tres subligas de acuerdo con el tamaño:
    - SSL (tamaño pequeño).
    - Tamaño mediano.
    - Tamaño grande.

Ilustración 22: ejemplo de liga de humanoides de tamaño pequeño de RoboCup [17]

- **SSL (tamaño pequeño):** esta liga se basa en los robots de tamaño pequeño, es una de las más antiguas y está orientada a perfeccionar y resolver el problema de la cooperación multi-robot y el control en un entorno dinámico. Esta liga se detallará en el próximo subapartado, "2.4.3. Small Size League", debido a que es la liga en la que se centra este PFC.

Ilustración 23: ejemplo de liga de tamaño pequeño de RoboCup [17]

- **Tamaño mediano:** son robots físicos de 50 centímetros y es la liga más parecida al fútbol real. Esta liga está orientada a definir la plena autonomía de los robots, a la cooperación entre ellos y a mejorar los niveles de percepción del entorno.

Ilustración 24: ejemplo de liga de tamaño mediano de RoboCup [17]

- **Plataforma estándar:** liga en donde todos los equipos utilizan los mismos robots físicos de acuerdo con un estándar establecido. Los equipos trabajan en el software y en el compartimiento, obviando las mejoras relacionadas con la parte física. Actualmente se utilizan los robots Nao Aldebaran, aunque antiguamente se utilizaban los desaparecidos robots AIBO de Sony [14].

Ilustración 25: ejemplo de liga de plataforma estándar de RoboCup [17]

- **Simulación:** es una de las más antiguas ligas y está orientada a mejorar la estrategia del equipo. No hay robots físicos y todo se hace a través de un campo virtual. Existen dos subligas dentro de la simulación:
    - Simulación 2D.
    - Simulación 3D. [3]

Ilustración 26: ejemplo de liga de simulación 2D de RoboCup [17]

Ilustración 27: ejemplo de liga de simulación 3D de RoboCup [17]

2.4.3. Small Size League

La SSL es una liga de la RoboCup [17] en la que utiliza robots de tamaño pequeño, aunque también es popularmente conocida como la F180. Se centra en el problema de inteligencia artificial basado en multi-agentes corporativos y en el control de un entorno dinámico con un sistema centralizado.

Una cámara aérea comunica al ordenador las imágenes del campo de juego que, a su vez, transmite a los robots qué tienen que hacer en función de la situación que reciban.

El esquema que representa esta información se muestra a continuación, la cámara se muestra en la parte superior, el cable que va al ordenador representa la comunicación, que se realiza por UDP. El ordenador recibe la información de la colocación de los robots y la pelota en un paquete y elabora a sí los movimientos que quiere enviar a cada robot. De tal manera, el ordenador realiza la comunicación con los robots a través de UDP y les indica qué movimientos tienen que seguir. En la competición real, se utiliza un ordenador para cada robot, por lo que cada terminal se encarga de mandar las órdenes a uno de los robots de forma distribuida. Se muestra un esquema a continuación:

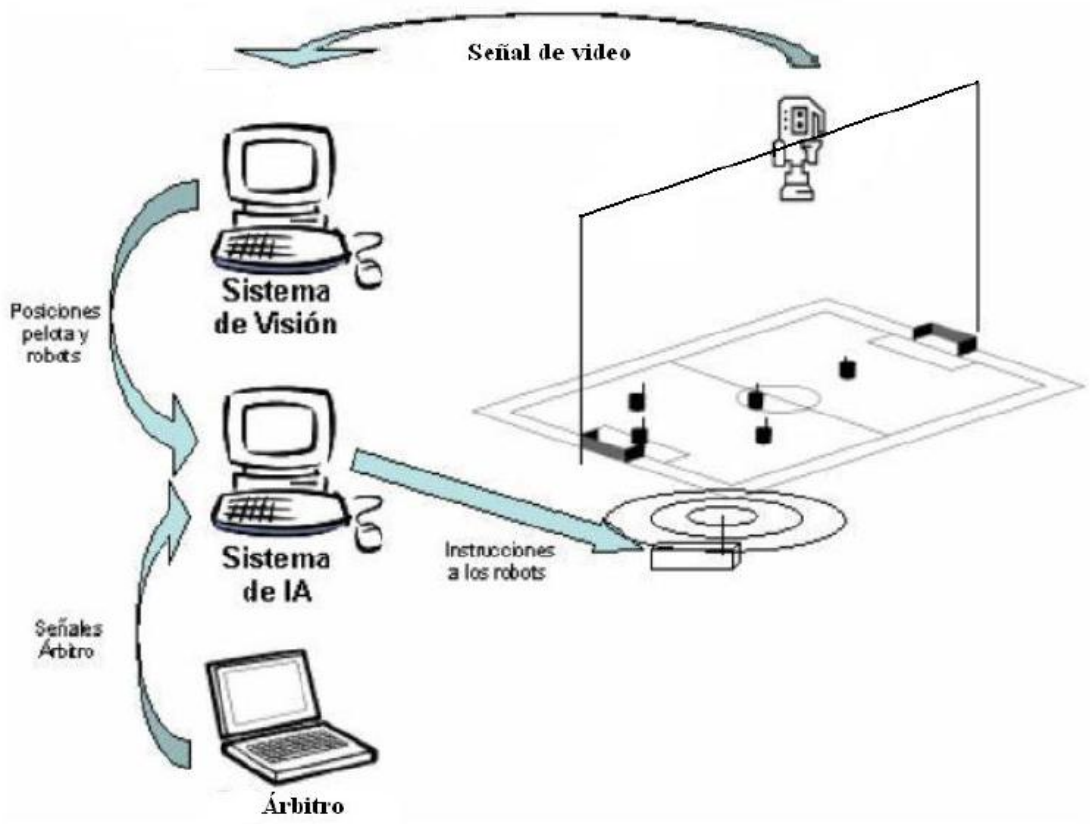


Ilustración 28: representación de la comunicación de la SSL [9]

2.4.3.1. Equipos

Los equipos en activo son:

- BRocks de Turquía.
- CMDragons de Estados Unidos.
- Eagle Knights – Robobulls de Méjico y Estados Unidos.
- ER-Force de Alemania.
- Immortals de Irán.

- KIKS de Japón.
- MRL de Irán.
- Odens de Japón.
- Omid de Irán.
- Owaribito-CU de Japón.
- Parsian de Irán.
- RFC Cambridge de Estados Unidos.
- RoboDragons de Japón.
- RoboJackets de Estados Unidos.
- RoboFEI de Brasil.
- Robopet de Brasil.
- RoboTurk de Turquía.
- Skuba de Tailandia.
- Strive de China.
- Thunderbots de Canadá.
- TPOTS de Singapur.
- Wright Eagle de China.
- ZjuNlict de China. [9]

También existen otros equipos formados que no han llegado a participar o porque no han sido aceptados, ya que cabe destacar que para poder participar es necesario presentar un vídeo de presentación. Mientras que el número de equipos en activo es 23, el número de equipos que intentan participar es de 52. [9]

Así, solo el 31% de los equipos creados están en activos y se les permite participar en la competición, esto se puede consultar en el siguiente gráfico circular:

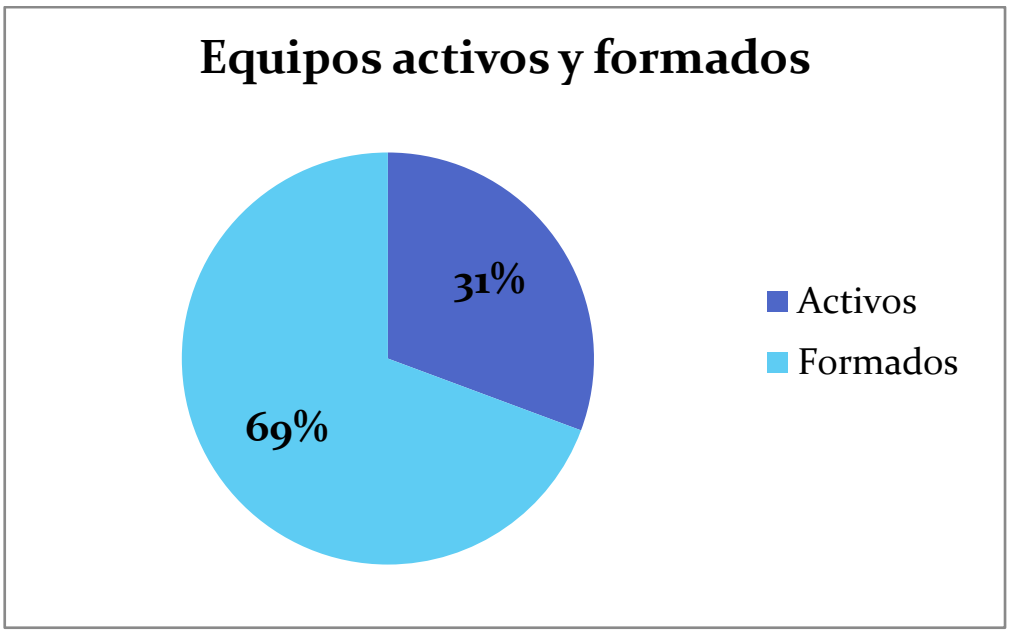


Ilustración 29: relación entre equipos en activo y formados [9]

Asimismo, la relación entre los distintos equipos y los países a los que pertenecen, haciendo distinción entre si es un equipo formado o activo, se representa a continuación:

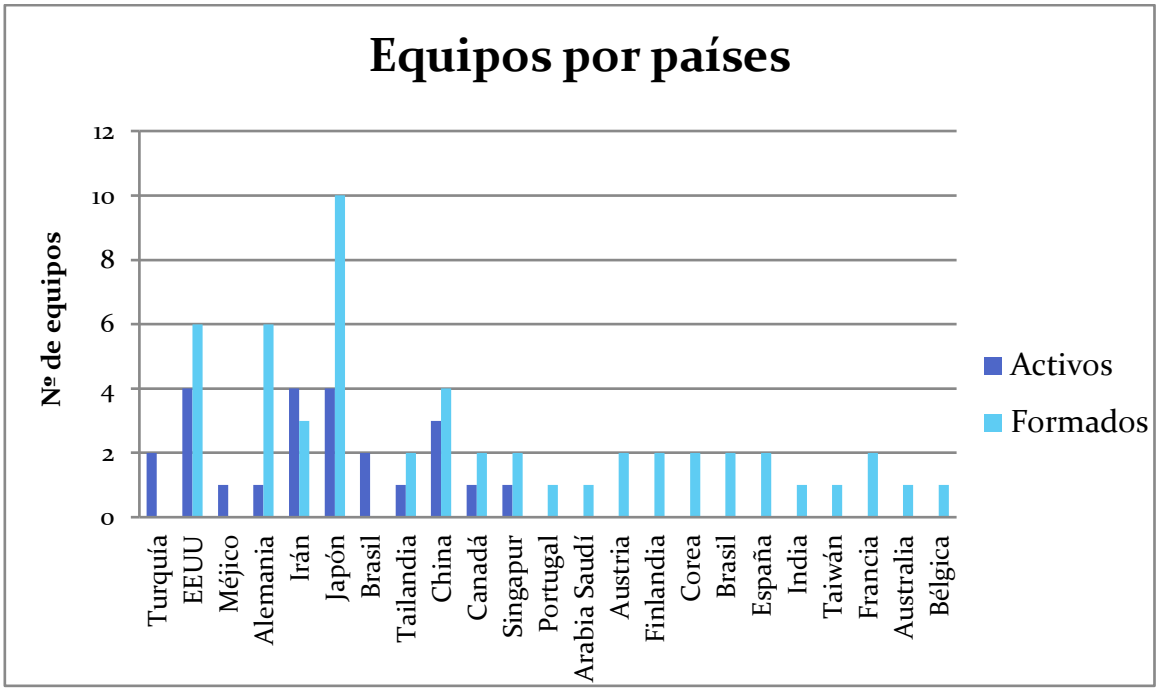


Ilustración 30: participación por países en la SSL [9]

En España en concreto hay solo dos equipos formados y ninguno activo. Estos equipos son el equipo "Girona" y el equipo "Rogi2", ninguno de los dos ha realizado ninguna publicación necesaria para poder participar de forma activa en la competición. [9]

Dentro de los equipos en activo, existe un ranking de victorias a lo largo de los años que se muestra a continuación, aclarando que en las cuatro últimas columnas aparecerán los años en los que el equipo consiguió esa posición:

Posición	Equipo	Primeras posiciones	Segundas posiciones	Terceras posiciones	Cuartas posiciones
1ª	CMDragons	1997 1998 2006 2007	2008 2010		
2ª	Cornell Big Red	1999 2000 2002 2003	2005	2003	
3ª	Skuba	2009 2010 2011		2008	
4ª	FU Fighters	2004 2005	1999 2000 2002	2003	2001
5ª	PasmaZ	2008	2007	2006 2009	
6ª	Lucky Star	2001		1999 2000 2002 2004	
7ª	RoboRoos		1998 2003 2004		
8ª	Field Rangers		2001	2005	2006
9ª	5dpo		2006	1998	
10ª	RoboDragons		2009	2007	
11ª	Naist		1997		
12ª	Immortals		2011		
13ª	MRL			2010 2011	
14ª	CMRoboDragons				2004 2005
15ª	ZjuNlict				2007 2008
16ª	KIKS				2010 2011
17ª	Cambridge				1998
18ª	Odeons				2009

Tabla 1: ranking de victorias en SSL [9]

2.4.3.2. Reglas

La SSL [9] publica todos los años unas reglas que los participantes tienen que seguir para poder concursar en esta competición. Sin embargo, es importante resaltar que cada año se modifican ligeramente estas reglas y que este PFC se pasa en las de 2011. Estas conforman diecisiete leyes y dos apéndices que se detallarán con más precisión en el "Anexo V: Reglamento 2011", aunque se introducirá en este punto un pequeño resumen sobre ellas:

- **LEY 1:** se establecen las dimensiones concretas del terreno de juego.
- **LEY 2:** descripción del balón.
- **LEY 3:** definición del número de robots por cada equipo.
- **LEY 4:** establecimiento de las características del equipo de robots.
- **LEY 5:** competencias del árbitro.
- **LEY 6:** competencias del árbitro asistente.
- **LEY 7:** duración del partido completo.
- **LEY 8:** cuándo se inicia y se reanuda el juego.
- **LEY 9:** cuándo el balón se para y está en juego.
- **LEY 10:** cuándo se puntúa.
- **LEY 11:** cuándo se produce fuera de juego.
- **LEY 12:** establece cuándo se considera falta y cuándo una conducta antideportiva.
- **LEY 13:** cuándo se asigna un tiro libre.
- **LEY 14:** define cuándo se realiza un tiro de penalti.

- **LEY 15:** define cuándo se hace un saque de banda.
- **LEY 16:** describe cuándo se hace un saque de puerta.
- **LEY 17:** cuándo se produce un saque de esquina.
- **Apéndice A:** describe las reglas de competencia de los participantes.
- **Apéndice B:** describe el sistema de visión y el rol de experto en visión.

En general, las leyes describen todo el comportamiento que se tiene que producir el terreno de juego, definiendo así todas las dimensiones, tanto del terreno de juego como de los robots, así como las reglas de juego, qué está permitido y qué no está permitido.

2.5. Simulador de la SSL

Hasta el momento no existe un simulador estándar utilizado por todos los equipos para simular las habilidades y movimientos de los robots que forman el equipo. Sin embargo, los equipos se ven en la obligación de desarrollar sus propios simuladores antes de aplicar los comportamientos en los robots físicos, por eso mismo se hallan algunos simuladores públicos que algunos equipos veteranos han desarrollado y publicado.

Existen muchos simuladores para la liga SSL [9], pero no hay muchos compartidos de forma pública. Hay algunos desarrollados y otros en los que se están trabajando, suelen ser publicados en repositorios públicos para que personas interesadas puedan colaborar.

Tras una búsqueda exhaustiva por los distintos equipos de la competición y consultas directas con los equipos a través de correo electrónico, se concluyó que no hay apenas simuladores de la SSL [9] públicos y, los que existen, están olvidados o pocas

personas trabajan en ellos. A continuación se citará el estado del arte perteneciente a los simuladores que se han consultado:

- **Twenta:** proyecto que nace con el fin de aportar un simulador sencillo e intuitivo, además de gratuito y de código abierto, que sea una referencia para los desarrolladores de equipos SSL [9]. Sin embargo, la actividad actualmente en el proyecto es nula y el trabajo no es muy elaborado. [4]

Ilustración 31: logotipo de simulador Twenta [4]

- **Carnegie Mellon University:** simulador programado en C++ [20], nace en el 2002 y pertenece al equipo CMDragons. Es muy elaborado pero no está disponible de forma pública el software completo. En su página web incluyen publicidad e información sobre él, pero por el momento no se puede utilizar si no se forma parte del equipo. [5]

Ilustración 32: logotipo del equipo CMDragons [5]

- **RoboCup-Simulator:** proyecto con ánimo de ser una propuesta para equipos que no tienen el simulador, actualmente no está terminado y pocas personas colaboran con él, se considera un proyecto prácticamente abandonado. [6]

- **GrSim:** simulador creado por el equipo Parsian Robotic [7] y programado en C++ [20], fue publicada su primera versión en julio 2011 y actualmente trabajan para una segunda versión mejorada. Incluye simulación de los robots, así como compatibilidad con el sistema de visión SSL-Vision [11]. Imita con exactitud la dinámica de la SSL [9] de robots físicos. [7] [8]

Ilustración 33: logotipo grSim [7] [8]

Como se puede apreciar, realmente disponibles y que puedan ser utilizados solo existe grSim [7] [8], cuyos creadores en todo momento están dispuestos a ofrecer ayuda y datos adicionales sobre el trabajo que han desarrollado. Por eso, finalmente se optó por elegir este simulador, que era el que mayor cobertura, seguridad y confianza ofrecía.

DESARROLLO TÉCNICO DEL PFC

3. Desarrollo Técnico del PFC

En este punto se describirá el trabajo realizado a lo largo del presente PFC, cuyo objetivo es la creación de una arquitectura que establezca las bases para crear un equipo para la liga SSL [9] de la competición RoboCup [17]. Debido a la envergadura de este PFC, se dividirá este capítulo en distintos subpuntos de acuerdo con los bloques de trabajo que se han ido elaborando, siendo estos:

- **Requisitos de usuario:** descripción de qué hará el sistema, dividido este punto en las distintas partes del desarrollo técnico.
- **Simulador grSim [7] [8]:** se describirá el trabajo realizado con el simulador utilizado, comentando el funcionamiento de este.
- **Cliente Java [18] de envío del simulador grSim [7] [8]:** cliente creado para enviar información al simulador.
- **Cliente Java [18] de recepción del simulador grSim [7] [8]:** cliente creado para recibir información del simulador, que incluye la posición de los robots y de la pelota.
- **Utilización de Arquitectura 3T para desarrollo de habilidades de la SSL [9]:** usando la arquitectura 3T [36] ya existente, se ha adaptado a las necesidades de la SSL [9].
- **Habilidades:** diseñadas habilidades básicas para robots relacionadas con el fútbol, como pueden ser el pase o el tiro a portería.
- **Aplicación en robots físicos:** todo el trabajo relacionado con los robots físicos, que incluye la modificación del simulador grSim [7] [8] para la adaptación de estos, así como las comunicaciones.

Por último, comentar que toda la información que incluye este PFC que fuera interesante para los creadores del simulador grSim [7] [8] ha sido compartida con este

equipo, pudiendo colaborar a desarrollar su proyecto. Se ha colaborado cediéndoles los clientes programados en Java [18], toda la documentación elaborada y las pruebas realizadas, ya que el simulador [7] [8] como tal no constaba de esta información. Para más información, se puede consultar en la web [7], donde en la parte de desarrolladores (*developers* en inglés) figura Verónica Raspeño, autora del presente PFC y Alejandro Caparrós Hernando en su propio PFC:

Ilustración 34: desarrolladores de grSim [7] [8]

3.1. Requisitos de usuario

En este apartado se detallan los requisitos que se han identificado en la fase de análisis del trabajo. Cada uno de los requisitos viene acompañado de ciertos datos compactados en una tabla de la siguiente forma:

ID	Identificador		
Nombre	Nombre de requisito		
Descripción	Descripción de requisito		
Necesidad	Necesidad de requisito	**Prioridad**	Prioridad de requisito
Estabilidad	Estabilidad de requisito	**Claridad**	Claridad de requisito

Tabla 2: ejemplo de requisito

Cuyos campos significan:

- **ID:** identificador del requisito, con la estructura REQ-TIPO-XX, siendo:
  - **TIPO:**
    - **SIM:** requisitos del simulador grSim [7] [8].
    - **ENV:** requisitos de cliente Java [18] de envío.
    - **REC:** requisitos de cliente Java [18] de recepción.
    - **ARQ:** requisitos de arquitectura 3T [36].
    - **HAB:** requisitos de habilidades.
    - **FIS:** requisitos de aplicación en robots físicos.
  - **XX**: el número de requisito.
- **Nombre:** identificativo del requisito, es un pequeño resumen de lo que representa el requisito.

- **Descripción:** breve descripción del requisito.
- **Necesidad:** del requisito, cómo de necesitado es para el sistema, los posibles valores son:
    - Esencial.
    - Deseable.
    - Opcional.
- **Prioridad:** importancia del requisito. Siendo posibles los siguientes valores:
    - Alta.
    - Media.
    - Baja.
- **Estabilidad:** cómo de estable es el requisito, pudiendo ser:
    - Estable.
    - No estable.
- **Claridad:** si el requisito es claro o no, pudiendo ser:
    - Alta.
    - Media.
    - Baja.

Asimismo, primero se detallarán los requisitos referentes a al simulador grSim [7] [8], tras ello, se incluirán los requisitos relacionados con el cliente Java [18] de envío y de recepción. Posteriormente se incluirán los requisitos propios de la arquitectura 3T [36],

los requisitos de las habilidades programadas en este PFC y, por último, los requisitos relacionados con la aplicación en robots físicos.

3.1.1. Requisitos de simulador grSim

En este apartado se detallarán los requisitos relacionados con el simulador utilizado, grSim [7] [8]:

ID	**REQ-SIM-01**		
Nombre	Sistema operativo		
Descripción	El simulador utilizado será ejecutable en cualquier sistema operativo Unix.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 3: REQ-SIM-01

ID	**REQ-SIM-02**		
Nombre	Simulación		
Descripción	El simulador simulará el terreno de juego, robots y balón propios de la SSL [9].		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 4: REQ-SIM-02

ID	**REQ-SIM-03**		
Nombre	SSL-Vision		
Descripción	El simulador simulará el procesamiento de imágenes, así como el SSL-Vision [11] propio de la liga SSL [9].		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 5: REQ-SIM-03

3.1.2. Requisitos de cliente Java de envío

Estos requisitos corresponden a los que detallan las características del cliente Java de envío:

ID	**REQ-ENV-01**		
Nombre	Lenguaje programación		
Descripción	Se hará un cliente en lenguaje Java [18] que permita enviar paquetes al simulador grSim [7] [8] y a los robots físicos.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 6: REQ-ENV-01

ID	**REQ-ENV-02**		
Nombre	Interfaz		
Descripción	Se creará una interfaz en Java [18] en donde se puedan configurar todos los parámetros relacionados con los robots.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 7: REQ-ENV-02

ID	**REQ-ENV-03**		
Nombre	Movimiento robots simulados		
Descripción	El cliente permitirá mover los robots en el simulador grSim [7] [8] con el paquete que envíe.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 8: REQ-ENV-03

ID	REQ-ENV-04		
Nombre	Movimiento robots físicos		
Descripción	El cliente permitirá mover los robots físicos con el paquete que envíe.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 9: REQ-ENV-04

ID	REQ-ENV-05		
Nombre	Estructura paquete		
Descripción	El paquete enviado seguirá un estándar dado por la organización de la competición RoboCup [17].		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 10: REQ-ENV-05

ID	REQ-ENV-06		
Nombre	Google Protobuf [16]		
Descripción	El paquete enviado estará codificado por Google Protobuf [16].		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 11: REQ-ENV-06

3.1.3. Requisitos de cliente Java de recepción

Estos requisitos corresponden a los que detallan las características del cliente Java [18] de recepción:

ID	**REQ-REC-01**		
Nombre	Lenguaje programación		
Descripción	Se hará un cliente en lenguaje Java [18] que permita recibir paquetes al simulador grSim [7] [8] y a los robots físicos.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 12: REQ-REC-01

ID	**REQ-REC-02**		
Nombre	Procesamiento de paquetes		
Descripción	El cliente será capaz de procesar el paquete recibido, siempre que respete la estructura dada por la organización de la competición RoboCup [17].		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 13: REQ-REC-02

ID	**REQ-REC-03**		
Nombre	Google Protobuf [16]		
Descripción	El paquete recibido estará codificado por Google Protobuf [16], el cliente será capaz de interpretarlo.		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 14: REQ-ENV-03

3.1.4. Requisitos de arquitectura 3T

Estos requisitos corresponden a los que detallan las características de la arquitectura 3T par la SSL [9]:

ID	**REQ-ARQ-01**		
Nombre	Arquitectura SSL [9]		
Descripción	Creación de una arquitectura para la SSL [9] de la RoboCup [17].		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 15: REQ-ARQ-01

ID	**REQ-ARQ-02**		
Nombre	Lenguaje programación		
Descripción	La arquitectura se programará en el lenguaje de programación Java [18].		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 16: REQ-ARQ-02

ID	**REQ-ARQ-03**		
Nombre	Tipo arquitectura		
Descripción	Se utilizará la arquitectura 3T [36], basada en la creada por José Luis Cebrián Díaz. [31]		
Necesidad	Esencial	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 17: REQ-ARQ-03

ID	REQ-ARQ-04		
Nombre	Comunicación con simulador		
Descripción	La arquitectura se comunicará con el simulador grSim [7] [8].		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 18: REQ-ARQ-04

ID	REQ-ARQ-05		
Nombre	Recepción datos		
Descripción	La arquitectura será capaz de procesar y utilizar los datos enviados por el simulador grSim [7] [8].		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 19: REQ-ARQ-05

ID	REQ-ARQ-06		
Nombre	Envío datos		
Descripción	La arquitectura será capaz de enviar datos al simulador grSim [7] [8] y que este los procese y utilice.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 20: REQ-ARQ-06

ID	REQ-ARQ-07		
Nombre	Datos de entrada robots simulados		
Descripción	Los datos de entrada serán proporcionados exclusivamente por el paquete que se recibe del grSim [7] [8] en la simulación.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 21: REQ-ARQ-07

ID	REQ-ARQ-08		
Nombre	Datos de entrada robots físicos		
Descripción	Los datos de entrada serán proporcionados exclusivamente por el paquete que se recibe del SSL-Vision [11] en los robots físicos.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 22: REQ-ARQ-08

ID	REQ-ARQ-09		
Nombre	Hilo recepción paquetes		
Descripción	Se tendrá un hilo de ejecución para recibir los paquetes que envía el simulador [7] [8] en caso de la simulación y el SSL-Vision [11] en caso de los robots físicos.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 23: REQ-ARQ-09

3.1.5. Requisitos de habilidades

Estos requisitos corresponden a los que detallan las características de las habilidades:

ID	REQ-HAB-01		
Nombre	Habilidades		
Descripción	Se crearán distintas habilidades básicas que imiten el comportamiento de robots que juegan al fútbol.		
Necesidad	Esencial	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 24: REQ-HAB-02

ID	REQ- HAB-02		
Nombre	Hilo habilidad esquivar		
Descripción	Se creará un hilo de ejecución para realizar la habilidad de esquivar por parte de los robots.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 25: REQ-HAB-02

ID	REQ- HAB-03		
Nombre	Habilidades complementarias		
Descripción	Las habilidades se complementarán con otras habilidades pertenecientes al PFC de Alejandro Caparrós Hernando, aun no publicado [32].		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 26: REQ-HAB-03

ID	REQ- HAB-04		
Nombre	Habilidades robot portero		
Descripción	Un grupo de habilidades se centrará en emular el comportamiento del robot portero.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 27: REQ-HAB-04

ID	REQ- HAB-05		
Nombre	Habilidad robot defensa		
Descripción	Un grupo de habilidades se centrará en emular el comportamiento de los robots defensa.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 28: REQ-HAB-05

ID	REQ- HAB-06		
Nombre	Habilidad robot atacante		
Descripción	Un grupo de habilidades se centrará en emular el comportamiento de los robots atacantes.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 29: REQ-HAB-06

ID	REQ- HAB-07		
Nombre	Habilidad buscar portería		
Descripción	Una de las habilidades consistirá en que el robot portero vaya hasta la portería.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 30: REQ-HAB-07

ID	REQ- HAB-08		
Nombre	Habilidad defender portería robot portero		
Descripción	Una de las habilidades consistirá en que el robot portero defienda la portería, de tal manera que evite que el balón se meta dentro de ella.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 31: REQ-HAB-08

ID	REQ- HAB-09		
Nombre	Habilidad defender portería robot defensa		
Descripción	Una de las habilidades consistirá en que el robot		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 32: REQ-HAB-09

ID	REQ-HAB-10		
Nombre	Habilidad buscar pelota		
Descripción	Una de las habilidades consistirá en que el robot sea capaz de buscar la pelota.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 33: REQ-HAB-10

ID	REQ- HAB-11		
Nombre	Habilidad conducir pelota		
Descripción	Una de las habilidades consistirá en que el robot sea capaz de conducir la pelota.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 34: REQ-HAB-11

ID	REQ- HAB-12		
Nombre	Habilidad pase		
Descripción	Una de las habilidades consistirá en que el robot tire la pelota otro robot del mismo equipo.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 35: REQ-HAB-12

ID	REQ- HAB-13		
Nombre	Habilidad esquivar		
Descripción	Una de las habilidades consistirá en que el robot esquive a otros robots, sin chocarse.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 36: REQ-HAB-13

ID	REQ- HAB-14		
Nombre	Habilidad disparar		
Descripción	Una de las habilidades consistirá en que el robot dispare el balón hacia la portería.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 37: REQ-HAB-14

3.1.6. Requisitos de aplicación en robots físicos

En este apartado se detallarán los requisitos propios de la aplicación en los robots físicos, de tal manera que se consigue ampliar el presente PFC:

ID	REQ-FIS-01		
Nombre	Estándares del robot		
Descripción	Los robots físicos tienes que seguir los estándares propuestos por la competición de la RoboCup [17].		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 38: REQ-FIS-01

ID	REQ-FIS-02		
Nombre	Robot utilizado		
Descripción	Los robots físicos que se utilizarán serán los propuestos por dos proyectos anteriores publicados [34] [35].		
Necesidad	Deseable	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 39: REQ-FIS-02

ID	REQ-FIS-03		
Nombre	Modificación simulador grSim [7] [8]		
Descripción	Se modificará el simulador grSim [7] [8] para que simule exactamente al robot físico con el que se realicen las pruebas.		
Necesidad	Deseable	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 40: REQ-FIS-03

ID	REQ-FIS-04		
Nombre	Conexiones		
Descripción	Se utilizará el firmware creado por el compañero Isaac Ruiz Agrazal para conectar el robot con la arquitectura.		
Necesidad	Deseable	**Prioridad**	Alta
Estabilidad	Estable	**Claridad**	Alta

Tabla 41: REQ-FIS-04

ID	REQ-FIS-05		
Nombre	Google Protobuf [21]		
Descripción	Se utilizará el sistema de visión, que incluye la codificación de los paquetes a través de Google Protobuf [21].		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 42: REQ-FIS-05

ID	REQ-FIS-06		
Nombre	Sistema de visión		
Descripción	Se utilizará la cámara Stingray [16] para el sistema de visión.		
Necesidad	Deseable	Prioridad	Alta
Estabilidad	Estable	Claridad	Alta

Tabla 43: REQ-FIS-06

3.2. Simulador grSim

En este apartado se tratará toda la información relacionada con el simulador utilizado para representar el comportamiento de los robots. Para ello se ha hecho uso del simulador grSim, creado por el equipo participante de la competición RoboCup [17], Parsian Robotic. [7] [8]

En todo momento se hará referencia a dos equipos, siempre identificados por un color, este puede ser únicamente azul o amarillo. Esta información relacionada con el formato y otro tipo de especificaciones de la liga se puede consultar en el "Anexo V: Reglamento 2011".

Por último, todos los paquetes, así como las estructuras de estas, han sido captados y analizados por el software Wireshark [61], a través de la siguiente interfaz:

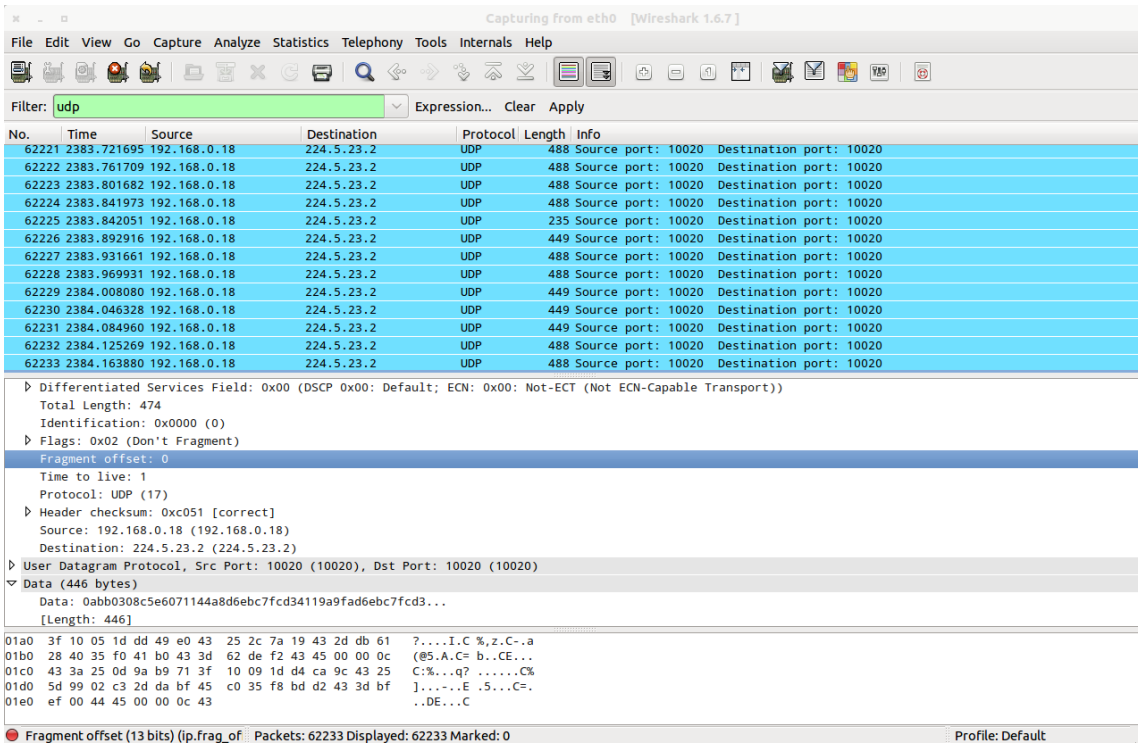


Ilustración 35: interfaz Wireshark [61]

3.2.1. Funcionamiento

El simulador simula no solo la representación gráfica, sino también el funcionamiento interno de la competición SSL [9] de la RoboCup [17].

El sistema simula lo que debería enviar el sistema de visión y lo envía al ordenador que lo está ejecutando por UDP por *multicast*. El paquete que se manda es exactamente igual al que manda el sistema de visión en los robots reales, como se representa a continuación:

Descripción	Paquete recibido del simulador
Información sobre el paquete: • Número de *frame*. • Tiempo de captura. • Tiempo de envío. • Id de la cámara.	detection { frame_number: 0 t_capture: 0.0 t_sent: 0.0 camera_id: 0

Información sobre la pelota:
 	
• Grado de precisión de los datos.
• Coordenada x.
• Coordenada y.
• Coordenada z.
• Píxel en la coordenada x de la representación gráfica.
• Píxel en la coordenada y de la representación gráfica.
	balls {
	confidence: 0.0
	x: 0.0
	y: 0.0
	z: 0.0
	pixel_x: 0.0
	pixel_y: 0.0
	}

Información sobre el equipo amarillo, salen cinco bloques como este:
 	
• Grado de precisión de los datos.
• Id del robot.
• Coordenada x.
• Coordenada y.
• Orientación en radianes.
• Píxel en la coordenada x de la representación gráfica.
• Píxel en la coordenada y de la representación gráfica.
	robots_yellow {
	confidence: 0.0
	robot_id: 0
	x: 0.0
	y: 0.0
	orientation: 0.0
	pixel_x: 0.0
	pixel_y: 0.0
	}
	[…]

Información sobre el equipo azul, salen cinco bloques como este:
 	
• Grado de precisión de los datos.
• Id del robot.
• Coordenada x.
• Coordenada y.
• Orientación en radianes.
• Píxel en la coordenada x de la representación gráfica.
• Píxel en la coordenada y de la representación gráfica.
	robots_blue {
	confidence: 0.0
	robot_id: 0.0
	x: 0.0
	y: 0.0
	orientation: 0.0
	pixel_x: 0.0
	pixel_y: 0.0
	}
	[…]
	}

Tabla 44: paquete del sistema de visión que envía grSim [7] [8]

La arquitectura recibe el paquete del sistema de visión y realiza las habilidades pertinentes, para mandar otro paquete al simulador que, con la información recibida, procederá a realizar los movimientos y actividades mandadas. El paquete que se envía es el siguiente:

Descripción	**Paquete enviado al simulador**

La información que incluye el paquete es:
• Retardo que produce el paquete.
• El equipo que es entre azul y amarillo.
• Información propia del robot:
  ○ Identificador del robot entre 0 y 4.
  ○ Disparo normal.
  ○ Disparo elevado.
  ○ Velocidad tangencial.
  ○ Velocidad normal.
  ○ Velocidad angular.
  ○ Activación o desactivación del *spinner*.
  ○ Activación de la velocidad de cada rueda de forma independiente:
  ▪ Velocidad de la rueda 1.
  ▪ Velocidad de la rueda 2.
  ▪ Velocidad de la rueda 3.
  ▪ Velocidad de la rueda 4, inexistente, por lo que siempre es cero.
	commands {
	timestamp: 0.0
	isteamyellow: true
	robot_commands {
	id: 0
	kickspeedx: 0.0
	kickspeedz: 0.0
	veltangent: 0.0
	velnormal: 0.0
	velangular: 0.0
	spinner: false
	wheelsspeed: false
	wheel1: 0.0
	wheel2: 0.0
	wheel3: 0.0
	wheel4: 0.0
	}
	}

Tabla 45: paquete de la arquitectura al grSim [7] [8]

3.2.2. Sistema de visión en grSim

El simulador grSim [7] [8] no solo emula el comportamiento de los robots y la situación en el terreno de juego de estos, sino que, además, plasma el funcionamiento real del sistema de visión de esta liga.

En este punto se va a detallar cómo envía el simulador la información para que los clientes reciban la información del estado del partido.

Este sistema codifica los paquetes con información utilizando Google Protobuf [21], este sistema permite codificar estructuras de datos de una forma eficiente.

En la siguiente imagen se muestra el funcionamiento de Google Protobuf [21]:

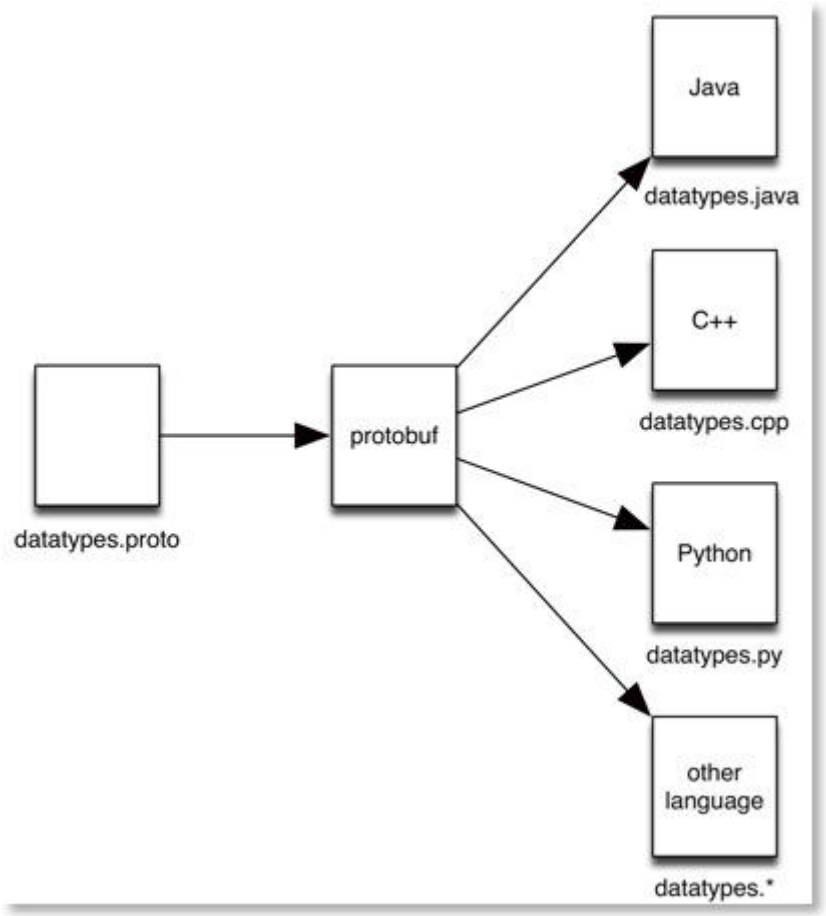


Ilustración 36: esquema del funcionamiento de Google Protobuf [21]

El simulador envía una estructura de datos codificada con Google Protobuf [21] y, una vez recibidos los datos, se transforma en código para permitir tratar la información obtenida y poder almacenarla.

3.2.3. Interfaz

El simulador de grSim [7] [8] ofrece una interfaz gráfica para representar el campo de fútbol, así como los diez robots, la pelota y otra información relevante. La relación de las dimensiones del campo y todo lo contenido en él guardan las mismas dimensiones con distinta escala respecto a los robots físicos. Esta cualidad logra facilitar las pruebas a través del simulador y aportar unas conclusiones fehacientes.

La representación global del simulador grSim [7] [8] es la siguiente:

Ilustración 37: representación simulador grSim [7] [8]

Respecto a la interfaz, en la parte derecha se puede encontrar la representación gráfica, cada robot tiene un número asociado que aparece en la parte superior de este. Este número va acorde con los colores, representados en cinco círculos, el central marca el equipo (azul o amarillo) y los otros cuatro definen la equivalencia con el número. Así, el robot azul con el número cero tendrá el círculo central azul, los dos círculos superiores rosas y los dos inferiores verdes.

Desde la propia simulación se pueden modificar algunos parámetros relacionados con los robots y la pelota, extrayendo dos tipos de menú: pulsando un robot o pulsando el campo.

Se puede seleccionar directamente uno de los robots y aparece el siguiente menú:

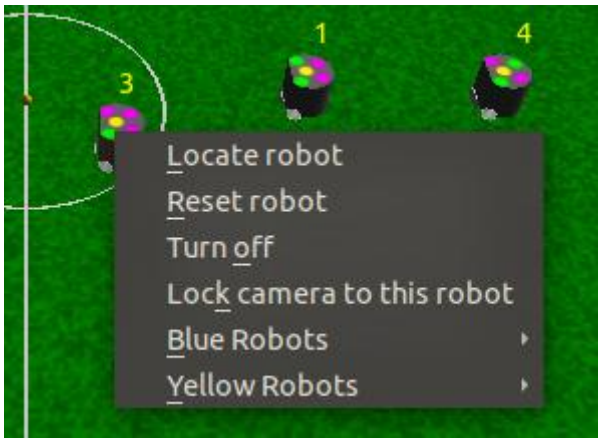


Ilustración 38: representación menú de robot en grSim [7] [8]

Se puede modificar la localización del robot, así como recuperar la orientación inicial con "*reset robot*", bloquear el robot y parar su movimiento con "*turn off*", localizar la cámara asociada al robot, sacar fuera del campo a un equipo u otro y ponerlos en formación de la siguiente manera:

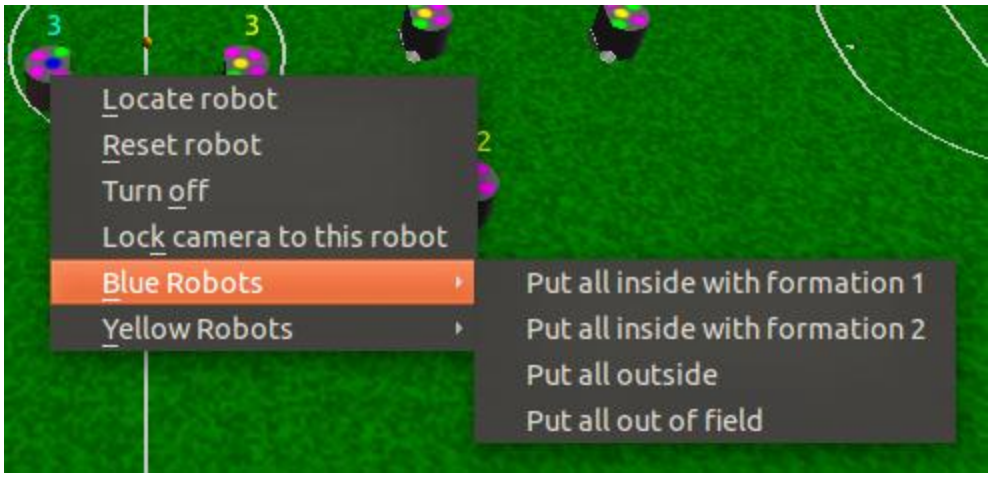


Ilustración 39: representación menú de equipo en grSim [7] [8]

Mientras que si se pulsa sobre el campo, aparecerá este menú:

Ilustración 40: representación menú de campo en grSim [7] [8]

Este menú permite modificar la posición de la pelota, así como la localización del robot seleccionado en ese momento y modificaciones de cámara y equipos.

En el menú superior de la parte izquierda aparece la información relacionada con los robots, la pelota, el campo y los datos de conexión. Debajo de este se especifican los datos del robot seleccionado, en la captura anterior el robot cero. Este menú representa la vista aérea del robot y permite, a través de distintas opciones, modificar datos de este, que son:

- **Team:** permite seleccionar el equipo, entre "*blue*" (azul) o "*yellow*" (amarillo).
- **Index:** este dato corresponde al número asociado del robot, se puede elegir entre cero y cuatro.
- **Velocity:** campo que no se puede modificar y que marca la velocidad que lleva el robot en ese momento.
- **Reset:** botón que permite reiniciar el robot, al presionarlo el robot se orienta como estaba al iniciar el simulador.
- **Locate:** este botón proporciona la opción de localizar el robot directamente. Al pulsarlo, en el campo aparece un círculo que se puede mover a través del botón y que mueve el robot a donde se traslade el círculo. Se representa a continuación:

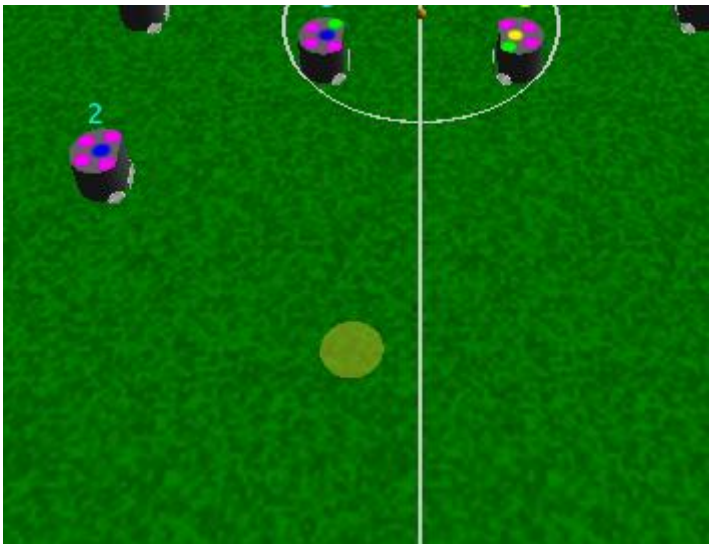


Ilustración 41: representación de Locate en grSim [7] [8]

- **Turn off:** permite bloquear al robot, por lo que si está en movimiento se para inmediatamente.

- **Set position:** con este botón se pueden cambiar datos relacionados con la localización del robot, que incluye la posición en "x", la posición en "y" y la orientación del robot, que hay que poner necesariamente en grados, a través de este menú:

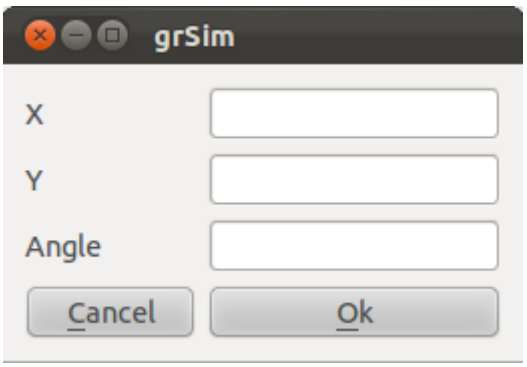


Ilustración 42: representación de Set position en grSim [7] [8]

Como se puede apreciar en la interfaz, la ventana emergente cambia el eje X e Y del robot, así como el ángulo. Los datos del eje se dan en centímetros y los datos del ángulo se dan en grados.

En la parte inferior aparece la información que el simulador quiere hacer llegar al usuario, avisando del estado de este, cualquier error o problema que pueda surgir, etc.

3.2.4. Simulación

El simulador tiene ciertas particularidades que son necesarias comentar para su pleno entendimiento.

Todos los objetos que aparecen en el campo, ya sean los robots o la pelota, así como cualquier posición representada se basa en dos coordenadas, la x y la y. Para determinar estas dos coordenadas es necesario dibujar dos ejes en el campo para así conocer el signo (positivo o negativo) de cada una de ellas. Se aporta la representación gráfica, marcando el signo con la siguiente notación "(signo coordenada x, signo coordenada y)":

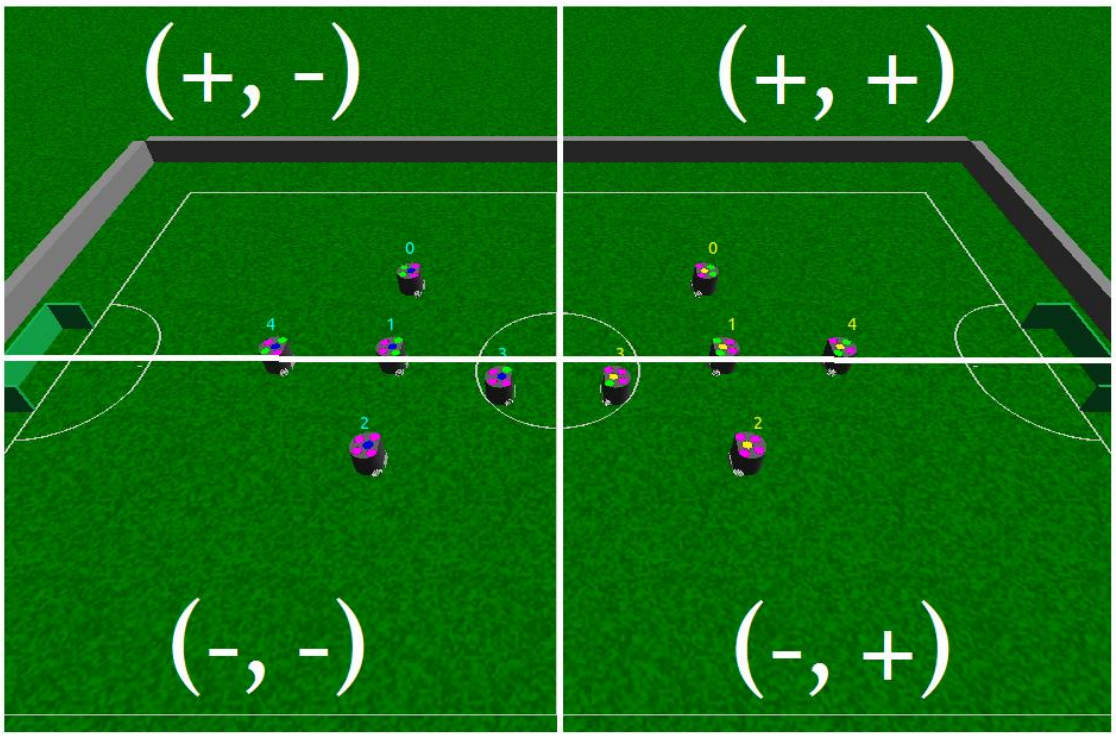


Ilustración 43: signos coordenadas de simulador grSim [7] [8]

Además, para determinar la orientación de los robots es necesario conocer la notación que utiliza el simulador. La forma es trazar una circunferencia en el campo e ir dando los valores que corresponden en grados, siendo el cero la parte derecha y π en la parte izquierda. Se entiende que π, según la notación trigonométrica, equivale a 180 grados. Se muestra lo comentado gráficamente:

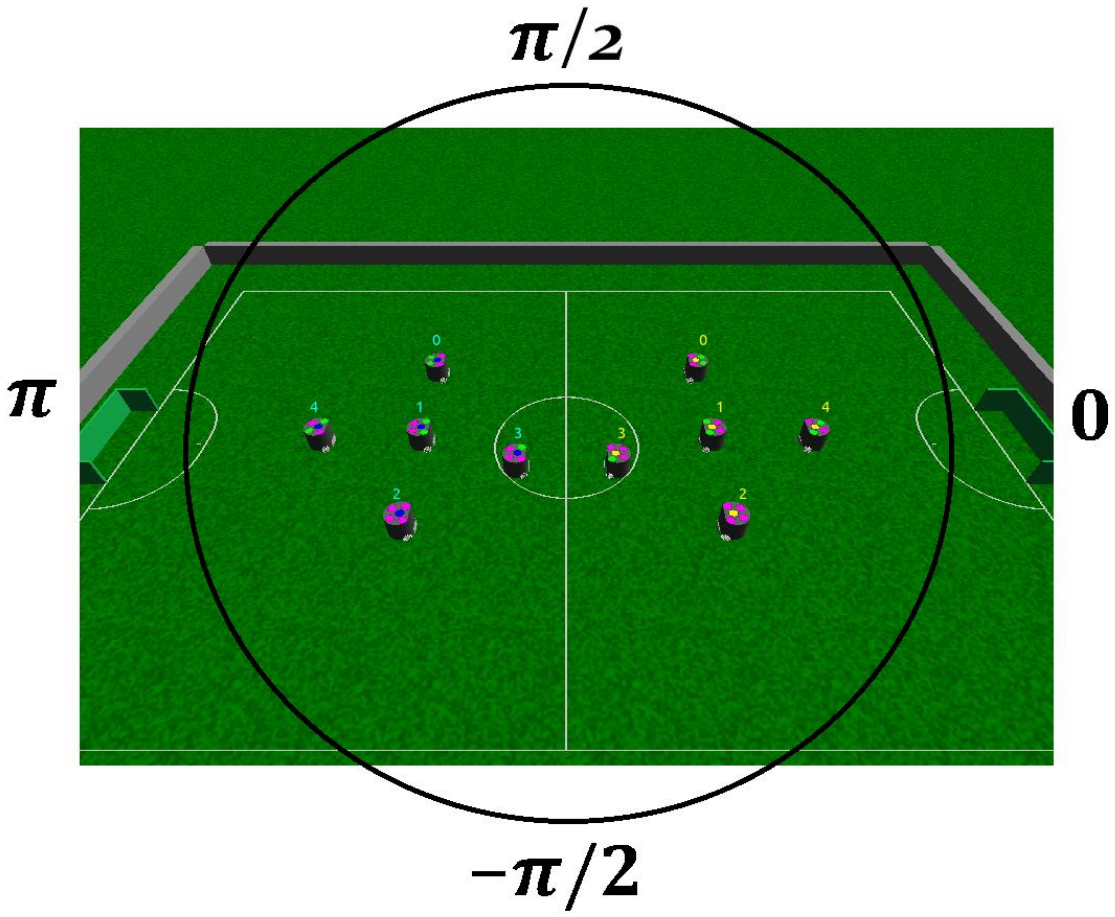


Ilustración 44: orientación de simulador grSim [7] [8]

Sin embargo, el simulador en el paquete recibe por parte de la arquitectura la orientación de los robots en radianes, ya que es así como se lleva a cabo en los robots físicos. Para calcularlo, simplemente hay que transformar los grados obtenidos a radianes con la siguiente fórmula:

$$radianes = \frac{grados \times \pi}{180}$$

3.3. Cliente Java de envío del simulador grSim

A lo largo de este punto se desarrollará la descripción del cliente Java [18] básico para manejar el simulador grSim [7] [8] a través de una interfaz.

3.3.1. Funcionamiento

El funcionamiento del cliente Java [18] para el simulador grSim [7] [8] utilizado permite usar el simulador de una forma sencilla y directa, logrando así poder mover los robots a nivel básico y comprobar que la información está siendo recibida.

El cliente Java creado [18] está basado en el cliente básico que aporta el simulador grSim [7] [8] en el lenguaje de programación C++ [20] y se encarga de verificar el pleno funcionamiento del envío de paquetes UDP, que logran hacer moverse a los robots según se definan los parámetros.

El cliente en Java [18] envía el paquete UDP que contiene la información que exige el simulador para poder comprenderlo, de acuerdo con lo que el usuario escribe en la interfaz. El formato de este paquete se puede encontrar en el apartado 4.1.1. Funcionamiento en la Tabla 3.

3.3.2. Interfaz

El simulador ofrece una interfaz para que se escriban los parámetros relacionados con los robots del simulador para lograr moverlos y ponerlos en funcionamiento. La interfaz creada es de la siguiente forma:

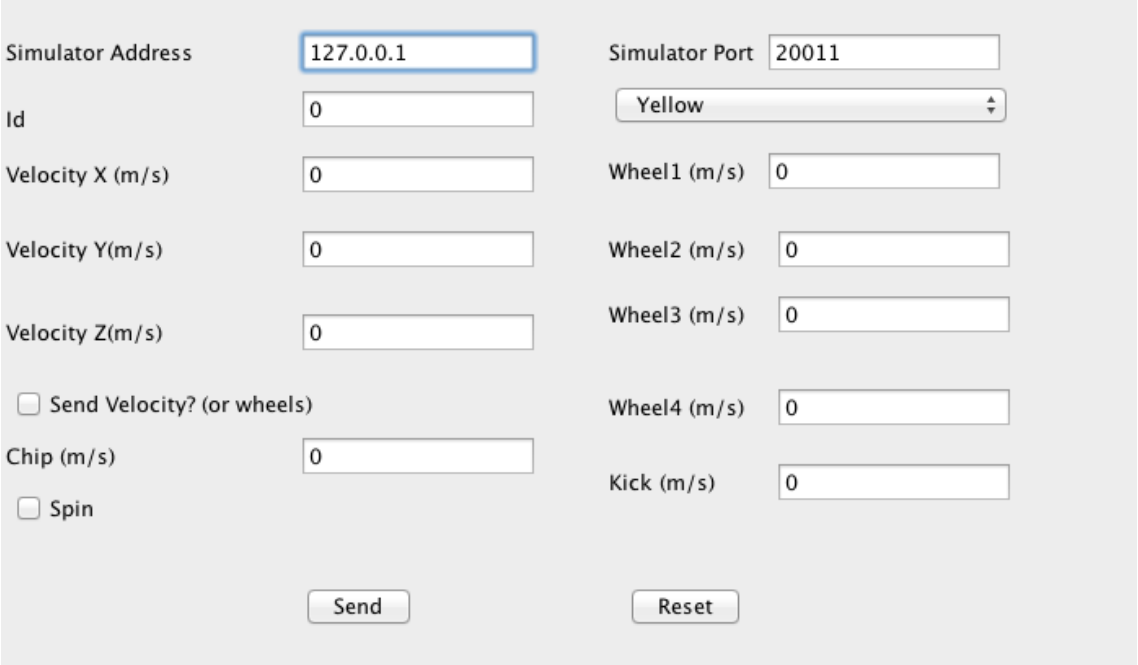


Ilustración 45: interfaz cliente Java [18] del simulador grSim [7] [8]

En la primera columna, el primer campo es para rellenar la dirección IP del simulador, a donde se enviará el paquete UDP.

El siguiente campo corresponde con el identificador del robot, por cada equipo hay cinco robots, numerados del cero al cuatro, tanto en un equipo como en otro. Si se introduce un identificador equivocado, el simulador lo interpreta y no mueve ningún robot.

Tras el identificador se puede rellenar la velocidad en los tres ejes de coordenadas (X, Y, Z) representada en m/s, de tal manera:

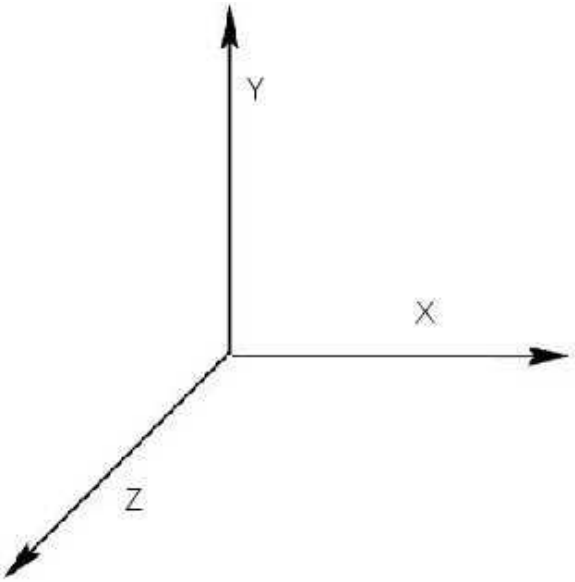


Ilustración 46: eje de tres coordenadas

La velocidad debe ser un número flotante, por lo que debe incluir decimales separados con un punto o enteros sin decimales, ejemplos son:

Tipo	Ejemplo
Entero	1, 2, 3, …
Flotante	1.0, 1.1, 1.2, …

Tabla 46: ejemplo número velocidad

La velocidad puede ser negativa o positiva, así, el robot se moverá para una dirección u otra. Se representa un ejemplo de dos ejes (X, Y), según la dirección de las flechas y si la velocidad es negativa o no, el robot se moverá de tal manera:

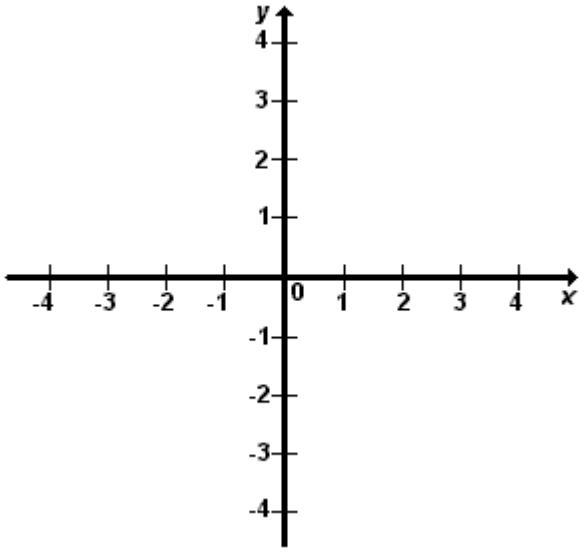


Ilustración 47: eje de dos coordenadas

Si se rellenan una o varias velocidades el simulador lo interpretará de igual forma. Después, el simulador combinará las velocidades de los distintos ejes y realizará el movimiento que corresponda con la combinación de las tres direcciones, como aparece en la siguiente ilustración:

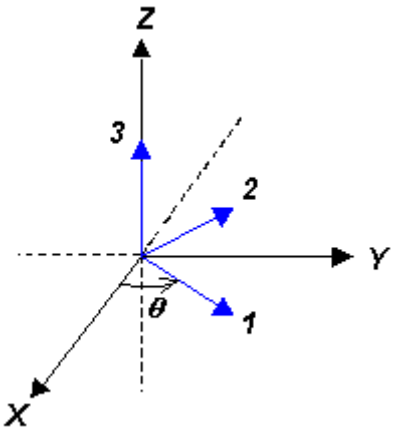


Ilustración 48: eje de dos coordenadas con combinaciones de direcciones

Seguidamente se puede marcar el *checkbox* o casilla de verificación relacionado con el envío de la velocidad a las ruedas de forma independiente. Si esta casilla está marcada, se enviará al simulador la velocidad que se marque a las ruedas, si no está marcada, se enviará la velocidad de los ejes como se ha detallado anteriormente.

Debajo del *checkbox* se puede incluir la información relacionada con el "chip". Este parámetro sirve que el robot lance la pelota con cierta altura, esta altura se escoge a través de la interfaz, se representa en m/s y debe seguir el formato que cita la Tabla 4.

Tras ello hay otro *checkbox* que marca el spin, esto sirve para activar el control de la pelota por parte del robot, a través del sistema de *dribbler* que se detalló en el apartado "2.3.3.2. Reglas".

El primer parámetro de la segunda columna es el puerto del simulador, que se tendrá que escribir de acuerdo con la IP para que el paquete UDP llegue adecuadamente.

Tras ello hay una lista en la que se puede seleccionar el color del equipo del robot que se quiere mover. En este caso se ofrecen dos opciones: "*yellow*", que corresponde al color amarillo y "*blue*", que equivale al color de robot azul.

Después existen cuatro campos para incluir la velocidad de las ruedas de forma independiente. Estos parámetros solo funcionarán si está activado el *checkbox* que se detalló anteriormente y deben seguir el formato que se describió en la Tabla 4. Inicialmente el simulador estaba preparado para robots con cuatro ruedas, por eso mismo la interfaz representa ese campo con esa cuarta rueda, se realizó de esa manera para que fuera útil con robots de tres o cuatro ruedas. Debido a que los robots físicos que se han utilizado constan de tres ruedas, el simulador posteriormente se adaptó modificando los robots de cuatro a tres ruedas. Por lo tanto, si el usuario rellena esta cuarta rueda, la información no interfiere en el comportamiento del robot.

El último parámetro es "*kick*", este sirve para lanzar la pelota sin ningún tipo de elevación. La velocidad de disparo se define en m/s y sigue el formato que se comentó anteriormente.

Finalmente, existen dos botones en la parte inferior de la interfaz. El primer botón, "*send*", sirve para enviar la información al simulador y el segundo botón, "*reset*", sirve para borrar lo escrito en la interfaz y salgan los valores predeterminados.

3.4. Cliente Java de recepción del simulador grSim

De forma análoga al punto anterior, se ha elaborado un cliente Java [18] de recepción de los datos que envía el simulador grSim [7] [8], para poder procesarlos y mandar las órdenes a los robots en base a ello.

Este cliente en Java [18] sirve para recibir los paquetes del simulador [7] [8] y del SSL-Vision [11]. Si se utilizando el simulador [7] [8], los paquetes serán enviados directamente por este, simulando, así mismo, el sistema de visión que se usa en la SSL [9] oficial. Sin embargo, si se usan los robots físicos, esta información es enviada por el SSL-Vision [11] únicamente.

Con esta información, la arquitectura puede usar esos datos para conocer la situación de los robots y de la pelota y poder actuar en consecuencia.

Estos paquetes que recibe el cliente creado están codificados por el simulador [7] [8] con Google Protobuf [21], tal como sucede en la liga SSL [9] real. Esto es debido a que el simulador intenta simular exactamente cómo funciona la liga SSL [9] de la competición RoboCup [17], de tal manera, esta liga utiliza la codificación de paquetes Google Protobuf [21].

El cliente recibe un paquete enviado directamente del simulador [7] [8] o del SSL-Vision [11], en caso de que se usen los robots físicos, y lo muestra por pantalla para su posterior análisis. Este paquete viene descrito en el punto "3.3.1. Funcionamiento", el cual contiene información sobre los robots de ambos equipos y la pelota.

Para lograr esto en el simulador [7] [8], el cliente Java [18] se conecta con la dirección *multicast* de aquel.

3.5. Utilización de Arquitectura 3T para desarrollo de habilidades de la SSL

En este apartado se desarrollará la utilización de la arquitectura 3T [36] en este proyecto. Para ello, se incluirán distintos diagramas que ayudan a comprender el funcionamiento de la arquitectura y de la inclusión de las modificaciones para incrustar las habilidades de la SSL [9]. Hay que recordar que esta arquitectura está basada en un PFC elaborado por el compañero José Luis Cebrián Díaz [31].

3.5.1. Diseño arquitectónico

En este punto se va a desarrollar el diseño arquitectónico de la arquitectura 3T [36] diseñada, reutilizando la arquitectura ya existente de otro PFC [31].

Esta arquitectura se encuentra modularizada para facilitar su desarrollo y la inclusión de nuevas funcionalidades. En la siguiente imagen se muestra la arquitectura:

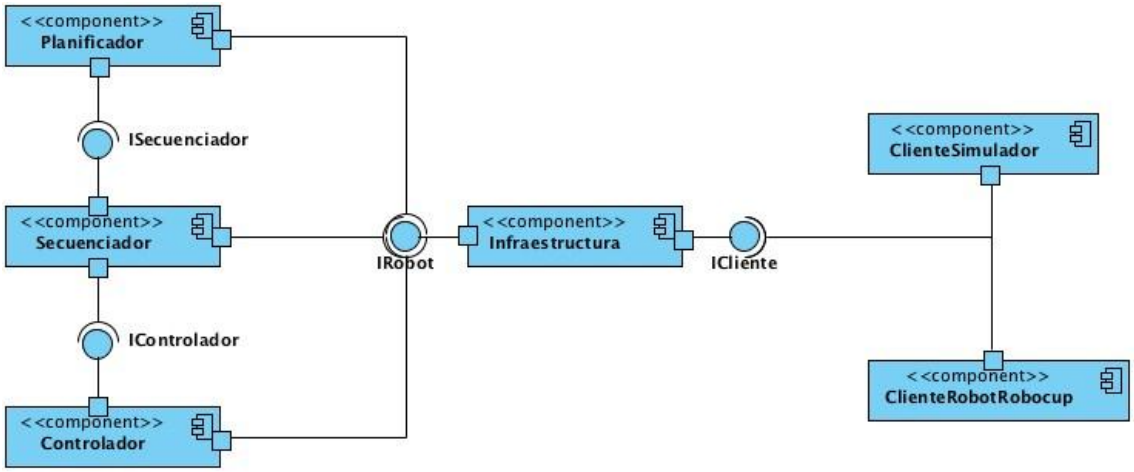


Ilustración 49: arquitectura 3T [31] [36]

El subsistema Infraestructura es el que hace uso de la arquitectura para cualquier robot con características similares al Pioneer 3-DX. En este caso ha sido necesario adaptarlo para los robots de la RoboCup [17] SSL [9], por lo tanto, se ha incluido una interfaz Robot y una clase RobotRoboCup para adaptar la arquitectura a las características de estos robots.

El siguiente componente es el controlador, el cual se muestra en la siguiente imagen:

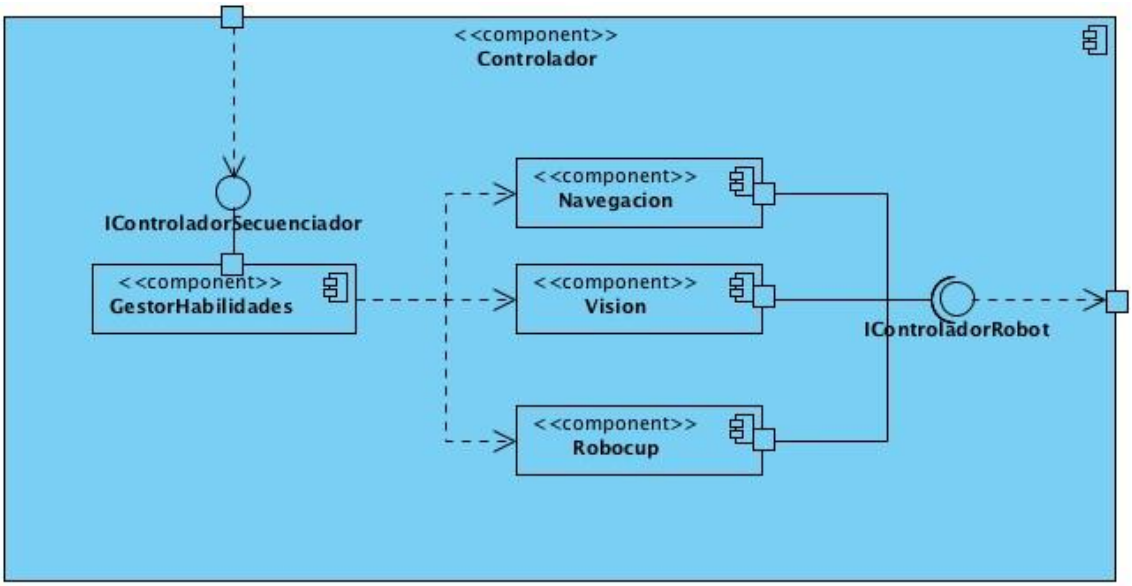


Ilustración 50: controlador

Este componente está compuesto por 4 subcomponentes. El primero de ellos es el Gestor de Habilidades, el cual se encarga de enviar al secuenciador acciones necesarias para utilizar las habilidades del robot, activándolas o desactivándolas según vaya siendo necesaria su ejecución. Además, este subcomponente se encarga de controlar el hilo de ejecución del controlador, el cual gestiona sus comunicaciones y monitoriza toda la ejecución.

Los otros tres subcomponentes son los que contienen las distintas habilidades implementadas y modularizadas en subcomponentes. Por un lado, se encuentran los subcomponentes de Navegación y Visión, que ya estaban desarrollados en el PFC de José Luis Cebrián [31] y el subcomponente RoboCup [17], el cual ha sido desarrollado en el presente trabajo, en colaboración con el PFC de Alejandro Caparrós Hernando [32]. En el subcomponente RoboCup [17] se han incluido todas las habilidades desarrolladas para los robots de la liga SSL [9], tanto para los robots físicos como para el simulador grSim [7] [8].

Sería posible añadir nuevos módulos de habilidades en el futuro que agrupen habilidades con características en común.

El siguiente componente que se va a detallar es el secuenciador, el cual se muestra en el siguiente diagrama:

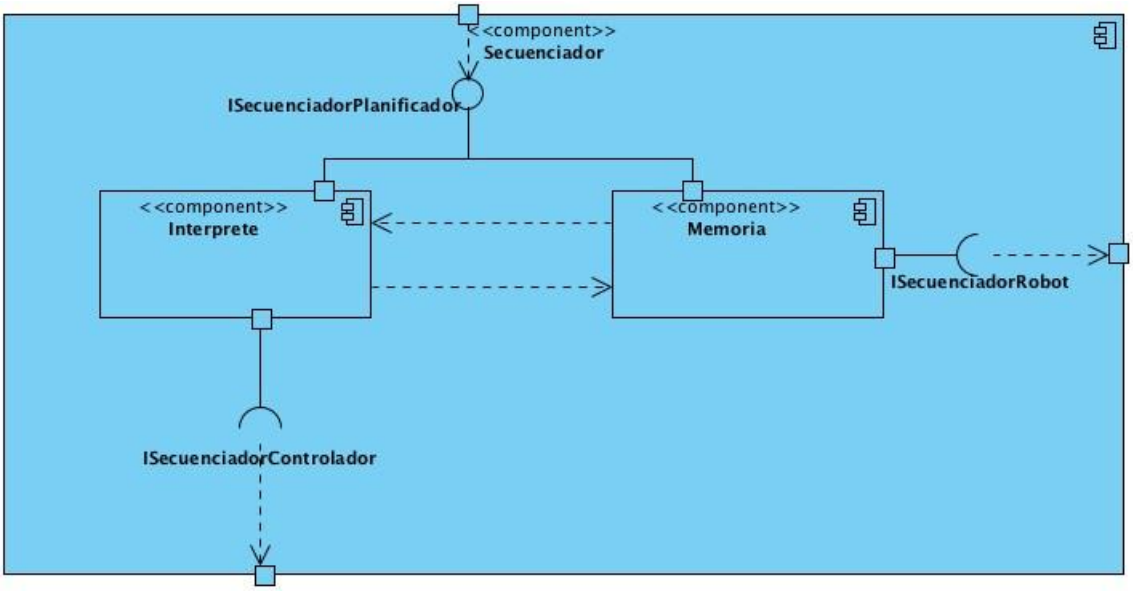


Ilustración 51: secuenciador

Este componente está compuesto por el subcomponente Intérprete y el subcomponente Memoria. El intérprete recibe todos los planes del secuenciador y los interpreta convirtiendo las tareas y las acciones en habilidades que ejecutará el controlador. Además, activa y desactiva conjuntos de habilidades que siguiendo la secuenciación permite al robot cumplir con la meta definida.

Por otro lado, la memoria almacena planes utilizados anteriormente y el resultado de estos. Monitorizando esta información, es posible realizar ciertas modificaciones en el plan y así ayudar al secuenciador en la secuenciación de tareas.

El último componente que forma la arquitectura del sistema es el Planificador, el cual se muestra en la siguiente imagen:

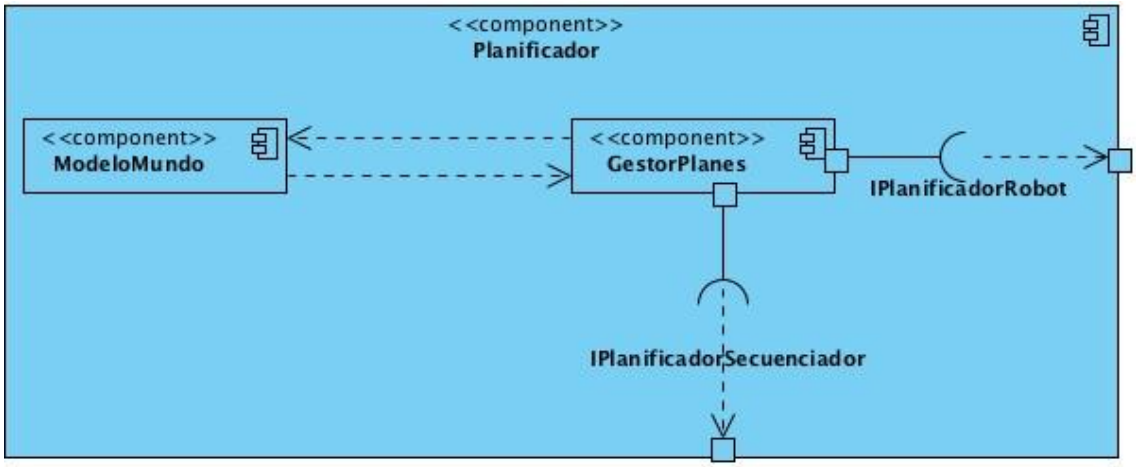


Ilustración 52: planificador

Este componente no se ha tenido en cuenta para el desarrollo del sistema, dado que no ha sido necesario realizar planes, y por lo tanto, se explicará muy brevemente su funcionamiento.

Está formado por dos subcomponentes que funcionan de forma conjunta para crear planes que utilizará el robot: el gestor de planes y el modelo del mundo.

Por un lado, se encuentra el Gestor de Planes, el cual incluye un planificador genérico que utiliza un algoritmo de búsqueda para la creación de planes a partir de las metas definidas. El otro subcomponente es el Modelo del Mundo, es el encargado de representar el entorno en el que se sitúa el robot.

3.5.2. Diseño detallado del Sistema

A partir del diseño arquitectónico se detalla el diseño del sistema. Para ello se muestran diagramas de clases, en los que se muestran las clases más relevantes y donde se puede ver como se comunican las distintas clases del sistema.

En primer lugar se detalla el componente Infraestructura, el cual se muestra en la siguiente imagen:

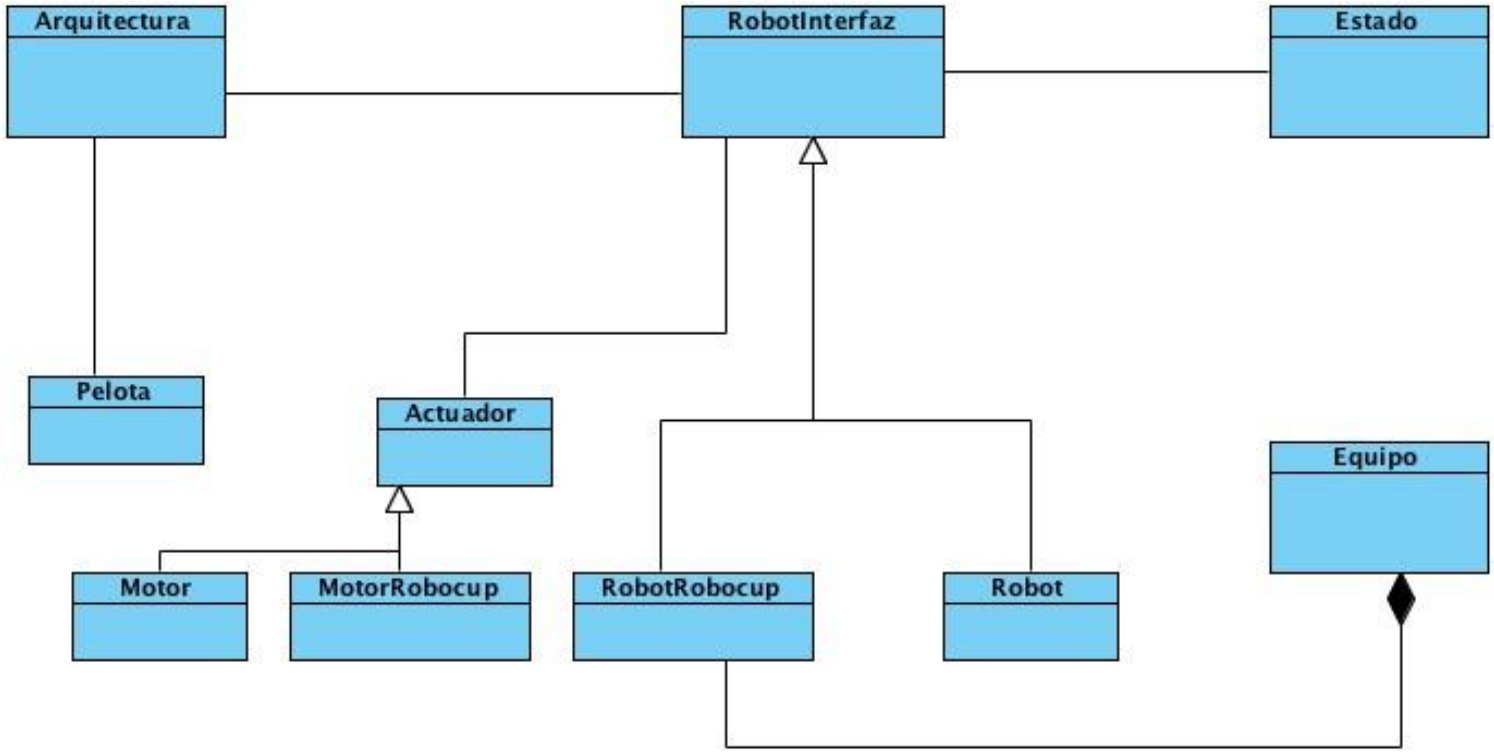


Ilustración 53: diagrama de clases de Infraestructura

Como puede observarse en el anterior diagrama, se ha definido una interfaz llamada RobotInterfaz de la que heredan dos clases, la clase Robot y la clase RobotRoboCup. La clase Robot era la que se creó en el PFC anterior [31] y se ha creado la clase RobotRoboCup, en la cual se incluyen características propias de los robots que forman los equipos de la RoboCup. Estas características son, por ejemplo, si este robot es portero, el equipo al que pertenecen los robots, la posesión de la pelota o el identificador dentro del propio equipo, ya que cada robot consta de un número identificativo dentro de su equipo.

Las clases que implementan RobotInterfaz permiten acceder al resto de la arquitectura, a su estado y al resto de componentes. La clase Arquitectura es la que se encarga de ejecutar todos los hilos correspondientes a las capas del sistema de controlador, secuenciador y planificador.

El siguiente componente es el Controlador, el cual se muestra en la siguiente imagen:

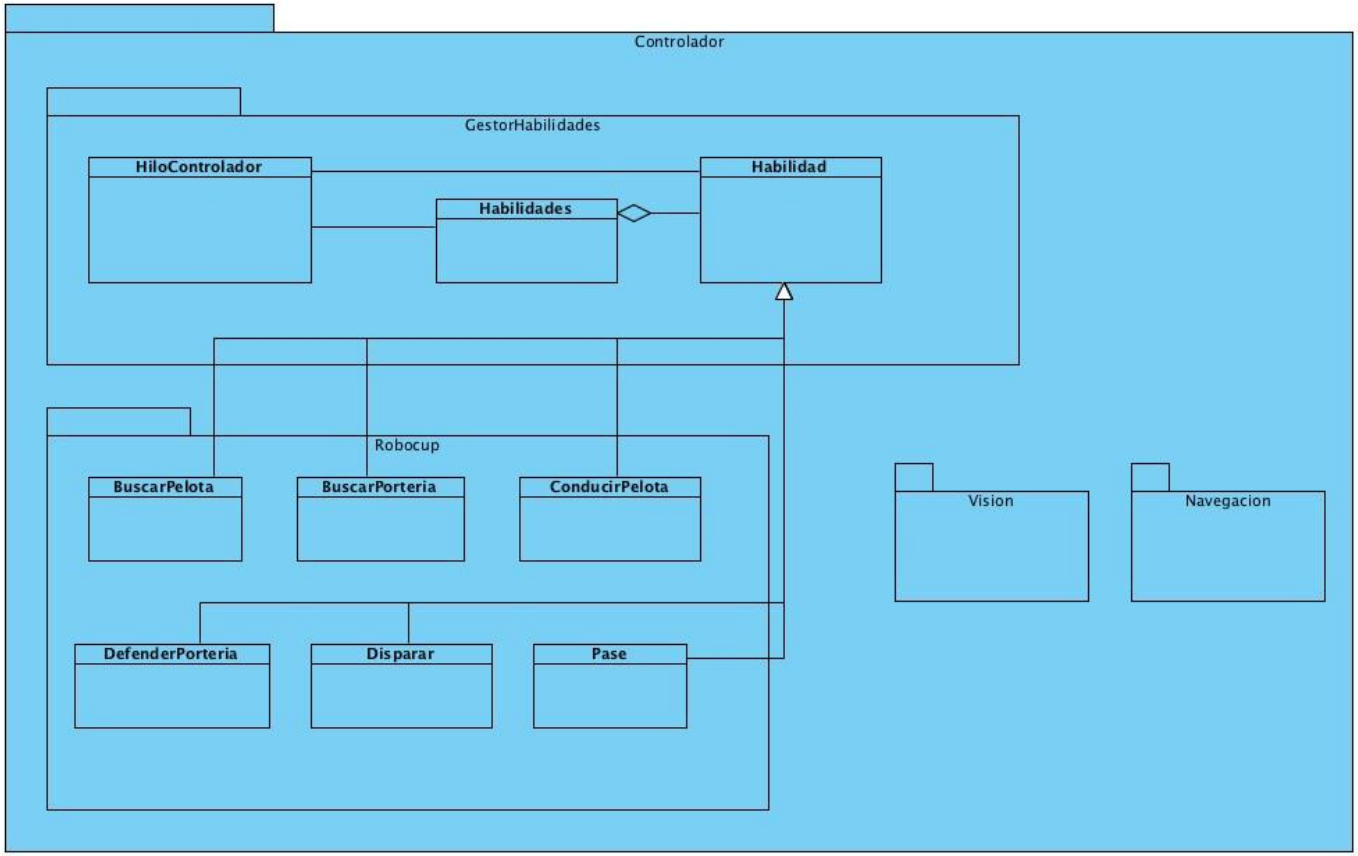


Ilustración 54: diagrama de clases de Controlador

Este diagrama del componente Controlador, este se divide a su vez en distintos subcomponentes, los cuales son GestorHabilidades, RoboCup, Visión y Navegación. Esta parte se corresponde con la capa reactiva de la arquitectura.

Los subcomponentes Visión y Navegación contienen habilidades implementadas anteriormente en el PFC de José Luis Díaz Cebrián [31] y, por lo tanto, no se prestará atención a ellos.

El subcomponente GestorHabilidades contiene la clase HiloControlador, que es el hilo de ejecución de esta capa, además, esta clase proporciona todos los métodos necesarios para que el Secuenciador haga uso de ellos.

De la clase Habilidad heredan todas las clases del resto de componentes, aunque en este caso solo se presta atención a las habilidades del subcomponente RoboCup. A su vez, la clase Habilidades es un agregado de objetos de tipo Habilidad. Estas clases son utilizadas por HiloControlador para poder hacer uso de las habilidades de lasque el robot dispone.

Dado que este proyecto se ha basado en el desarrollo de habilidades para los robots utilizados en la liga SSL [9] de la competición de la RoboCup [17], el componente RoboCup es el que tiene mayor peso en este proyecto. En este componente se han implementado todas las habilidades que realizará el robot. Dado que estas habilidades heredan de la clase Habilidad, todas implementan los métodos ejecutar() y comprobarExito(), en los cuales se define el comportamiento que deben cumplir los robots y las condiciones de éxito para que estas habilidades se den por realizadas respectivamente.

El siguiente componente es el Secuenciador, la capa intermedia de la arquitectura. Este componente se detalla en el siguiente diagrama:

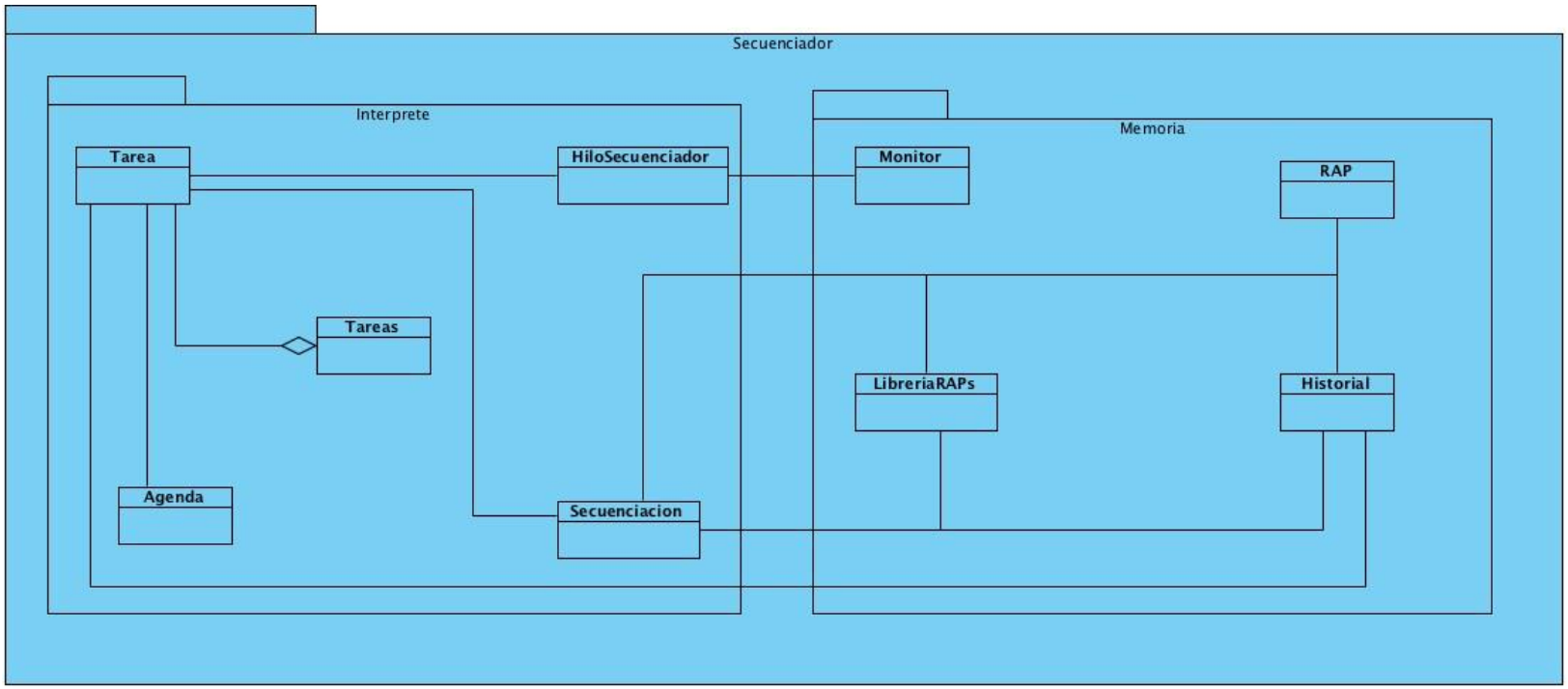


Ilustración 55: diagrama de clases de Secuenciador

Este componente trabaja a nivel reactivo y deliberativo y se divide en los subcomponentes Intérprete y Memoria.

El subcomponente Intérprete se encuentra la clase HiloSecuenciador, cuyo funcionamiento es muy similar al de la clase HiloControlador, es decir, contiene el hilo de ejecución de esta capa. Esta clase hace uso de la clase Agenda, esta es una agenda de tareas para almacenar los planes de la capa superior como elementos más utilizables que son objetos de la clase Tarea.

La clase Secuenciación se encarga de realizar las operaciones necesarias para secuenciar las tareas y asignarles conjuntos de habilidades.

Por otro lado, la clase Monitor monitoriza la ejecución del sistema y el estado del robot para informar de los cambios a la clase HiloSecuenciador y, con ello, volver a planificar la agenda si esto fuera necesario.

El subcomponente Memoria también contiene una clase Historial que almacena los planes pasados y la clase LibreriaRAPs, que contiene los RAP definidos para la arquitectura y las tareas que pueden llevar a cabo los robots. En esta clase se incluyeron los RAP utilizados para ejecutar las habilidades de los robots de la RoboCup [17].

La última capa de la arquitectura es el componente Planificador, el cual queda reflejado en el siguiente diagrama:

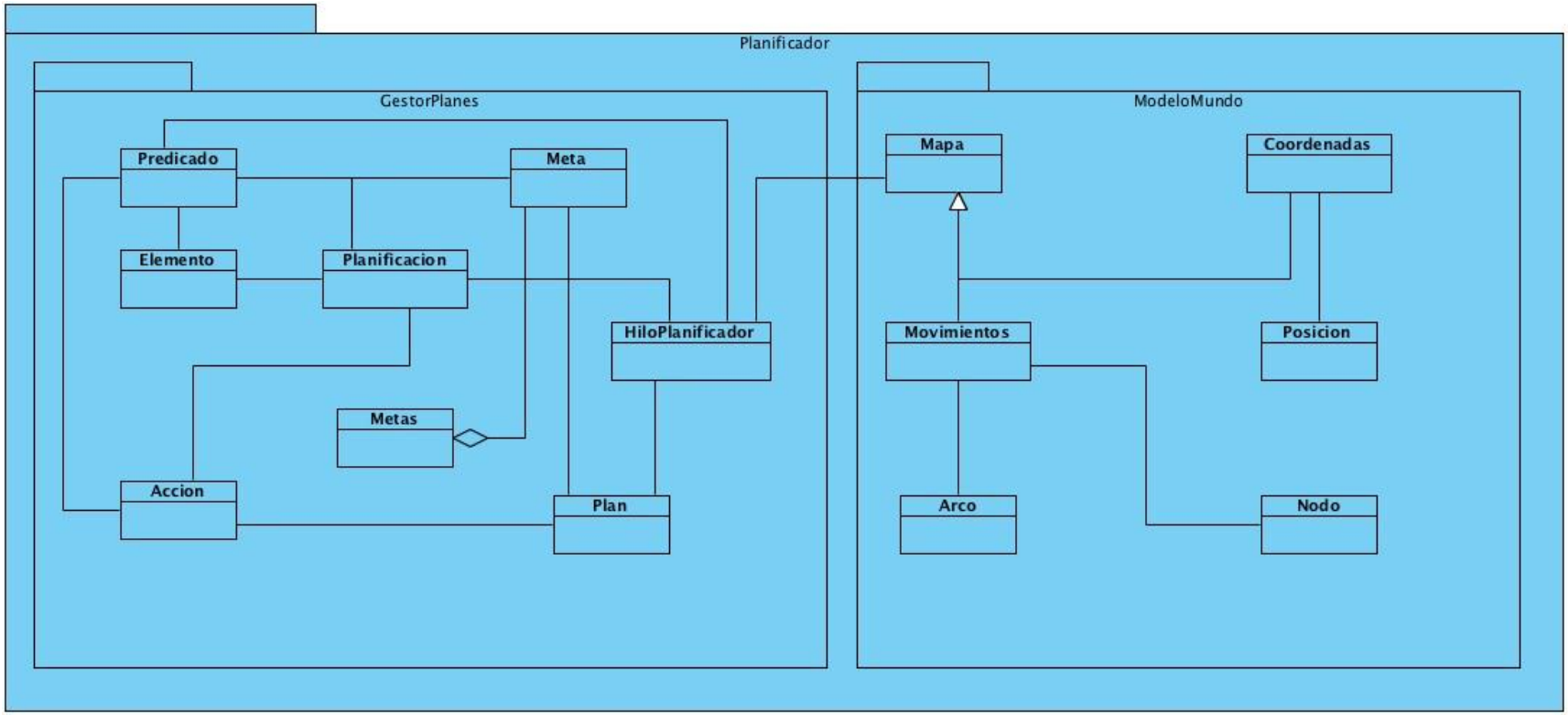


Ilustración 56: diagrama de clases de Planificador

Este componente se divide en dos subsistemas: el GestorPlanes y el ModeloMundo. El modelo del mundo contiene los elementos que representan el entorno del robot. Este subcomponente no es utilizado en este proyecto, dado que el entorno es el terreno de juego y no es necesario incluir ningún tipo de información adicional aparte de la que recibe el sistema.

Respecto al subcomponente GestorPlanes contiene una clase llamada HiloPlanificador, que representa el hilo de ejecución de esta capa de la arquitectura. El planificador está compuesto por las clases Planificación, Predicado, Acción y Elemento. La clase Planificación tiene una función similar a la clase Secuenciación del componente Secuenciador, se encarga de elaborar un plan a partir de unas metas definidas y unas acciones definidas en la clase Plan. Para ello hace uso de predicados, acciones y los elementos que los componen. Las acciones pueden asociarse a las tareas por medio de las submetas, lo que permite escoger RAP y habilidades para ejecutarlas.

El planificador no es utilizado en el proyecto como tal, simplemente se definen unas metas y unas acciones para que se ejecuten secuencias de habilidades y de esta forma los robots puedan ejecutar estas acciones.

3.5.3. Diagrama de secuencia

A continuación se ofrece el diagrama de secuencia que describe las interacciones que se producen en el sistema para llegar a ejecutar una habilidad:

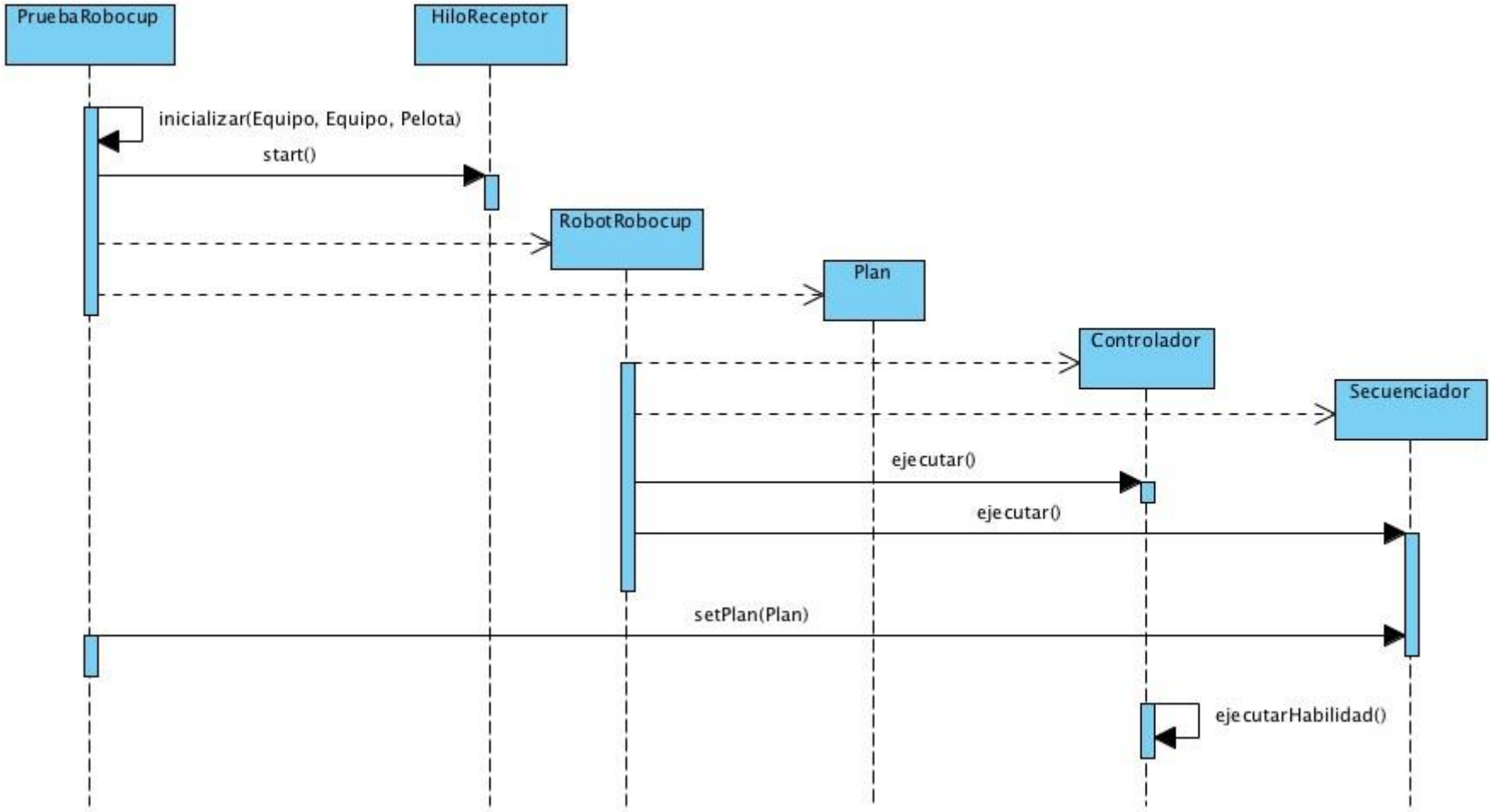


Ilustración 57: diagrama de secuencia

Como se puede apreciar en la imagen superior, la clase PruebaRobocup funciona como clase principal y se encarga de realizar las llamadas necesarias para ejecutar un plan, ya sea para el simulador o para el robot físico.

Esta clase, en primer lugar, inicializa las posiciones de los robots de los dos equipos que participan en el juego y la posición de la pelota. Posteriormente, esta clase inicializa el hilo receptor, que se encarga de ejecutar el hilo de recepción de información de la clase HiloReceptor, el cual actualizará continuamente las posiciones de los robots y de la pelota. De esta forma se consigue una alta precisión en la ejecución de todas las habilidades.

Una vez se crea un objeto de tipo RobotRobocup, se inicializan los hilos Controlador y Secuenciador y se asigna al secuenciador el plan creado para llevar a cabo su ejecución.

El controlador recibirá las habilidades que se ejecutarán según el plan establecido y las habilidades se encargarán de enviar los comandos pertinentes a los robots físicos o a los robots del simulador, según corresponda.

3.6. Habilidades

El grueso principal del presente PFC consiste en la programación de las habilidades básicas de robots para la liga SSL [9] de la RoboCup [17]. Por eso mismo, ha sido necesario realizar una combinación de estas habilidades con otro PFC, aun no publicado, perteneciente a Alejandro Caparrós Hernando [32].

Se detallará a continuación una tabla con las habilidades elaboradas y el porcentaje que corresponde al trabajo que se ha desempeñado. Para conocer al detalle las habilidades con porcentaje mayoritario realizado por Alejandro Caparrós Hernando, se recomienda consultar su trabajo una vez publicado [32]. Además, citar que desde este momento, el color azul oscuro hará referencia al trabajo desempeñado por Alejandro Caparrós Hernando, mientras que el color azul claro corresponderá al trabajo elaborado por Verónica Raspeño Gutiérrez:

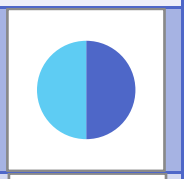
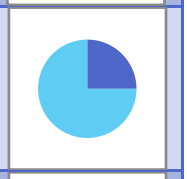
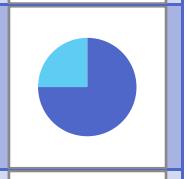
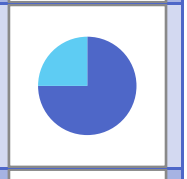
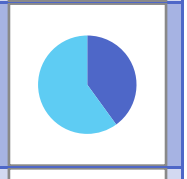
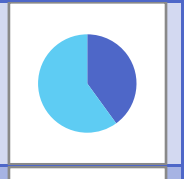
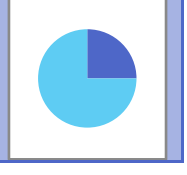
Rol	Habilidad	% PFC Formación de defensa	% PFC Formación de ataque	Gráfico
Todos	Giro óptimo	50%	50%	
Portero	Buscar portería	25%	75%	
	Defender portería	75%	25%	
Jugador defensor	Defender portería	75%	25%	
Jugador atacante	Buscar pelota	40%	60%	
	Conducir pelota	40%	60%	
	Pase	60%	40%	
	Esquivar	60%	40%	
	Disparar	25%	75%	

Tabla 47: habilidades elaboradas para la liga SSL [9]

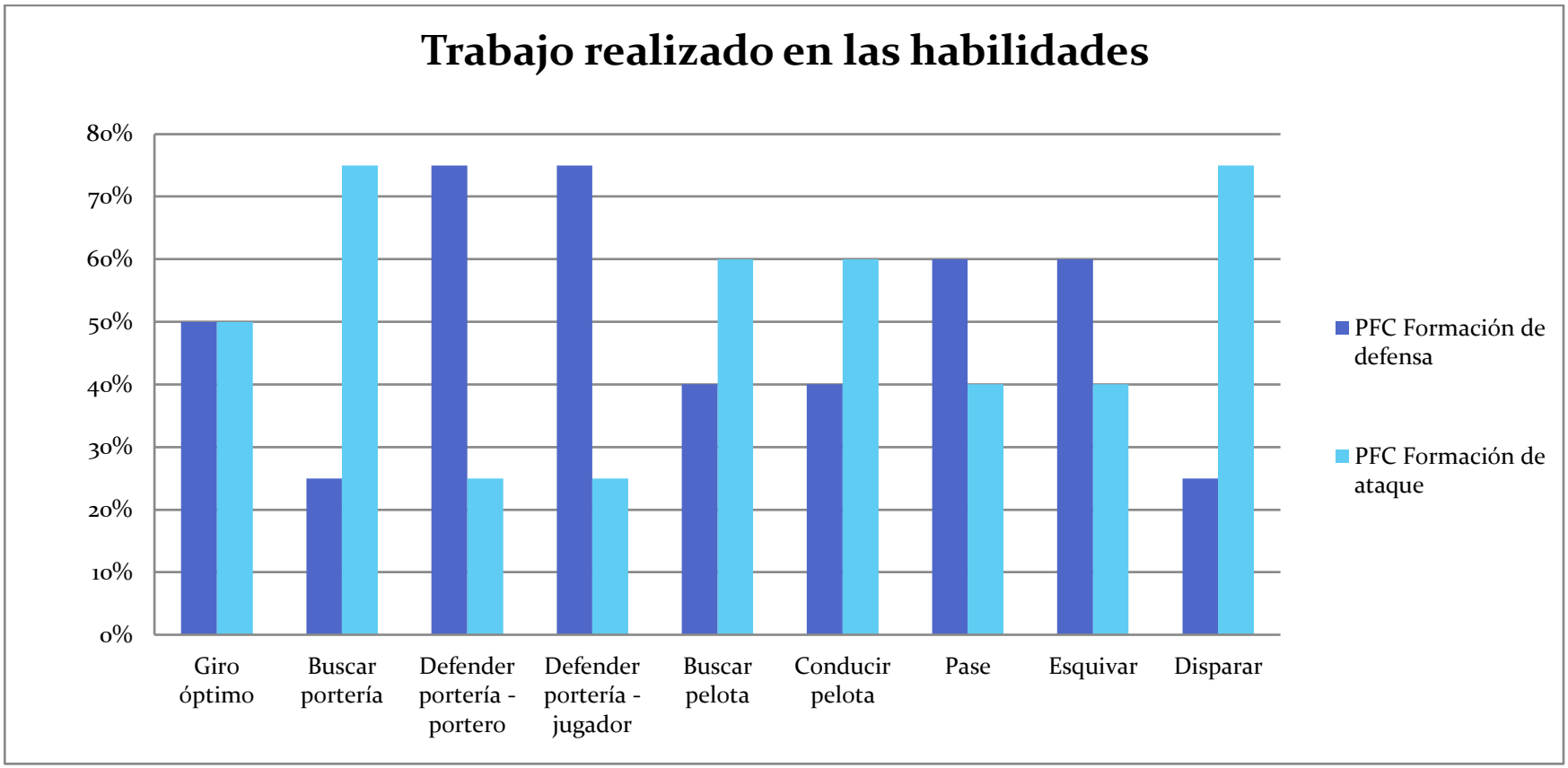


Ilustración 58: gráfica de habilidades elaboradas para la liga SSL [9]

A lo largo de los siguientes puntos se desarrollarán las habilidades únicamente que tienen mayor o igual porcentaje por parte de Verónica Raspeño Gutiérrez, siendo estas: giro óptimo, buscar portería en portero, buscar pelota, conducir pelota y, por último, disparar.

3.6.1. Giro óptimo (todas las habilidades)

Esta característica se ha otorgado a todas las habilidades creadas que incluyan giros, tanto giro sobre el robot mismo como giros alrededor de un objeto.

El giro óptimo significa girar hacia la derecha o hacia la izquierda según el camino más corto. La forma de calcularlo es la siguiente: el robot previamente sabe a qué punto se quiere dirigir, por ejemplo, si va hacia la portería o hacia las coordenadas de la pelota que le ha enviado el sistema de visión, calcula internamente su orientación y la orientación del lugar hacia donde quiere ir respecto a él.

Para calcular la orientación del robot simplemente hay que recurrir al paquete que envía el simulador grSim [7] [8] o, en su defecto, el SSL-Vision [11] en los robots físicos.

Para calcular la orientación del objetivo del robot (orientado hacia la portería, coordenadas concretas o pelota) hay que realizar las operaciones trigonométricas propias de calcular la orientación dados dos puntos, tomando como referencia el punto del objeto. Los dos puntos que se cogen son únicamente el de las x y el de las y, correspondientes a la segunda dimensión de tal manera:

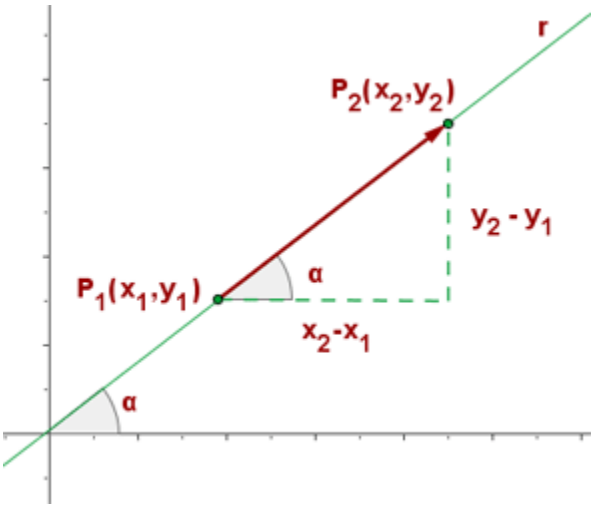


Ilustración 59: cálculo dirección entre dos puntos

Una vez calculada la orientación, se obtiene la forma óptima para llegar a ese punto, girando para la derecha o para la izquierda. Lo difícil en este aspecto es controlar las coordenadas y orientación, ya que pueden ser positivas o negativas y eso interfiere en el resultado final.

Dentro del giro óptimo, existen dos variantes: giro del robot sobre sí mismo, giro del robot en torno a otro objeto.

El giro del robot sobre sí mismo se realiza en la mayoría de las ocasiones, siempre que el robot quiere ir a una posición y su orientación no es la correcta. Por ejemplo, en la siguiente imagen, si el robot quisiera ir hacía la pelota, realizaría un giro como representa la flecha azul:

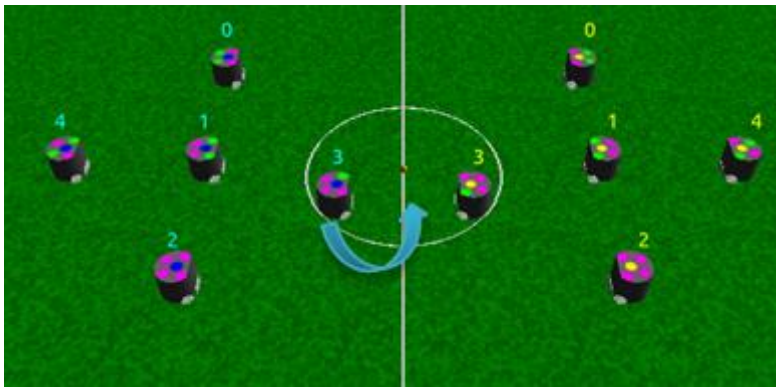


Ilustración 60: giro óptimo del robot sobre sí mismo

Por otro lado, existe el giro sobre un objeto, que siempre es la pelota. Esto se da cuando el robot va hacia la pelota y luego quiere situarse orientado a, por ejemplo, la portería y mantener la pelota. La forma más eficiente de hacerlo sin riesgo a perder la pelota es que el robot gire siendo el centro del giro la pelota, orientándose a las coordenadas objetivo, hasta situarse. Un ejemplo de giro es el siguiente:

3.6.2. Buscar portería en portero

Esta habilidad pertenece al robot portero, este robot se asigna de forma inmediata antes de comenzar el partido, como se explica detalladamente en el "ANEXO V: Reglamento 2011".

Una de las habilidades propias del robot portero es que sea capaz de ir a la portería, ya que no se va a poder mover del área de penalti. La habilidad se encarga primeramente de averiguar la portería que le corresponde al robot. Por eso mismo, mira el color al que se le ha asignado y la portería que está estipulada para ese equipo, a partir de ello, el robot va a la portería.

Para ir a la portería, primero se sitúa en dirección hacia ella, girando sobre sí mismo, para después dirigirse hacia el área de penalti. Internamente, el robot calculará la

trayectoria que tiene que recorrer, definiendo una línea entre su posición y la portería, de tal manera que sepa hacia dónde tiene que dirigirse.

Una vez que llega a la portería, de nuevo gira sobre sí mismo para colocarse de cara al terreno de juego y poder así defender con mayor precisión, aunque la defensa como tal no está incluida en esta habilidad.

El robot previamente conoce qué portería le corresponde, cuando comienza el partido se asignan los colores y el campo que le pertenece a cada equipo, por lo que el robot portero debe ser consciente de a qué portería debe dirigirse.

Se muestra a continuación el movimiento que sigue el robot. En este caso, el ejemplo viene dado en el equipo azul y el portero es el robot número 4:

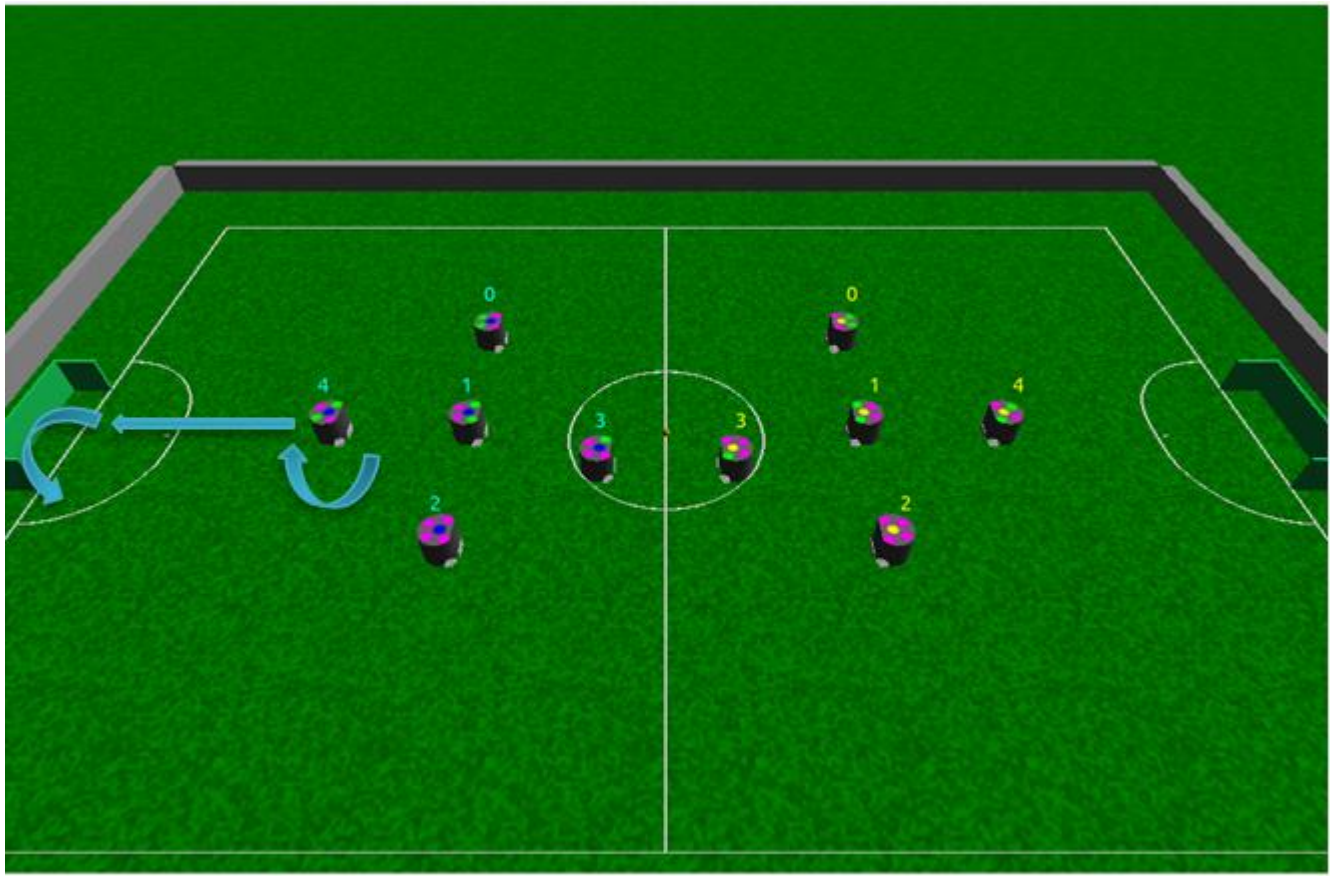


Ilustración 61: habilidad buscar portería

3.6.3. Buscar pelota

Esta habilidad es una de las más completas, ya que requiere diferentes movimientos encadenados que logran realizar la actividad completa. Por lo tanto, lo más complejo de esta habilidad es que el robot sea capaz de hacer todos los movimientos seguidos, de forma fluida.

Lo primero que hace esta habilidad es buscar dónde está la pelota. Para ello, comprueba las coordenadas de la pelota a través del paquete que recibe del sistema de visión, en este caso simulado por el simulador grSim [7] [8].

Posteriormente, el robot gira sobre sí mismo para situarse orientado hacia la pelota. Se conoce la orientación del robot, ya que es un dato incluido en el paquete del robot, por lo que se calcula la orientación objetivo a partir de las coordenadas del robot y de la pelota. Por último, se calcula el giro óptimo y el robot procede rotar sobre sí mismo.

Tras ello, se dirige en línea recta hacia la pelota. Como puede que el robot tenga cierta desviación, porque en las posiciones existe cierto redondeo, mientras avanza, va corrigiendo.

Cuando llega a la pelota, el robot gira en torno a esta para orientarse frente a la portería, en busca de meter un gol. De esta habilidad, se pasaría a la habilidad del pase o a la de conducir portería, según conviniese.

En la siguiente imagen se muestra el comportamiento del robot realizando la habilidad de buscar paso, en este caso es el robot o el que realiza la acción:

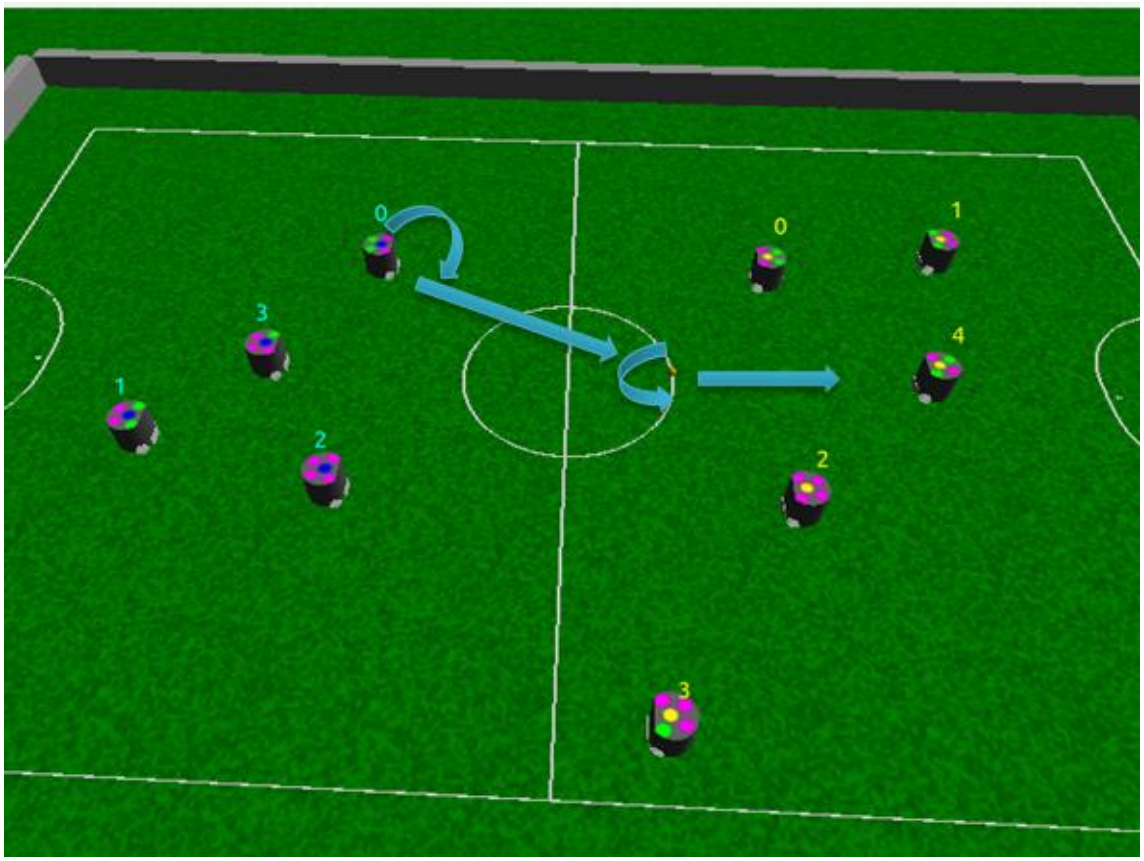


Ilustración 62: habilidad buscar pelota

3.6.4. Conducir pelota

Conducir la pelota es una habilidad que se basa meramente en que el robot mantenga la posesión de la pelota y que se dirija hacia la portería para poder meter un gol.

Para esta habilidad, es necesario que el robot previamente esté en posesión de la pelota, por lo que la habrá tenido que obtener a través de otra habilidad, como la habilidad de buscar pelota.

Tras estar en posesión, el robot intenta avanzar con la pelota, pretendiendo en todo momento llegar a la portería para poder conseguir meter un gol.

El robot debe tener la constancia en todo momento de que está en posesión de la pelota, esto se calcula debido al paquete que se recibe del sistema de visión. Este paquete no solo manda la posición de los robots sino que, además, manda dónde está colocada la pelota. Esta habilidad debe observar dónde está la pelota y si se encuentra cerca del robot y, no solo eso, si está situado en la parte delantera del robot, a mitad aproximadamente de él, para poder asegurar que se encuentra cerca del disparador del robot.

En la siguiente ilustración se ve la habilidad en acción, en este caso es el robot 3 quien posee la pelota y la conduce a lo largo del terreno de juego:

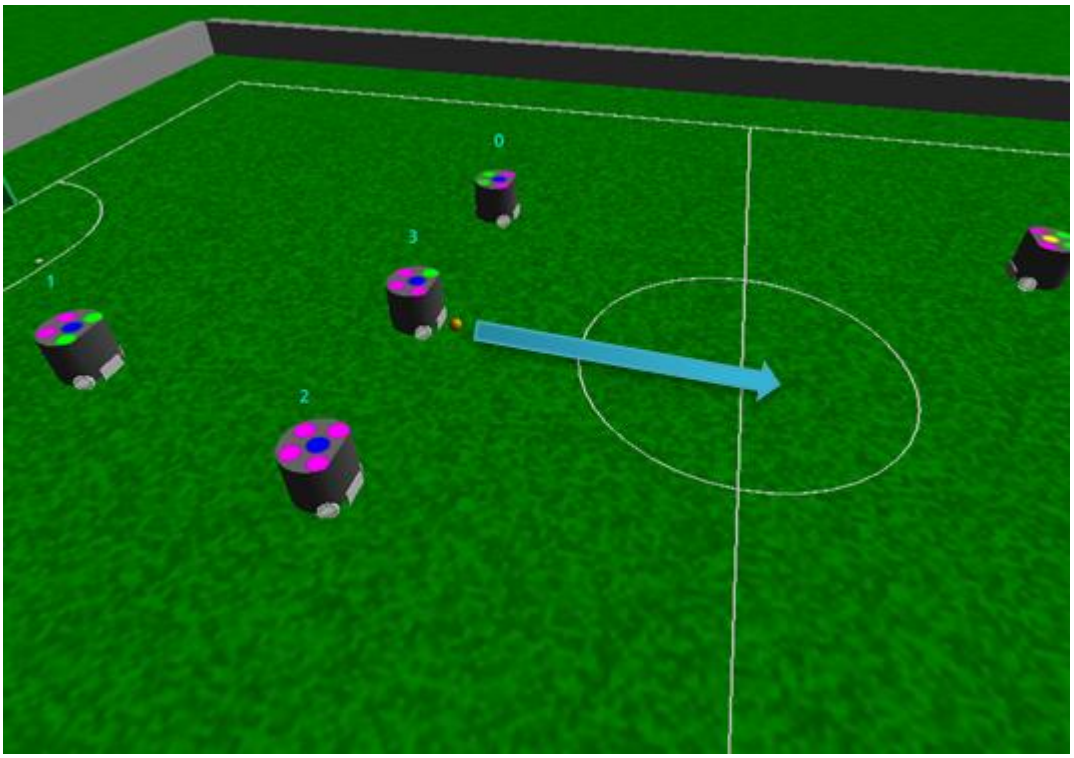


Ilustración 63: habilidad conducir pelota

3.6.5. Disparar

La habilidad de disparar consiste en lanzar la pelota cuando el robot la tiene en su poder, con el objetivo de marcar un gol. Por lo tanto, el robot tiene que estar lo suficientemente cerca de la portería como para poder meter un gol.

Hasta que no se cumplen estas dos exigencias, es decir, que el robot esté cerca de la portería y que el robot tenga posesión de la pelota, no se activa la habilidad de disparar.

Hay dos opciones de tiro: la pelota elevada o sin elevar. La primera opción equivale a que el robot utiliza su disparador elevando la pelota, por lo que esta realiza un movimiento parabólico, pudiendo superar al portero y meter un gol. La segunda opción es un tiro al ras del suelo, el disparador lo único que hace es golpear la pelota para darle impulso y que pueda llegar a la portería.

A continuación se muestra una ilustración a modo de ejemplo que sitúa al robot justo enfrente de la portería y tiene el balón en su posesión, en el momento en el que se dispone a lanzar, sin elevar la pelota:

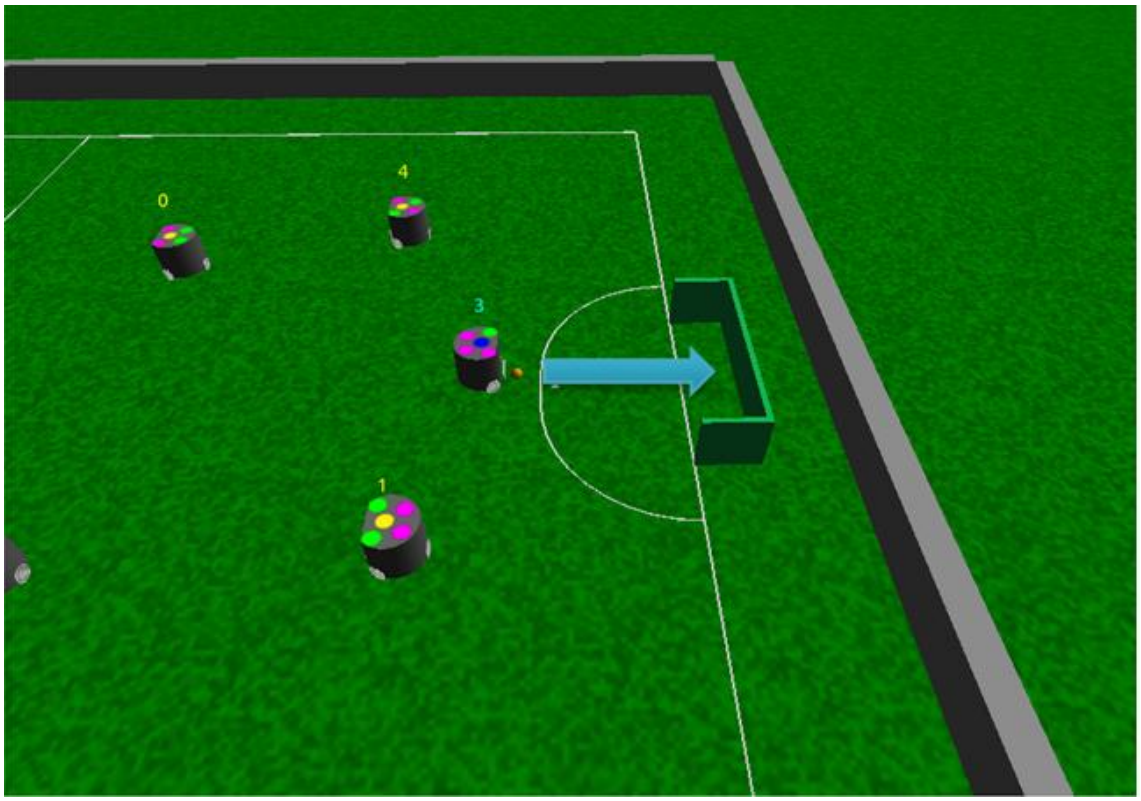


Ilustración 64: habilidad disparar a portería sin elevar pelota

Asimismo, se muestra un ejemplo elevando la pelota:

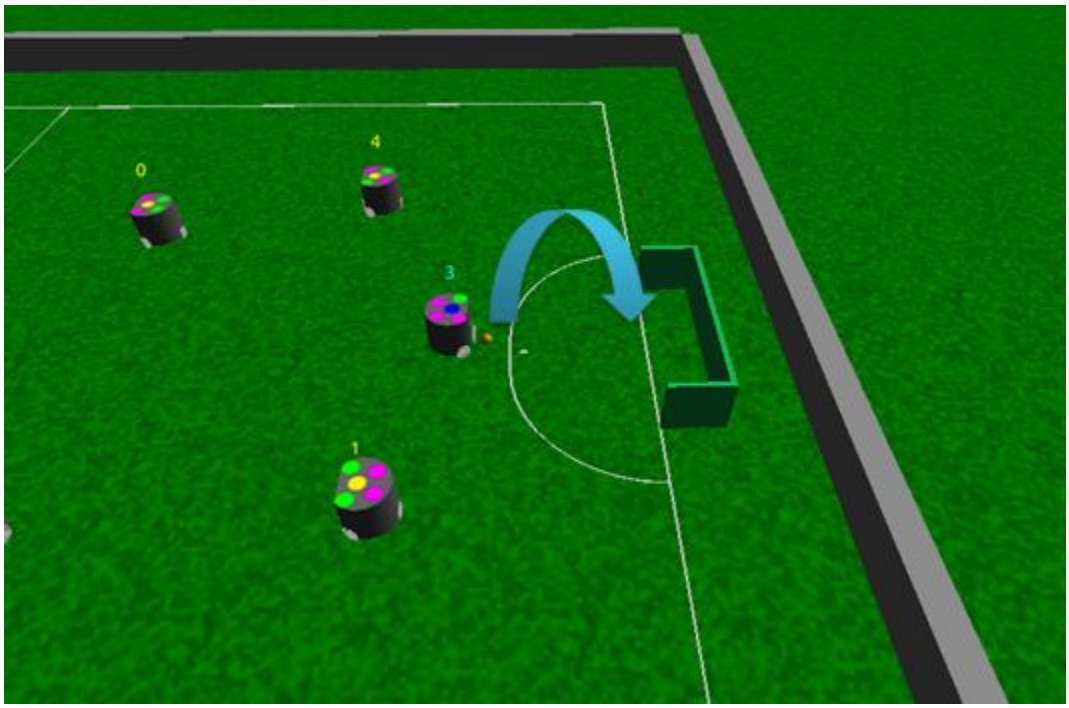


Ilustración 65: habilidad disparar a portería elevando pelota

3.7. Aplicación en robots físicos

Aunque este PFC está orientado a la creación de las bases para elaborar un equipo de fútbol para la SSL [9], así como el desarrollo de las habilidades básicas simuladas, se ha ampliado para poder probar lo desarrollado en los robots físicos.

En este punto se desarrollarán las modificaciones necesarias en el simulador grSim [7] [8] para lograr simular exactamente el robot físico y la elaboración de las conexiones establecidas.

3.7.1. Especificaciones del robot físico

Las especificaciones del robot físico se basan en dos proyectos comentados anteriormente [34] [35] que los han diseñado y construido.

El cerebro del robot físico se constituye en el sistema de procesamiento, que se encarga de procesar y enviar las órdenes a la estructura. Además, este sistema tiene que

ser capaz de interactuar de forma inalámbrica con un ordenador, del cual recibirá las órdenes que debe ejecutar:

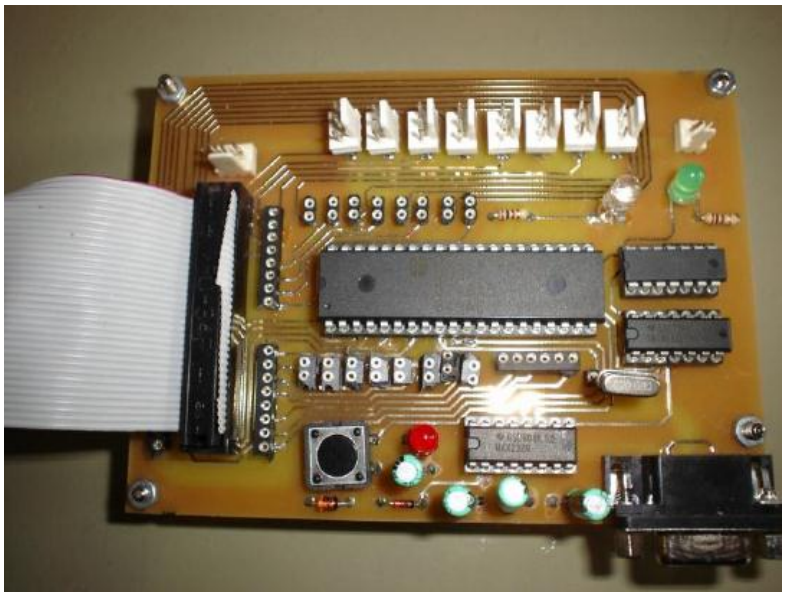


Ilustración 66: sistema de procesamiento de robot físico [34] [35]

Asimismo, el robot debe constar de un módulo de comunicación *Wi-Fi* que pueda transmitir y recibir información de forma inalámbrica, el robot utiliza el controlador comercial RCM4400-Rabbit:

Ilustración 67: controlador comercial RCM4400-Rabbit [34] [35]

Respecto a la estructura, el robot sigue la forma propia que proponen las reglas en el capítulo "2.3.3.2. Reglas", respectando las dimensiones y los colores, colocando la batería como muestra la siguiente imagen:

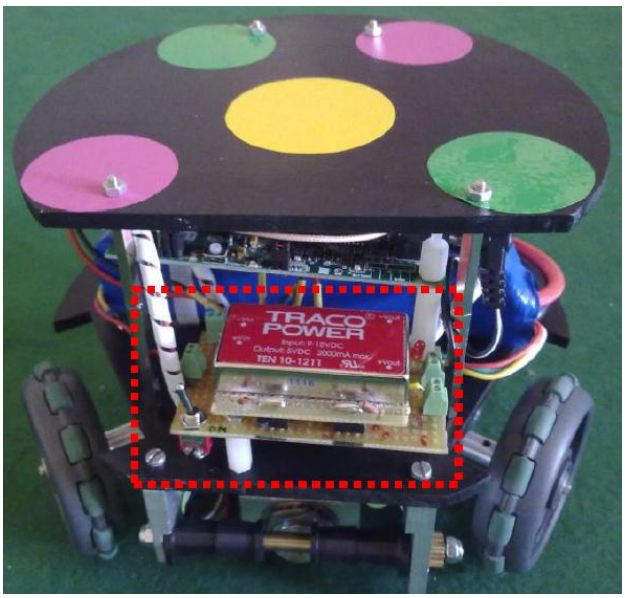


Ilustración 68: ubicación placa alimentación robot físico [34] [35]

Como decisión de los creadores del robot, se ha optado por tres ruedas en vez de cuatro, que suele ser lo común, rebajando los costes y ofreciendo, asimismo, un buen resultado. Las ruedas elegidas son:

Ilustración 69: rueda robot físico [34] [35]

Resaltar como parte de la estructura el sistema de *dribbler*, que finalmente se ha optado por una barra con forma de cilindro que permite mantener la pelota y poder regatear sin perderla, quedando de tal manera:

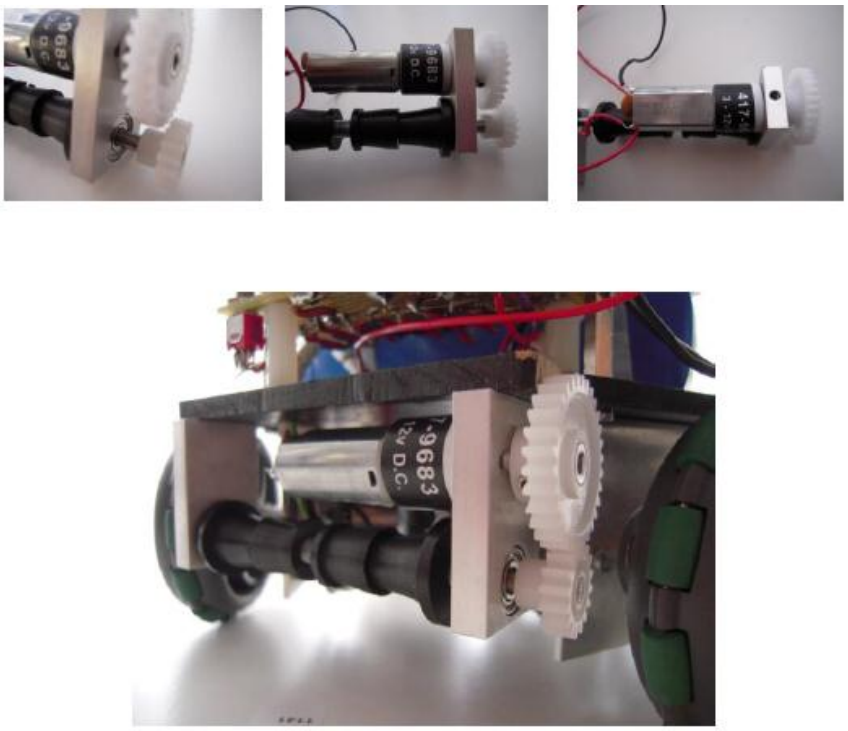


Ilustración 70: sistema de *dribbler* de robot físico [34] [35]

El resultado final con el sistema de *dribbler* colocado es el siguiente:

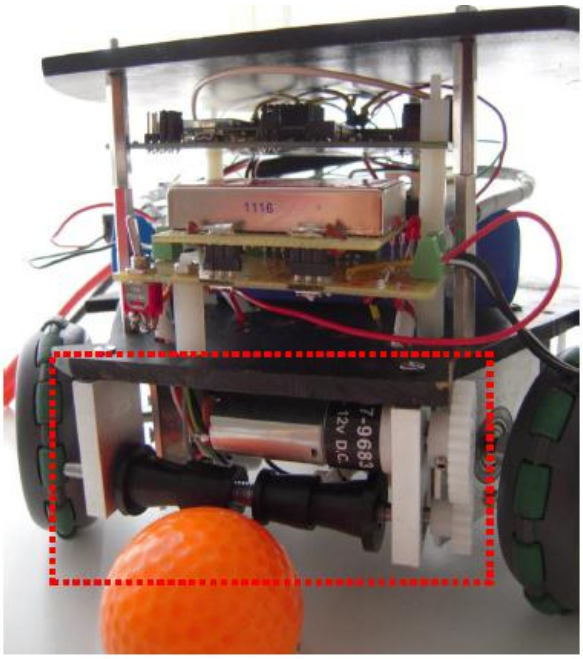


Ilustración 71: sistema de *dribbler* incorporado en el robot físico [34] [35]

Quedando como resultado final:

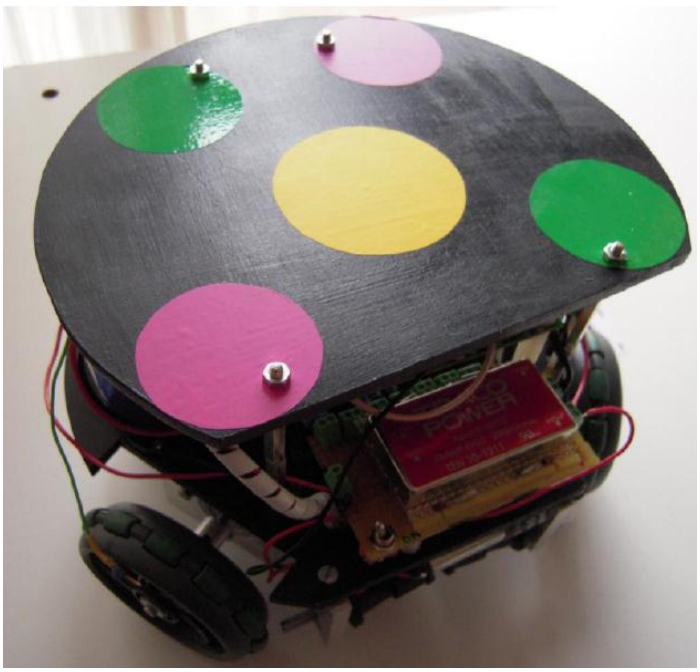


Ilustración 72: robot físico final [34] [35]

3.7.2. Modificaciones de simulador para adaptación a robots físicos

El simulador grSim [7] [8] ofrece una interfaz como se ha detallado en el punto 3.1.2. Interfaz, que incluye una representación gráfica del terreno de juego, así como del comportamiento de los robots.

Los robots simulados aportados en grSim [7] [8] constan de cuatro ruedas, debido a que el equipo Parsian Robotic [7], creador de dicho simulador, utiliza robots físicos con exactamente cuatro ruedas.

Se ofrece una imagen del simulador grSim [7] [8] que muestra cómo los robots están apoyados sobre cuatro ruedas, se puede apreciar en los dos robots centrales que están frente a la pelota, en la parte delantera tiene dos ruedas y en la parte trasera otras dos:

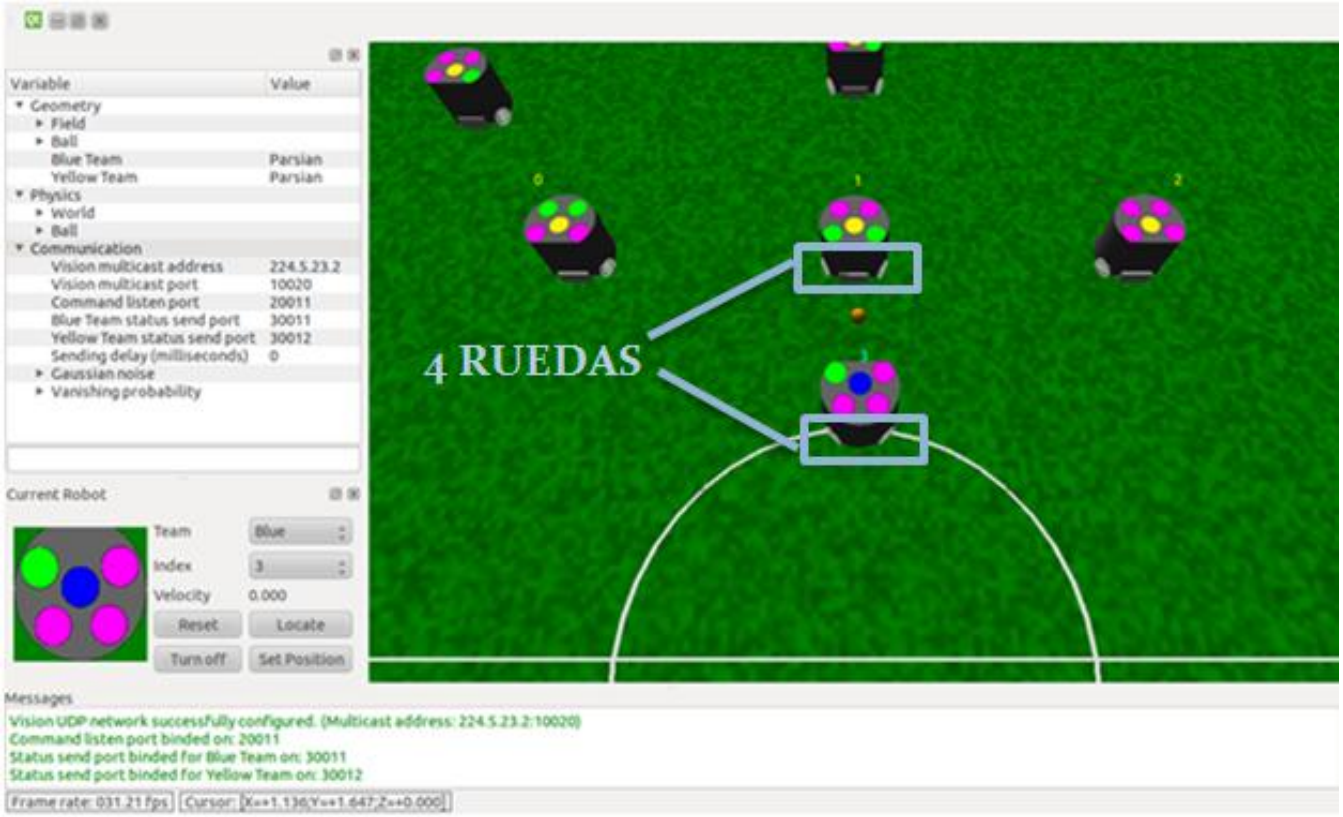


Ilustración 73: representación simulador grSim [7] [8] con robots de cuatro ruedas

Sin embargo, los robots físicos creados en la Universidad Carlos III de Madrid por el Departamento de Ingeniería de Sistemas y Automática desde el Laboratorio de Sistemas Inteligentes en los que se apoya este PFC constan únicamente de tres ruedas distribuidas como muestra el siguiente esquema:

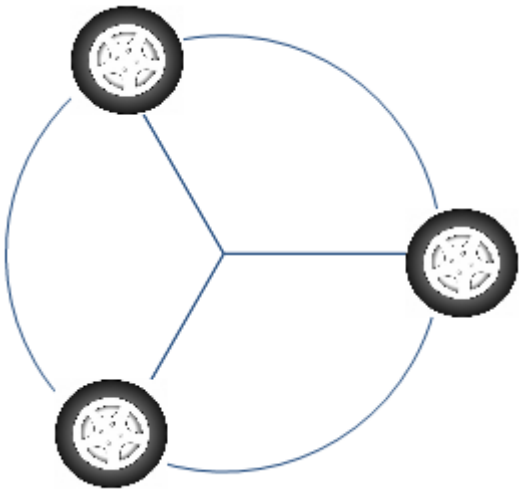


Ilustración 74: esquema distribución de ruedas de robot físico

De tal manera que la parte inferior del robot físico es:

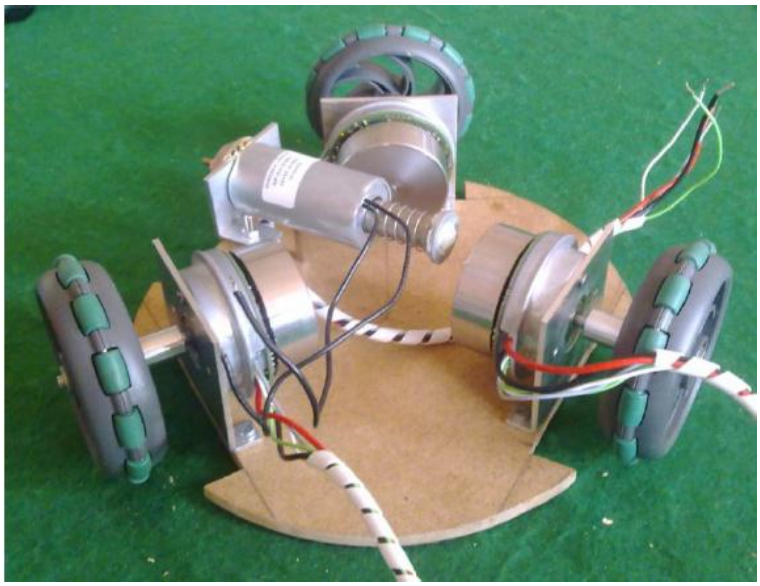


Ilustración 75: vista interior del robot [33] [34] [35]

Por este motivo se procedió a realizar ciertas modificaciones del código del simulador grSim [7] [8] para conseguir que los robots funcionaran con tres ruedas.

Para ello se suprimió la cuarta rueda y se colocaron las tres ruedas distribuidas a lo largo de la circunferencia del robot de forma equitativa, de tal manera que coincidieran con los robots físicos.

En la siguiente ilustración se puede apreciar las modificaciones realizadas en los robots, pasando de cuatro a tres ruedas:

Ilustración 76: representación simulador grSim [7] [8] con robots de tres ruedas

3.7.3. Creación de conexiones

Para la creación de conexiones entre la arquitectura diseñada para la SSL [9] y los robots físicos se ha utilizado un firmware intermediario creado por el compañero Isaac Ruiz Agrazal, del Área de Ciencias de la Computación e Inteligencia Artificial de la Universidad Carlos III de Madrid.

Esta conexión se encarga de recibir los datos de la arquitectura y procesarlos para que los robots puedan entenderlos, asegurándose que no haya ningún problema. El firmware consiste en transformar lo que se manda en el paquete, para que los robots puedan comprenderlo y actuar. Esto sucede, por ejemplo, porque los robots no utilizan las mismas unidades de velocidad.

3.7.4. Sistema de visión

Dado que los robots de la SSL [9] no cuentan con sensores externos para obtener datos del exterior y así conseguir un control reactivo, se hace uso de un sistema de visión, el cual envía a los miembros de los equipos las posiciones de la pelota y todos los robots que están participando en el partido.

En el comienzo de esta competición las normas permitían a cada equipo tener su propio sistema de visión para obtener las posiciones de los robots. Esto suponía un coste adicional a los equipos y era un factor que frenaba el progreso de la competición. Además, la mayoría de los equipos utilizaban sistemas muy similares, lo que producía pocos resultados significativos en la experimentación en este campo. Todo esto llevo a la creación de un sistema de visión global y estándar llamado SSL-Vision [11].

En 2009, el comité de la liga decidió utilizar este sistema de visión para que lo utilizasen todos los equipos que participasen en la competición. Este sistema ha sido desarrollado por varios voluntarios de los equipos participantes y actualmente está en pleno desarrollo y mejora.

El sistema de visión que utilizan los robots físicos está fundamentado en el SSL-Vision [11], un estándar proporcionado por los coordinadores de la competición RoboCup

[17]. Este funciona de forma similar a lo simulado en el simulador grSim [7] [8], sin embargo, coge los datos no de la simulación, sino de lo que recibe de la cámara.

Utiliza, asimismo, Google Protobuf [21] para codificar los datos que envía, al igual que hacía el simulador grSim [7] [8], de tal manera no haya mucha diferencia entre los robots simulados y los robots físicos.

En la siguiente ilustración se muestra una captura de la interfaz de la aplicación citada anteriormente:

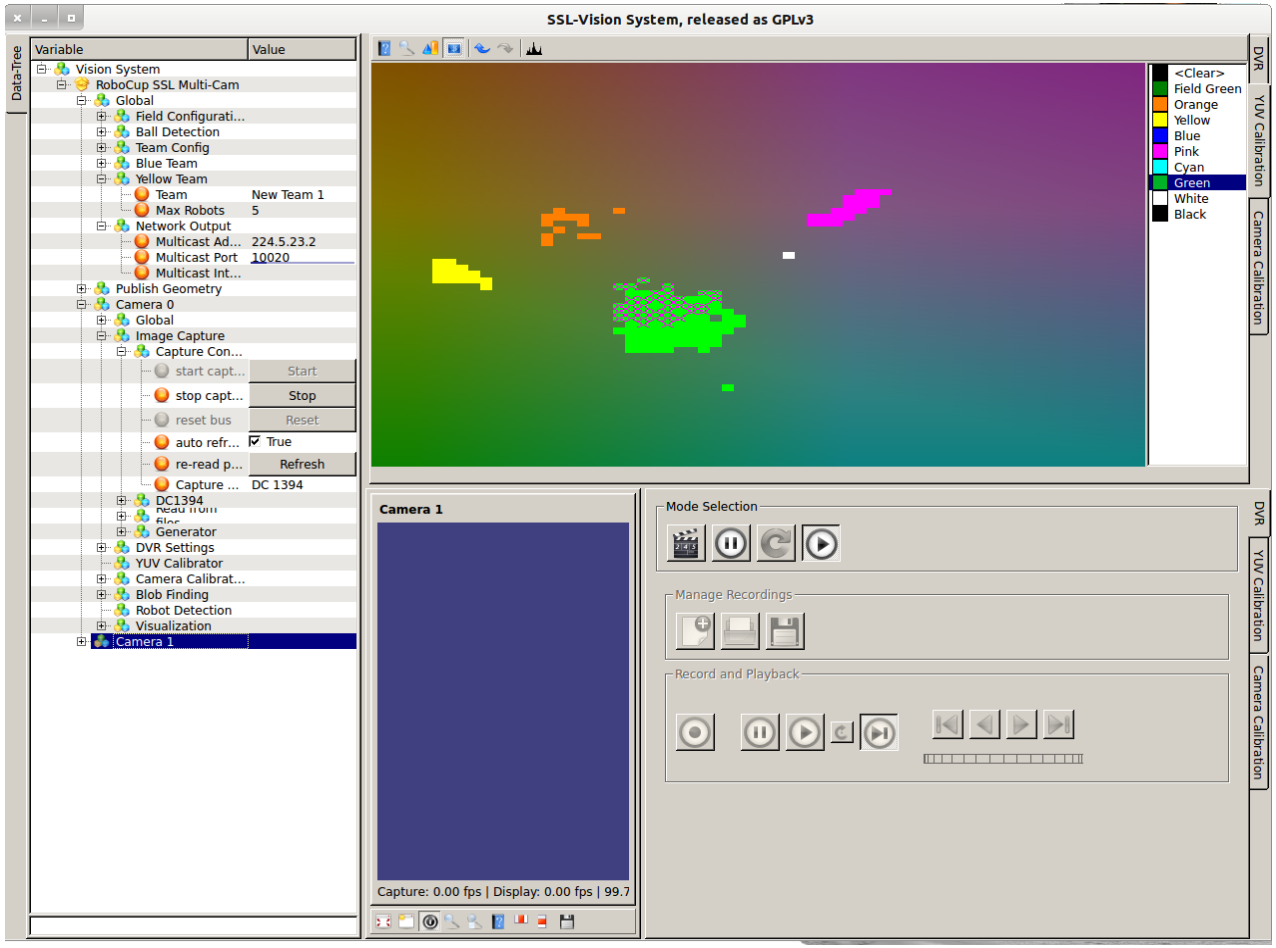


Ilustración 77: interfaz del sistema de visión

Asimismo, la siguiente ilustración capta la cámara que, a través del SSL-Vision [11], transforma en posiciones asociadas a la situación de los robots y la pelota:

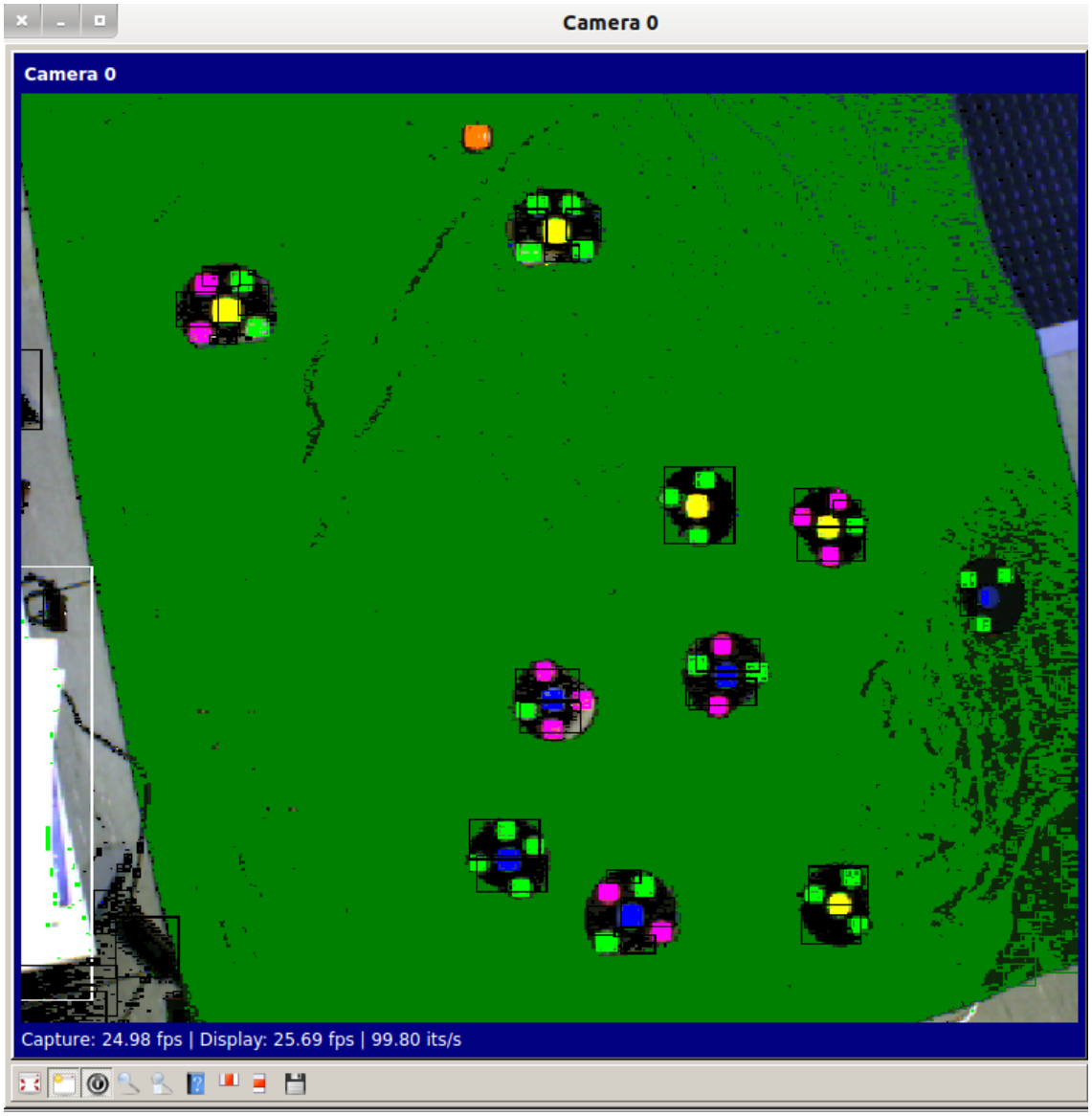


Ilustración 78: recepción cámara para SSL-Vision [11]

Que, al traducirlo, queda representado en el esquema del campo como:

Ilustración 79: representación del campo a través de la recepción de cámara

EXPERIMENTOS

4. Experimentos

En este capítulo se detallarán los experimentos que se han realizado. Debido a que este PFC se puede ver en dos bloques diferenciados, por un lado, los experimentos en el simulador y, por otro, los experimentos en los robots físicos, este punto se va a dividir de esa manera. Así, se diferenciarán los experimentos que se han producido en el simulador y los experimentos propios de los robots físicos. Tras ello, se realizará una comparativa entre las dos experimentaciones, resaltando qué diferencias se encuentran en implementar lo descrito con un simulador y con unos robots físicos.

A lo largo de este capítulo se comentarán experimentos que se han realizado y se hará referencia a unos vídeos grabado sobre estos, pudiendo consultarse en la página para compartir vídeos: Youtube [37].

Hay que recordar que la experimentación con los robots físicos forma parte de una ampliación de este PFC, por lo que es menos elaborada y desarrollada que la simulación con grSim [7] [8], siendo el objetivo principal de este trabajo la implementación de una arquitectura de control 3T [36] en la RoboCup [17] SSL [9].

4.1. Experimentos en robots simulados

Este apartado va a mostrar los experimentos que se han desarrollado en el simulador, siempre orientado a probar las habilidades elaboradas, así como el pleno funcionamiento de la transmisión de datos y las comunicaciones realizadas.

Dentro de las pruebas de los movimientos de los robots, todas ellas se han hecho con robots de ambos equipos, tanto el equipo amarillo como el equipo azul.

4.1.1. Pruebas cliente de recepción Java

Estas pruebas están relacionadas con el cliente para el simulador grSim [7] [8] de recepción de datos de este en el lenguaje de programación Java [18].

- **Nombre prueba:** prueba cliente de recepción Java [18].

- **Objetivo:** asegurar que las comunicaciones entre la arquitectura y el cliente grSim [7] [8] funcionan, recibiendo y procesando los datos que este envía.

- **Explicación de la prueba:** el cliente de recepción Java [18] es una muestra que ejemplifica la comunicación para, posteriormente, incrustarlo en la arquitectura de la SSL [9], por lo que es necesario asegurar el pleno funcionamiento antes de establecer habilidades y elevar la complejidad.

- **Ejecución:** para realizar esta prueba se ha creado un pequeño cliente en Java [18] en el que se crea un socket *multicast* para recibir los paquetes de información. Una vez recibido el buffer con la información, se decodifica utilizando Google Protobuf [16] y se imprime para comprobar que los datos se han recibido correctamente.

- **Resultados:** los datos se envían por parte del simulador [7] [8] correctamente y el cliente los procesa.

  El resultado queda como:

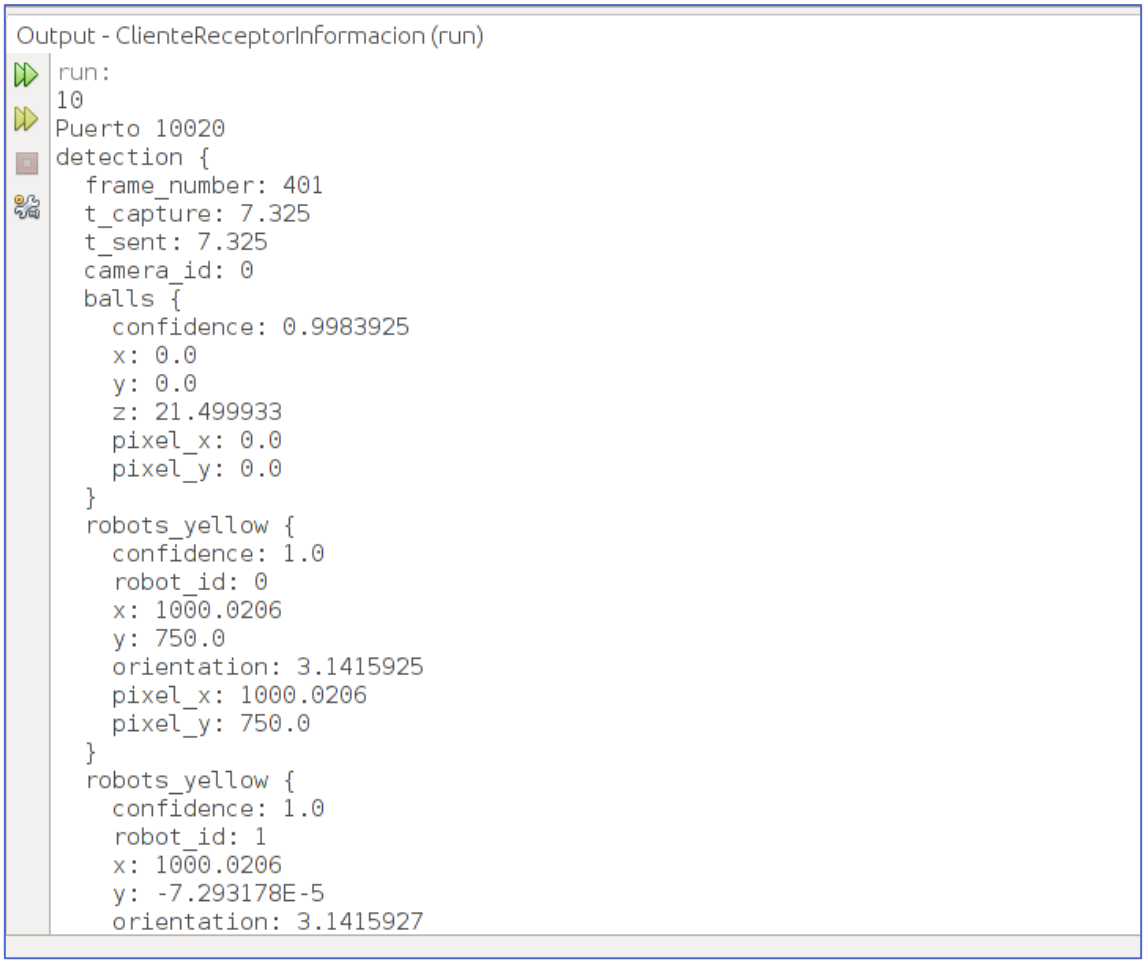


Ilustración 80: resultado prueba cliente de recepción

- **Problemas surgidos:** problemas con el tamaño del paquete, ya que no existe ningún tipo de documentación sobre este. Además, ciertos problemas relacionados con la codificación que realiza Google Protobuf [16] ya que, de nuevo, carece de mucha documentación.

- **Conclusiones:** prueba útil para demostrar que el simulador [7] [8] envía la información y que el cliente en Java [18] es capaz de recibirlo, procesarlo e imprimirlo. Esto es necesario para poder realizar cualquier prueba relacionada con las habilidades.

4.1.2. Pruebas cliente de envío Java

Estas pruebas están relacionadas con el cliente para el simulador grSim [7] [8] de envío de datos de este en el lenguaje de programación Java [18].

- **Nombre de prueba:** envío básico de información a un robot.

- **Objetivo:** asegurar que las comunicaciones entre la arquitectura y el cliente grSim [7] [8] funcionan, enviando los datos a este y realizando actividades simples moviendo los robots simulados.

- **Explicación de la prueba:** el cliente de envío Java [18] es una muestra que ejemplifica la comunicación para, posteriormente, incrustarlo en la arquitectura de la SSL [9], por lo que es necesario asegurar el pleno funcionamiento antes de establecer habilidades y elevar la complejidad.

- **Ejecución:** se ha creado un pequeño cliente en Java [18] el cual prueba que el simulador grSim [7] [8] recibe e interpreta correctamente los comandos que se le envían. En este cliente se abre un socket de datagramas que se conecta con el simulador grSim [7] [8] y se envía un paquete codificado con Google Protobuf [16], en este paquete se mandan los datos indicados en "3.2.1. Funcionamiento".

  La interfaz completada con la prueba queda como:

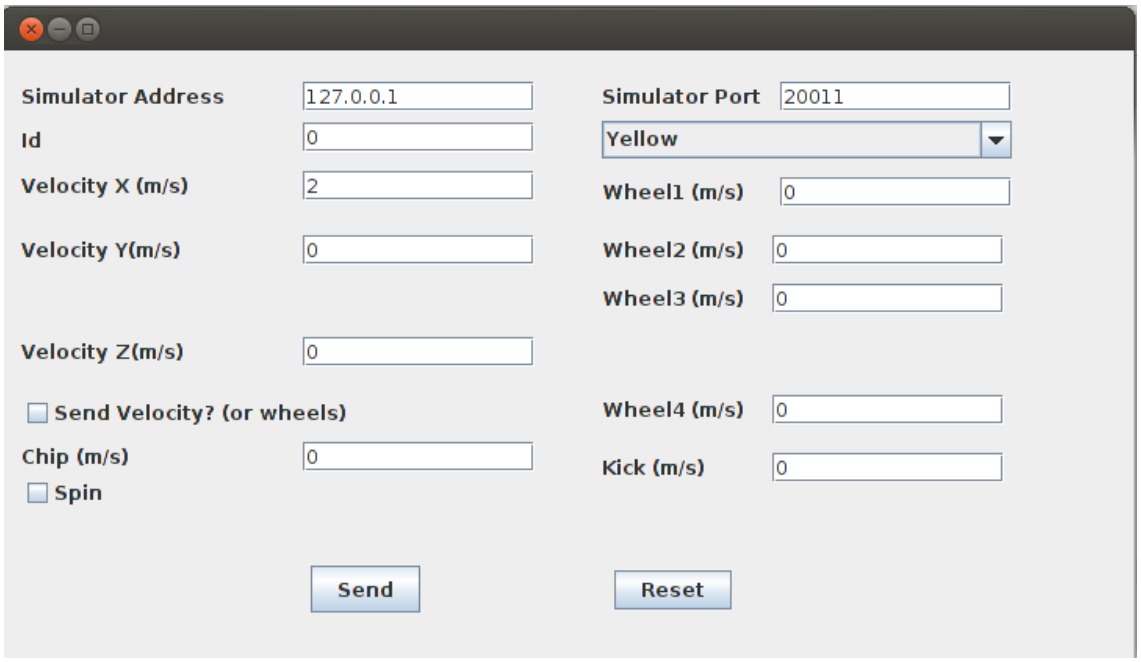


Ilustración 81: prueba cliente de envío con interfaz rellenada

Como puede comprobarse, para que el robot efectúe un movimiento recto y sencillo es necesario completar la dirección IP del simulador [7] [8], el puerto del mismo y la velocidad, en este caso en el eje de las X. Esta prueba, en concreto, y otra más similar, se pueden ver en dos vídeos elaborados [40] [41].

- **Resultados:** el robot se mueve con las instrucciones que se le dan, que es un movimiento sencillo y básico.

- **Problemas surgidos:** encontrar el paquete correcto que enviar para que el simulador [7] [8] lo interpretara correctamente. Además, averiguar el socket al que se tenía que mandar la información.

- **Conclusiones:** el cliente de envío ofrece una interfaz y el pleno funcionamiento esperado, por lo que es útil para hacer pruebas básicas con el simulador grSim [7] [8] en el lenguaje de programación Java [18].

4.1.3. Pruebas habilidad giro óptimo

Estas pruebas están relacionadas con la habilidad de giro óptimo en los robots. Esta habilidad es común para todos los robots y compatible para muchas de las habilidades que hacen uso de esta.

La primera prueba realizada es:

- **Nombre de la prueba:** habilidad del giro óptimo según las agujas del reloj.
- **Objetivo:** demostrar el funcionamiento de la habilidad del giro óptimo con la dirección de las agujas del reloj.
- **Explicación de la prueba:** esta habilidad versa sobre los giros que produce el robot cuando se mueve. Es necesario que, para que los movimientos sean eficientes, los giros se hagan hacia el lado óptimo, llegando al objetivo por el trayecto más corto. Esta prueba demuestra el funcionamiento del giro óptimo en la dirección de las agujas del reloj.
- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], que calcula el giro adecuado para que sea óptimo. En este caso, comprueba la posición de la pelota y la orientación del robot protagonista y gira hacia el lado derecho.
- **Resultados:** gira para la dirección que debe de forma satisfactoria. El resultado se puede comprobar en el vídeo aportado. [42]
- **Problemas surgidos:** problemas relacionados con los signos y los cuadrantes, sobre todo con la distribución y nomenclatura del campo, que a priori era desconocida porque el simulador grSim [7] [8] carecía de documentación, debido a que es un proyecto joven.
- **Conclusiones:** se comprueba que el giro es óptimo hacia el lado de las agujas del reloj.

La segunda prueba realizada es:

- **Nombre de la prueba:** habilidad del giro óptimo en contra de las agujas del reloj.

- **Objetivo:** demostrar el funcionamiento de la habilidad del giro óptimo con la dirección en contra de las agujas del reloj.

- **Explicación de la prueba:** esta habilidad versa sobre los giros que produce el robot cuando se mueve. Es necesario que, para que los movimientos sean eficientes, los giros se hagan hacia el lado óptimo, llegando al objetivo por el trayecto más corto. Esta prueba demuestra el funcionamiento del giro óptimo en la dirección contraria a las agujas del reloj.

- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], que calcula el giro adecuado para que sea óptimo. En este caso, comprueba la posición de la pelota y la orientación del robot protagonista y gira hacia el lado izquierdo.

- **Resultados:** gira para la dirección que debe de forma satisfactoria. El resultado se puede comprobar en el vídeo aportado. [43]

- **Problemas surgidos:** problemas relacionados con los signos y los cuadrantes, sobre todo con la distribución y nomenclatura del campo, que a priori era desconocida porque el simulador grSim [7] [8] carecía de documentación, debido a que es un proyecto joven.

- **Conclusiones:** se comprueba que el giro es óptimo hacia el lado contrario a las agujas del reloj.

4.1.4. Prueba habilidad buscar portería

Estas pruebas están relacionadas con la habilidad de los robots de buscar portería..

- **Nombre de la prueba:** habilidad buscar portería en el robot portero.
- **Objetivo:** demostrar el funcionamiento de la habilidad de la búsqueda de la portería por parte del robot portero.
- **Explicación de la prueba:** esta habilidad se basa en que el robot portero inicialmente está situado en el centro del campo. Por lo tanto, el robot debe ser capaz de buscar la portería para poder defenderla satisfactoriamente.
- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], que calcula la posición de la portería, según sea la de un lateral u otro. Una vez conocida la posición de esta, se gira sobre sí mismo y se dirige hacia ella, cuando llega a las coordenadas indicadas, se gira de nuevo sobre sí mismo para mirar al terreno de juego.
- **Resultados:** colocación en la portería propia de forma adecuada, mirando hacia la dirección correcta. El resultado se puede comprobar en el vídeo aportado, que combina el buscar la portería con el defender portería, habilidad no contemplada en este PFC. [44]
- **Problemas surgidos:** problemas al estructurar el conocimiento previo de la portería a la que tenía que ir el robot.
- **Conclusiones:** se comprueba que el robot está dotado del movimiento suficiente para poder llegar a la portería y situarse con actitud defensiva en el lugar que le corresponde.

4.1.5. Pruebas habilidad buscar pelota

Estas pruebas están relacionadas con la habilidad de los robots de buscar pelota.

- **Nombre de la prueba:** habilidad buscar pelota en robot no portero.
- **Objetivo:** demostrar el funcionamiento de la habilidad de la búsqueda de la pelota por parte de cualquier robot del equipo que no sea el portero.
- **Explicación de la prueba:** esta habilidad se basa en que el robot sea capaz de, a través del sistema de visión, recibir dónde está situada la pelota y vaya hacia ella, realizando un movimiento directo, óptimo y fluido.
- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], que recibe el paquete con información del terreno de juego, incluida la posición de la pelota. Con ella, se calcula la dirección que debe tomar el robot el línea recta para llegar a la posición de la pelota, este gira sobre sí mismo con el giro más óptimo para colocarse en esta dirección y, finalmente, va en línea recta hacia la pelota, logrando llegar a su objetivo.
- **Resultados:** el robot se encuentra lejos de la pelota y logra buscarla y llegar a donde la pelota está situada. El resultado se puede comprobar en los vídeos aportados, que incluye la combinación de dos habilidades, la de buscar pelota y la de conducirla. [45] [46] [47]
- **Problemas surgidos:** problemas con el giro óptimo, además de las coordenadas negativas y positivas del campo sin ningún tipo de documentación, problemas para calcular la dirección que tiene que tomar el robot y problemas con el redondeo de decimales, ya que si se redondeaba mucho, el robot se pasaba y no obtenía bien la dirección y si se ajustaba en exceso, no llegaba a encontrar la dirección tras las comprobaciones pertinentes.
- **Conclusiones:** habilidad funcionando correctamente, lo que permite que el robot esté dotado de la inteligencia suficiente para encontrar la pelota en el

campo e ir hacia ella. Es una habilidad muy útil, debido a que le permite al robot buscar la pelota y tener posesión de ella.

4.1.6. Pruebas habilidad conducir pelota

Estas pruebas están relacionadas con la habilidad de los robots de conducir pelota.

- **Nombre de la prueba:** habilidad conducir pelota en robot no portero.
- **Objetivo:** demostrar el funcionamiento de la habilidad de la conducción de la pelota por parte de cualquier robot del equipo que no sea el portero.
- **Explicación de la prueba:** esta habilidad es necesaria para que el robot, una vez que esté en posesión de la pelota, la conduzca, para poder estar lo suficientemente cerca y orientado para poder pasar a la habilidad de disparar y marcar un gol.
- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], el robot es consciente de que tiene la posesión de la pelota y conduce a través del terreno de juego, en dirección hacia la portería opuesta.
- **Resultados:** el robot se encuentra en posesión dela pelota y logra mantener esta posesión y trasladarse. El resultado se puede comprobar en los vídeos aportados, que incluye la combinación de dos habilidades, la de buscar pelota y la de conducirla. [45] [46] [47]
- **Problemas surgidos:** problemas para que el robot fuera consciente de que tenía la posesión de la pelota.
- **Conclusiones:** habilidad necesaria para que el robot sea capaz de conducir la pelota a través del terreno de juego sin perder la posesión.

4.1.7. Pruebas habilidad disparar

Estas pruebas están relacionadas con la habilidad de los robots de disparar, para poder lograr meter gol.

- **Nombre de la prueba:** habilidad disparar a portería en robot no portero con lanzamiento elevado.

- **Objetivo:** demostrar el funcionamiento de la habilidad de disparar a portería por parte de cualquier robot del equipo que no sea el portero y que tenga posesión de la pelota, elevando la pelota.

- **Explicación de la prueba:** esta habilidad se basa en el lanzamiento a portería cuando el robot está en posesión de la pelota, con el objetivo de lograr meter un gol al equipo contrario.

- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], el robot es consciente de que tiene posesión de la pelota y que está orientado hacia la pelota. Se acerca con la pelota hacia la portería y, cuando está lo suficientemente cerca, lanza con elevación con el objetivo de marcar un gol al otro equipo.

- **Resultados:** el robot lanza hacia la portería, intentando meter un gol y ganar el partido. El resultado se puede comprobar en el vídeo aportado. [48]

- **Problemas surgidos:** problemas para que el robot fuera consciente de que tenía la posesión de la pelota.

- **Conclusiones:** habilidad necesaria para que el robot, cuando esté en posesión de la pelota, sea capaz de meter un gol, tirando de forma elevada, pudiendo despistar a otros jugadores del equipo contrario.

La segunda prueba realizada es:

- **Nombre de la prueba:** habilidad disparar a portería en robot no portero con lanzamiento al ras del suelo.

- **Objetivo:** demostrar el funcionamiento de la habilidad de disparar a portería por parte de cualquier robot del equipo que no sea el portero y que tenga posesión de la pelota, lanzando al ras del suelo.

- **Explicación de la prueba:** esta habilidad se basa en el lanzamiento a portería cuando el robot está en posesión de la pelota, con el objetivo de lograr meter un gol al equipo contrario.

- **Ejecución:** se ejecuta la habilidad a través de la arquitectura del SSL [9], el robot es consciente de que tiene posesión de la pelota y que está orientado hacia la pelota. Se acerca con la pelota hacia la portería y, cuando está lo suficientemente cerca, lanza empujando la pelota y sin elevación, con el objetivo de marcar un gol al otro equipo.

- **Resultados:** el robot lanza hacia la portería, intentando meter un gol y ganar el partido. El resultado se puede comprobar en el vídeo aportado. [49] [50] [51]

- **Problemas surgidos:** problemas para que el robot fuera consciente de que tenía la posesión de la pelota.

- **Conclusiones:** habilidad necesaria para que el robot, cuando esté en posesión de la pelota, sea capaz de meter un gol, tirando sin elevar la pelota, pudiendo despistar a otros jugadores del equipo contrario.

4.2. Experimentos en robots físicos

Además de las pruebas añadidas relacionadas con los robots simulados, se han realizado algunas pruebas en los robots físicos. Estas son más limitadas, debido a que solo se dispone un robot para a SSL [9], perteneciente a la Universidad Carlos III de Madrid, en el Área de Ciencia de Computación e Inteligencia Artificial. Las pruebas han sido realizadas en el laboratorio 2.2.B06, con el siguiente entorno:

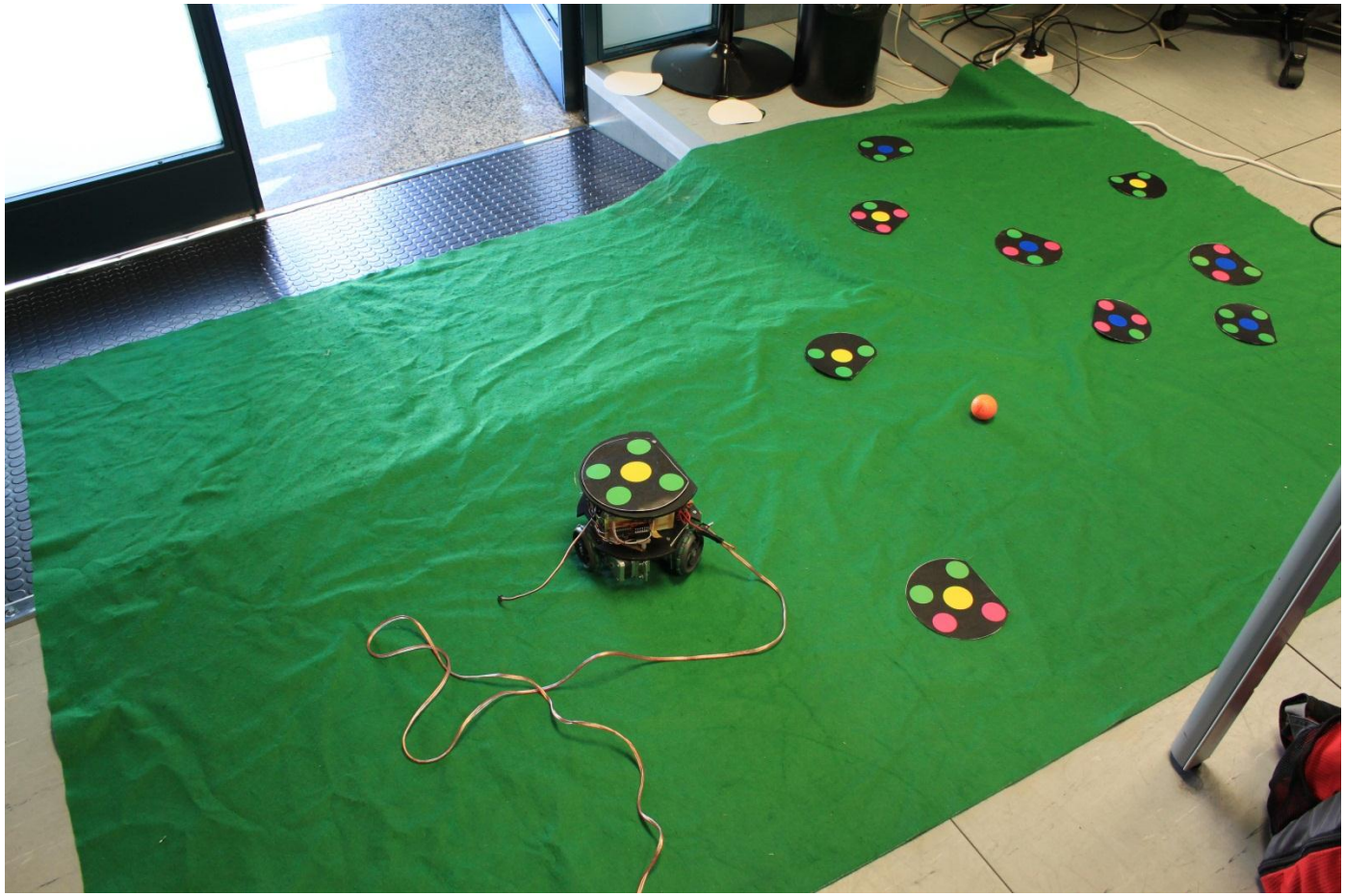


Ilustración 82: entorno para pruebas con robots físicos

Las pruebas creadas se han basado en demostrar el pleno funcionamiento del robot físico con el cliente Java [18] de envío de datos, tratándose las pruebas de las habilidades en el proyecto de Alejandro Caparrós Hernando [32].

El robot que se ha utilizado tiene que estar enchufado con cable para recibir suministro de batería, pero toda la información la recibe a través de UDP, por Wi-Fi. El robot usado se muestra a continuación:

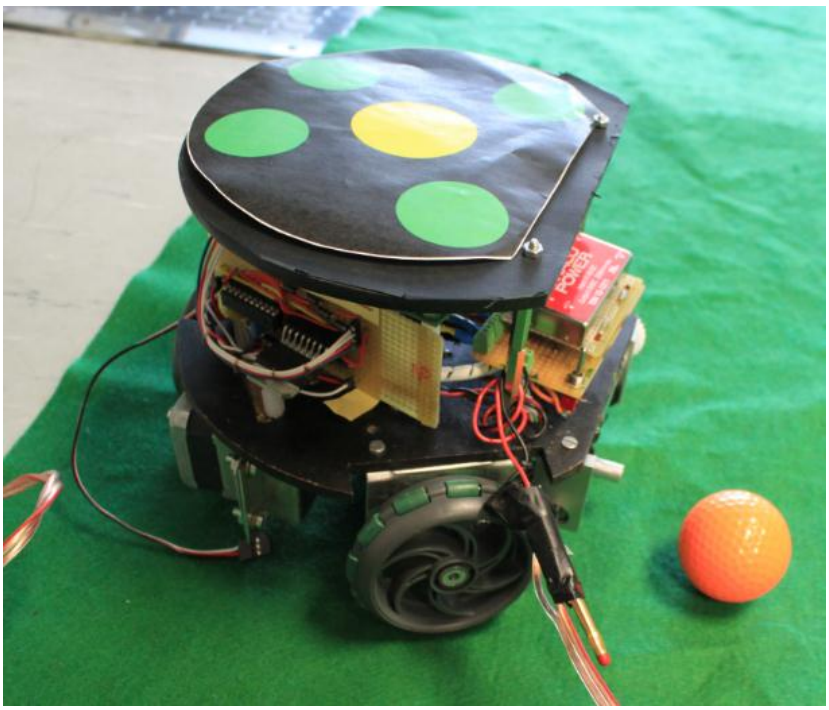


Ilustración 83: robot físico para las pruebas

Además, se detallan las pruebas que se han realizado:

- **Nombre de la prueba:** trayectoria recta hacia la pelota uno.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** esta prueba es la más sencilla y se basa en comprobar las capacidades del robot, si tiene inercia, si gira recto de forma exacta, etc. Además, se quiere verificar que la información a través del cliente Java [18] llega al robot físico correctamente.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se selecciona la velocidad por ruedas, se marca la velocidad de la rueda dos y tres por igual, la segunda negativa y la tercera positiva, para que vaya en línea recta.

- **Resultados:** el robot avanza, pero se tuerce ligeramente hacia la derecha, debido a que las dos ruedas no funcionan igual. El resultado se puede comprobar en el vídeo aportado. [52]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad.

Se cita la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta hacia la pelota dos.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** tras conocer los resultados de la anterior prueba, se opta por modificar ligeramente la velocidad de las ruedas, no poniéndolas exactamente igual, sino algo menos a la rueda más rápida, para ajustarlo de manera manual.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se selecciona la velocidad por ruedas, se marca la velocidad de la rueda dos y tres muy similar, a diferencia de que la segunda se pone algo menos, ya que en la anterior prueba se apreció que iba más rápido, la segunda negativa y la tercera positiva, para que vaya en línea recta.

- **Resultados:** el robot avanza, pero se tuerce ligeramente hacia la izquierda, debido a que las dos ruedas no funcionan igual. El resultado se puede comprobar en el vídeo aportado. [53]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta hacia la pelota tres.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** esta prueba es la más sencilla y se basa en comprobar las capacidades del robot, si tiene inercia, si gira recto de forma exacta, etc. Por lo tanto, se ajustarán los parámetros anteriores.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se selecciona la velocidad por ruedas, se ajusta de forma más precisa para lograr que el robot avance en línea recta.

- **Resultados:** el robot avanza y se consigue que llegue a la pelota en línea recta. El resultado se puede comprobar en el vídeo aportado. [54] [55]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad. Se ha logrado que el robot vaya hacia la pelota, pero de forma manual y a través de ensayo y error.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta y giro a la izquierda.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** esta prueba consiste en que el robot avance en línea recta para, posteriormente, gire a la izquierda. Se demuestra así que el robot es capaz de girar a la izquierda de forma definida.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se seleccionan los parámetros correspondientes para que el robot haga el movimiento deseado.

- **Resultados:** el robot avanza y se girar a la izquierda como se deseaba. El resultado se puede comprobar en el vídeo aportado. [56]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas. Asimismo, existe inercia y los motores no van de forma simultánea.

- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad. Se ha logrado que el robot haga el movimiento buscado de forma concreta, pero es necesario mejora la precisión.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta y giro a la derecha.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** esta prueba consiste en que el robot avance en línea recta para, posteriormente, gire a la derecha. Se demuestra así que el robot es capaz de girar a la derecha de forma definida.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se seleccionan los parámetros correspondientes para que el robot haga el movimiento deseado.

- **Resultados:** el robot avanza y se consigue que llegue a la pelota en línea recta. El resultado se puede comprobar en el vídeo aportado. [57]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad. Se ha logrado que el robot haga el movimiento buscado de forma concreta, pero es necesario mejora la precisión.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta y giro ligero.
- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.
- **Explicación de la prueba:** en este caso se quiere que el robot vaya hacia la pelota, para ello debe avanzar en línea recta y, después, girar ligeramente.
- **Ejecución:** se ejecuta el cliente Java [18] de envío y se seleccionan los parámetros correspondientes para que el robot haga el movimiento deseado.
- **Resultados:** el robot avanza y gira ligeramente, llegando a la posición de la pelota. El resultado se puede comprobar en el vídeo aportado. [58]
- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas, además de los motores, que no van con la misma exactitud.
- **Conclusiones:** es necesario mejorar la calidad de las ruedas y de los motores para que funcionen de manera exacta y, si no es así, sean capaz de corregir, para quitar la aleatoriedad. Se ha logrado que el robot vaya hacia la pelota, pero de forma manual y a través de ensayo y error.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta y loop.
- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.
- **Explicación de la prueba:** en este caso se realiza una prueba más complicada, se quiere que el robot vaya en línea recta y, después, realice un loop llegar a la posición de la pelota.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se seleccionan los parámetros correspondientes para que el robot haga el movimiento deseado.

- **Resultados:** el robot realiza el movimiento deseado de forma fluida y rápida. El resultado se puede comprobar en el vídeo aportado. [59]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** se demuestra en esta prueba que el robot se mueve de forma fluida y que consigue movimientos rápidos y directos.

Se introduce la siguiente prueba:

- **Nombre de la prueba:** trayectoria recta, giro a la izquierda y a la derecha.

- **Objetivo:** demostrar el funcionamiento del robot y verificar que no tiene inercia. Se pretende conocer las deficiencias del robot para poder ajustarlas de manera precisa y aportar mejoras al robot físico.

- **Explicación de la prueba:** esta prueba pretende que el robot comience andando en línea recta para, posteriormente, realizar un aparcamiento, haciendo un movimiento a la derecha y a la izquierda.

- **Ejecución:** se ejecuta el cliente Java [18] de envío y se seleccionan los parámetros correspondientes para que el robot haga el movimiento deseado.

- **Resultados:** el robot avanza y se consigue realizar el movimiento de aparcamiento de forma directa. El resultado se puede comprobar en el vídeo aportado. [60]

- **Problemas surgidos:** problemas con la calidad de las ruedas, así como la precisión de estas.

- **Conclusiones:** de nuevo esta prueba demuestra el funcionamiento rápido y fluido del robot.

4.3. Resultados

A lo largo de este punto se van a detallar los resultados que han ofrecido todas las pruebas realizadas, tanto en el simulador como en los robots físicos, aportando una comparativa entre ellos.

Por un lado, tras realizar las pruebas de conexión se concluye que estas han sido satisfactorias, ya que se ha conseguido que las conexiones con el simulador y el sistema SSL-Vision [11] funcionen correctamente. Esto significa que la implementación de los clientes de envío y recepción se integra en la arquitectura con las clases "EnviarPaquete" para el envío de comandos y "RecepcionPaquete" para la recepción del paquete con información enviada por el simulador o por SSL-Vision [11]. Estas pruebas verifican el pleno funcionamiento de la conexión y transmisión de información, cerciorando que esta llega y se recibe correctamente.

Respecto a las habilidades probadas en el simulador, como se puede apreciar en los vídeos aportados, funcionan correctamente y logran una buena precisión en todos los movimientos.

Gracias a la habilidad del giro óptimo, se logra dotar a los robots de un movimiento más natural y fluido, ya que siempre giran por el camino más corto, logrando imitar de una forma fiable a las personas. Esta habilidad se combina con el resto para conseguir unos movimientos lógicos y continuados, aportando naturalidad al comportamiento.

En general todas las habilidades muestran precisión y concretan la actividad sencilla a la que están destinados. El objetivo de la sencillez de las habilidades es poder combinarlas para lograr movimientos más complejos, a través del planificador.

Las pruebas de los robots físicos han aportado más problemas y dificultades, debido a diferentes factores relacionados con el entorno. Uno de los problemas más resaltables es el sistema de visión, ya que es complicado captar los colores de los robots, esto es porque entra en juego características como la luz ambiente de la habitación, la recepción de la cámara, el ajuste de los colores, etc.

Otro problema es que, en el simulador, los robots mueven sus ruedas de forma homogénea y armónica, mientras que los robots físicos no tienen exactamente la misma precisión. Estos intentan mover de forma equitativa todas las ruedas, pero cada rueda tiene un motor y no es 100% preciso. Esto significa que si en el simulador se le marca al robot que vaya en línea recta, lo hace satisfactoriamente y con una precisión exacta, mientras que si al robot físico se le marca el mismo movimiento, lo más probable es que este se desvíe. Esto se produce, también, por el rozamiento, la inercia, los distintos motores, etc.

CONCLUSIONES Y LÍNEAS FUTURAS

5. Conclusiones y Futuras líneas de trabajo

A lo largo de este capítulo se detallarán las conclusiones arrojadas por este proyecto y, finalmente, las futuras líneas de trabajo, comentando los posibles trabajos futuros a modo de ampliación.

5.1. Conclusiones

En el primer capítulo del presente documento se citaban los objetivos de los que constaba el PFC desarrollado, es en este punto donde se realiza un análisis en perspectiva del trabajo realizado, describiendo lo que se ha logrado y las conclusiones que esto aporta.

El principal objetivo de este PFC era la creación de una arquitectura para la liga SSL [9] de la competición RoboCup [17], con ciertas habilidades básicas que plasmasen las bases para la elaboración de un equipo de esta liga. Este objetivo ha sido el pilar de todo este proyecto, obteniendo como resultado una arquitectura de control 3T [36] capaz de crear los cimientos para la creación futura de un equipo capaz de presentarse a la competición. Se ha cumplido el objetivo principal, ya que se ha definido una arquitectura capaz de integrarse con los robots físicos y el simulador grSim [7] [8] para el posterior control de estos.

Además, se han alcanzado los ojetivos específicos establecidos. El primero de ellos, la adaptación de la arquitectura 3T [36] elaborada en el proyecto de José Luis Díaz Cebrián [31], partiendo del trabajo realizado por este compañero se han realizado las adaptaciones pertinentes para lograr incrustar lo necesario para que la arquitectura fuera capaz de realizar las habilidades para los robots, tanto físicos como simulados.

Tras ello, se ha llegado a realizar la adaptación completa del simulador grSim [7] [8], pudiendo la arquitectura 3T [36] enviar y recibir información, siendo esta capaz de procesar, entender y manipular para poder enviar las órdenes a los robots simulados. Asimismo, se ha comprendido completamente el funcionamiento del simulador grSim [7] [8], creando documentación útil para este PFC y para el equipo Parsian Robotic [7],

con el cual se ha colaborado directamente y se ha cedido la información creada para mejorar la documentación del simulador [7] [8].

Se ha cumplido el objetivo de trabajar directamente con un equipo existente, real y en activo, a través de la colaboración con el simulador grSim creado por este equipo, Parsian Robotic. [7] [8]. Esto permite conocer de primera mano el trabajo real y práctico que tiene este PFC y, además, ayudar a este equipo a generar documentación inexistente hasta el momento sobre el simulador [7] [8]

Por otro lado, se consiguió modificar el simulador grSim [7] [8] para adaptar los robots simulados a los robots físicos de los que se constaba. Este trabajo se ha apoyado en dos PFCs [33] [34] que detallan el prototipo del robot físico creado en la Universidad Carlos III de Madrid por el Departamento de Ingeniería de Sistemas y Automática en el Laboratorio de Sistemas Inteligentes. Estos PFCs [33] [34] describieron y elaboraron un robot físico para la SSL [9] donde se decidió crear el robot con tres ruedas para abaratar los costes, respecto a cuatro, que suele ser lo más común. Por eso mismo, el simulador grSim [7] [8] solo simulaba robots de cuatro ruedas, eso llevó a tener que modificar la simulación, lográndolo satisfactoriamente.

Asimismo, se han elaborado las habilidades básicas propias de la SSL [9], orientadas al ataque, ya que existe otro PFC complementario de Alejandro Caparrós Hernando [32] que desarrolla las habilidades de la defensa en los robots. Esto cumple otro de los subjetivos definidos por este proyecto y que concreta las bases para la creación de un equipo completo para la participación de la competición de la RoboCup [17] en la liga SSL [9].

Además de esto, se han probado las habilidades tanto en el simulador grSim [7] [8] como en los robots físicos, comprobando su pleno funcionamiento y describiendo las diferencias entre las dos plataformas.

A modo de conclusión general, se han superado todos los hitos propuestos de este PFC y se ha logrado definir una arquitectura de control para la SSL [9].

5.2. Futuras líneas de trabajo

El presente PFC establece las bases para la creación de un equipo de fútbol para la liga SSL [9] de la competición RoboCup [17]. Sin embargo, este trabajo ha sido planteado desde el comienzo para que pueda ser fácilmente ampliado y poder obtener así un equipo que se pudiera presentar a esta competición.

Por este motivo, las habilidades aportadas pueden ser mejoradas, buscando una mayor precisión y unos movimientos más naturales y fluidos. Se podrían asimismo, desarrollar mejoras en las habilidades para lograr imitar de forma más precisa a un jugador de fútbol, pudiendo llegar a conseguir, en un futuro, un robot capaz de imitar a la perfección a un jugador humano.

Las habilidades creadas en las que se ha enfocado este proyecto están orientadas al ataque, complementándolas con el PFC de Alejandro Caparrós Hernando [32], que elabora las habilidades de defensa, pero el conjunto de estas habilidades se podrían aumentar. Se podría incluir más habilidades que aporten mayor conocimiento a los robots, por ejemplo, relacionadas con la formación, tirar un penalti, robar la pelota a otro robot, tiro libre, tiro lateral, comienzo de juego, etc.

En este proyecto, los robots solo pueden tener uno de los dos roles principales: portero y no portero. Esta restricción podría ser ampliada creando nuevos roles más específicos, por ejemplo, los robots defensas y los robots atacantes, definiendo diferentes comportamientos y actitudes, propios del rol que tienen. Esto es similar a lo que en este PFC se hace entre el portero y el resto de robots, ya que son distintos roles.

De forma inminente, el siguiente paso que se podría realizar sería, a partir de las habilidades básicas desarrolladas, utilizar el planificador para poder hacer estrategias propias de un partido de fútbol. De esta forma, el planificador usaría las habilidades básicas para realizar estrategias, por ejemplo, tener una actitud defensiva o, por el contrario, optar por atacar.

Otra posible futura línea de trabajo sería la realización de planes complejos que permitan la ejecución de las habilidades en función de las condiciones del juego, el

resultado, el comportamiento de los otros robots, etc. Hilando con el establecimiento del planificador, la idea sería que los robots adquirieran uno u otro comportamiento, en base a las características del partido en tiempo real. Esto sería útil para poder definir técnicas, actitudes y comportamientos más elaborados.

Todo el trabajo desarrollado se puede ampliar a otras ligas, como en los robots medianos o los robots humanoides. Esta es la línea de trabajo más compleja, porque cada liga tiene sus propias reglas y su propia plataforma. La liga de simulación sería la liga en la que mejor podría aplicarse la arquitectura creada en este PFC, ya que se ha utilizado un simulador y la adaptación sería más directa, aunque no por ello compleja.

Como parte de la arquitectura, hay algunas mejoras que se podrían añadir. La primera, la implementación completa del planificador, ya que por el momento solo se han realizado las habilidades básicas. Además, se podrían incluir no solo un planificador, sino distintos tipos en la capa deliberativa, para utilizar el que se necesite en cada momento. Asimismo, se podría realizar otro tipo de arquitectura, en este PFC se ha creado una arquitectura híbrida 3T [36], pero existen distintas arquitecturas, también útiles e interesantes para aplicar en la SSL [9].

Concretando las mejoras posibles en el robot físico utilizado, se podría mejorar aportando mayor precisión en las ruedas o diseñando unas ruedas omnidireccionales, evitando el movimiento lateral que produce al moverse.

Por parte del simulador usado (Parsian Robotic) [7] [8], se podrían mejora los gráficos o que fuera más sencillo el cambio de tres a cuatro ruedas en la propia interfaz, con un botón o una opción en el menú.

En general, es un tema que está en pleno auge y que se puede mejorar en muchos aspectos en distintos líneas de trabajo. A través de este PFC se asientan las bases y el marco de trabajo de forma sencilla para que, futuros estudiantes, investigadores o personas interesadas puedan elaborar distintos estudios, investigaciones y proyectos en este ámbito.

REFERENCIAS

Referencias

Las referencias consultadas son las siguientes:

1) RAE:
www.rae.es, consultado el 14/11/2011.

2) Karel C. R.U.R.:
http://us.penguingroup.com/nf/Book/BookDisplay/0,,0_9780141182087,00.htm, consultado el 11/11/2011.

3) RoboCup Wikipedia:
http://es.wikipedia.org/wiki/RoboCup, consultado el 14/11/2011.

4) Twenta:
http://code.google.com/p/tewnta/, consultado el 14/11/2011.

5) Robosoccer:
http://www.cs.cmu.edu/~robosoccer/small/, consultado el 14/11/2011.

6) RoboCup-Simulator:
http://code.google.com/p/robocup-simulator/feeds, consultado el 14/11/2011.

7) Parsian Robotic:
http://www.parsianrobotic.ir/grsim/, consultado el 14/11/2011.

8) VALIALLAH, M., KOOCHAKZADEH, A., SHIRY GHIDARY, S., *RoboCup 2011: Robot Soccer World Cup XV*, 2011.

9) Small Size Informatik:
http://small-size.informatik.uni-bremen.de/, consultado el 14/11/2011.

10) Robots y Aprendizaje:
http://www.fleifel.net/ia/robotsyaprendizaje.php, consultado el 19/11/2011.

11) SSL-Vision:
http://code.google.com/p/ssl-vision/, consultado el 21/11/2011.

12) Aibo Sony:
http://support.sony-europe.com/aibo/, consultado el 15/12/2011.

13) Nao Aldebaran:
http://www.aldebaran-robotics.com/, consultado el 15/12/2011.

14) Sony:
http://www.sony.es/, consultado el 15/12/2011.

15) Wikipedia:
http://es.wikipedia.org/ , consultado el 15/12/2011.

16) Cámara Stingray, sistema de visión:
http://www.1stvision.com/cameras/AVT/Stingray-F-046-B-C.html, consultado el 20/1/2012.

17) RoboCup:
http://www.robocup.org/, consultado el día 4/4/2012.

18) Java:
http://www.java.com, consultado el día 9/4/2012.

19) C++:
http://www.cplusplus.com/, consultado el día 9/4/2012.

20) Oracle Corporation:

http://www.oracle.com, consultado el día 9/4/2012.

21) Google Protobuf:
http://code.google.com/p/protobuf/, consultado el día 29/5/2012.

22) Wordreference:
http://www.wordreference.com, consultado el día 9/4/2012.

23) Gimp:
http://www.gimp.org/, consultado el día 17/4/2012.

24) Microsoft Office 2010:
http://office.microsoft.com/es-es/, consultado el día 17/4/2012.

25) Netbeans:
http://netbeans.org/, consultado el día 17/4/2012.

26) Ubuntu:
http://www.ubuntu.com/, consultado el día 17/4/2012.

27) Windows 7:
http://windows.microsoft.com/es-ES/windows7/products/home, consultado el día 17/4/2012.

28) MACKWORTH, A. K., *On Seeing Robots*, University of British Columbia, Vancouver, B.C. (Canadá).
Se puede consultar en:
http://www.cs.ubc.ca/~mack/Publications/CVSTA93.pdf, consultado el 31/5/2012.

29) BASU, A., Computer Vision: Systems, Theory and Applications, World Scientific Series in Computer Science, Hardcover.

30) Noticia ABC sobre Deep Blue: http://www.abc.es/20100211/historia-/deep-blue-201002111420.html, consultado el 31/5/2012.

31) DÍAZ CEBRIÁN, José Luis, *Implementación de una Arquitectura de Control Híbrida para Robots Autónomos*, Universidad Carlos III de Madrid, Leganés (Madrid), 2011.

32) CAPARRÓS HERNANDO, Alejandro, *Implementación de Arquitectura de Control 3T en la Robocup Small Size League: Formación de defensa*, Universidad Carlos III de Madrid, Leganés (Madrid), 2012.

33) GARCÍA LÓPEZ, Álvaro, *Desarrollo del Sistema de Locomoción de una Plataforma Hardware para RoboCup Small Soccer League (SSL)*, Universidad Carlos III de Madrid, Leganés (Madrid), 2010.

34) ESCUDERO JIMÉNEZ, Lidia, *Sistema de Procesamiento, Alimentación y Estructura de un Microrobot*, Universidad Carlos III de Madrid, Leganés (Madrid), 2012.

35) MARTÍN GÓMEZ, José Luis, *Diseño del Sistema de Lomoción y Dribbler de un Microrobot*, Universidad Carlos III de Madrid, Leganés (Madrid), 2012.

36) BONASSO, R.P., FIRBY, J., GAT, E., KORTENKAMP, D., MILLER, D., SLACK, M., Experiences with an Architecture for Intelligent Reactive Agents. Journal of Experimental and Theoretical Artificial Intelligence, vol. 9, no. 2. 2007.

37) Youtube:
http://www.youtube.com, consultado el día 2/6/2012.

38) POZA LUJÁN, J. L., POSADAS YAGÜE, J. L., *Revisión de las Arquitecturas del Control Distribuido*, 2009.
Se puede consultar en:
http://riunet.upv.es/bitstream/handle/10251/6407/Arquitecturas%20de%20control%20distribuido.pdf, consultado el 2/6/2012.

39) FIRBY, R. J., *Adaptive Execution in Complex Dynamic Worlds*, Yale University, 1989.

40) Prueba cliente de envío Java 1:
http://www.youtube.com/watch?v=XofqGrTm52c&feature=youtu.be, consultado el día 26/6/2012.

41) Prueba cliente de envío Java 1:
http://www.youtube.com/watch?v=EsnGRW2pfJY&feature=youtu.be, consultado el día 26/6/2012.

42) Prueba giro óptimo a la izquierda:
http://www.youtube.com/watch?v=JfjsG5IHF8w, consultado el día 26/6/2012.

43) Prueba giro óptimo a la derecha:
http://www.youtube.com/watch?v=MyT1AU1ZFVk&feature=youtu.be, consultado el día 26/6/2012.

44) Prueba ir a portería en portero:
http://www.youtube.com/watch?v=j1GaUB84Ks8, consultado el día 26/6/2012.

45) Prueba buscar y conducir pelota 1:
http://www.youtube.com/watch?v=0KKtF3-GSpM, consultado el día 26/6/2012.

46) Prueba buscar y conducir pelota 2:
http://www.youtube.com/watch?v=K5BmNZTsn50, consultado el día 26/6/2012.

47) Prueba buscar y conducir pelota 3:
http://www.youtube.com/watch?v=JZ94ioFuVLE, consultado el día 26/6/2012.

48) Prueba disparar sin elevar pelota:
http://www.youtube.com/watch?v=zIAwFxOQWiE, consultado el día 26/6/2012.

49) Prueba disparar elevando pelota:
http://www.youtube.com/watch?v=QwDlcjPMi_U, consultado el día 26/6/2012.

50) Prueba disparar elevando pelota 2:
http://www.youtube.com/watch?v=-yda5xyjTyc&feature=youtu.be, consultado el día 26/6/2012.

51) Prueba disparar elevando pelota 2:
http://www.youtube.com/watch?v=Whw-YTeARMQ&feature=youtu.be, consultado el día 26/6/2012.

52) Prueba robot físico trayectoria recta uno:
http://www.youtube.com/watch?v=1k5Zbs2ubAI&feature=youtu.be, consultado el día 26/6/2012.

53) Prueba robot físico trayectoria recta dos:
http://www.youtube.com/watch?v=LRVrRNUABio&feature=youtu.be, consultado el día 26/6/2012.

54) Prueba robot físico trayectoria recta tres:
http://www.youtube.com/watch?v=LOkqdDciApM&feature=youtu.be, consultado el día 26/6/2012.

55) Prueba robot físico trayectoria recta desde frente:
http://www.youtube.com/watch?v=vopgQXR1Ob8&feature=youtu.be, consultado el día 26/6/2012.

56) Prueba robot físico trayectoria recta y giro a la izquierda:
http://www.youtube.com/watch?v=4jzcq16M4pM&feature=youtu.be, consultado el día 26/6/2012.

57) Prueba robot físico trayectoria recta y giro a la derecha:
http://www.youtube.com/watch?v=Hmmz6UHN9z4&feature=youtu.be, consultado el día 26/6/2012.

58) Prueba robot físico trayectoria recta y giro ligero:
http://www.youtube.com/watch?v=75lHIWssMSE&feature=youtu.be, consultado el día 26/6/2012.

59) Prueba robot físico trayectoria recta y loop:
http://www.youtube.com/watch?v=3A4TKwehqxU, consultado el día 26/6/2012.

60) Prueba robot físico trayectoria recta, giro a la izquierda y a la derecha:
http://www.youtube.com/watch?v=yLwHU06bHog&feature=youtu.be, consultado el día 26/6/2012.

61) Wireshark:
http://www.wireshark.org/, consultado el día 3/7/2012.

ANEXOS

ANEXO I: ACRÓNIMOS Y DEFINICIONES

ANEXO I: Acrónimos y Definiciones

En el presente apartado se van a mostrar los acrónimos y definiciones que se van a usar a lo largo de toda la memoria.

1. Acrónimos

En este punto se introducen los acrónimos utilizados:

Acrónimo	Significado
3T	Three-Tier
AMD	Advanced Micro Devices
CPU	Central Processing Unit
CSMA/CD	Acceso Múltiple por Detección de Portadora con Detección de Colisiones
GHz	Gigahercio
Hz	Hercio
F180	Small Size League
GB	GigaByte
GIMP	GNU Image Manipulation Program
GNU	GNU is Not Unix
IBM	International Business Machines
Id	Identificador
IP	Internet Protocol
ISM	Industrial, Scientific and Medical
IVA	Impuesto al valor agregado
J-League	Liga Profesional Japonesa de Fútbol
MB	MegaBytes
mm	Milímetros

Acrónimo	Significado
m/s	Metros/segundos
PFC	Proyecto de Fin de Carrera
PFCs	Proyectos de Fin de Carrera
RAE	Real Academia Española
RAM	Random-Access Memory
RAP	Reactive Action Package
RUR	Rossum's Universal Robots
SI	Sistema Internacional de Unidades
SSL	Small Size League
UBC	University of British Columbia
UDP	User Datagram Protocol
VAIO	Visual Audio Intelligence Organizer
WPAN	Redes Inalámbricas de Área Personal

Tabla 48: acrónimos

2. Definiciones

A continuación se introducen las definiciones utilizadas:

- **AIBO de Sony [14]:** robot mascota fabricado por Sony [14], tiene forma de perro y dispone de sensores que le evitan chocar contra objetos, además de una cola que funciona de antena y consta de sentido del tacto. [12]

- **Aldebaran Robotics:** empresa francesa dedicada al desarrollo y fabricación de robots. [13]

- **Apple Inc.:** empresa multinacional estadounidense que diseña y produce equipos electrónicos y de software. [15]

- **ATI Technologies Inc.:** fue una de las mayores empresas de hardware que diseñaba tarjetas gráficas, fue comprada por AMD en el año 2006, pero mantuvo su nombre para algunos productos. [15]

- **Bluetooth:** especificación industrial para WPAN que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz. [15]

- **C++:** lenguaje de programación orientado a objetos híbrido diseñado a mediados de los años 1980 por Bjarne Stroustrup. [15] [19]

- **Checkbox:** o casilla de verificación es un elemento de interacción de la interfaz gráfica de usuario. [15]

- **Driblar:** en algunos deportes, esquivar a un contrario al mismo tiempo que se avanza con el balón, también llamado regatear. [1] [21]

- **Dual Core:** doble núcleo.

- **Ethernet:** estándar de redes de área local para computadores con acceso al medio por contienda CSMA/CD. [15]

- **Frame:** es una imagen particular dentro de una sucesión de imágenes que componen una animación. La continua sucesión de estos fotogramas producen a la vista la sensación de movimiento, fenómeno dado por las pequeñas diferencias que hay entre cada uno de ellos. [15]

- **Gary Kaspárov:** ajedrecista profesional, escritor y activista político ruso. [15]

  GIMP: programa de edición de imágenes digitales en forma de mapa de bits, tanto dibujos como fotografías. [15]

- **GNU:** movimiento y comunidad de software y conocimiento libres. [15]

- **Hercio:** unidad de frecuencia del SI.

- **Intel Core i7:** es una familia de procesadores de cuatro núcleos de la arquitectura Intel x86-64. [15]
- **Intel Corporation:** es el mayor fabricante de circuitos integrados del mundo. [15]
- **Internet Protocol:** protocolo no orientado a conexión, usado tanto por el origen como por el destino para la comunicación de datos, a través de una red de paquetes conmutados no fiable y de mejor entrega posible sin garantías.
- **Java:** lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems en principios de los años 90. [15]
- **Linux:** núcleo libre del sistema operativo basado en Unix y de software libre. [15]
- **Loop:** bucle o giro.
- **Mac OS X:** sistema operativo desarrollado y comercializado por Apple Inc. que ha sido incluido en su gama de ordenadores Macintosh desde 2002. [15]
- **Macintosh:** ordenadores comercializados por la marca Apple Inc.
- **Máquina Virtual:** software que emula a una computadora y puede ejecutar programas como si fuera un ordenador real. [15]
- **Microsoft Office 2010:** suite de oficina que abarca e interrelaciona aplicaciones de escritorio, servidores y servicios para los sistemas operativos Microsoft Windows y Mac OS X. [24]
- **Microsoft Excel 2010:** programa para manejar hojas de cálculo perteneciente a la suite de Microsoft, Microsoft Office 2010. [24]
- **Microsoft PowerPoint 2010:** programa para procesamiento de textos perteneciente a la suite de Microsoft, Microsoft Office 2010. [24]

- **Microsoft Project 2010:** es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuestos y analizar cargas de trabajo.

- **Microsoft Word 2010:** programa para manejar hojas de cálculo perteneciente a la suite de Microsoft, Microsoft Office 2010. [24]

- **Multicast:** o en español multidifusión, es el envío de la información en la red a múltiples destinos de forma simultánea. [15].

- **Nao Aldebaran:** robot autónomo, programable y de mediana estatura, desarrollado por la empresa francesa Aldebaran Robotics. [13]

- **Netbeans:** entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. [15] [25]

- **NVidia Corporation**: es una empresa multinacional especializada en el desarrollo de unidades de procesamiento gráfico y tecnologías de circuitos integrados para estaciones de trabajo, ordenadores personales y dispositivos móviles. [15]

- **Oracle Corporation:** una de las mayores compañías de software del mundo. Sus productos van desde bases de datos hasta sistemas de gestión. [17]

- **Pioneer 3-DX:** robot móvil.

- **RAP:** concepto implementado por Firby en la arquitectura 3T [31] para poder asignar habilidades a las tareas. [39]

- **RoboCup** [17]**:** proyecto internacional para promover, a través de competencias integradas por robots autónomos, la investigación y educación sobre la inteligencia artificial. [3]

- **Round-Robin:** método para seleccionar todos los elementos en un grupo de manera equitativa y en un orden racional, normalmente comenzando por el primer elemento de la lista hasta llegar al último y empezando de nuevo desde el primer elemento. [15]

- **RCM4400-Rabbit:** controlador comercial de comunicación Wi-Fi. [34] [35]

- **Socket:** concepto abstracto por el cual dos programas (posiblemente situados en computadoras distintas) pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada. [15].

- **Sony:** empresa de origen japonés, siendo una de las mayores fabricantes de electrónica, audio, vídeo profesional, videojuegos y tecnologías de la información y comunicación. [14]

- **Spinner:** mecanismo para controlar la pelota que llevan los robots de la SSL, aceptado en la competición. [15]

- **Sun Microsystems:** empresa informática que se dedicaba a vender ordenadores, componentes informáticos, software y servicios informáticos. Fue adquirida en el año 2009 por Oracle Corporation. [15]

- **Tarjeta amarilla:** tarjeta que se utiliza en distintas disciplinas deportivas, concretamente en fútbol significa que el jugador es amonestado oficialmente, la acumulación de dos tarjetas en el mismo partido significa la tarjeta roja y expulsión. [15]

- **Tarjeta roja:** tarjeta que se utiliza en distintas disciplinas deportivas, concretamente en fútbol significa que el jugador es expulsado. [15]

- **Ubuntu:** sistema operativo que utiliza un núcleo Linux y su origen está basado en Debian. [15]

- **Unix:** sistema operativo portable, multitarea y multiusuario.

- **Vaio:** marca de ordenador personal fabricado por Sony. [14] [15]
- **Velcro:** marca registrada desde 1951 que equivale a una cinta con ganchos y otra con fibras enmarañadas en bucle que forman un sistema de apertura y cierre rápido. [15]
- **Windows 7:** versión más reciente de Microsoft Windows, línea de sistemas operativos producida por Microsoft Corporation.
- **Wi-Fi:** mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. [15]
- **Wireless:** también llamada comunicación inalámbrica, es aquella en la que los extremos de la comunicación no se encuentran unidos por un medio de propagación físico, sino que se utiliza la modulación de ondas electromagnéticas a través del espacio. [15]
- **Wireshark:** analizador de protocolos utilizado para realizar análisis y solucionar problemas en redes de comunicaciones, para desarrollo de software y protocolos, y como una herramienta didáctica para educación. [15] [61]
- **x86-64:** arquitectura basada en la extensión del conjunto de instrucciones x86 para manejar direcciones de 64 bits. [15]

ANEXO II: GESTIÓN DEL PROYECTO

ANEXO II: Gestión del Proyecto

Durante este anexo se describirá la gestión íntegra del proyecto, detallando en particular las fases que incluyen este trabajo, las subtareas en las que se ha dividido y, finalmente, una representación gráfica con el diagrama de Gantt.

El presente PFC ha sido elaborado de forma simultánea con otras actividades, asignaturas y trabajos, por lo que el tiempo dedicado a él no ha sido completo. Por eso mismo, la duración del trabajo ha sido prolongada si se basan en los créditos a los que corresponden.

1. Fases

En este punto se detallarán todas las fases correspondientes al proyecto en los siguientes puntos, así como las tareas dentro de cada una de ellas, que describen con más exactitud todo el trabajo:

- **Documentación:** fase referente a la búsqueda de información relacionada con este proyecto, así como la investigación de los distintos puntos tratados. se copila y clasifica la información para su fácil acceso.
    - **Investigación simuladores:** tarea orientada a la búsqueda de un simulador específico para la liga SSL [9], que cumpliera las necesidades pertinentes. En cierta medida complicado, porque suele ser algo privado de cada equipo participante y no suele ser público.
    - **Lectura PFCs:** lectura de los PFCs relacionados con este trabajo, así como otros de interés. El estudio de los PFCs se ha centrado en captar la esencia de la competición, así como conocer los trabajos que posteriormente se han utilizado o de los que ha partido este trabajo.

- **Análisis:** estudio de las actividades que se van a realizar, analizando la repercusión y el trabajo que van a conllevar. En esta parte se hace una estimación del tiempo de elaboración del PFC, así como la división de tareas.
    - **Estudio del simulador:** manejo y conocimientos del simulador grSim [7] [8], un simulador sin apenas documentación.
    - **Estudio de la arquitectura 3T [36]:** estudio de la arquitectura ya existente, para poder adaptarla a las exigencias de la liga SSL [9].
    - **Identificación de requisitos:** definición de los requisitos para plasmar mejor qué hace exactamente el trabajo creado. Estos se han dividido según los bloques del PFC.

- **Diseño:** fase correspondiente al diseño de la arquitectura creada, describiendo el diseño arquitectónico, así como el detallado.
    - **Diseño arquitectónico:** se realizará un diseño arquitectónico que incluirá diagramas de componentes, diagramas de clases y diagramas de interacción.
    - **Diseño detallado:** se creará una especificación y ampliación del diseño arquitectónico a través del diseño detallado.

- **Implementación:** parte práctica del análisis y diseño, donde realmente se programa y se ejecuta lo realizado en las anteriores fases. Por un lado se procederá a hacer la implementación de la arquitectura y, por otro, las modificaciones pertinentes en el simulador.
    - **Cliente Java [18] envío:** cliente para el fácil envío del paquete al simulador grSim [7] [8] o al sistema de visión.

- **Cliente Java[18] recepción:** cliente para la fácil recepción del paquete que envía el sistema de visión o el simulador grSim [7] [8].

- **Habilidades:** programación y desarrollo de las habilidades básicas para establecer las bases para formar un equipo para participar en la liga SSL [9] de la competición RoboCup [17].

- **Aplicación robots físicos:** comunicaciones y transferencia de acciones a los robots físicos.

- **Pruebas:** fase donde se demuestra que la implementación funciona y que el análisis y el diseño están correctamente realizados. Consiste en realizar pruebas en el simulador grSim [7] [8] y la arquitectura, así como en los robots físicos.

  - **Pruebas en simulador:** pruebas pertinentes realizadas en el simulador, desde probar los clientes a probar las habilidades y los movimientos de los robots.

  - **Pruebas en robots físicos:** pruebas realizadas en los robots físicos para ver su pleno funcionamiento.

- **Memoria:** elaboración de la presente memoria, que incluye toda información que se ha obtenido a lo largo de este trabajo.

  - **Recopilación documentación:** tras tener toda la documentación, se realiza una recopilación para seleccionar lo que se introduce en el documento.

  - **Realización memoria:** elaboración del documento que describe todo el trabajo elaborado a lo largo del PFC.

2. Ciclo de vida

Para la realización de este PFC se ha seguido un ciclo de vida en espiral, adecuado a este tipo de trabajo. El esquema que representa el ciclo de vida es el siguiente:

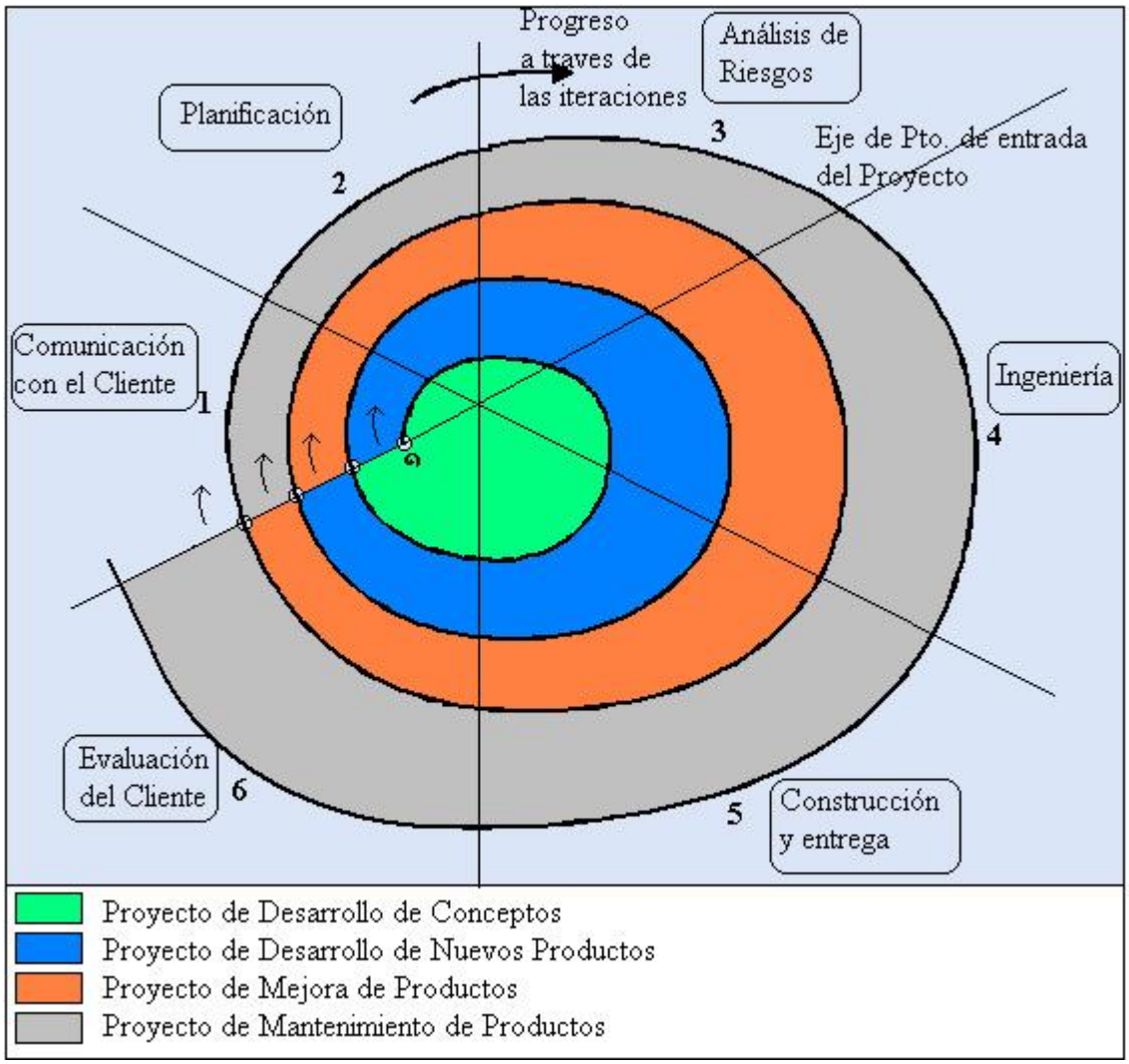


Ilustración 84: ciclo de vida de espiral [15]

Este modelo evolutivo consigue unificar las iteraciones del modelo de prototipos y el control y el trabajo sistemático característico del modelo en cascada. Este ciclo de

vida ha proporcionado el suficiente potencial para el desarrollo directo de versiones incrementales de forma sencilla, logrando en las últimas iteraciones versiones consistentes y elaboradas.

3. Diagrama de Gantt

En este punto se detallará el diagrama de Gantt basado en las fases que incluye este proyecto. No solo eso, se mostrarán cada una de las subtareas que incluyen las fases, describiéndolas en duración y en el periodo de tiempo elaboradas:

	Nombre de tarea	Duración	Comienzo	Fin
1	**Proyecto de Fin de Carrera**	**200 días**	**lun 12/09/11**	**vie 15/06/12**
2	**Documentación**	**26 días**	**lun 12/09/11**	**dom 16/10/11**
3	Investigación simuladores	16 días	lun 12/09/11	dom 02/10/11
4	Lectura PFCs	11 días	lun 03/10/11	dom 16/10/11
5	**Análisis**	**26 días**	**lun 17/10/11**	**lun 21/11/11**
6	Estudio del simulador	12 días	lun 17/10/11	mar 01/11/11
7	Estudio de la arquitectura 3T	11 días	lun 24/10/11	lun 07/11/11
8	Identificación de requisitos	10 días	mar 08/11/11	lun 21/11/11
9	**Diseño**	**86 días**	**mar 22/11/11**	**mar 10/01/12**
10	Diseño arquitectónico	22 días	lun 21/11/11	mar 20/12/11
11	Diseño detallado	14 días	mié 21/12/11	lun 09/01/12
12	**Implementación**	**71 días**	**lun 09/01/12**	**sáb 14/04/12**
13	Cliente Java envío	6 días	lun 09/01/12	lun 16/01/12
14	Cliente Java recepción	6 días	lun 16/01/12	lun 23/01/12
15	Habilidades	56 días	mar 24/01/12	mar 10/04/12
16	Aplicación robots físicos	22 días?	mar 24/01/12	mié 22/02/12
17	**Pruebas**	**86 días**	**lun 16/04/12**	**dom 08/06/12**
18	Pruebas en simulador	31 días	lun 16/04/12	dom 27/05/12
19	Pruebas en robots físicos	7 días	dom 27/05/12	dom 03/06/12
20	**Memoria**	**175 días**	**lun 17/10/11**	**vie 15/06/12**
21	Recopilación documentación	77 días	lun 17/10/11	mar 31/01/12
22	Realización memoria	98 días	mié 01/02/12	vie 15/06/12
23				

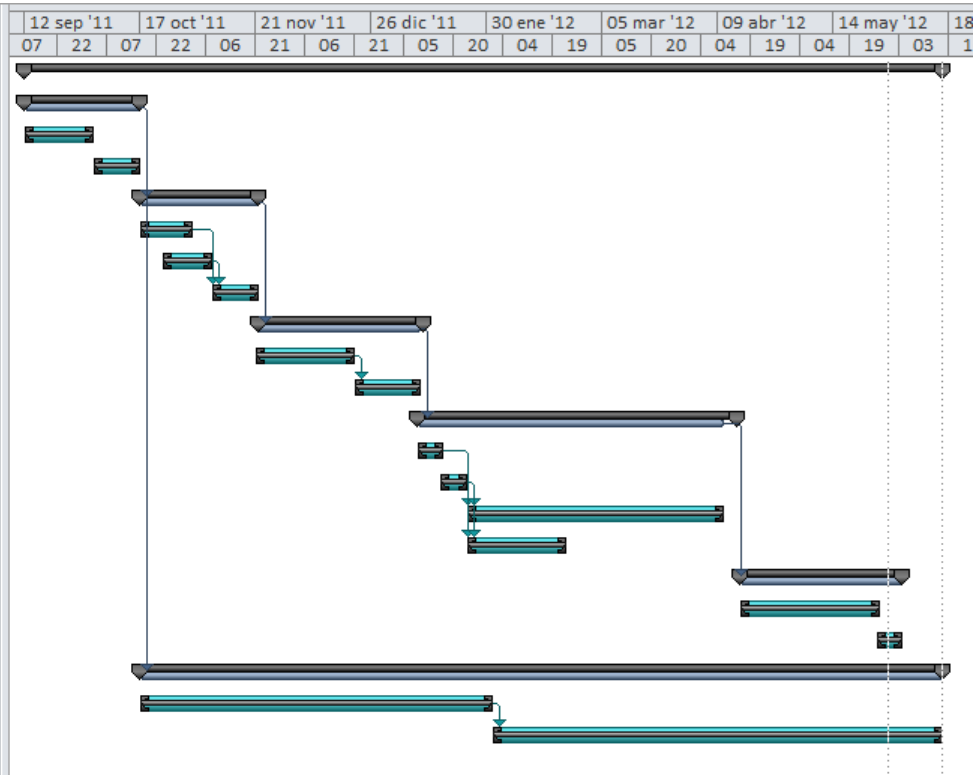


Ilustración 85: diagrama de Gantt

Como puede apreciarse, la duración del PFC es de 200 días, teniendo en cuenta que la elaboración de este trabajo ha sido combinada con otras actividades relacionadas con los estudios realizados. Esto significa que ha habido momentos a lo largo de los 200 días que la productividad ha sido muy baja.

4. Presupuesto

En el presente capítulo se detallará el presupuesto establecido de acuerdo al tiempo y los gastos que ha supuesto el presente PFC.

Por un lado, se incluirá la descripción de este trabajo y, por otro lado, el desglose del presupuesto de forma detallada.

4.1. Descripción del PFC

En este punto se incluye una descripción del trabajo realizado, añadiendo el nombre del PFC, la autora del mismo, la duración y la tasa de costes indirectos, se muestra en una tabla:

Nombre del PFC	**Implementación de Arquitectura de Control 3T en la Robocup Small Size League: Formación de Ataque**
Autora	Verónica Raspeño Gutiérrez
Duración	9 meses a tiempo parcial
Tasa de costes indirectos	20%

Tabla 49: descripción del PFC

4.2. Desglose del presupuesto

En este subapartado se desarrollará el desglose del presupuesto, incluyendo el personal, el material y, finalmente, el resumen de todos los gastos, con y sin IVA.

4.2.1. Personal

Se procede a introducir el gasto de personal que ha supuesto el presente trabajo:

Apellidos, nombre	N.I.F.	Categoría	Dedicación (horas/persona)	Coste (horas/persona)	Coste total (€)	Firma de conformidad
Raspeño Gutiérrez, Verónica	**49065492-Y**	**Analista**	**75**	**50**	**3.750,00**	
Raspeño Gutiérrez, Verónica	**49065492-Y**	**Programador**	**125**	**35**	**4.375,00**	

Tabla 50: coste de personal

4.2.2. Material

Se procede a introducir el gasto del material que ha supuesto el presente trabajo:

Producto	Coste (€)	Uso dedicado al PFC (%)	Dedicación (meses)	Periodo de depreciación (meses)	Tiempo desde la fecha de facturación (meses)	Coste imputable (€)
Sony Vaio VPCS13C5E [14]	1167'81	100%	9	60	12	233'56
Microsoft Windows 7	0'00	100%	9	60	24	0'00
Microsoft Office 2010	139'00	100%	9	60	24	55'6
Ubuntu 11	0'00	100%	9	60	10	0'00
grSim [7] [8]	0`00	100%	9	60	5	0'00
Gimp	0'00	100%	9	60	12	0'00
Cámara AVT Stingray [16]	858'00	100%	9	60	12	171'6
Robot Físico	1.143'18	100%	9	60	3	57'16
Total	**3.307'99**	-	-	-	-	**517'92**

Tabla 51: coste de material del PFC

4.2.3. Resumen del PFC

Se procede a introducir el resumen del presupuesto que ha supuesto el presente trabajo:

Concepto	Coste sin IVA (€)	IVA (€)	Coste con IVA (€)
Personal	8.125,00	1.462,50	9.587,50
Material	517,92	93,23	611,15
Coste indirecto	1.103,58	198,64	1302,22
Total	**9.746,50**	**1.745,37**	**11.500,87**

Ilustración 86: resumen del presupuesto del PFC

ANEXO III: MANUAL DE USUARIO

ANEXO III: Manual de Usuario

En este manual de usuario se detallarán todos los pasos para poder ejecutar el simulador grSim [7] [8], así como la arquitectura diseñada.

1. Manual del simulador grSim

El simulador es utilizado para realizar todas las pruebas de la arquitectura elaborada y de las habilidades programadas.

1.1. Requisitos mínimos del simulador grSim

Para poder instalar el simulador grSim [7] [8] es necesario cumplir unos requisitos técnicos mínimos. El simulador grSim [7] [8] funciona correctamente en un ordenador moderno Dual Core que utilice un sistema operativo Linux y una buena tarjeta gráfica.

Un ejemplo claro de una configuración típica podría ser:

- Dual Core CPU (2.0 Ghz+).
- 1 GB of RAM.
- 256MB nVidia or ATI graphics card.
- Ubuntu Linux 9.10+.

1.2. Instalación del simulador grSim

Para instalar el simulador en un ordenador deben seguirse los siguientes pasos que se muestran a continuación:

- Descargar e instalar los siguientes paquetes en el equipo:
  - libqt4-dev
  - libgl1-mesa-dev

  - libglu1-mesa-dev
  - VarTypes
  - ode
  - protobuf

- A continuación deben introducirse los siguientes comandos por consola:
  - cd proto/pb
  - protoc *.proto --cpp_out=../
  - qmake Simulator.pro
  - make

- El archivo binario se almacena en grSim/bin.

1.3. Uso del simulador grSim

Recibir información del simulador grSim [7] [8] es exactamente igual que recibirla del sistema de visión SSL-Vision [11] utilizando Google Protobuf [21].

Para enviar comandos al simulador se hace uso de Google Protobuf [21].

2. Manual de la arquitectura 3T

En este manual de usuario se detallará la ejecución de la arquitectura 3T [36] para robots de la Robocup [17], tanto para robots físicos reales como para el simulador grSim [7] [8]. Esta arquitectura contiene la implementación de todas las habilidades de la Robocup [17] detalladas anteriormente.

El objetivo de este manual es ayudar al usuario y guiarle en la ejecución de la arquitectura, así como facilitar la labor de inclusión de nuevas funcionalidades y

módulos. El manual detalla toda la información importante para el usuario, de la forma más precisa posible, a fin de facilitar el aprendizaje de uso del sistema y permitir una rápida toma de contacto y uso.

2.1. Requisitos mínimos de la arquitectura 3T

Los requisitos mínimos que deben cumplirse para ejecutar la arquitectura 3T son los que se indican a continuación:

- Sistema operativo Windows, Mac OS X o Linux.
- Java Runtime Environment (JRE).
- Dual Core CPU (2.0 Ghz+).
- 1 GB of RAM.

2.2. Ejecución de la arquitectura 3T

Para llevar a cabo la ejecución de las distintas habilidades se han definido varias clases ejecutables, correspondientes a cada una de las habilidades.

Las clases que se han definido para ejecutar las distintas habilidades son:

- **PruebaPortero:** ejecuta la habilidad del portero, en la cual, este busca la portería y una vez situado en la línea de gol, se mantiene controlando la posición de la pelota.
- **PruebaBuscarPelota:** la ejecución de esta clase permite que el robot busque la pelota y se sitúe frente a ella. Para ello realiza el giro óptimo para encontrar la pelota lo más pronto posible.
- **PruebaConducirPelota:** esta clase ejecuta en primer lugar la habilidad de buscar pelota y una vez situado el robot frente a la pelota, este la conduce hasta la portería rival.

- **PruebaDisparo:** esta prueba ejecuta la habilidad de conducir pelota, pero cuando llega a la portería contraria realiza un disparo.

Para elegir el robot que va a ejecutar la habilidad, debe introducirse en el parámetro args [0] el identificador del robot y en el parámetro args[1] y el número cero para indicar si el equipo es el azul y uno si el equipo es amarillo, de tal manera:

Equipos	Color	Número asociado
Equipo azul		0
Equipo amarillo		1

Tabla 52: relación de identificador de equipo

ANEXO IV: MANUAL DE REFERENCIA

ANEXO IV: Manual de Referencia

En el manual de referencia se explica toda la funcionalidad del sistema y el conjunto de elementos que componen el mismo.

El sistema es una arquitectura 3T [36] que permite la ejecución de habilidades en robots de la Robocup [17], cumpliendo las reglas de la competición.

1. Características del sistema

Las características principales de la arquitectura 3T [36] para el control de robots de la Robocup [17] son:

- **Modularidad:** la arquitectura tiene una estructura modular que facilita su futura ampliación con nuevas funcionalidades y la inclusión de nuevas habilidades para los robots.

- **Estándares de la Robocup [17]:** dado que la arquitectura es capaz de recibir e interpretar la información enviada por el sistema SSL-Vision [11], esta sigue el estándar de la competición y puede utilizarse para el control de robots físico a través del sistema SSL-Vision [11].

- **Posibilidad de incluir planes:** puesto que se utiliza una arquitectura híbrida, es posible crear planes para que los robots ejecuten las habilidades determinadas en función de unas condiciones y así conseguir los objetivos fijados.

- **Fiabilidad:** dado que el sistema actualiza las posiciones de los robots continuamente, la información que maneja la arquitectura sobre los robots es fiable y gracias a esto se obtienen movimientos precisos.

2. Usuarios del sistema

El sistema solo dispone de un tipo de usuario, el cual es capaz de ejecutar las habilidades en el simulador grSim [7] [8] o en un robot físico, utilizando el sistema SSL-Vision [11].

Este usuario debe ser una persona que tenga conocimientos sobre el funcionamiento básico de la Robocup SSL [9] para conocer cuál es el funcionamiento de estos robots para llevar a cabo las habilidades.

3. Funcionalidad del sistema

La funcionalidad del sistema se basa en el control de robots a través de la arquitectura 3T [36]. En este proyecto se han implementado nuevas habilidades para que los robots pertenecientes a la categoría SSL [9] de la Robocup [17] sean capaces de ejecutarlas. Hay que destacar que el sistema está diseñado para controlar un robot, por lo tanto, para controlar más de un robot de forma simultánea deberá realizarse en otro ordenador o creando otro proceso de ejecución en el mismo.

Como funcionalidades cabe destacar:

- Ejecución de habilidades simples para los robots de la Robocup [17]. Las habilidades que se han desarrollado en este proyecto y en el proyecto de Alejandro Caparrós Hernando [32] son:
    - Giro óptimo del robot.
    - Buscar portería.
    - Defender portería.
    - Buscar pelota.
    - Conducir pelota.
    - Pase.

  - Esquivar obstáculos.
  - Disparar a portería.

- Ejecución de planes que contienen distintas habilidades y mediante los cuales se obtiene una meta definida previamente.

ANEXO V: REGLAMENTO 2011

ANEXO V: Reglamento 2011

Las leyes que conforman el reglamento de 2011 son:

- LEY 1 – El terreno de juego.
- LEY 2 – El balón.
- LEY 3 – El número de robots.
- LEY 4 – El equipo de robótica.
- LEY 5 – El árbitro.
- LEY 6 – El árbitro asistente.
- LEY 7 - La duración del partido.
- LEY 8 – El inicio y la reanudación del juego.
- LEY 9 - El balón en juego y parado.
- LEY 10 - El método de puntuación.
- LEY 11 - Fuera de juego.
- LEY 12 - Faltas y conducta antideportiva.
- LEY 13 - Tiros libres.
- LEY 14 - El tiro de penalti.
- LEY 15 - El saque de banda.
- LEY 16 - El saque de puerta.

- LEY 17 – El saque de esquina.
- Apéndice A - Reglas de Competencia.
- Apéndice B – Expertos en Visión.

1. LEY 1 – El terreno de juego

- **Dimensiones:**

El campo debe respetar unas dimensiones acordadas que se muestran en la siguiente ilustración, siendo las dimensiones en milímetros:

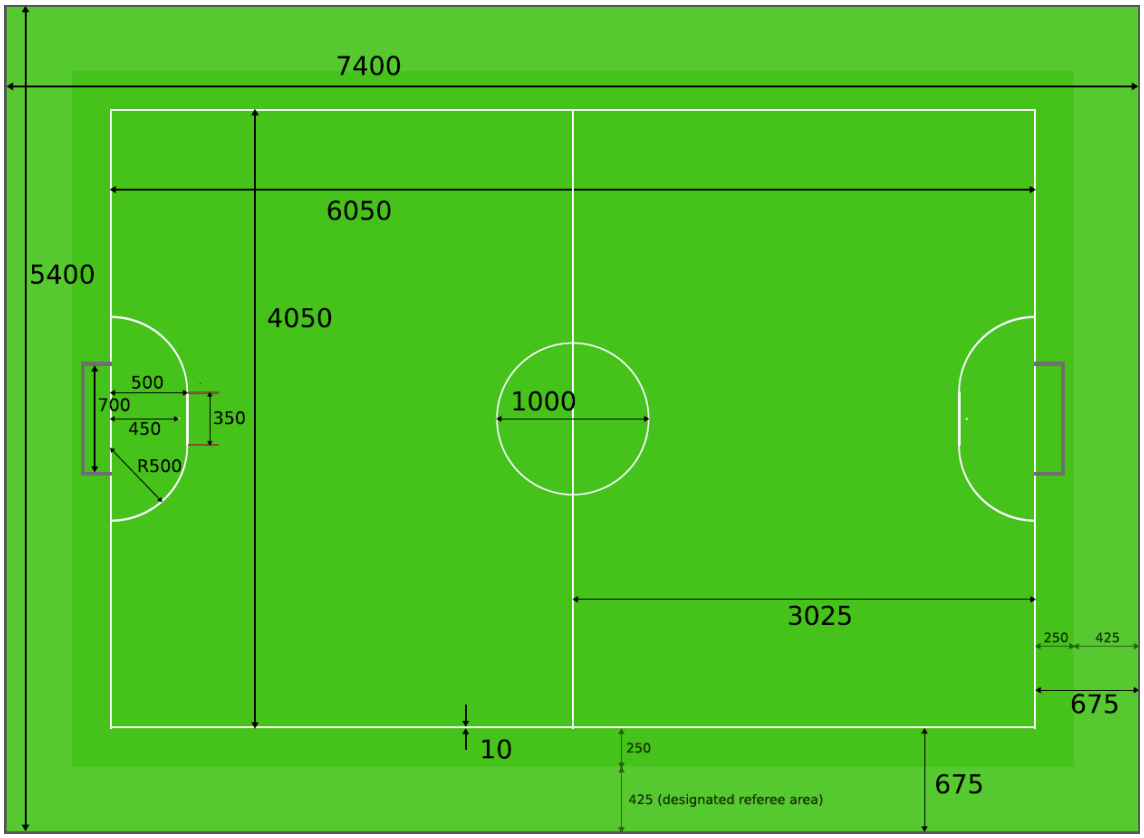


Ilustración 87: representación del campo de la SSL

Por lo que la longitud es de 6.050 mm y la anchura es de 4.050 mm.

- **Superficie del campo:**

La superficie del terreno de juego debe ser verde, con dos posibles opciones de material: fieltro o moqueta. El suelo de debajo de terreno de juego tiene que estar totalmente liso, duro, plano y nivelado.

La zona del campo se verá aumentada 675 mm más allá de las líneas delimitantes. Existirá, además, una pared de 100 mm de altura que impedirá que la pelota y los robots salgan fuera del campo.

- **Líneas de campo:**

El campo tendrá dibujadas ciertas líneas (representadas en la ilustración 7) y equivalen a las áreas de ambos equipos, las bandas y el punto central del campo. Todas las líneas tienen un grosor de 10 mm de ancho y son blancas. La marca del centro del campo indica en el punto medio del campo y se caracteriza por estar rodeado de un círculo de 1000 mm con un diámetro.

- **Área de defensa:**

Existen dos áreas de defensa, una por cada portería, cada una de ellas tiene dos cuartos de círculo con un radio de 500 mm. Estas secciones de círculo se unen por una línea paralela a la línea que parte verticalmente el campo.

- **Punto de penalti:**

Dentro de cada área de la defensa se marca con un punto de penalti que se sitúa a 450 mm desde el punto medio entre los postes y equidistante a ellos. La marca es un círculo de 10 mm de diámetro de pintura blanca.

- **Porterías:**

Existen dos porterías en el terreno de juego colocadas en la parte central de cada límite de gol.

Las porterías deben tener una altura de 180 mm (tal y como se muestra en la ilustración 8) y estar formadas por tres paredes, dos verticales y una horizontal, de 20 mm de espesor de tal forma que:

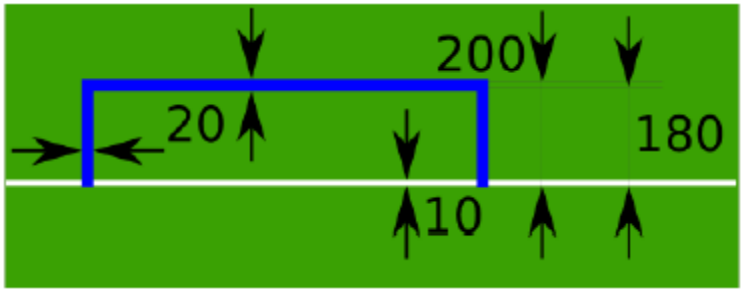


Ilustración 88: representación de la portería de la SSL

El material de las porterías debe ser de color blanco y la cara interna tiene que estar cubierta con un material absorbente de energía, como puede ser la espuma, para ayudar a controlar el impacto de la pelota y disminuir la velocidad de las desviaciones.

- **Equipo para montaje de las cámaras:**

La barra de montaje tendrá 4 metros de longitud sobre el terreno de juego. Esta se colocará por encima de la línea media del campo que divide este. Debe montarse de forma segura para que no se pueda caer bajo ningún concepto, ni doblarse o torcerse durante el juego.

- **Sistema de visión compartida:**

Cada campo está provisto de un sistema centralizado de visión compartida y un conjunto de cámaras compartidas. Este equipo de visión compartida utiliza el software SSL-Vision [11] para comunicar los datos de localización a los equipos vía Ethernet en formato paquete, que será anunciado por los desarrolladores del sistema compartido de visión antes de la competición. Los equipos tendrán que asegurarse de que sus sistemas son compatibles con la salida del sistema compartido de visión y de que sus sistemas son capaces de manejar las propiedades típicas de los datos de sensorización del mundo real proporcionados por el sistema de visión compartida (incluyendo ruido, retraso, o detecciones ocasionales fallidas y errores de clasificación).

Además del equipo de visión compartida, los equipos no pueden montar sus propias cámaras u otros sensores externos, a menos que sean específicamente anunciados o permitidos por los respectivos organizadores de la competición.

El sistema de visión compartida en cada campo está bajo mantenimiento de uno o más expertos de visión. El proceso de selección de estos expertos será comunicado por los organizadores de la competición.

Existen dos cámaras recomendadas y comúnmente utilizadas que son la AVT Stingray F046C [16] y la AVT Marlin F046C. En la siguiente ilustración se representa la cámara Stingray [16], usada para el sistema de visión aéreo:

Ilustración 89: cámara Stingray [16] usada para el sistema de visión de la SSL

La información técnica de ambas cámaras se muestra en la siguiente tabla:

Modelo	Resolución	Velocidad de captura	Codificación de imagen
AVT Stingray [16]	780x580	60 Hz	YUV422
AVT Marlin	780x580	35 Hz	YUV422

Tabla 53: información técnica de las cámaras del sistema de visión

- **Decisiones del Comité Técnico de la SSL:**

Decisión 1: el comité organizador local debe proporcionar una luz difusa de condiciones uniformes, de aproximadamente 500 LUX como mínimo. No se utilizará un equipo de iluminación especial para proporcionar estas condiciones. El brillo no está garantizado ni se espera que esté completamente uniforme a través de la superficie del campo. Se espera que los equipos sean autosuficientes para hacer frente a las variaciones que se produzcan cuando se utiliza la iluminación ambiente. El comité organizador dará a conocer detalles de la iluminación de acuerdo a la competición tan pronto como sea posible.

Decisión 2: ningún tipo de publicidad comercial, ya sea real o virtual, está permitido en el terreno de juego y el equipo de campo (incluidas las redes y las áreas que delimitan) desde el momento en que los equipos entran en el terreno de juego hasta el descanso y desde este hasta el momento en que vuelven a entrar en el terreno de juego hasta el final del partido. En particular, ningún material de publicidad de cualquier tipo puede aparecer dentro de los objetos o las paredes. Los equipos ajenos (cámaras, micrófonos, etc.,) también se ajustarán a estas normas.

Decisión 3: el color específico y la textura de la superficie no se especifican y puede variar de una competición a otra (como los campos de fútbol reales pueden variar). La superficie por debajo de la alfombra estará nivelada y dura. Entre las superficies autorizadas se incluyen: cemento, linóleo, pisos de madera, madera contrachapada, mesas de ping-pong y tableros de partículas. Moqueta o superficies acolchadas no están permitidas. Se pondrá todo el empeño en asegurar que la superficie sea plana, sin embargo, corresponde a los equipos individuales el diseño de sus robots para hacer frente a la ligera curvatura de la superficie.

2. LEY 2 – El balón

- **Calidades y medidas:**

El balón de la competición tiene que tener unas características previamente acordadas que son:

- Ser una bola de golf color naranja.
- Ser totalmente esférica.
- Tener 43 mm de diámetro.
- Pesar 46 gramos de masa.

Similar a la siguiente imagen:

Ilustración 90: pelota utilizada en la SSL [17]

- **Sustitución de una pelota defectuosa:**

  Si el balón se vuelve defectuoso durante el transcurso de un partido:

  - El partido se detiene.
  - El partido se reanudará mediante la colocación de la bola de sustitución en el lugar donde la primera bola se convirtió en defectuosa.
  - Si esto sucede durante una jugada (saque de esquina, tiro libre, penalti, saque de banda o saque inicial) se parará el partido y se reanudará en la misma situación que se detuvo.

  El balón no puede ser cambiado durante el partido sin la autorización del árbitro.

3. LEY 3 – El número de robots

- **Robots:**

Cada partido está compuesto por dos equipos, cada uno con cinco robots. Obligatoriamente uno de ellos debe tener el rol de portero, mientras que el resto puede comportarse como atacante o defensa. El portero debe ser designado antes de comenzar el partido y nunca empezará el partido si los dos porteros no están asignados a un robot.

- **Intercambios:**

Un robot puede ser intercambiado siempre que se respeten estas condiciones:

  - El intercambio solo puede darse durante una interrupción del juego.
  - El árbitro debe ser informado antes del intercambio.
  - El robot que entra en sustitución entrará en el campo de juego después de que el otro robot haya salido del campo.
  - El robot intercambiado entrará al terreno de juego en la línea del centro.

- **Cambiar el portero:**

Cualquiera de los robots puede pasar a ser el portero, siempre que se cumpla:

  - El árbitro esté informado antes de efectuarse el cambio.
  - El cambio se debe realizar durante una interrupción del partido.

- **Robots expulsados:**

Un robot que ha sido expulsado puede ser intercambiado por otro robot que salga del campo.

- **Decisiones del Comité Técnico SSL:**

Decisión 1: cada equipo debe tener un único controlador de robot encargado de realizar el intercambio cuando sea necesario. No hay otros miembros del equipo que puedan invadir el área que rodean el campo. El movimiento de los robots por el controlador no está permitido.

4. LEY 4 – El equipo de robótica

- **Seguridad:**

Todos los robots no deben tener nada en su construcción que pueda ser peligroso para sí mismo, para otro robot o para los seres humanos.

- **Forma:**

El robot debe constar de un cilindro de 180 mm de diámetro y tener una altura máxima de 150 mm. Asimismo, la parte superior del robot debe ajustarse al tamaño y forma del patrón estándar que se describe a continuación. La forma debe estar de acuerdo a la siguiente ilustración:

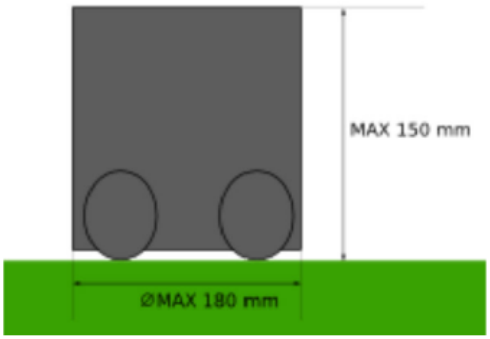


Ilustración 91: representación del robot según las dimensiones permitidas

- **Locomoción:**

Las ruedas del robot deben ser de un material que no dañe a la superficie del juego.

- **Comunicación inalámbrica:**

Los robots deben comunicarse con el ordenador a través de wireless o de redes localizadas fuera del campo.

- **Color del equipo:**

Cada equipo debe adquirir un color: amarillo o azul. Todos los equipos deben poder asignarse cualquiera de los dos colores, ya que a priori no tienen uno asignado. El color del equipo asignado es utilizado como la marca central de la parte superior de todos los robots del equipo y se detalla en el siguiente punto.

- **Patrón estándar:**

Es necesario que todos los equipos participantes sigan el patrón estándar para definir el número de robot y el color asignado.

Para asegurar la compatibilidad con los patrones estandarizados del sistema compartido de visión, todos los equipos deben asegurarse de que los robots tienen una superficie plana en su parte superior con espacio suficiente disponible. El color de la parte superior del robot será de color negro o gris oscuro y tener un acabado mate (no brillante) para evitar los deslumbramientos. El patrón estándar del SSL-Vision [11] está garantizado para reconocer un círculo de 85 mm de radio que cortará la parte frontal del robot a una distancia de 55 mm desde el centro de tal manera:

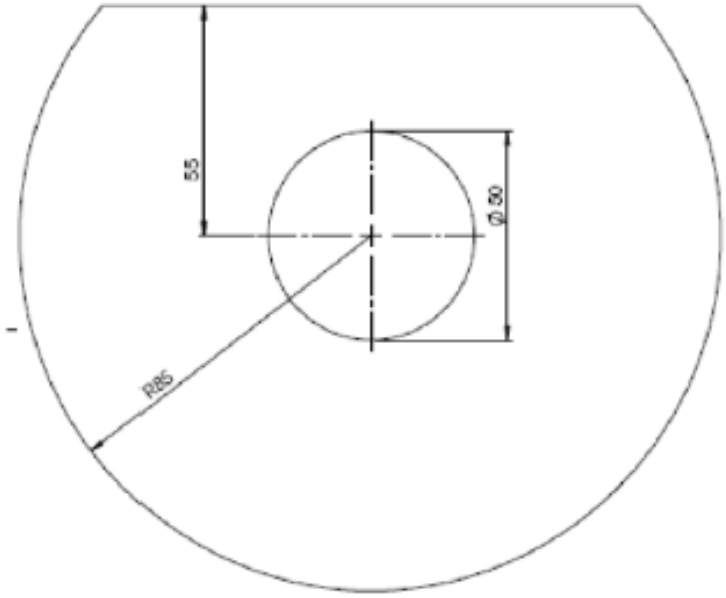


Ilustración 92: representación del área mínima superior del robot

El patrón estándar que se usa para la distribución de los cinco círculos que representan los colores es el siguiente:

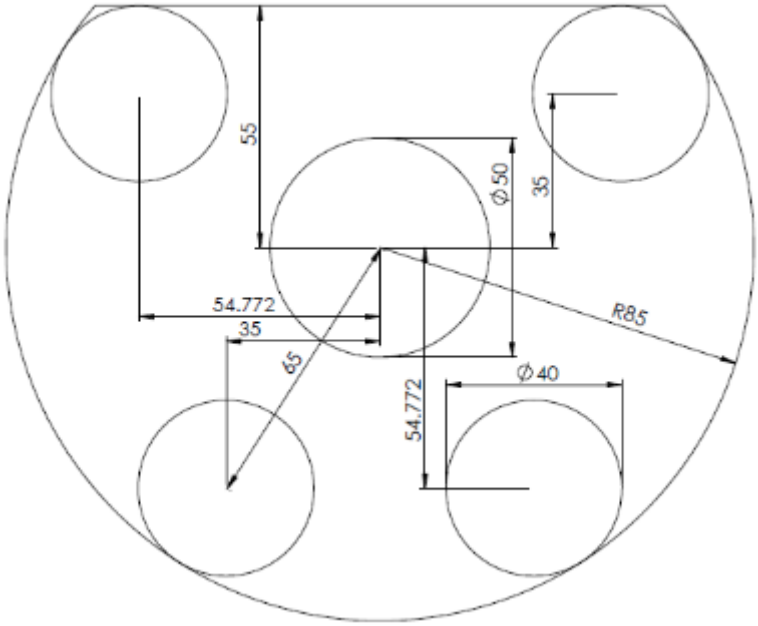


Ilustración 93: representación de los colores identificativos de cada robot

Asimismo, cada robot utilizará un patrón estandarizado con una combinación de colores exclusiva para cada robot, por eso mismo, no puede haber dos robots con la misma combinación de colores. El color del punto central solo puede ser azul o amarillo y determina el equipo al que pertenece el robot.

El papel de colores estandarizado tendrá los colores asignados de la siguiente manera:

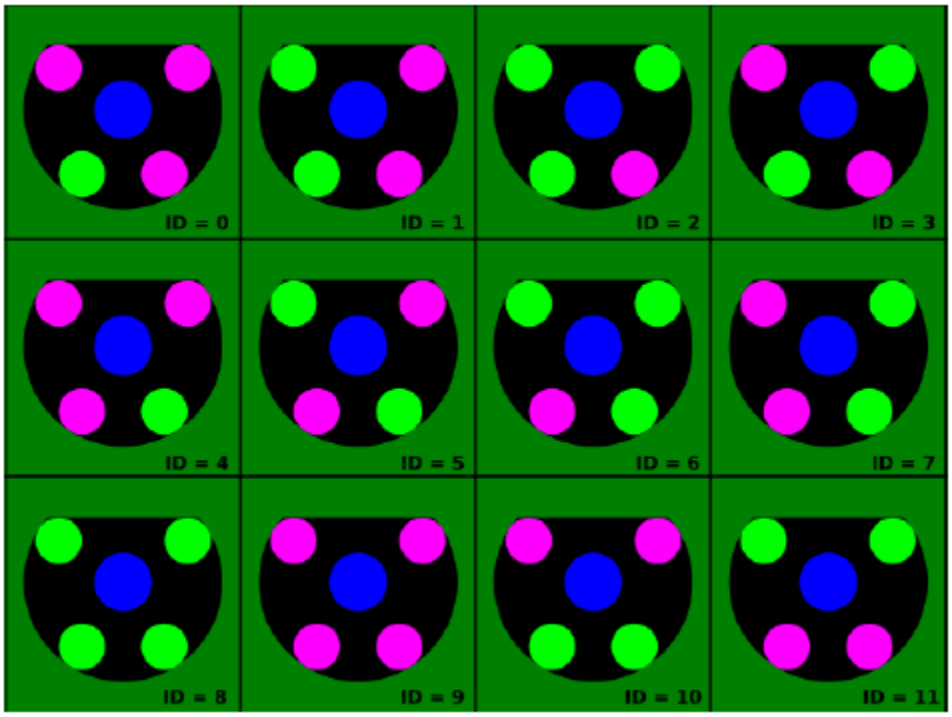


Ilustración 94: distribución de los colores identificativos de cada robot

- **Autonomía:**

Los robots serán completamente autónomos. Las operaciones humanas están prohibidas y no se permite introducir información en el equipo durante el juego, excepto durante el descanso o tiempo de espera.

- **Regateo:**

Los dispositivos que ejercen activamente el movimiento en la pelota para mantener el contacto con el robot se permiten bajo ciertas condiciones:

  - El giro ejercido sobre la pelota debe ser perpendicular al plano del campo.
  - No se permiten dispositivos verticales o parcialmente verticales para mantener la pelota en contacto con el robot.

- Tiene que seguir este esquema:

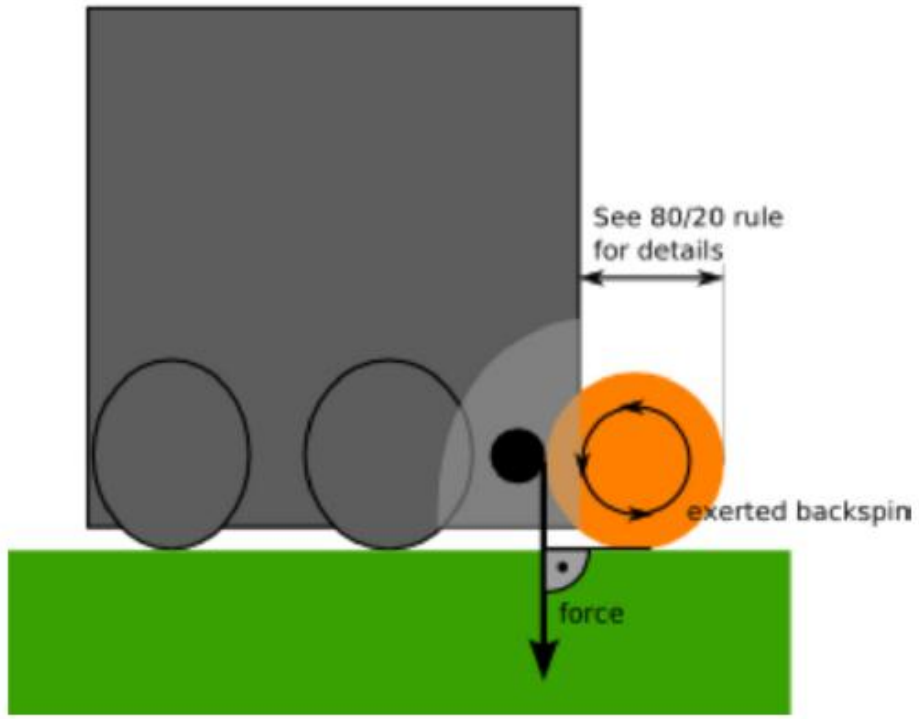


Ilustración 95: disparador del robot

- **Infracciones/Sanciones:**

  Para cualquier tipo de infracción de la presente Ley:

  - El juego no necesita ser detenido.
  - El robot infractor es instado por el árbitro a abandonar el terreno de juego como medida punitiva.
  - El robot deja el campo de juego cuando la pelota deja de estar en juego.

- Ningún robot obligado a abandonar el terreno de juego para sancionar a su equipo vuelve a entrar sin el permiso del árbitro.
- El árbitro comprueba que el equipo del robot es correcto antes de permitir que vuelva a entrar en el terreno de juego.
- Al robot solo se le permite volver a entrar en el terreno de juego cuando el balón está parado.
- Un robot que ha sido obligado a abandonar el terreno de juego debido a una infracción de la presente ley y que entra (o vuelve a entrar) al terreno de juego sin el permiso del árbitro es amonestado y se le mostrará tarjeta amarilla.

- **Reanudación del juego:**

Si el juego es detenido por el árbitro debido a que se hace necesario tomar alguna precaución, el partido se reanudará con un tiro libre indirecto a lanzar por un robot de la parte contraria, desde el lugar donde se encontraba el balón cuando el árbitro detuvo el partido.

- **Decisiones del Comité Técnico de la SSL:**

Decisión 1: **l**os participantes que utilizan las comunicaciones inalámbricas notificarán al comité organizador local el método de comunicación inalámbrica, potencia y frecuencia. El comité organizador local será notificado de cualquier cambio después de la inscripción tan pronto como sea posible. Con el fin de evitar interferencias, un equipo debe ser capaz de seleccionar entre dos frecuencias portadoras antes del partido. El tipo de comunicación inalámbrica se ajustará a las normas legales del país donde se celebre la competición. El cumplimiento de las leyes locales es responsabilidad de los equipos que compiten, no de la Federación RoboCup [17]. El tipo de comunicación

inalámbrica puede también ser restringido por el comité organizador local. El comité de organización local dará a conocer cualquier restricción a la comunidad lo antes posible.

Decisión 2: están permitidos los dispositivos de golpeo y disparo.

Decisión 3: están prohibidas las puntas de metal y el material velcro, con el propósito de la locomoción.

Decisión 4: está totalmente prohibida la comunicación inalámbrica a través de Bluetooth.

Decisión 5: los colores oficiales serán proporcionados por el comité organizador. Los equipos deben usar lo colores oficiales a menos que ambos equipos no estén de acuerdo.

Decisión 6: adhesivos, como pegamento o cinta, no pueden ser utilizados con fines de control del balón o para construir dribladores (sistemas de regateo). El uso de dispositivos que utilizan, por ejemplo, un adhesivo para adherir la pelota a un robot, se consideran una violación de la Regla 12, Decisión 4, por "la eliminación de todos los grados de libertad de la pelota". Además, el uso de adhesivos para cualquier propósito en el robot que provoque residuos sobre el balón o el campo se considera como daño y son sancionados siguiendo la "Ley 12".

Decisión 7: antes del primer partido de la competición se realiza un chequeo de las normas. Si se considera que algún componente de cualquier equipo infringe una norma, el robot debe modificarse para ser compatible, y permitir su participación en los partidos.

5. LEY 5 - El árbitro

- **Autoridad del árbitro:**

Cada partido es controlado por un árbitro que tiene plena autoridad para hacer cumplir las Reglas de Juego en relación con el partido para el que ha sido nombrado.

- **Atribuciones y Deberes:**

El árbitro:

- Hace cumplir las Leyes del Juego.
- Controla el partido en colaboración con los árbitros asistentes.
- Se asegura de que cualquier pelota utilizada cumpla los requisitos de la "Ley 2".
- Asegura que el equipo de robótica cumple con los requisitos de la "Ley 4"
- Informa a los árbitros asistentes de cuándo comienzan y terminan los períodos de tiempo, de conformidad con la "Ley 7".
- Detiene, suspende o termina el partido, a su discreción, por cualquier infracción de las leyes.
- Detiene, suspende o termina el partido debido a interferencias externas de cualquier clase.
- Detiene el partido si, en su opinión, es probable que un robot cause daños graves a los seres humanos, otros robots o a sí mismo y asegura que se retira del terreno de juego.
- Colocar la bola en una posición neutral, si se queda atrapada durante el juego.

- Permite que el juego continúe si el equipo contra el que se ha cometido una falta se beneficia de tal ventaja y penaliza la falta original si no se produce dicha ventaja en ese momento.

- Castiga con la pena máxima cuando un robot comete más de una falta en el mismo tiempo.

- Toma medidas disciplinarias contra los robots infractores y puede expulsarles. No está obligado a tomar esta medida inmediatamente, pero debe hacerlo cuando la pelota sale del terreno de juego.

- Toma medidas contra los miembros del equipo que no se comporten de una manera responsable; puede, a su discreción, expulsarlos del terreno de juego y sus alrededores inmediatos.

- Actúa con el asesoramiento de los árbitros asistentes en relación con incidentes que no ha visto.

- Garantiza que ninguna persona no autorizada invada el terreno de juego.

- Reanudará el partido después de haber sido detenido.

- Proporciona al comité técnico un informe del partido que incluye información sobre cualquier medida disciplinaria adoptada contra los responsables del equipo y cualquier otro incidente ocurrido antes, durante o después del partido.

- Comprueba el estado del sistema de visión compartida con el/los experto/os en visión (véase en "Apéndice B" de este mismo punto) antes de cada partido.

- Recoge la confirmación del experto/os en visión de que ambos equipos reciben los datos de localización del sistema compartido de visión correcta y exactamente.

- Para el juego cuando el/los experto/os en visión lo digan durante un partido y deje que el/los experto/os en visión diagnostiquen y arreglen el problema. Si el/los experto/os en visión confirman que el problema está resuelto entonces el juego será reanudado inmediatamente.

- **Decisiones del árbitro:**

El árbitro solo puede cambiar una decisión al darse cuenta de que esta es incorrecta o si se lo aconseja el árbitro asistente, siempre que continúe el juego parado.

- **Equipo de señalización del árbitro:**

El dispositivo necesario se suministra para convertir las señales del árbitro en serie y Ethernet. Las señales de comunicación se transmiten a ambos equipos. Los equipos serán operados por el árbitro asistente. Los detalles del equipamiento serán suministrados por la organización local de Comité antes de la competición.

- **Señales del árbitro:**

Durante el juego, el árbitro dará la señal de inicio y fin del juego en la forma habitual. El árbitro asistente enviará señales que reflejarán las decisiones del árbitro a cada uno de los equipos. Ninguna interpretación de las señales del árbitro por los operadores humanos está permitida.

La señal del silbato indica que el árbitro ha parado el juego y que todos los robots deben separarse 500 mm de la pelota para que el árbitro pueda colocar la pelota para reiniciar el sistema. Todos los robots tienen la obligación de 500 mm de la bola mientras esta se mueve a la posición de reiniciar.

Cuando se produce un gol (véase "Ley 10"), o una precaución o se produce una salida de la pelota del campo de juego (véase "Ley 12"), una señal de información es enviada a los equipos para indicar la decisión del árbitro.

La señal de reinicio indicará el tipo de reinicio.

Los robots deben moverse a posiciones legales a la recepción de esta señal. Para reiniciar otras acciones que no sean un saque inicial (véase "Ley 8") o un penalti (véase "Ley 14"), el robot que saque puede patear el balón cuando esté listo, sin esperar más señales del árbitro.

Para un saque inicial (véase "Ley 8"), o un penalti (véase "Ley 14"), una señal de arranque será enviada para indicar que el robot que lance puede proceder. Esta señal será distinta a otros tipos de señales de reinicio del juego.

Se enviarán señales que indiquen los períodos de tiempo de espera y el tiempo perdido también se enviará cuando sea necesario.

Se considerará que el árbitro ha dado una señal cuando el árbitro asistente envíe esta señal a los equipos mediante las comunicaciones.

- **Decisiones del Comité Técnico de la SSL:**

Decisión 1: el árbitro (o el árbitro asistente, según el caso) no es el responsable de las siguientes actividades:

- Cualquier tipo de perjuicio sufrido por un componente del equipo o un espectador
- Cualquier daño a la propiedad de cualquier tipo.
- Cualquier otra pérdida sufrida por cualquier persona, club, empresa, asociación u otro organismo, que es debido o que puede ser debido a cualquier decisión que se tome en virtud de los términos de las leyes del juego o en el caso de los procedimientos normales requeridos para conservar, reproducir y controlar un partido. Esto puede incluir:
    - La decisión de que la condición del terreno de juego o sus alrededores son tales como para permitir o no que un partido que tenga lugar.

- La decisión de abandonar un partido por cualquier razón.
- Una decisión en cuanto a la condición de los accesorios o equipos utilizados durante un partido como el campo y la pelota.
- La decisión de detener o no detener un partido debido a la interferencia del espectador o cualquier problema en el área de los espectadores.
- La decisión de detener o no detener el juego para permitir que un robot dañado pueda ser eliminado del campo de juego para su reparación.
- La decisión de solicitar o insistir en que un robot dañado se retire del terreno de juego para su reparación.
- La decisión de permitir o no permitir ciertos colores.
- La decisión (en la medida en que esta puede ser su responsabilidad) para permitir o no permitir a las personas (incluyendo el equipo o funcionarios del estadio, oficiales de seguridad, fotógrafos u otros medios, representantes, etc.) estar presentes en las inmediaciones del campo de juego
- Cualquier otra decisión que pueda tomar de acuerdo con las Reglas de Juego o de conformidad con sus obligaciones bajo los términos de la Federación RoboCup [17] o las normas o regulaciones bajo las cuales se juega el partido.

Decisión 2: los hechos relacionados con el partido serán incluidos tanto si se marca un gol o no, así como el resultado del encuentro.

Decisión 3: el árbitro usará un bastón negro o algún otro dispositivo o herramienta para el reposicionamiento de la pelota, reduciendo el riesgo de interferencias con los sistemas de visión o SSL-Vision [11].

Decisión 4: el árbitro podrá ser asistido por aplicaciones autónomas de arbitraje proporcionadas por uno o ambos de los equipos que compiten, si ambos equipos están de acuerdo. Asimismo, el árbitro podrá ser asistido por una aplicación autónoma o semi-autónoma proporcionada por un equipo que no participe en el partido, según el criterio del árbitro; teniendo en cuenta que la aplicación deberá ser operada y monitorizada de manera neutral.

Decisión 5: la región externa de la superficie del campo que es más allá de 250 mm de distancia de la línea divisoria y es utilizada como zona de paseo designado por el árbitro y/o el árbitro asistente durante el juego. Los equipos deben controlar a sus robots para permanecer fuera de esta zona para no interferir con los árbitros. Los árbitros no son responsables de cualquier obstrucción a los robots o sistemas de visión dentro del área. Sin embargo, los árbitros deberán llevar ropa y zapatos que no contengan ningún color reservado para la bola o los marcadores de los robots.

6. LEY 6 - El árbitro asistente

- **Deberes:**

Las funciones del árbitro asistente, respetando las decisiones del árbitro, son las siguientes:

- Actuar como medidor de tiempo y llevar un registro del juego.
- Operar el equipo de comunicaciones para transmitir las señales del árbitro sobre los enlaces de comunicaciones.

- Supervisar a los operadores de robots para evitar que señales ilegales sean enviadas a los robots.
- Indicar cuándo se solicita un intercambio.
- Indicar cuándo una mala conducta o cualquier otro incidente se ha producido fuera de la vista del árbitro.
- Indicar cuándo se comete una infracción si los asistentes se acercan más a la acción que el árbitro (esto incluye, en determinadas circunstancias, las faltas cometidas en la defensa del área).
- Indicar si en los penaltis, el guardameta se ha movido hacia delante antes de que el balón ha sido golpeado y si el balón ha cruzado la línea de meta.

- **Asistencia:**

Los árbitros asistentes ayudan al árbitro a controlar el partido, de acuerdo a las Leyes del Juego. En el caso de una interferencia, el árbitro liberará de sus obligaciones al árbitro asistente.

- **Decisiones del Comité Técnica de la SSL:**

Decisión 1: se utilizará un segundo árbitro asistente si es posible. El segundo árbitro asistente se encarga de ayudar al árbitro durante el juego y a vigilar el cumplimiento de las Leyes del Juego.

7. LEY 7 - La duración del partido

- **Períodos del juego:**

El partido tiene dos periodos iguales de 10 minutos, salvo mutuo acuerdo del árbitro y los dos equipos. Cualquier acuerdo para alterar los períodos de juego (por

ejemplo, para reducir cada mitad a 7 minutos a causa de un horario limitado) debe hacerse antes el inicio del juego y deben cumplir con las normas de competencia.

- **Intermedio:**

Los equipos tienen derecho a un intermedio a mitad del tiempo medio de un intervalo que no deberá exceder de 5 minutos. Las normas de competencia deben indicar la duración del intermedio o descanso. La duración del descanso puede ser modificado únicamente con el consentimiento de ambos equipos y el árbitro.

- **Tiempos de espera:**

A cada equipo se le otorga cuatro tiempos de espera al comienzo del partido. Se permite un total de 5 minutos para todos los tiempos de espera. Por ejemplo, un equipo puede pedir tres tiempos de espera de un minuto de duración y, posteriormente, sólo tienen un tiempo de espera de hasta dos minutos de duración. Los tiempos de espera sólo pueden ser consumidos durante una interrupción del juego. El tiempo es controlado y registrado por el árbitro asistente.

- **Compensación por el tiempo perdido:**

Se tiene en cuenta cualquier período de tiempo perdido debido a la evaluación de los daños en los robots, la eliminación de los robots dañados en el terreno de juego y cualquier otra causa que suponga la pérdida de tiempo. La compensación por el tiempo perdido es a discreción del árbitro.

- **Tiempo extra:**

Serán aplicadas las normas de competencia, podrán prever dos tiempos suplementarios iguales, según las condiciones de la "Ley 8".

- **Abandonar el partido:**

Un partido abandonado se repite mientras las normas no dispongan de lo contrario.

- **Decisiones del Comité Técnica de la SSL:**

Decisión 1: el comité hará todo lo posible para proporcionar acceso a los equipos de la competición al menos dos horas antes del inicio de la competición. También se esforzará por permitir al menos una hora de tiempo de configuración antes de cada partido. Los participantes deben ser conscientes, sin embargo, que puede ocurrir que este tiempo no se pueda proporcionar.

8. LEY 8 - El inicio y la reanudación de juego

- **Preliminares:**

Si ambos equipos tienen una frecuencia preferida común para las comunicaciones inalámbricas, el comité organizador local asignará la frecuencia para la primera mitad del partido. Si ambos equipos tienen un color preferido común, el comité organizador local asignará el color de la primera la mitad del partido.

La forma de asignar esta información es a través del lanzamiento de una moneda y el equipo que gane el sorteo decidirá qué portería atacará en la primera la mitad del partido, cambiando la portería en el segundo tiempo. El otro equipo realiza el saque para comenzar el partido. En la segunda parte, se procederá al contrario, el equipo que gane el sorteo tiene el saque inicial para comenzar la segunda mitad del partido.

Sin embargo, si los equipos no están de acuerdo para cambiar campos, pueden permanecer en los mismos que el primer tiempo con el consentimiento del árbitro.

- **Saque desde el centro del campo:**

Un saque desde el centro del campo es una forma de iniciar o reiniciar el juego en las siguientes condiciones:

  - En el inicio del partido.

- Después de la anotación de un gol.
- Al comienzo de la segunda parte del partido.
- Al comienzo de cada período de tiempo adicional.

El procedimiento a seguir es el siguiente:

- Todos los robots se encuentran en su propia mitad del campo.
- Los oponentes del equipo que realiza el saque del partido están por lo menos a 500 mm de la bola hasta que el balón está en el juego.
- El balón está parado en el centro del campo hasta que el árbitro da la señal de saque.
- El árbitro da la señal de saque.
- La pelota está en juego cuando es pateada y se mueve hacia delante.
- El lanzador no podrá tocar el balón por segunda vez hasta que haya tocado a otro robot.

- **Situación de la pelota:**

La colocación de la pelota es una forma de reanudar el partido tras una parada temporal que haya sido necesaria, mientras la pelota estaba en juego, por alguna razón no mencionada en las normas.

- **Circunstancias especiales:**

Un tiro libre concedido al equipo defensor dentro de su propia área de defensa se realiza desde la posición de tiro cercana a donde se produjo la infracción, elegida por el propio equipo.

Un tiro libre concedido al equipo atacante en el área de defensa de sus oponentes es lazado desde la posición legal predefinida de tiro libre más cercana al lugar donde se produjo la infracción.

Una pelota que esté en condiciones de reiniciar el partido después de que la jugada ha sido detenida temporalmente en el interior de la zona defensiva se coloca sobre la posición legal de tiro libre más cercana a donde se encontraba el balón cuando la jugada se detuvo.

9. LEY 9 - El balón en juego y parado

- **Balón parado:**

La pelota está parada cuando se han cruzado los límites del campo por suelo o aire o el juego ha sido detenido por el árbitro.

Si la pelota sale fuera del terreno de juego, los robots deben seguir estando a 500 mm de la bola mientras esta se coloca, hasta que la señal de reinicio sea dada por el árbitro.

- **Balón en juego:**

La pelota está en juego en cualquier otro momento.

- **Infracciones/Sanciones:**

Si en el momento en que la pelota entra en el terreno de juego un miembro del equipo que saca está a una distancia inferior de 200 mm de la zona de defensa del oponente, se concede un tiro libre indirecto al equipo contrario. Este tiro se lanzará desde la ubicación el que se encontraba la pelota cuando se produjo la infracción (véase "Ley 13").

Si después de que la pelota entra en el terreno de juego el robot que tira toca el balón por segunda vez antes de que lo haya tocado otro robot, se concede un tiro libre indirecto al equipo contrario. Este tiro se lanzará desde la ubicación el que se encontraba la pelota cuando se produjo la infracción (véase "Ley 13").

Si después de que la pelota entra en el terreno de juego el robot que tira sostiene el balón antes de que lo haya tocado otro robot, se concede un tiro libre indirecto al equipo contrario. Este tiro se lanzará desde la ubicación el que se encontraba la pelota cuando se produjo la infracción (véase "Ley 13").

Si después de darse una señal para reiniciar el juego el balón no entra en el juego en menos de 10 segundos, se procederá a detener el juego por una señal del árbitro, los robots se moverán a 500 mm de la pelota y se indicará un saque neutral.

- **Decisiones del Comité Técnica de la SSL:**

Decisión 1: para todos los reinicios en que las leyes establecen que la pelota está en juego bien sea golpeándola o regateando, los robots deben claramente hacer lo posible para que esta se mueva. Se entiende que la pelota puede permanecer en contacto con el robot o ser golpeada por el robot varias veces a corta distancia, pero bajo ninguna circunstancia el robot mantendrá el contacto o se mantendrá tocando la pelota después de haber recorrido una distancia de 50 mm, a menos que el balón haya tocado antes a otro robot. Los robots pueden utilizar los dispositivos de regateo y patada en los lanzamientos de las faltas.

Decisión 1: la zona de exclusión de 200 mm de la zona de la defensa del oponente se designa para permitir a la defensa de los equipos tomar una posición defensiva contra un lanzamiento sin la interferencia de los oponentes.

10. LEY 10 - El método de puntuación

- **Puntuación de gol:**

Se marca un gol cuando el conjunto de la pelota pasa por encima de la línea de meta, entre las paredes de meta o por debajo del travesaño, sin que se haya cometido una infracción de las reglas de juego con anterioridad por parte del equipo que anota el gol.

- **Equipo ganador:**

El equipo que anota el mayor número de goles durante un partido es el ganador. Si los dos equipos marcan un número igual de goles, o si no marcó ningún gol, el partido se da como empatado.

- **Normas de competencia:**

Para los partidos que terminan en un empate, las normas de competencia podrán estipular un tiempo suplementario u otro método, siempre aprobado por la Federación RoboCup [17], para determinar el ganador del partido.

11. LEY 11 - Fuera de juego

La regla del fuera de juego no se usa en esta competición.

12. LEY 12 - Faltas y conducta antideportiva

- **Tiro libre directo:**

Un tiro libre directo es concedido al equipo adversario si un robot comete cualquiera de las siguientes infracciones:

- Hacer contacto sustancial con un oponente.
- Retener un oponente.
- Sostener el balón deliberadamente (excepto para el guardameta dentro de su ámbito de la defensa propia).
- El segundo robot de la defensa y ocupa el área de la defensa del equipo de tal forma que afecte sustancialmente el juego.

- **Tiro libre indirecto:**

Un tiro libre indirecto es concedido al equipo adversario si el guardameta, dentro de su propia área defensiva, comete cualquiera de las siguientes infracciones:

- Transcurren más de quince segundos mientras sostiene la pelota antes de liberarla de su posesión.
- Tiene el balón de nuevo después de haber sido liberado de su posesión y no lo ha tocado otro robot.

Un tiro libre indirecto, además, es concedido si:

- Entra en contacto con el portero y el punto de contacto está en el área de defensa.
- Conduce el balón a una distancia superior a 500 mm.
- Toca la pelota de tal manera que la parte superior de la bola alcanza una altura superior a 150 mm respecto del suelo y el balón entra en la meta de su oponente, salvo que haya sido tocado previamente por un compañero de equipo, o que manteniéndose en contacto con el suelo alcance dicha altura y entre en la meta de su oponente debido a un rebote.
- Patea la pelota de tal manera que supera los 10 m/s de velocidad.

- Comete cualquier otra infracción no citada, por la que se interrumpirá el juego por precaución o para expulsar al robot.

- **Tiro de penalti:**

Un tiro de penalti se otorga si alguna de las anteriores cuatro infracciones es cometida por un robot dentro del área de defensa propia, independientemente de la posición de la pelota, siempre y cuando esta esté en juego.

- **Sanciones disciplinarias:**

Un equipo será amonestado y recibirá una tarjeta amarilla si uno de os robots:

- Es culpable de conducta antideportiva.
- Es culpable de graves y violentos contactos.
- Infringe persistentemente las Reglas de Juego.
- Retrasa la reanudación del juego.
- No respeta la distancia reglamentaria cuando el juego se reanude con un saque de meta, saque de esquina o tiro libre.
- Modifica o provoca daños en el campo o pelota.
- Deliberadamente entra o se desplaza dentro de la zona de tránsito del árbitro.

Al recibir una tarjeta amarilla, un robot del equipo penalizado debe moverse inmediatamente fuera y ser sacado del campo. Después de dos minutos de juego, tiempo que medirá el árbitro o el asistente, el robot puede entrar de nuevo en el campo en la próxima parada del juego.

- **Expulsión de sancionados:**

Un equipo recibe la tarjeta roja si uno de los robots o el equipo es culpable de un comportamiento antideportivo grave. El número de robots en el equipo se reduce en uno después de cada tarjeta roja.

- **Decisiones del Comité Técnico de la SSL:**

Decisión 1: un contacto calificado como importante es aquel suficiente para desalojar al robot de su orientación actual, posición o movimiento, en el caso de que se esté moviendo. Cuando los dos robots se mueven a velocidades similares y la causa de contacto no es evidente, el árbitro permitirá que el juego continúe. Esta Ley está diseñada para proteger a los robots que son lentos o permanecen quietos en el momento del contacto y, por tanto, deben ser detectados por los sistemas de evasión de obstáculos.

Decisión 2: existen precauciones para evitar contactos graves y violentos. Ejemplos claros de infracciones son: el movimiento incontrolado, las malas evasiones de obstáculos, empujar o girar rápidamente mientras se está junto a un oponente. En un escenario típico, el árbitro podrá advertir al equipo, y se espera que se modifique su sistema a fin de reducir la violencia de su juego. Si el árbitro aún no está satisfecho, dictará una amonestación.

Decisión 3: un robot que se coloca en el campo, pero claramente no es capaz de moverse, será sancionado por conducta antideportiva.

Decisión 4: se considera que un robot está reteniendo el balón si tiene el control de la pelota mediante la eliminación de todos sus grados de libertad, normalmente, sujetando la pelota con el cuerpo del robot o rodeándolo para evitar el acceso de otros. El 80% de la superficie de la bola debe ser visible desde arriba, de forma que estará fuera de la parte convexa del robot. Otro robot debe ser capaz de quitar el balón al que lo posee. Esta limitación se aplica también a todos los dispositivos de regateo y golpeo, incluso si tal infracción es momentánea. Se muestra en la siguiente ilustración, siendo la vista aérea del robot:

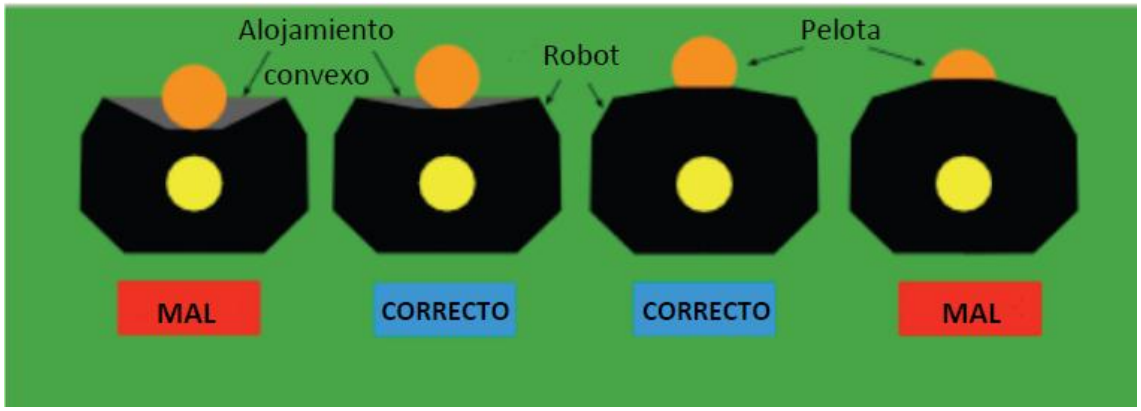


Ilustración 96: posición de la pelota respecto al robot

Decisión 5: un robot comienza el regateo cuando tiene contacto con el balón y se detiene el regateo cuando hay una separación observable entre la pelota y el robot.

La restricción de la distancia en el regateo sirve para evitar que un robot con una mecánica superior pueda tener un indiscutible control de la pelota en el ataque. La restricción de la distancia permite a los atacantes dar y recibir pases, girar con el balón y detenerse con la pelota. Los sistemas de regateo se pueden utilizar para regatear a grandes distancias con el balón, siempre y cuando el robot pierda habitualmente la posesión. El comité técnico espera que la regla de distancia sea auto-forzada, es decir, que los equipos dispongan de un software que la cumpla con antelación y se les pueda pedir una demostración previa a una competición.

El sistema de regateo se representa en la siguiente ilustración:

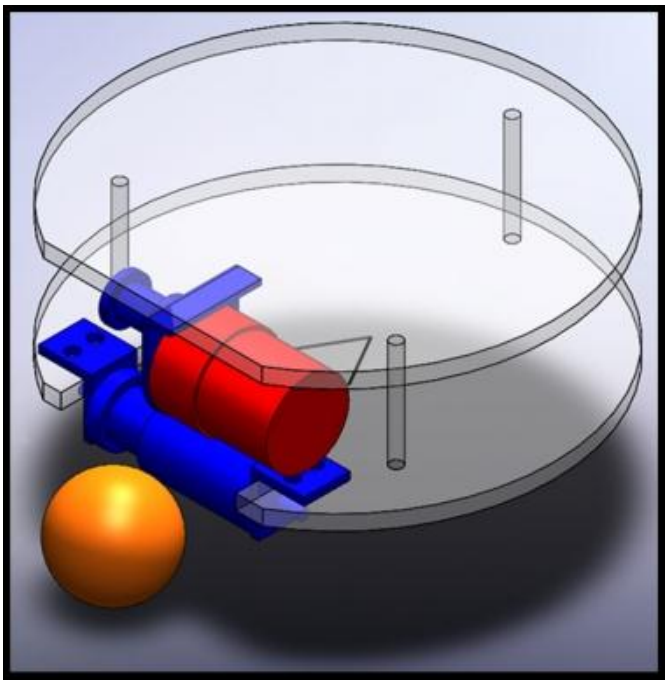


Ilustración 97: sistema dribler

Los árbitros podrán seguir señalando faltas y pueden señalar amonestaciones por situaciones de violación sistemática de dicha regla.

Decisión 6: la limitación de velocidad de disparo de la pelota previene que un robot con un disparo mecánicamente superior tenga demasiada ventaja sobre sus oponentes o que se patee la pelota a una velocidad no apta para los espectadores.

Decisión 7: cita las siguientes condiciones:

- Si un robot lanza la pelota por encima de un adversario y el balón, posteriormente, entra en la propia portería, después de permanecer por debajo de 15 mm de altura todo el tiempo tras haber tocado al robot oponente, el equipo oponente también obtiene un tanto.

- Si un robot lanza la pelota por encima de un adversario y el balón, posteriormente, entra en propia portería, después de haber estado por

encima de 150 mm durante algún tiempo (y no habiendo estado en contacto permanente con el suelo), después de tocar al robot oponente, el equipo oponente no puntúa.

Decisión 8: la infracción cometida al entrar deliberadamente en la zona de tránsito del árbitro fue añadido para desalentar a los equipos de la conducción de vehículos por esta zona para obtener ventajas tácticas. En particular, debe prevenir que los equipos exploten el hecho de que otros equipos no podrían tener cobertura de visión del árbitro caminando por dicha área. Se entiende que, en ocasiones, un robot puede entrar en la zona si está fuera de control, o si ha sido empujado a esta área. Estos casos no deben ser considerados infracciones. Sin embargo, la decisión final en cuanto a lo que constituye una violación deliberada del reglamento la decide el árbitro.

13. LEY 13 - Tiros libres

Serán directos o indirectos. Tanto en los directos como en los indirectos, la pelota debe ser parada cuando se comete la falta y el robot lanzador no puede tocar el balón por segunda vez hasta que lo haya tocado otro robot.

- **Tiro libre directo:**

Si un tiro libre entra directamente en la meta del oponente, se concede un gol.

Si un tiro libre entra directamente a gol en propia meta, se concede un gol al equipo oponente.

- **Tiro libre indirecto:**

Si un tiro libre indirecto entra directamente en la meta del oponente, se concede un saque de puerta.

Si un tiro libre indirecto entra directamente en la propia meta del equipo, se concede un saque de esquina al equipo contrario.

- **Procedimiento par los tiros libres:**

Si el tiro libre se concede dentro del área de defensa, el tiro libre se lanza desde un punto a 600 mm de la línea de gol y a 100 mm desde la línea de contacto más cercana a donde se produjo la infracción.

Si el tiro libre es concedido al equipo atacante a 700 mm de la zona de defensa, la pelota se traslada al punto más cercano a 700 mm desde el área de defensa.

Si el tiro libre se lanza desde el lugar donde se produjo la infracción.

Todos los robots oponentes se colocarán a una distancia mínima de 500 mm de la pelota y esta está en juego cuando es dada y se mueve.

- **Infracciones/Sanciones:**

Si cuando se lanza un tiro libre, el oponente más cercano a la bola no se encuentra a la distancia requerida, el tiro se repetirá.

Para otro tipo de infracción relacionada con esta Ley, el tiro se repetirá.

14. LEY 14 - El tiro de penalti

Un tiro de penalti se otorga contra un equipo que comete una de las cinco infracciones por las que se concede un tiro libre directo, dentro de su área de defensa y mientras la bola está en juego. El tiempo que se emplee para un tiro de penalti se añadirá al final de cada mitad o al final de los períodos de tiempo extra

- **Posición de la bola y los robots:**

Para el lanzamiento del penalti, el balón se coloca en el punto de penalti. El robot que lanza el penalti tiene que estar debidamente identificado. Asimismo, el guardameta defensor se mantiene entre los postes de la portería, toca la línea de meta y se orienta hacia la cara externa de la meta, hasta que el balón es lanzado. Se le permite el movimiento antes de que el balón haya sido golpeado, siempre y cuando no se infrinja alguna de las anteriores restricciones. El resto de los robots se encuentran dentro del terreno de juego y detrás de una línea paralela a la línea de gol y a 400 mm detrás del punto penalti.

- **El árbitro:**

El árbitro dará la señal de tiro de penalti cuando los robots estén colocados correctamente como marca el anterior punto. Es el árbitro quien decide si el tiro penal se ha completado.

- **Procedimiento:**

El robot que lanza el penalti, golpea la pelota hacia delante, no pudiendo tocar el balón por segunda vez hasta que haya sido tocado por otro robot. La pelota está en juego cuando es golpeada y se mueve hacia delante.

Cuando un tiro de penalti se lanza durante el curso normal del juego, el tiempo se ha ampliado en la primera mitad o al final del partido para permitir que un lanzamiento de penalti sea lanzado, se concede un gol si entra directamente o si antes de que el balón pase entre los postes y por debajo del travesaño, la pelota toca uno o ambos postes de la portería, el travesaño y/o el portero.

- **Infracciones/Sanciones:**

Si el árbitro da la señal de un tiro de penalti y, antes de que el balón esté en juego, se produce una de las siguientes situaciones:

- El robot que lanza el penalti infringe las Reglas del Juego:
    - El árbitro permitirá que continúe la jugada.

  - Si el balón entra en la meta, se repetirá el tiro.
  - Si el balón no entra en la meta, el lanzamiento no se repetirá.

- El guardameta infringe las Reglas de Juego:
  - El árbitro permitirá que continúe la jugada.
  - Si el balón entra en la meta, se concede un gol.
  - Si el balón no entra en la meta, se repetirá el tiro.

- Un compañero del robot que lanza entra en el área de los 400 mm detrás del punto de penalti:
  - El árbitro permitirá que continúe la jugada.
  - Si el balón entra en la meta, se repetirá el tiro.
  - Si el balón no entra en la meta, el lanzamiento no se repetirá.
  - Si el balón rebota en el guardameta, el travesaño o el poste de la meta y es tocado por el presente robot, el árbitro interrumpirá el juego y reanudará el partido con un tiro libre indirecto a favor del equipo que defiende.

- Un compañero del guardameta penetra en la zona de los 400 mm detrás del punto de penalti:
  - El árbitro permitirá que continúe la jugada.
  - Si el balón entra en la meta, se concede un gol.
  - Si el balón no entra en la meta, se repetirá el lanzamiento.

- Un robot de ambos equipos, de la defensa y el equipo atacante, infringen las Reglas de Juego:

- El tiro se repetirá.
- Tras el cumplimiento de la pena toda infracción que se enumeran en la "Ley 9" se tratará en consonancia.
- El balón es tocado por un agente externo, y se mueve hacia delante:
- El tiro se repetirá.
- El balón rebota en el terreno de juego tras tocar al guardameta, el travesaño o los postes, y es entonces tocado por un agente externo:
- El árbitro detiene el juego.
- El juego se reanudará con un toque neutral en el lugar donde la pelota tocó al agente externo (véase "Ley 13").

15. LEY 15 - El saque de banda

Un saque de banda es un método de reinicio el juego. Sin embargo, un gol no puede ser marcado directamente desde un saque de banda.

Un saque de banda se concede:

- Cuando la totalidad de la pelota pasa por encima del límite de contacto (línea de banda), ya sea por tierra o por aire.
- Desde el punto, a 100 mm, perpendicular a la línea de banda donde la pelota cruzó el límite.
- Al equipo contrario al último robot que toca el balón.

- **Procedimiento:**
    - El árbitro pone el balón en la posición designada.
    - Todos los robots oponentes se distancian por lo menos 500 mm de la pelota.
    - La pelota está en juego cuando es pateada y se mueve.
- **Infracciones/Sanciones:**

Cuando un saque de banda se realiza y un oponente está más cercano a la pelota de la distancia requerida el saque de banda, se repetirá. Para cualquier otra infracción el tiro se repetirá.

16. LEY 16 - El saque de puerta

Un saque de puerta es un método de reinicio el juego. Un gol puede ser anotado directamente por un saque de puerta, pero sólo si entra en la portería contraria.

Un saque de puerta es otorgado cuando la totalidad de la pelota, después de haber sido tocada por un robot del equipo atacante, pasa por encima de la línea de límite de gol, ya sea por tierra o aire y no se concede un tanto de conformidad con la "Ley 10".

- **Procedimiento:**
    - La pelota es golpeada desde el punto a 500 mm de la línea de gol y a 100 mm de la línea de banda más cercano a donde la pelota pasó por la línea de gol.
    - Los opositores estarán a 500 mm de la bola hasta que el balón está en juego.

- El lanzador no puede jugar el balón por segunda vez hasta que haya tocado a otro robot.

- La pelota está en juego cuando es golpeada y se mueve.

- **Infracciones/Sanciones:**

Toda infracción que se enumeran en la "Ley 9" se tratará en consonancia. Para cualquier otra infracción de la presente Ley el tiro se repetirá.

17. LEY 17 – El saque de esquina

Un saque de esquina es un método de reinicio el juego. Un gol puede ser anotado directamente de un saque de esquina, pero solamente contra el equipo contrario.

Un saque de esquina se concede cuando la totalidad de la pelota, después de haber tocado un robot del equipo defensor, pasa por encima de la línea de gol, ya sea por tierra o aire, y no se concede un gol de conformidad con la "Ley 10".

- **Procedimiento:**

  - La pelota es golpeada desde la esquina más cercana, a 100 mm en la línea de gol y de la línea de banda.

  - Los contrarios siguen estando a 500 mm de la bola hasta que el balón está en juego.

  - El lanzador no puede jugar el balón por segunda vez hasta que haya tocado a otro robot.

  - La pelota está en juego cuando es golpeada y se mueve.

- **Infracciones/Sanciones:**

Toda infracción que se enumeran en la "Ley 9" se tratará en consonancia. Para cualquier otra infracción de la presente Ley el tiro se repetirá.

18. Apéndice A - Reglas de Competencia

Este apéndice describe los procedimientos adicionales necesarios para completar las reglas de la SSL.

- **Tiempo extra:**

Si el resultado del partido es de empate después del final del segundo período y el partido necesita terminar con un ganador, se jugará un tiempo extra (véase "Ley 7" y "Ley 10"). Antes de la primera mitad del tiempo extra habrá un intervalo que no deberá exceder de 5 minutos.

- **Periodos de juego durante el tiempo extra:**

El tiempo extra dura dos períodos iguales de 5 minutos, salvo mutuo acuerdo entre el árbitro y los dos equipos participantes. Cualquier acuerdo para alterar los períodos de tiempo extra debe hacerse antes del inicio del juego y deben cumplir con las normas de competencia.

- **Descanso:**

Los equipos tienen derecho a un descanso en el intervalo entre las dos mitades del tiempo extra. El plazo de tiempo no debe exceder de 2 minutos.

La duración del descanso en dicho intervalo de tiempo puede ser modificado únicamente con el consentimiento de ambos equipos y el árbitro.

- **Tiempos de espera:**

Cada equipo tiene asignado dos tiempos de espera en el comienzo del tiempo extra. Se permite un total de 5 minutos para todos los tiempos de espera. El número de tiempos de espera y el tiempo, no utilizados en el juego regular, no se agregan. Los tiempos de espera en el tiempo extra siguen las mismas reglas que en el juego regular (véase la "Ley 7").

- **Tanda de penaltis:**

Si el partido termina en empate después del final de la segunda parte de la prórroga, el resultado final se decidirá en los penaltis.

- **Preparación:**

Antes del inicio de los penaltis, habrá un intervalo que no deberá exceder de 2 minutos. Este tiempo se designa para ser utilizado por los equipos en el diálogo con el árbitro y sus asistentes para comprobar que la posición del portero es correcta (en la línea) y que todas las demás normas se cumplen (véase "Ley 14"). El árbitro determina qué equipo tiene que defender la portería, así como qué equipo tiene que lanzar el primer penalti.

- **Procedimiento:**

Durante los tiros desde el punto de penalti, un máximo de 2 robots por equipo estarán en el campo con el fin de evitar interferencias. Los tiros desde el punto penalti se harán alternativamente por parte de ambos equipos hasta que cada equipo haya lanzado cinco disparos. Si se toma una decisión para un equipo, los lanzamientos se interrumpirán por decisión del árbitro. Para todos los lanzamientos, se aplican las normas de la "Ley 14". Un segundo tiro (por ejemplo, si la pelota rebota en la portería o un poste de la portería o el robot que lanza recuperase la pelota) no puntuará; ya que el penalti no será válido si el lazador vuelve a tocar la pelota después del primer disparo. Durante los lanzamientos, desde el punto penalti no habrá tiempos muertos. Los robots pueden ser intercambiados entre los lanzamientos siguiendo las reglas de intercambio de la "Ley 3". Como el intercambio de los campos entre ambos equipos costaría demasiado tiempo y se forzaría a los equipos a variar sus sistemas, se usarán ambas porterías.

Si después de 10 tiros no hay un vencedor, cada equipo tiene un lanzamiento de penalti en el mismo orden en que lo hicieran anteriormente. Este procedimiento (un penalti por equipo) se continúa hasta que haya un vencedor.

19. Apéndice B – Expertos en Visión

Durante las competiciones, los expertos en visión están a cargo del sistema compartido de visión de cada campo. La asignación y el tiempo del período de servicio son designados por los organizadores de la competición. Esto se debe realizar de tal forma que cada sistema de visión compartido tenga asignado, al menos, un experto en visión.

- **Deberes:**

  El experto en visión tiene los siguientes deberes:

  - Comprobar el hardware del sistema compartido de visión e informar de cualquier problema relacionado al árbitro y a los organizadores locales.
  - Hacer el proceso de calibración del SSL-Vision [11] cuando sea necesario o los equipos lo requieran durante los tiempos de configuración.
  - Calibrar o realizar el mantenimiento durante el partido del SSL-Vision [11] cuando el árbitro lo requiera.
  - Antes de cada partido, comprobar que ambos equipos reciben los paquetes del SSL-Vision [11] correctamente.
  - Antes de cada partido, comprobar que ambos equipos utilizan los correctos patrones estandarizados, que la altura de sus robots está calibrada con exactitud y que los datos de localización recibidos son correctos.

- Vigilar el estado del sistema compartido de visión durante el partido y reportar inmediatamente cualquier tipo de problema al árbitro.

- Recibir las quejas de los equipos sobre el sistema de visión compartido durante el partido y, si fuera necesario, preguntar al árbitro para parar el juego de tal forma que se pueda diagnosticar y solucionar el problema.

- Avisar al árbitro si hay alguna queja no solucionable de algún equipo acerca del sistema de visión. En este caso, el árbitro, tiene la autoridad definitiva para fallar en cualquier modo con respecto a sus poderes y deberes (véase "Ley 5"), incluyendo la habilidad para avisar y/o sancionar a los equipos de mal comportamiento si las exigencias de los equipos son infundadas y continúan obstruyendo el juego (véase Sanciones Disciplinares en "Ley 12").

ANEXO VI: CAMBIO REGLAMENTO 2012

ANEXO VI: Cambio en el Reglamento 2012

1. Cambio en el Reglamento 2012

A largo de este punto se comentará el cambio que se ha sufrido en el reglamento durante el año 2012 en la liga SSL [17] de la competición RoboCup [17]. Estos cambios han sido comentados y consensuados con los equipos participantes, que han aportado su punto de vista.

El presente PFC se ha basado en la competición de la SSL [9] de la RoboCup [17] del año 2011, por lo que las reglas que se han seguido han sido las de este año. Sin embargo, en el año 2012 se han incluido algunas modificaciones y mejoras que la organización de la competición ha considerado positivas.

Los cambios introducidos este año presente en el reglamento de la RoboCup [17] SSL [9] son:

	2011	**2012**
Número de robots por equipo	5	6
Radio del área	500 mm	800 mm
Distancia entre portería y borde	750 mm	450 mm
Lanzamiento indirecto	Si un robot dispara la pelota con una velocidad superior a 10 m/s	Si un robot dispara la pelota con una velocidad superior a 8 m/s

Tabla 54: modificaciones reglas RoboCup [17] 2012 respecto a las reglas RoboCup [17] 2011

Además, se han añadido otras:

- Se añade a la concesión de tiro libre directo la siguiente norma: "Se concederá tiro directo si se produce un vuelco, una rotura o la pérdida de una pieza del robot de manera que dé ventaja de forma injusta al equipo".

- En la "Ley 12" se ha añadido una novena decisión que dice: "Si un robot vuelca o se rompe por algún motivo y este no constituye ningún peligro para otro robot o ser humano, el árbitro permitirá que el juego continúe hasta que se produzca otra introducción en el juego. La decisión final sobre lo que constituye peligro o ventaja injusta recae sobre el árbitro".

- En el apéndice A se ha incluido el apartado de criterio de clasificación Round-Robin, según este criterio la clasificación de cada equipo en cada grupo será determinada por los siguientes criterios:

    - Mayor número de puntos obtenidos en todos los partidos de grupo.

    - Diferencia de goles en todos los partidos de grupo.

    - Mayor número de goles marcados en los partidos de grupo.