INVESTIGACIÓN EN FÚTBOL ROBÓTICO SOBRE PLATAFORMA LEGO MINDSTORM

TESIS QUE PRESENTA:

DIEGO MAURICIO GUTIERREZ RAMIREZ
JUAN PABLO OCAMPO RAMIREZ
ELVIN VALDES AVILA

INGENIERO EDGAR AGUIRRE
DIRECTOR DE PROYECTO

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS
FACULTAD DE INGENIERÍA
TECNOLOGÍA EN ELECTRÓNICA
BOGOTÁ
2013

HOJA DE ACEPTACIÓN

Observaciones

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

_

_____

Director del Proyecto

_____

Vo.Bo. Revisor

Fecha presentación

INTRODUCCIÓN

Uno de los mayores retos en robótica, es la acción coordinada de múltiples robots o agentes, rama de estudio conocida como Sistemas Multiagente. Este estudio se facilita al simular con los agentes una actividad humana con características de cooperación e interacción entre varios participantes. Una actividad que cumple satisfactoriamente estas características y de reconocimiento mundial, es el fútbol.

En robótica, los sistemas multiagente, son sistemas que se componen de varios agentes inteligentes (Robots), que interactúan entre ellos de manera lógica y coordinada. De esta manera, es posible resolver problemas que serían imposibles de solucionar por un solo agente.

Siendo objetivo de esta investigación, enfocarse en el fútbol robótico, como método de solución e implementación de un sistema multiagente, se enfocan estos sistemas en el modelado de estructuras sociales específicamente, sin embargo, esto no excluye sus resultados y aplicabilidad a otros enfoques de estos sistemas como son el comercio online o la respuesta a desastres de diferentes tipos en la comunidad.

Debido a la complejidad de interacción social, que implica el fútbol, es un excelente ejercicio a implementar en la robótica, para el análisis de comportamiento en las diferentes arquitecturas que se pueden implementar para dichos sistemas, ya que este deporte exige una combinación entre autonomía, igualdad y jerarquía, abriendo así una amplia perspectiva en la aplicación de estos sistemas.

Contenido

CAPÍTULO 1

¿QUE ES ROBÓTICA?

Citando a Craig, en su libro *Introducción a la Robótica. 3ed.:* “En la robótica, la conexión entre el campo de estudio y nosotros mismos es inusualmente obvia. Y, a diferencia de una ciencia que solamente analiza, el objetivo actual de la robótica requiere que la ingeniería se desvíe hacia la síntesis.”

“La robótica se relaciona en sí con el deseo de sintetizar algunos aspectos de la función humana mediante el uso de mecanismos, sensores, actuadores y computadoras, lo que representa un enorme compromiso y requiere una multitud de ideas provenientes de varios campos clásicos.”

Karel Capek (1890 - 1938) en 1921 fue el creador de la palabra robot, para una obra de teatro. Afirmaba su origen es de la palabra eslava robota, la cual significa trabajo realizado de una manera forzada.

La mayoría de los expertos en Robótica afirman que es complicado dar una definición universal de Robot. Algunos se atreven a argumentar que, el robot es Ingenio mecánico controlado electrónicamente, capaz de moverse y ejecutar de forma automática acciones diversas, siguiendo un programa establecido.

Se han planteado diferentes definiciones de robótica entre las cuales se encuentran las siguientes [1]:

- Ingenio mecánico controlado electrónicamente, capaz de moverse y ejecutar de forma automática acciones diversas, siguiendo un programa establecido.
- Máquina que en apariencia o comportamiento imita a las personas o a sus acciones como, por ejemplo, en el movimiento de sus extremidades
- Un robot es una máquina que hace algo automáticamente en respuesta a su entorno.
- Un robot es un puñado de motores controlados por un programa de ordenador.
- Un robot es un ordenador con músculos.

Teniendo en cuenta las diferentes definiciones, se afirmar que la robótica, es el vínculo máquina-humano que facilita la ejecución de acciones solicitadas por el hombre de forma inteligente.

Realmente dar una definición exacta de robótica es muy difícil. Pero, para este caso no es necesario que todo el mundo esté de acuerdo con la definición de robótica. Tal vez Joseph Engelberg (padre de la robótica industrial) lo resumió inmejorablemente cuando dijo: "Puede que no se capaz de definirlo, pero sé cuándo veo uno". [1]

Clases de robots

Se puede tomar de una forma muy general la clasificación de Robots, como de la siguiente forma: [1]

Androide

La palabra androide es de origen griego, donde “andros” significa hombre y “eidos” se traduce como apariencia. Esta palabra se le asigna a un artefacto robótico que imita la figura humana masculina. (Para el caso de las mujeres el término correcto sería llamarlos ginoides) el androide también trata de incorporar ciertas actitudes propias de los humanos, según la programación previa, por lo cual no deciden sus conductas.

La construcción de androides, que imitan algunos comportamientos humanos, está dada por la ingeniería. Un claro ejemplo de un androide, como los tomados de ciencia ficción en el año 2000 fue creado por la empresa japonesa Honda, ASIMO un humanoide capaz no solo de caminar sino de correr y subir y bajar escaleras, que ha ido perfeccionándose para responder a instrucciones, identificar objetos, tomarlos, girar, entre otros movimientos.

Móviles

Los robots móviles son mecanismos capaces de actuar de forma autónoma a la hora de resolver un problema.

Hay diferentes tipos de robots móviles, que se comprenden en: vehículos con ruedas, caminantes, humanoides, submarinos, aéreos y espaciales.

El esquema general de un sistema robot se resume en lo siguiente: [1]

- Sensores externos que captan una percepción del entorno: Visión, tacto, audición, proximidad, etc.
- Sensores internos que miden el estado de la estructura mecánica: giros, desplazamientos, velocidades, etc.

- Actuadores: Sistemas electromecánicos que generan fuerza y par para el movimiento: Sistemas eléctricos, mecánicos o neumáticos.
- Sistemas de control que aseguren el funcionamiento correcto de los movimientos (bucles de control), planificación (trayectorias), etc.

Cabe hacer un énfasis en los vehículos con ruedas, ya que a la hora de trabajar sobre terrenos duros y llanos, facilitan la movilidad de un mecanismo robótico, aumentando la velocidad y capacidad de reacción según la situación en que se encuentre el robot. Entre las posibilidades de diseñar un sistema con ruedas están:

Configuración Ackerman

Configuración habituada en los vehículos convencionales, y por lo tanto es una configuración muy probada y estable. Se basa en una estructura de cuatro ruedas colocadas en dos ejes, donde sólo las dos ruedas delanteras permiten un giro sobre el eje. [2]

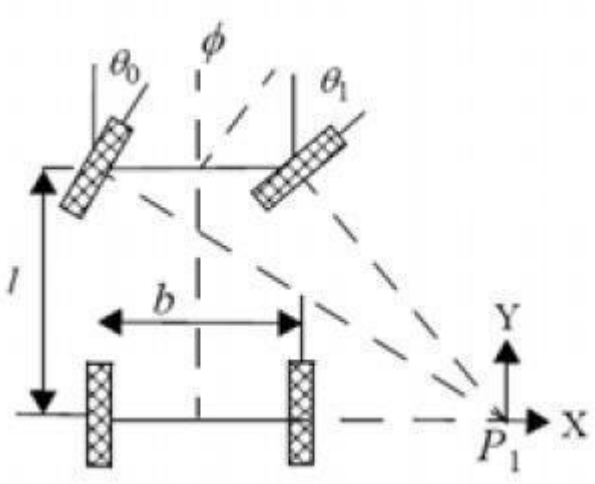


Figura 1.1. Plataforma convencional

Configuración triciclo

Posee un gran parecido a la configuración anterior pero aporta una mayor simplicidad en la construcción, ya que sólo son necesarias 3 ruedas. No obstante, aporta menos estabilidad al sistema. [2]

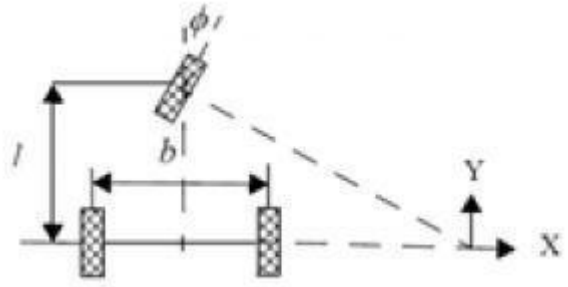


Figura 1.2. Plataforma diferencial

Pistas de deslizamiento

Son vehículos tipo oruga en los que tanto la impulsión como el direccionamiento se consiguen mediante pistas de deslizamiento. Al realizar un giro se aumenta la velocidad de una de las pistas para producir el giro en el vehículo. Pueden considerarse funcionalmente análogas al skid steer (cargadora compacta). Las pistas actúan de forma análoga a ruedas de gran diámetro. La locomoción mediante pistas de deslizamiento es útil en navegación o en terrenos irregulares, en los cuales presenta un buen rendimiento. En este caso, la impulsión está menos limitada por el deslizamiento y la resistencia al desgaste es mayor. [2]

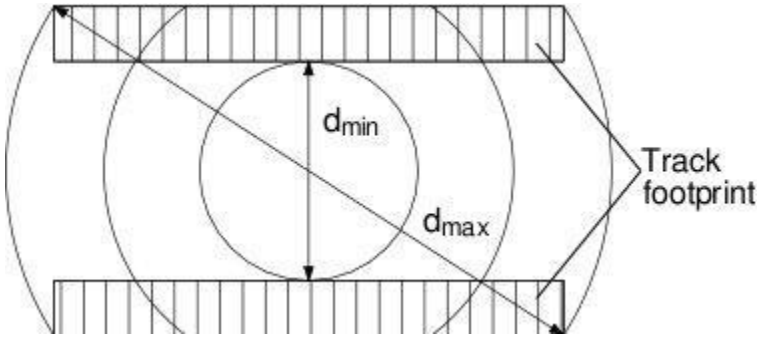


Figura 1.3. Vehículo tipo oruga

Industriales

Para definir un robot industrial se debe tener en cuenta el concepto japonés y euroamericano de lo que es un robot y un manipulador. Porque, mientras para los japoneses un robot industrial es cualquier dispositivo mecánico dotado de articulaciones móviles destinado a la manipulación, el mercado occidental es más restrictivo, exigiendo una mayor complejidad, sobre todo en lo relativo al control.[1]

La definición más comúnmente aceptada es la de la Asociación de Industrias Robóticas (RIA), que dice: “Un robot industrial es un manipulador multifuncional reprogramable, capaz de mover materias, piezas, herramientas, o dispositivos especiales, según trayectorias variables, programadas para realizar diversas tareas”

Esta definición, ligeramente modificada, ha sido adoptada por la Organización Internacional de Estándares (ISO) que define al robot industrial como:

“Manipulador multifuncional reprogramable con varios grados de libertad, capaz de manipular materias, piezas, herramientas o dispositivos especiales según trayectorias variables programadas para realizar tareas diversas.”

A pesar de las definiciones el robot industrial tiene también una clasificación, que esta desde manipuladores, pasando por inteligentes, controlados etc. Por ejemplo:

- Manipuladores: Sistemas mecánicos multifuncionales, con un sencillo sistema de control, que permite gobernar el movimiento de sus elementos, de los siguientes modos:

    - Manual: Cuando el operario controla directamente la tarea del manipulador.
    - De secuencia fija: cuando se repite, de forma invariable, el proceso de trabajo preparado previamente.
    - De secuencia variable: Se pueden alterar algunas características de los ciclos de trabajo.

- De repetición y aprendizaje: Estos manipuladores se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. En este tipo de robots, el operario en la fase de enseñanza, se vale de una pistola de programación con diversos pulsadores o teclas, o bien, de joystick, o a veces, desplaza directamente la mano del robot.

- Robots por control de computador: Sistemas mecánicos multifuncionales, controlados por un computador, que habitualmente suele ser un microordenador. El control por computador dispone de un lenguaje específico, compuesto por varias instrucciones adaptadas al robot, con las que se puede confeccionar un programa de aplicación utilizando solo el terminal del computador, no el brazo. A esta programación se le denomina textual y se crea sin la intervención del manipulador.

- Robots inteligentes: Estos robots son programados por computadora, pero antes de su ejecución en la industria además, son capaces de relacionarse con el mundo que les rodea a través de sensores y tomar decisiones en tiempo real (auto programable).

La clasificación anterior ya existe, aunque los casos más futuristas están en estado de desarrollo en los centros de investigación de robótica. Por ende se presenta la siguiente clasificación. [1]

	Clasificación de los robots según la AFRI
Tipo A	Manipulador con control manual o telemando.
Tipo B	Manipulador automático con ciclos preajustados; regulación mediante fines de carrera o topes; control por PLC; accionamiento neumático, eléctrico o hidráulico.
Tipo C	Robot programable con trayectoria continua o punto a punto. Carece de conocimiento sobre su entorno.
Tipo D	Robot capaz de adquirir datos de su entorno, readaptando su tarea en función de estos.

Tabla 1. (AFRI) Asociación Francesa de Robótica Industrial. [1]

Tomado del Dr. en Física, Ingeniero Téc. De Telecomunicaciones Víctor R. González Fernandez de la página web del Centro de Formación del Profesorado e Innovación Educativa de Valladolid,

Médicos

Son, fundamentalmente, prótesis para disminuidos físicos que se adaptan al cuerpo y están dotados de potentes sistemas de mando. Con ellos se logra igualar con precisión los movimientos y funciones de los órganos o extremidades que suplen. [1]

Teleoperadores

De esta clasificación de robots se puede decir que no son exactamente robots, ya que realmente se controlan remotamente por un operador humano, pero al obtener mecanismos y circuitos electrónicos, además de ser útiles y sofisticados, se les considera como telerobots. [1]

Poliarticulados

Bajo este grupo están los robots de muy diversa forma y configuración cuya característica común es la de ser básicamente sedentarios, aunque excepcionalmente pueden ser guiados para efectuar desplazamientos limitados y estar estructurados para mover sus elementos terminales en un determinado espacio de trabajo según uno o más sistemas de coordenadas y con un número limitado de grados de libertad. [1]

Zoomórficos

Estos podrían entrar en la clasificación de los androides, ya que se caracterizan por sus sistemas de locomoción que asemejan a diversos seres vivos. [1]

A pesar de la disparidad morfológica de sus posibles sistemas de locomoción es conveniente agrupar a los robots zoomórficos en dos categorías principales: caminadores y no caminadores. El grupo de los robots zoomórficos no caminadores está muy poco evolucionado. Cabe destacar, entre otros, los experimentados efectuados en Japón basados en segmentos cilíndricos biselados acoplados axialmente entre sí y dotados de un movimiento relativo de rotación. En cambio, los robots zoomórficos caminadores multípedos son muy numerosos y están siendo experimentados en diversos laboratorios con vistas al desarrollo posterior de verdaderos vehículos terrenos, pilotados o autónomos, capaces de evolucionar en superficies muy accidentadas. [1]

Híbridos

Estos robots corresponden a aquellos de difícil clasificación cuya estructura se sitúa en combinación con alguna de las anteriores ya expuestas, bien sea por conjunción o por yuxtaposición. De igual forma pueden considerarse híbridos algunos robots formados por la yuxtaposición de un cuerpo formado por un carro móvil y de un brazo semejante al de los robots industriales. En parecida situación se encuentran algunos robots antropomorfos y que

no pueden clasificarse ni como móviles ni como androides, tal es el caso de los robots personales. [1]

¿QUE ES UN AGENTE?

Un agente puede ser un ser humano un animal, un software, o en el caso que nos interesa un robot móvil el cual es un sistema situado dentro y como parte de un entorno, que siente y nota ese entorno y actúa en él, en el tiempo, para cumplir su propia agenda o metas, siendo capaz de apreciar los resultados obtenidos y de volver a actuar tomándose en consideración. [3]

Características de un agente

Un agente básicamente posee tres características como en la figura 3, una percepción, donde recibe datos de un grupo de sensores, otra que es una caja negra donde define el comportamiento y decide, y por último una actuación donde ejecuta las acciones, todo esto dentro de un entorno o ambiente físico.

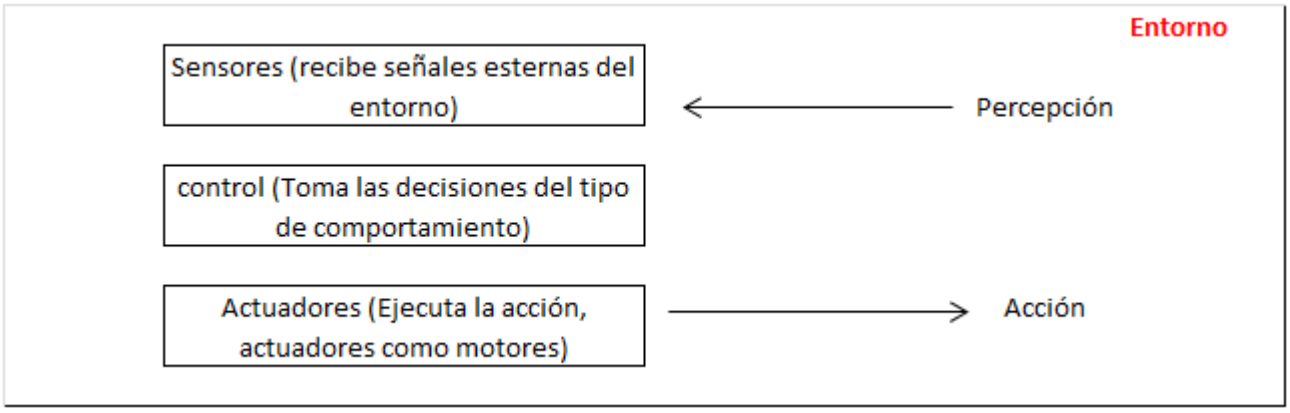


Figura 1.4. Características de un agente. [3]

Tipos de Agente

En esencia se definen dos tipos de agentes, autónomos y dependientes, donde los agentes autónomos son capaces de tomar sus propias decisiones y los dependientes dependen de otros agentes o reglas para decidir, Los tipos de agentes dependen de las aplicaciones que deseemos realizar, aplicaciones como recoger elementos, identificar patrones, realizar formaciones, también estos agentes los identificamos por su capacidad de cooperación, por su autonomía y su movilidad, así mismo puede funcionar, por el tipo de arquitectura que puede ser deliberativa o reactiva y por el tipo de organización que en el caso que trabajaremos es cooperativa, donde entra a jugar la cooperación entre agentes y la autonomía de cada uno, esto supone la capacidad de resolver problemas en conjunto y de cómo cada agente cumple un tarea diferente y coopera con otro resolviendo tareas individuales que ayudan al problema en conjunto, esto ya nos da un grupo de

características, que indican el papel o rol de cada entidad que participa esto indica que un solo móvil no realiza todas las tareas, quiere decir que manejamos una arquitectura descentralizada y cada agente tiene un objetivo. [3]

¿QUE ES UN SISTEMA MULTIAGENTE?

Se puede definir sistema multiagente (MAS) como la interacción de varios agentes para realizar una tarea, dependiendo el tipo de tarea crece la complejidad. Básicamente se puede definir como un modelo general de un sistema multiagente así: [3]

Modelo de agente

Los agentes realizan tareas o persiguen objetivos, como responsabilidades, control y estado mental del agente. Figura 1.3

Modelo de objetivos y tareas

Identificación de objetivos generales y descomposición en objetivos más concretos que se pueden asignar a agentes.

Modelo de interacción

Qué interacciones existen entre agentes/roles, comunicación.

Modelo de organización

Estructura del sistema multiagente, roles, relaciones de poder.

Modelo de entorno

Entidades y relaciones con el entorno del Sistema Multiagente.

Dentro de estos modelos se pueden trabajar agentes autónomos o dependientes al compartir información entre ellos o solamente como entes autónomos en la toma de decisión.

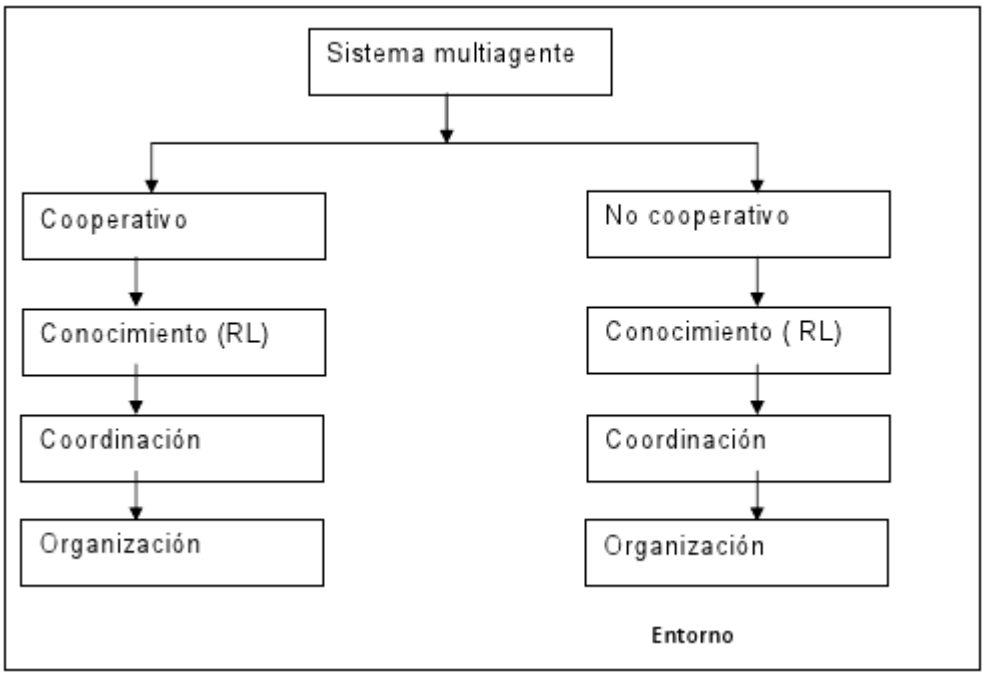


Figura 1.5 Modelo Sistema Multiagente [3]

Un sistema multiagente puede ser cooperativo o no puede existir una coordinación para realizar las tareas, pero también no puede haber comunicación ni coordinación entre los agentes, y aun así sigue siendo un sistema multiagente.

CAPÍTULO 2

ESTUDIO DE ROLES DE FÚTBOL ROBÓTICO

2.1 Caracterización de acciones de un Arquero

2.1.1 El Arquero Humano.

El arquero es pieza fundamental para la defensa del equipo, el cual debe poseer gran capacidad para impedir goles, función que, dependiendo la eficiencia de este, influye directamente sobre la estrategia a tomar a partir de este momento como también sobre el estado anímico del equipo.

Este también puede asumir actitudes ofensivas como punto inicial del ataque al ser quien toma la decisión de la forma en que la pelota volverá a estar en juego después de que esta ha abandonado el campo de juego por una de sus líneas finales.

“La exigencia mental de todo jugador culmina frente al arco, lo cual presupone que el arquero tenga características muy especiales precisamente para enfrentar la situación. [4]

2.1.2 Área de influencia

El arquero se desenvuelve en su mayor parte dentro del área de gol como se muestra en la imagen:

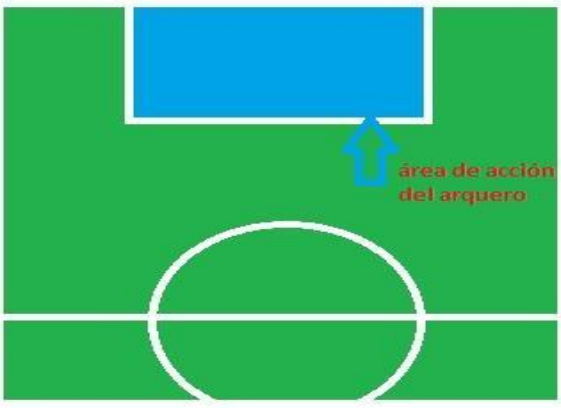


Figura 2.1 Área de acción del arquero

La zona azul de la imagen anterior muestra la zona de mayor interacción del arquero con el juego, ya que es en esta zona donde él puede utilizar sus manos que son en muchas ocasiones las que mayor contacto con el balón tienen en lugar de sus piernas como los demás jugadores. La importancia de demarcar desde ahora la zona de acción del portero radica en delimitar el área de acción de nuestro agente robótico y con ello limitar sus funciones a las más importantes para medir su desempeño.

2.2 Características Técnico-Tácticas:

Entre las muchas características técnico-tácticas que debe poseer un buen arquero, se hacen notables su capacidad de reacción para interceptar la pelota cuando esta lleva trayecto a gol y se encuentra dentro de su área de acción. [4]

El despeje del balón es importante en una acción "*uno a uno"* con el delantero, ya que con esto expulsa de la zona de peligro el balón, disminuyendo la posibilidad de gol en contra.

La ubicación en un momento sin riesgo de gol (Estado de reposo), debe ser cercana a la línea de gol bajo el arco, con el fin de seguir desde allí el desarrollo del juego y así poder reducir el ángulo de tiro y disminuir las probabilidades de gol del oponente. Esto implica que debe tener una gran capacidad de decisión (Control tiempo espacio) para saber cuándo alejarse de la línea de gol para "*achicar"* la distancia entre el delantero y él, para cerrar el espacio en el que el balón puede dirigirse a gol.

2.3 Variables a evaluar en un Arquero en Fútbol Robótico

Para evaluar el desempeño del agente en una posición y/o función determinada, se hace necesario definir ahora una escala de medición del éxito en el desempeño del robot en su posición correspondiente, para esto se deben identificar y definir variables cuantitativas que describan en su conjunto el "comportamiento" que se espera tenga el agente al momento de ejecutar su rol.

2.3.1 Identificación de variables

Con base en el análisis realizado en la sección 3.2, podemos abstraer 3 características medibles de importancia para evaluar el desempeño del robot en posición de arquero, a continuación se mencionan 3 aspectos fundamentales para el buen desempeño del agente:

-Orientación Espacial (O):

Esta variable es de mucha importancia, ya que representa la capacidad que tiene el robot de retornar a su posición original en "*Estado de reposo"* después de haberse desplazado por el campo de juego para tomar parte del mismo en un momento dado. Para nuestro caso, la orientación se identificará en adelante con la letra O.

La posición de reposo en la cual el agente estará vigilante de la dinámica del juego se representa en la siguiente figura:

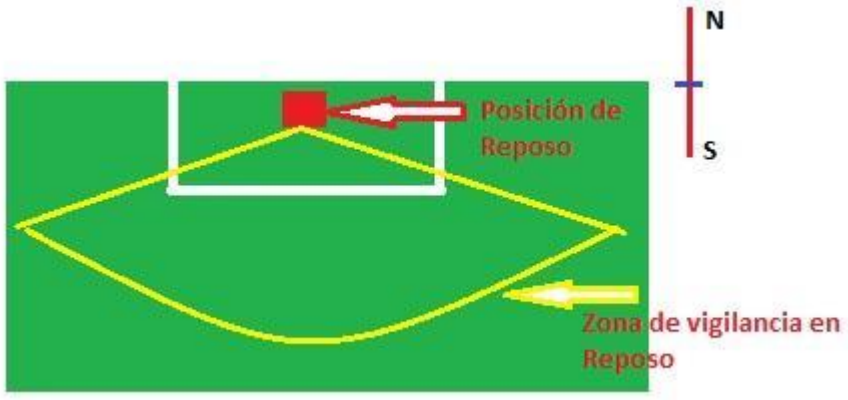


Figura 2.2. Posición de reposo del arquero

Se observa en la figura que se hace necesario definir una orientación previa de la cancha con el fin de poder indicar al agente cuál será la orientación correcta que debe asumir para estar vigilante ante una posible acción de gol. Para que esta medición sea posible, suponga una circunferencia con centro en el origen del plano polar, en la cual se representan los 4 puntos cardinales mediante el barrido de un vector r de radio constante de los 360° de dicha circunferencia, es decir:

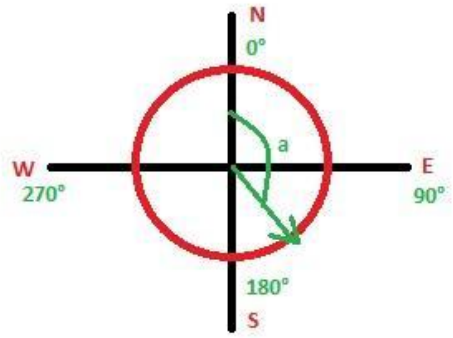


Figura 2.3. Orientación del arquero

Donde a es el ángulo barrido por el vector mencionado anteriormente, de lo que se puede deducir que:

Punto Cardinal	*Ángulo Equivalente "a" (Deg)*
Norte	*0*
Este	*90*
Sur	*180*
Oeste	*270*

Tabla 2. Orientación arquero

Teniendo ya definida esta convención o equivalencia, es posible medir la efectividad con que el agente se orienta en el campo de juego y la capacidad de retornar a su punto de reposo correspondiente, por lo tanto, nuestro arquero debe mantener su orientación mientras esté en reposo en un valor de 180. Por lo tanto se tiene:

$$O = 180 \qquad (1)$$

Donde (1) se cumple siempre y cuando el agente se encuentre en estado de reposo y vigilancia como se mencionó con anterioridad.

Estas representaciones y deducciones tendrán una importante utilidad en capítulos posteriores al enfrentar nuestros supuestos con la lógica utilizada en la plataforma a programar.

-Reducción de espacio (Achique):

La reducción de espacios o achique es la capacidad del robot para identificar que la pelota se aproxima conducida por un oponente con intención de gol. Un buen achique debe ser el que disminuye en mayor medida la posibilidad de gol mediante el "ahogo" del balón impidiendo que este sea disparado en una trayectoria que pueda tornarse imposible de seguir para el arquero.

De lo anterior se deduce que la distancia entre la pelota y el robot es un indicador de cuando el robot debe empezar a avanzar hacia la pelota, dentro de sus límites de acción, para *"achicar"* el espacio de disparo para el delantero, representado gráficamente:

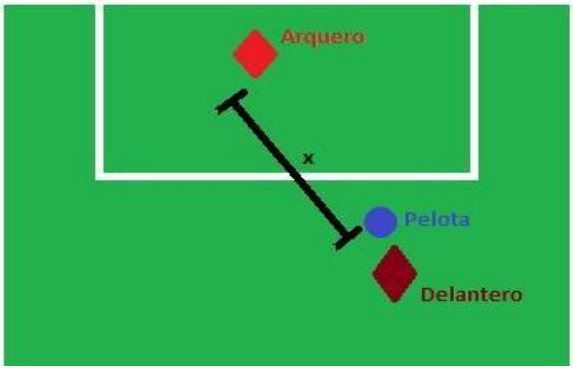


Figura 2.4 Reducción de espacio del arquero

Donde "x" es la variable que controla la distancia entre el balón y el arquero, la cual es determinante para la reacción de achique; por lo tanto se debe determinar un límite inferior para esta variable, en el cual se "active" el agente para salir al achique.

-Despeje de balón:

Es la variable que mide la efectividad en el despeje del balón por parte del arquero en una situación de riesgo, esta variable evalúa procesos como son:

- Efectividad de despeje. Entendiéndose esta como un despeje seguro, que no entregue el balón a su adversario
- Rapidez: Velocidad con que el arquero es capaz de despejar el balón.

Para el primer ítem se debe tener en cuenta la dirección de la cual proviene la pelota disparada por el delantero, determinando así la dirección final en la cual se lanzará el balón, lo anterior se ilustra de la siguiente forma:

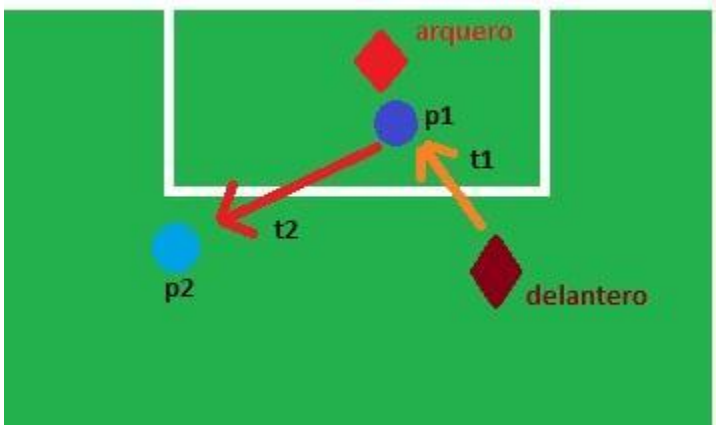


Figura 2.5 Despeje del balón del arquero

Donde t1 es la trayectoria aplicada por el disparo del delantero desde su posición, p1 es la posición de la pelota al momento de ser interceptada por el portero, t2 es la trayectoria a aplicar por el arquero evitando que la pelota vuelva a quedar en posesión del delantero contrario y finalmente p2 es una *posición que está sobre t2*; esto indica que p2 no está definida en coordenadas fijas pero sí en dirección. Es evidente de esto que: $t2 \neq t1$ tanto en dirección como en sentido.

El segundo Ítem depende más de las características mecánicas del robot para imprimir la fuerza y la velocidad necesaria de reacción antes de ser atacado por el delantero oponente.

Finalizando este análisis podemos tabular nuestras variables objetivo para su uso posterior:

Variable	Símbolo
Orientación Espacial	O
Reducción de espacio	x
Despeje de balón	t2

Tabla 3 Símbolos

Las variables definidas en este capítulo se deben implementar de acuerdo a la plataforma a usar y de acuerdo a las especificaciones técnicas de la misma y sus diferentes componentes.

2.4 Caracterización de acciones de un defensa

2.4.1 El Defensa Humano

El jugador en la posición de defensa es aquel que desarrolla su rol en su propia área, cubriendo el sector, ya sea central o lateral, oponiéndose a la entrada de los rivales o la pelota, a parte de su posición (central o lateral), existen dos tipos de defensa: los que marcan a los rivales y los que se encargan de marcar el balón utilizando la anticipación y la ubicación. El ideal es que se usen los dos tipos pero pocos son capaces de lograrlo eficazmente, por esto surgen los términos como stopper y liberó especializándose cada uno en un estilo defensivo, pero generalmente hablando de un defensa se pide que tenga un gran sentido de la ubicación y la anticipación, además de la facilidad de control de la pelota y un buen rechazo de balón. [4]

2.4.2 Zona de Juego

El Defensa se desenvuelve en su mayor parte dentro de su misma área como se muestra en la imagen:

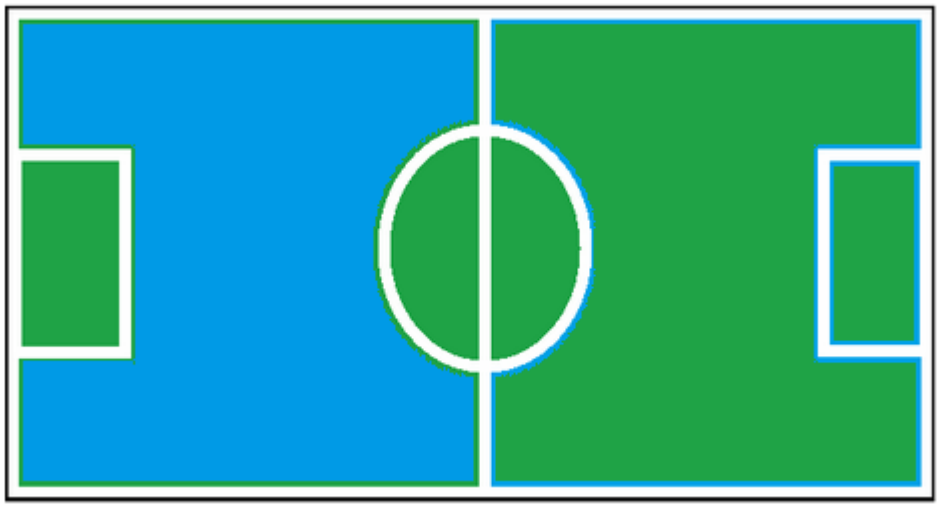


Figura 2.6 zona de juego del defensa

La zona pintada de azul de la imagen anterior muestra la zona en la que interactúa el defensa en el juego, que en muchas ocasiones es donde transcurre la mayor parte de las acciones de un enfrentamiento, la importancia de delimitar esta zona es para asumir como el campo de acción de nuestro agente robótico y con ello limitar sus funciones a las más importantes para medir su desempeño.

2.5 Características Técnico-Tácticas

Entre las características más notables que debe poseer un buen defensa líbero para nuestro caso es como su nombre lo indica libre de defensa, que debe ir o anticipar el balón, debe ser indispensable en el la ubicación y la anticipación, acompañado de una visión de campo intuitiva, que le permita anticipar los movimientos del oponente, con buen control de balón y precisión en sus pases, y sorpresivo físicamente para sumarse a un ataque cuando se necesario. En donde sus principales cualidades técnico-tácticas serían la marcación y recuperación de la pelota y el remate y/o pase de despeje. [4]

2.6 Variables a Evaluar en un Defensa Robótico

Para evaluar el desempeño de un agente en posición determinada, se hace necesario definir la escala de medición de éxito en el desempeño del agente robótico en su rol determinado, para esto se deben identificar y definir las variables cuantitativas que describen los ¨comportamientos¨ que se espera que tenga el agente al ejecutar su rol.

2.6.1 Identificación de las variables

Con base a lo analizado en la sección 4.2 podemos tomar 3 características medibles de importancia a evaluar en el desempeño del robot en la posición de defensa, a continuación se mencionan los aspectos fundamentales para un buen desempeño:

-Orientación Espacial (O):

Esta variable es la más importante ya que es la que determina la posición inicial del agente después de una acción defensiva, con el fin de retornar a una ubicación apropiada para realizar de nuevo sus acciones determinadas.

La posición indicada para un agente defensivo se representa en la siguiente figura:

Figura 2.7 Zona de vigilancia del defensa

Se puede observar que se necesita la orientación del campo de juego para indicarle cuál será su posición inicial o cuando se encuentra en reposo, para esta tarea representaremos gráficamente una brújula con sus puntos cardinales de la siguiente manera:

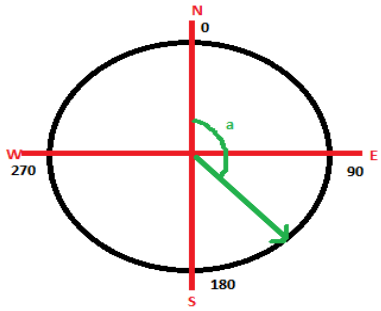


Figura 2.8 Orientación del defensa

En donde a es el ángulo recorrido por el vector con lo que podemos deducir el ángulo correspondiente a cada punto cardinal de la siguiente manera: Norte =0; Este=90: Sur=180; y Oeste=270.

Teniendo en cuenta estos datos ya es posible tener una escala de control de efectividad de ubicación del agente en el campo de juego, y la capacidad de retomar su punto inicial. Para el cual tomamos como referencia el sur, lo que quiere decir que el punto O de nuestro agente es igual a: O=180 esto será verdad si nuestro agente está en reposo sin balón.

Ubicación y demarcación del Balón

La ubicación y marcación del balón es la capacidad que tiene el agente ofensivo para identificar la posición de la pelota, cuando esta se encuentre en su campo de acción defensivo, negándolo o dificultando el oponente el acceso al área de gol o negando la posibilidad que la pelota entre en el pórtico.

De lo anterior se destaca que el límite de acción es la distancia a la cual se encuentra la pelota, descartando si esta viene dominada por un oponente o es por causa de un tiro al arco. Representado gráficamente sería

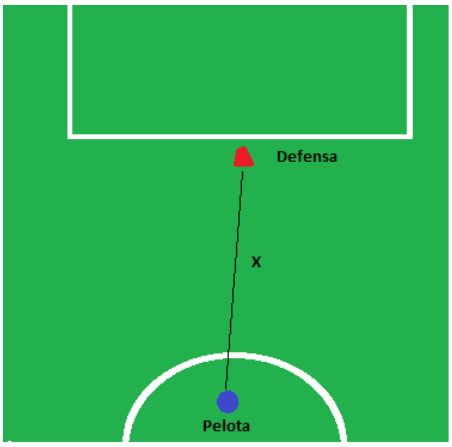


Figura 2.9. Orientación del defensa frente al balón de juego

En donde la distancia representada por la x es la variable de la distancia a la que se encuentra la pelota de nuestro agente defensivo, el cual es el que se encarga de que el agente presione al oponente en el caso de que se encuentre en este rango de acción.

Despeje de balón

Esta es la variable con la se mide la efectividad del despeje del el defensa en caso de una acción ofensiva del rival, en donde se evaluará la precisión, la efectividad y la rapidez de reacción en este proceso.

Para realizar un buen despeje es necesario saber la dirección en la que viene la pelota ,para despejar en un sentido y dirección diferente, en donde llamaremos a esta d2, que sería la trayectoria de despeje de la pelota cuando está en posesión del defensa en donde tendríamos las siguientes variables para un uso posterior.

Variable	Símbolo
Orientación Espacial	O
Ubicación y marcación del balón	x
Despeje de balón	d2

Tabla 4. Símbolos de las variables del defensa

2.7 Caracterización de acciones de un delantero

2.7.1 Delantero Humano

El delantero en el fútbol es la pieza fundamental del ataque, ya que se destaca por ser el atacante más cercano a la portería contraria, por ende cumple la labor de la marcación de goles. Además el delantero se podría decir que es el primer filtro de defensa de un equipo ya que este puede hacer presión al contrario desde su propia área. [4]

Una de las cualidades más fuertes de un delantero es la capacidad de desmarcarse del contrario para la perfecta ubicación a la hora de anotar goles. Es aquí donde el talento del jugador sobresale ya que resalta su calidad de definición, frente al portero rival.

2.7.2 Zona de juego o área de influencia

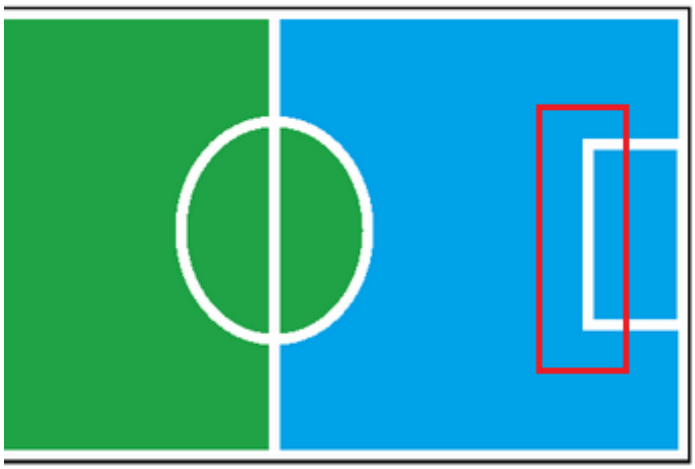


Figura 2.10 zona de juego del delantero

El delantero se ubica en la zona ofensiva del campo demarcada en la figura X con color azul.

Esta zona se define como zona de ataque donde el delantero frecuenta la zona demarcada en la figura 2.10 en color rojo. El delantero antes de recibir el balón fija su mirada hacia la zona contraria a la de anotar gol, ya que de esto depende la recepción del balón.

Al obtener el balón el delantero fija su mirada al arco rival, buscando lo más posible la zona de remate a gol.

El robot tiene en cuenta donde está ubicado gracias a dos factores; el principal que es una brújula magnética que indica el punto hacia donde está mirando el robot, en este caso se le indica al robot que su punto de ataque está orientado hacia el norte.

El otro factor de ubicación para el robot es el color; teniendo en cuenta que debe jugar solo en su zona verde y si él toca un color distinto busca reubicarse en el sector de ataque, espera un tiempo prudente para la detección del balón.

2.8 Características técnico-tácticas

El delantero para poder marcar goles debe contar con buen dominio del cuerpo, ser ágil, sobre todo en el dribleo, hábil en la toma de decisiones y de potente definición. El dominio de balón es fundamental, tanto en la recepción de pelotas altas y bajas como en la manejo del balón a través del campo. [4]

Importante también la compatibilidad con sus compañeros para la asociarse con ellos al momento de ejecutar acciones claves como ejecutar paredes, recepción de centros en carrera y obviamente la perfecta ubicación para rematar con fuerza y dirección.

Crear el espacio a sus compañeros y posibilitar sorpresivos cambios de dirección de pelota y cambios de frente en ataque son de mucha importancia táctica...

Las principales cualidades tácticas son:

a) marcar goles en jugada individual.

b) ubicación espacial

2.9 Variables a evaluar en un Delantero en Fútbol Robótico

2.9.1 Identificación de variables

Con base en el análisis realizado en la sección 5.2, podemos abstraer características medibles de importancia para evaluar el desempeño del robot en posición de arquero, a continuación se mencionan 2 aspectos fundamentales para el buen desempeño del agente:

-Marcar goles en jugada individual:

Es la variable que mide la efectividad en el tiro al arco por parte del delantero, teniendo en cuenta la obstaculización de los rivales para poder rematar a gol. Esta variable evalúa procesos como son:

- Efectividad de tiro. Entendiéndose esta como un tiro exitoso a gol, sin entregar el balón a su adversario.
- Evasión del rival: Se entiende por la acción de eliminar a un rival para obtener un espacio libre de obstáculos.

Para el primer ítem se analiza la ubicación del balón y del agente para que el disparo cumpla la trayectoria deseada:

Figura 2.11 Posiciones del delantero

En la imagen de la izquierda se puede analizar que la distancia (d) del delantero con respecto a la pelota (p1) es alejada y no está plenamente ubicada para un buen disparo a

gol y la posible trayectoria final (t2) es un tiro desviado, mientras que la imagen de la izquierda muestra que la pelota (p1) se encuentra plenamente ubicada para realizar el disparo. El delantero tiene en cuenta que la distancia (d) de él, con respecto a p1 sea menor a 3 cm y que p1 puede cumplir satisfactoriamente t2 para un óptimo control y manejo de tiro. Puede que el disparo no esté dirigido en coordenadas pero sí en dirección.

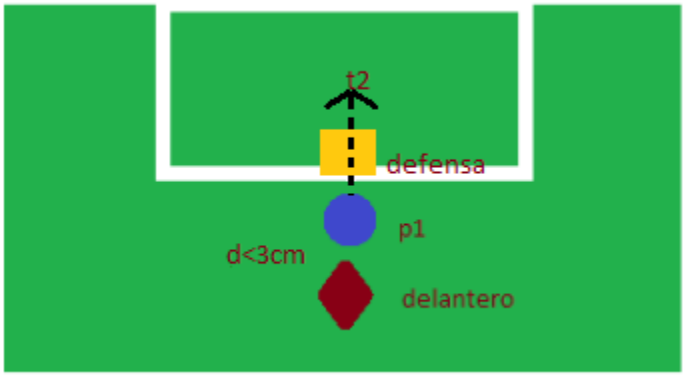


Figura 2.12 Posiciones del delantero y el defensa

El segundo ítem depende de la cantidad de obstáculos que influyan en la trayectoria de la pelota, ya que de esto depende una segunda evaluación, teniendo en cuenta las características mecánicas del robot para imprimir la fuerza y la velocidad necesaria como para poder evadir el rival antes de ser atacada por el defensa oponente.

El delantero conoce qué trayectoria debe seguir la pelota y al detectar una obstrucción a menos de 20 cm, sabe que t2 no se cumplirá y debe tomar un paso atrás para desatascar la pelota y cambiar de dirección para luego cumplir evaluar t2 y p1 y rematar al arco.

-*Orientación Espacial (O) de la zona de remate a gol:*

Esta variable representa la capacidad del robot de ubicarse mirando hacia el norte, en espera de un despeje del contrario para interceptar la pelota, se tendrá en cuenta la variable ya manejada con la letra O para aclarar la posición en la que se ubica el delantero.

El *Estado de reposo* del delantero es diferente ya que por la capacidad activa y de ataque del delantero es necesario el movimiento constante de este, entonces para este caso se

llamará *estado inicial*, que es la posición con la cual el jugador inicia apenas arranca el partido, de ahí en adelante el robot estará en constante rotación espacial.

La posición inicial en la cual el agente estará vigilante de la dinámica del juego se representa en la siguiente figura:

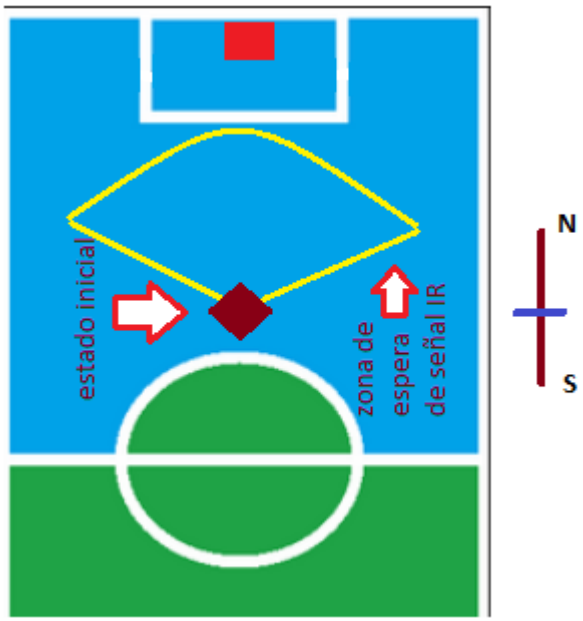


Figura 2.13 Orientación de la cancha

En la figura anterior se define una orientación de la cancha con el fin de indicarle al agente la ubicación correcta hacia el ataque.

Teniendo ya definida esta convención o equivalencia, es posible medir la efectividad con que el agente se orienta en el campo de juego, el delantero debe mantener su orientación mientras esté en estado inicial en un valor de 0. Por lo tanto se tiene:

$$O = 0 \quad (2)$$

Donde (2) se cumple siempre y cuando el agente se encuentre en estado de reposo y vigilancia como se mencionó con anterioridad.

Estas representaciones y deducciones tendrán una importante utilidad en capítulos posteriores al enfrentar nuestros supuestos con la lógica utilizada en la plataforma a programar.

CAPÍTULO 3.

PLATAFORMAS IMPLEMENTADAS

Fútbol Robótico sobre la plataforma LEGO MINDSTORMS NXT

3.1 La plataforma LEGO MINDSTORMS NXT

El Kit de desarrollo en robótica Lego Mindstorms NXT es un entorno programable enfocado en el aprendizaje de la robótica al alcance de todos ya que sus costos son menores a otros sistemas dirigidos a este campo de estudio. Esta plataforma está basada en la programación de acciones de forma gráfica a través del software de desarrollo provisto por la empresa, aunque también es compatible con labview, Robolab, RobotC, Matlab y Simulink, entre muchos otros. Para nuestro caso de estudio nos basamos estrictamente en el uso del software provisto por la compañía, el NXT - G. [5]

Esta plataforma cuenta con una unidad programable para desarrollo de acciones conocida como el Bloque inteligente NXT. LEGO MINDSTORMS permite crear modelos de sistemas integrados capaces de ser controlados por un computador. Son muchas las opciones pedagógicas y tecnológicas que presta el LEGO MINDSTORMS, a pesar de ser un juego de robótica para niños.

Lego permite realizar trabajos, en su gran mayoría se basan en sistemas de control digital, aunque existen fuentes que apuntan al desarrollo de sistemas de control mediante redes neuronales o lógica difusa

3.2 Dispositivos de entrada - salida de información de Lego Mindstorms

Este Kit está compuesto por un bloque inteligente como se mencionó anteriormente, y adicional cuenta con servomotores y diversos sensores otros dispositivos capaces de detectar información tales como comunicación Bluetooth. En este espacio nos centraremos en los elementos necesarios para la programación de los agentes para fútbol robótico entre los cuales se encuentran: [6]

1 Bloque Inteligente NXT

2 Sensor de Color

3 Sensor Ultrasónico

4 Sensor Infrarrojo (IR)

5 Brújula

6 Servomotores

7 Pelota Infrarroja

3.2.1 Bloque Inteligente NXT

Figura 3.1. *Bloque lego NXT*

El NXT es el cerebro del Robot Mindstorms. Es un sistema inteligente, controlado por un ordenador, que permite al LEGO MINDSTORMS cobrar vida y realizar diferentes operaciones. Esta cuenta con: [6]

- Puertos de Motor: El NXT tiene tres puertos de salida para el montaje de motores identificados con letras A, B y C

- Puertos de Sensor: El NXT tiene cuatro puertos de entrada para el montaje de sensores identificados con los números 1, 2, 3 y 4.

- Puerto USB: Conecta un cable USB al puerto USB y descarga programas de su computador al NXT (o cargar datos del robot a su computador) también se puede usar la conexión Bluetooth para cargar o descargar, del PC a NXT y viceversa.

- Loudspeaker: Reproducciones con sonidos reales cuando se esté corriendo el programa.

- Botones del bloque NXT: Botón naranja: On/enter/Run. Flechas grises: Usadas para moverse de izquierda a derecha en el menú del NXT.Boton gris oscuro: Borrar/volver

- Especificaciones técnicas

- Microcontrolador de 32-bit ARM7
- 256 kbytes FLASH, 64 kbytes RAM
- Microcontrolador de 8-bit AVR
- 4 Kbytes FLASH, 512 byte RAM
- Conexión inalámbrica Bluetooth clase II v2.0 complaciente.
- Puerto USB de velocidad completa(12 Mbit/s)
- 4 Puertos de entrada, 6-wire cable digital platform (un puerto incluye un IEC tipo 61158 4/EN 50 170, puerto de expansión compatible para usos futuros)
- 3 puertos de salidas, 6-wire cable digital platform
- Display gráfico LCD de 100 x 64 pixeles
- Parlante alto con calidad de sonido de 8 khz. Canal de sonido con 8-bit de resolución y 2-16 khz de rango de ejemplo.
- Fuente de energía dependiente de 6 baterías AA.

3.2.2 Sensor de Color

Figura 3.2 *sensor de color*

El sensor de color es uno de los sensores que da al robot una visión.

El sensor de color tiene tres diferentes funciones. El sensor de color permite al robot distinguir entre color, luz y oscuridad. Él puede detectar 6 diferentes colores, leer la intensidad de la luz en un cuarto y medir la misma intensidad de luz de colores en las superficies. El sensor de color también puede ser usado como lámpara de color.

Sugerencias para el uso:

Se puede usar el sensor de color para ordenar los ladrillos de LEGO, o para hacer un robot que siga una línea roja o cambie de dirección cuando detecte un punto rojo.

También se puede usar el sensor como lámpara de color para darle al robot iluminación es capaz de emitir 3 colores de luz.

Funcionamiento [7]

El sensor ilumina la superficie del objetivo con tres fuentes de luz (diodos emisores de luz LEDs). La diferencia que hay de luz ambiente y las fuentes de luz del sensor se utiliza para medir la luz de cada color absorbida por la superficie del objeto. Los valores de color se procesan y corrigen la dispersión en el espectro de emisión de luz de cada led.

El ladrillo NXT recibe tres valores: el nivel de ROJO, el nivel de VERDE y el nivel de AZUL. El valor correspondiente a cada color está comprendido entre 0 y 255.

Color	Rojo	Verde	Azul
Negro	0	0	0
Blanco	255	255	255
Rojo	255	0	0
Verde	0	255	0
Azul	0	0	255
Amarillo	255	255	0

Figura 3.3. Relación de colores del sensor [7]

El sensor actualiza las lecturas a razón de 100 muestras por segundo. [7]

3.2.3 Sensor Ultrasónico

Figura 3.4. Sensor de ultrasonido

Este sensor le da al robot la capacidad de tener visión. Este sensor le ayuda al robot a detectar objetos. También se puede usar este sensor para hacer que el robot evite obstáculos, cense y mida distancias, además de detectar movimiento. [6]

El sensor de ultrasonido mide distancias en centímetros y en hinches. Esto puede medir distancias de 0 a 255 centímetros con una precisión de +/- 3 cm. El rango en pulgadas es de 0 a 100 pulgadas.

El sensor de ultrasonido usa el mismo principio científico de los murciélagos; este mide la distancia por medio del cálculo del tiempo que toma para una onda de sonido en golpear un objeto y volver. Justo como el eco.

El sensor se destaca al momento de detectar objetos de grandes dimensiones como una superficie dura, esta mostrará las mejores lecturas. En cambio los objetos hechos de tela o de materiales suaves o que tengan superficie curva (como la bola) muy finas o pequeñas para ser detectados y pueden ser difíciles para el censado del sensor de ultrasonido.

Nota: si dos o más, sensores de ultrasonido operan en el mismo cuarto puede interrumpir la lectura el uno del otro.

3.2.4 Sensor Infrarrojo

El sensor infrarrojo NXT IRSeeker V2 es un detector de señales infrarrojas multielemento con capacidad para detectar señales de diferentes fuentes emisoras de infrarrojo como la pelota infrarroja, controles remotos y hasta luz solar. [8]

Figura 3.5. Sensor RSeeker V2

El IRSeeker V2 usa un sistema avanzado de procesamiento de señales digitales que le permite hacer un filtro de las señales recibidas y usar solo las señales requeridas.

-Funcionamiento:

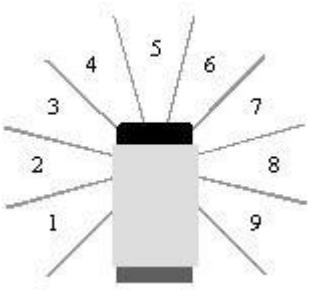


Figura 3.6. *Campo de visión sensor RSeeker V2*

El IRSeeker V2 divide la zona en la cual es capaz de percibir las señales emitidas por una fuente, en 9 sub-secciones la cuales indican que el dato que envía el sensor al bloque NXT, será un valor numérico entero, que corresponde a la dirección de donde proviene la señal. Si el valor retornado por el sensor es 0, significa que no se percibe ninguna señal en el medio. Haciendo un análisis rápido podemos observar, según se indica en la imagen anterior, que cada zona barre aproximadamente 30° de circunferencia con centro en el sensor, esto es algo que puede ser útil al momento de programar, ya que de allí, se deduce fácilmente que la zona 0 o de no recepción o de zona muerta de aproximadamente 60° al respaldo del sensor, o que tiene un rango de “visión” de 300°. [8]

3.2.5 Brújula

Figura 3.7.Brújula

La brújula digital magnética mide el campo magnético de la tierra y calcula un ángulo de partida. El sensor de la brújula se conecta a un puerto de sensor NXT usando un patrón de

alambre de NXT y utiliza el protocolo de comunicación digital I2C. El título actual se calcula con una precisión de 1 ° y actualiza 100 veces por segundo. [8]

3.2.6 Servomotores

Figura 3.8. Servomotor

El servomotor interactivo es el encargado del movimiento, teniendo este internamente un sensor de rotación que mide la velocidad y la distancia, y envía estos datos al bloque lógico NXT, esto permite tomar medidas precisas y un control del motor completo dentro de un grado de precisión. [6]

3.2.7 Pelota Infrarroja

Figura 3.9 .bola de juego

La bola electrónica infrarroja tiene cuatro modos de funcionamientos, adecuados para jugar fútbol robótico, también da la opción de ser un transmisor infrarrojo para ser buscado por el robot. [8]

La pelota está equilibrada, cuenta con 20 led infrarrojos, que dan cobertura completa de señal IR, cumpliendo con los requisitos impuestos por Robocup Jr, de pulso modulado.

Esta bola cuenta con dos modos adicionales de pulso de 1200 Hz y 600 Hz, proporcionando mayor flexibilidad y rango de detección entregando hasta 5 metros o (15 pies).

La gran cualidad de la bola IR, es su capacidad de ser detectada en condiciones difíciles de iluminación de fondo.

Esta bola opera en conjunto con el sensor IR Seeker. Mide 75 mm (~ 3 pulgadas) de diámetro. Su alimentación eléctrica requiere 4 pilas AAA.

CAPÍTULO 4

DISEÑO Y CONFIGURACIÓN

4.1 Estructura física base del robot

La estructura base para un óptimo funcionamiento del robot tanto en velocidad como en fuerza, se basa en la unión de varios refuerzos de piezas fijas que están ensambladas al motor para generar estabilidad al momento de ejercer movimientos bruscos o al estrellarse. Después de analizar varios modelos o estructuras (figura A y B), y poniéndolos en pruebas preliminares se puede constatar que el modelo de cintas de orugas, Con una patada lateral, o de empuje frontal para el caso del delantero no es la más óptima, por lo tanto se realiza un montaje de triciclo.

Figura 4.1 *Modelo de cinta de Oruga*

Por normatividad todos los jugadores deben tener un chasis base iguala para todos, ya la parte de accesorios (sensores, llantas, modificaciones) depende de la funcionalidad de cada robot, para nuestro caso se empieza la construcción de chasis por lo que sería la caja de la dirección

Figura 4.2 *caja de dirección*

Paso seguido se adaptan a esta caja los soportes de los servomotores, para que queden firmes se soportan en dos puntos, quedando la parte principal de esta manera.

Figura 4.3 *Soporte de los motores*

Teniendo ya la base para sostener los motores los instalamos sobre el chasis quedando en una posición simétrica como lo muestra la imagen

Figura 4.4 *Motores en el modelo*

Una vez montados los motores sobre el chasis lo que resta es poner soportes para dejar el chasis rígido y terminado con la opción de la barra de soporte la que se vería así:

Figura 4.5 *Parte fija del chasis*

Después de tener el chasis armado, procedemos a instalar la parte de la dirección que parar este casi se realizará con una rueda loca, quedando el chasis principal terminado

Figura 4.6 *Chasis terminado*

Ya para terminar el modelo base para todos los jugadores, le instalamos el bloque lógico, y le conectamos los motores, quedando el agente base de esta manera

Figura 4.7 *Base final de los agentes*

4.1.1 Arquero

Una vez tenemos el modelo base empezamos a diferenciar a nuestros agentes por su funcionalidad y por los sensores utilizados y la ubicación en donde se requieren estos mismos. Por lo tanto empezaremos por el arquero.

Figura 4.8 *Imagen del arquero*

en la imagen del arquero nos podemos dar cuenta de la ubicación de los sensores tanto del IRSeeker como el del compás ubicación determinada por las medidas de instalación en donde indican que estos sensores para su mejor funcionamiento deben estar a 5cm del bloque lógico y a 15 cm de los motores, y un sensor de color que se encuentra ubicado detrás de la pala de contención (armadura robusta resistente a los fuertes impactos, provenientes del el oponente o el mismo balón) con esta configuración de triciclo el arquero tiene una mejor reacción y reflejos al acercarse la pelota.

4.1.2 Defensa

Para el agente en posición de defensa fue un poca más fácil puesto que la labor de este es despejar la pelota, para este fin se utiliza la misma configuración del arquero la diferencia entre estas dos plataformas es el frente ya que la pala defensiva es curva a diferencia del arquero para tener un poco más de maniobrabilidad y no tan grande para que el agente sea más ligero más veloz y pueda ir a la pelota más rápido, por lo tanto este agente se vería de la siguiente manera.

Figura 4.9: *Imagen del Defensa*

4.1.4 Delantero

Por último tenemos la configuración del agente en posición de Delantero, el cual por su rol es necesario la configuración de un mecanismo de pateo, para lo que se le diseñó un pie mecánico con un servomotor adaptado a la parte delantera. Sumado a esto, el agente en posición de delantero se le adiciona el sensor de ultrasonido para la detección u golpe de la pelota, la variación en la colocación de los sensores en este diseño fue para evitar interferencia entre ellos por eso el sensor de brújula y el IRSeeker están ubicados al mismo lado, por lo tanto este agente se vería así:

Figura 4.10: *Imagen del Delantero*

4.2 Bloques de Programación

El software NXT permite el paso del programa desarrollado en PC o Mac al bloque inteligente de Lego Mindstorms. Este software se basa en la programación mediante bloques de acciones, lo que hace de esta tarea muy intuitiva y fácil ya que solo debemos estar conscientes de lo que sucede en la capa “más superficial” de la programación, ya que solo nos interesa configurar las tareas objetivo sin tener que preocuparnos por la configuración de los periféricos,

Los bloques que se describen a continuación no abarcan la totalidad de los existentes para el software NXT, pero sí explica todos los necesarios para la programación en fútbol robótico.

4.2.1 Bloque Loop

Figura 4.11. *Bloque loop*

Este bloque es utilizado para repetir secuencias de código. En donde por su configuración se puede parametrizar para que termine el bucle ya sea por el número de repeticiones, una señal lógica o un sensor, o solo dejarlo en un bucle infinito.

4.2.2 Bloque Sensor de Color

Figura 4.12. *Bloque sensor de color*

Este es el gráfico del bloque del sensor del color el cual su configuración se divide en dos partes, la primera es la de detectar distintos colores y la segunda medir la intensidad de la luz. En el gráfico vemos que en número uno es el puerto en donde se conecta, el dos vemos el modo en el que se encuentra trabajando el sensor, y el tercero es el concentrador de datos.

El bloque de sensor de color tiene dos modos: uno para la detección de diferentes colores y el otro para medir la intensidad de la luz.

Para generar un "verdadero" en la señal, se utiliza el menú desplegable en el panel de configuración para seleccionar dentro del rango o generar un "falso" es decir, una señal fuera del rango seleccionado. Si la configuración predeterminada para el bloque del sensor de color está dentro del rango establecido para la detección del amarillo. Será un "verdadero" de la señal y afirmara el color amarillo y la "falsa" de la señal será cualquier otro color.

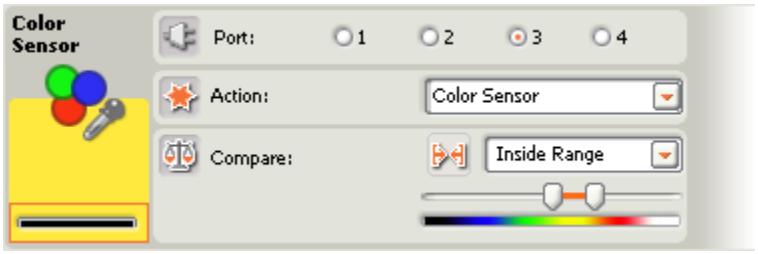


Figura 4.13. *Configuración bloque de color*

Detección del sensor de luz propia Luz Reflejada

El sensor de luz enciende su propio diodo emisor de luz y detectar si esta luz se refleja de nuevo a él. Esta función es especialmente útil en condiciones de luz difíciles, como las habitaciones son muy oscuras o en condiciones de luz cambiantes. La función también permite que el sensor de luz para servir como un telémetro de corta distancie. Con el "Light" activado, un sensor de luz que se acerca un objeto reflectante detecta altos niveles de luz reflejada.

Configuración del concentrador del bloque sensor de color de Datos

Se puede controlar el sensor de color de forma dinámica mediante la conexión de los cables de datos al cubo del bloque del sensor de color de los datos.

Figura 4.14. *Bloque de color*

Se abre en el centro de un bloque de datos, haga clic en la pestaña en el borde inferior izquierdo del bloque después de haber sido colocado en el área de trabajo.

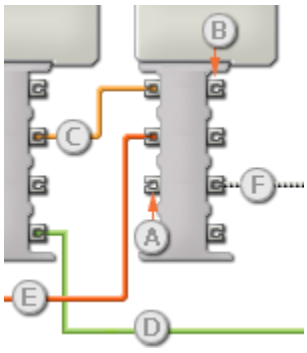


Figura 4.15. *Conexión sensores*

[A] Input plug

[B] Output plug

[C] Number data wire (yellow)

[D] Logic data wire (green)

[E] Text data wire (orange)

[F] Broken data wire (gray)

Pasar datos desde el enchufe de entrada a la clavija de salida

Si el enchufe de entrada tiene un conector de salida correspondiente, los datos de entrada pasarán a través de la clavija de entrada a la clavija de salida sin ser cambiado. En este caso, sólo se puede utilizar la clavija de salida si la clavija de entrada está conectado a un cable de entrada de datos; conectar un cable de datos de salida a dicha clavija de salida sin una entrada conectada cable de datos hará que el cable de salida de datos a ser "roto" (y de color gris).

Los cables de datos llevar a tipos específicos de datos

Cada cable de datos transporta un tipo específico de datos entre los bloques.

Colores Cable de Datos

Los cables de datos se identifican con colores específicos: cables que transportan datos de números son de color amarillo, los cables que transportan datos lógicos son de color verde y los cables que transportan datos de texto son de color naranja.

"Rotos" cables de datos

Si intenta conectar un cable de datos a un enchufe del tipo de datos incorrecto, el cable de datos se rompe (y de color gris). Usted no será capaz de descargar el programa si un cable de datos está roto.

Para los enchufes que acepten mayores rangos de entrada (ejemplo: 0 - 100), el tapón obligará a cualquier entrada fuera de su área de distribución a la medida. Por ejemplo, si un bloque Mover enchufe Power recibe un valor de entrada de 150, el bloque va a cambiar el valor de entrada a 100 (es decir, un número dentro del rango del enchufe Power).

	Plug	Data Type	Possible Range	What the Values Mean	This Plug is Ignored When...
	Port	Number	1 - 4	Corresponds to the numbered input port on the NXT.	Never ignored
	Range	Logic	True / False	True = Inside Range False = Outside Range	In Light Sensor Mode
	Color Range A	Number	0 - 6	0 = Left of Black 1 = Between Black and Blue 2 = Between Blue and Green 3 = Between Green and Yellow 4 = Between Yellow and Red 5 = Between Red and White 6 = Right of White	In Light Sensor mode
	Color Range B	Number	0 - 6	0 = Left of Black 1 = Between Black and Blue 2 = Between Blue and Green 3 = Between Green and Yellow 4 = Between Yellow and Red 5 = Between Red and White 6 = Right of White	In Light Sensor mode
	Greater/Less	Logic	True/False	Logic used in comparison: True = Greater, False = Less	In Color Sensor Mode
	Trigger Point	Number	0 - 100	Value to compare against	In Color Sensor Mode
	Generate Light	Logic	True/False	Determines if the sensor's own LED is on or not	In Color sensor mode
	Lamp Color	Number	0-2	0 = Red 1 = Green 2 = Blue	In Color Sensor mode
	Yes/No	Logic	True/False	Result of comparison	Never ignored
	Detected Color	Number	1-6	1 = Black 2 = Blue 3 = Green 4 = Yellow 5 = Red 6 = White	In Light Sensor mode

Tabla 5

4.2.3 Bloque Comparar

Figura 4.16. *Bloque para comparar valores*

Con este bloque se puede determinar si un número es menor (<), mayor (>), o igual a otro (=), para poder realizar límites con los datos que recibe o variables fijas suministradas manualmente.

4.2.4 Bloque Switch

Figura 4.17. *Bloque switch*

El switch mostrado en el gráfico en sus dos presentaciones es el bloque encargado de elegir entre dos secuencias de código, ya seo por configuración en un sensor o la toma de decisión de un verdadero o falso, en cualquiera de los dos casos se activará la secuencia que corresponda a cada acción determinada.

4.2.5 Bloque Mover

Figura 4.18. *Bloque de motor*

Este es el bloque que permite controlar con exactitud el movimiento del motor, aumentando o disminuyendo la velocidad hasta alcanzar una velocidad establecida o detenerlo, en el número 1 vemos el puerto al que se conecta en el 2 nos muestra la dirección del motor, en el 3 el nivel de la potencia , en el 4 la duración del movimiento que puede ser ilimitada, en grados o segundos y en el 5 encontramos el concentrador de datos en donde se le ingresan los datos en forma dinámica o manual en donde se le puede configurar los grados de salida, la dirección, tiempos de espera, la duración , la potencia entre otros.

4.2.6 Bloque IRSeekerV2 Mejorado

Este bloque utiliza una combinación de modos de señal AC y DC para mejorar la recepción de la señal, tal como se muestra nuevamente en la figura:

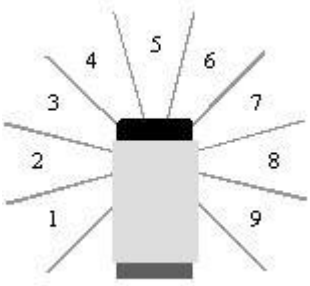


Figura 4.19. *Sensor IRSeekerV2*

De allí obtenemos la dirección de la señal Infrarroja (de aquí en adelante IR), que es equivalente al número que retorna el sensor al bloque nxt, el fabricante expresa esta como IRdirection. Este bloque también cuenta con una salida de potencia de la señal y una salida lógica que dice si hay o no recepción de una señal IR [8]

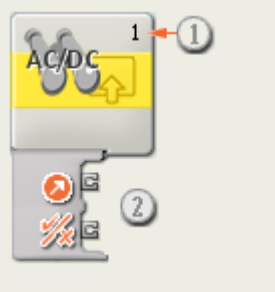


Figura 4.20. *Bloque de configuración IRSeekerV2*

Adicional a la IRDirection, que envía un dato numérico del 1 al 9, este bloque cuenta con otra modalidad llamada DegDirection, la cual nos da la dirección de la fuente en grados; la ecuación que rige esta modalidad viene definida como:

$$DegDirection = (IRDirection - 5)*30 \quad (5)$$

De (5) es fácilmente deducible que nuestro 0° se encuentra justo al frente del sensor, y que el valor del ángulo es relativo ya que este es negativo si la fuente se encuentra hacia la izquierda, o positivo si esta se encuentra hacia la derecha del sensor.

Nótese también que si IRDirection = 0, entonces DegDirection = -150, valor de salida que, si este bloque está conectado por medio de la salida DegDirection, a la entrada Steering del bloque mover, causará un giro de los motores con el fin de buscar un fuente de señal IR a la redonda del sensor, rastreando con este giro, los puntos que anteriormente eran “muertos”.

Las opciones de conexión de este bloque con otros bloques se muestran a continuación:

Figura 4.21. *Bloque sensor IRSeekerV2 configuración*

Dónde:

1 Retorna el número del puerto al que está conectado el IRSeeker V2

2 Retorna el valor de la dirección (IRDirection)

3 Retorna la dirección en grados (DegDirection)

4 Retorna una prueba lógica de la recepción de señal IR

5 Retorna el valor asociado a la potencia de la señal

4.2.7 Bloque Sensor de brújula

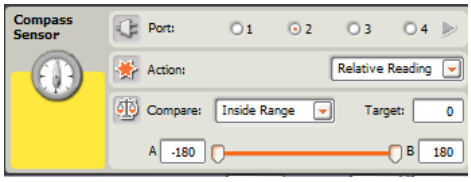


Figura 4.22. *Configuración brújula*

Este sensor es el que se encarga de una navegación precisa. El sensor NXT Compass es una brújula digital que mide el campo magnético de la Tierra y emite un valor que representa el rumbo actual. El rumbo magnético se calcula con una precisión de 1 ° y se devuelve como un número de 0 a 359. Tomando como dirección de barrido el sentido horario. El sensor

NXT Brújula magnética actualiza el epígrafe 100 veces por segundo. El sensor de la brújula funciona en dos modos, el modo de lectura y el modo de calibración. En el modo de lectura, el rumbo actual se calcula y se reinicia cada vez que el programa NXT ejecuta un comando de lectura. En modo Calibrar la brújula puede ser calibrado para compensar anomalías generadas por el campo magnético, tales como los que rodean los motores y baterías, manteniendo de ese modo la máxima precisión.

4.2.8 Bloque Math

Figura 4.23. *Bloque matemático*

Este es el bloque que realiza las operaciones matemáticas simples como la adición, sustracción, división y multiplicación, también realiza operaciones más complejas como valor absoluto, y raíz cuadrada, al igual que muchos bloques se le pueden ingresar los datos digitados o en forma dinámica.

4.2.9 Bloque Variable

Figura 4.24.*Bloque de variable*

Las variables son sitios en donde se almacenan valores dentro de la memoria del bloque lógico, otros bloques del programa pueden leer el valor actual de la variable o cambiar el dato si fuese necesario

4.2.10 Bloque Espera

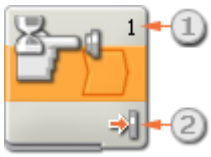


Figura 4.25. *Bloque de espera*

Este es el bloque encargado de dar tiempos de espera para poder detectar determinada condición antes de continuar, el numero 1 indica que puerto controla el bloque y el número 2 es el método elegido de activación sea chocado, liberado o censado de un sensor de luz, de sonido o el ultrasonido.

CAPÍTULO 5

ALGORITMOS DE COMPORTAMIENTO DE LOS AGENTES

5.1 Algoritmo de comportamiento del Arquero

De acuerdo a la sección 2.3 del libro, se han acordado 3 variables a controlar para el agente arquero, la cuales son:

1 Orientación Espacial

2 Reducción de espacio

3 Despeje de balón

Donde cada una de ellas requiere un tratamiento algorítmico especial y la creación de variables de control según los sensores a utilizar. Los diferentes algoritmos diseñados para el control de estas variables se explican a continuación.

5.1.1 Control de Orientación

La orientación del agente se da en diferentes ocasiones, bien sea al inicio del juego para estar listo cuando este empiece, o después de una intervención del arquero en una jugada de gol; debido a que una nueva orientación se debe realizar cada vez que suceda una intervención del agente, esta operación requiere de un único algoritmo al igual que la ubicación inicial.

5.1.1.1 Orientación Inicial

Para la orientación inicial debemos tener en cuenta que el arquero debe permanecer direccionado hacia el sur y lo más cercano al punto medio de su área de acción tal como se explica en la sección 2.3, recordemos que:

$O = 180°$

Esto es de acuerdo a las características del Compass Sensor que se utiliza.

Antes de esto debemos intentar ubicar el agente lo más cercano al centro de su área, debido a que no contamos con un sistema standard para la ubicación de los agentes, ni con vista superior o aérea para indicarles a estos su posición, se hace necesario diseñar un algoritmo que permita un análisis de reconocimiento del área, para esto, contando con las herramientas de Lego Mindstorms, contamos con el Compass Sensor y con contadores que registran el número de veces que se ejecuta un ciclo, con lo cual podemos contar cuantas veces se ejecuta un ciclo mientras el agente avanza hacia una de las líneas laterales que delimitan su área. Gráficamente esto es:

Figura 5.1 *Reconocimiento de límite*

Nótese que si nuestro arco va dirigido hacia el sur, el límite izquierdo es el oriente, por lo que nuestro agente debe dirigirse con una dirección O=90 según la especificación técnica del Compass Sensor. El algoritmo para esta tarea es el siguiente:

Considérese la variable ctrl2 inicializada en ctrl2=0; esta variable controla el giro del agente buscando el oriente:

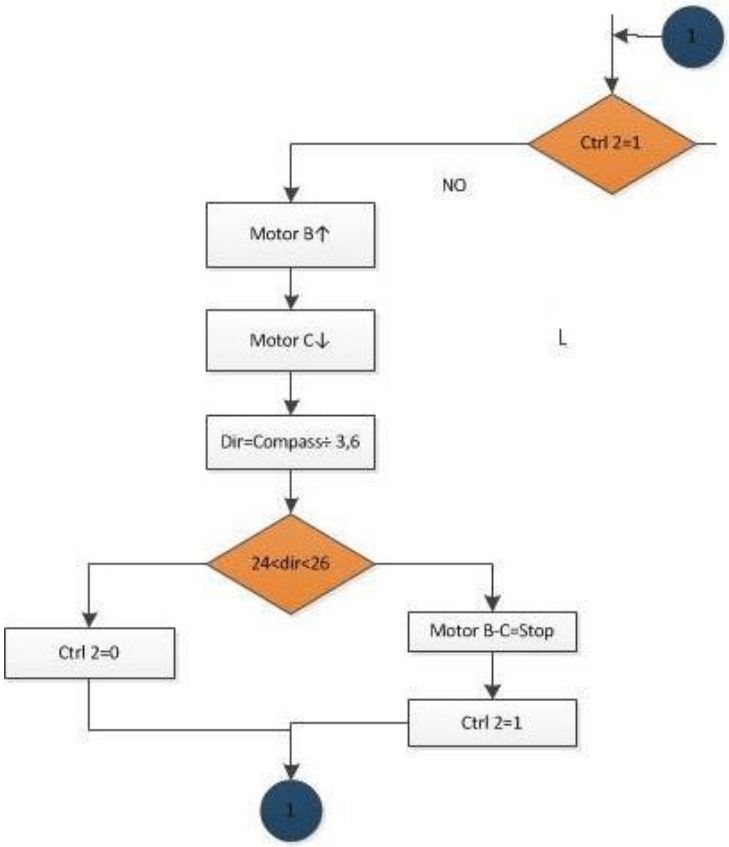


Figura 5.2

Recuérdese que como nuestro Range del Lego Mindstorms solo acepta valores entre 0 y 100, se debe realizar la conversión de escala por lo que se llama una variable:

$Dir = Compass \div 3{,}6$

Mientras Dir no se encuentre en el intervalo (24<Dir<26), el agente estará girando en busca del Oriente que es la dirección correspondiente a estos valores, una vez se cumpla esta condición, se debe detener el agente y:

ctrl2=1

Con lo que abandonamos este Loop...

Una vez ubicados en esta dirección debemos dirigirnos hasta la línea oriental, para esto debemos censar constantemente el color de la superficie de desplazamiento hasta que la misma sea blanca, indicándonos que estamos sobre esta línea. Considérese la variable:

CH3=0;

Esta controla el desplazamiento y altera su valor cuando el sensor de color indique que se ha llegado a la línea blanca tal como se muestra a continuación:

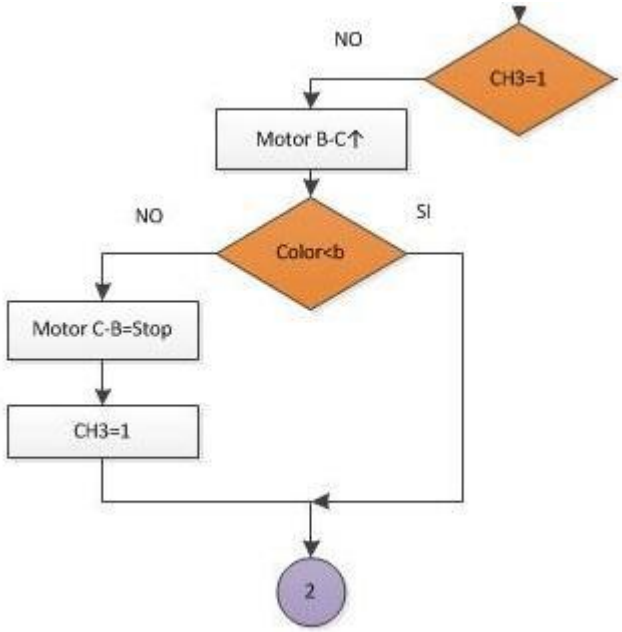


Figura 5.3

Dónde Color=6, es el valor de la variable Color, cuando el sensor detecta el blanco. Los valores de los colores primarios que puede censar Lego se explicó con anterioridad en la sección de bloques a utilizar. Una vez se incumpla la condición de color, la variable CH3 se incrementa en 1 para abandonar el ciclo correspondiente.

Detectado el límite oriental, ahora debemos dirigirnos “de espalda” hacia el límite occidental con el fin de contar, bien sea, el número de veces que se repite el ciclo encargado de

movernos en reversa, o el número de veces que giran los motores del agente, esto como un control de tiempo de desplazamiento. Considérese ahora la variable

CH1=0;

La cual se incrementa en 1 (CH1++) cada vez que se ejecuta el ciclo encargado de desplazarse en reversa:

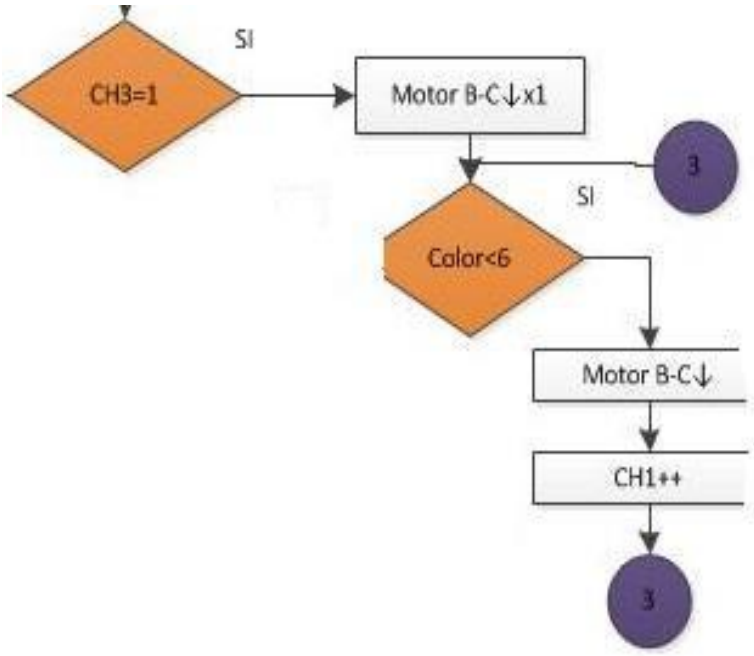


Figura 5.4

Una vez se incumpla esta condición de color, (Color<6), se debe detener el agente, y por matemática básica deducimos que si nos desplazamos esta vez nuevamente hacia el oriente, creamos una variable CH2 y dicho desplazamiento se ejecute siempre y cuando se cumpla que:

$$CH2 < ((CH1 \div 2) + 1)$$

Para esto, se hace requisito que, de la misma forma que CH1, CH2 se incremente en 1 (CH2++), a medida que se realiza el desplazamiento pero esta vez hacia adelante; con esto

podemos definir un punto cercano al centro de nuestra área de acción para ubicarnos tal como se muestra en la sección 2.3.

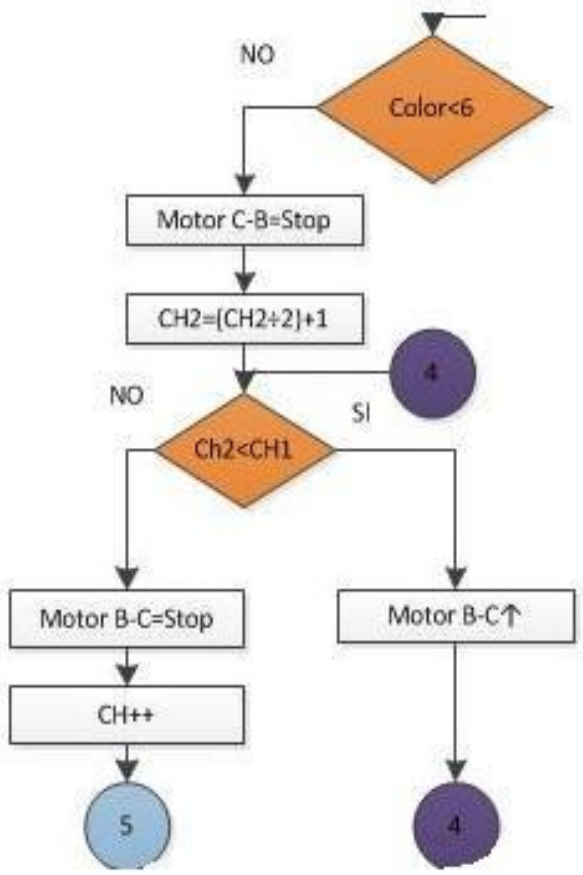


Figura 5.5

En la figura anterior observamos que en el momento que CH2>CH1, la variable CH se incrementa en 1; la cual controla la totalidad del desplazamiento horizontal, que es lo que se ha expuesto hasta este momento, lo anterior se puede ver en su totalidad a continuación:

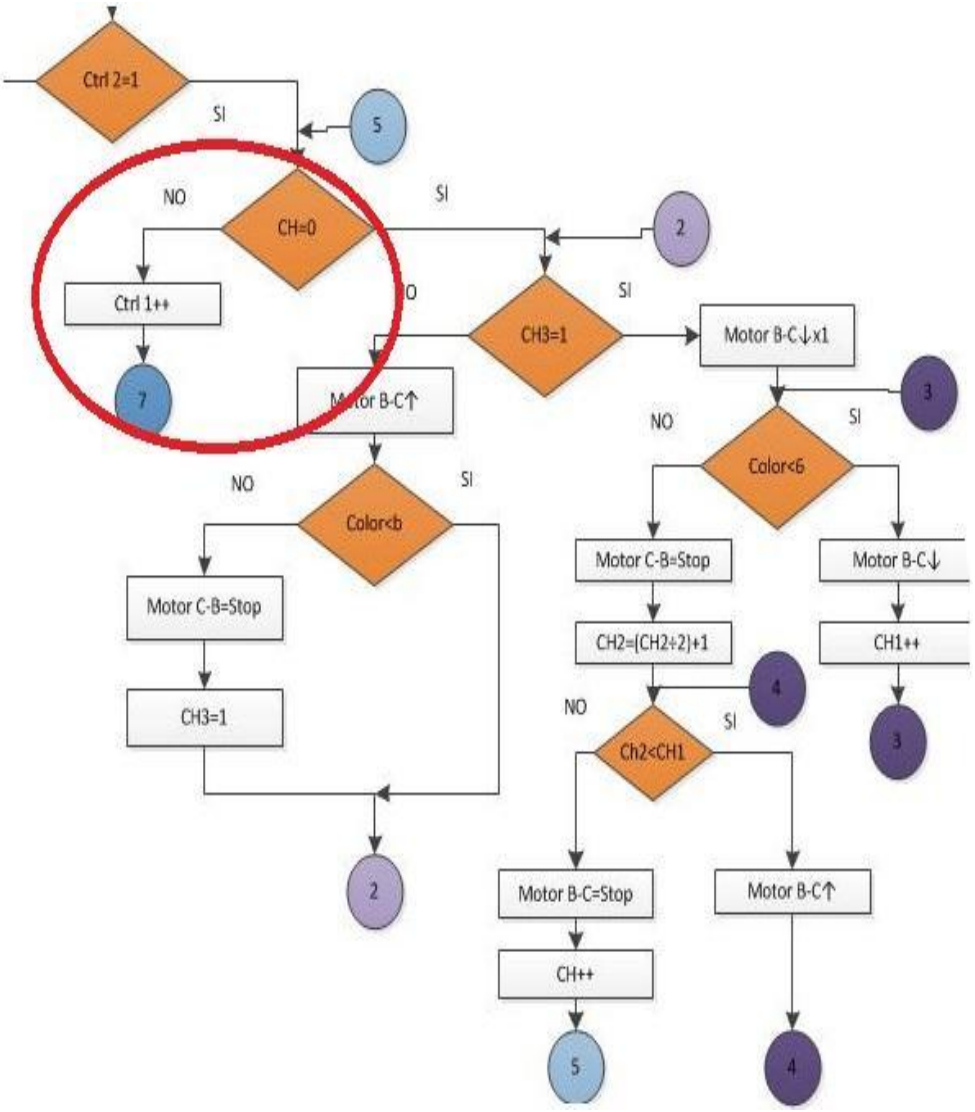


Figura 5.6

Nótese también en la imagen anterior, que al incumplir la condición CH=0, se ejecuta la instrucción Ctrl_1++, la cual es la variable de control del Loop que se ejecuta mientras se busca la posición central antes de proceder a buscar el sur.

En este punto debemos ahora buscar la orientación hacia el norte, para lo cual se establece:

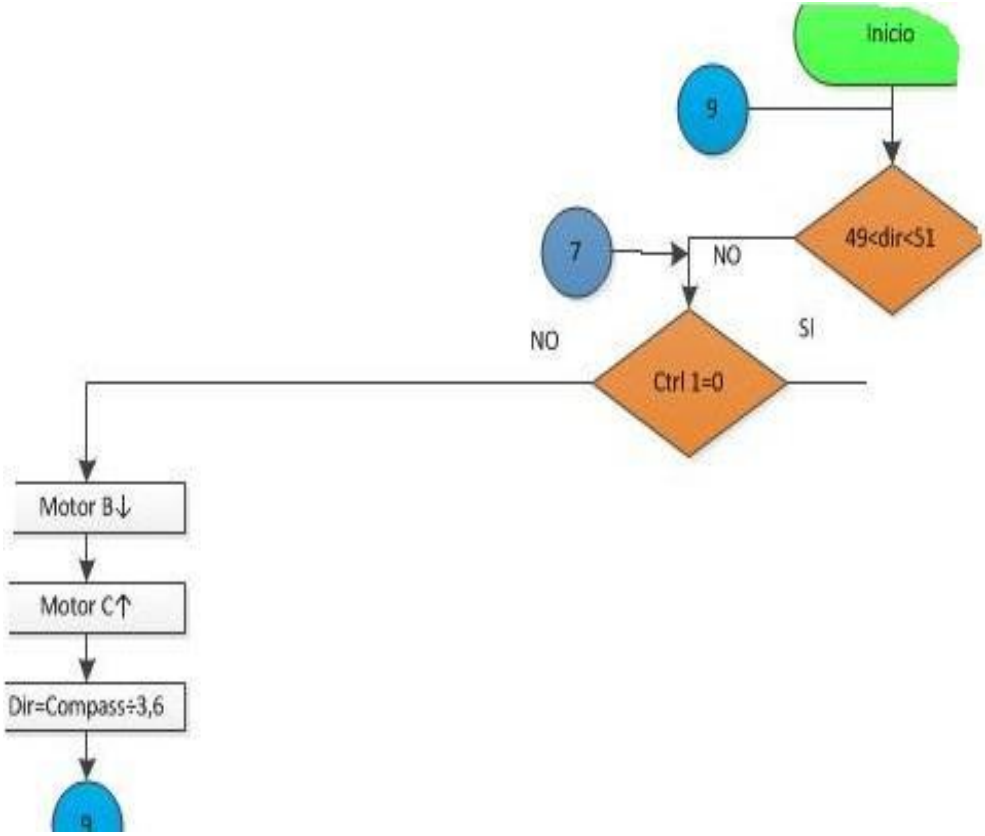


Figura 5.7

Con lo que el algoritmo completo para la ubicación es:

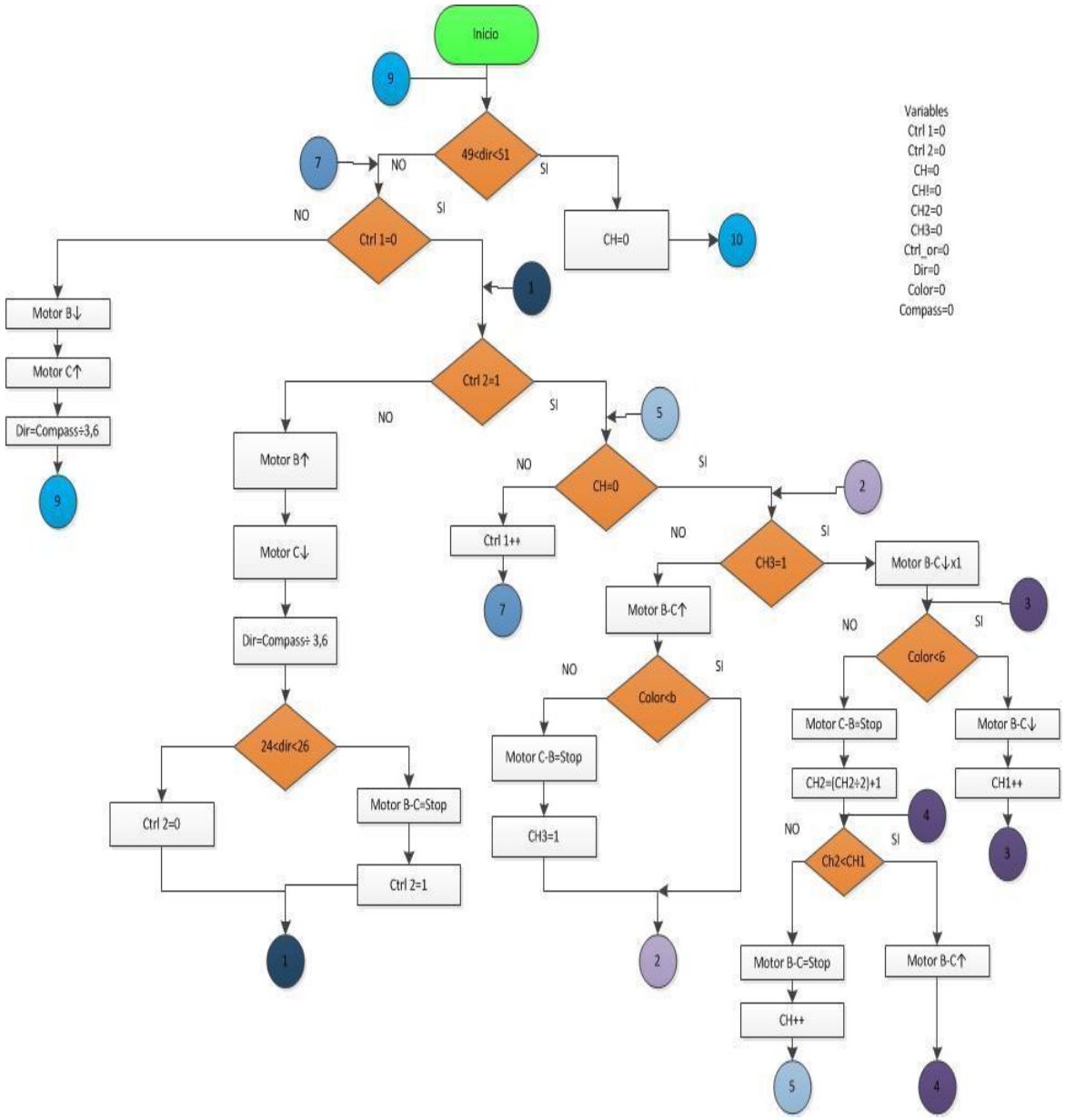


Figura 5.8

5.1.1.2 Orientaciones posteriores

Para las orientaciones posteriores solo se debe reiniciar en 0 el valor de CH2, y ya teniendo el valor de (CH1/2)+1, sólo resta desplazarse nuevamente a una de las líneas laterales y repetir el procedimiento para desplazarse mientras se incrementa en 1 CH2 por cada vez que se ejecute el ciclo para el desplazamiento. Esto puede realizarse siempre y cuando no se esté censando la presencia de la pelota. Esto es:

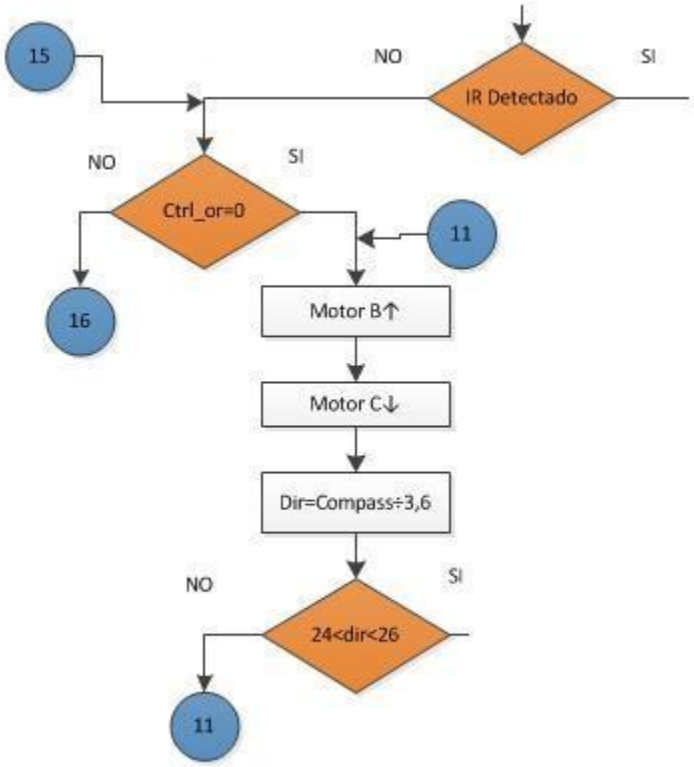


Figura 5.9

La variable Ctrl_or, recordando, es la variable que permite la reorientación por una vez y que esta no sea repetitiva en caso de que no se detecte la bola al repetir el ciclo, si esta condición es falsa entonces se retorna al principio del programa para censar nuevamente hasta que ocurra otra acción que requiera un movimiento que implique una futura reorientación. Recordemos que (24<dir<26) es la instrucción que permite que el agente se reoriente hacia el Sur, condición que, al ser falsa, reinicia las instrucciones de giro indicadas por el conector 11. Una vez cumplida la condición tenemos:

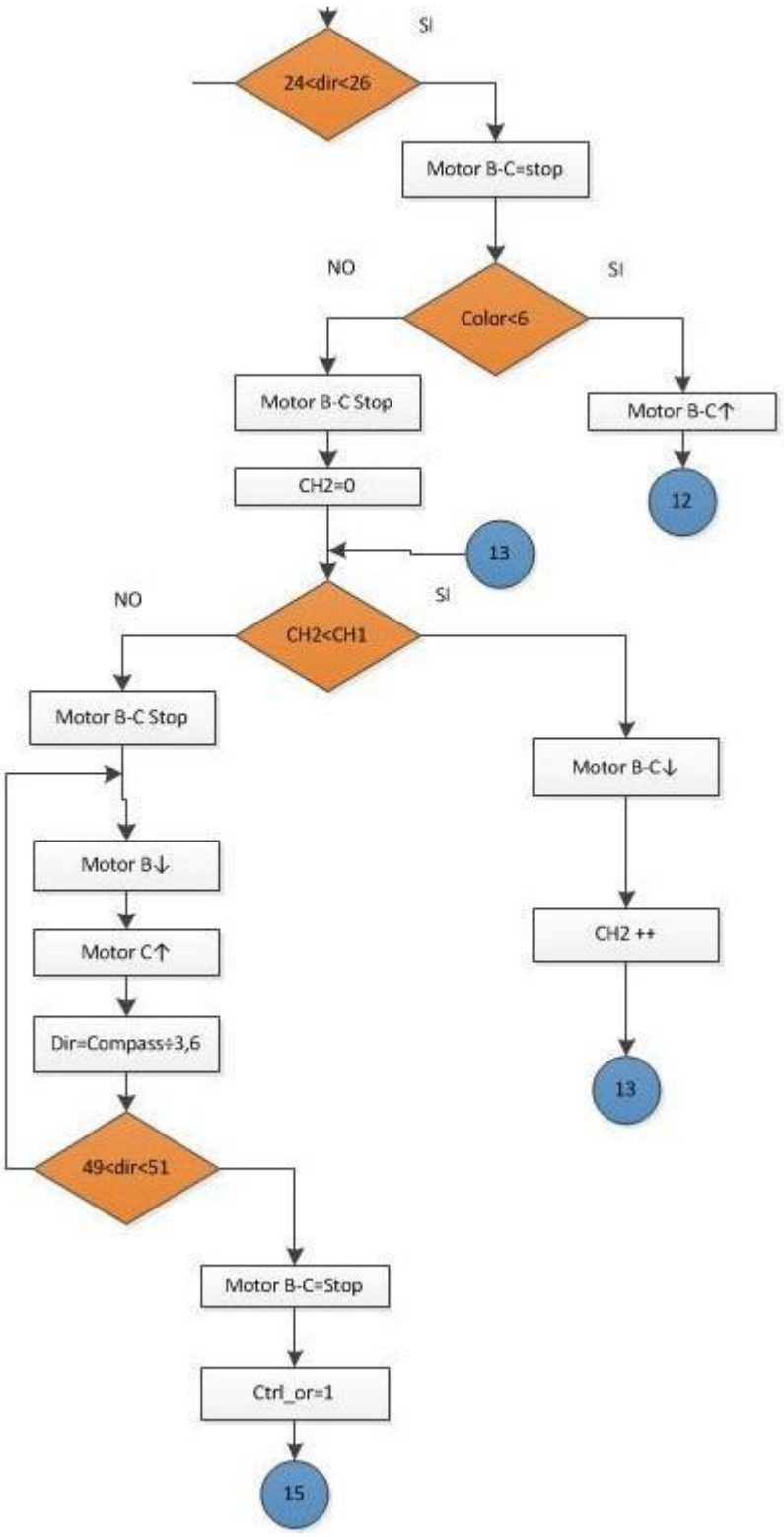

SI
24<dir<26
Motor B-C=stop
NO
SI
Color<6
Motor B-C Stop
Motor B-C↑
CH2=0
12
13
NO
SI
CH2<CH1
Motor B-C Stop
Motor B-C↓
Motor B↓
CH2 ++
Motor C↑
Dir=Compass÷3,6
13
49<dir<51
Motor B-C=Stop
Ctrl_or=1
15

Figura 5.10

Recordando que toda esta instrucción, es la misma que la ejecutada en la orientación inicial para los valores de CH1, y reiniciando CH2 en CH2=0. Nótese que al terminar la reorientación, Ctrl_or toma el valor de 1 para evitar re entradas innecesarias en la etapa de reorientación. Este valor vuelve a ser 0 cuando se ejecuta una acción relacionada con la detección de la pelota IR.

5.1.2 Límites de acción y Reducción de Espacio

Ya estando en acción, viene el control de la segunda variable del portero la cual es la reducción de espacio para interceptar la pelota y evitar el gol. Esta acción se puede controlar mediante la vigilancia permanente del agente, en búsqueda de la señal IR que genera la bola infrarroja utilizada en este tipo de competencias, y que ha sido debidamente explicada en capítulos anteriores.

Por otra parte, una condición importante para que el agente pueda avanzar hacia una reducción de espacio es *no traspasar las líneas que delimitan su zona*. Por lo tanto la reducción de espacio es *condición* de otro evento. Esto se controla nuevamente con la variable que contiene el valor del sensor de Color (Color<6), tal como se muestra en la siguiente figura:

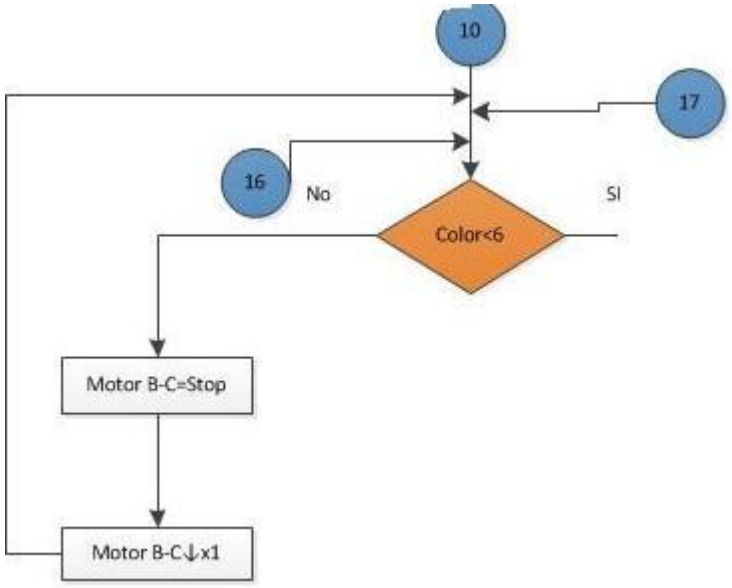


Figura 5.11

En la anterior figura se observa la continuidad del conector 10 que enlaza el algoritmo correspondiente a la primera orientación descrita anteriormente, con las demás variables a manejar en el algoritmo global.

Luego se observa que la primera condición que “vigila” el control de las demás variables, es el no traspasar los límites planteados. En caso de ser falso el agente se detiene para luego girar sus motores una revolución en sentido inverso al de avance. Esto garantiza que el agente suspenderá cualquier acción que se encuentre realizando al llegar a la línea limítrofe.

En el caso de que se cumpla Color<6 se tiene que:

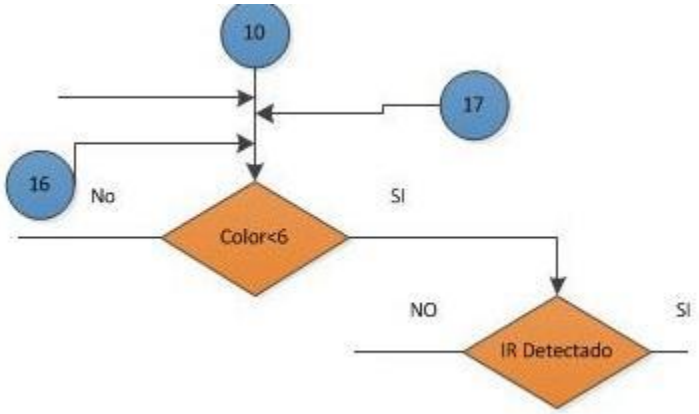


Figura 5.12

Por lo tanto, para que el agente salga a reducir espacio, primero debe saber que la bola está presente, luego, debido a que el espacio es limitado, debe tener una distancia máxima con respecto a la pelota, para salir a achicar, disminuyendo así la posibilidad de que el agente se encuentre en los límites de su área.

Para esto se hace necesario configurar otra variable que represente el valor de la opción strengh descrita para el bloque IRSeeker. Según lo descrito para este bloque, observamos que el valor de Strengh aumenta a medida que la “IR Source” se acerca y disminuye si ocurre lo contrario; por lo tanto la configuración del algoritmo viene dada por:

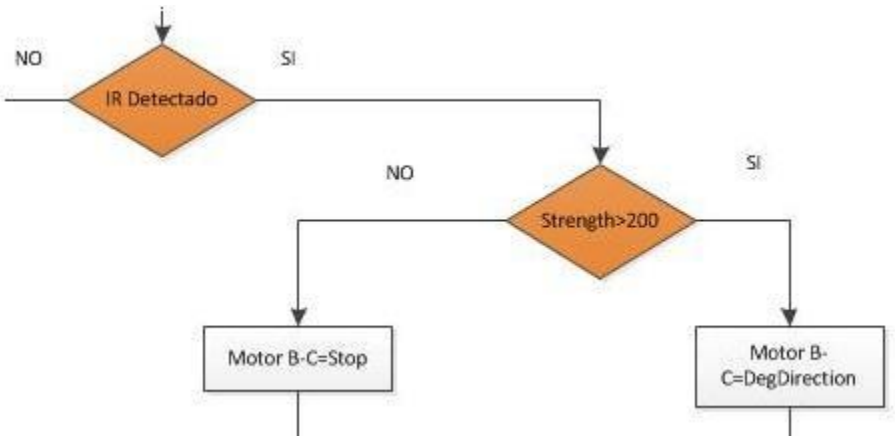


Figura 5.13

SI dicho valor es superior a 200. El agente “decide” salir a la reducción de espacio, mientras que si esta condición es falsa, el agente se detiene. Recuérdese que DegDirection es una funcionalidad del bloque Move descrita en la sección correspondiente a Lego.

Esta configuración permite al agente NO ir al “achique”, si la bola está demasiado lejos de su área de acción y así evitar salir de su posición adecuada sin necesidad.

Si la bola está en contacto con el agente, esto es, con su strengh máximo, entonces se debe proceder al despeje de balón.

5.1.3 Despeje de balón y Límites de acción

Luego de ejecutadas las instrucciones anteriores implícitas en el condicional que se mostró en la figura anterior, se tiene que:

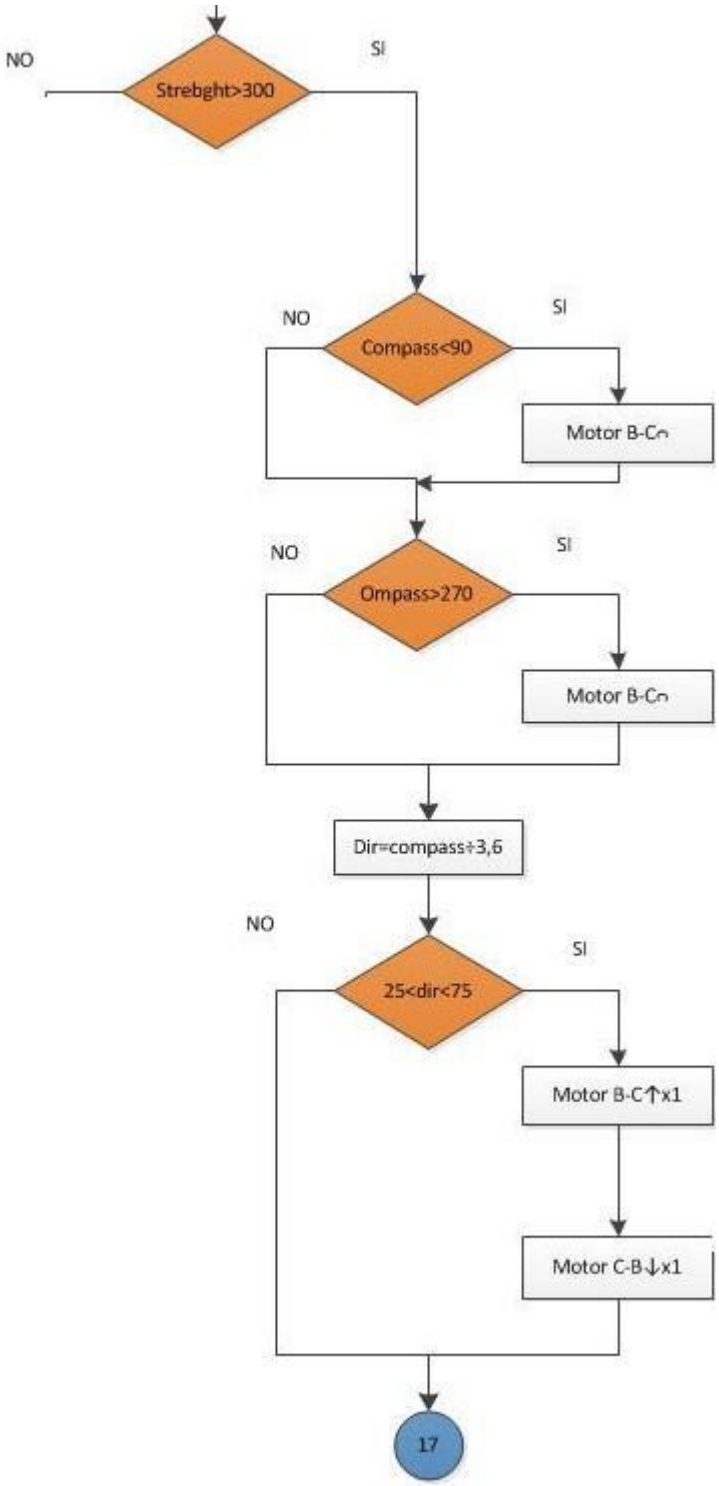

NO
Strebght>300
SI
NO
Compass<90
SI
Motor B-C
NO
Ompass>270
SI
Motor B-C
Dir=compass÷3,6
NO
25<dir<75
SI
Motor B-C↑x1
Motor C-B↓x1
17

Figura 5.14

Como se mencionó anteriormente, si el valor de la condición strengh se acerca a su valor máximo, lo cual quiere decir que la pelota está muy cerca al agente, éste debe despejarla, pero para esto debe asegurar que su despeje no causará un autogol. Para esto se debe censar la orientación del agente y según sea esta, se determina una acción a ejecutar.

Como se evidencia en el segmento de código anterior, según sea la orientación del agente, la cual se comprueba mediante la lectura del Compass Sensor, el agente “decide” girar en uno u otro sentido, y tal como se observa en el último condicional de la anterior gráfica, el agente despeja la pelota solamente si se encuentra orientado en una sección que barre, en sentido horario, las direcciones entre el Oriente hasta el Occidente, pasando por el Sur; nunca por el Norte ni sus aledaños.

En caso de que la bola se aleje nuevamente, por cualquier circunstancia del agente, entonces se tiene la sencilla instrucción que se muestra a continuación:

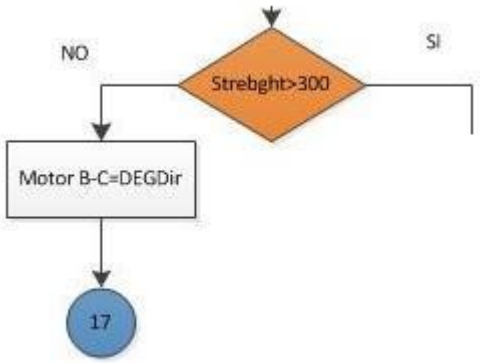


Figura 5.15

Lo que indica que sigue la pelota y retorna al inicio del programa.

El diagrama de flujo del algoritmo completo de reorientación, reducción de espacio y despeje es:

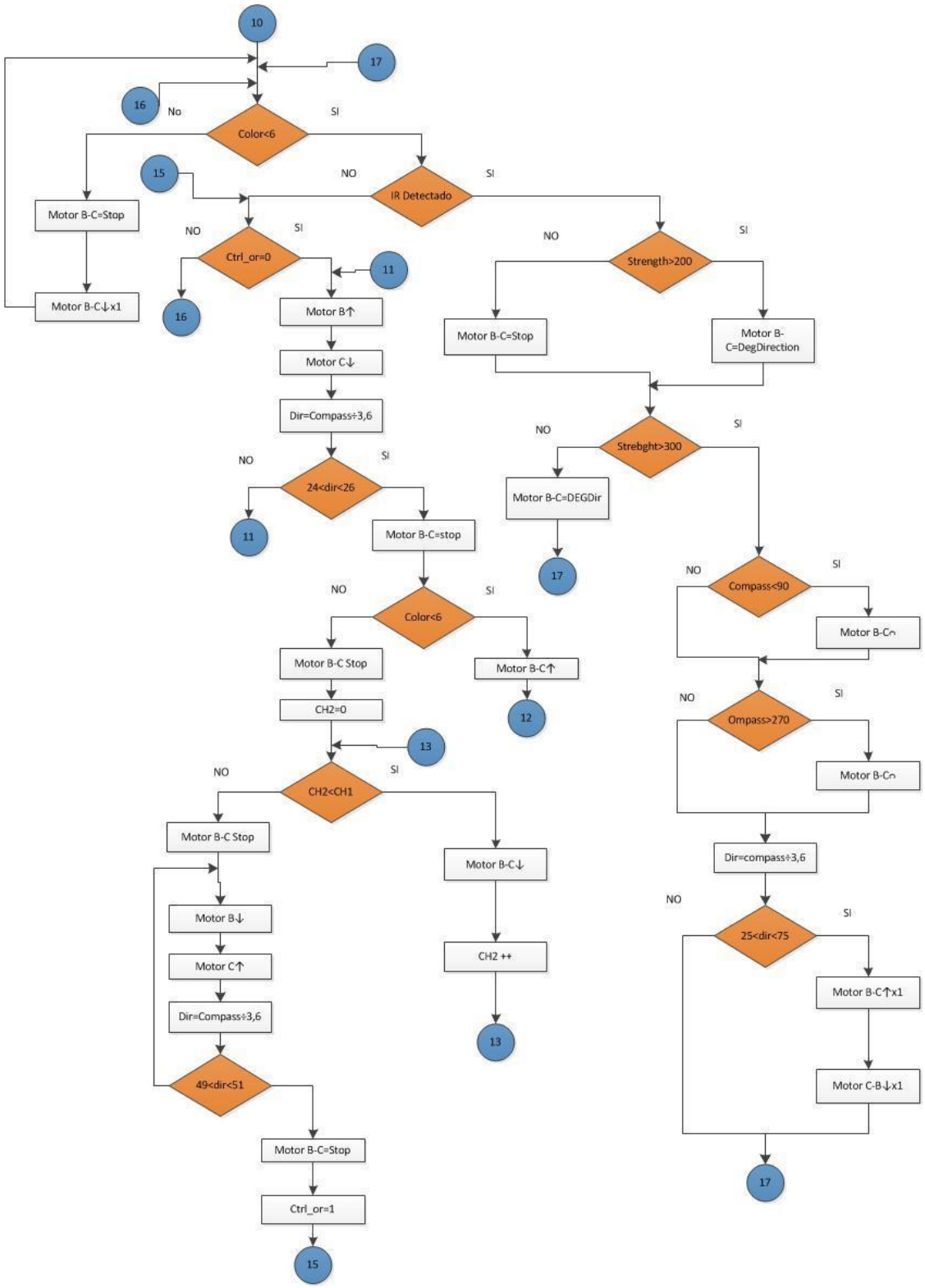

10
17
16
No
Color<6
SI
15
NO
IR Detectado
SI
Motor B-C=Stop
NO
Ctrl_or=0
SI
11
Motor B-C↓x1
16
Motor B↑
Motor C↓
Dir=Compass÷3,6
NO
24<dir<26
SI
11
Motor B-C=stop
NO
Color<6
SI
Motor B-C Stop
Motor B-C↑
CH2=0
12
13
NO
CH2<CH1
SI
Motor B-C Stop
Motor B-C↓
Motor B↓
CH2 ++
Motor C↑
13
Dir=Compass÷3,6
49<dir<51
Motor B-C=Stop
Ctrl_or=1
15
NO
Strength>200
SI
Motor B-C=Stop
Motor B-C=DegDirection
NO
Strebght>300
SI
Motor B-C=DEGDir
17
NO
Compass<90
SI
Motor B-C↷
NO
Ompass>270
SI
Motor B-C↷
Dir=compass÷3,6
NO
25<dir<75
SI
Motor B-C↑x1
Motor C-B↓x1
17

Figura 5.16

Recuérdese que este algoritmo complementa el diagrama completo que se mostró en la sección dedicada a la primera orientación y estos, están enlazados mediante el conector 10.

5.2 Algoritmo de comportamiento de Defensa

Algoritmos de comportamiento

Para realizar el algoritmo de comportamiento se tomó en cuenta tanto los movimientos requeridos, como las limitaciones del software utilizado , por lo tanto el algoritmo inicia con el encendido del robot, para que los sensores de este empiecen a trabajar , el primer sensor y para el caso del agente defensivo el más importante es el sensor de color , con el cual trabajaremos los comportamientos de este dependiendo del sector de la cancha en donde se encuentre, este sensor lee los colores en el siguiente orden: Negro, Azul, Verde, Amarillo, Rojo, Blanco en donde el color amarillo, Rojo, y Azul el agente no realiza ningún movimiento puesto que una cancha de futbol no posee estos colores, por lo tanto los que utilizaremos serán el blanco, el negro y por supuesto el verde, así:

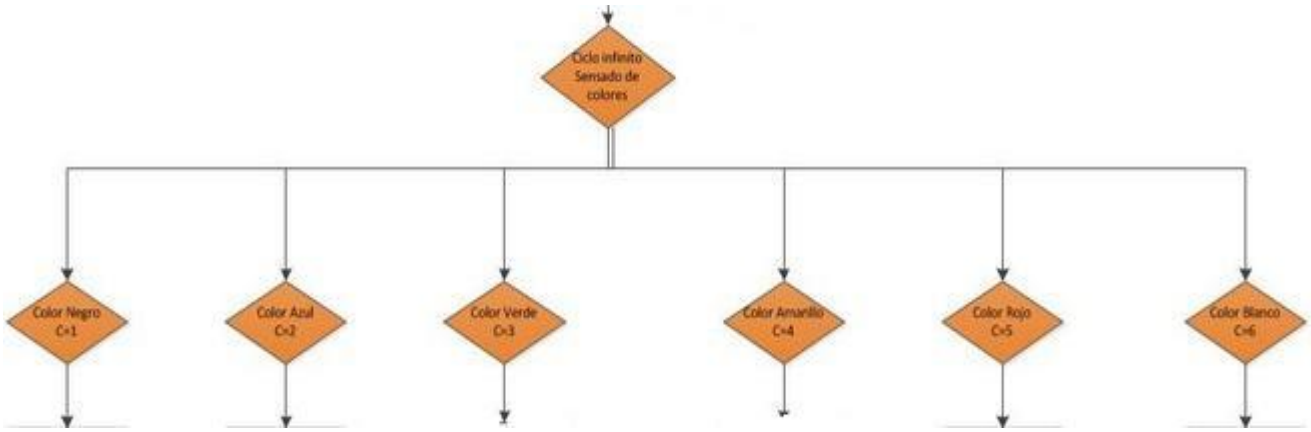


Figura 5.17

El caso del color Blanco Y Negro son los utilizados para delimitar el campo de acción del agente, por consiguiente, el blanco es el área de la bomba que es de este color , en este caso el agente gira 180 grados y vuelve a censar el color, esto es para apartarse del área del arquero ya que esta es su zona de acción y el color negro delimita la mitad de la cancha (colores que se tomaron para evitar confusiones con la bomba central) en este caso el agente lo único que realizará es devolverse.

Figura 5.18

La parte principal del programa se encuentra en el color verde puesto que es el mismo que el color de la cancha, por consiguiente, cuando el sensor detecta el verde se activa el IRseeker y calculara la distancia de la pelota si esta se prendida y en el rango de acción, por lo tanto si estas condiciones se cumplen, avanzará hasta la ubicación de la pelota.

Figura 5.19

Cuando el agente tiene ya localizada la pelota medirá de nuevo la distancia, con el fin de acercarse más a la bola y cuando está a una distancia mínima realizará la emulación de una patada o despeje del balón acelerándose un poco y empujando a la bola, movimiento que repetirá las veces que sea necesario para el despeje que se detendrá cuando la bola salga de su campo de acción o el rango de censado, todo esto sucede cuando la bola está en su campo de acción o esta prendida la pelota, de lo contrario realizará el proceso de ubicación

Figura 5.20

En este caso se activa la brújula que es la encargada del proceso de ubicación, con la cual nos dará el valor en grados de su posición, si no está ubicado el agente hacia el sur este girara hasta localizar la ubicación requerida y se quedara quieto esperando por la pelota puesto que es un defensa líbero

En donde todo el diagrama de flujo de comportamiento de un agente en posición defensiva quedaría de la siguiente manera:

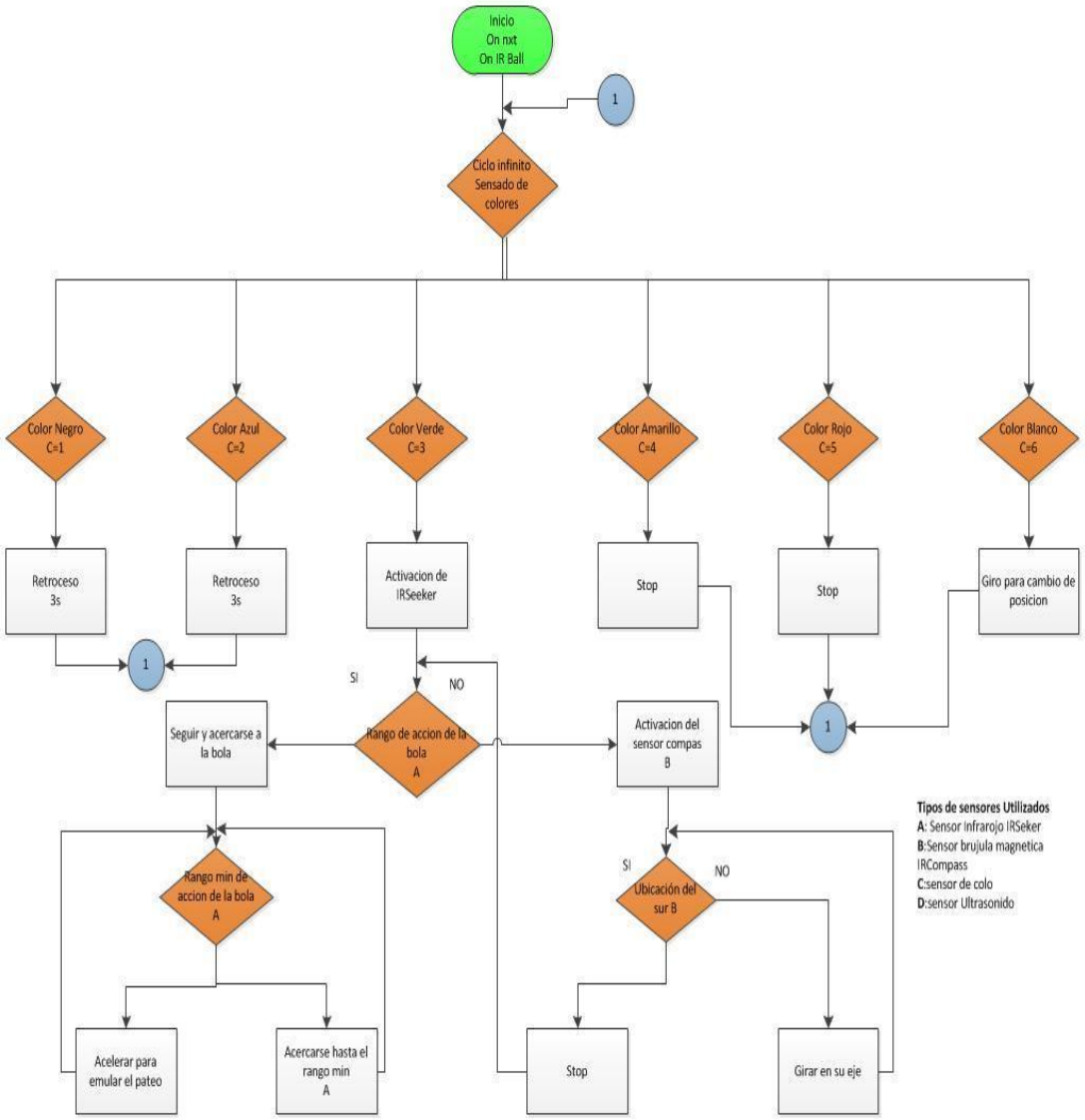


Figura 5.21

5.3 Algoritmos de comportamiento de un delantero

Antes de implementar los algoritmos hay que tener en cuenta las entradas a usar, como lo son los sensores y las variables para el manejo de la programación, además hay que considerarlas con respecto a las salidas que en este caso son los motores.

Entradas

- Compass = brújula: sensor de orientación para el agente
- Ir Seeker = infrarrojo: sensor de búsqueda de la pelota
- Ultrasonido = ultrasonido: sensor que determina distancias

Variables

- Number1 = Variable que tomará todos los valores del compass
- Ir Direction = Variable que arroja el Ir Seeker para determinar la posición de la pelota
- Strenght = Variable que arroja el Ir Seeker de intensidad de señal de la pelota.

Salidas

- Motor = Ejecución de patada al balón
- Motor B = Control rotacional del agente
- Motor C = Control rotacional del agente

Programación

Inicialmente se tiene en cuenta la orientación del robot por ende la primer ejecución a realizar es que encuentre el arco rival o en su defecto que mire al norte que es donde estará ubicado el arco contrario. Para esto el norte en grados en cero, entonces utilizamos un bucle infinito para que ejecute una acción hasta que encuentre el compass en 0 así:

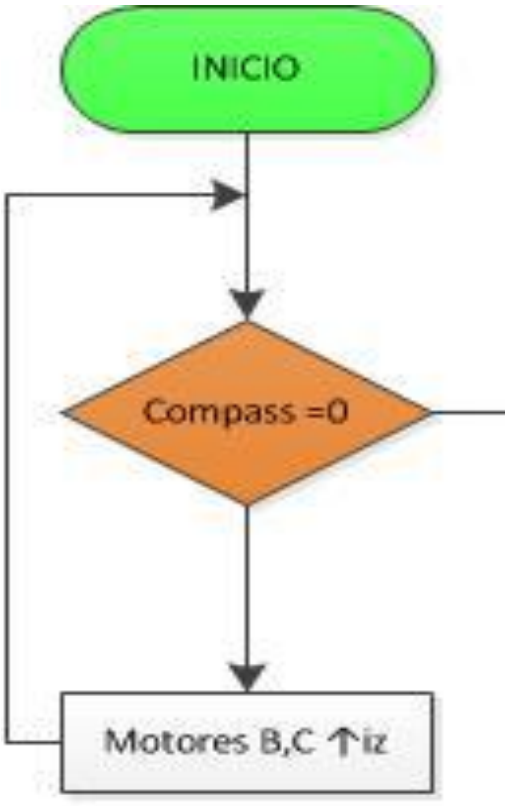


Figura 5.22

Como se puede apreciar en la figura anterior se genera un inicio, luego un ciclo infinito donde los motores B y C rotarán hacia la izquierda hasta que la entrada compass sea igual a 0, cuando cumpla esta condición saldrá de este ciclo y ejecutará la siguiente acción.

La siguiente acción sería no ejecutar acciones, con el fin de que el agente pueda reconocer su entorno, dado esto le decimos que pare los motores B y C y vaya a la siguiente acción.

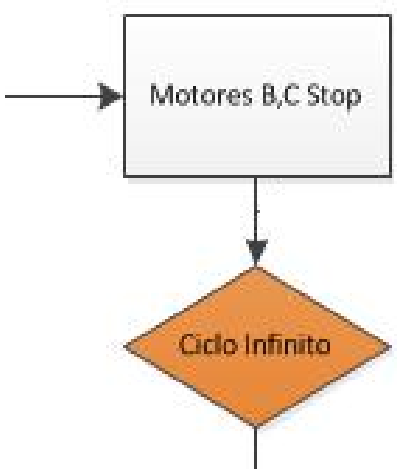


Figura 5.23

Como se puede apreciar en la figura después de haber salido del ciclo infinito de condición 0, pasa a parar los motores y procede al ciclo infinito, el cual no tiene fin; este se ejecutará de forma indeterminada.

Al empezar el nuevo ciclo afirmamos que Number1 va a ser igual a compass, luego si compass está es menor de 75 y mayor 285 pase al leer el Ir Seeker y revise si es igual a 1. Si compass no está en el rango estipulado, pasara a tomar otra decisión, que tiene en cuenta la variable Number1, preguntando si Number1 es mayor a 180 mueva los motores B y C hacia adelante a la izquierda, de lo contrario mueva los motores B y C hacia atrás a la derecha; esto con el fin de ubicar al robot hacia el norte y que él decida por donde le queda más rápido el giro hacia el norte.

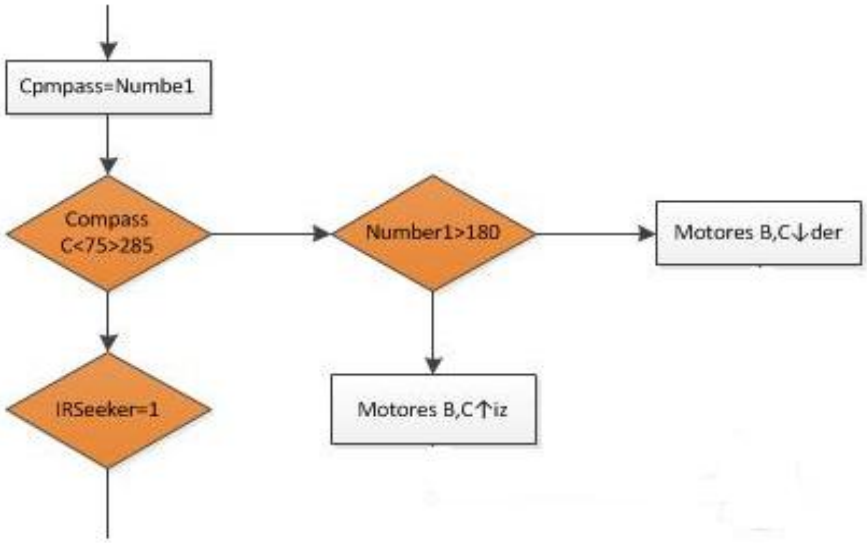


Figura 5.24

En la bifurcación del IrSeeker se tiene en cuenta si este es igual a 1 de lo contrario no importa, solo tiene en cuenta este número: el cual indica si el robot lee la pelota, al leerla pasa a la siguiente ejecución:

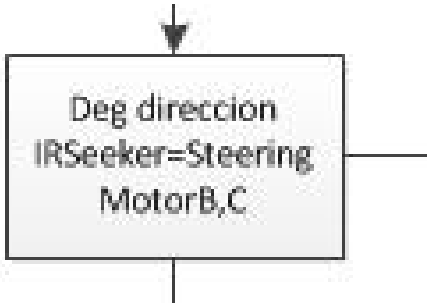


Figura 5.25

Esta ejecución indica que la variable Deg direction arrojada por el Ir seeker, le inclina al robot la ubicación de la pelota, al igualarla con el steering de los motores B y C, que es por decirlo así el conductor del robot generara que él estos sigan la pelota.

De esta ejecución salen dos bifurcaciones las cuales una comprende el ultrasonido y otra las variables dadas por IrSeeker.

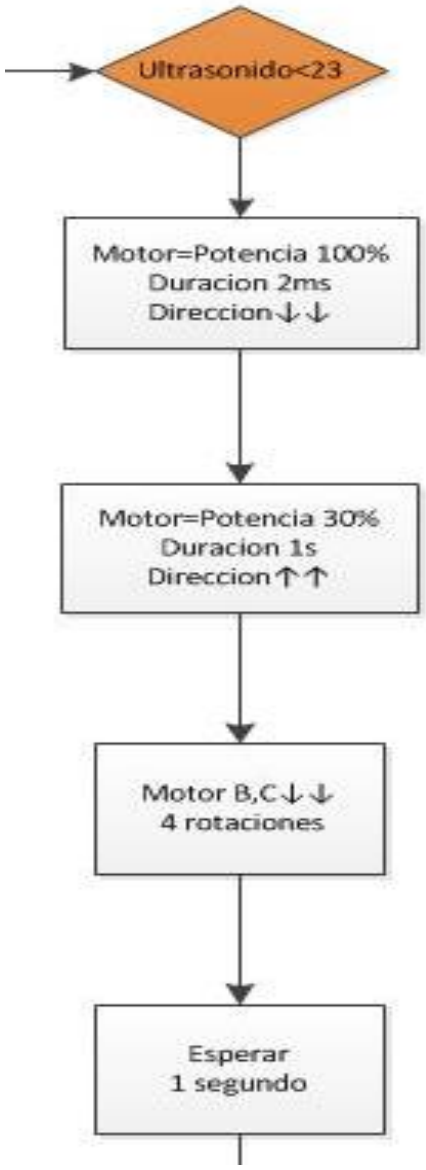


Figura 5.26

La bifurcación del ultrasonido como se puede apreciar en la figura anterior, da la condición de menor a 23, que se refiere a 23 centímetros, esto con el fin de que si encuentra un obstáculo en el camino y lleva la pelota como el agente no tiene más sensores para determinar qué camino coger, el patea la pelota y corre hacia atrás 4 rotaciones de sus

motores B y C, esperando un segundo, a ver dónde coge la pelota y volver a seguirla. En el caso contrario de no tener obstáculos a menos de 23 centímetros la bifurcación no toma ninguna acción y continúa con el programa.

Figura 5.27

Cabe aclarar las ejecuciones del motor de patada que lo llamamos en este caso la salida "Motor", quien en esta ejecución, se debió medir la velocidad de patada y retroceso según la potencia, para ese caso se configura el motor a un 100% de potencia, con una duración de giro de 2 milisegundos, hacia adelante.

Figura 5.28

Y para devolver la patada también se calculó el regreso de modo que al devolverse no haga torque el motor con el chasis del robot. En este caso se dijo que Motor es igual a potencia al 30% y una duración de un segundo.

Sin olvidar que de la ejecución dada por el Deg Direction, salían dos bifurcaciones ahora vamos con la bifurcación de las variables entregadas por el Ir seeker

Figura 5.29

Ahora tenemos que dos variables que son IrDirection y Strenght, la condición dice que si IrDirection es mayor a 6 y Strenght a 232 ejecute las siguientes acciones, que en este caso son las de patear el balón, hay que tener en cuenta que el solo patea si se cumplen las dos condiciones. ¿En qué se diferencia esta patada de la patada que genera el sensor de ultrasonido? Sencillo hay que tener en cuenta que la lectura que genera el IrSeeker es

específica y las condiciones también son específicas así que cuando la pelota está en una posición acorde a la forma de la patada del robot el pateara, esto para garantizar que el balón tendrá una potencia y colocación exacta hacia el arco rival.

Figura 5.30

En algunas partes del algoritmo se puede visualizar este símbolo el cual determina la continuación de un proceso.

En la página siguiente esta la visualización completa del algoritmo para tener una mejor comprensión del proceso.

Algoritmo completo

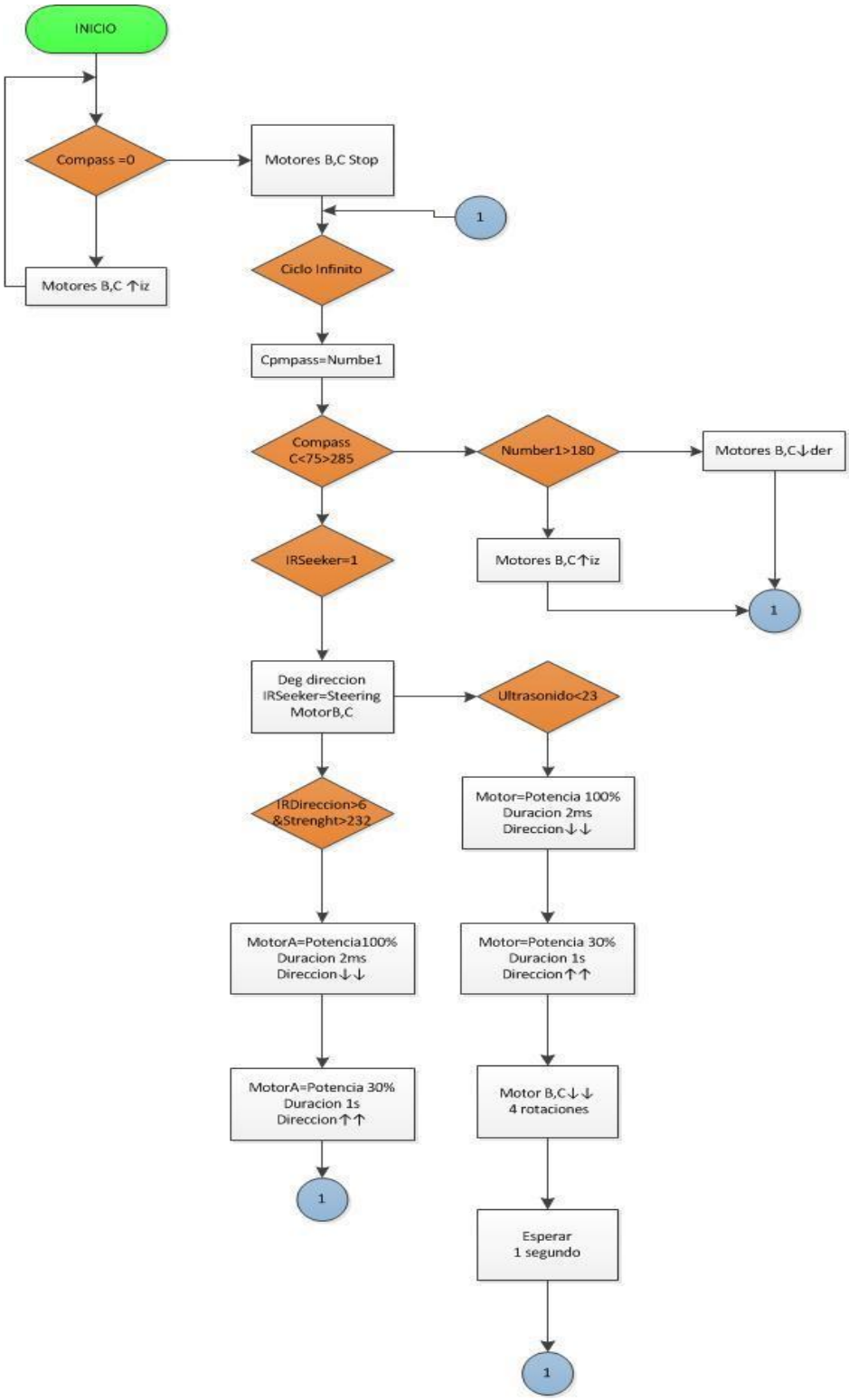

INICIO
Compass =0
Motores B,C Stop
1
Motores B,C ↑iz
Ciclo Infinito
Cpmpass=Numbe1
Compass
C<75>285
Number1>180
Motores B,C↓der
IRSeeker=1
Motores B,C↑iz
1
Deg direccion
IRSeeker=Steering
MotorB,C
Ultrasonido<23
IRDireccion>6
&Strenght>232
Motor=Potencia 100%
Duracion 2ms
Direccion↓↓
MotorA=Potencia100%
Duracion 2ms
Direccion↓↓
Motor=Potencia 30%
Duracion 1s
Direccion↑↑
MotorA=Potencia 30%
Duracion 1s
Direccion↑↑
Motor B,C↓↓
4 rotaciones
1
Esperar
1 segundo
1

Figura 5.31

CAPÍTULO 6

IMPLEMENTACIÓN DE ALGORITMOS EN LA PLATAFORMA LEGO

6.1 Arquero

A continuación se muestra la implementación del código mostrado en el capítulo anterior sobre la plataforma seleccionada, debido a que se ha explicado exhaustivamente la función de cada bloque utilizado en el software Lego NXT, no ahondaremos en este aspecto, concentrándonos solamente en su configuración de acuerdo al diagrama de flujo correspondiente en el capítulo anterior.

6.1.1 Ubicación Inicial

Inicialmente el arquero busca ubicarse cerca al centro del arco y orientado hacia el sur, lo cual se logra mediante la siguiente configuración de bloques:

Iniciando desde el ciclo más externo hacia los ciclos internos tenemos:

6.1.1.1Orientación final hacia el sur: Este es el ciclo más externo del algoritmo de ubicación el cual viene limitado por:

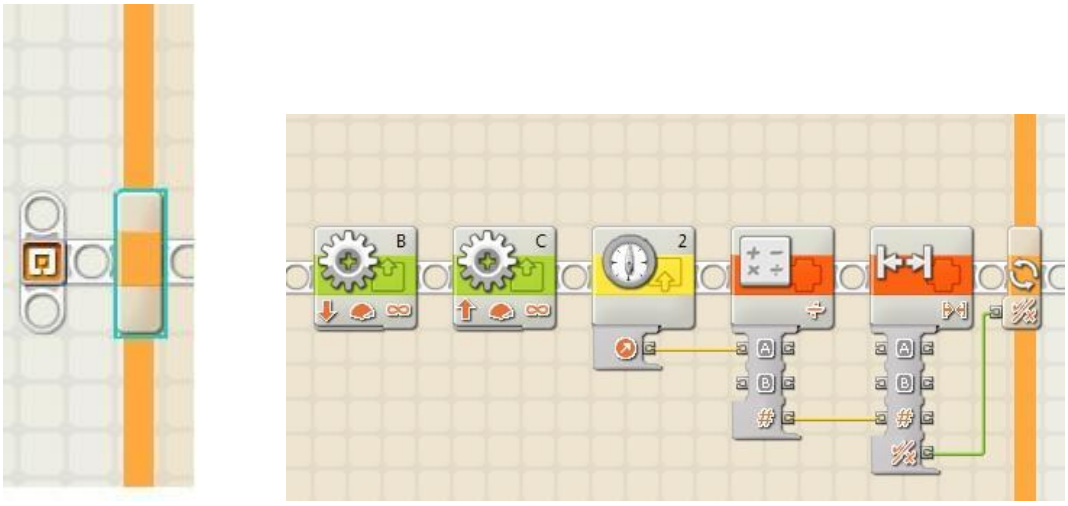


Figura 6.1

El cual siempre se ejecutará siempre y cuando:

Figura 6.2

Pero como se explicó en capítulo anterior, el rango que nos da el compass sensor, es de escala 0-359 mientras que el bloque range nos obliga a tener una escala 0-100, por lo cual se hace necesario aplicar la ecuación lineal para la conversión de escalas:

$$compass = 3{,}6 * range$$

Despejando range:

$$range = \frac{compass}{3{,}6}$$

Ya que cada unidad del range representa 3,6° de cambio en el compass sensor. Por lo tanto nuestro interés es que el range se encuentre entre 49 y 51 lo que representa una orientación deseada entre 176,4° y 183,6°. Mientras este ciclo se ejecute, sin pasar por sus otros ciclos internos, como se verá más adelante, se produce un sentido de giro inverso de un motor con respecto al otro provocando el giro en búsqueda de la orientación deseada.

6.1.1.2 Control para ubicación en el centro del área

Este proceso se realiza mediante una variable de control ctrl_1 de tipo número, la cual se inicializa en 0, valor para el cual se ingresa al ciclo verdadero del switch mostrado comparándose en el bloque comparación del software:

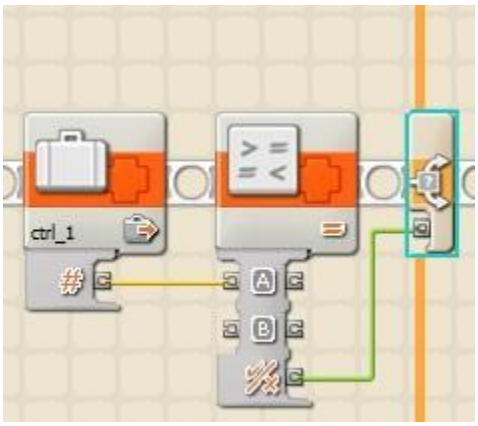


Figura 6.3

En caso de que ctrl_1 sea mayor a 0, el ciclo negativo está vacío dejando que el agente realice cualquier acción que venga haciendo con anterioridad.

6.1.1.3 Control del valor de ctrl_1

Para que se obtenga un control total de la ejecución del switch del numeral anterior se debe poder manipular el valor de ctrl_1 desde adentro de las instrucciones positivas del mismo switch, ya que debemos recordar que este proceso de ubicación recorrerá varias veces esta comparación debido a que se encuentra dentro del loop explicado en 6.1.1.1:

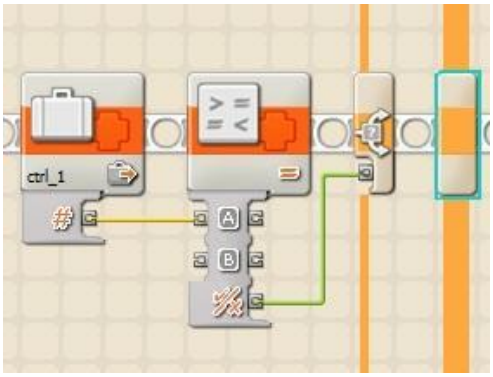


Figura 6.3

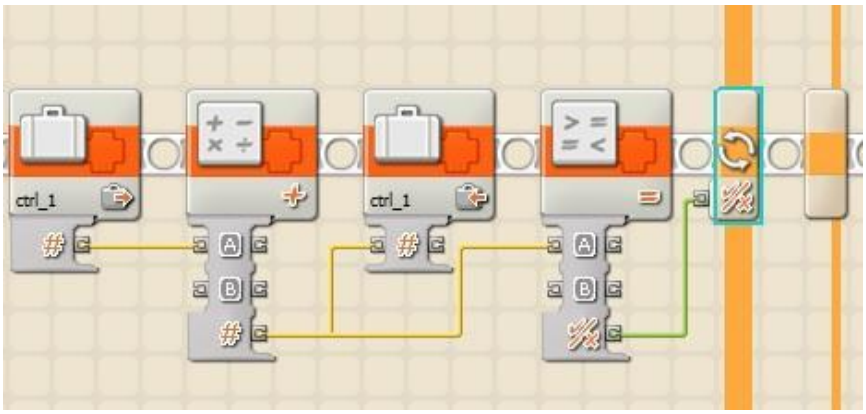


Figura 6.4

Donde se muestra que cuando se llega al final de este ciclo, la variable ctrl_1 se incrementará en 1 mediante el bloque de operación matemática:

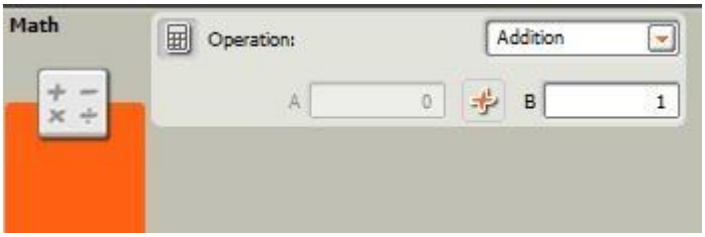


Figura 6.6

6.1.1.4 Giro al occidente (90°) - Proceso de ubicación en punto medio del área

Para iniciar el proceso de ubicación, el agente debe reconocer su espacio, por lo que se propone realizar un giro buscando el occidente (si aún no está orientado hacia tal punto), para avanzar hacia la línea límite de esa dirección. Esto se logra de la siguiente forma:

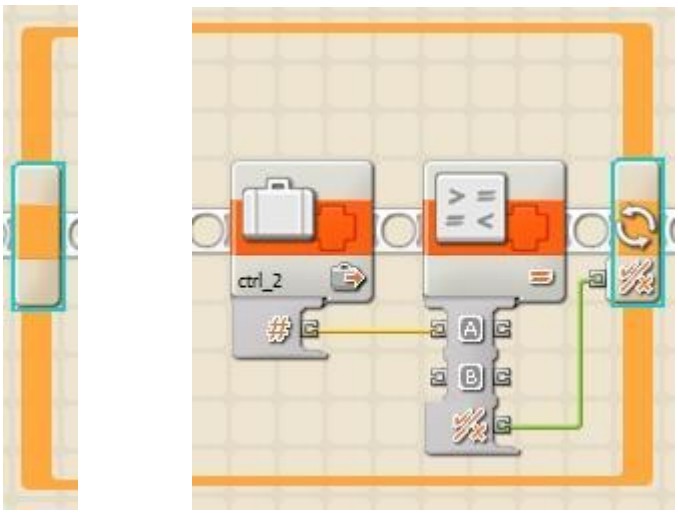


Figura 6.7

Se tiene una variable de tipo número llamada ctrl_2 la cual está inicializada en 0 tal como se especificó en el diagrama de flujo de arquero, luego esta se compara en el bloque compare quien arroja un valor lógico que controla la permanencia del programa en el ciclo loop, donde se comprueba que:

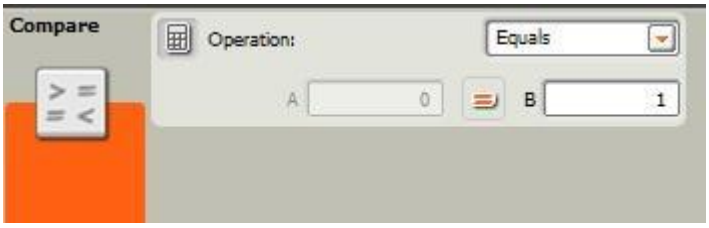


Figura 6.8

Mientras que esta condición sea falsa, el ciclo se repite.

6.1.1.5 Control de giro hacia 90°

Para controlar el giro del agente se implementa el siguiente algoritmo:

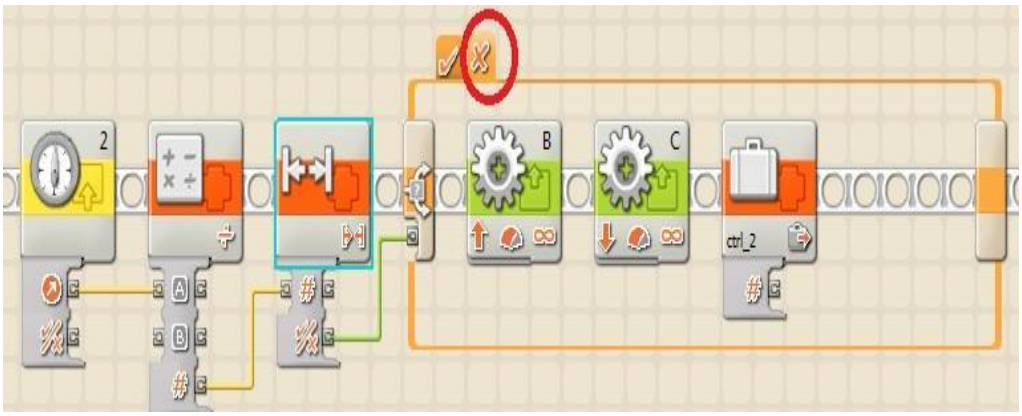


Figura 6.9

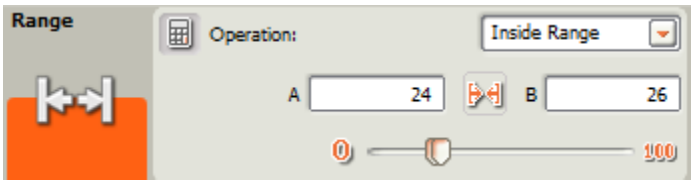


Figura 6.10

Que indica que el rango deseado debe estar entre 24 y 26, equivalente a un intervalo de (86,4° , 93,6°), mientras esto NO OCURRA se ejecuta la acción falsa que gira los motores en sentidos contrarios para buscar compass=90° y mantiene la variable ctrl_2 en cero. Por el contrario si este cumple la condición de estar dentro del rango:

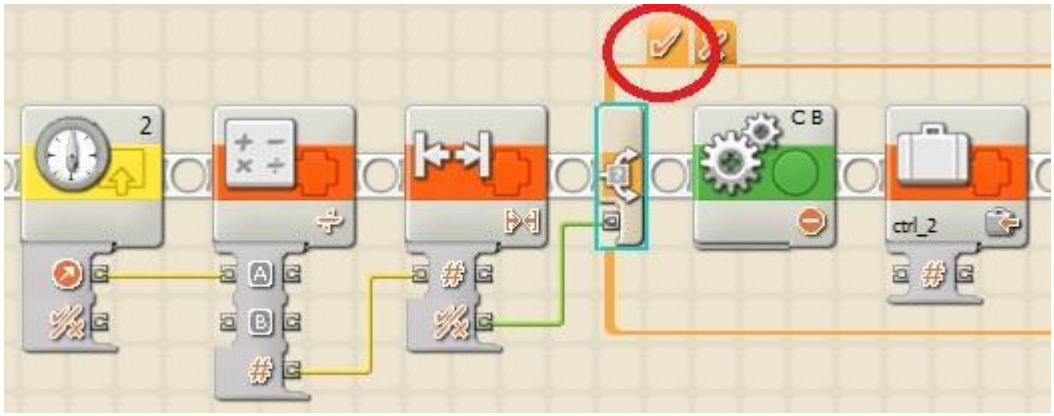


Figura 6.11

Se provoca la detención de los motores y la variable ctrl_2 se le asigna un valor de 1, lo que provoca la salida posterior del ciclo loop.

6.1.1.6 Control de desplazamiento horizontal total (CH)

En este momento se utiliza la variable CH, inicializada en cero al igual que las anteriores, ingresando a un loop de control de repetición de eventos:

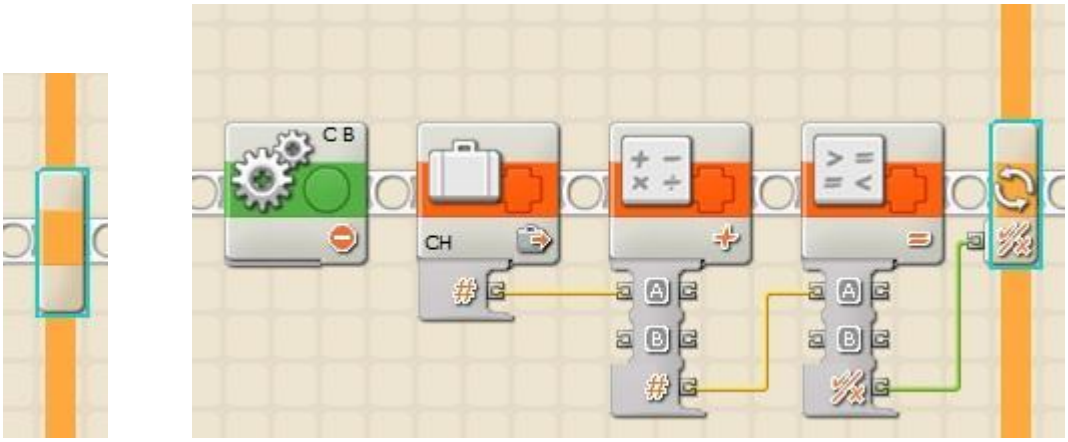


Figura 6.12

Este ciclo termina cuando la condición CH=0 NO se cumple, además de provocar un frenado de los motores. En este caso CH se incrementa en 1 antes de ser evaluada nuevamente. Es de resaltar que esto no se cumplirá inmediatamente al ejecutarse el programa ya que dentro de este ciclo existen otros que se repiten antes de que se pueda llegar al final anteriormente descrito.

6.1.1.7 Control de desplazamiento hacia occidente (CH3)

Ingresado dentro del ciclo de control de CH, entramos a un nuevo loop que se encarga de dirigir el agente hasta la línea límite en O=90°:

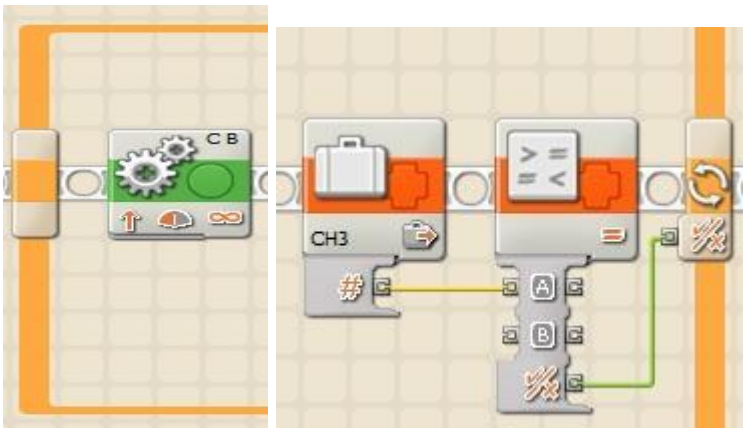


Figura 6.13

Este loop se ejecuta mientras la condición es Falsa comparando siempre en el bloque compare, si CH3=1, a su vez que se activan los motores para que el agente se desplace hacia adelante.

6.1.1.8 Control de frenado dentro del loop CH3

Debido a que ya el agente se encuentra avanzando se hace necesario controlar el momento de frenado mediante la siguiente configuración:

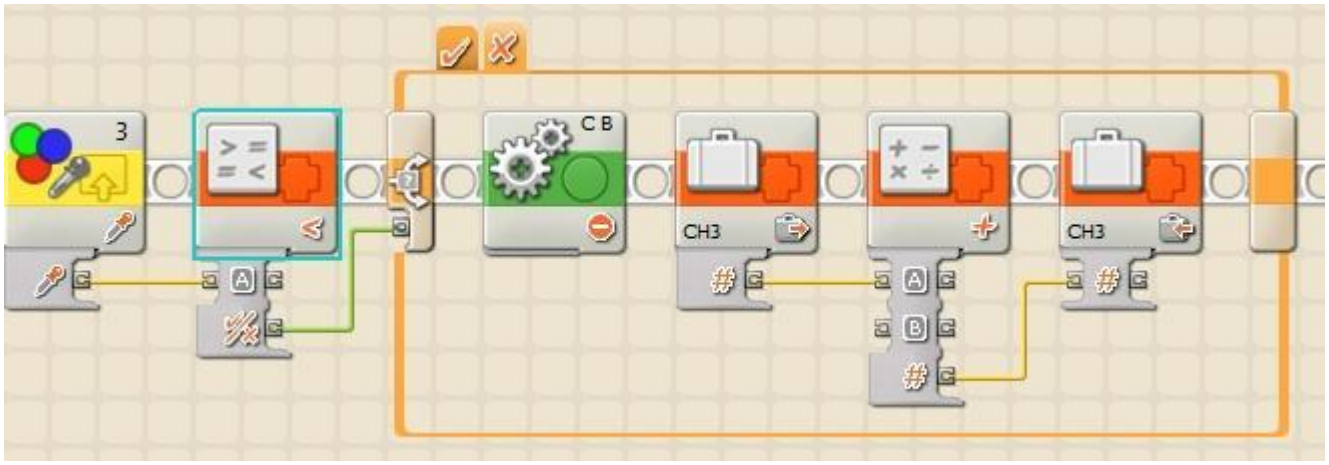


Figura 6.14

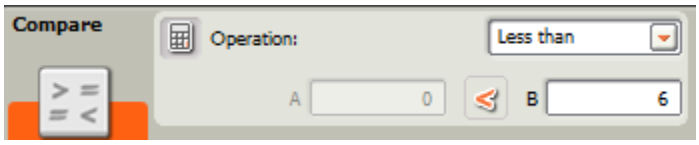


Figura 6.15

Aquí hacemos uso del sensor de color ya que cuando se detecte que el color es blanco (Color=6), este ejecuta la acción de frenado de los motores como se observa en la figura 8.14 e incrementa en 1 el valor de CH3, por lo tanto, si es verde el color el valor lógico del compare es verdadero, por lo tanto esta opción es vacía para que el motor siga girando de acuerdo a la figura 6.13

6.1.1.9 Búsqueda de límite Oriental. (CH1)

Estando ya ubicados en el límite occidental, nos desplazamos ahora hacia el límite oriental en reversa, contando el número de veces que se repite el ciclo loop que permite este desplazamiento y almacenando este valor en la variable CH1 de tipo número.

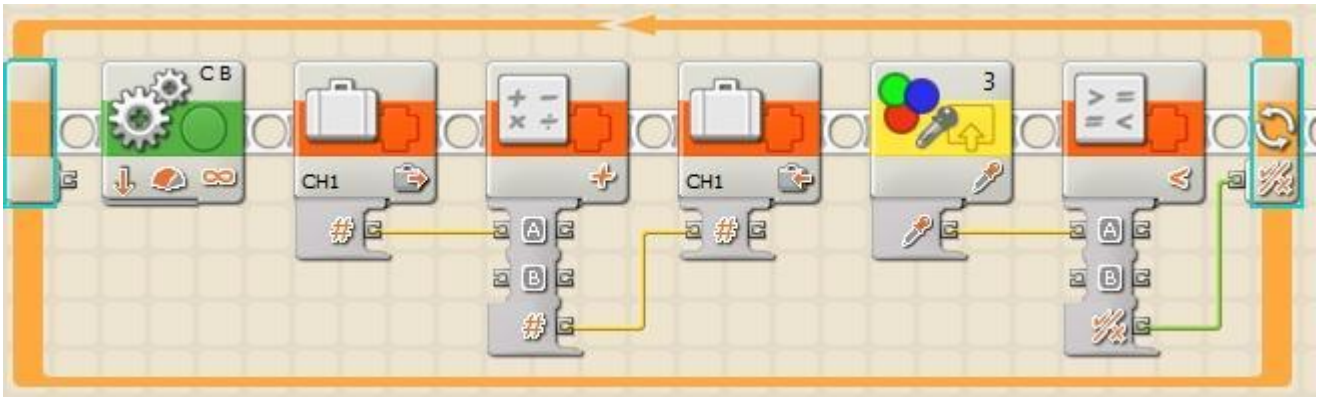


Figura 6.16

En 8.16 se muestra que los motores ahora giran en reversa, hasta que el sensor de color detecte nuevamente color blanco (valor 6) lo que provoca la salida de este ciclo; adicionalmente la variable CH1, quien inicialmente vale 0, se incrementa en 1 cada vez que se ejecute este ciclo.

6.1.1.10 Tratamiento de la variable CH1

Una vez terminado el ciclo de 8.1.1.9, el valor de CH1 habrá cambiado, como este valor depende del número de veces que se ejecute el ciclo anterior, entonces podemos deducir que si el agente se desplaza dentro de un ciclo con duración $\frac{CH1}{2}$veces, entonces el desplazamiento será cercano a la mitad del ocurrido en 6.1.1.9. Para lograr esto se opera con CH1 de la siguiente forma:

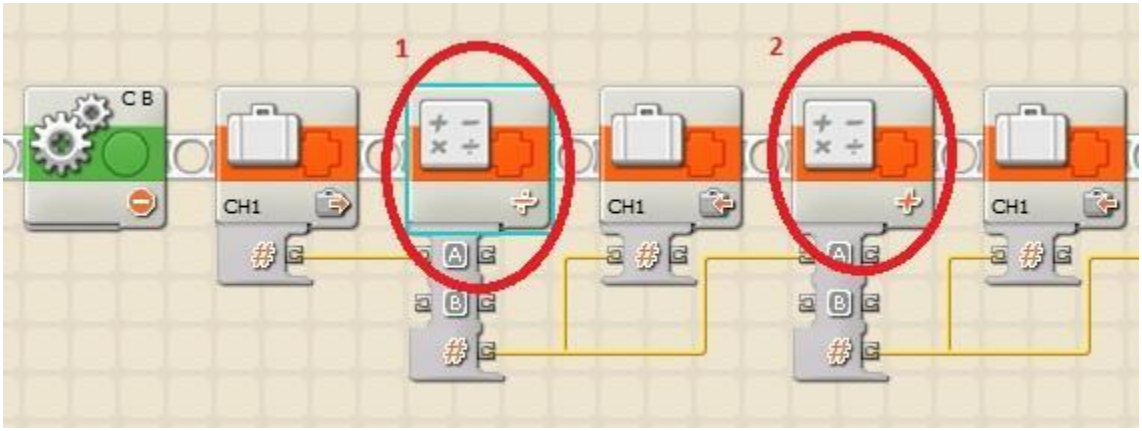


Figura 6.17

Donde en 1 se divide CH1 en 2 y en la operación 2 se adiciona un valor constante que permite corregir el error de ubicación o disminuirlo.

6.1.1.11 Desplazamiento hacia el centro del área (CH2)

Una vez tratada la variable CH1, es conveniente compararla con otra variable (CH2) la cual se va incrementando en 1 cada vez que se ejecuta un nuevo loop, que se encarga de desplazar nuevamente el agente hacia el occidente pero evitando que llegue hasta la frontera de esta orientación:

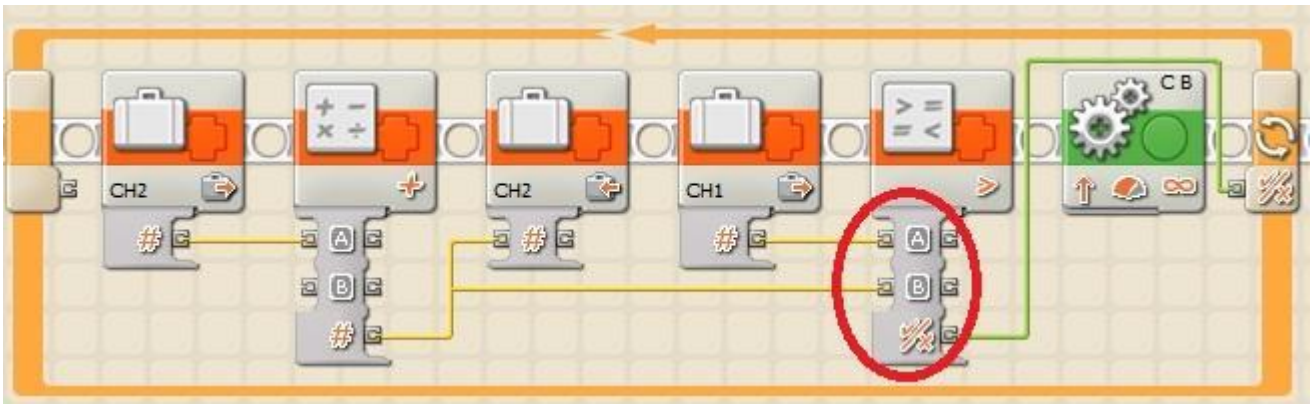


Figura 6.18

Nótese que a medida que se ejecuta este loop, se compara el valor de CH2 con respecto a CH1, y mientras este valor sea menor a CH1, el ciclo se repetirá, lo que garantiza que este se termina cuando ch2, variable de control de ejecución del loop, sea del valor de CH1.

6.1.1.12 Orientación final

Una vez el programa sale del loop 6.1.1.11, se detienen los motores:

Figura 6.18

Acción que precede el al incremento de la variable CH tal como se indicó en la sección 6.1.1.6, acto seguido se modifica el valor de ctrl_1, provocando el giro de búsqueda de O=180 culminando de esta manera la ubicación inicial.

6.1.2 Juego

Una vez culminado el proceso de ubicación y orientación inicial, se detienen los motores y se inicializa una variable llamada ctrl_or = 1.

6.1.2.1 Ingreso al ciclo infinito y loop de color

Ya en juego el programa entra en un ciclo infinito que permite que siempre esté vigilante ante cambios de estado de la pelota y color de fondo:

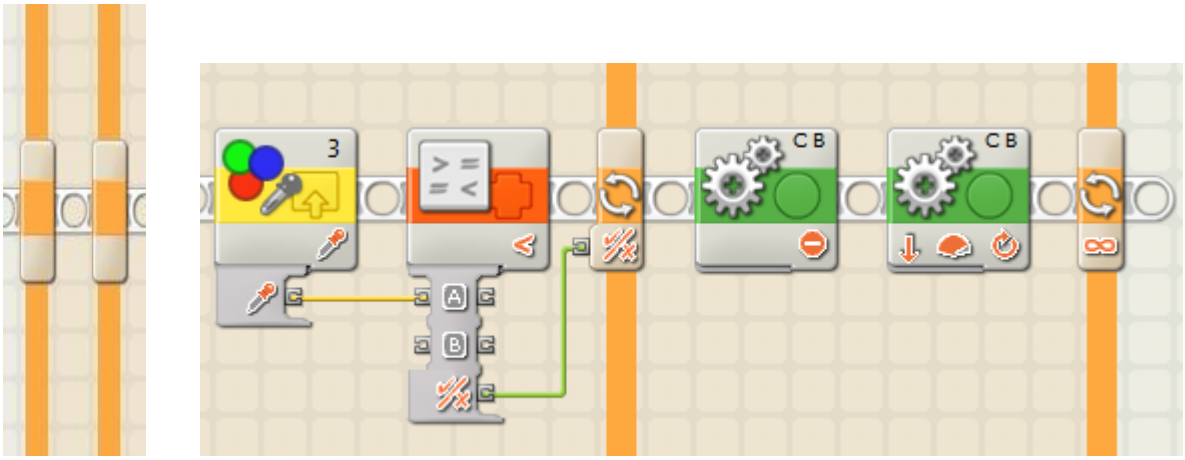


Figura 6.19

La figura muestra que si se incumple la condición de color <6 (que no sea blanco), el programa saldrá del loop color, frena los motores y se devuelve una rotación de motores.

6.1.2.2 Acciones dentro del área (Color <6)

6.1.2.2.1 En presencia de la bola IR

Siempre que exista presencia de la bola IR, es decir que el sensor IRSeeker este “True”, se producen las siguientes acciones:

6.1.2.2.1.1 Seguimiento de la bola IR (Achique e intercepción)

Cada vez que la bola se encuentra cerca o dentro del área, el agente busca acercarse a la misma para evitar así el disparo del oponente:

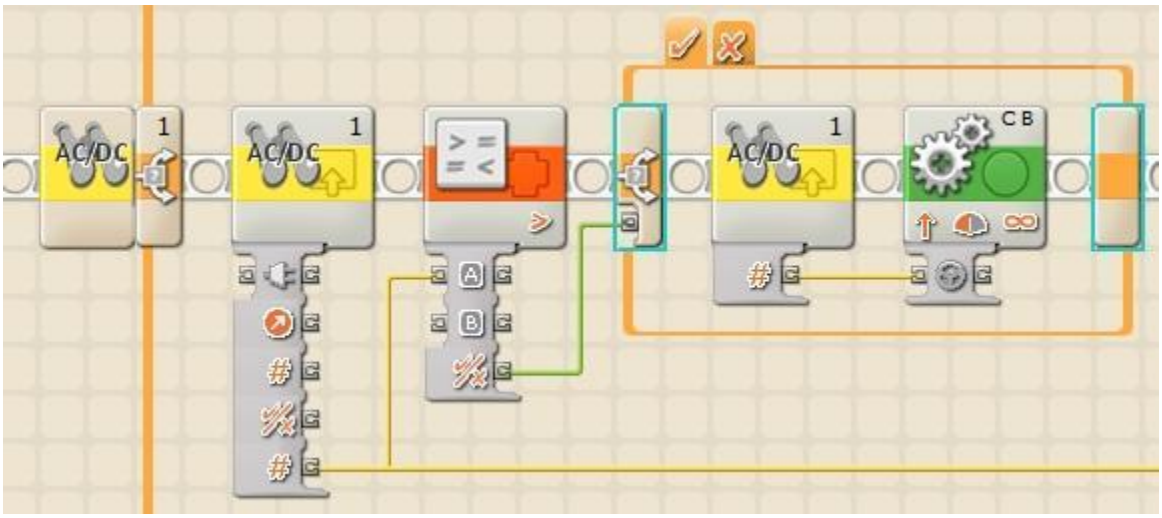


Figura 6.20

Aquí observamos que si una vez detectada la señal IR, este procede a medir su stregth, el cual, si es mayor a 200, es decir que la señal IR esté siendo generada desde una fuente menor a 20cm de distancia, el mismo sensor IR se encargará de manejar los motores de acuerdo a la posición de la bola para dirigirse hacia ella. De lo contrario los motores se frenan para evitar intentos de seguimiento por fuera del área y no recurrir a redundantes intentos fallidos por salir de allí cruzando la línea fronteriza.

6.1.2.2.1.2 Decisiones con posesión de la bola IR

Una vez cerca de la bola IR, se reescribe el valor de ctrl_or =0 e iniciamos un nuevo switch encargado del despeje de la bola:

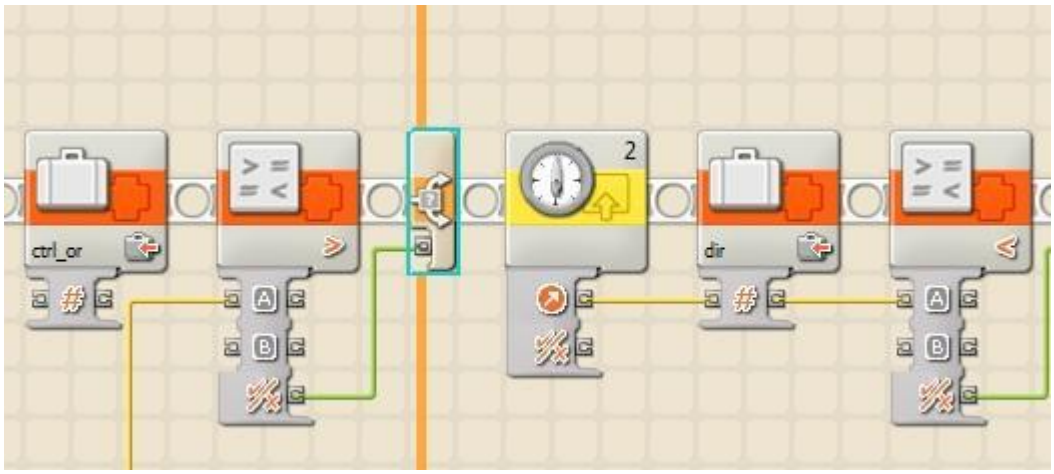


Figura 6.21

Dentro de este medimos la orientación del compass y según el valor arrojado tomamos las siguientes decisiones:

-Compass < 90 o compass>270:

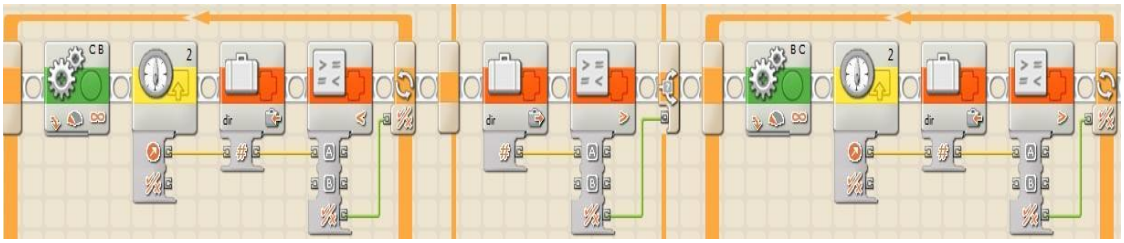


Figura 6.22

Se giran los motores en sentido horario o antihorario según respectivamente según sea el caso, hasta que el compass o arroje valores fuera del rango que no se desea, esto con el fin de evitar autogol.

- 90<compass<270

Una vez se encuentre el agente con orientación que puede ir desde el sur oriente al sur occidente, pasando por el sur, se puede despejar la bola sin riesgo de autogol:

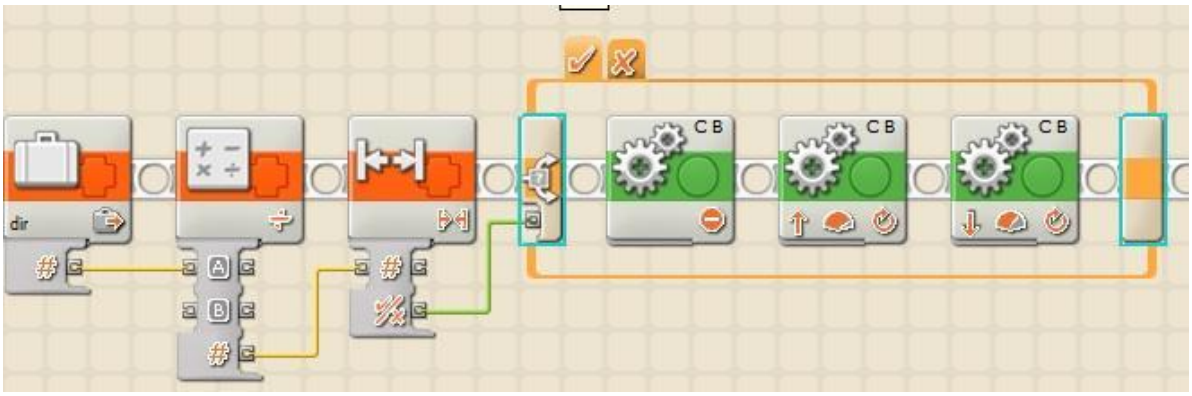


Figura 6.23

Figura que indica que la dirección debe encontrarse entre un rango de 25 a 75 que equivale a 90<dir<270 en grados, cumplido esto el motor se impulsa hacia adelante con toda la potencia durante un giro y luego se devuelve.

6.1.2.2.2 Decisiones en Ausencia de IR

En ausencia de señales IR, el agente se ubicará de nuevo en la posición dada por la sección 8.1.1 teniendo en cuenta que al tener ya el valor de CH1, entonces solo se procede a ejecutar nuevamente el segmento de algoritmo de la sección 6.1.1.11

6.2 Defensa

Para la implementación de los algoritmos de defensa en la plataforma LEGO MINDSTORM nos basamos en su programación por medio de bloques gráficos en donde la programación quedaría de la siguiente manera

Figura 6.24. *Ciclo infinito*

La programación empieza con este loop o ciclo infinito en donde el programa se repetirá las veces que sea necesario. La programación continúa con el sensor de color que se ve así

Figura 6.25. *Configuración sensor de color*

En la primera imagen vemos el símbolo del sensor en donde el número 1 es el puerto donde está conectado el 2 la acción que este está realizando (puesto que este sensor de color también puede ser sensor de luz) y el 3 es el concentrado de datos que lo podemos configurar en un panel de control que es el que vemos en la figura de arriba, en esta configuración se encuentra la opción de realizar una acción en cada color que el sense en el siguiente orden:

1 = Negro

2 = Azul

3 = Verde

4 = Amarillo

5 = Rojo

6 = Blanco

Teniendo esta referencia en cuenta pasamos al siguiente proceso que es una toma de decisión

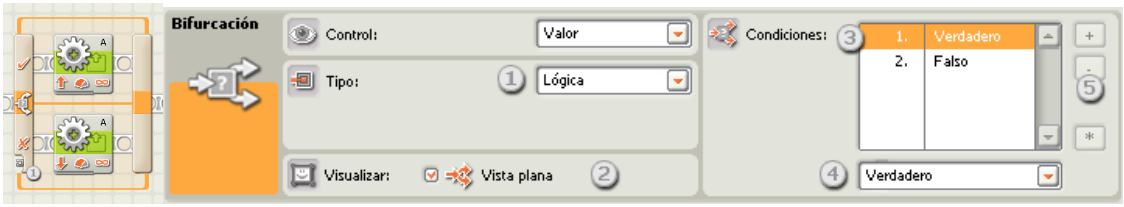


Figura 6.26. *Configuración bifurcación*

En el programa es llamado bifurcación en donde se toma una decisión lógica si/no o V/F entre otras, pero en nuestro caso este va a ser el encargado de la selección del color, por consiguiente el dato del color que le envía el sensor él lo va a asimilar como un proceso del programa. Teniendo en cuenta lo anteriormente mencionado empezaremos a describir el proceso en cada color, que en este caso en específico se utilizará el color VERDE, BLANCO, Y NEGRO. Tomamos estos colores puesto que el verde y el blanco son los colores básicos de una cancha de fútbol, y el color negro lo tomamos para delimitación de terreno de acción.

Empezamos con los procesos del color blanco y negro en donde el comportamiento del agente es retroceder en el caso del color negro, y en el caso del color blanco retroceder y girar.

Figura 6.27. *Bloque motores*

Este es el bloque que nos permite mover el agente, en este podemos ver los puertos de salida a los motores que queremos controlar en el número 1, en el número 2 muestra la dirección en la que se moverá uno o los dos motores, en el número 3 graduamos la intensidad o potencia con la que queremos que se mueva y en el número 4 se muestra la configuración de la propiedad de duración como ilimitada, grados, rotaciones o segundos.

Ahora bien, teniendo en cuenta los dos procesos anteriores que delimitan la zona del agente defensivo entramos al programa más complejo que es el de ubicación, control y dominio del balón, por lo tanto empezamos con el sensor IRSeeker que se ve de esta manera:

Figura 6.27. Bloque sensor infrarrojo

En donde este es el encargado de la localización de la pelota puesto que esta está enviando continuamente señales infrarrojas para ubicar su posición y la distancia a la que se encuentra el agente defensivo, por lo tanto después de tener el dato de la distancia de la pelota entrara en una toma de decisión que será: si la pelota se encuentra o no en el rango de acción si no está en el rango de acción realizará el siguiente proceso

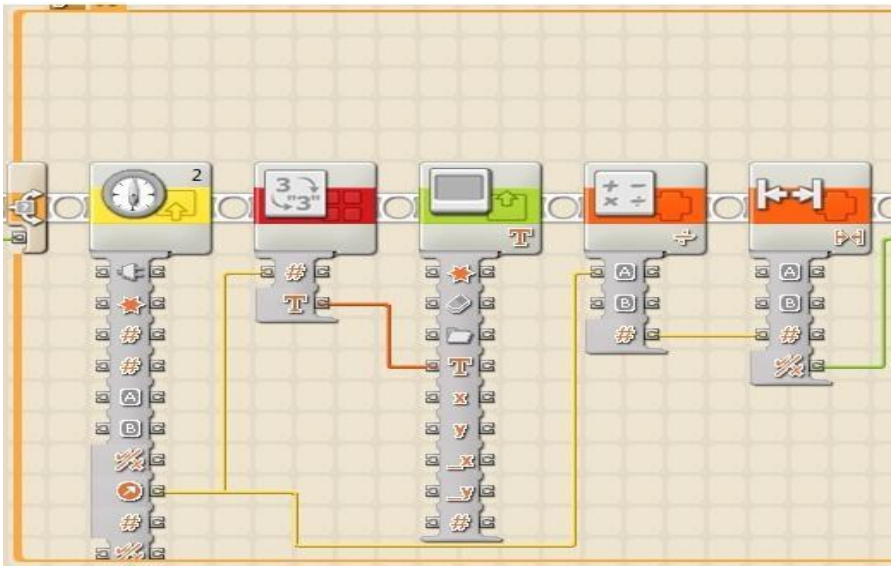


Figura 6.28.

En la imagen anterior podemos observar de color amarillo el sensor de compás en donde su panel de configuración se ve así:

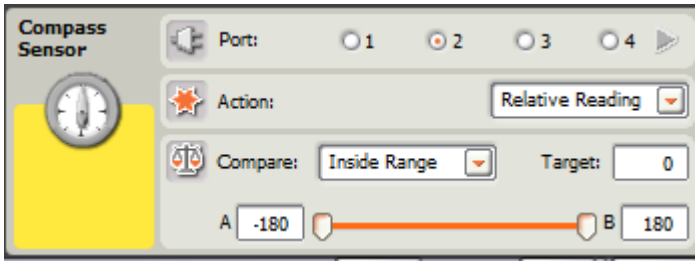


Figura 6.29. *Configuración compass*

Donde podemos observar en la primera fila el puerto en donde está conectada en la segunda fila la acción que realiza, que en este caso es una lectura relativa, y en la tercera fila parametrizamos esa lectura relativa, limitando el ángulo que se quiere leer.

Paso seguido está el bloque ¨Número a texto este bloque convertirá un número (por ejemplo, la lectura de un sensor) en texto, que puede visualizarse en la pantalla (Que en la figura es el bloque de color verde) para la toma de datos que utilizaremos más adelante.

A continuación va el bloque de matemáticas que se encargara de hacer una conversión matemática necesaria para nuestra toma de datos de ubicación.

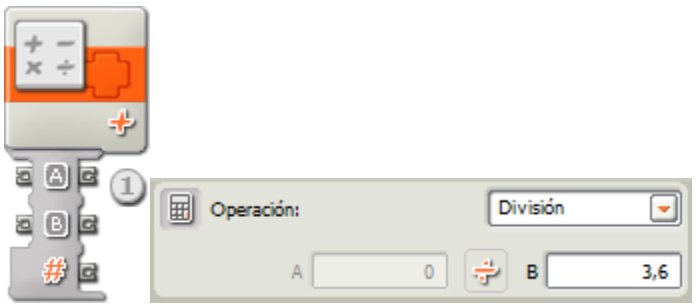


Figura 6.30. Configuración bloque matemático

En donde el dato que toma el sensor entra a este bloque por la entrada A y divide ese número por 3,6, operación que realizamos para que el parámetro que estamos evaluando este entre los límites adecuados para el bloque del intervalo.

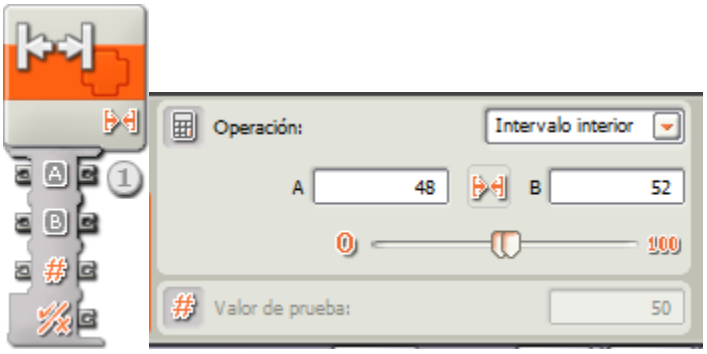


Figura 6.31.

Este bloque se encarga de determinar si un número se encuentra adentro o afuera de un intervalo de número, pero como vemos en la figura este rango va de 0 a 100 y el sensor de compás nos arroja un dato de -180 a 180 grados por lo tanto la operación matemática que realizamos es para convertir un dato de grados a un intervalo de 1 a 100, por eso en el bloque matemático dividimos la entrada A que son los grados, sobre 3,6 para que esté en el rango del intervalo.

Tomando como base la lectura del sensor de compás, en donde -180 es el norte y 180 es el sur, y tenemos que ubicar a nuestro agente mirando a sur parametrizamos nuestro intervalo de la siguiente manera:

Conversión de escala: 180÷3,6=50

A este resultado le agregamos una tolerancia de 2 puntos quedando un intervalo de 48 a 52 que en grados es 175 a 185 grados que es el rango para la ubicación del agente mirando al sur.

Una vez el sistema obtiene este dato entra en otra toma de decisión del paso que va a realizar a continuación que sería:

Si está en el rango del intervalo: quedarse quieto se vería así

Figura 6.32.

Si no está en el rango del intervalo: girar sobre su eje (cada motor en un sentido) hasta quedar en el rango delimitado y se vería

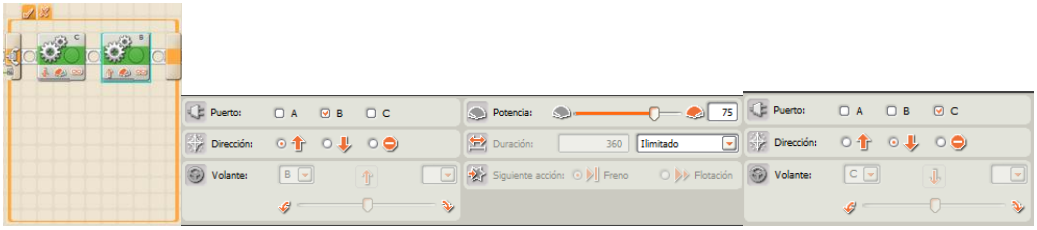


Figura 6.33.

esta parte de la programación se realiza cuando el robot no ha detectado la pelota en caso contrario, o sea, que detecte la pelota, lo primero que realizará es empezar a censar la distancia a la que se encuentra la pelota, y así poder realizar una comparación para realizar la toma de la siguientes decisiones

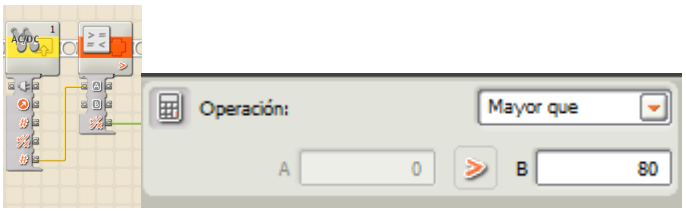


Figura 6.34.

Para delimitar el campo de acción del agente tomamos como lime 80 dato tomado por el mismo sensor que nos arroja estos datos por medio del display, por lo tanto si no se encuentra en el rango de la distancia el agente se quedara quieto. Pero si la pelota está en el rango, está la seguirá hasta tenerla a una distancia mínima y seguirá con el dominio de la pelota hasta que estas condiciones cambien, esto en el programa se vería de la siguiente manera

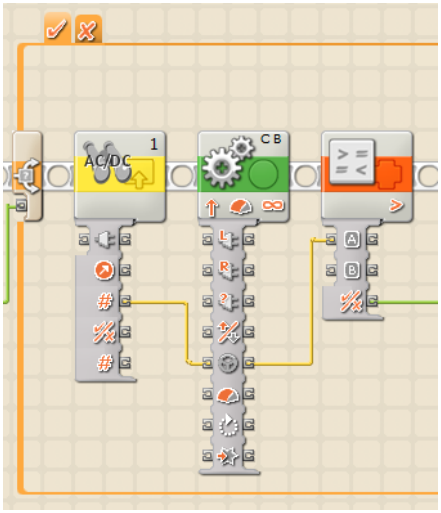


Figura 6.35.

Teniendo en cuenta que el bloque amarillo es el sensor IR Seeker, el bloque verde, el movimiento de los motores y el bloque rojo el de comparación para limitar su distancia mínima.

6.3 Delantero

La lógica de este agente parte de estar en dos ciclos loop uno lógico y otro infinito.

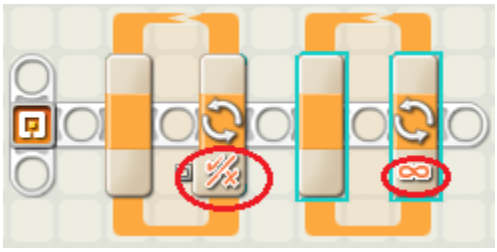


Figura 6.36

El lógico va a cumplir la función de ubicar al agente cuando inicia el partido y el infinito se encargara de las acciones del jugador durante todo el partido.

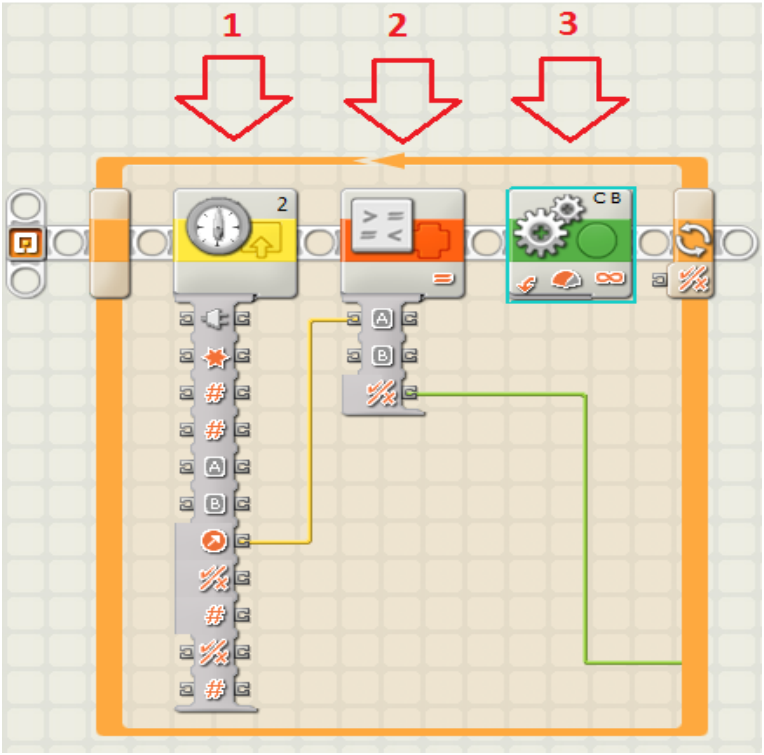


Figura 6.37

Como se puede ver en la gráfica anterior se encuentra la acción completa del Ciclo lógico.

La configuración de cada bloque está indicada por la numeración superior de la imagen de 1 a 3

1. Bloque Compass sensor.

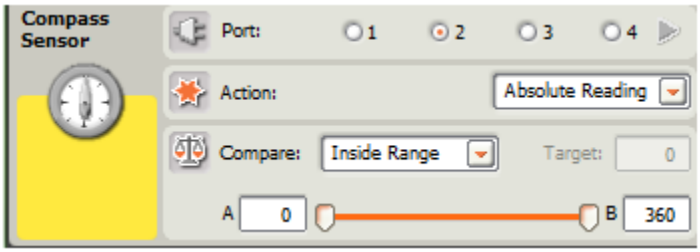


Figura 6.38

Se le indica al bloque el puerto de entrada donde se conectará el sensor, en nuestro caso el puerto 2. Este bloque en el modo de acción está ubicado en absolute reading, con el fin de que tenga una lectura completa de 0 a 360 grados, en la forma de comparación está situado en Inside Range, con el fin que tenga en cuenta la lectura completa de 0 a 360 grados, aunque este también cuenta con la lectura de rango exterior en el caso de tener en cuenta sólo números por fuera de un rango.

2. Bloque de comparación.

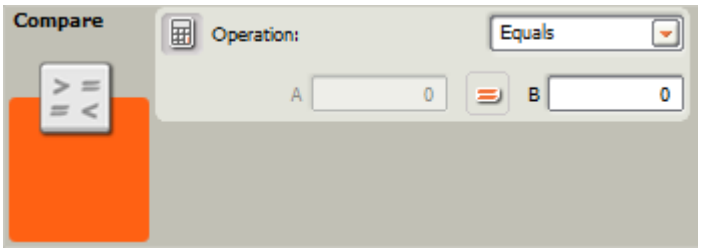


Figura 6.39

Teniendo en cuenta el bloque anterior que tiene en cuenta los números de 0 a 360, este va a tomar solo uno, en este caso el cero (0), que según la brújula magnética es el norte absoluto y es donde está ubicado el arco rival. Por ende el recibe todos los números que

vienen de la brújula magnética por el puerto A y compara a ver cuál es igual a cero (0), en caso de ser verdadera la afirmación se saldrá del loop lógico, de lo contrario seguirá en este.

3. Bloque Giro motor B y C

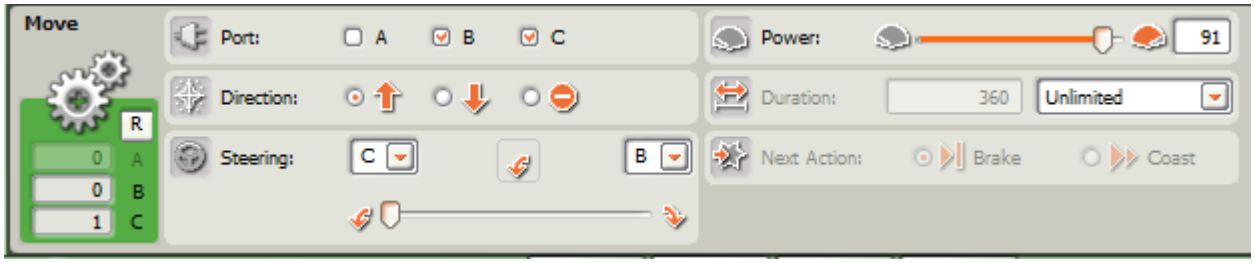


Figura 6.40

El giro del motor se efectuará siempre que la comparación anterior no sea afirmativa de la siguiente manera: Primero se denominan los puertos de salida por los cuales se conectaran los motores, se indica la dirección y el sentido, en este caso el giro se efectuará a la izquierda, con un power (potencia) de 91, se aplica esta potencia para que este movimiento sea muy rápido y poco notorio. En el caso de la duración hay 4 opciones ilimitado, grados, segundos y rotaciones, se utilizó la ilimitada ya que en grados no sabemos en qué posición será ubicado el agente por ende no se sabrá el número exacto de grados que tendrá que girar ya que este siempre tendrá que varias, y en el caso de rotaciones de rotaciones y segundos los dos efectúan sus acciones sin importar si hay una condición durante su proceso por ende seguirían derecho y nunca frenaría el agente.

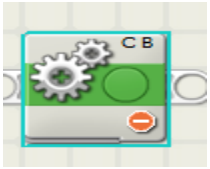


Figura 6.41 a

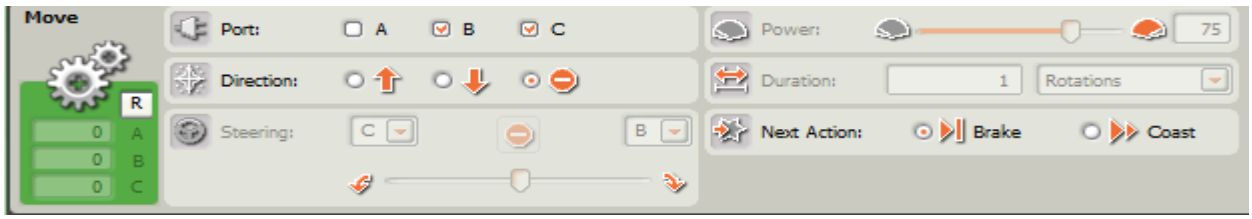


Figura 6.41b

Al concluir el Loop lógico pasa a un siguiente bloque de frenado total de motores, la configuración del este bloque es la más sencilla simplemente se seleccionan los puertos de los motores a frenar y en la opción de dirección damos en el símbolo de alto y todo por se ubicara por defecto.

Después de este bloque viene el bloque infinito, que por ser tan denso se va a explicar por secciones y pasos.

Los siguientes bloques se explican en la numeración 4, 5 y 6.

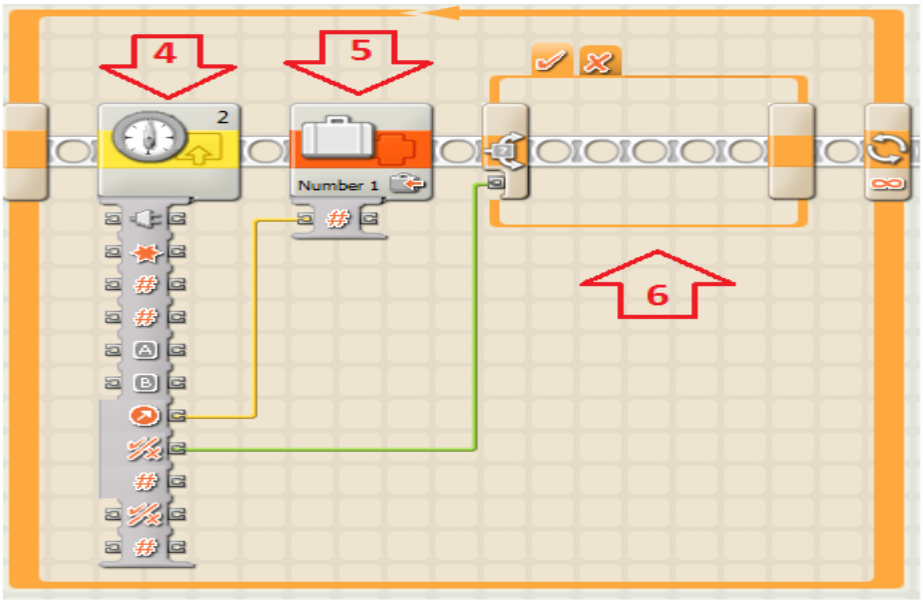


Figura 6.42

4. Bloque compass sensor.

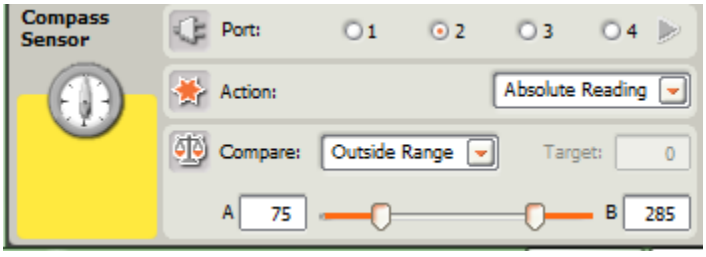


Figura 6.43

La configuración de este bloque con respecto al anterior no varía mucho, se tiene en cuenta el mismo puerto de entrada, el mismo modo de acción, y se varía la comparación de rango

inferior a rango exterior (Outside Range), de 75 a 285 ósea que los números de 74 a 284 serán números para el fuera de rango y por ende falsos.

5. Bloque variable.

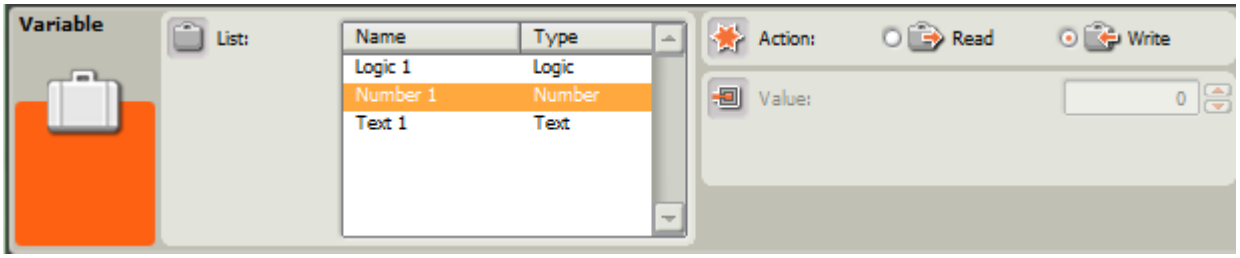


Figura 6.44

Este bloque tiene la capacidad de almacenar o leer variables, usamos el (write) para almacenar las variables que entren por el puerto del bloque. Luego se escoge el tipo de variable a utilizar, entre lógica, o numérica o texto, en este caso numérica. Cabe aclarar que también nos es permitido crear más variables según como sea necesitado.

6. Bifurcación lógica.

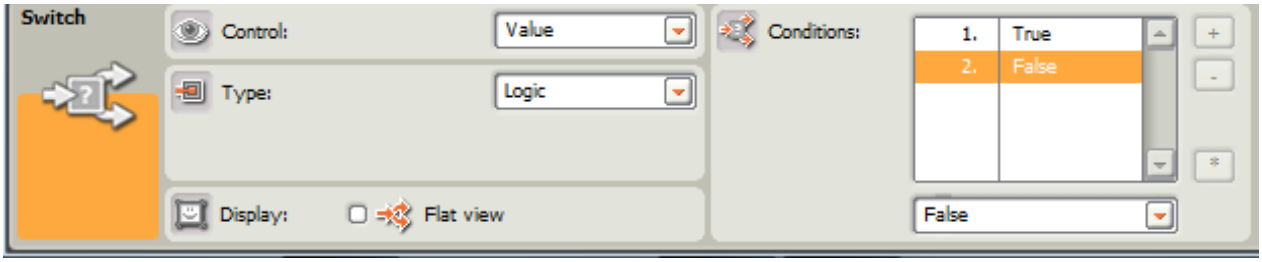


Figura 6.45

Esta es la primera bifurcación que analiza las opciones de verdadero y falso, las bifurcaciones al igual que los ciclos loop, también tiene opciones de operación, puede ser numérica, lógica o de texto.

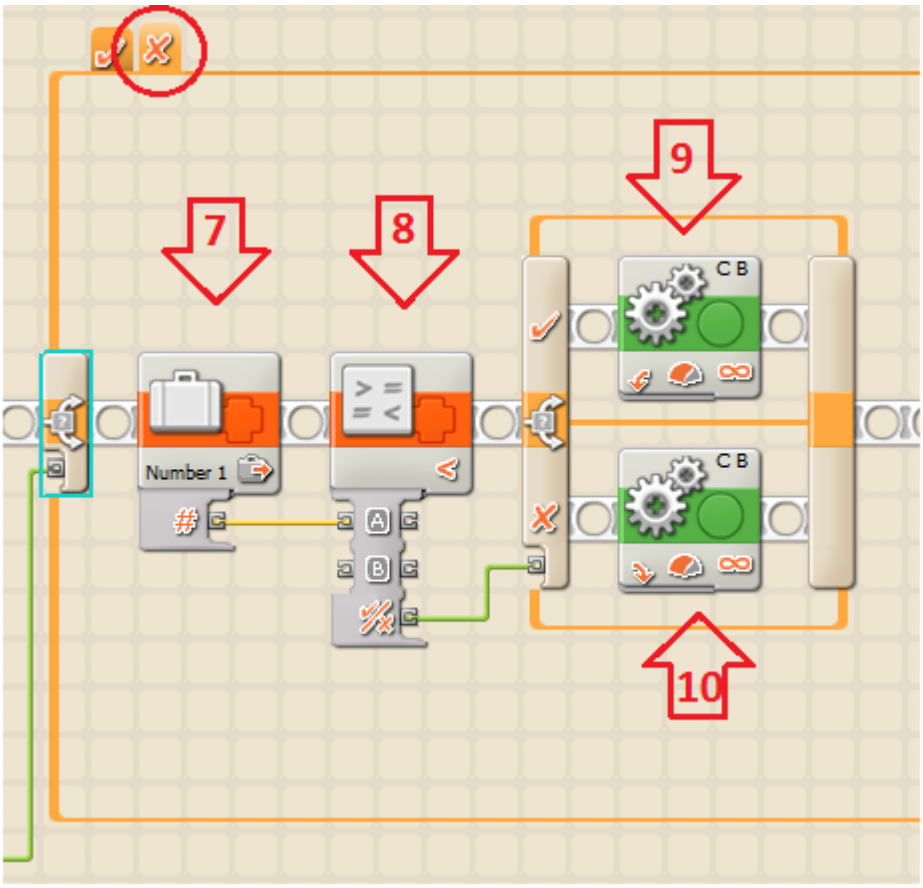


Figura 6.46

Los siguientes bloques se explican con la numeración 7, 8, 9 y 10. Teniendo en cuenta que esta es la bifurcación anterior con la decisión falsa, como lo marca el círculo.

7. Bloque variable.

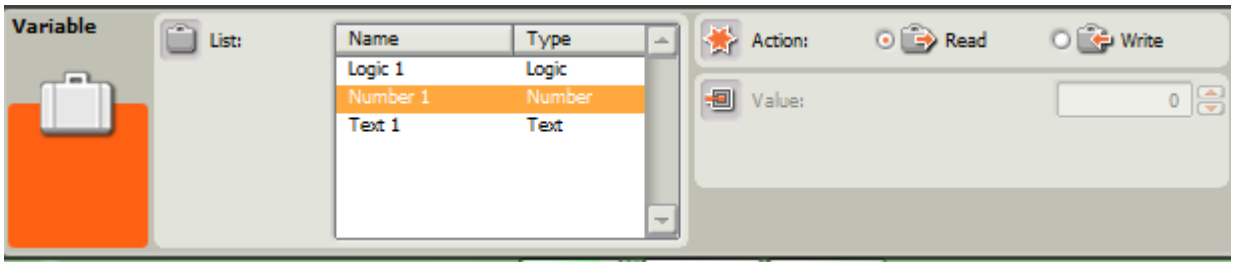


Figura 6.47

Este es el mismo bloque de almacenar variables en este caso se usa para leer la variable number 1 que previamente se almacenó.

8. Bloque comparación.

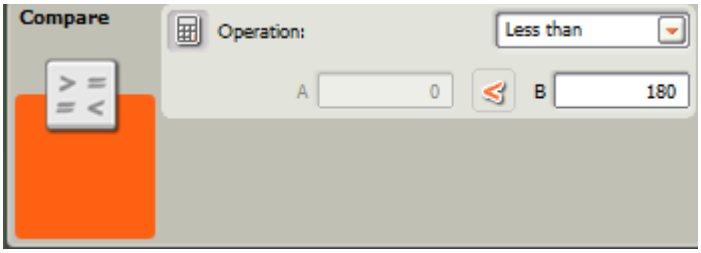
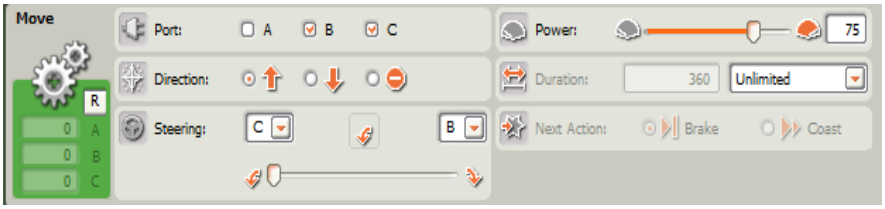
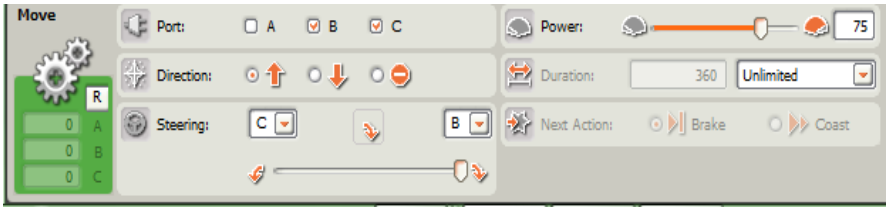


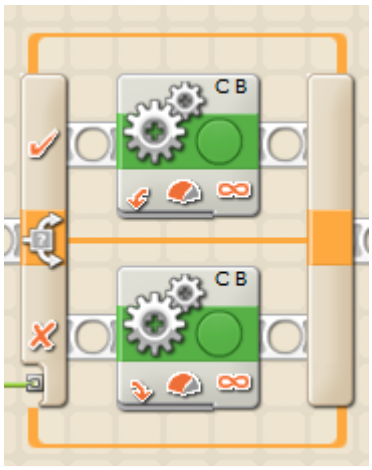
Figura 6.48

La comparación que realiza este bloque es: si lo que entra por el puerto A que es la variable tomada de los valores del compass sensor, es menor que 180 debe tomar la bifurcación positiva, que realiza el giro a la izquierda y si es mayor a 180 realizará un giro a la derecha; esto con el fin de lograr que el robot siempre durante el juego mire hacia el norte y tome el camino más cercano de su posición hacia el norte.

9 y 10. Bloques de giro motor B y C.







Figuras 6.49 a *(Izquierda)*, b *(superior)*, c *(Inferior).*

Estos dos bloques son los que cumplen las peticiones del bloque anterior por medio de la bifurcación lógica, se dejó un potencia de 75% para ambos para una respuesta rápida y a cada uno se le dio su respectiva dirección, con duración ilimitada.

Los siguientes bloques del 11 al 17, corresponden a la bifurcación positiva del compass sensor.

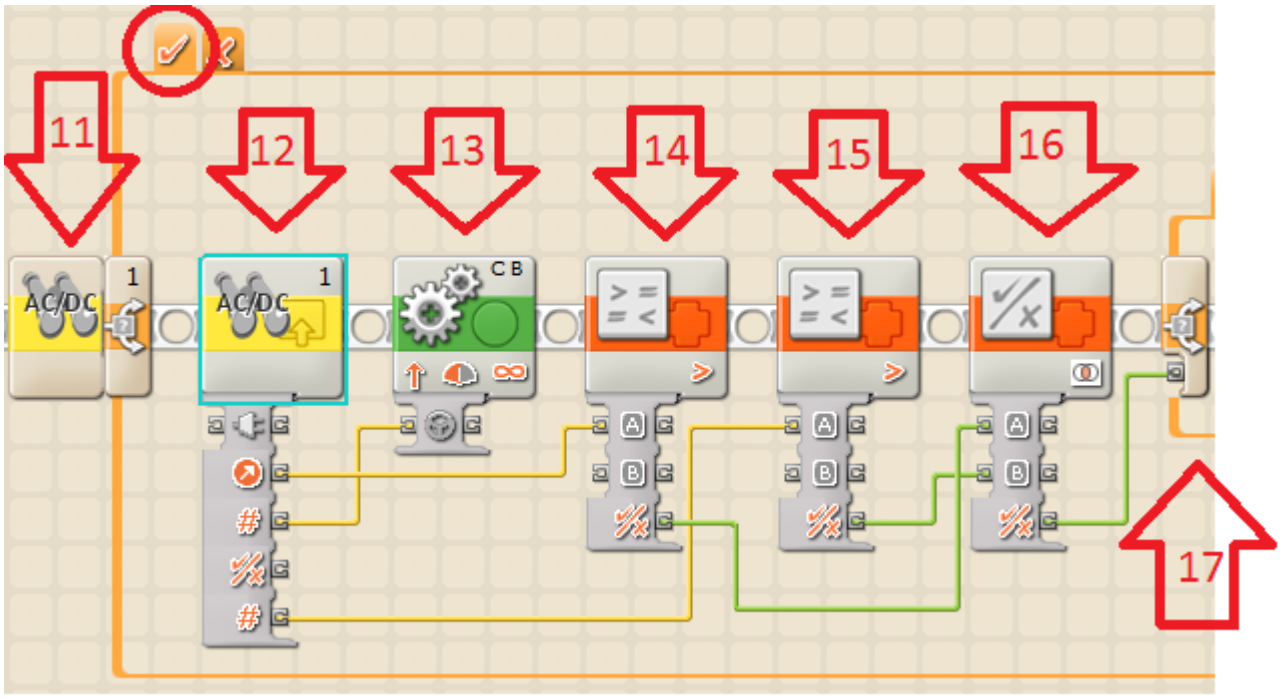


Figura 6.50

11. Bifurcación IRSeekerV2.

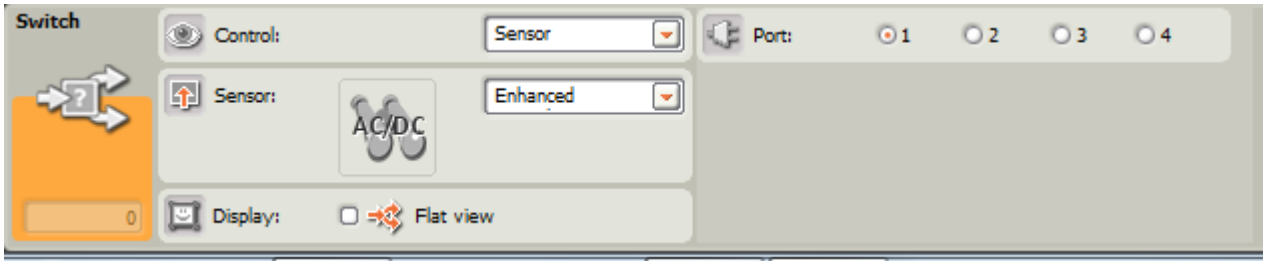


Figura 6.51

Esta bifurcación nos brinda la información de si la pelota se encuentra encendida o no, y tomará dos posibles caminos en este caso como lo indica el círculo rojo, es la decisión verdadera.

Como se puede apreciar en la imagen, no hay cambios posibles en la bifurcación ya que esta solo permite escoger si se maneja una variable o un sensor y que tipo de sensor.

12. Bloque IRSeekerV2.

El bloque IRSeekerV2 cumple dos funciones una con el degdirection y otra con el strenght.

El degdirection manda una señal en grados de la ubicación de la pelota la cual recibe el siguiente bloque.

El strenght, envía la potencia de señal que recibe de los infrarrojos enviados por la pelota, esta señal es analizada por el siguiente bloque.

Este bloque no tiene menú de modificaciones así que las funciones a utilizar son validadas de su panel inferior, donde salen los cables amarillos.

13. Bloque movimiento motor B y C.

Este bloque al ser manejado por el degdirection, simplemente se le despliega el panel inferior y se conecta a la imagen de volante conocida como steering, el cual cumplirá con seguir la pelota por el número de grados enviados por el bloque IRSeekerV2.

14. Bloque comparación IR Direction.

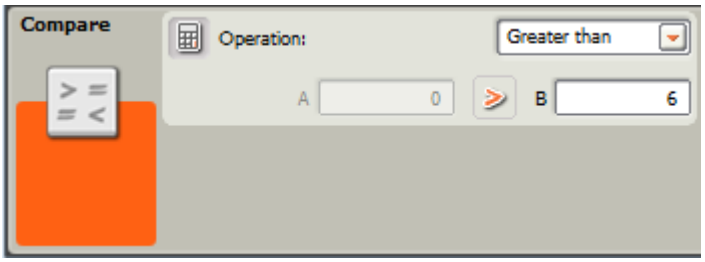


Figura 6.52

Este bloque es el primer indicio de la ubicación de la pelota para poder patear al arco teniendo en cuenta que según el esquema del sensor hay 10 posibles posiciones de la pelota con respecto al sensor conocido como IR direction; desde 0 qué es visibilidad nula de

la pelota, el 1 que es ubicación de la pelota en la parte inferior izquierda, hasta el 9 que es ubicación de la pelota en la parte inferior derecha. Entonces al tomar la comparación mayor que 6 estamos diciendo que si la pelota está ubicada en la parte derecha del sensor por que 5 es cuando tiene la pelota en frente. Por ende cumple una de dos condiciones para patear el balón.

15. Bloque comparación strength.

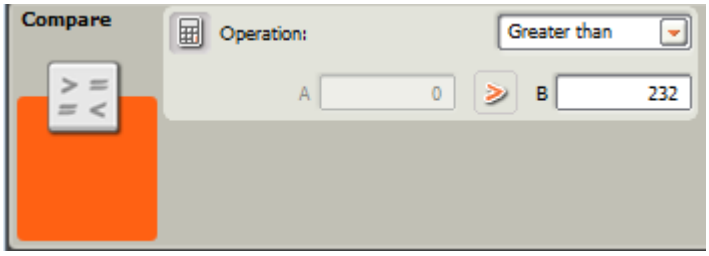


Figura 6.53

En este bloque también se tiene en cuenta la señal enviada por IRSeekerV2, en este caso la del strenght, que nos indica la potencia de la señal por parte de la pelota, en pocas palabras que tan cerca está la pelota del sensor. Según las mediciones marcadas en la práctica se decidió que 232 es 3 cm porque las especificaciones técnicas arrojadas por hitechnic, no arrojan una ecuación o una constante para evaluar la potencia de señal enviada por la pelota.

16. Bloque lógico

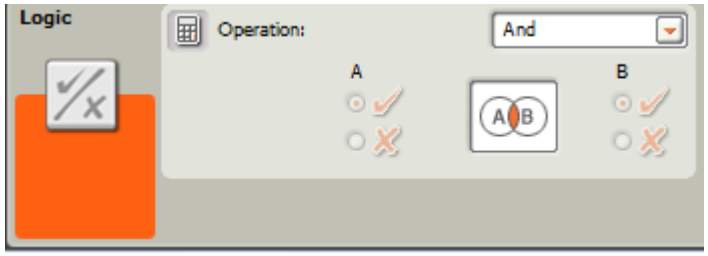


Figura 6.54

Este bloque recibe la lógica del Bloque comparación IR Direction y de Bloque comparación strength, cumplirá la misma función que una compuerta and que si de los dos bloques recibe un uno su resultado será uno, del resto de posibilidades lógicas resultara 0.

17. Bifurcación de disparo al arco

Esta simplemente recibe la señal del bloque lógico y realiza la opción verdadera y en la falsa no ejecuta ninguna acción ya que no interesa.

La acción que ejecuta esta bifurcación es la siguiente:

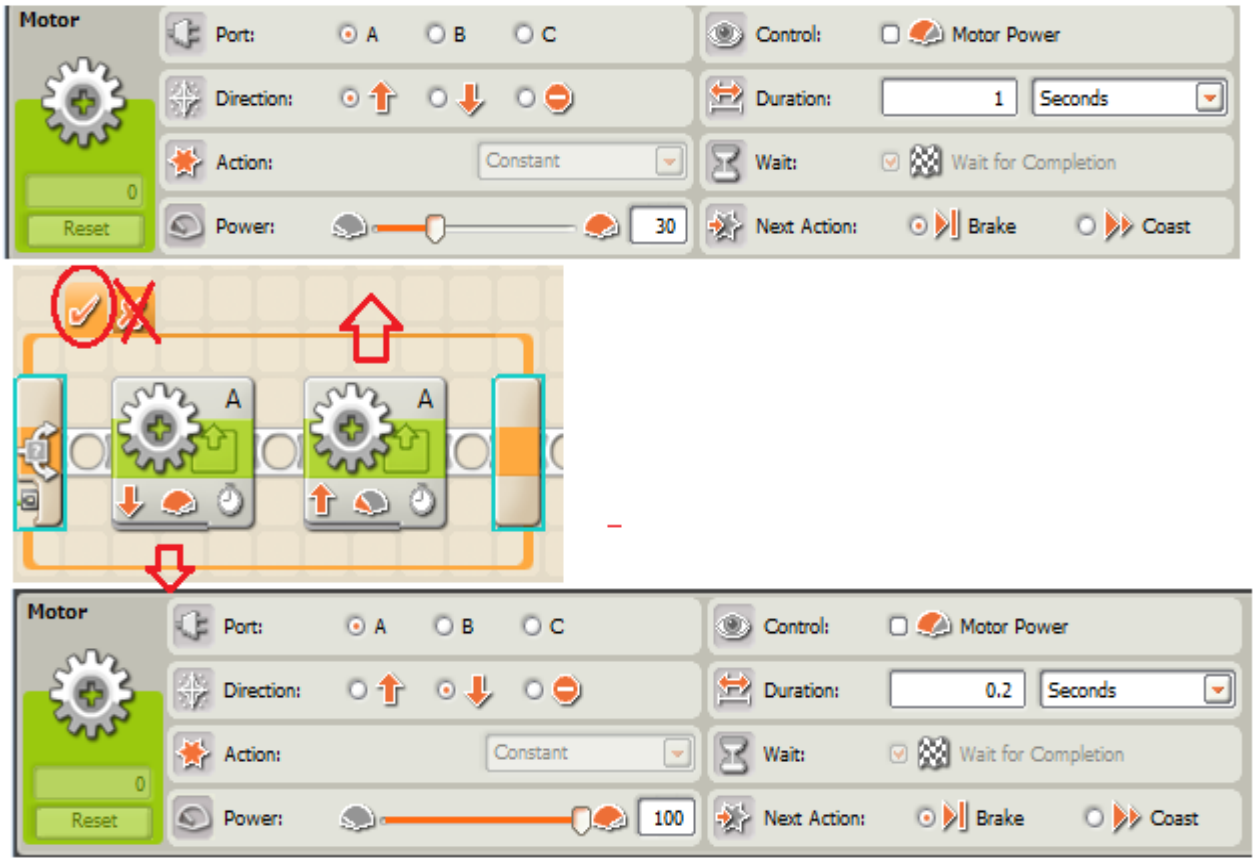


Figura 6.55

Se sobreentiende que la bifurcación sólo opera en positivo, y que el primer motor girara a una potencia de 100 por durante 2 milésimas de segundo y después el mismo motor girara en sentido contrario a una potencia de 30 por un segundo, para devolver la patada de disparo.

La siguiente explicación corresponde a los números del 18 al 22 de la figura.

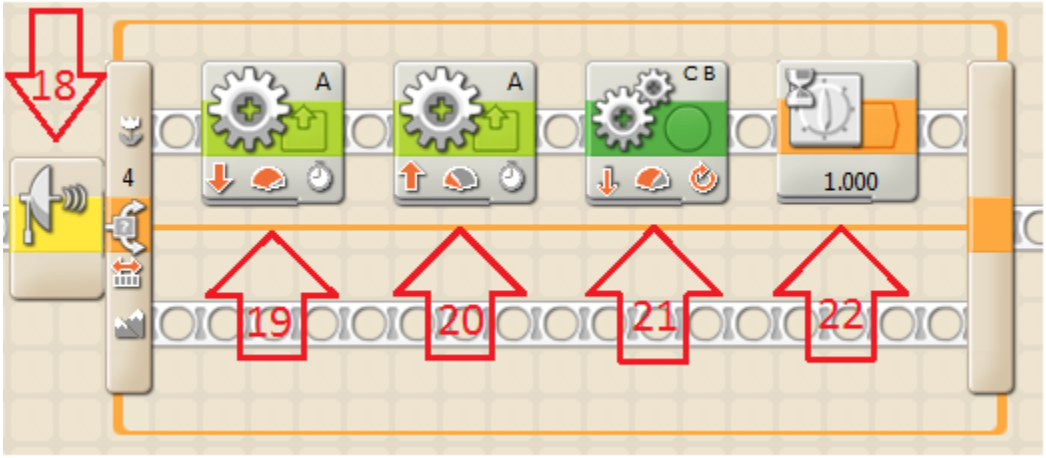


Figura 6.56

18. Bifurcación sensor ultrasonido.

Figura 6.57

Hay que tener en cuenta que por la falta de entradas para sensores la limitación para saber si tenemos un oponente o una pared en frente es grande. Por eso al tener un solo sensor optamos por volverlo un multifunción, que no será muy listo pero si muy útil a la hora del enfrentamiento.

Primero el rango de distancia que tiene el sensor es de 255 cm de los cuales él solo tendrá en cuenta los últimos 35 cm, con el fin de censar ese rango y en caso de tener una obstrucción, el generara una acción sin darle importancia a la condición falsa. Cabe aclarar que el sensor también puede medir en pulgadas.

19 y 20. Movimiento motor A.

Este lleva la misma configuración del sensor de patada, ya que como se mencionó anteriormente al haber un solo sensor él tiene que cumplir con varias funciones. La razón de que ejecute esta acción es que el robot al tener un obstáculo a menos de 35 cm, lo más probable es q sea un defensa y como nuestro agente no tiene la capacidad de hacer pases ni de eludir rivales, él lo que hará es disparar directamente a puerta, como la distancia de censado es tan lejana hay mayor probabilidad de que la pelota no toque al defensa y vaya directamente a puerta.

21. Movimiento Motores B y C.

Figura 6.58

Este movimiento es muy importante ya que este es el resultado de que el sensor de ultrasonido detecta una obstrucción que aparte de ser un defensa puede ser una pared, como realmente el robot nunca va a estar muy cerca de las paredes no importa si antes a dado una patada, y el movimiento hacia atrás si ayudara a darle más espacio al agente para que busque la pelota y siga jugando, teniendo en cuenta que el árbitro del partido si ve que la pelota está en un punto muerto puede moverla. También el hecho de que de reversa frente a un defensa le da tiempo a la pelota o al defensa de despejar la pelota y robarla.

¿Por qué 4 rotaciones? se eligieron 4 rotaciones porque al girar cuatro veces los motores hacia atrás da una distancia de 30 cm, que sumados los 35 cm de distancia de obstrucción da 65 cm lo cual no permitirá que el robot entre en un conflicto de programación por estar pensando siempre lo mismo al tener más espacio para buscar la bola.

22. Bloque de espera.

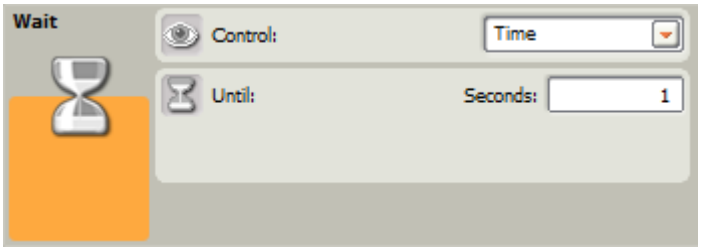


Figura 6.59

Este bloque tiene la cualidad de frenar todas las acciones del robot por un periodo definido de tiempo, que puede ser, por la señal positiva de un sensor o por tiempo definido. En este caso se utilizó un segundo para darle tiempo al entorno de ajustarse a las necesidades de ataque.

Retomamos la bifurcación IRSeekerV2 Negativa, en los números 23 y 24 de la figura siguiente:

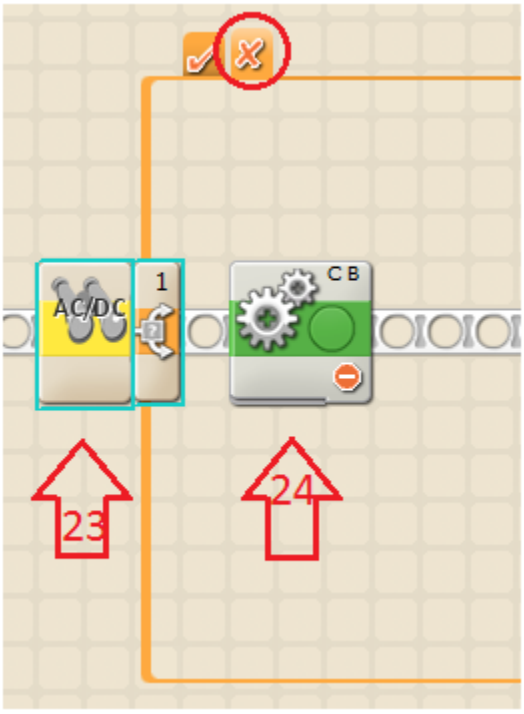


Figura 6.60

23. Bifurcación IRSeekerV2 negativa

Como se puede observar señalado por el círculo es la bifurcación vista anteriormente pero la parte negativa, donde ejecutará el bloque 24.

24. No movimiento de motor B y C.

Este bloque explicado previamente como el que no efectuará ningún tipo de movimiento es el que rige en esta bifurcación negativa.

CAPÍTULO 7

PRUEBAS EXPERIMENTALES

Estas pruebas buscan demostrar la efectividad, entendiendo esta como la capacidad de lograr un efecto deseado, que tienen los algoritmos implementados y explicados en los 2 capítulos anteriores, para que el agente desempeñe la función deseada, aquí se explicará la forma y método utilizados para cada experimento propuesto y en el capítulo siguiente se muestran los resultados obtenidos y su análisis correspondiente.

7.1 Pruebas de arquero

Con base a las variables que deseamos controlar en las pruebas (Orientación, ubicación y achique, se desarrollaron una serie de experimentos que permiten evaluar el desempeño del agente cumpliendo las funciones relacionadas a estas variables:

7.1.1 Prueba de ubicación.

Esta prueba consiste en determinar la efectividad que tiene el agente de ubicarse cerca al punto central de su área de acción en el momento que no se esté presentando una amenaza de gol, se busca que la distancia del agente con respecto a las líneas limítrofes a oriente y occidente de su área sea la misma en ambos sentidos tal como se muestra a continuación:

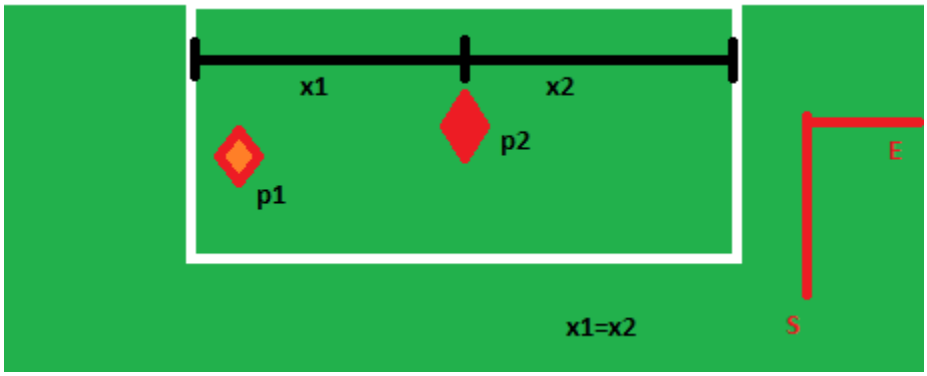


Figura 7.1

Sea p1 una posición inicial en un instante de tiempo t=0, con distancias x1 y x2 con respecto a las líneas Occidental y Oriental respectivamente, del área de acción del arquero, donde p1

es arbitraria para x1 y x2 iniciales. Se busca medir la tolerancia promedio en la ubicación de p2, en el instante t=2 con respecto a su valor teórico. Para esto obtenemos un promedio de los x1 y x2 reales si ejecutamos *n* veces el algoritmo, medimos x1 y x2 y acudiendo a la estadística sabemos que:

$$X1 = \frac{\sum_{i=0}^{n} x1_n}{n} \quad [3]$$

Sabiendo que $X1_{real} = 40cm$

De igual manera sucede con x2. Tal como se indica en la imagen, la distancia se mide desde el eje de simetría del agente, supóngase los valores:

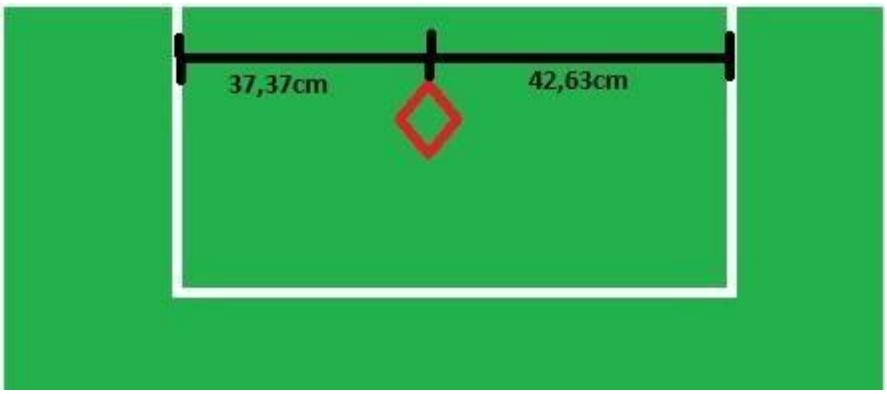


Figura 7.2

De [3] deducimos el error de la posición real:

$$\%Error = \frac{x1_{teórico} - x1_{real}}{x1_{real}} * 100 \quad [4]$$

Así se determina la confiabilidad del algoritmo en la búsqueda del punto medio entre oriente y occidente del área.

Los análisis de estas pruebas se encuentran en el capítulo siguiente, dedicado a ello.

7.1.2 Prueba de orientación:
Aquí se mide la efectividad del agente para orientarse de acuerdo a la igualdad (1) definida en la sección 2.3.1 donde se dijo que:

O=180

En este caso se definió una posición inicial medida en grados por la rotación u orientación inicial del agente con respecto a O=0 (Norte) siendo E=90 y así sucesivamente según se indica en la tabla consignada en la misma sección.

Una descripción gráfica de dicho procedimiento se muestra en la siguiente imagen:

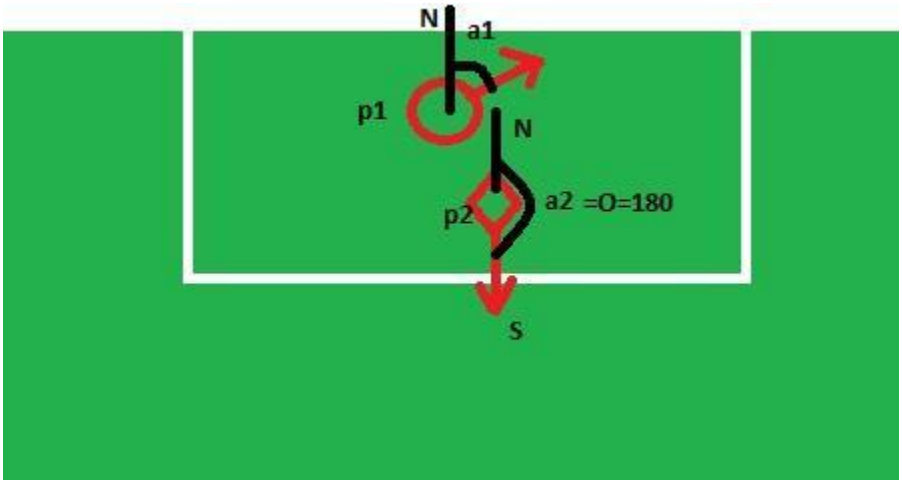


Figura 7.3

Donde p1 es la posición inicial con un ángulo a1 con respecto al norte; p2 es la posición deseada con el ángulo O. De igual forma a la prueba anterior, se pretende obtener un margen de tolerancia aceptable al comparar la orientación O real vs la orientación teórica, para esto se implementa la forma similar de [3] donde:

$$O_{real} = a2_{real} = \frac{\sum_{i=0}^{n} a_i}{n} \quad [5]$$

Por medio de [5] es posible determinar el porcentaje de error de O real respecto a O teórico:

$$\%Error = \frac{O_{real} - O_{teórico}}{O_{teórico}} * 100 \quad [6]$$

A partir de esta medida determinamos la confiabilidad que ofrece el código implementado en cuanto a orientación se refiere, para esto se somete el robot a la modificación de su orientación girando la superficie en la que reposa logrando que este se reubique cada vez y observando así el resultado de esta.

7.1.3 Prueba de achique

Esta consiste en lanzar la bola al arquero para que este determine la dirección de la que proviene y se interponga entre ella y el arco a fin de evitar el gol.

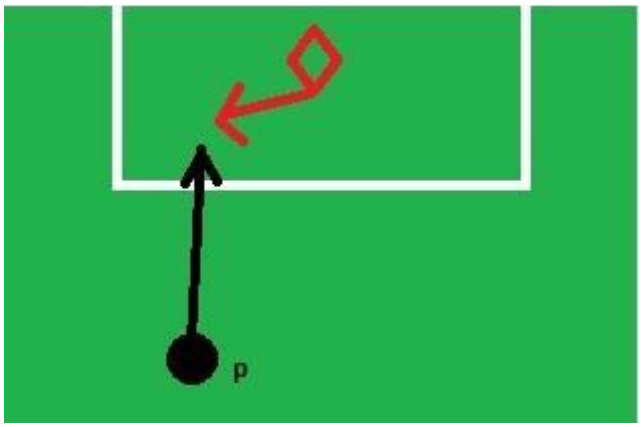


Figura 7.4

De esta se cuenta el número de veces que se logró dicho objetivo y se aplican las mismas fórmulas anteriores para determinar la confiabilidad del algoritmo. Esto permite medir la efectividad en la reducción de espacio y oportunidad de gol que ofrece el arquero ante una situación de amenaza.

7.2 Pruebas defensa

7.2.1 Prueba de ubicación Inicial

La prueba de ubicación inicial del agente defensivo consiste en determinar la efectividad de este, en la búsqueda de la ubicación de sur, cuando la pelota no se encuentre en su rango de acción, para realizar este cálculo se realiza el experimento repetitivo de la ubicación en tres diferentes posiciones, de la siguiente manera:

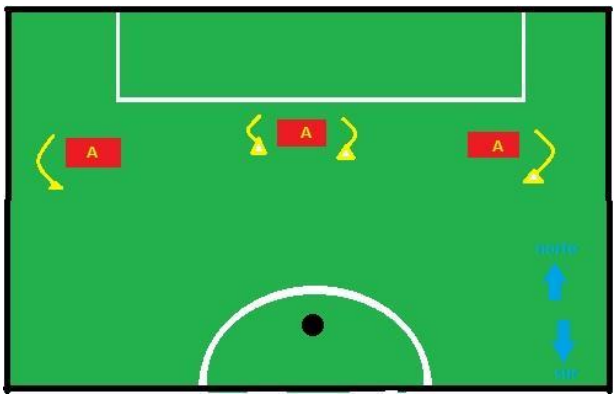


Figura 7.5 *prueba de ubicación*

En el gráfico se muestra de color rojo la posición del agente defensivo en los lugares escogidos para la prueba y de amarillo el sentido de giro que podría tomar para llegar a su referencia, en este caso el sur, en donde los resultados que nos arrojó el experimento fue:

Posicion	Prueba1	Prueba2	Prueba3	Prueba4	Prueba5	Prueba6	Prueba7	Prueba8	Prueba9
A	1	1	1	2 -4	2 -5	1	1	1	1
B	1	1	1	1	1	1	1	1	1
C	1	2 -5	1	1	2 -3	1	1	1	1

Tabla 6

En donde tomamos las letras como la ubicación del Agente, los numero 1 son pruebas exitosas en donde el agente queda ubicado en el rango permitido, los número 2 son pruebas erróneas en donde se deja saber cuánto es el desfase en grados de la ubicación requerida.

7.2.2 Prueba de despeje de La Pelota

Para la prueba de despeje de la pelota se tomaron los siguientes puntos de referencia:

a) Despeje Perfecto: que es cuando la pelota es enviada a terreno contrario.

b) Despeje Relativo: Que es cuando el agente despeja pero no pasa de la mitad del campo de juego. Se despeja hacia un lado.

c) Despeje errado o sin despeje que se da por varios factores, que pueden ser por factores como la demora en la lectura del sensor o mucha velocidad de la pelota por lo cual no se realiza el despeje. Para este experimento se realizó la siguiente dinámica con pelota quieta...

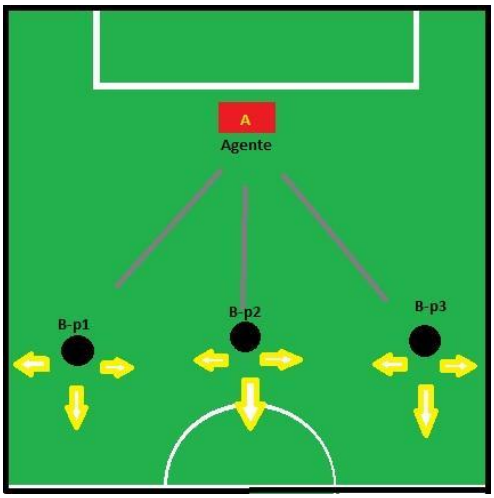


Figura 7.6 *Dinámica de Despeje*

En la figura se observa las posiciones escogidas para la prueba en donde la pelota se ubica en los laterales y el centro, en donde el agente defensivo, se acerca a la bola y la rechaza, mostrando con las flechas algunas de las posibles direcciones de la pelota, que para poder medir su efectividad se realiza la siguiente toma de datos:

# de prueba	Posición 1	Posicion 2	Posicion 3
1	1	1	1
2	1	1	2
3	2	1	1
4	1	1	1
5	2	1	1
6	1	1	2
7	1	1	1
8	2	1	1
9	2	1	1
10	1	1	2

Tabla 7 *Datos con pelota quieta*

En la tabla anterior podemos analizar el comportamiento del agente después de una serie de repeticiones desde los puntos elegidos para esta prueba que se realiza con pelota quieta, al realizar la prueba con la pelota en movimiento nos da como resultado la siguiente tabla

# de prueba	Posición 1	Posicion 2	Posicion 3
1	2	1	1
2	1	2	2
3	2	1	1
4	2	3	3
5	2	1	1
6	3	2	2
7	1	1	2
8	2	3	3
9	3	2	1
10	1	1	2

Tabla 8 *Datos con pelota en movimiento*

7.3 PRUEBAS DE DELANTERO

Las variables a controlar son la orientación espacial y el remate a gol. De estas dos surgen otro tipo de variables las cuales podemos llamar como secundarias que son consecuencia del entorno. Aunque se pretende tener en cuenta las variables secundarias cabe aclarar que por las limitantes del robots no son posibles de controlar, por ende el agente se centrará en las dos variables principales a controlar.

7.3.1 Prueba de orientación inicial

O=0

En esta prueba se determinara la eficiencia del agente para orientarse al iniciar el partido, hacia el norte, antes de colocar la pelota en juego. Colocando el agente en diferentes posiciones para medir su respuesta frente a cualquier ubicación y orientación expuesta a él, en la cancha.

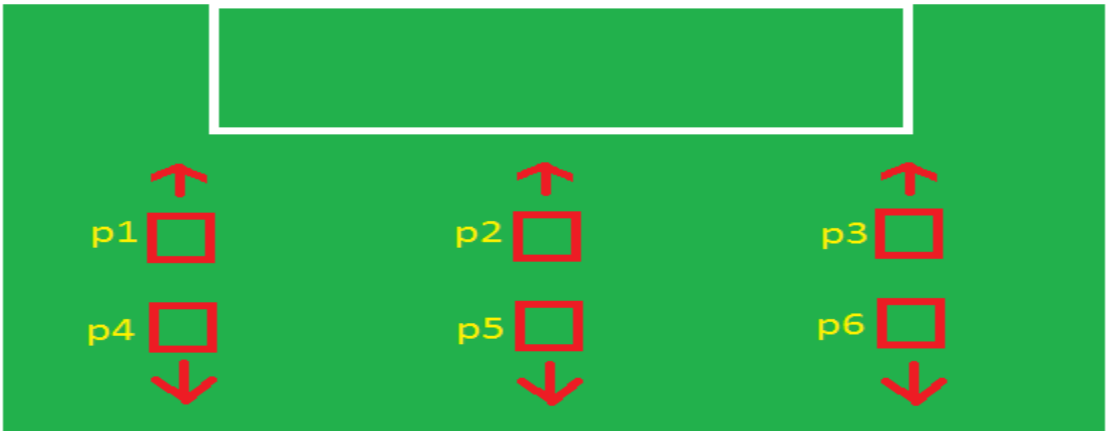


Figura 7.7

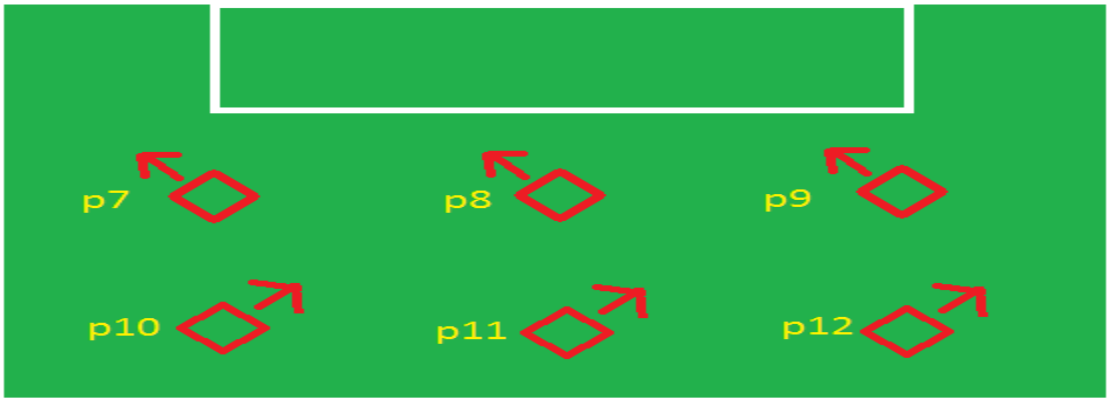


Figura 7.8

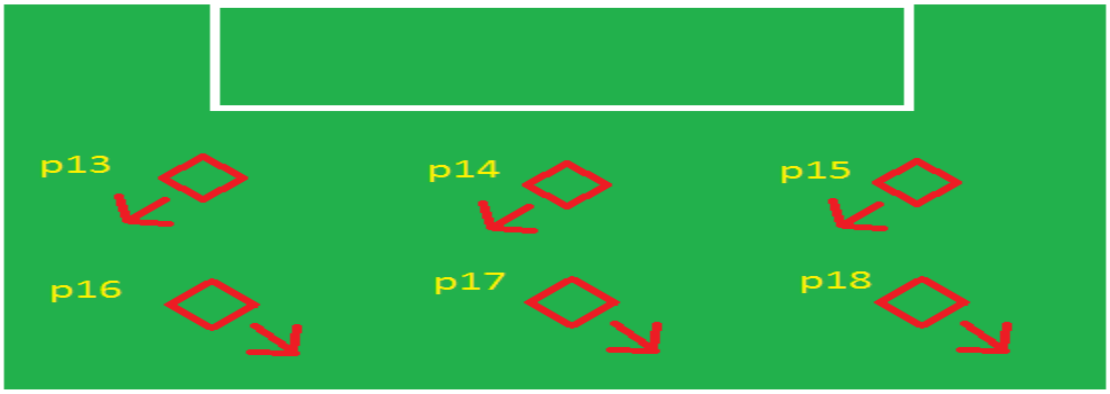


Figura 7.9

En las figuras anteriores se puede apreciar las posibles ubicaciones del agente frente al arco donde por cada orientación se tomaron 3 muestras y por cada posición se tomaron 6 muestras, dando una muestra total de 18 posibilidades. De lo cual podemos inferir que de *n* veces de expuesto el experimento sin importan la ubicación u orientación va a tener un margen de error x. Dando como promedio una eficiencia en el número de acierto y una deficiencia en el número de desaciertos.

La ejecución deseada por el robot, consiste en que gire hacia la izquierda en su propio eje hasta que coincida el valor en grados equivalente al norte en este caso el 0. En la siguiente figura se puede apreciar.

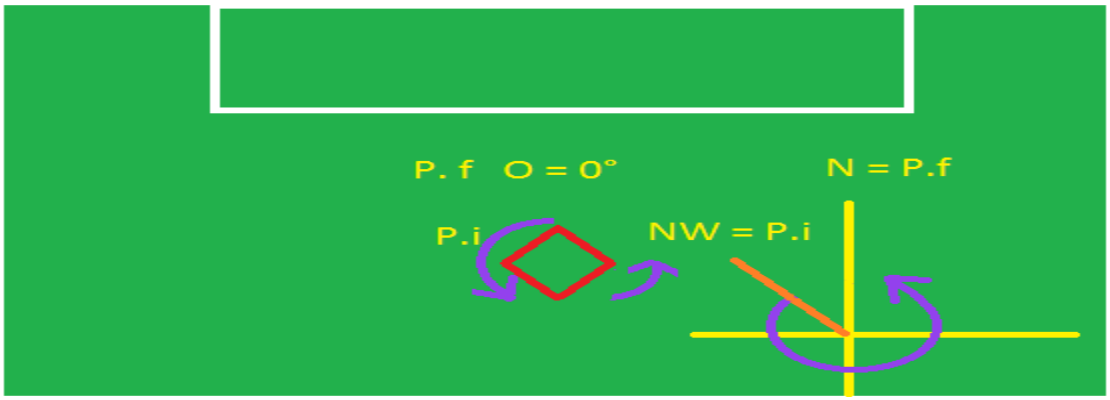


Figura 7.10

7.3.2 Marcar goles en jugada individual.

En esta variable influyen otro tipo de variables como: la pelota, ubicación de la pelota, ubicación del robot, orientación espacial, dirección y tipo de pegada. Para reducir estas variables al máximo, se optó por realizar pruebas en tres diferentes posiciones y orientaciones iniciales para evaluar la conducta del agente, ubicando también la pelota en tres posiciones diferentes y obtener la ubicación ideal para el disparo al arco.

También se analiza la trayectoria del balón haciendo un modelo de posibilidades en ejecución de tiro.

En la siguiente figura se muestra una de las posibilidades que fueron evaluadas durante el experimento.

Primera posibilidad:

Figura 7.11

Segunda posibilidad:

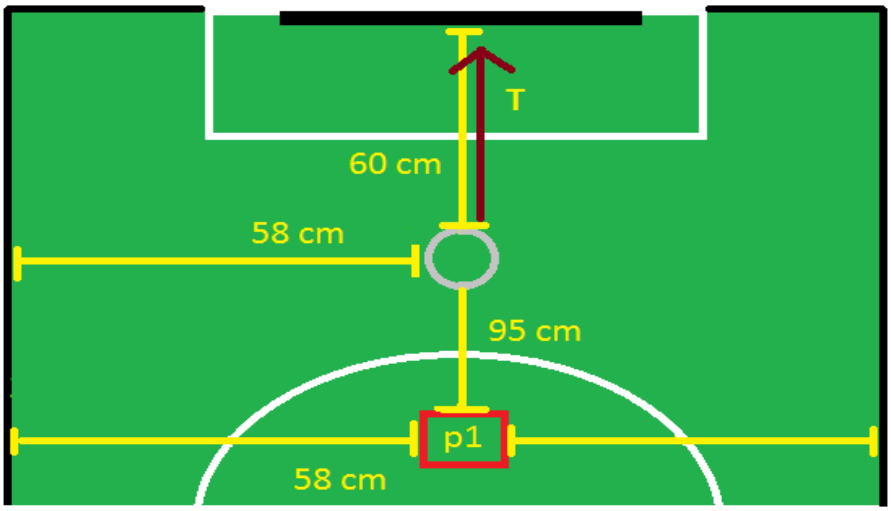


Figura 7.12

Tercera posibilidad:

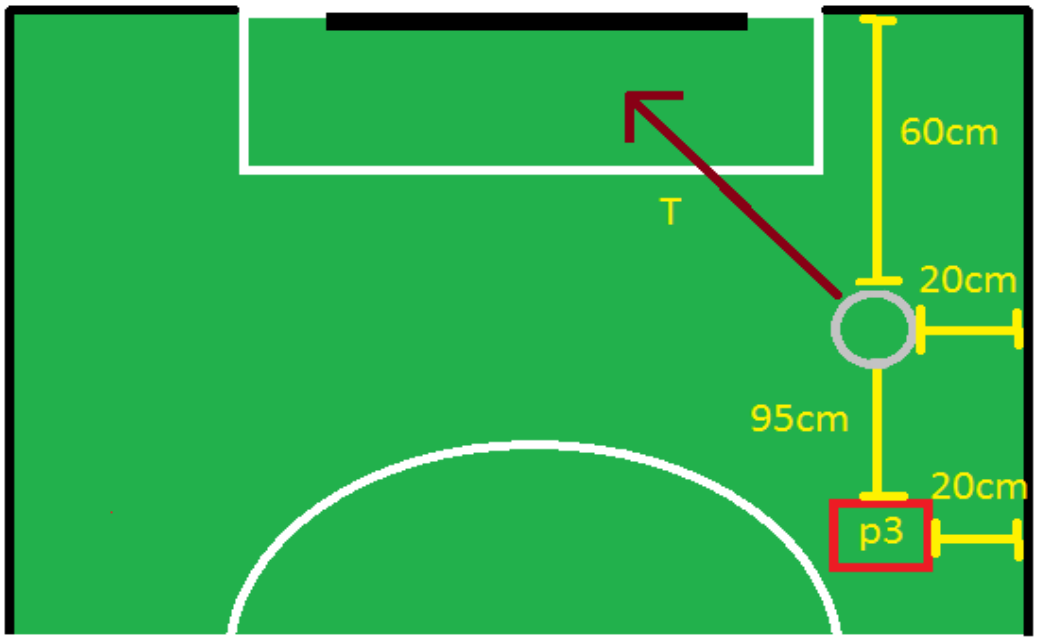


Figura 7.13

Estas son las tres posibilidades a evaluar, donde siempre se colocaba el robot y la bola a la misma distancia con respecto a la cancha, para medir su verdadera efectividad en una jugada repetitiva. Solamente se cambió su posición horizontal para medir cuál de estos tres era la posición más adecuada para un seguro gol.

Cabe aclarar que en este experimento no se tuvo en cuenta el defensa ya que las probabilidades de gol en cada caso bajan, abruptamente.

CAPITULO 8

ANÁLISIS DE RESULTADOS

8.1 Resultados experimento con arquero

8.1.1 Prueba de Ubicación

Siguiendo las condiciones expuestas en el capítulo anterior respecto a las pruebas a aplicar, los resultados obtenidos se consignan en la siguiente tabla:

Prueba de Ubicación Arquero				
# Prueba	**X1i(cm)**	**X2i(cm)**	**X1f(cm)**	**X2f(cm)**
1	7	73	35	45
2	14	66	35,3	44,7
3	21	59	37	43
4	28	52	38,5	41,5
5	35	45	38	42
6	42	38	36	44
7	49	31	39	41
8	56	24	37,7	42,3
9	63	17	39,2	40,8
10	70	10	38	42
Total			373,7	426,3
Promedio			**37,37**	**42,63**

Tabla 9

Donde X1i y X2i corresponden a las distancias en p1 antes de iniciar el proceso de ubicación, mientras X1f y X2f son las distancias finales obtenidas después del proceso de ubicación. Espacialmente se representa en la siguiente imagen:

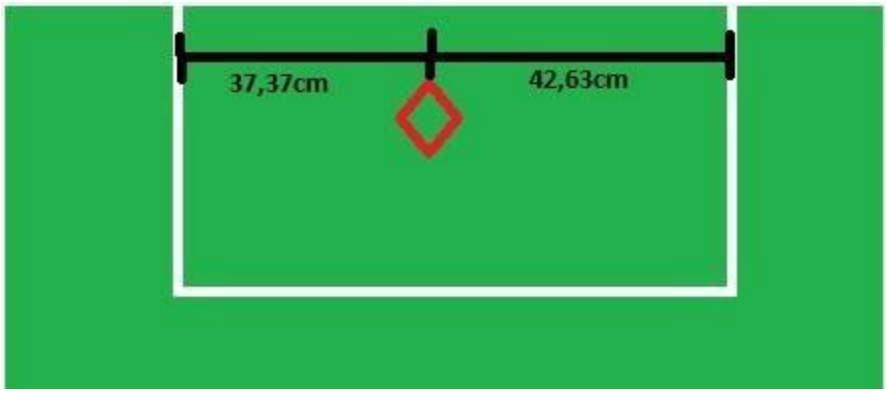


Figura 8.1

Si determinamos en [4] el % de error obtenemos:

$$\%Error = \frac{37{,}37-40}{40} * 100 = -6{,}575\%$$

En la ecuación anterior el valor negativo indica que hay una *contracción de la distancia* X1, lo que sería positivo si tomáramos como variable a medir X2 ya que esta se alarga en igual proporción, este error es muy bajo en relación a las herramientas con las que se cuentan demostrando así una eficiencia del 93,425%, resultado altamente confiable para una programación compleja sobre una plataforma de software orientada especialmente a programas simples.

También nos muestra este resultado la capacidad que tiene el robot para reubicarse después de haber ejecutado una acción concerniente a sus roles.

8.1.2 Resultados prueba de orientación del arquero

Los resultados obtenidos son:

Prueba de orientación del arquero		
#Prueba	a1(deg)	a2(deg)
1	0	169
2	36	171
3	72	176
4	108	170
5	144	174
6	180	169
7	216	175
8	252	172
9	288	171
10	324	174
Suma		1721
Promedio		172,1

Tabla 10

Obteniendo una ubicación final promedio:

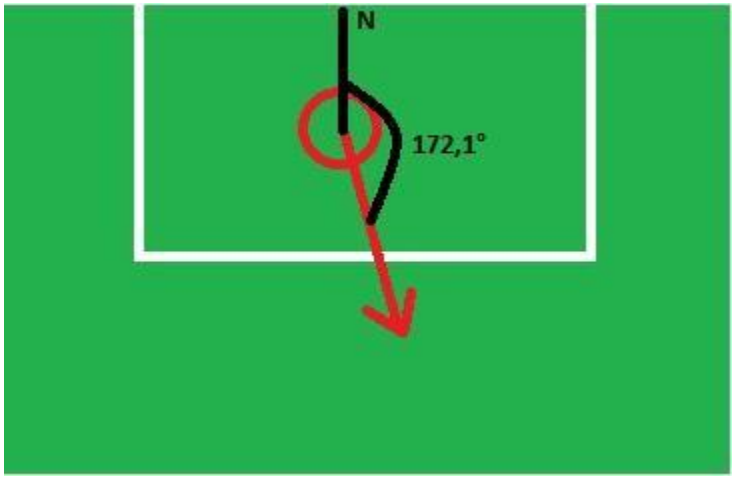


Figura 8.2

Obteniendo el % de error mediante [6] tenemos:

$$\%Error = \frac{172{,}1 - 180}{180} * 100 = -4,\underline{44}\%$$

Tolerancia muy baja y totalmente aceptable ya que con esto cubre el rango total de detección de la bola, puesto que esta proviene siempre desde el sur inicialmente y con base en la especificación técnica del IRSeeker dada anteriormente, estará siempre alerta a la pelota IR.

8.1.3 Resultado de prueba de achique

De las pruebas realizadas se obtuvieron los siguientes resultados:

Prueba achique		
# Prueba	atajado	gol
1	0	1
2	1	0
3	1	0
4	1	0
5	1	0
6	0	1
7	1	0
8	0	1
9	1	0
10	1	0
Total	7	3
Promedio	0,7	0,3

Tabla 11

De la tabla anterior podemos concluir que el algoritmo tiene una efectividad del 70% para intervenir en la trayectoria de gol evitando así la caída de su portería. Este se puede mejorar si existe la posibilidad de ingresar al algoritmo funciones de matemática más compleja pero que no están disponibles en el software de Lego, sería adecuado programar desde otra plataforma.

8.2 Resultado de pruebas de Defensa

Los resultados obtenidos en la prueba de ubicación inicial de agente defensivo arroja un desempeño sobresaliente en esta tarea ya que el porcentaje obtenido es más o menos del 80% con una desviación estándar de 3 grados cuando no logra completar la tarea, este rango de error en la ubicación se puede dar por distintos factores, donde el principal de todos es la interferencia de ondas con los sensores puesto que estos son muy sensibles a ondas extrañas.

Ya en la parte de la tarea que de despeje del balón como lo vemos en la tabla 8 y 9, podemos determinar que el defensa en posición central es mucho más efectivo que el defensa en despeje lateral teniendo un 70% de efectividad de despeje por los laterales, esto teniendo la pelota quieta, porque en movimiento los valores cambian drásticamente bajando su efectividad a un 60% en cualquier sector de la cancha. Datos determinados por el ensayo de prueba y error en donde se halla el promedio de efectividad en porcentaje del algoritmo diseñado para el defensa. Por consiguiente se puede concluir que la efectividad de este agente se podría mejorar con la comunicación directa entre agentes para lo cual se necesitaría un accesorio externo que permitirá esta posibilidad y el mejoramiento del algoritmo

8.3 Resultados experimento con delantero

8.3.1 Prueba de orientación

Según el capítulo anterior se emplearon unas condiciones experimentales con el fin de obtener unos resultados, donde abarque la efectividad y desatino del robot frente a una ejecución.

El experimento arrojó la siguiente tabla:

Posición de prueba	Optima	Falsa	Desvió
Norte	0°		
Norte	0°		
Norte	0°		
Sur	0°		
Sur	0°		
Sur	0°		
Noroeste	0°		
Noroeste	0°		
Noroeste		330°	-30°
Noreste	0°		
Noreste	0°		
Noreste	0°		
Suroeste		270°	-90°
Suroeste		285°	-75°
Suroeste	0°		
Sureste	0°		
Sureste	0°		
Sureste	0°		

Tabla 12

El desvío total fue de 195°, donde la sumatoria de intentos en grados fallidos da 885°, en la ubicación óptima no hay desvíos ya que el robot solo frena cuando marca 0°, por ende no tiene rango de desvío.

El rango de error está dado por:

$$Err = \frac{18-15}{18} \times 100$$

El cual arroja un resultado del 16.666% de error, dando como conclusión un 83,3334 de acierto del robot en esta ejecución.

8.3.2 Prueba de anotación de gol en jugada individual

Los resultados esperados son la ubicación más adecuada para un posible gol donde salieron las siguientes tablas del experimento del capítulo anterior.

Tabla de la primera posición

Intento	Optimo	Fallido	Desviación
1		Palo	Palo Izq
2		Afuera	55 cms izq
3	Gol		
4		Palo	Palo Izq
5	Gol		
6		Afuera	62 cms izq
7		Afuera	61,5 cms izq
8	Gol		
9		Palo	Palo Izq
10		Palo	Palo Izq

Tabla 13

En esta posición la efectividad del delantero es muy poca con una tendencia al lado izquierdo gracias a su diseño con la patada en la parte derecha.

Error = 70%

Efectividad = 30%

Veces en palo = 42.85% de las fallidas

Afuera= 57.14 de las fallidas

Tabla Segunda posición:

Intento	Optimo	Fallido	Desviación
1	Gol		
2	Gol		
3		Palo	Palo der
4	Gol		
5	Gol		
6	Gol		
7		Afuera	Fuera de rango
8		Afuera	17 cm izq
9	Gol		
10		Afuera	18,2 cm der

Tabla 14

Esta tabla demuestra que la efectividad de tiro sube un 50% con respecto al anterior posición, en un solo tiro la pelota salió por fuera de la cancha, del resto cumplió con su objetivo.

Error = 40%

Efectividad = 60%

Veces en palo = 25% de las fallidas

Afuera= 75% de las fallidas

Tabla tercera posición:

Intento	Optimo	Fallido	Desviación
1		palo	Palo izquierdo
2	Gol		
3		afuera	36,3 cms lzq
4		palo	Palo izquierdo
5	Gol		
6		afuera	64,4 cms der
7	Gol		
8		afuera	3,1 cms izq
9	Gol		
10	Gol		

Tabla 15

Esta tabla no es tan efectiva como la segunda pero sí más que la primera, dando una pequeña inclinación de los fallos al lado izquierdo. Es notorio que esta posición es la más regular con respecto a sus lanzamientos, ya que entrega un 50/50 de probabilidad de fallos y aciertos.

Error = 50%

Efectividad = 50%

Veces en palo = 40% de las fallidas

Afuera= 60% de las fallidas

CAPITULO 9

CONCLUSIONES

Luego de hacer diversos experimentos y probar diferentes configuraciones para cada rol, se encuentra que el hardware si tiene la capacidad de desempeñar sus funciones adecuadamente, pero el software limita el desempeño del hardware; los resultados muestran que las configuraciones permiten desempeñar el rol correspondiente pero que las grandes limitantes de librerías y/o bloques que se puedan usar para una programación avanzada, interfieren con el buen desarrollo de cada rol.

Estos resultados arrojan un éxito en cuanto a la eficiencia ya que, por ejemplo, los bloques de operación matemática sólo permiten las 4 operaciones básicas más raíces o potencias, pero no permite el uso de relaciones trigonométricas entre otras. Aun así se logra satisfactoriamente controlar las variables propuestas para cada agente.

Se concluye además, en cuanto al aspecto de programación, que en sistemas multiagente de carácter vigilante se hace necesario la repetitividad de acciones mediante el anidamiento de ciclos, lo que permite entrar y salir de los mismos, y que se debe ejecutar las acciones pertinentes de reconocimiento del terreno para orientarse mejor al momento de iniciar una acción o al momento de terminarla según sea el caso correspondiente. También se evidencia que la plataforma utilizada, Lego NXT, ocupa gran cantidad de espacio en los códigos debido a su entorno gráfico, lo que provoca un problema con la memoria interna del bloque NXT, esto puede solucionarse con entornos de desarrollo más avanzados como MatLab o LabView que están más enfocados a la programación y el desarrollo profesional propiamente dicho así se deje de lado las características intuitivas del entorno NXT.

Es necesario destacar que a futuro se debe controlar la fuente de energía ya que en este caso, al ser un algoritmo de vigilancia constante y múltiples acciones consume una gran cantidad de energía, lo que deja obsoleto el clásico sistema de baterías AA.

Otras conclusiones que se dan se expresan a continuación:

- El fútbol es una actividad útil para los científicos que trabajan en inteligencia artificial robótica, porque exige al robot percibir su entorno, usar sus sensores para elaborar un modelo de ese entorno, y utilizar los datos para razonar y emprender las acciones apropiadas.

- Tal como sucede con los jugadores humanos, el fútbol también exige a los robots la comunicación y cooperación necesarias entre jugadores de un mismo equipo, y lo más importante es que requiere la habilidad de aprender, ya que gracias a ella los equipos pueden ajustar sus tácticas para neutralizar las de sus oponentes y así tener mejores oportunidades de derrotarlos.

- El desarrollo en robótica permite la automatización eficaz de tareas complejas y/o repetitivas con lo que se contribuye a la eliminación de riesgos a la integridad humana del trabajador, permitiendo su desempeño en labores menos riesgosas o que sean propensas a la falla por agotamiento.

- El comportamiento óptimo del robot además de la programación y el ensamblaje de piezas, depende de factores ajenos como lo es campos magnéticos, luz y agentes contaminantes.

- El reconocimiento y orientación del compás magnético de LEGO MINDSTORMS, es afectado por el campo magnético como el de los motores, celulares etc.

- La eficacia del Sensor receptor IrSeekerV2, del LEGO MINDSTORMS, es alterada sin dar respuesta por factores ajenos, como la luz solar o halógena de alta potencia. Además la detección de la pelota también es dañada por señales encontradas de wifi en un lugar cerrado.

- Es necesario tapar el encapsulado del sensor de color para que las sombras no afecten su detección y detecte todos los colores como negro.

- El software LEGO MINDSTORMS NXT 2.0, es un software muy útil y con gran cantidad de herramientas, pero cuando se trabajan algoritmos largos y el manejo de variables el software tiende a fallar y arrojar errores constantes, se recomienda el uso de algoritmos concretos y directos, para hacer la programación sencilla y de pocas lecturas.

- En la práctica fue evidente que aunque el sensor IR cuenta con un avanzado sistema digital para el procesamiento de señales como lo indica las especificaciones técnicas del fabricante, este era muy vulnerable a un entorno muy saturado de señales de diversas clases, ya que al intentar rastrear la pelota IR oficial, este se “confundía” y tomaba decisiones erradas, fallo que tornó torpe el desempeño de los agentes. De igual forma este aumento de señales, provoca una lectura inestable de la brújula causando desorientación en el robot. Esto demuestra que para llevar a cabo una competencia “sana” de fútbol robótico en la plataforma Lego, es necesario garantizar un entorno limpio en cuanto a señales circundantes se refiere.

CAPÍTULO 10

REFERENCIAS:

A. De internet

[1]. Tomado de la plataforma educativa CFIE de Valladolid, “Asesoría Tecnológica” Avalaible Online http://cfievalladolid2.net/tecno/cyr_01/robotica/intro.htm *Fecha de consulta Noviembre 2 del 2012*

[2]. Tomado del weblog colectivo Xataka Ciencia de Websog SL / de Anibal Ollero y Guillermo Heredia de la Universidad de Sevilla, Avalaible Online http://www.xatakaciencia.com/robotica/robots-moviles-i *Fecha de consulta Noviembre 2 del 2012*

[4] Tomado de la web: Futbol y Entretenimiento, grupo de estudio, dirigido por Juan Felipe Merino. Avalaible Online http://futbolmerino.jimdo.com/2011/06/09/caracter%C3%ADsticas-del-jugador-de-f%C3%BAtbol-por-posici%C3%B3n/ *Fecha de consulta Noviembre 2 del 2012*

[5] Tomado Del Blog web, Blogspot, Creadop por Carlos Uriel Ariza Angarita. Avalaible Online http://carlosurielariza.blogspot.com/. Fecha de consulta 1 de Diciembre del 2012

[6] Tomado de slideshare, Trabajo realizado para la asignatura de Diseño de Microrrobots Móviles de la Universidad de Alcalá por: Estela Díaz López Ignacio Esperabe de Arteaga del Alamo Rubén Fernández Carnicero David Gualda Gómez Julián Manzano D’Onofrio José Antonio Martín Esteban Javier Mateos Andaluz Luis de Santiago Rodrigo Noviembre 2006. Avalaible Online http://www.slideshare.net/naciendo/lego-nxt./ Fecha de consulta 28 de noviembre 2012

[7] Tomado de la web Lrobotikas, robótica educativa y creativa. Avalaible Online http: //lrobotikas.net/wiki/index.php?title=Sensor_de_color, Fecha de consulta 1 de Diciembre 2012.

[8] Tomado de Hitechnic, avalaible online www. Hitechnic.com

[9] www.robocup.com información futbol robótico página oficial - Robot World Cup, (2005) www.robocup.org

B. Tesis o proyectos de grado

[3] Aprendizaje de sistema multiagente por medio Aprendizaje de refuerzo, por Edgar Alirio Aguirre Buenaventura a la Universidad Distrital, Francisco José de Caldas, facultad de Tecnología.

C. Otras referencias

- Aplicaciones.virtual.unal.edu.co/drupal/files/esafutbolpaper2006.pdf. Pag 1 – 2.

- Sahota Michael K., Mackworth Alan K. (1994) “Can Situated Robots Play Soccer?” Canadian AI-94, Department of Computer Science, University of British Columbia, Vancouver Canada.

- www.redusoi.org/docs/publicaciones/P6-Utilizacion de LEGO NXT en docencia universitaria.pdf. Pag. 3 – 4.

- RobotC. www.robotc.net. Jun 2009.

- www.revista.unal.edu.co/index.php/avances/article/download/9988/10520

- http://ro-botica.com/mindstorms_accesorios.asp

D. Textos y documentos

[10] Texto pdf tomado de http://wiki.robocup.org/wiki/Junior#Rules, Link del archivo fuente http://www.ai.rug.nl/robocupathome/documents/rulebook2010_FINAL_VERSION.pdf

CAPÍTULO 11

ANEXOS

Se incluyen los tres programas de delantero, defensa y arquero como anexos, como indicio para el desarrollo de jugadores de fútbol con LEGO MINDSTORM, entregando también programas diseñados de prueba para el funcionamiento de cada sensor.

Anexo A

Justificación y Objetivos del Proyecto de investigación en fútbol robótico con Lego NXT-G

OBJETIVO

Diseño e implementación de algoritmos de detección y movimiento para las posiciones de arquero, defensa y delantero de fútbol robótico en plataforma móvil LEGO Mindstorm.

OBJETIVOS ESPECÍFICOS

1. Analizar roles de arquero, defensa y delantero de un partido de fútbol real para seleccionar los comportamientos básicos que se desean implementar
2. Definición de las selecciones y actividades de la plataforma del robot lego para las posiciones de arquero, defensa y delantero, según el análisis de roles.
3. Implementar los algoritmos que gobiernen el comportamiento de cada agente de acuerdo a su posición, mediante el software NXT-G.
4. Validar los algoritmos implementados, con pruebas uno a uno en el campo de fútbol, enfrentando al delantero y el portero, o, al delantero y al defensa , para corroborar las acciones de cada una de las posiciones.

Anexo B

Definiciones y características del fútbol robótico

¿Qué es futbol Robótico?

El fútbol robótico es una actividad llevada a cabo por estudiantes, aficionados y científicos de la robótica, que consiste en la construcción y programación de agentes robóticos, dispuestos a jugar fútbol en un campo de fútbol de pequeñas dimensiones. Esta actividad tiene como referente principal exigir al robot a percibir su entorno, apoyándose de los sensores que utiliza para reconocer ese entorno, y utilizar los datos para razonar y emprender las acciones necesarias.

También el fútbol robótico al igual que el fútbol humano, le pide a los robots que se comunique y coopere a los compañeros de campo, teniendo en cuenta los conocimientos adquiridos durante el transcurso del partido.

¿Qué es RoboCup?

La evolución de las máquinas, tanto en su hardware como en su software, cada día tiende acercarse a competir con organismos biológicos. El campo del fútbol robótico, es un claro ejemplo de esta evolución. Esta práctica que como competencia fue iniciada por RoboCup en 1997, con el fin de promover la creación de robots autónomos por medio de investigación y educación sobre inteligencia artificial. Esta práctica se ha diversificado en competencias como: RoboCup Soccer, RoboCup Rescue, RoboCup Junior y RoboCup@Home esta última iniciada desde el 2006. [9]

Las competiciones robóticas a nivel mundial han estado desde hace más de 20 años en un escenario de intercambio académico. Con el tiempo los robots se han convertido en sistemas cada vez más rápidos e inteligentes. Hay diferentes tipos de certámenes internacionales de competencias robóticas que se efectúan periódicamente en la actualidad.

Una competencia muy reconocida en este ámbito es RoboCup, esta es una organización internacional, registrada en Suiza, que busca unificar el gran esfuerzo realizado a nivel

internacional para promover la ciencia y la tecnología con los juegos de fútbol de robots y agentes de software.

La idea de los robots que juegan al fútbol fue mencionado por primera vez por el profesor Alan Mackworth (Universidad de British Columbia, Canadá) en un artículo titulado "On Seeing Robots", y más tarde publicado en un libro Visión Equipo: Teoría de sistemas, y aplicaciones.

De forma simultánea, un grupo de investigadores japoneses organizaron un Taller sobre Grandes Desafíos en Inteligencia Artificial en octubre de 1992 en Tokio, discutiendo posibles problemas alrededor de esta temática; Este taller dio lugar a un debate serio para evaluar la posibilidad de utilizar el fútbol con el fin de promover la ciencia y la tecnología. Una serie de investigaciones se llevaron a cabo incluyendo un estudio de viabilidad de la tecnología, una evaluación del impacto social, y un estudio de viabilidad financiera. Además, las reglas fueron redactadas, así como el desarrollo de prototipos de robots y sistemas de fútbol simulador. Como resultado de estos estudios se concluyó que el proyecto es factible y deseable. En junio de 1993, un grupo de investigadores, incluyendo Minoru Asada, Yasuo Kuniyoshi y Hiroaki Kitano, decidieron poner en marcha un concurso de robótica, tentativamente llamado el Robot de J-League.[2] Sin embargo, se recibieron reacciones contundentes de los investigadores por fuera de Japón, solicitando que la iniciativa se extendiera como un proyecto internacional, en consecuencia, se cambió el nombre del proyecto por el de Robot World Cup Initiative, "RoboCup" para abreviar.

Paralelamente a este debate, varios investigadores han estado utilizando el juego del fútbol como indicio para su investigación. Está el caso de Itsuki Noda, en Electrotécnica Laboratorio (ETL), un centro público de investigación en Japón, quien estaba llevando a cabo la investigación multi-agente con el fútbol, comenzando el desarrollo de un simulador para los partidos de fútbol. [9]

El simulador se convirtió en el servidor oficial de fútbol de la RoboCup. Independientemente, en el laboratorio de la Universidad de Osaka el profesor Minoru Asada y la profesora Manuela Veloso con su estudiante Peter Stone en la Carnegie Mellon University han estado trabajando en robots que juegan fútbol, gracias a la participación de estos pioneros del campo, RoboCup pudo despegar y ser lo que hoy es considerado uno de los eventos más importantes de este campo. [9]

A su vez, el equipo de Noda anunció el desarrollo de la primera versión de un servidor planeado para fútbol robótico programado en (LISP) que permite la investigación en sistemas multiagente; posterior a dicho lanzamiento, se lanzó una versión desarrollada en lenguaje C++, que fue puesto a disposición del público interesado en la web, haciendo una primera demostración de dicho servidor en 1995 [9]

Durante la Conferencia Internacional Conjunta sobre Inteligencia Artificial (IJCAI-95), celebrada en Montreal, Canadá, en agosto de 1995, se hizo el anuncio de organizar el Primer Robot World Cup Juegos de Fútbol y conferencias en conjunto con IJCAI-97 Nagoya.

Al mismo tiempo, se tomó la decisión de organizar Pre-RoboCup-96, a fin de identificar problemas potenciales asociados con la organización de RoboCup a gran escala. La decisión fue tomada para proveer dos años de preparación y tiempo de desarrollo, a fin de que el grupo inicial de los investigadores pudieran comenzar robot y desarrollo de equipos de simulación, así como dar tiempo de espera para los horarios de sus fondos.

Pre-RoboCup-96 se llevó a cabo durante la Conferencia Internacional de Robótica y Sistemas de Inteligencia (IROS-96), Osaka, de 4 a 8 noviembre 1996, con ocho equipos que compiten en la liga de simulación y demostración del robot real para la liga de tamaño medio. Aunque de alcance limitado, esta competencia fue la primera competición con partidos de fútbol para la promoción de la investigación y la educación.

Los primeros juegos oficiales RoboCup y conferencia se celebraron en 1997 con gran éxito. Más de 40 equipos participaron (real y simulación combinados), y asistieron más de 5.000 espectadores.

También el RoboCup tiene competiciones diferentes en el campo de la robótica para promover otras habilidades y variedades a los diferentes concursantes de los países participantes:

Robocup Rescue: él rescate de desastres es uno de los problemas sociales más graves que implica un gran número de agentes heterogéneos en el entorno hostil. La intención del proyecto RoboCup Rescue es promover la investigación y el desarrollo en este ámbito socialmente significativo en varios niveles que involucran multi-agente coordinación del trabajo en equipo, físicos agentes robóticos para la búsqueda y rescate, infraestructuras de información, asistentes personales digitales, un simulador estándar y los sistemas de apoyo a las decisiones , puntos de referencia de evaluación para estrategias de rescate y sistemas robóticos que están integrados en un sistema integral en el futuro. [9]

RoboCup @ Home: La RoboCup @ Home liga tiene como objetivo desarrollar el servicio y la tecnología robótica asistencial con relevancia para futuras aplicaciones domésticas personales. Se trata de la mayor competición internacional anual de robots de servicios autónomos, y forma parte de la iniciativa RoboCup. Son un conjunto de pruebas de rendimiento que se utiliza para evaluar las habilidades de los robots y el rendimiento en un entorno realista de origen no estandarizado. La atención se centra en los siguientes ámbitos, pero no se limitan a: Human-Robot-Interacción y Cooperación, navegación y cartografía en entornos dinámicos, visión por computador y reconocimiento de objetos en condiciones de luz natural, manipulación de objetos, comportamientos de adaptación, integración comportamiento, inteligencia ambiental, de Normalización y el Sistema de la Integración. [9]

RoboCup Junior: RoboCup Junior es una iniciativa educativa orientada a los proyectos que patrocina eventos robóticos locales, regionales e internacionales para jóvenes estudiantes. Está diseñado para introducir RoboCup a alumnos de primaria y secundaria, así como a estudiantes universitarios que no tienen los recursos para que se involucren en las ligas mayores todavía. El enfoque de la liga junior está en la educación. El torneo ofrece a los

participantes la oportunidad de participar en programas de intercambio y de compartir la experiencia del encuentro con sus pares de otros países.

RoboCup Junior ofrece varios retos, cada uno haciendo hincapié en los aspectos cooperativos y competitivos. Para los jóvenes estudiantes, RoboCup Junior proporciona una introducción emocionante en el campo de la robótica, una nueva forma de desarrollar habilidades técnicas a través de la experiencia práctica con la electrónica, hardware y software, y una oportunidad muy motivadora para aprender a trabajar en equipo, compartiendo tecnología con los amigos. [9]

Las tres modalidades que el RoboCup Junior son futbol, Danza y Rescate:

- Futbol Junior: consiste de un 2-2 con un ambiente totalmente dinámico, que consiste en el seguimiento de una pelota emisora de luz, en un campo cerrado.

- Danza Junior: Uno o más robots se unen con la música, vestidos con algún traje y moviéndose en armonía creativa.

- Rescate Junior: Los robots deben identificar las víctimas, dentro de recreados escenarios de desastres, que varían en complejidad desde la línea siguiente en una superficie plana para escoger los caminos a través de obstáculos en terrenos irregulares. [9]

Aparte de RoboCup está el FiraCup organizado por la Universidad Nacional de Malasia, el Departamento de Educación Politécnica (Ministerio de Educación Superior) y MYSET (Sociedad Malasia de Ingeniería y Tecnología) del 24 hasta el 29 de agosto de 2013.

Este el principal evento internacional en donde se reúnen expertos en robótica a desafiarse unos a otros en los eventos de estilo olímpico robóticos (como el fútbol, levantamiento de pesas, correr y maratón). Durante los últimos 15 años, el evento fue conducido por dos expertos en robótica es decir, el profesor Dr. Jong-Hwan Kim, KAIST (Corea) y Vadakkepat Prahlad (Universidad Nacional de Singapur). Hasta la fecha, el evento ha reunido un fuerte interés de Corea, Japón, Taiwán, China, Eslovaquia, Alemania, Canadá, Malasia, India, Singapur, Emiratos Árabes Unidos, Indonesia, Reino Unido, etc. Este evento es para robot de muy alto nivel.

Normatividad

1. EQUIPO

1.1 Regulaciones [10]

Un equipo se conforma de uno o varios miembros o agentes, en donde cada equipo debe tener un capitán. El capitán es la persona responsable de la comunicación con el árbitro. El equipo puede sustituir a su capitán durante la competición. No se permiten al capitán para llevar cualquier ropa amarilla o azul que puede ser vista por el robot (para evitar la interferencia con el color de objetivo).

1.2 Violaciones

No se permiten a los equipos que no respeten el reglamento al participar. El árbitro puede requerir que el capitán de equipo se cambie de ropa o sea sustituido por otro miembro de equipo si la interferencia con el color de objetivo es sospechada

2. ROBOTS

2.1 Número de robots / sustituciones

Se permiten a cada equipo tener dos robots para las sustituciones. Se prohíben la substitución de robots dentro de un equipo o con robots otros equipos.

2.2. Interferencia

No se permiten a los robots ser coloreados de amarillo o azul para evitar la interferencia con los colores de los objetivo. Las partes amarillas o azules usadas en la construcción de los robot deben o ser ocultas por otras partes para evitar confusiones de percepción por otros robots o pueden ser pintadas con un color neutro. El robot no debe emitir luz infrarroja. Sin embargo, la distancia del sensor infrarrojo puede ser usada mientras este no afecte a otros

robots. Si los robots son pintados, ellos deben ser pintados de color mate. Para evitar problema con los sensores infrarrojos. Las partes menores son irrelevantes mientras los robots no sean afectados. El equipo que se crea afectado tiene el deber de probar que es afectado.

2.2.3 Control

No se permiten el empleo de mando o control a distancia de ninguna clase. Los robots deben ser iniciados a mano por la gente y ser controlado autónomamente.

2.2.4 Comunicación

No se permiten a los robots usar varios tipos o clase de comunicación durante el juego amistoso a no ser que la comunicación entre dos robots sea vía la clase bluetooth 2 o la clase 3(Distancia de acción aproximadamente 20 metros). Los equipos son responsables de su comunicación. La disponibilidad de frecuencias no puede ser garantizada.

2.2.5 Agilidad

Los robots deben ser construidos y programados de tal manera que su movimiento no sea limitado en sólo una dimensión (que quiere decir un eje). Ellos deben moverse en todas las direcciones, por ejemplo para dar la vuelta. Los robots deben responder a la pelota en un movimiento directo y avanzado. Los robots deben tener unas dimensiones un poco mayor que el pórtico para que no entren en él. Por eso es importante el uso del travesaño.

2.2.6 Manejo

Todos los robots deben tener una manija estable para sostenerlos y levantarlos. La manija debe ser fácilmente accesible, por ejemplo en la parte superior del robot. Las dimensiones de la manija no pueden exceder la limitación de altura de 22 cm. La manija no debe ser usada para montar los componentes del robot.

2.2.7 Las regulaciones adicionales de las sub ligas

Un torneo puede ser organizado en sub ligas diferentes. Cada sub liga (p.ej. " la Liga Abierta " " y la Liga de Peso Ligera ") puede tener sus propias regulaciones adicionales, incluyendo regulaciones que afectan la construcción de robots. Tales regulaciones serán pasados por el Fútbol RoboCup Junior el Comité Técnico y hacer una parte de esta regla.

2.2.8 Violaciones

No se permiten a los robots que no cumplen con los sus datos específicos/regulaciones para jugar. Si las violaciones son descubiertas durante un juego, en ese momento, el equipo es descalificado para aquel juego. Si esas violaciones similares ocurren repetidamente, el equipo puede ser descalificado del torneo.[10]

2.3. CAMPO DE JUEGO

2.3.1 Tipos de campo

Hay dos tipos de campo diferentes, denominados Fútbol A y B, los cuales pueden ser utilizados en el torneo

2.3.2 Dimensiones del campo

El Fútbol A: El campo de juego es de 122 cm por 183 cm. Las esquinas son planas.

El Fútbol B: El campo de juego es de 182 cm por 223 cm.

Marcación del terreno Fútbol B

El fútbol tiene un campo de juego el cual está comprendido por diferentes zonas las cuales están divididas y marcadas por líneas blancas. Las dos líneas más largas se denominan

líneas de banda, las dos líneas más cortas se denominan líneas de meta. El terreno de juego está dividido por una línea media la cual está en el medio del campo de banda a banda. El centro del campo estará marcado con un punto en la mitad de la línea media, alrededor del cual se traza un círculo con un radio de 35 cm.

El área de meta se encuentra en ambos sentidos del terreno, donde se demarcan dos líneas perpendiculares a la línea de meta, estas líneas se adentraran 180 cm en el terreno de juego y se unirán con una línea paralela a la línea de meta, esta es el área penal. En cada área penal se marcará un punto penal a 200 cm de distancia del punto medio de la línea entre los postes.[6]

2.3.3 Paredes

Las paredes se colocan alrededor del campo, incluso detrás de las porterías. Las paredes son de 14 cm de altura y se pintan de color negro mate.

2.3.4 Pórtico

Fútbol A: la anchura de cada pórtico es de 45 cm, centrada en cada uno de los lados más cortos del campo. El arco es de 14 cm de altura, tiene una barra transversal en la parte superior. El interior de la meta, incluyendo pisos, paredes y crossbar están pintadas un lado amarillo, el azul del otro lado. El exterior está pintado de negro.

Fútbol B: la anchura de cada pórtico es de 60 cm, centrada en cada uno de los lados más cortos del campo. El arco es de 10 cm de altura, tiene una barra transversal en la parte superior. El interior de la meta, incluyendo pisos, paredes y crossbar están pintadas un lado amarillo, el azul del otro lado. El exterior está pintado de negro.

2.3.5 Suelo

El piso de la cancha consta de tapizado verde con superficie dura.

2.3.6 Círculo central

Se hará un círculo central en el campo con un diámetro de 65 cm, pintada con línea blanca.

2.3.7 Área de penalti

Fútbol A y B: El punto de penalti está a 30 cm de la portería y en la mitad del ancho de la cancha. Las áreas penales son marcadas por una línea blanca entre 10mm y 20mm de ancho.

2.3.8 Iluminación y condiciones magnéticas

El campo se debe colocar de manera que la influencia de la luz por infrarrojos externos, sea lo más baja como sea posible y que el campo magnético de la tierra perturbe lo menos posible. Estas condiciones no se puede garantizar, sin embargo, los equipos deben llegar a los torneos preparados para calibrar sus robots basados en la iluminación y magnéticos con las condiciones en el lugar.[10]

2.3.9 Diseño del campo de juego

El campo de juego diseñado y construido para realizar pruebas, está compuesto por MDF como material principal en la superficie de juego, adicional se ha pintado con vinilo y las paredes que encierran el campo de juego son de pino.

Se hace necesario tener un control sobre el diseño bien sea modelado en CAD, desarrollando planos o mediante cualquier otra herramienta de desarrollo para planear el diseño final, como anexo se entregan planos de desarrollo de la cancha.

El diseño realizado, fue inicialmente modelado como se muestra a continuación:

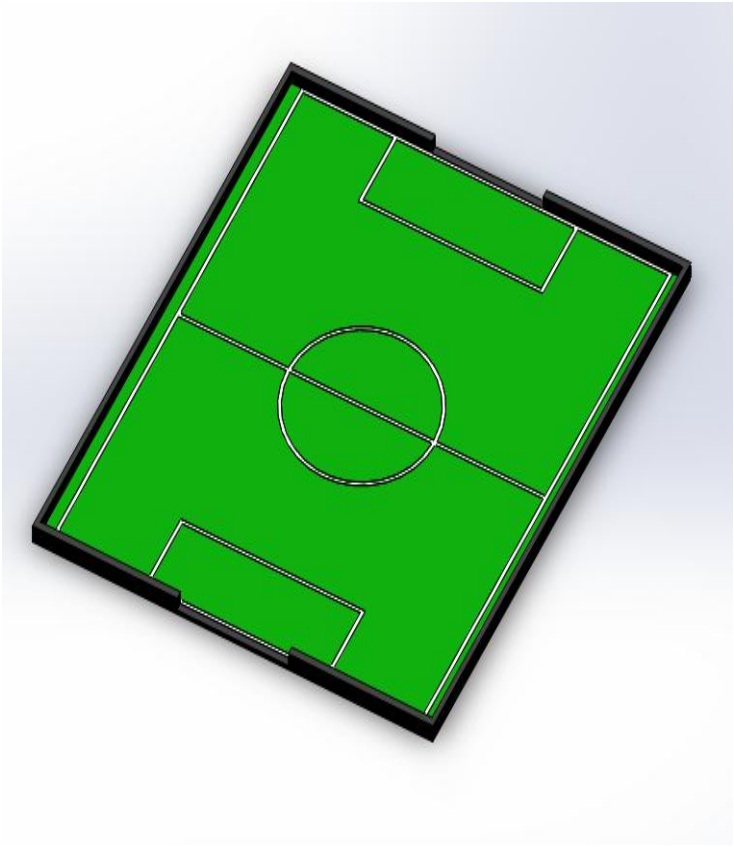


Figura 11.1.*Campo de juego*

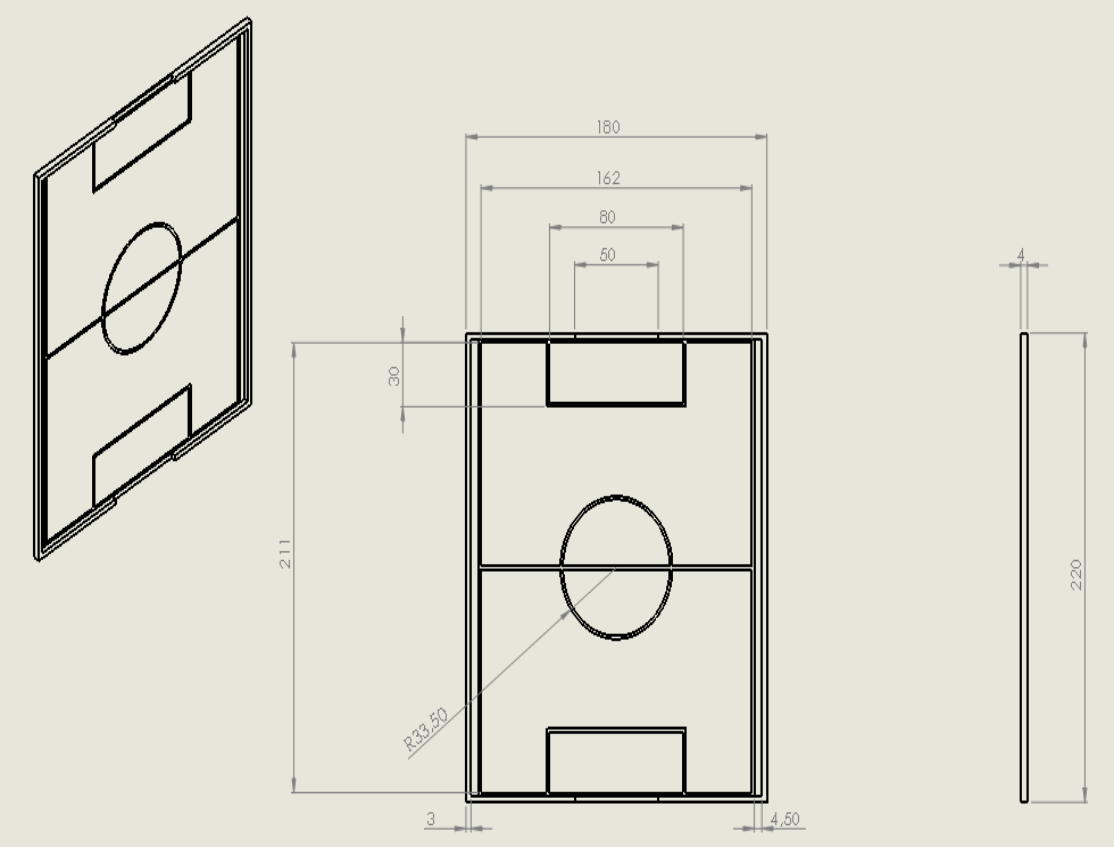


Figura 11.2*. Dimensiones del Campo de Juego*

2.4. BOLA DE JUEGO

2.4.1 Especificaciones generales de la bola

Se utiliza una bola electrónica bien equilibrada. La cual emite rayos infrarrojos (IR). Esta bola es utilizada para fútbol A y B.

2.4.2 Proveedores oficiales de bolas de impulsos

Actualmente, hay una bola que ha sido aprobado por el fútbol RoboCupJunior

Comité Técnico:

- RCJ-05 RoboSoccer pelota que opere en modo A (impulsos) hecha por EK Japón / Elekit (www.elekit.co.jp). Detalles técnicos están en la "Especificación Técnica de balón de fútbol por impulsos", que se adjunta a este reglamento.

2.4.3 Bolas del torneo

Bolas para el torneo deberá estar a disposición de los organizadores. Los organizadores no son responsables de proporcionar pelotas para la práctica.

2.5. Modo de juego

2.5.1 procedimiento y Duración de juego

El juego consta dos tiempos. La duración de cada tiempo es de 10 minutos. Se cuenta con un receso de 5 minutos en medio los tiempos. El reloj del juego correrá para la duración de las mitades sin pararse (excepto si o cuando el árbitro quiere consultar a un funcionario). El reloj de juego será controlado por el árbitro o un ayudante. Los equipos, como se supone, están en la mesa 5 minutos antes de iniciar el juego. Los equipos pueden ser castigados con un minuto, por la discreción del árbitro si ellos son tardíos para el principio del juego. Si un equipo no hace un informe 5 minutos antes del principio del juego, esto pierde el juego y conceden un 5-0 triunfo al equipo victorioso.

2.5.2 Encuentro antes del partido

En el principio, antes de la primera mitad del juego, el árbitro tirara una moneda. El equipo mencionado primero escoge el lado de la moneda. El ganador del sorteo puede escoger el lado de la cancha, o empezar primero con el dominio del balón. El perdedor de este tomara la otra opción. Después de la primera mitad, los equipos cambiarán lados. El equipo que no empieza en la primera mitad del juego con el dominio de la pelota, empezará para comenzar la segunda mitad del juego con esta.

2.5.3 El Saque inicial

Cada mitad del juego comienza con un saque inicial. Todos los robots deben estar localizados sobre su propio lado del campo. Todos los robots deben estar parados. La pelota es colocada por el árbitro en el centro del campo. El equipo que empieza sitúa sus robots sobre el campo primero. Los robots no pueden ser colocados, ni permanecer detrás de la línea de objetivo o en el hacia fuera el área. Los robots no pueden ser movidos una vez que ellos han sido colocados. El otro equipo empezará ahora a colocar sus robots el final defensivo del campo. Todos los robots del segundo equipo deben estar al menos a 34cm lejos de la pelota (que quiere decir fuera del círculo de centro).

El árbitro puede ajustar la colocación de los robots. Al sonar el pito del árbitro (por lo general es un silbido), todos los robots serán prendidos inmediatamente por cada capitán. Cualquier robot que empiece más temprano será quitado por el árbitro del campo y tratado como un robot dañado.

2.5.4 Interferencia humana

Excepto en el saque inicial, la interferencia humana (p.ej. tocando los robots) durante el juego no es permitido, a menos que sea permitido por el árbitro. Los violadores pueden ser descalificados del juego.

2.5.5 El movimiento de pelota

Un robot no puede acaparar una pelota. La propiedad de una pelota quiere decir el control lleno, que toma la pelota quitando todos sus grados de libertad. Los ejemplos para la propiedad de la pelota incluyen la fijación de una pelota al cuerpo del robot, el rodeo de una pelota que usa el cuerpo del robot para prevenir el acceso por otros, rodeando la pelota o de algún modo atrapando la pelota con cualquier parte del cuerpo del robot. Si una pelota deja de rodar mientras un robot se mueve o una pelota no rebota, esto es una indicación buena que la pelota es atrapada. La única excepción a la propiedad es el empleo de un tambor rotativo que imparte la vuelta dinámica trasera sobre la pelota para guardar la pelota sobre su superficie. Llaman un drible a tal dispositivo.

Los otros jugadores deben ser capaces de tener acceso a la pelota. [10]

2.5.6 La Anotación

Un punto es anotado cuando toda la pelota está dentro del pórtico, y esta golpea la pared trasera de la cancha.

2.5.7 Falta de progreso

La falta de progreso ocurre si no hay ningún progreso en el juego durante un período razonable de tiempo y la situación probablemente no cambia. La falta típica de situaciones de progreso es cuando la pelota es golpeada entre robots o entre el robot y la pared o ningún robot es capaz de descubrir la pelota en su posición. El árbitro se llamará " la falta de progreso " y moverá la pelota al punto desocupado neutro más cercano. Si esto no soluciona la falta de progreso, el árbitro puede mover la pelota a manchas diferentes neutras.

2.5.8 Robots Dañados

Si un robot es dañado, este tiene que ser retirado del campo y debe ser arreglado antes de que este pueda jugar otra vez. Un robot dañado debe permanecer en el campo durante al menos un minuto.

Un robot es dañado especialmente cuando:

- Este no responde a la pelota.
- Que el agente está continuamente moviéndose en el mismo objetivo.
- Que el agente se vuelque o que quede pegado a una pared o una esquina y no pueda liberarse rápidamente.

Después de que un robot ha sido arreglado será colocado sobre el punto desocupado neutro más cerca de donde ha sido su salida, y directamente no apuntando hacia a la pelota. Un robot sólo puede ser devuelto al campo si el daño ha sido reparado. Sólo el árbitro decide si un robot es dañado. Un robot sólo puede ser retirado o devuelto con el permiso del árbitro.

2.5.9 Defensa Múltiple

La defensa múltiple ocurre si más de un robot del equipo de defensa entra en su área de castigo con alguna parte y esta afecta considerablemente el juego. El robot que este más lejos de la pelota será movido al centro al punto neutro. Si la múltiple defensa pasa repetidamente, el robot será considerado dañado.

2.5.13 Interrupción de Juego

En principio, un juego no será parado. El árbitro puede parar el juego si hay una situación sobre o alrededor del campo del cual el árbitro quiere hablar con un funcionario del torneo o el mal funcionamiento de pelota y un reemplazo no está fácilmente disponible. Cuando el árbitro ha parado el juego, todos los robots deben ser parados y permanecer sobre el campo intacto.

2.6. El CÓDIGO DE CONDUCTA

2.6.1 Juego limpio

Se espera que el objetivo de todos los equipos sea jugar un juego limpio, el juego de fútbol de robot. Se espera que todos los robots sean construidos con la consideración a otros participantes. No se permiten a los robots para interferencia deliberada con o el daño a otros robots durante el juego normal . No se permiten a los robots causar daño al campo o la pelota durante el juego normal. No permiten a la gente causar la interferencia deliberada con robots o daño al campo o la pelota.

2.6.2 Compartir

Uno de requisitos que ha sido parte de mundo RoboCup y competiciones Robo CupJunior es que acontecimientos tecnológicos y curriculares deberán ser compartidos con otros participantes durante y después de la competición.

2.6.3 El Espíritu

Se espera que todos los participantes, estudiantes, mentores y padres igualmente, respetaran la misión de RoboCupJunior. ¡No es si usted gana o pierde, lo importante es cuánto usted aprende!.

2.6.4 Violaciones/Descalificaciones

Los equipos que violan el código de conducta pueden ser descalificados del torneo. Es también posible descalificar y excluir de manera remota la participación en el torneo sólo de una persona o un solo robot. En los casos menos severos, las violaciones del código de conducta, darán a un equipo una advertencia, mostrándole una tarjeta amarilla. En los casos severos o repetidos de las violaciones del código de conducta un equipo puede ser descalificado inmediatamente sin una advertencia al mostrarle la tarjeta roja.

2.7. LA RESOLUCIÓN DE CONFLICTO

2.7.1 Árbitro y Árbitro Asistente

Todas las decisiones durante el juego son tomadas por el árbitro o el árbitro asistente, quien es responsable de la mesa, el campo, y las personas y objetos que lo rodean. Durante el juego, las decisiones de los árbitros son finales. Cualquier argumento con un árbitro o el ayudante puede causar una advertencia. Si el argumento sigue u otro argumento ocurre, esto puede causar la descalificación inmediata del juego. En la conclusión del juego, el árbitro pedirá a los capitanes firmar la hoja del encuentro. Por haber firmado la hoja del encuentro los capitanes aceptan la cuenta final de parte del equipo entero.

2.7.2 La Regla de clarificación

La Regla de clarificación puede ser hecha por los miembros del Fútbol RoboCupJunior el Comité Técnico, si fuera necesario aún durante un torneo.

2.7.3 modificación de Reglas

Si circunstancias especiales, como problemas imprevistos o capacidades de un robot ocurren, las reglas pueden ser modificadas por los miembros del Fútbol RoboCupJunior el Comité técnico, si fuera necesario aún durante un torneo.

“Basada en la normatividad por robocup del documento http://www.ai.rug.nl/robocupathome/documents/rulebook2010_FINAL_VERSION.pdf”