



**UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN**

Desarrollo de un prototipo de robot humanoide que busque, encuentre y patee una pelota de forma autónoma e inteligente

Por:
Jennifer Dos Reis De Nóbrega
Juliana León Quinteiro

Realizado con la asesoría de:
Ivette Carolina Martínez

PROYECTO DE GRADO
Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar al título de
Ingeniero en Computación

Sartenejas, Noviembre 2014

Resumen

RoboCup [201a] es una competencia de fútbol, iniciada en 1997, donde contribuyen las áreas de robótica, investigación e inteligencia artificial. Entre sus categorías se encuentra RoboCup Soccer [201b], la cual consiste en la participación de pequeños robots humanoides que se enfrentan a otro equipo para jugar fútbol. El objetivo de esta competencia es lograr que en el año 2050 el equipo campeón logre vencer al ganador del año en la copa mundial de la FIFA (International Federation of Association Football).

En este proyecto se describe la construcción de Junny, un robot humanoide autónomo e inteligente de tamaño pequeño (38 cm de alto), capaz de detectar la ubicación de una pelota y acercarse a ella para patearla. Sus metas se enmarcan parcialmente dentro de las reglas de la categoría RoboCup Soccer.

Junny ha sido construido con las piezas del kit Bioloid Premium [INCb] del fabricante ROBOTIS [INCc]. Del kit se ha excluido la tarjeta CM-510 para sustituirla por la tarjeta controladora Arbotix, que será la que controle los 16 motores Dynamixel Ax-12+ (para mover al robot) y 2 servomotores analógicos (para mover la cámara). Además se ha agregado un mini computador Raspberry Pi, con su cámara [112a], para que el robot pueda detectar la posición de la pelota de forma autónoma.

En la Raspberry Pi se usa el lenguaje C++ y se ejecuta un solo programa encargado de detectar la posición de la pelota y decidir qué movimientos son necesarios para llegar a ella. La manera de elegir las acciones se ha realizado con aprendizaje por reforzamiento. Para captar la imagen de la cámara se ha utilizado la librería raspicam_cv [Val]. Para filtrar y procesar la imagen se ha usado detección de regiones por color, con ayuda de las librerías OpenCv [Tea].

La Arbotix, además de controlar los motores para ejecutar los movimientos deseados, se encarga de monitorizar que el robot se encuentre balanceado, para ello usa el sensor Gyro de Robotis [INCa]. Si detecta un desbalance de un cierto tamaño puede saber si se ha caído y levantarse.

Todos estos componentes deben ser coordinados para que se logre cumplir la tarea de seguir y patear la pelota. Por ello se hizo necesaria la comunicación entre la Arbotix y la Raspberry Pi. La herramienta empleada para ello ha sido el framework ROS (Ros Operating System) [Ros].

Agradecimientos

AGRADECIMIENTOS

Índice general

Índice general	viii
Índice de cuadros	x
Índice de figuras	xi
1. Introducción	1
2. Marco teórico	5
2.1. Robótica	5
2.2. Robótica Inteligente (Agentes Inteligentes)	6
2.2.1. Paradigmas de robótica	7
2.2.1.1. Paradigma Jerárquico	7
2.2.1.2. Paradigma Reactivo	7
2.2.1.3. Paradigma Híbrido	8
2.3. Inteligencia Artificial	8
2.3.1. Aprendizaje de Máquinas	8
2.3.2. Aprendizaje por reforzamiento	9
2.3.3. Q- learning	9
2.4. Visión Artificial	10
2.4.1. Filtros	11
2.4.2. Transformaciones Morfológicas	11
2.4.2.1. Dilatación	11
2.4.2.2. Erosión	12
3. Construcción de un Robot Humanoide	14
3.1. Diseño y Construcción	14

3.1.1.	Componentes de hardware	14
3.1.2.	Construcción	19
3.2.	Detección de la pelota	24
3.2.1.	Herramientas software para la detección	24
3.2.2.	Obtención de la imagen	25
3.2.3.	Procesamiento de la imagen	25
3.3.	Busqueda y Pateo	26
3.3.1.	Herramientas software para la busqueda de la pelota	27
3.3.2.	Comportamiento	27
3.3.3.	Movimiento del esqueleto	28
3.3.4.	Movimiento de la cámara	29
3.3.5.	Representacion del mundo en estados	29
3.3.6.	Comunicación Arbotix - Raspberry (ROS)	31
3.4.	Aprendizaje	32
3.4.1.	Modelo Apredizaje	33
3.4.1.1.	Estado	33
3.4.1.2.	Acciones	33
3.4.1.3.	Medida de desempeño	34
3.4.1.4.	Eleción de la acción	35
4.	Consideraciones Especiales: Obstaculos y Soluciones	36
5.	Experiementos y Resultados	37
6.	Conclusiones y Recomendaciones	40
Bibliografía		41
A. Archivos intermedios		44

Índice de cuadros

Índice de figuras

2.1. Dilatación	12
2.2. Erosión	13
3.1. Bioloid Premium Kit	15
3.2. Motores Dynamixel conectados en serie	15
3.3. Sensor Gyro	16
3.4. Chip FTDI conectado a la tarjeta Arbotix	16
3.5. Extensor de puertos Bioloid	17
3.6. Micro Servo motor analógico TG9e	17
3.7. Tarjeta Raspberry Pi con descripción de los puertos	17
3.8. Cámara Raspberry Pi	18
3.9. Batería Lipo	18
3.10. Vista frontal del robot, tipo B, tomada del manual. Se puede apreciar la identificación ‘ID’ de cada motor Dynamixel Ax-12+. Nota: los motores 9 y 10 no se utilizan.	19
3.11. Vista trasera del robot con la tarjeta CM-510, tomada del manual del kit Bioloid.	20
3.12. Manual de instrucciones y piezas del robot	20
3.13. Vista trasera del robot con la Arbotix	21
3.14. Cámara Raspberry Pi conectada al puerto CSI de la tarjeta	21
3.15. Vista delantera del robot con la cámara y servomotores instalados	22
3.16. Tarjeta controladora Arbotix y componentes conectados	23
3.17. Circuito con entrada de 11.1 V. Una salida de 5 V para los micro servomotores analógicos y tarjeta Raspberry Pi. Otra salida de 11.1 V para alimentar la controladora Arbotix.	24
3.18. Posiciones de la cámara	30

3.19. Campo de visión del robot con el número de cada región. Cada cuadro blanco demarcado con líneas negras representa la posición de la cámara. La región de pateo de la pelota se encuentra en los cuadros 15 y 16	31
3.20. Estados definidos según la posición de la cámara	33
3.21. Distancias relativas de la pelota establecidas para otorgar la recompensa. . .	35

Capítulo 1

Introducción

RobotCup [201a] es una competencia de fútbol iniciada desde 1997 donde contribuyen las áreas de robótica, investigación e inteligencia artificial. Entre sus categorías se encuentra RobotCup Soccer [201b], la cual consiste en la participación de pequeños robots humanoides que se enfrentan a otro equipo para jugar fútbol. El objetivo de esta competencia es lograr que en el año 2050 el equipo campeón logre vencer al ganador del año en la copa mundial de la FIFA (International Federation of Association Football). Algunas destrezas de robots con forma de humanos son caminar, percibir el mundo y tomar alguna acción sobre él. Una de las más avanzadas muestras en el área es el robot ASIMO [Co11], creado por la compañía Honda, cuyos últimos avances incluyen la predicción de trayectoria de objetos para poder esquivarlos.

El desarrollo de esta investigación presenta un robot humanoide (Junny) autonomo e inteligente de tamaño pequeño (38 cm de altura) cuyos objetivos estan basados en las reglas de la competencia RobotCup. En investigaciones con este mismo enfoque se puede encontrar el trabajo de Sven Behnke cuyo título es “See, walk, and kick: Humanoid robots start to play soccer” donde se describe la construcción del equipo de robots que participaron en la RobotCupSoccer en el año 2006. El artículo cubre el diseño mecánico y electrónico, además el

software utilizado para la percepción, control de comportamiento, comunicación y simulación de los robots. [BSS⁺].

Existen equipos que han participado durante varios años consecutivos en la competencia Robocup, logrando mejoras en sus diseños y técnicas; tal es el caso del equipo MRL que ha participado en los años 2011, 2012, 2013 y 2014 en la categoría “Humanoid League”, han iniciado con el hardware del robot DARwIn-OP y con el tiempo han modificado los componentes electrónicos para agregar eficiencia y estabilidad. Para el balance han utilizado un giróscopio y sensores de aceleración, y para la visión una cámara conectada por usb al CPU principal [SSA⁺].

En el desarrollo de habilidades más específicas con respecto a la competencia Robot-Cup Soccer, en el artículo de investigación de Seung-Joon Yi, Stephen McGill y Daniel D. Lee [YML], se refieren a dos posibles estrategias para el pateo de la pelota donde los factores fundamentales para un buen desempeño es la fuerza y la rapidez con que se patea, los investigadores ponen en práctica dos estrategias de pateo en distintas circunstancias del juego basado en la cinemática y dinámica de equilibrar el cuerpo al momento de realizar el pateo.

Se planteo como objetivo de este proyecto construir un prototipo de robot humanoide que sea capaz de detectar la cercanía de una pelota, acercarse a ella y patearla, reincorporandose a la posición de pie en caso de perder el equilibrio y caer mientras camina. Para cumplir con este objetivo se ha desglosado un conjunto de objetivos específicos que se describen a continuación:

1. Diseño y construcción de un humanoide con piezas del kit de robótica Bioloid Premium, sustituyendo su tarjeta controladora CM-510 por la tarjeta de software libre ArbotiX para controlar los motores Dynamixel y otros sensores.
 - a) Instalación y configuración de la tarjeta ArbotiX.
 - b) Instalación y configuración de la tarjeta Raspberry Pi.

- c) Instalación y configuración de la cámara Raspberry Pi.
- d) Instalación de servomotores para el movimiento de la cámara
- e) Instalación del giroscopio Gyro.

2. Detección de la pelota

- a) Captura de imagen con la cámara Raspberry Pi a través de la librería raspicam cv.
- b) Procesamiento de la imagen para extraer información de la posición de la pelota con las librerías de OpenCV.

3. Búsqueda de la pelota y pateo de la misma.

- a) Creación de las poses necesarias para caminar, girar, levantarse y patear usando el software pypose.
- b) Programación de transiciones de movimientos.
- c) Control de servomotores para el movimiento de la cámara.
- d) Establecer mecanismo de comunicación entre la tarjetas ArbotiX y Raspberry Pi.
- e) Programación de algoritmo de planificación de acciones que lleve al humanoide a acercarse a la pelota.
- f) Detección de movimientos angulares bruscos que sugieran una caída, a través de la lectura del giroscopio
- g) Identificación del momento en que la pelota se encuentre en una zona adecuada para patear.

La siguiente investigación se divide en 5 capítulos. El capítulo 2 se refiere a una base teórica de conceptos e información necesarias para el soporte del desarrollo de este proyecto

que se encuentra en el capítulo 3 donde se explica todo el procedimiento realizado para llevar a cabo el producto,. El capítulo 4 describe la serie de consideraciones importantes que se deben tener a la hora de reproducir esta investigación como los problemas claves y las soluciones que se les dio. En el capítulo 5 se encuentran los experimentos y sus resultados. Por último estan las conclusiones y recomendaciones en la sección 6.

Capítulo 2

Marco teórico

En este capítulo se presentan los conceptos que conforman la base teórica para comprender el presente trabajo. Primero se brinda una descripción de los términos relativos a la robótica y las partes principales de un robot. Posteriormente se describen algunos conceptos que tienen que ver con la robótica inteligente, como los paradigmas, la inteligencia y la visión artificial para la detección de objetos.

2.1. Robótica

El presente trabajo se basa en la construcción de un robot humanoide, por lo tanto es importante definir qué es un robot, qué significa que sea humanoide y cuáles son algunos de sus componentes principales.

- **Robótica:** Es la rama de la tecnología que se encarga del diseño, construcción, operación y aplicación de los robots. [Pre14]
- **Robot:** Son agentes físicos que ejecutan tareas para manipular el mundo físico. Para ello deben estar equipados con actuadores y sensores [pet09]. La apariencia no es una

característica útil para la definición de un robot [AiR00], por lo tanto puede ser de diferentes formas, ya sea con ruedas, con piernas o ninguna de ellas. Una de las formas que puede adoptar un robot es la de humano, de hecho en la cultura popular el término “robot” generalmente connota una apariencia humana [AiR00]. Según el diccionario de la Universidad de Oxford, el término humanoide se refiere a tener una apariencia o característica parecida al de un ser humano [Pre14], por lo tanto a los robots con forma de humano se les denomina robots humanoides.

- **Sensores:** Son los dispositivos encargados de percibir el ambiente que rodea al robot. Según Murphy R.R estos miden algún atributo del mundo. Un sensor recibe energía del entorno (sonido, luz, presión, temperatura, etc) y transmite una señal a una pantalla o computador ya sea de forma análoga o digital [AiR00]. Algunos sensores son: cámaras, giroscopios, sensores de proximidad, entre otros.
- **Actuador:** Es aquella parte del robot que convierte comandos de software en movimientos físicos. Por ejemplo ruedas, piernas, pinzas, entre otros [pet95].
- **Servomotor:** Es un motor eléctrico, considerado como actuador, que permite ser controlado tanto en velocidad como en posición [AiR00].
- **Giróscopio:** Es un sensor utilizado para medir y mantener la orientación, se mide a través del momento angular [Con14].

2.2. Robótica Inteligente (Agentes Inteligentes)

Es importante diferenciar cuando un robot es inteligente o no. Cuando un robot es operado a distancia, y no es capaz de cumplir sus tareas sin la intervención de un humano, entonces no se considera como inteligente. Tampoco se considera inteligente si las tareas que ejecuta

se hacen sin sentido o de manera repetitiva. En cambio cuando un robot puede interactuar con el mundo de manera autónoma se considera que es un robot o agente inteligente [AiR00]. Existen diferentes estrategias o enfoques de cómo aplicar la inteligencia en un robot. Esta sección se dedica a describir los enfoques que en [AiR00] se definen como paradigmas.

2.2.1. Paradigmas de robótica

Según Robin Murphy en [AiR00], existen tres paradigmas en los cuales se clasifica el diseño de un robot inteligente, estos paradigmas pueden ser descritos de dos maneras: la relación entre las primitivas básicas de la robótica: percibir, planificar, actuar; o de la forma en que los datos son percibidos y distribuidos en el sistema.

Percibir se refiere al procesamiento útil de la información de los sensores del robot. Planificar, cuando con información útil, se crea un conocimiento del mundo y se generan ciertas tareas que el robot podría realizar. Por último actuar consiste en realizar la acción correspondiente con los actuadores del robot para modificar el entorno.

2.2.1.1. Paradigma Jerárquico

Este paradigma es secuencial y ordenado. Primero el robot percibe el mundo y construye un mapa global. En base al mapa ya percibido y con “los ojos cerrados”, el robot planifica todas tareas necesarias para lograr la meta. Luego ejecuta la secuencia de actividades según la planificación realizada. Una vez culminada la secuencia se repite el ciclo percibiendo el mundo, planificando y actuando [AiR00].

2.2.1.2. Paradigma Reactivo

El paradigma reactivo omite por completo el componente de la planificación y solo se basa en percibir y actuar. El robot puede mantener un conjunto de pares percibir-actuar,

estos son llamados comportamientos y se ejecutan como procesos concurrentes. Un comportamiento toma datos de la percepción del mundo y los procesa para tomar la mejor acción independientemente de los otros procesos [AiR00].

2.2.1.3. Paradigma Híbrido

El paradigma híbrido es una mezcla de los dos paradigmas anteriores. Primero se planifica cuál es la mejor manera de cumplir el objetivo principal, descomponiendo la tarea general en sub-tareas y decidiendo que comportamientos sirven para cumplir cada una. De allí en adelante se ejecutan los comportamientos (percibiendo y actuando), hasta que el plan sea ejecutado, si es necesario se puede volver a planificar. Vale la pena acotar que la información de los sensores se encuentra disponible para el planificador, de manera que pueda crear un modelo del mundo y tomar decisiones en base a él [AiR00].

2.3. Inteligencia Artificial

La inteligencia artificial es un término relacionado con la computación que puede ser aplicado a la robótica para crear robots inteligentes. El término “inteligencia artificial” ha tenido varias definiciones. Ocho de ellas, las cuales nacieron a finales del siglo XX, se encuentran organizadas en [pet95] bajo cuatro categorías: pensar y actuar de forma humana, pensar y actuar de forma racional. De ellas se puede entender que la inteligencia artificial tiene que ver con lograr que una máquina o un robot resuelva problemas de manera inteligente, es decir, de manera que parezca que el razonamiento y comportamiento humano las ha resuelto.

2.3.1. Aprendizaje de Máquinas

El aprendizaje de máquinas es un área de la inteligencia artificial que está relacionada con la pregunta de cómo construir programas de computadora que automáticamente mejoren

con la experiencia. Se dice que un programa aprende de la experiencia E con respecto a una tarea T y desempeño P si el desempeño en la tarea T, medido por P, mejora con la experiencia E [Mit97].

2.3.2. Aprendizaje por reforzamiento

El aprendizaje por reforzamiento es un tipo de aprendizaje de máquinas que se basa en un sistema de recompensas positivas y negativas. Las recompensas se pueden dar en cada estado o una sola vez al llegar al estado final.

El objetivo del agente es aprender de las recompensas para escoger la secuencia de acciones que produzca la mayor recompensa acumulada. [Mit97]

El agente existe en un entorno descrito por algunos estados S. Puede ejecutar un conjunto de acciones A. Cada vez que ejecuta una acción a_t en algún estado s_t el agente recibe una recompensa r_t . El objetivo es aprender una política $\pi : S \rightarrow A$ que maximice la suma esperada de esas recompensas con descuento exponencial de las recompensas futuras. [Mit97] El resultado de tomar las acciones puede ser determinista o no, en el caso de este proyecto no es determinista, es decir, existen porcentajes de probabilidad de pasar a un estado u otro al tomar una acción en un estado en particular.

2.3.3. Q- learning

Es un método de aprendizaje por reforzamiento que, dado un estado, compara las utilidades esperadas de las posibles acciones a tomar sin necesidad de saber el estado resultante, por tanto no se necesita tener un modelo del entorno [pet95].

La forma de aprender la política $\pi : S \rightarrow A$ es de forma indirecta, a través de la función $Q(s, a)$. La función representa el valor de la máxima recompensa acumulada, con descuento de las recompensas futuras, que puede ser alcanzada desde el estado s y aplicando a como la

primera acción [Mit97]. Se encuentra definida como:

$$Q(s, a) = r(s, a) + \gamma V^*(\delta(s, a))$$

En donde $r(s, a)$ es la recompensa o castigo dado según el resultado de haber tomado la acción a en el estado s . $\delta(s, a)$ es el estado obtenido luego de tomar la acción a en el estado s . γ es el descuento que se le aplica a las recompensas futuras. La función $V^*(s')$ genera el máximo valor Q que puede ser alcanzado desde el estado s' . Esto es,

$$V^*(s') = \max_{a'}(Q(s', a'))$$

De esta forma obtenemos una definición recursiva,
que puede ser calculada de manera iterativa, actualizando los valores una vez tomada la acción y observando el estado resultante que desencadenó.

2.4. Visión Artificial

Una manera de obtener información del ambiente es con la visión artificial. Esta consiste en usar un dispositivo (cámara) que capta el espectro electromagnético y produce una imagen. La representación de la imagen se almacena como una matriz de píxeles, cada píxel es un elemento que guarda información de una región en el espacio captado. Si se usa una cámara de luz, la información de cada píxel será el color. [AiR00]

Por lo general luego de obtener una imagen se requiere extraer información de ella, por lo cual se han desarrollado diferentes algoritmos que ayudan en esta tarea. Existen varios algoritmos que se dedican a la transformación de las imágenes para reducir ruidos, compensar problemas de iluminación, extraer formas, identificar objetos, entre otros. En esta sección se

describen dos de las técnicas de transformación para reducir el ruido basadas en la dilatación y erosión de la imagen.

2.4.1. Filtros

El filtrado de imágenes es una técnica para la transformación de imágenes, que consiste en destacar sus características más relevantes en base a un propósito en particular.

Generalmente en la tarea de extracción de información de una imagen se utilizan filtros para descartar zonas o características que no son importantes para el patrón deseado y para determinar el área deseada ya sea por patrones de forma o color.

En la investigación, los algoritmos de filtrado aplicados a las imágenes fueron: Clausura Morfológica y Apertura Morfológica, filtros que aplican las técnicas de erosión y dilatación a las imágenes.

2.4.2. Transformaciones Morfológicas

Las transformaciones morfológicas básicas son llamadas dilatación y erosión, se utilizan en amplia variedad de contextos como la eliminación del ruido, aislamiento de elementos individuales y elementos de unión dispares en una imagen. [Boo08]

2.4.2.1. Dilatación

La dilatación es una convulsión (algo que se le aplica a toda la imagen) entre alguna imagen (o región de una imagen), que llamaremos A y un núcleo que llamaremos B, el núcleo, que puede ser de cualquier forma o tamaño, tiene un solo punto de anclaje definido. Para mayor claridad el nucleo es esencialmente una matriz de tamaño fijo de coeficientes numéricos junto con un punto de anclaje en dicha matriz, que normalmente se encuentra en el centro . Muy a menudo, el núcleo es un pequeño cuadrado o disco sólido. El núcleo puede ser pensado como

una plantilla o mascara, y su efecto para la dilatación tal como un operador de máximo local sobre la imagen, se calcula el máximo valor de los píxeles común a B y reemplazamos el píxel de la imagen en el punto de anclaje con ese valor máximo. Esto causa regiones brillantes dentro de una imagen y la hacen crecer. Este crecimiento es el origen del término “operador de dilatación” [Boo08].

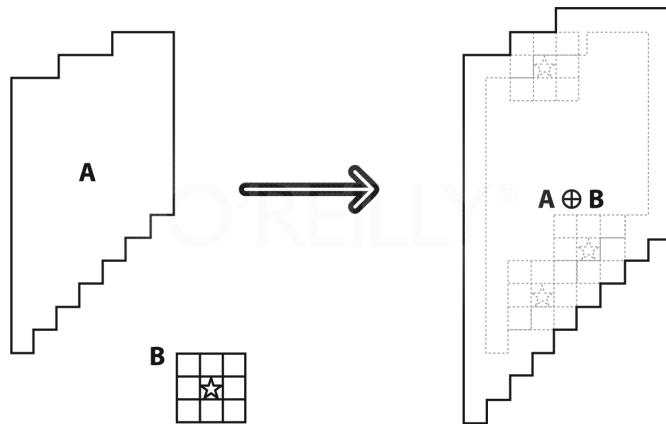


Figura 2.1: Dilatación

2.4.2.2. Erosión

La erosión es la operación inversa a la dilatación. Esta acción del operador es equivalente a la erosión el cálculo de un mínimo local sobre el área del núcleo. La erosión genera una nueva imagen a partir de la original, utilizando el siguiente algoritmo: como el núcleo B es analizado sobre la imagen, se calcula el mínimo valor del píxel superpuesto por B y se reemplaza el píxel de la imagen con un punto de anclaje de valor mínimo [Boo08]. Vease en la figura 2.2

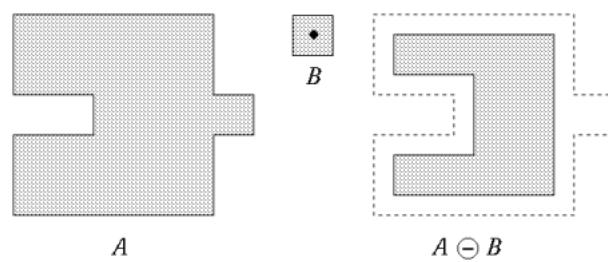


Figura 2.2: Erosión

Capítulo 3

Construcción de un Robot Humanoide

En el presente capítulo se describe todas las actividades que se han llevado a cabo para lograr construir un robot humanoide capaz de detectar la ubicación de una pelota, de un color determinado, buscarla, y al llegar a ella patearla. También se describen las actividades realizadas para lograr que en caso de tener un desbalance y caer, sepa levantarse.

3.1. Diseño y Construcción

El primer paso ha sido la construcción de la parte física del robot. Para tal fin se procedió a la elección del diseño y ensamblaje de las piezas. En la sección 3.1 se describe cada uno de los componentes utilizados para armar el robot, y luego, en la sección 3.1.1 se explica cómo se integraron esas piezas para obtener un humanoide adaptado a los objetivos de este proyecto.

3.1.1. Componentes de hardware

A continuación se presenta una descripción de todos los elementos utilizados para la construcción del robot humanoide.

- Bioloid Premium kit: Es un kit de robótica con piezas modulares que permite armar

diferentes tipos de robot pero principalmente humanoides. Su empaque se puede observar en la figura 3.1. El fabricante, ROBOTIS, incluye un manual con varios modelos de robots con instrucciones de ensamblaje. Provee una tarjeta controladora, CM-510, a la que se conectan los motores Dynamixel y algunos sensores que se programan a través de la interfaz de ‘RoboPlus’ [INCb].



Figura 3.1: Bioloid Premium Kit

- Motores Dynamixel Ax-12+: Son actuadores inteligentes y modulares que incorporan un reductor de engranajes, un motor DC de presión y un circuito de control con funcionalidad de red lo cual permite formar series o cadenas de motores (figura 3.2), todo en un solo paquete [INC06].

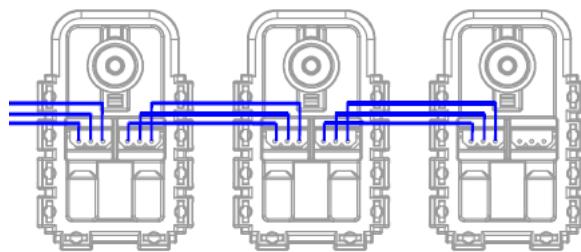


Figura 3.2: Motores Dynamixel conectados en serie

- Gyro: Es un giroscopio de la marca Robotis que mide la velocidad angular. Se encuentra diseñado para mantener el balance del robot y ser usado para otras aplicaciones de movimiento [INCa]. En figura 3.3 se puede observar su estructura.



Figura 3.3: Sensor Gyro

- Arbotix: El controlador ArbotiX es una solución de control avanzado para manejar algunos tipos de servos Dynamixel y robots basados en Bioloid. Incorpora un potente microcontrolador AVR, radio inalámbrica XBEE, conductores de motor dual, y cebaderas de estilo servo de 3 pines para entrada/salida digital y analógica [LLC].
- FTDI (Future Technology Devices International) : Es una tarjeta controladora (figura 3.4) que ofrece el servicio de conversión de datos de USB a UART. Permite la comunicación entre diferentes dispositivos [Ele].



Figura 3.4: Chip FTDI conectado a la tarjeta Arbotix

- Extensor de puertos Bioloid : Permite aumentar el número de cadenas de servos conectados a la tarjeta (ver figura 3.5) [WL].
- Micro servo motor analógico TG9e: Es un pequeño servomotor cuya capacidad de torque alcanza los 1.50 kg-cm [Hob]. Permite ser controlado en posición en un rango de 180°. Ver figura 3.6.



Figura 3.5: Extensor de puertos Bioloid



Figura 3.6: Micro Servo motor analógico TG9e

- Raspberry Pi: La Raspberry Pi es un ordenador del tamaño de una tarjeta de crédito a la que se puede conectar un televisor y un teclado. Se trata de un pequeño ordenador capaz de ser utilizado en proyectos de electrónica y para muchas de las tareas que una PC de escritorio hace, como hojas de cálculo, procesadores de texto y juegos [112b]. Ver figura 3.7.



Figura 3.7: Tarjeta Raspberry Pi con descripción de los puertos

- Cámara Raspberry Pi: Es un sensor encargado de captar imágenes y grabar videos de alta definición. Se conecta a la Raspberry Pi con un cable de cinta plana de 15 cm en el puerto CSI. Tiene 5 megapíxeles de foco fijo que soporta los modos de video de 1080x30, 720x60 y VGA90. Puede ser manejada con las librerías MMAL, V4L u otras librerías de terceros como la de Python [112a].(figura 3.8)



Figura 3.8: Cámara Raspberry Pi

- Batería de polímero de litio (Lipo): Es la fuente de poder usada para los motores y componentes electrónicos. La batería usada es de 11.1 voltios y 1 amperio. [Rob]



Figura 3.9: Batería Lipo

- Circuito con regulador de 5v: Es un circuito diseñado y construido para este proyecto cuya finalidad es regular la entrada de la corriente. Por una de las salidas se expulsa 5v y por la otra se mantiene el mismo voltaje de entrada.

3.1.2. Construcción

Para la construcción del robot se utilizó el kit de piezas Bioloid Premium de marca Robotis el cual incluye motores Dynamixel Ax-12+, una tarjeta controladora CM-510, un sensor Gyro, un manual, entre otros elementos. El manual incluye las instrucciones de como armar varios modelos de humanoide, el utilizado en este proyecto es el tipo B, haciendo uso de 16 motores. En las figuras 3.10 y 3.11 se puede observar la estructura del robot que aparece en el manual del kit.

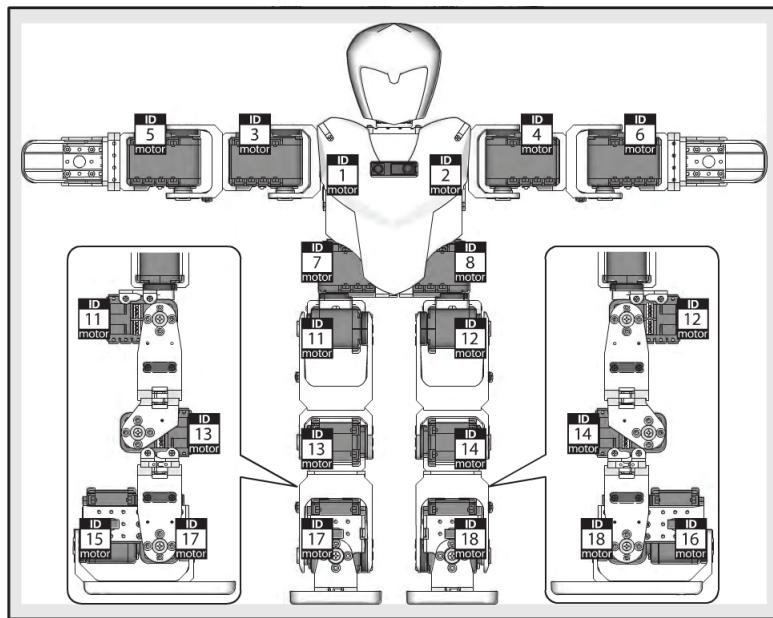


Figura 3.10: Vista frontal del robot, tipo B, tomada del manual. Se puede apreciar la identificación ‘ID’ de cada motor Dynamixel Ax-12+. Nota: los motores 9 y 10 no se utilizan.

El kit bioloid incluye una tarjeta controladora CM-510 la cual fue sustituida por la tarjeta controladora de software libre Arbotix. La utilización de la tarjeta Arbotix permite una mayor flexibilidad en el control de motores y la incorporación de una variedad de sensores no soportados por la tarjeta CM-510. Además, la tarjeta Arbotix posee mayor soporte y amplitud en la comunicación entre distintos dispositivos.

En la figura 3.13 se puede observar la estructura del robot con la Arbotix incorporada.

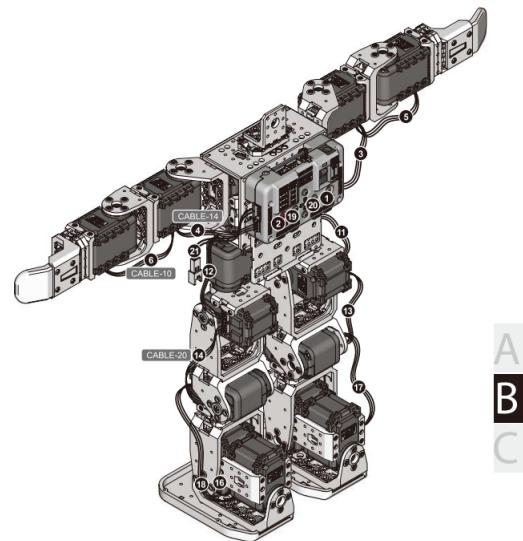


Figura 3.11: Vista trasera del robot con la tarjeta CM-510, tomada del manual del kit Bioloid.



Figura 3.12: Manual de instrucciones y piezas del robot

En la parte interna del tronco del robot se sitúa el sensor Gyro.

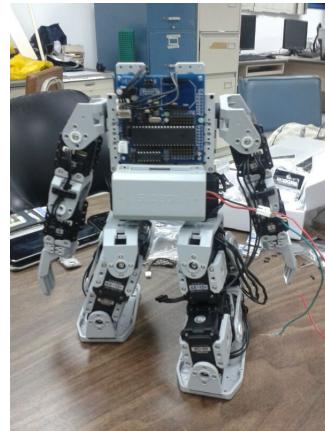


Figura 3.13: Vista trasera del robot con la Arbotix

Para el movimiento de la cámara se ha incorporado dos micro servomotores, uno para el movimiento horizontal y otro para el vertical. La conexión de uno de estos motores se ilustra en la figura 3.16, en donde se puede observar que se encuentra conectado al puerto B de los denominados 'Hobby Servo ports'. La cámara ha sido conectada a la Raspberry Pi en el puerto CSI (ver la figura 3.14). El resultado de estas tres piezas instaladas en el robot se puede apreciar en la figura 3.15.



Figura 3.14: Cámara Raspberry Pi conectada al puerto CSI de la tarjeta

Los motores Dynamixel se conectan a la controladora Arbotix por medio de los puertos Bioloid de la tarjeta. El diseño de Junny posee cuatro extremidades: dos brazos y dos piernas,

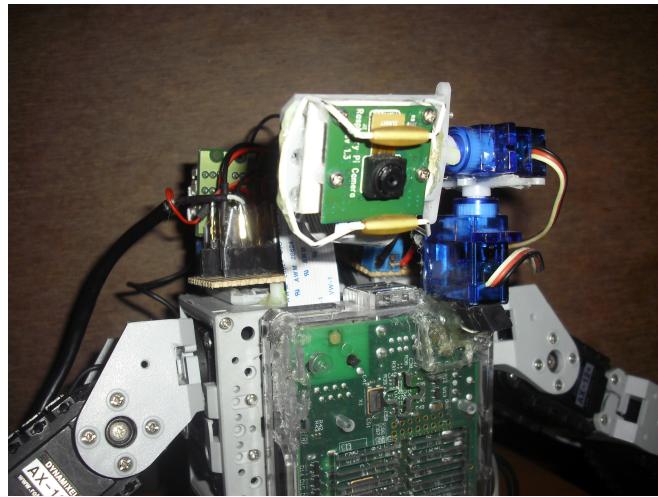


Figura 3.15: Vista delantera del robot con la cámara y servomotores instalados

por lo que naturalmente, el conjunto de motores, se puede descomponer en cuatro cadenas o series separadas. Sin embargo la Arbotix sólo cuenta con tres puertos Bioloid. Se consideró la opción de unir dos extremidades pero ello implicaba limitaciones en el movimiento del robot, por lo tanto se optó por agregar un expansor de puertos Bioloid y así conectar cada extremidad en un puerto diferente. La forma en la que se ha conectado estos motores se ejemplifica en la figura 3.16.

La comunicación de la tarjeta de Arbotix con la computadora, incluso con la Raspberry Pi, se realiza a través del puerto FTDI por medio un chip conectado como lo ilustra la figura 3.16.

Como fuente de poder se utilizó una batería de polímero de litio de 11.1 V y 1 amp. Debido a que no todos los componentes poseen las mismas exigencias con respecto a voltaje y amperaje, se realizó un regulador (ver figura 3.17) con una salida de 5 voltios para la tarjeta Raspberry Pi y dos micro servomotores, y otra salida de 11.1 V para la tarjeta Arbotix que a su vez alimenta a los componentes conectados en ella (motores Dynamixel y Giroscopio). Si bien la tarjeta Arbotix posee un regulador interno de cinco voltios la opción de conectar todo a la salida de 5 V del regulador no era posible, dado que los motores Dynamixel requieren

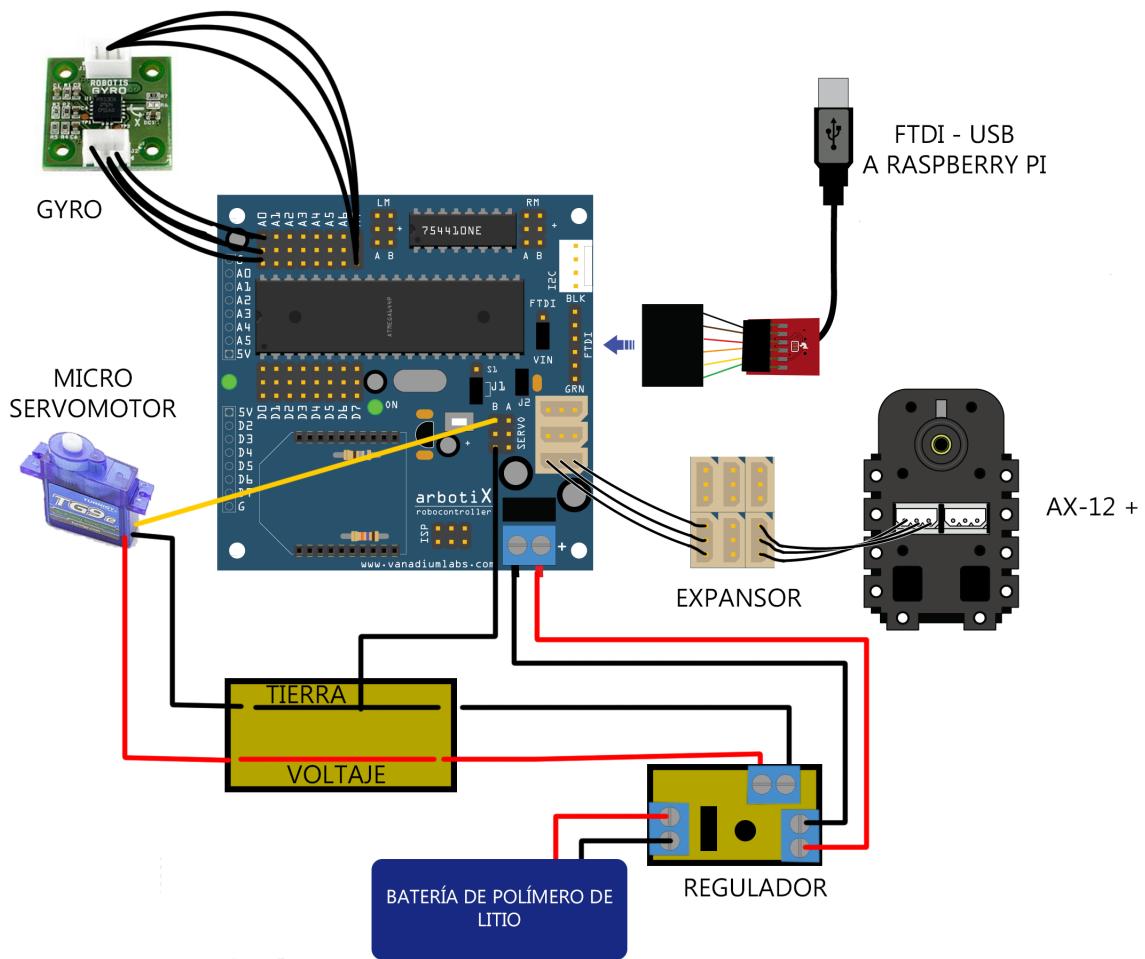


Figura 3.16: Tarjeta controladora Arbotix y componentes conectados

alimentación de 11 voltios.

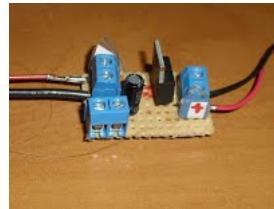


Figura 3.17: Circuito con entrada de 11.1 V. Una salida de 5 V para los micro servomotores analógicos y tarjeta Raspberry Pi. Otra salida de 11.1 V para alimentar la controladora Arbotix.

3.2. Detección de la pelota

La recopilación de información del medio ambiente, para detectar la posición de la pelota, se realizó por medio de la cámara Raspberry Pi dado que una cámara otorga mayor información y más precisa que muchos de los sensores de proximidad o contacto de bajo costo.

3.2.1. Herramientas software para la detección

Para la programación del robot se necesitaron distintas herramientas de software que permitieron ir desarrollando el comportamiento del robot. A continuación se presenta la descripción de las herramientas utilizadas.

- OpenCv (Open Source Computer Vision Library): Es una librería de visión de computadoras y aprendizaje de máquinas de código abierto. Ha sido diseñada para acelerar el uso de la percepción de máquinas y para proveer una estructura común en las aplicaciones de visión de computadoras. Registrada bajo la licencia BSD, de código abierto. [Tea]

3.2.2. Obtención de la imagen

Si decidimos dejar esta sección debemos decir que no se puede extraer la imagen de la cámara con la librería opencv por lo cual se debió extraer la imagen con ayuda de la librería raspicam_cv que devuelve una estructura de datos compatible con OpenCV. No se si ponerlo aquí o en la de obstáculos.

3.2.3. Procesamiento de la imagen

El procesamiento de las imágenes es realizado por la mini computadora Raspberry Pi sobre la cual ha sido instalado el sistema operativo Raspbian.

Con ayuda de la librería OpenCv, en C++, se filtra y procesa la imagen para obtener la posición de la pelota. Esta librería también ofrece funciones para capturar la imagen de algunos tipos de cámara, sin embargo con el módulo de cámara de la Raspberry Pi no funciona. Por ello se ha utilizado la librería raspicam cv robidouille que permite obtener la imagen en una estructura de datos que puede ser utilizado por las funciones de OpenCv.

Para encontrar la ubicación de la pelota en un momento dado y de forma autónoma se ha decidido aplicar detección por ‘blobs’, o reconocimiento de regiones, esta técnica consiste en filtrar la imagen por color, por ello es importante que el de la pelota no se repita en el ambiente y así poder obtener la posición de la pelota dentro de la imagen.

La imagen es captada en el modelo de color RGB y se transforma al HSV. Luego se aplica la función inRange de OpenCv para obtener una imagen en blanco y negro, en donde se identifica con blanco la zona con el color de la pelota y el resto de la imagen en negro.

Para disminuir el ruido y los posibles elementos aislados que pueda tener la imagen con la que se está trabajando se han aplicado los filtros o transformaciones de morfología en apertura y morfología en clausura de la librería Opencv, basadas en las operaciones básicas de dilatación y erosión. La morfología en apertura es una transformación que consiste en

aplicar la operación de erosión seguido de la operación de dilatación REF. La morfología de clausura es una transformación que aplica la dilatación seguido de la erosión.

De esta forma se logró ubicar la pelota con la cámara Raspberry Pi con buenos resultados en la mayoría de los casos.

3.3. Busqueda y Pateo

Para poder buscar y patear la pelota, además de tener la capacidad para detectarla (como se explicó en el capítulo anterior), debe ser capaz de moverse en su entorno, tener una representación del mundo que lo ayude a orientarse y planificar una estrategia para elegir el conjunto de movimientos que lo lleven a acercarse a la pelota.

En esta sección se explica el desarrollo de las actividades que han sido necesarias para ejecutar la búsqueda y pateo de la pelota, con excepción de la estrategia de toma de acciones, que se explicará en la siguiente sección.

Primero, en la sección (herramientas) se da una breve descripción de las herramientas de software que apoyaron las tareas de búsqueda y pateo. En la sección (movimientos) se explica cual fue el conjunto de movimientos creados para el esqueleto del robot.

El sensor principal de Junny, es el observador de su ambiente, la cámara. En la sección (movCamara) se explica las posiciones que puede adoptar. Luego en la sección (mundo) se explica la manera en la que Junny organiza la representación visual que capta del mundo, que se ha decidido dividir por estados.

Debido a que el movimiento del robot se controla desde la tarjeta Arbotix y la detección de la pelota se hace desde la Raspberry Pi, se debió establecer la comunicación entre ambas tarjetas. Este proceso se explica en la sección (comunicacion).

Una vez con la representación del mundo en estados, los movimientos programados y la comunicación de las tarjetas, solo faltaría decidir que acción tomar en cada estado. Esta

estrategia se explica en la siguiente sección (numero de la sección).

3.3.1. Herramientas software para la búsqueda de la pelota

Decir más o menos para qué usamos cada herramienta, breve.

- Pypose: Software especializado en el control de los servomotores Dynamixel Ax-12. Una de las más importantes características es que, luego de haber fijado a mano las posiciones de los motores, permite la lectura simultánea de esas posiciones para captar la pose del robot. Con esta herramienta es posible formar una secuencia de poses que generen un movimiento, por ejemplo, caminar [Fer10].
- IDE Arduino: Es un entorno de desarrollo para escribir y cargar código en la tarjeta Arduino. Otras tarjetas con microcontroladores AVR también son compatibles, como la ArbotiX. El lenguaje de programación del IDE de Arduino es una implementación de Wiring el cual está basado en Processing [Ard14].
- ROS: ROS (Robot Operating System) es un framework que proporciona bibliotecas y herramientas para ayudar a los desarrolladores de software a crear aplicaciones robóticas. Proporciona abstracción de hardware, de dispositivos, bibliotecas, visualizadores, paso de mensajes, gestión de paquetes y más. ROS se encuentra bajo licencia de código abierto, la licencia BSD.

3.3.2. Comportamiento

Deupa es un robot humanoide implementado de forma autónoma e inteligente que sigue un comportamiento bajo el paradigma híbrido (sección 2). El sensor principal (cámara) es el observador del mundo, que posee una serie de movimientos determinados (sección REF) con los cuales escanea el mundo y combinados con la serie de movimientos del esqueleto es

capaz de encontrar la pelota. Al determinar la posición de la pelota Debupa logró aprender (sección APRENDIZAJE) la mejor acción a relizar para estar mas cerca de ella y al llegar poder patearla.

3.3.3. Movimiento del esqueleto

El robot debe desplazarse para poder patear la pelota por ello se describen y definen los movimientos del esqueleto que se fueron utilizados.

Con fines explicativos, en este proyecto, la palabra "pose" se referire a la posición específica de los 16 motores que constituyen el esqueleto del robot. Un conjunto de poses ejecutadas en secuencia se denominará "acción de movimiento".

Las acciones de movimiento establecidas son:

- Caminar hacia adelante
- Caminar hacia adelante
- Caminar hacia adelante
- Girar a la izquierda
- Girar a la izquierda
- Girar a la derecha
- Girar a la derecha
- Levantarse cuando ha caído boca abajo
- Levantarse cuando ha caído boca arriba
- Patear con la pierna derecha

- Patear con la pierna izquierda

Existen también dos acciones de movimiento que no se encuentran relacionadas con la posición de los motores del esqueleto del robot sino a la posición de los motores que controlan la posición de la cámara. Estas se explicarán en la sección de movimiento de la cámara.

Las poses han sido fijadas a través de la tarjeta controladora Arbotix y el software Pypose. De esta manera se ha fijado y guardado un conjunto de poses para cada acción de movimiento. Estas acciones de movimientos son utilizadas en el programa, en lenguaje Wiring, a ser ejecutado en Arbotix. La programación en Arbotix se ha realizado bajo el ambiente del IDE de Arduino.

3.3.4. Movimiento de la cámara

La cámara ha sido instalada sobre dos micro servomotores analógicos, otorgándole dos grados de libertad. El servomotor ubicado en la parte inferior se encarga del movimiento horizontal y el superior del movimiento vertical. Las acciones de movimiento relacionadas con el movimiento de la cámara se reduce a 6 posiciones fijas en la figura se puede ver ?? cuya distribución obedece al objetivo de que la cámara obtenga una amplia visión, sin dejar espacios no visibles.

Estos motores se controlan desde la Arbotix usando la librería HServo. Esta librería solo puede ser usada para los motores conectados en los puertos Hobby A y B (pines 12 y 13) (ver la figura ??). Brinda la ventaja de un control más preciso, evitando que los motores tengan una vibración ya que los pulsos son generados por temporizadores de hardware.

3.3.5. Representacion del mundo en estados

El hecho de que la cámara tenga dos grados de libertad para moverse es una gran ventaja, ya que se puede obtener un mayor rango de visión. Debupa puede mirar hacia la derecha o

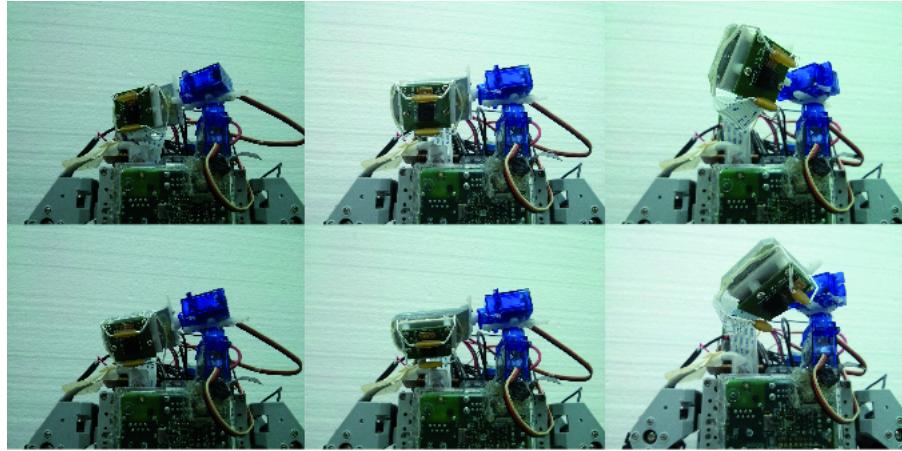


Figura 3.18: Posiciones de la cámara

izquierda sin tener que mover sus piernas, también puede mirar hacia abajo para verificar que la pelota esté en sus pies, para patear, o hacia arriba para ubicar la pelota a mayor distancia.

La estrategia para poder llegar a la pelota consiste en mover la posición de la cámara hasta encontrar la pelota, en caso de encontrarla, dependiendo de su ubicación dentro de la imagen y la posición de la cámara se toma una acción diferente, en caso de no encontrarla el robot gira con los pies para cambiar su orientación física e iniciar nuevamente el movimiento de la cámara para hallar la pelota. Cuando se tiene la pelota en una posición cercana a los pies se realiza la acción de patear.

En la siguiente sección se explicará la manera en la que se dividen las regiones en una imagen para determinar la acción a tomar y el orden en el que se mueve la cámara.

Deupa debe tomar una acción diferente dependiendo de la posición de la cámara y de la pelota en la imagen. Sin embargo esto genera una gran cantidad de estados, por lo cual se ha decidido discretizarlos de la siguiente manera:

La cámara tiene 6 (2x3) posibles posiciones. La visión horizontal abarca 3 cuadros, aproximadamente 160 grados, por razones de la estructura del robot no se le ha podido agregar un rango más grande. La visión vertical abarca 2 cuadros, llega a captar la imagen desde sus pies hasta más de 2 metros hacia adelante.

Desde cada posición de la cámara se obtiene una imagen. Las imágenes de la cámara en posición central, y en posición central inferior son las más importantes y prioritarias, pues si la pelota se detecta en ellas significa que el robot está cerca de poder patearla. Estas dos imágenes se dividen en subregiones, para tener mayor precisión en las acciones que Debupa deba tomar. Una representación sencilla de la discretización del ambiente se puede apreciar en la figura ???. El área de pateo son las regiones 15 y 16. Por ejemplo, en el cuadro del central inferior, cuando la pelota se encuentra del lado derecho de la pantalla (region 13 o 17) el robot debe girar a la derecha para situarse de frente a la pelota.

Las imágenes capturadas desde cada posición de la cámara se solapan para evitar perder de vista a la pelota.

Las acciones específicas a tomar según el estado en que se encuentre la pelota se realizan por medio de lo aprendido en el entrenamiento realizado con aprendizaje por reforzamiento. Los detalles de este aprendizaje se describen en la sección 3.4.

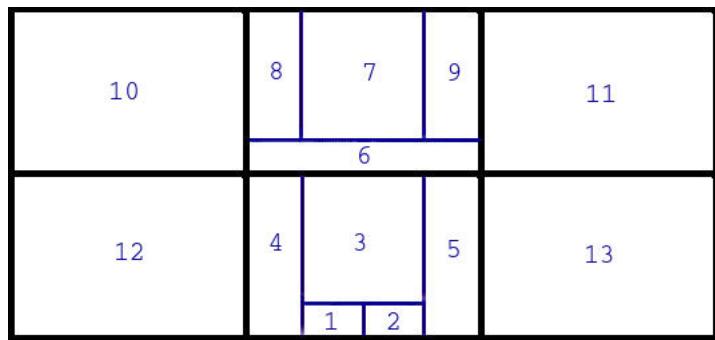


Figura 3.19: Campo de visión del robot con el número de cada región. Cada cuadro blanco demarcado con líneas negras representa la posición de la cámara. La región de pateo de la pelota se encuentra en los cuadros 15 y 16

3.3.6. Comunicación Arbotix - Raspberry (ROS)

La Raspberry Pi procesa la información de la cámara y la Arbotix controla los actuadores. Para coordinar los movimientos del robot según la posición de la pelota se estableció una

forma de comunicación entre ambas tarjetas.

Se ha establecido la Arbotix como servidor de peticiones y a la Raspberry Pi como cliente. Dentro de la Raspberry Pi se ejecuta el proceso de decidir qué acción debe tomar el robot. Una vez determinada la acción se envía la petición a la Arbotix para que esta la ejecute. Este proceso es bidireccional y síncrono, es decir, la Raspberry envía la petición y se bloquea hasta que la Arbotix retorne la respuesta de su culminación.

Para la implementación de la comunicación se ha usado ROS con su versión Hydro y se ha utilizado la interfaz de comunicación basada en servicios que no es más que un método de comunicación basado en el paradigma de resquest / reply con el concepto de maestro esclavo.

REF

3.4. Aprendizaje

En el área de inteligencia artificial asociada a robótica existen variadas técnicas que permiten que un robot pueda ”aprender.”^a realizar alguna tarea. En este caso particular se utilizó la técnica de aprendizaje por reforzamiento basado en MDP (Procesos de decisión Markovianas) que consiste en dar recompensas positivas o negativas dependiendo del desempeño del robot.

Segun el [Mit97] todo aprendizaje se define por la realizacion de una tarea T por medio de una experiencia E medido por un desempeño D . La tarea es aprender cual es la mejor accion a tomar dependiendo de un estado s para llegar a la pelota, la experiencia es generada por TANTOS experimentos realizados y el desempeño se mide con la realizacion de la accion que genera una distancia menor a la pelota.

Se utilizó el modelo de aprendizaje Q-learning para realizar la tarea definida se presenta y describe la implementación del aprendizaje utilizado a continuación.

3.4.1. Modelo Apredizaje

Definición de el modelo implementado

3.4.1.1. Estado

Un estado es definido por la posicion de la pelota, por lo tanto dada la division de las regiones de la cámara se obtienen 14 estados definidos en la siguiente imagen 3.20.

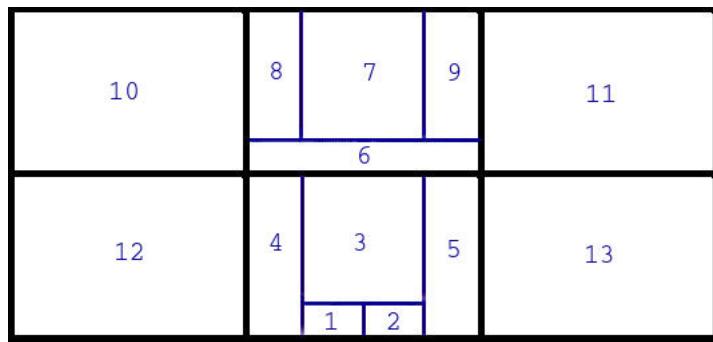


Figura 3.20: Estados definidos según la posición de la cámara

El estado número 14 se define cuando en un barrido completo de la camara no se encuentra la pelota

3.4.1.2. Acciones

Las posibles acciones a realizar dependiendo el estado en que se encuentre son

- Caminar un poco
- Caminar medio
- Caminar largo
- Girar poco derecha
- Girar derecha

- Girar poco izquierda
- Girar izquierda

3.4.1.3. Medida de desempeño

La recompensa positiva o negativa se otorga dependiendo de la medida de desempeño de un estado al siguiente estado. Se entiende como un buen desempeño que dada una distancia inicial a la pelota, con una acción se llega al siguiente estado en el cual la distancia a la pelota debe ser menor que la distancia inicial, de lo contrario sera un mal desempeño y por lo tanto una recompensa negativa.

El criterio de proximidad (medida de distancias relativas) se definió en función de las regiones generadas por los movimientos de la camara con valores en el intervalo de [1 - 10] donde 1 es lo mas cercano al robot (Zona de pateo) y 10 es lo mas lejano que puede ver en un barrido de la cámara. Se estableció el castigo como

$$C(s, s') = -ds'/10$$

dado un estado s a un estado s' hay penalización si la distancia de s' es mayor a la de s entonces la distancia obtenida en s' se divide entre 10 el cual es el máximo valor de distancia relativa a la pelota con ello los valores de la penalización se encuentran en el rango de [0 - 1]. Analogamente se define el premio como

$$P(s, s') = 1/ds'$$

obteniendo de la misma manera un rango de valores para las recompensas positivas de [0 - 1]

Una mejor visualización se puede apreciar en la figura 3.21

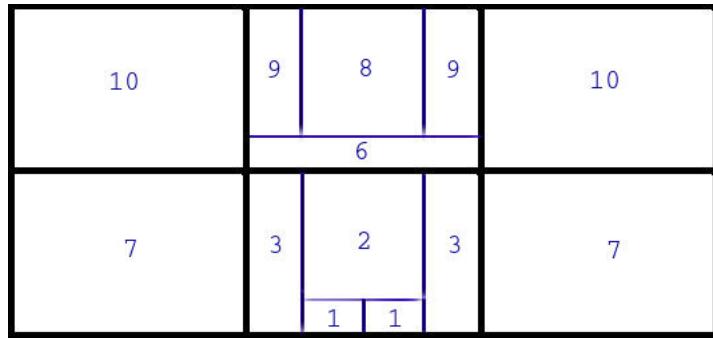


Figura 3.21: Distancias relativas de la pelota establecidas para otorgar la recompensa.

3.4.1.4. Elección de la acción

Se entiende que dado un estado se tienen 7 posibles acciones que en el entrenamiento del robot el aprende cual es mejor realizar según el estado en el cual se encuentra. Para que el aprendizaje obtenga mejores resultados se debe tener un equilibrio entre la exploración y la explotación.

Para variar entre la explotación y la exploración se utilizó la función de probabilidad definida como

$$P(a_i|s) = k^{Q(s,a_i)} / \sum_j k^{Q(s,a_j)}$$

Capítulo 4

Consideraciones Especiales:

Obstaculos y Soluciones

Quema de motores

No teníamos monitor, TightVNC se utilizó para la visualización y control de la interfaz gráfica de Raspbian en la Raspberry Pi desde un computador remoto, pues es conveniente poder observar lo que el robot percibe para llevar un control y una supervisión de su comportamiento.

servidor de ROS version mala

Conectar bien el FTDI

Al principio no cargaban los programas a la arbotix, comprar isp y cargar bootloader

Capítulo 5

Experimentos y Resultados

Se realizaron varios experimentos para comprobar la movilidad del robot. Primero se debe definir los movimientos que representan una unidad. Cada una de las siguientes acciones constituyen una unidad: caminar hacia adelante (Ade) , girar hacia la derecha (Der), girar hacia la izquierda (Izq), patada con la pierna derecha (PatearD), patada con la pierna izquierda (PatearD), levantarse en la posición supino, levantarse en posición prono.

El primer experimento consistió en verificar la ejecución de las unidades de acciones y determinar su funcionamiento, para ello se procedió a realizar la ejecución de cada unidad 50 veces, se tomó nota de las veces que lo realizaba de forma esperada (Positivo), y de aquellas que no lo hacía, si podría recuperarse de su mala acción (Corrigió) o no. Los resultados pueden observarse en la figura 4. En cuanto al balance del robot, solo una vez se cayó, sin embargo, logró levantarse satisfactoriamente.

En el segundo experimento se evaluó la ejecución de combinaciones de varias unidades de movimiento. Las combinaciones elegidas fueron:

- Caminar hacia adelante y patear con la pierna derecha (APD)
- Caminar hacia adelante y patear con la pierna izquierda (API)

- Caminar hacia adelante, girar hacia la derecha y patear con la pierna derecha (ADPD)
- Caminar hacia adelante, girar hacia la derecha, patear con la pierna izquierda (ADPI)
- Caminar hacia adelante, girar hacia la izquierda y patear con la derecha (AIPD)
- Caminar hacia adelante, girar hacia la izquierda y patear con la pierna izquierda (AIPI)
- Girar hacia la izquierda y patear con la pierna derecha (IPD)
- Girar hacia la derecha y patear con la izquierda (DPI)
- Girar hacia la derecha y patear con la pierna derecha (DPD)
- Girar hacia la izquierda y patear con la pierna izquierda (IPI)

Se realizaron 30 pruebas de cada una de estas combinaciones. Los resultados se han dividido en dos gráficos (figuras 5 y 6) para mejorar su visualización. En el peor caso (ADPD), Debupa logra patear la pelota 26 de las 30 veces al final del movimiento combinado. El robot sólo se ha caído dos veces, pero ha logrado levantarse con éxito (figura 5).

Experimentos con comportamientos integrados

Para medir el desempeño con respecto a la coordinación de los movimientos del robot y la detección de la pelota, se ha realizado un conjunto de pruebas. La primera prueba (prueba 1) consiste en la colocación de la pelota en la zona de pateo (zona justo al frente de los pies) y observar si el robot cumple con el objetivo de patearla. Para esta primera prueba se realizaron 10 corridas cuya duración aproximada fue de 30 segundos cada una. La segunda prueba (prueba 2) consistió en ubicar la pelota a 50 centímetros del robot en línea recta, la duración promedio de cada una de las 10 corridas ha sido de 2:10 minutos. Las siguientes dos pruebas, prueba 3 y prueba 4, consistieron en colocar la pelota 10 centímetros a la derecha y a la izquierda respectivamente de la posición de la prueba 2. Para ambas pruebas se realizaron 10 corridas con un promedio de duración de 5 minutos. En estas últimas tres pruebas se evaluó si el robot lograba llegar hasta la pelota y patearla.

Los resultados de las pruebas, en general, han sido satisfactorios dado que el robot logró llegar y patear la pelota en el porcentaje de los casos. Sin embargo el tiempo en el que cumple la tarea podría ser mejorado considerablemente si se evita reiniciar la posición de la cámara al estado inicial cada vez que el robot ejecuta una acción de movimiento.

En la prueba 2 el robot solo se ha caído dos veces, de las cuales en una ocasión se levantó y en la otra se quedó sin energía. En el resto de las pruebas no se ha caído.

Capítulo 6

Conclusiones y Recomendaciones

La presente investigación ha nacido de la motivación por hacer que en Venezuela se incursoe en proyectos que involucren humanoides autónomos e inteligentes. Se ha inspirado especialmente en la categoría Robocup soccer de la competencia internacional Robocup. Desde 1997, fecha en la que inició la competencia, Venezuela nunca ha participado en categorías con humanoides, mientras que países latinoamericanos como México, Brasil y Colombia sí han tenido avances en este campo. Si bien este proyecto no cumple con todas las reglas de la competencia se espera que éste pueda dar pie a continuar investigaciones dentro de Venezuela.

Los componentes utilizados en este proyecto son relativamente económicos comparados con otros en el mercado. La integración del kit Bioloid Premium con la Arbotix y la Raspberry Pi, ha hecho posible construir un humanoide inteligente sin tener que invertir exorbitantes cantidades de dinero. Una de las contribuciones más importantes es la coordinación y paralelismo exitoso entre todos los componentes utilizados.

Las mejoras que se pueden incorporar al proyecto podrían ser: la inclusión de aprendizaje por reforzamiento para patear de forma exitosa, incluir que la patada sea en dirección al arco e incluso añadir aprendizaje por reforzamiento para hacer que el robot pueda predecir la posición de una pelota en movimiento para que pueda patearla en el momento indicado.

Bibliografía

- [112a] RASPBERRY PI FOUNDATION UK REGISTERED CHARITY 1129409. Camera module setup.
- [112b] RASPBERRY PI FOUNDATION UK REGISTERED CHARITY 1129409. What is a raspberry pi?
- [201a] RoboCup 2014. Robocup 2014.
- [201b] RoboCup 2014. Robocup soccer.
- [AiR00] *Introduction to AI Robotics*. 2000.
- [Ard14] Arduino. Arduino, 2014.
- [Boo08] *Learning OpenCV*. 2008.
- [BSS⁺] Sven Behnke, Michael Schreiber, Jörg Stückler, Reimund Renner, and Hauke Strasdat. See, walk, and kick:humanoid robots start to play soccer.
- [Co11] Honda Motor Co. Honda unveils all-new asimo with significant advancements, 2011.
- [Con14] Wolfram Demonstrations Project Contributors. Gyrocospe, 2014.
- [Ele] SparkFun Electronics. Ftdi sparkfun.

- [Fer10] M Fergs. Pypose, 2010.
- [Hob] HobbyKing. Turnigy tg9e 9g.
- [INCa] Robotics INC. Gyro sensor.
- [INCb] Robotis INC. Bioloid.
- [INCc] Robtis INC. Robotis.
- [INC06] Robotics INC. *Dynamixel AX-12*, 2006.
- [LLC] Trossen Robotics LLC. arbotix robocontrollers.
- [Mit97] Tom M. Mitchell. Machine learning. 1997.
- [pet95] *Artificial Intelligence A Modern Approach*. 1995.
- [pet09] *Artificial Intelligence A Modern Approach*. 2009.
- [Pre14] Oxford University Press. Robotics, 2014.
- [Rob] Robotis. Lipo 11.1v battery set lbs-10.
- [Ros] Ros. About ros.
- [SSA⁺] Mostafa E. Salehi1, Reza Safdari, Erfan Abedi, Bahareh Foroughi, Amir Salimi, Emad Farokhi, Meisam Teimouri, and Roham Shakiba. Mr1 team description paper for humanoid kidsize league of robocup 2014.
- [Tea] OpenCV Developers Team. About.
- [Val] Emil Valkov. Raspberry pi camera with opencv.
- [WL] Williams and Wang LLC. Dynamixel ax/mx 6 port extension hub.

[YML] Seung-Joon Yi, Stephen McGill, and Daniel D. Lee. Improved online kick generation method for humanoid soccer robots.

Apéndice A

Archivos intermedios

APENDICE