UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Aprendizaje motor en robots humanoides a partir de la imitación humana

Andrés Omar Aguirre Dorelo

Programa de Posgrado en Informática
PEDECIBA Informática, Instituto de Computación, Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Junio de 2018

UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Aprendizaje motor en robots humanoides a partir de la imitación humana

Andrés Omar Aguirre Dorelo

Tesis de Maestría presentada al Programa de Posgrado en Informática, PEDECIBA Informática de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Magister en Informática.

Directores:
Dr. Prof. Javier Ernesto Baliosian De Lazzari
Dr. Prof. Gonzalo Daniel Tejera López

Director académico:
Dr. Prof. Javier Ernesto Baliosian De Lazzari

Montevideo – Uruguay
Junio de 2018

Aguirre Dorelo, Andrés Omar

Aprendizaje motor en robots humanoides a partir de la imitación humana / Andrés Omar Aguirre Dorelo. - Montevideo: Universidad de la República, PEDECIBA Informática, Instituto de Computación, Facultad de Ingeniería, 2018.

XXVIII, 96 p.: il.; 29, 7cm.

Directores:

Javier Ernesto Baliosian De Lazzari

Gonzalo Daniel Tejera López

Director académico:

Javier Ernesto Baliosian De Lazzari

Tesis de Maestría – Universidad de la República, Programa en Informática, 2018.

Referencias bibliográficas: p. 77 – 83.

1. Aprendizaje por Imitación, 2. Control Motor, 3. Robótica Evolutiva, 4. Robots Humanoides, 5. Locomoción Bípeda. I. Baliosian De Lazzari, Javier Ernesto, Tejera López, Gonzalo Daniel, . II. Universidad de la República, Programa de Posgrado en Informática. III. Título.

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE TESIS

_____

Dr. Prof. Juan Miguel Santos (revisor)

_____

Dr. Prof. Martín Nicolás Pedemonte Quintas

_____

Dr. Prof. Martín Llofriu Alonso

Montevideo – Uruguay
Junio de 2018

Dedico esta tesis a mi esposa
Diana y mis padres Ana y Omar
por el apoyo, el aguante y el
amor.

Agradecimentos

Quisiera agredecer a: Diana Magano, Ana Dorelo, Omar Aguirre, Graciela Sarasúa, Björn Mattsson, Gustavo Noble, Guillermo Reisch, Marcelo Baliero, Octavio Perez, Sylvia da Rosa, Raquel Dorelo, Victoria Aguirre, Carlos Magano, María del Huerto Sarasúa, Luis Michelena, Federico Andrade, Martín Giachino, Jorge Visca, el Programa de Desarrollo de las Ciencias Básicas, La Comisión Académica de Posgrado de la Universidad de la República y a Pyxis, por apoyarme de una manera u otra para la concreción del presente trabajo.

A Martín Pedemonte por sus enseñanzas acerca de algoritmos genéticos.

A Marcos Sander, Facundo Benavides, Patricia Polero y Daniel Larrosa por el aporte de ideas y discusiones que colaboraron en la concreción de este trabajo.

A Andrés Vasilev por el apoyo en la ejecución de los experimentos en la plataforma física y aporte de ideas.

A Javier Baliosian y Gonzalo Tejera por su guía y apoyo durante el proyecto.

Los datos utilizados en este proyecto fueron obtenidos de mocap.cs.cmu.edu.

No aprendes a caminar siguiendo las reglas. Aprendes haciendo y cayendo

Richard Branson

RESUMEN

El control motor en robots humanoides es una tarea compleja debido al alto número de grados de libertad que deben ser tratados; la mayoría de las soluciones presentadas para el área de control de movimiento dependen, en gran medida, de la información específica del dominio sobre el robot o la tarea motora concreta a desarrollar, lo cual, dificulta la posibilidad de generalizar la solución.

El aprendizaje por demostración surge como una alternativa más sencilla para la programación de habilidades motoras en robots, pero hasta ahora la mayoría de las arquitecturas propuestas sólo han sido validadas con tareas motoras que no comprometen la estabilidad del robot.

Este trabajo presenta el desarrollo de una arquitectura de aprendizaje por imitación que no requiere de un modelo analítico del robot ni del comportamiento motor a ser aprendido. Su validación se realiza utilizando la marcha como el comportamiento motor a ser aprendido por un robot bípedo. La solución propuesta toma como entrada, información de captura de movimiento a partir de la ejecución de la tarea por un grupo de maestros humanos; la salida es calculada mediante un algoritmo genético para el que se definen operadores de selección, mutación y cruzamiento, acordes al problema de aprendizaje por imitación; estos operadores aplican sobre una codificación de cromosoma implementada mediante una serie temporal donde el conjunto de las posiciones angulares de cada pose corresponden a cada gen del cromosoma representado por dicha serie. A partir de esto, el robot puede reproducir la tarea asignada utilizando su propio repertorio motor.

Palabras claves:
Aprendizaje por Imitación, Control Motor, Robótica Evolutiva, Robots Humanoides, Locomoción Bípeda.

ABSTRACT

Motor control in humanoid robots is a challenging task due to the high number of degrees of freedom that must be dealt with; most solutions presented for the motion control area rely, heavily, on domain specific information about the robot or the concrete motor task to be developed, which makes it difficult or not viable to generalize the solution.

Learning by demonstration arises as an easier alternative for programming motor skills in robots, but, until now most of the proposed architectures are only validated using motor tasks which do not compromise the robot stability.

This work presents the development of a learning by imitation architecture that does not require an analytical model of the robot or the motor behavior to be learned. Its validation is done using the gait of a biped robot as the motor behavior to be learned.

Our proposed solution takes the motion captured information from a group of human masters executing the task as its input, and the output is computed by a genetic algorithm for which selection, mutation and crossing operators are defined, according to the problem learning by imitation; these operators apply on a chromosome coding implemented by a time series where the set of the angular positions of each pose correspond to each gene of the chromosome represented by the named serie. Beginning with this, the robot can reproduce the assigned task using its own motor repertoire.

Keywords:
Imitation Learning, Motor Control, Evolutive Robotics, Humanoid Robots, Biped Locomotion.

Tabla de contenidos

Lista de figuras

Lista de tablas

Lista de siglas

AG Algoritmo Genético
API Application Programming Interface
CMU Universidad de Carnegie Mellon, por sus siglas en inglés
COG Centro de gravedad, por sus siglas en inglés
COM Centro de masa, por sus siglas en inglés
CPG Centro generador de patrones, por sus siglas en inglés
DTW Distorsión dinámica temporal, por sus siglas en inglés
LFD Aprendizaje por demostración, por sus siglas en inglés
MOCAP Motion Caputure
PBD Programación por Demostración, por sus siglas en inglés
PC Personal Computer
PCA Principal Components Analysis
PGM Medida de marcha pasiva, por sus siglas en inglés
USB Universal Serial Bus
XML Extensible Markup Language
ZMP Punto de momento cero, por sus siglas en inglés

Capítulo 1

Introducción

La metodología tradicionalmente empleada en la implementación de comportamientos motores, en robots, se basa en el desarrollo de controladores especializados para realizar un conjunto predeterminado de tareas en entornos altamente limitados y deterministas [15]. Típicamente la programación de estos controladores requiere de la incorporación en su lógica de un amplio conocimiento de la arquitectura del robot y su entorno mediante una especificación analítica del modelo realizada manualmente por un humano utilizando sus conocimientos de física [10]. Otras implementaciones toman un enfoque más simple respecto a la información modelada por el controlador, pero requiere de un mayor esfuerzo para la implementación de movimientos; estas implementaciones utilizan técnicas similares a la de STOP-MOTION, utilizada en animación, para la implementación del control motor, un ejemplo de ese enfoque puede encontarse en los trabajos [39, 46].

No transcurrió mucho tiempo para que quedase claro, ya en 1994 Kuniyoshi e Inaba lo advertían [35], que mediante el uso de estos enfoques no se podría lograr controlar robots que requiriesen trabajar en entornos muy variables, donde el aprendizaje de nuevos movimientos o la adaptación de los ya existentes fuese requerido, como ocurre en el caso de los robots humanoides utilizados para interactuar con los seres humanos en sus ambientes cotidianos [15], lo que se conoce como robótica de servicio.

1.1. ¿Por qué imitar?

La imitación es un enfoque intuitivo para la resolución del problema de la programación de comportamientos motores. La idea de que un robot aprenda a generar sus propios comandos motores simplemente observando a un instructor hacerlo, resulta mucho más tentador que pasar horas programando un pequeño movimiento de apenas unos segundos, como ocurre al utilizar técnicas del tipo de stop-motion; también el aprendizaje por imitación, al igual que otras técnicas de aprendizaje, resulta una opción más tentadora respecto a modelar, de forma manual, una función analítica que represente un tipo específico de movimiento basada en la percepción humana de la física [44]; los robots autónomos del futuro necesitan poder generar su modelo desde información extraída de las secuencias de datos accesibles al robot [45].

A medida que los robots comienzan a ser utilizados en ambientes humanos, la capacidad de aprender e imitar comportamientos realizados por éstos basado únicamente en la observación será de suma importancia [34].

La imitación es un mecanismo fundamental de aprendizaje que podemos encontrar en varios sistemas biológicos, especialmente entre nosotros, los seres humanos [34]. Varios estudios han revelado que los niños pasan una cantidad sustancial de tiempo imitando comportamientos previamente observados [50]. Por otra parte, estudios similares realizados en animales también sugieren que utilizan un proceso de aprendizaje imitativo [52]. Sorprendentemente, esta capacidad parece estar presente ya al nacer. De hecho, se ha demostrado que humanos y monos recien nacidos, pueden imitar los gestos faciales y manuales ejecutados por personas mayores incluso antes de haber visto su propia cara o la de muchos otros adultos [21]. Esta capacidad ha sido explicada desde el área de la neurociencia gracias al descubrimiento hecho por Giaccamo Rizzollati, inicialmente en los monos y luego verificado también en los humanos, de un tipo de neurona que se activa tanto cuando realizamos una acción como cuando vemos a otro realizarla, estas neuronas llamadas *espejo* [27] son la base del mecanismo de imitación.

Algunas de estas neuronas parecen codificar las particularidades de los movimientos en relación con los objetos en el entorno del animal [43]; comportándose como un sistema de espejo, es decir, uno que utiliza los mismos códigos neuronales para caracterizar una acción, ya sea ejecutada u observada por el agente [5].

La mayoría de las habilidades cognitivas humanas de nivel superior, pueden estar arraigadas evolutivamente a nuestra capacidad de observarnos e imitarnos unos a otros; tal como propone Ramachandran en [49], las neuronas espejo permiten a un individuo imitar los movimientos de otro, preparando así el escenario para la compleja herencia lamarckiana o cultural que caracteriza a nuestra especie y nos libera de las limitaciones de una evolución puramente basada en genes. Un ejemplo de como este sistema de espejo enriquecido por imitación permitió a las sociedades humanas, a lo largo de muchos milenios de invención y evolución cultural, alcanzar nuestras habilidades actuales, puede encontrarse en la construcción del lenguaje [5].

Es natural pensar que esta capacidad de aprendizaje innata al ser humano, y algunas especies animales, pueda ser modelada en un robot para permitir aprender nuevos comportamientos, en lugar del enfoque clásico de realizar una especificación exhaustiva en el controlador.

Desde la década de los ochenta, el aprendizaje mediante técnicas de imitación se ha utilizado para programar robots (también conocido como programación por demostración - PBD por sus siglas en inglés). En esa década, PBD surge en el campo de robots industriales como una forma de automatizar la programación. La principal razón para tomar este camino fue un desarrollo más sencillo y una importante reducción de los tiempos asociados al mantenimiento. Desde entonces, y debido a su éxito, muchas propuestas de aprendizaje por imitación han sido desarrolladas y aplicadas a la robótica [55].

Utilizaremos la palabra imitar para dar a entender que el observador no está simplemente replicando las acciones del instructor, sino que está intentando alcanzar el objetivo de la acción perseguida por el instructor realizando una nueva acción similar a la observada [24].

1.2. Motivación en lucir como humanos

El interés de la comunidad científica en el aprendizaje por imitación, a partir de la imitación de instructores humanos, ha crecido en los últimos años en gran parte debido al auge en el desarrollo de la robótica humanoide; una razón sugerida por Brooks y elaborada recientemente por Pfeifer y Bongard para el desarrollo de la robótica humanoide, es que la morfología de los cuerpos humanos puede definir la forma en que pensamos y utilizamos nuestro intelecto, por lo que si deseamos construir robots con inteligencia humana, entonces la forma

del robot también debe ser humana [25]. Un ejemplo de esta sinergia entre el desarrollo físico y cognitivo puede verse en el progreso del aparato respiratorio de los seres humanos, el cual fue evolucionando para permitir modular la voz acompasado por el desarrollo del cerebro para permitir la evolución del lenguaje; Según Arbib [5] el cerebro y el cuerpo humano evolucionaron de tal manera que tenemos la movilidad de la mano, la laringe y la cara adecuada para generar gestos que se puede usar en el lenguaje y los mecanismos cerebrales necesarios para producir y percibir rápidamente secuencias generadas de dichos gestos.

Otros autores, como Bongard [11] , Fitzpatrick [26] y Wehner [61] entre otros, son motivados por razones fundadas desde un punto de vista pragmático, sostienen que la razón para la creación de robots humanoides es que serán capaces de operar en los mismos entornos que los humanos operan. Esto les permitirá desempeñarse mejor en toda una gama de situaciones en las que un robot no humanoide sería bastante impotente.

Las técnicas de PBD son generalmente identificadas como una oportunidad para manejar la complejidad asociada al control motor de este tipo de robots, pero no es la única área donde el uso de estas técnicas pueden generar beneficio; la interacción entre humano y robot se simplifica y mejora al utilizar los canales de comunicación ya existentes entre los humanos, donde aspectos como gestos y expresiones son de suma importancia; de esta manera las personas pueden encontrar más fácil tratar con un robot humanoide que con una estructura puramente mecánica. De forma similar, las personas ya poseen la habilidad para realizar varias tareas deseables para los robots humanoides, estas habilidades serán más fáciles de transferir, utilizando aprendizaje por imitación, a un robot humanoide que a un robot con un cuerpo drásticamente diferente [10].

1.3. Motivación en la utilización de la técnica de imitación

No solo es necesario que los robots luzcan como humanos para poder cumplir con las motivaciones presentadas en la sección anterior, comportarse como tales permite desde un punto de vista ingenieril una fuente de obtención de soluciones factibles para resolver el problema del control, en una situación que

presenta muchos grados de libertad, como es el caso del control motor de un humanoide.

Por otro lado como fue mencionado en la sección 1.1, el aprendizaje por imitación presenta grandes oportunidades de aplicación para la implementación de tareas motoras en robots humanoides, debido a su simplicidad al derivar la política de control a partir de un conjunto de ejemplos brindados por un humano. A su vez esto permite ser un mecanismo extensible de programación, para una cantidad no acotada de tareas, incluso para usuarios sin conocimientos en programación, como propone el campo de la robótica orientada al servicio.

Desde un punto de vista asociado a las ciencias de la computación, existen algunas características que se destacan mientras se utiliza PBD como técnica de aprendizaje: en primer lugar, la forma en que funciona el proceso reduce el espacio de búsqueda de la solución; el aprendiz no se preocupa más por todas las posibles soluciones, sino que sólo trata con las que están más cerca de los comportamientos previamente observados (demostración) y encajan bien con sus capacidades. Además, en comparación con otros enfoques de aprendizaje automático, PBD puede en algunos casos mejorar el rendimiento general de aprendizaje [55].

En [12] se hace referencia a una serie de señales sociales que los seres humanos emplean naturalmente para estructurar y facilitar el proceso de aprendizaje para los pares. Por lo que, desde un punto de vista psicológico, dotar a un robot con la capacidad de aprender a partir de la imitación permite a los seres humanos generar el espacio para una interacción natural.

Las técnicas de PBD son mecanismos de transferencia humano-robot de comportamientos, donde la interacción necesaria para implementar la transferencia se da mediante elementos cercanos para los seres humanos, como es la interacción con un par; como asegura Breazeal en [12], los humanos ya son expertos en interacción social y es una interfaz universalmente entendida.

Finalmente, al dotar de capacidad de aprendizaje, el sistema robótico se vuelve más autónomo [55].

1.4. Trabajo relacionado

El aprendizaje por demostración es un área de larga data de investigación [34] que ha sido explorada por un número considerable de investigadores en

robótica. Sin embargo la mayoría de los enfoques de aprendizaje de habilidades motoras presentados en el área han sido validados usando habilidades motoras que no comprometen la estabilidad del robot, algunos ejemplos de esto son: aprender el movimiento articulado del brazo humano y agarrare de objetos / movimiento para lanzar objetos [34], un golpe de tenis y secuencias de tambores [56], gestos comunicativos y habilidades para la manipulación de objetos [15], aprendizaje de movimientos del torso de un humanoide [41].

Otros autores utilizan un acercamiento diferente para la adquisición del movimiento a ser aprendido, en lugar de observar la demostración del instructor humano ejecutando la tarea, el instructor muestra el movimiento a ser aprendido moviendo directamente los actuadores del robot adquiriendo las poses posturales usando directamente los sensores de posición, generalmente disponible en los servo motores, del robot. Esta técnica es conocida como Kinesthetic Bootstraping [7] y presenta la ventaja de no requerir de la resolución del problema de la correspondencia [3], consiste en identificar los aspectos necesarios que permiten transferir determinado movimiento, realizado por un instructor a un aprendiz, resolviendo la adaptación necesaria del movimiento a las capacidades del aprendiz; esta técnica presenta un problema de escalabilidad dado que los movimientos ejecutados por el instructor están asociados a un tipo específico de robot.

Debido a que la arquitectura de aprendizaje por imitación propuesta en este trabajo es validada utilizando el comportamiento motor de la marcha bípeda, también fueron relevados trabajos que tienen por objetivo aplicar la técnica de aprendizaje por imitación a dicho movimiento concreto.

Las técnicas tradicionales para resolver el problema de la marcha bípeda necesitan de un modelo analítico con especificación precisa de la dinámica del robot y su entorno para lograr una marcha estable, como veremos en la sección 2.3.1. El uso de PBD para el problema de la marcha es una solución prometedora, ya que es independiente del robot específico utilizado y podría producir más resultados similares a los humanos que los otros enfoques, cuando se utiliza un instructor humano.

Son pocos los investigadores que han publicado trabajos donde la marcha bípeda sea implementada mediante un enfoque de aprendizaje por imitación. Uno de los ejemplos más destacados es el trabajo de Chalodhorn [18] que presenta un enfoque en el cual no se requiere de un modelo analítico para el aprendizaje de las habilidades motoras mediante la imitación; el mismo ha sido

utilizado para el aprendizaje de habilidades motoras que presentan una alta componente dinámica, como es el caso de la caminata bípeda. El trabajo antes mencionado, de Chalodhorn, logra resultados similares a los presentados en el presente trabajo, pero difiere de esta propuesta en la técnica de inteligencia artificial empleada: la imitación fue modelada por medio de un proceso markoviano junto con un componente de circuito cerrado que emplea la información de un sensor giroscópico en el robot, para corregir las poses aprendidas por demostración. Se ha publicado un trabajo más reciente de Chaladhorn [19] que explora un enfoque más genérico para el aprendizaje de tareas mediante imitación, permitiendo al igual que en el presente trabajo aprender diferentes tipos de tareas motoras.

El gran número de grados de libertad involucrados en el problema de la imitación motora ha empujado a los investigadores en el área a utilizar técnicas de reducción dimensional para tratar con la complejidad asociada. Los métodos de reducción dimensional como el análisis de componentes principales (PCA, por sus siglas en inglés) son comúnmente utilizados para sacar provecho que los datos recolectados pueden presentar redundancia. Un subespacio de menores dimensiones puede calcularse asociado a una tarea concreta para proyectar en él los datos de movimiento humanos originales. PCA retiene una gran proporción de la variación presente en los datos originales, pero condiciona la exploración del espacio de soluciones, lo que motiva nuestra búsqueda de alternativas en este trabajo.

1.5. Objetivo de la tesis

El campo del aprendizaje por imitación ha identificado un conjunto de preguntas de investigación necesarias que aseguren un enfoque genérico para la transferencia de habilidades entre agentes y situaciones: ¿Qué imitar? ¿Cómo imitar? ¿Cuándo imitar? y ¿A quién imitar? [9], de las cuales ahondaremos en el Capítulo 2 presentado un conjunto más refinado.

El objetivo general de este trabajo se centra en la búsqueda de respuestas a las dos primeras preguntas. La pregunta de ¿Qué imitar? obtiene respuesta mediante la búsqueda de los aspectos relevantes en la demostración que permitan realizar una determinada tarea. Por otro lado la respuesta a la pregunta de ¿Cómo imitar? también referida como el problema de la correspondencia [16].

Como objetivos específicos de este proyecto se proponen: (i) la propuesta de un algoritmo que permita el aprendizaje motor mediante imitación humana teniendo en cuenta los resultados obtenidos al trabajar sobre los objetivos generales planteados. (ii) la transferencia del comportamiento de la marcha a un robot humanoide a partir de la imitación de instructores humanos; de esta forma el algoritmo propuesto es validado, utilizando uno de los movimientos más complejos e imprescindibles mencionados en el estado del arte para robots humanoides [11] [23] [26] debido a los múltiples grados de libertad a ser controlados en el robot de forma coordinada para mantener el balance durante la caminata.

1.6. Contribución

La contribución principal de este trabajo, consiste en la propuesta de un algoritmo capaz de resolver el problema del aprendizaje motor en robots humanoides a partir de información motora capturada en la ejecución, por instructores humanos, de la tarea a ser aprendida. El algoritmo puede ser aplicado a cualquier tipo de tarea motora, incluso aquellas que requieren resolver adicionalmente a la tarea motora objetivo la tarea del equilibrio, como es el caso de la marcha bípeda. Esta propuesta realiza una novel implementación de la técnica de aprendizaje por imitación mediante algoritmos genéticos; a través de la definición de operadores de: mutación, cruzamiento, selección; y la codificación genética elegida, las preguntas, formuladas en el área de aprendizaje por imitación, respecto a ¿Qué imitar? y ¿Cómo imitar? obtienen una respuesta poco explorada hasta el momento; obtener respuestas a estas preguntas es el paso inicial necesario para poder explorar soluciones al problema del "Mapeo Instructor Aprendiz" presentado en el Sección 2.1.2 y necesario para trabajar la transferencia de tareas motoras a un robot, siendo esta la razón para la selección de las preguntas a abordar en el alcance del trabajo, dejando el resto para su evaluación en los trabajos futuros.

Dentro de las contribuciones también se encuentra el estado del arte desarrollado, el cual puede ser consultado en el Capítulo 2.

Otro aspecto a destacar, con el que contribuye la propuesta, es la implementación de un enfoque multi-maestro, diseñado con el objetivo de que el algoritmo identifique los aspectos clave relativos a la ejecución de la tarea a ser aprendida, aprovechando la diversidad representada por diferentes inter-

pretaciones grabadas.

Se ha desarrollado un FRAMEWORK de aprendizaje motor para simplificar la aplicación del algoritmo propuesto, el mismo ha sido desarrollado con el fin de permitir simplificar el proceso de generalización de movimientos, evaluación y reproducción de tareas motoras, independientemente de la tarea motora concreta a ser aprendida. El framework ha sido diseñado de forma de simplificar su uso en diferentes plataformas robóticas sin necesidad de grandes cambios; el código fuente está disponible en línea [1], liberado con licencia GNU GPL V2, de esta forma los resultados presentados en este trabajo pueden ser reproducidos por otros investigadores donde nuevas aplicaciones pueden ser desarrolladas sobre el mismo. En este trabajo la genericidad y potencial del framework son validadas al ser éxitosamente aplicado en el aprendizaje de la marcha bípeda por un robot bípedo del tipo Bioloid Premiun.

1.7. Estructura del documento

La presente tesis se organiza de la siguiente manera: en el Capítulo 2 presentamos brevemente una introducción a los conceptos del estado del arte que fueron relevados para el desarrollo de la misma, en el Capítulo 3 se describe el mecanismo propuesto para determinar las ejecuciónes a utilizar y su transferencia como un comportamiento motor.

En el Capítulo 4 se presenta un análisis de los resultados obtenidos al aplicar la transferencia del comportamiento de la marcha a un robot humanoide utilizando únicamente información registrada en la ejecución de ese movimiento por un conjunto de instructores humanos.

Finalmente en el Capítulo 5 se presentan las conclusiones y el trabajo a futuro.

Capítulo 2

Estado del arte

En el presente capítulo se releva el estado del arte de las principales áreas utilizadas como pilar para el desarrollo de este trabajo. En la primer sección se estudian los conceptos fundamentales relativos al área de aprendizaje por imitación; luego se presenta una introducción al área de la Robótica Evolutiva y sus principales desafíos; seguidamente se realiza un análisis sobre las caminatas bípedas desde el punto de vista de su modelado y los criterios y métricas necesarios para evaluarlas.

Al final del Capítulo se presentan, a modo de conclusión, los aspectos relevantes para el diseño de la solución de la cual es objeto el presente trabajo.

2.1. Aprendizaje por Imitación

Se han utilizado diversas terminologías para referirse a este conjunto de trabajos. Estos incluyen la, ya mencionada, programación por demostración (PBD), el aprendizaje por demostración humana (LFD, por sus siglas en inglés) y el aprendizaje por imitación. Todas ellas se refieren a un paradigma general para permitir a los robots realizar autónomamente nuevas tareas desde la observación y el aprendizaje [10], particularmente en lo referente a robótica humanoide, desde la observación de los seres humanos realizando estas tareas.

2.1.1. Desafíos inherentes al aprendizaje por imitación

El área de aprendizaje por imitación también presenta su propio conjunto único de preguntas de investigación, dependiendo del autor pueden existir algunas diferencias; a continuación se presenta un conjunto propuesto por [13]

que incluye todas las preguntas relevadas en la bibliografía. Cada una de estas preguntas es un problema de investigación complejo que la comunidad robótica apenas ha empezado a abordar; las mismas se pueden sitentizar en:

- ¿Cómo sabe el robot cuándo imitar?
- ¿Cómo sabe el robot qué imitar?
- ¿Cómo el robot asigna las acciones observadas en comportamientos de respuesta?
- ¿Cómo el robot evalúa sus acciones, corrige errores y reconoce cuando ha logrado su objetivo?

¿Cómo sabe el robot cuándo imitar?

El robot debe decidir si es apropiado o no participar en un comportamiento imitativo basado en el contexto social actual, la disponibilidad de un buen instructor y las metas y motivaciones internas del robot. Por ejemplo, el robot puede necesitar elegir entre asistir a una oportunidad de aprendizaje o cumplir otra meta, como recargar sus baterías. Esta decisión se basará en el entorno social, la probabilidad de que el robot tenga otra ocasión de participar en esa oportunidad de aprendizaje en particular, el nivel actual de necesidad para cargar las baterías, la calidad de la instrucción y otras motivaciones y objetivos que compiten entre sí. Además, el robot también debe reconocer cuándo la imitación es una solución viable y actuar para generar el contexto social en el que puede aprender por observación, tal vez buscando un instructor o motivando al instructor para realizar una determinada tarea [13].

¿Cómo sabe el robot qué imitar?

Frente a un flujo entrante de datos sensoriales, el robot debe tomar una serie de decisiones para determinar qué acciones en el mundo son apropiadas para imitar. El robot primero debe determinar qué agentes en la escena son buenos instructores, siendo capaz de evitar malos instructores. Es decir el robot no solo debe ser capaz de distinguir la clase de estímulos (incluidos los humanos y quizás otros robots) que podría ser de interés imitar, sino también determinar si las acciones actuales de ese agente son dignas de imitación. No todos los humanos en toda ocasión serán buenos instructores, y la imitación solo puede ser apropiada bajo ciertas circunstancias [13].

Una vez que se ha seleccionado un instructor, ¿cómo determina el robot cuáles de las acciones del instructor son relevantes para la tarea?, ¿cuáles son parte del proceso social/de instrucción?, y ¿cuáles son circunstanciales? [13].

¿Cómo el robot asigna las acciones observadas en comportamientos de respuesta?

Esta pregunta es equivalente a responder ¿cómo imitar? y es de lo que trata el problema de la correspondencia, ver sección 2.1.2.

Una vez que el robot ha identificado los aspectos más destacados de la escena, ¿cómo determina qué acciones debería tomar? Cuando el robot observa a un instructor abriendo un frasco, ¿cómo convierte el robot esa percepción en una secuencia de acciones motoras que llevará al brazo para lograr el mismo resultado? El mapeo de un cuerpo a otro implica no solo determinar qué partes del cuerpo tienen una estructura similar sino también transformar los movimientos observados en aquéllos que el robot es capaz de realizar. Por ejemplo, si el instructor está desenroscando la tapa del frasco, el robot primero debe identificar que el movimiento del brazo y la mano son relevantes para la tarea y determinar que su propia mano y brazo son capaces de realizar esta acción. El robot debe observar los movimientos de la mano y el brazo del instructor y asignar esos movimientos a las coordenadas del motor de su propio cuerpo [13].

Un ejemplo de un mecanismo natural, candidato a nivel neurológico para resolver el problema de correspondencia, es el sistema de espejo [43], presentado en el Capítulo 1.

¿Cómo el robot evalúa sus acciones, corrige errores y reconoce cuando ha logrado su objetivo?

Una vez que un robot puede observar una acción e intentar imitarla, ¿cómo puede determinar si ha tenido éxito o no? Para comparar sus acciones con las del instructor, el robot debe ser capaz de identificar el resultado deseado y juzgar cuán similares fueron sus propias acciones a ese resultado. Si el robot está intentando desenroscar la tapa de un frasco, ¿ha tenido éxito si simplemente imita el instructor y gira la tapa pero deja la tapa sobre el frasco? ¿Es exitoso el robot si quita la tapa tirando en vez de girar? ¿Es exitoso el robot si rompe la jarra para abrirla? En ausencia de motivaciones internas que propor-

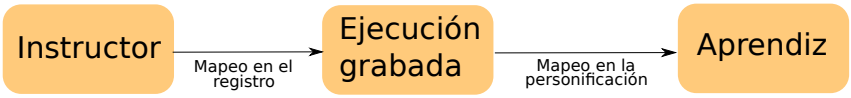

Figura 2.1: El mapeo entre el instructor y el aprendiz es implementado mediante dos funciones; *Record Mapping* correspondiente al mapeo en el registro y *Embodiment Mapping* que corresponde con el mapeo en la personificación. Figura extraída y modificada de [6].

cionen retroalimentación sobre el éxito de la acción, la evaluación dependerá de la comprensión de los objetivos e intenciones del instructor. Además, si el robot no ha tenido éxito, ¿cómo determina qué partes de su funcionamiento fueron inadecuadas?. El robot debe ser capaz de diagnosticar sus propios errores para mejorar su rendimiento [13].

A continuación se presentan técnicas que permiten realizar el mapeo entre el instructor y el aprendiz, necesario para poder obtener una representación de trabajo para la ejecución de una tarea por un instructor, como también una solución al problema de la correspondencia, planteado por la pregunta de investigación de ¿Cómo el robot asigna a las acciones observadas comportamientos de respuesta?.

2.1.2. Mapeo Instructor Aprendiz

Argall y colaboradores [6] discuten diferentes técnicas para ejecutar y registrar demostraciones, determinando que el conjunto de datos necesario para aplicar PBD está compuesto por pares de estado-acción registrados durante las ejecuciones del movimiento a aprender realizadas por el instructor. Estos estados y acciones deben ser utilizables por el aprendiz, existiendo diversos acercamientos para asegurarlo. Además, los autores definen la correspondencia entre el instructor y el aprendiz (que permite la transferencia de información) como dos funciones de mapeo: la función de *Record Mapping* $g_R(z, a)$ y la función de *Embodiment Mapping* $g_E(z, a)$ (ver Figura 2.1).

El primer mapeo (g_R) define cómo los estados z y las acciones a (alcanzados y ejecutadas, respectivamente por el instructor, durante la demostración) son registrados dentro de un conjunto de datos. Por otra parte, la función g_E refiere a cómo los pares de estado-acción registrados se asignan finalmente al aprendiz. De acuerdo con la ausencia o la presencia de g_R y g_E en el mapeo instructor-aprendiz, la correspondencia entre el instructor y el aprendiz podría clasificarse de la siguiente manera: teleoperación, sombreado, sensores en el

instructor y observación externa (véase la figura 2.2).

Los autores consideran dos categorías dependiendo de la función de mapeo g_E; cuando ésta corresponde con la función identidad, $g_E(z, a) \equiv I(z, a)$, la demostración se realiza en el mismo robot o en una plataforma físicamente idéntica, lo llaman *Caso de demostración*. Por el contrario, cuando $g_E(z, a) \neq I(z, a)$, hay diferencias entre el cuerpo del instructor y el del aprendiz, también conocido como problema de la correspondencia, ver Sección 2.1.2 para más información, lo llaman como *Caso de imitación*, ver Figura 2.2.

A continuación se enumeran las clases de correspondencia resultantes según la combinación de los posibles casos de las funciones de mapeo g_E y g_R:

a) Teleoperación: En esta técnica, el instructor mueve al robot aprendiz mientras se registra la demostración. Este enfoque es necesariamente apoyado por los propios sensores del robot. Tanto la función de mapeo g_E como la g_R corresponden con la de identidad, $g_R(z, a) \equiv I(z, a)$ y $g_E(z, a) \equiv I(z, a)$. Las ejecuciones de las demostraciones se pueden hacer de varias maneras. Por ejemplo, a través de la teleoperación humana mediante un joystick mientras el robot registra sus posiciones motoras o mueve sus miembros a través de la secuencia de posiciones deseadas. Sin embargo, aunque la teleoperación es el método más directo de transferencia de información, no siempre es aplicable, debido a sus fuertes requerimientos sobre las capacidades sensoriales del robot, la pérdida de autonomía que este método implica y la dificultad de controlar robots con muchos grados de libertad realizando tareas complejas, que requieren un movimiento sincronizado de cada actuador respecto a los demás, como es el caso de la marcha bípeda.

b) Sombreado: El robot aprendiz registra la ejecución usando sus propios sensores mientras imita los movimientos del instructor. Incluso aunque el mapeo de la función g_E es directo, $g_E(z, a) \equiv I(z, a)$, el mapeo definido por la función $g_R(z, a)$ no, $g_R(z, a) \neq I(z, a)$. Se requiere un componente algorítmico adicional para permitir que el robot se comporte como una sombra del instructor.

c) Sensores en el instructor: Esta técnica implica que los sensores sean colocados directamente sobre el cuerpo del instructor con el fin de registrar con precisión la demostración. El mapeo de la función g_R es directo: $gR(z, a) \equiv I(z, a)$. La precisión es la mayor ventaja de este enfoque. Sin embargo, la desventaja radica en la sobrecarga asociada a la necesidad de sensores especializados. Los trajes de sensor portátiles o los alrededores personalizados, como las habita-

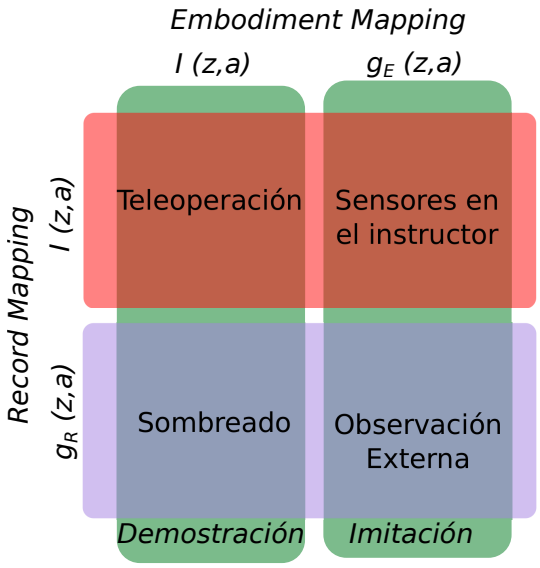

Figura 2.2: Clases de correspondencia entre el instructor y el aprendiz. Figura extraída y modificada de [6].

ciones equipadas con cámaras, son los más comunes.

d) Observación externa: Los sensores externos se utilizan para registrar los movimientos del instructor durante las actuaciones, aunque estos sensores pueden o no estar situados en el robot entrenado. La función de mapeo g_R no es directa: $g_R(z, a) \neq I(z, a)$.

El problema de la correspondencia

La identificación de cualquier forma de aprendizaje social, imitación, copia o mimetismo, presupone una noción de correspondencia entre dos agentes autónomos. Al juzgar si un comportamiento se ha transmitido socialmente, el observador debe identificar un mapeo entre el instructor y el aprendiz. Si el instructor y el aprendiz tienen cuerpos similares -por ejemplo, son animales de la misma especie, de edad similar y del mismo sexo-, entonces, para un observador humano, una correspondencia obvia es mapear las partes corporales correspondientes: brazo izquierdo del mapa del instructor al brazo izquierdo del aprendiz, el ojo derecho de los mapas del instructor al ojo derecho del aprendiz, la cola de los mapas del instructor a la cola del aprendiz. También hay una correspondencia obvia de acciones: levantar el brazo izquierdo por el instructor corresponde a levantar el brazo izquierdo por el aprendiz, la producción de señales vocales por el instructor corresponde a la producción de sonidos audibles similares por el aprendiz, recoger una fruta por el demostra-

dor corresponde a recoger una fruta del mismo tipo por el instructor. Además, hay una correspondencia en la experiencia sensorial: sonidos audibles, un toque, objetos y colores visibles, etc., evidentemente parecen ser detectados y experimentados de manera similar [43].

Qué tomar como "la correspondencia" parece relativamente claro en este caso. Como humanos, somos buenos para imitar y reconocer tales correspondencias. También está claro que la mayoría de los otros animales, robots y programas de software, por lo general no pueden reconocer tales correspondencias. Para juzgar que una conducta producida es una copia de una observada, se requiere al menos que respete dicha correspondencia. La fidelidad o precisión de la coincidencia de comportamiento obviamente puede variar, y no existe un umbral o límite absoluto que defina el éxito en oposición al fracaso de la coincidencia de comportamiento. Sin embargo, uno puede estudiar el grado de éxito en el uso de diferentes métricas y medidas de correspondencia [43].

Además, resulta que las correspondencias "obvias" entre cuerpos similares mencionados anteriormente no son las únicas posibles. Considera a un humano imitando a otro que está frente a él: si el instructor levanta su brazo izquierdo, ¿debería el aprendiz levantar su propio brazo izquierdo? ¿O debería levantar su derecha para hacer una "imagen reflejada" de las acciones del instructor? Si el instructor toma un pincel, ¿debería un aprendiz recoger el mismo pincel? o solo otro del mismo tipo? Si el instructor abre un contenedor para introducir chocolate en el interior, ¿debería el aprendiz abrir un contenedor similar de la misma manera, por ejemplo, desenvolviendo pero sin rasgar el papel circundante, o es suficiente abrir el contenedor de alguna manera? Las diferentes respuestas posibles a estas preguntas presuponen correspondencias diferentes [43].

Es sencillo recuperar la información cinemática del movimiento humano utilizando, por ejemplo, un sistema de captura de movimiento (MOCAP por sus siglas en inglés), pero imitar este movimiento con una dinámica de robot estable es un problema mucho más difícil. Un demostrador humano y un alumno robot humanoide pueden compartir una estructura cinemática similar, pero su dinámica suele ser muy diferente. Si el robot imita el movimiento cinemático exacto del ser humano, el movimiento resultante suele ser dinámicamente inestable [20].

Métricas y medidas de éxito en la correspondencia de comportamientos

Las métricas y medidas de error, pueden capturar la noción de la diferencia entre las acciones realizadas y las deseadas, pueden medir la diferencia entre los estados obtenidos y los deseados, o pueden medir la diferencia de secuencias entre ambos. Tales medidas pueden tomar valores discretos o continuos. Este es el papel de la medida de diferencia d en la declaración formal del problema de correspondencia anterior [43].

La evaluación de un candidato para el emparejamiento conductual depende en gran parte de la formalización de la medida del error d. Diferentes tipos de medidas dan como resultado diferentes tipos de emparejamiento y conducirían a tipos de aprendizaje de comportamientos que los etólogos y los psicólogos clasificarían como diferentes tipos de aprendizaje o copia social. Por ejemplo, si la medida de error d en la formulación anterior del problema de correspondencia ignora el componente de estado, entonces solo las acciones son suficientes para el éxito del intento de coincidencia de comportamiento [43].

La granularidad se refiere a la finura de la imitación, por ejemplo, en la cantidad de estados, acciones o subobjetivos coincidentes. Si la medida d ignora el componente de acción, entonces solo la secuencia de estados alcanzados es suficiente. La medida de diferencia puede requerir que ciertos estados o acciones se alcancen o coincidan estrechamente; por lo tanto, se puede usar para evaluar si se ha logrado o no una secuencia de subobjetivos, y en qué grado [43].

Formalizacion del aprendizaje por imitación

Billard y Grollman [54] establecen que la ejecución, por un humano o una máquina, de un comportamiento, define de manera implícita cierta política, la cual denotaremos Ω^y. El objetivo del aprendizaje por imitación, y también de otras técnicas de transferencia de política humano-robot tal como la programación o teleoperación, es crear una política análoga Ω^x que cause el comportamiento deseado en el robot, resolviendo el problema del "Mapeo Instructor Aprendiz". En el aprendizaje por imitación, la política es calculada de forma automática a través de algún operador U, el cual realiza el algoritmo de aprendizaje, guiado por la búsqueda de un óptimo para la métrica M. Dicha métrica sirve como medida de qué tan bien se realizó la tarea.

2.2. Robótica Evolutiva

La presente propuesta utiliza para la resolución del operador U, encargado de resolver el aprendizaje de la política y presentado en la sección 2.1.2, técnicas enmarcadas en el área de la robótica evolutiva. En esta sección se plantean las características más relevantes del área, las cuales fueron tomadas como referencia para el diseño presentado en el Capítulo 3.

La robótica evolutiva implica el uso de técnicas de computación evolutiva para desarrollar automáticamente algunas o todas las siguientes propiedades de un robot: el sistema de control, la morfología del cuerpo y las propiedades y el diseño del sistema de sensado y motor [32].

Debido a que la naturaleza de prueba y error de los algoritmos evolutivos requiere una gran cantidad de evaluaciones durante la optimización, en muchos experimentos de robótica evolutiva la optimización se lleva a cabo primero en simulación. Típicamente, un algoritmo evolutivo genera poblaciones de robots virtuales que se comportan dentro de una simulación basada en la física [11].

A cada robot se le asigna un valor de *fitness* basado en la calidad de su comportamiento. Los robots con bajo *fitness* se eliminan mientras que los robots que quedan se copian y se modifican ligeramente de forma aleatoria. Los nuevos robots se evalúan en el simulador y se les asigna un *fitness*, y este ciclo se repite hasta que haya transcurrido un período de tiempo predeterminado. El robot más adecuado puede ser fabricado como una máquina física y desplegado para realizar su comportamiento evolucionado [11].

2.2.1. Desafíos inherentes a la robótica evolutiva

Hay una serie de desafíos que enfrenta actualmente el campo, entre ellos se encuentra la transferencia de robots evolucionados en un ambiente simulado a máquinas físicas; problemas de escalabilidad; y la dificultad de definir funciones apropiadas de *fitness* para medir automáticamente el comportamiento [11].

El problema del *Reality Gap*.

Tanto la evolución biológica como la artificial son conocidas por explotar la relación potencial entre el animal (o robot) y su entorno para producir nuevos comportamientos. Por ejemplo, la propiedad ligera de las plumas, que se cree que evolucionó originalmente para la regulación del calor, se explotó más tarde

para volar [11].

Como ejemplo de las tendencias explotadoras de los algoritmos evolutivos aplicados a los robots, inicialmente se diseñó un robot para balancearse a lo largo de una viga suspendida. El robot estaba compuesto por un cuerpo principal colgado debajo de dos brazos y un pesado paquete de baterías sujeto al cuerpo principal. Poco a poco, el algoritmo evolutivo descubrió políticas de control para el robot que explotaban la presencia de las baterías, en lugar de luchar contra su peso. Estas políticas de control harían que el robot se moviera de forma tal que la batería lo hiciera hacia adelante debajo de su cuerpo antes de que cambiara de mano. Esto provocaría que el centro de masa del robot se moviera hacia adelante, requiriendo mucha menos fuerza para liberar el contacto con la viga y agarrarla más hacia delante. Esto imita la forma en que los primates explotan el peso de sus cuerpos, como un péndulo, para llevarlos al alcance de una nueva rama de árbol. También es una reminiscencia de la dinámica pasiva, productora de un ahorro de energía, de la locomoción bípeda [11].

Sin embargo, si los robots se optimizan en un simulador, la evolución artificial puede explotar simplificaciones o imprecisiones en cómo se simula la física. Dichas políticas de control evolucionadas pueden no reproducir el comportamiento deseado cuando se transfieren desde robots simulados a robots físicos. Por ejemplo, si el ruido no es modelado en el simulador, una política de control puede evolucionar para generar un comportamiento basado en un rango muy estrecho de valores de sensor. Si esta política de control se transfiere a un robot físico con un sensor que registra un rango más amplio de valores debido a limitaciones en sus componentes electrónicos o mecánicos, es posible que el robot físico no se comporte como se esperaba. Esta falla de las soluciones evolucionadas para "cruzar la brecha" de la simulación a la realidad se conoce como el problema del *reality gap* y es uno de los principales desafíos que enfrenta el campo de la robótica evolutiva [11].

Combinatoria de la evaluación

Es sabido que el tiempo requerido para evaluar un solo robot puede crecer exponencialmente con la cantidad de parámetros utilizados para describir el ambiente de la tarea. Por ejemplo, considere un robot que debe tomar m diferentes objetos bajo n diferentes condiciones de iluminación. Cada robot

debe ser evaluado para determinar qué tan bien agarra cada objeto bajo cada condición de iluminación, requiriendo $m \times n$ evaluaciones por robot. Si hay p parámetros que describen el entorno de la tarea y cada parámetro tiene s configuraciones diferentes, entonces cada robot debe evaluarse s^p veces [11].

Este es un desafío serio en el campo de la robótica evolutiva que aún no se ha resuelto. Sin embargo, una posible solución al mismo se puede abordar utilizando la coevolución. Considere una población de robots y una segunda población de entornos de tareas que compiten entre sí. Los robots evolucionan para tener éxito cuando se exponen a entornos extraídos del conjunto de los mismos en evolución, y los entornos lo hacen para frustrar las capacidades de los robots en evolución. Esto no es diferente de las presas que evolucionan para eludir a los depredadores, mientras que éstos últimos lo hacen para atrapar presas. Este enfoque podría, en el futuro, usarse para desarrollar robots que alcancen generalizar con éxito contra un subconjunto de entornos de tareas que puedan encontrarse cuando se fabriquen e implementen [11].

Diseño de la función de *fitness*

El objetivo original y continuo de la robótica evolutiva, es hacer la menor cantidad de suposiciones sobre la forma final del robot o el tipo de comportamiento que se debe generar. Sin embargo, diseñar una función de *fitness* que descubra rápidamente soluciones deseables sin sesgar hacia soluciones particulares, es notoriamente difícil. Por esta razón, se han realizado esfuerzos en el campo de la robótica evolutica para eliminar por completo el uso de una función de *fitness*. Un ejemplo reciente es la técnica de *novelty search*, que comienza con soluciones de candidatos simples y crea gradualmente soluciones más complejas a medida que avanza la optimización. El *fitness* de una solución dada es simplemente cuánto difiere de las soluciones generadas previamente. Este enfoque logra la generación de caminatas en robots bípedos simulados, un problema notoriamente difícil en robótica [11].

2.2.2. *Novelty Search with Local Competition*

Un desafío de larga data en la "vida artificial", es crear un algoritmo capaz de descubrir una gran diversidad de criaturas artificiales interesantes. Si bien los algoritmos evolutivos son buenos candidatos, por lo general convergen a una sola especie de criaturas. Para superar este problema, Lehman y Stanley

propusieron recientemente un método llamado *Novelty Search with Local Competition*. Este método, basado en algoritmos evolutivos multiobjetivo, combina las capacidades de exploración del algoritmo *novelty search* con una competencia de rendimiento entre individuos similares. La búsqueda de novedad con competencia local optimiza simultáneamente dos objetivos para un individuo c : (1) el objetivo de novedad ($novelty(c)$), que mide qué tan novedoso es el individuo en comparación con los encontrados anteriormente, y (2) el objetivo de competencia local ($Qrank(c)$), que compara la calidad del individuo ($quality(c)$) al rendimiento de los mismos en un vecindario, definido con una distancia morfológica, entendiendo distancia entre el individuo i y j como: $d(i,j) = 1 - sim(i,j)$, siendo $sim(i,j)$ la similitud entre el individuo i y el j. Con estos dos objetivos, el algoritmo favorece a los individuos que son nuevos, aquellos que son más eficientes que sus vecinos y aquellos que son intercambios óptimos entre la novedad y la "calidad local". Ambos objetivos se evalúan gracias a un repositorio que registra todas las familias de individuos encontradas y permite que el algoritmo defina vecindades para cada individuo. El objetivo de novedad se calcula como la distancia promedio entre el individuo actual y sus vecinos, y el objetivo de competencia local es el número de vecinos que c supera el criterio de calidad $quality(i)$. Los autores aplicaron con éxito este método para generar un gran número de criaturas con diferentes morfologías, todas capaces de caminar en línea recta. El algoritmo encontró una población heterogénea de diferentes criaturas, desde pequeños elementos saltarines hasta cuadrúpedos imponentes, todas caminando a diferentes velocidades de acuerdo con su estatura [23].

2.3. Locomoción bípeda

Debido a que la arquitectura de aprendizaje por imitación propuesta en este trabajo es validada utilizando el comportamiento motor de la marcha bípeda, también fueron relevados trabajos y enfoques importantes donde se estudia dicho problema, tanto desde el punto de vista de su modelado como desde su evaluación.

La caminata bípeda se ha vuelto más popular estos años, ya que los robots pueden hacer su trabajo en un entorno difícil y complejo por medio de la locomoción con patas. Pero la marcha bípeda todavía necesita más tiempo para lograr su objetivo [57].

2.3.1. Métodos para representar la marcha bípeda

Se han presentado varios métodos que se pueden clasificar en dos grandes grupos, uno está basado en modelos y otro es libre de éstos. En el enfoque basado en modelos, primero se diseña el modelo físico del robot y luego se construirá un controlador para él. El enfoque del punto de momento cero (ZMP, por sus siglas en inglés) y "péndulo invertido" son dos métodos de este enfoque [57]. En el enfoque sin modelo, la información sensorial se usa para hacer movimientos, no hay ninguna consideración del modelo físico en este método y las habilidades se implementan de una manera más fácil [57].

Los caminadores dinámicos pasivos y los Generadores Centrales de Patrones son métodos importanes del enfoque libre de modelo [57].

Modelo del péndulo invertido

El modelo de péndulo invertido de la marcha humana es uno de los más simples utilizados en la cinética humana para describir el cambio en la energía cinemática y potencial. El movimiento del péndulo invertido ofrece una buena visión de la trayectoria del Centro de Masa (COM, por sus siglas en inglés) del cuerpo humano. Al agregar una restricción que corrige el movimiento del COM en un plano horizontal, se puede derivar un modelo dinámico lineal, que describe una trayectoria para el COM adecuada para caminatas bípedas [59].

Métodos basados en ZMP

En la actualidad, los métodos dominantes para la locomoción bípeda con humanoides utilizan el criterio del ZMP para garantizar que el robot no se caiga.

El control del cuerpo del robot, de manera que el ZMP se asiente dentro del polígono de soporte del pie del robot, asegura que el pie permanezca plantado en el suelo, suponiendo que la fricción sea lo suficientemente alta para evitar el deslizamiento. El ZMP se puede usar para planificar patrones de marcha que hacen que el robot sea dinámicamente estable mientras camina.

Convencionalmente la locomoción bípeda se genera fuera de línea resolviendo una ecuación diferencial ordinaria con respecto al movimiento del centro de gravedad (COG, por sus siglas en inglés) dada una trayectoria deseada de la ZMP [26]. La locomoción bípeda generada sólo con el uso de ZMP puede

no ser similar a la humana. Generar un modo de andar humano en un robot humanoide bípedo ha sido un gran desafío [26].

Generadores Centrales de Patrones

Un enfoque frecuentemente utilizado para permitir que los robots humanoides caminen de manera estable, es aplicar heurísticas y configurar manualmente los patrones de marcha y sus parámetros. Existen técnicas basadas en generadores de patrones centrales (CPG, por sus siglas en inglés) para generar trayectorias conjuntas utilizando osciladores no lineales. En estos enfoques, es un problema difícil encontrar los parámetros adecuados para lograr un paso de marcha estable [61].

Otros autores utilizan técnicas de optimización para calcular estos parámetos, como propone Shafii en [57] donde además se utilizan series truncadas de Fourier para modelar los patrones de marcha.

Hacia una marcha más eficiente

Ejemplos de locomoción notablemente eficiente abundan en la naturaleza. Desde chitas corriendo hasta lémures saltando, la variedad y belleza del movimiento animal ha inspirado a los roboticistas durante décadas. Sin embargo, algunos de los ejemplos más impresionantes de caminatas humanoides, por ejemplo, el Asimo de Honda o el Atlas de Boston Dynamics, han empleado un control conjunto de alta impedancia para lograr una locomoción estable; en este contexto, la impedancia se refiere a la relación dinámica entre la posición (o velocidad) y la fuerza. Como resultado, la fluidez de sus movimientos está lejos de la de su contraparte biológica. De hecho, estos robots gastan un orden de magnitud más de energía que un humano típico al caminar [62].

Por el contrario, los caminadores dinámicos pasivos de McGeer [62] han podido lograr modos de andar sorprendentemente parecidos a los humanos sin ninguna actuación, confiando únicamente en su dinámica corporal cuidadosamente diseñada. Estas máquinas se movieron hacia abajo en terrenos inclinados, confiando en la conversión del potencial en energía cinética para equilibrar las pérdidas introducidas por los impactos y la fricción. Ejemplos posteriores de bípedos mínimamente accionados, capaces de caminar por su propia potencia en terreno llano, dieron los primeros pasos hacia la realización de máquinas de caminar eficientes.

2.3.2. Criterios y métricas necesarios para evaluar caminatas robóticas

La locomoción estable representa uno de los retos principales de la robótica humanoide, para la cual no existen actualmente estándares bien definidos que permitan la evaluación comparativa de las diferentes técnicas [58]. El principal problema es debido al hecho de que el control y los aspectos mecánicos de una máquina con piernas son intrínsecamente complejos y por tanto la evaluación y comparación de caminatas robóticas es una tarea difícil [58, 47, 31].

La búsqueda de criterios y métricas para la comparación de la locomoción robótica ha recibido atención durante la última década, la principal razón de ello es el interés en fijar estándares para simplificar la comparación de diferentes diseños e implementaciones de robots [58, 37].

En el campo de la robótica humanoide las métricas propuestas para evaluar tareas motoras se centran, por lo general, solamente en el resultado obtenido en la ejecución de dicha tarea, en lugar de examinar comportamientos sensomotores particulares [58].

La semejanza humana es considerada por varios autores como un criterio significativo para la evaluación de la calidad del diseño, bajo la perspectiva de suavidad, versatilidad y eficiencia energética [22]. Sin embargo no hay disponible una métrica generalizada, completa y aceptada, para evaluar la locomoción humana [58].

En el último siglo se ha recopilado una enorme cantidad información cuantitativa sobre la cinemática y la cinética a partir de estudios en humanos, y se han derivado varias métricas de los mismos. Producto de dichos estudios surgen métricas candidatas apropiadas para describir el comportamiento humano típico, que pueden aplicarse fácilmente al escenario robótico [58].

En [58] proponen un subconjunto de las posibles características humanas relacionadas con tareas motoras, particularmente la locomoción, que permita ser aplicado como métrica a diferentes plataformas bípedas más allá de su tamaño, peso, número de grados de libertad, o arquitectura de control. Para ello se definen diferentes escenarios y criterios de comparación. Los mismos criterios son relevados y utilizados en otros trabajos [37, 63, 36].

A continuación se enumeran las características de la marcha que han sido relevadas como más utilizadas:

Estabilidad

El criterio más general para la estabilidad robótica (y humana) es la robustez a la caída. La condición de caída es fácilmente detectable por inspección visual. Para convertir este criterio en métricas cuantitativas, se deben considerar magnitudes máximas que el robot puede tolerar antes de caerse [58].

Con mucha frecuencia se utiliza como métrica la distancia recorrida por el robot en un tiempo fijo, solo la distancia hacia adelante es tenida en cuenta para evitar comportamientos de caminata hacia atrás o los costados [37, 25, 31, 47].

Otros investigadores proponen utilizar la velocidad máxima y mínima, como también el número máximo de pasos antes de caerse. En escenarios más específicos se pueden definir las siguientes métricas: para caminatas en pendientes, el máximo permitido de inclinación respecto al suelo; para empujes, la máxima fuerza externa tolerada por el bípedo durante las fases de apoyo sencillo y apoyo doble; para terrenos irregulares, la dimensión máxima del obstáculo; y en el caso del terreno blando, la máxima suavidad del suelo; en condición de carga de peso, la carga máxima permitida; en caso de trayectoria curvada, un indicador de la la máxima fuerza centrífuga, como por ejemplo, la relación entre velocidad y radio de curvatura.

Para las transiciones voluntarias, la estabilidad apenas puede cuantificarse en una escala continua, por lo que estos experimentos muchas veces no pueden ser realizados en un solo ensayo; por esta razón, en la bibliografía se propone medir la tasa de éxito a través de una cantidad de ensayos [58].

Análisis cinemático

A nivel cinemático, el procedimiento básico para comparar robots con humanos es la correlación directa de perfiles cinemáticos angulares. En particular, la flexión de rodilla, los movimientos de la pelvis, y la colocación del pie, fueron relevados por [58] como indicadores fuertes de la similitud con los humanos.

En un nivel más global, la armonía de la marcha ha sido propuesta como una métrica para medir la sincronía y la simetría de los movimientos de la marcha de todo el cuerpo [33], obtenido por parámetros espacio/temporales o armónicos de aceleración del COM [58]. Según Minetti [42], un gran nivel de conciencia está emergiendo hoy sobre la importancia de la simetría en la marcha humana. Esto es impulsado por considerar que un patrón de marcha más simétrica podría estar asociado a algún ahorro de energía metabólica.

Otro buen candidato para comparar la cinemática entre robots y humanos es el método de distorsión dinámica temporal (DTW, por su sigla en inglés) [38], que mide la similitud temporal entre secuencias que varían en tiempo o velocidad [58]. Esta técnica es utilizada frecuentemente en el reconocimiento de voz; comparada con otras técnicas para la medición de similitud de series temporales tiene la ventaja de que las series de entrada no necesitan estar alineadas respecto al tiempo, y el error causado por la relación de no linealidad puede ser evitado. DTW ha sido aplicado ampliamente al análisis de secuencias de video, audio y datos gráficos. Inspirado por estas aplicaciones, DTW ha sido aplicado en el trabajo de Wang y colaboradores [60] también como medida intuitiva y eficaz para la evaluación de ejecuciones de la marcha.

Una forma alternativa de evaluar la similitud en el movimiento respecto al humano es mediante una inspección visual utilizando la pantalla de movimiento de puntos [58].

En [2] se postulan cinco criterios para evaluar la similitud dinámica entre bípedos de cualquier tamaño. Tres de estos criterios se refieren al movimiento: I) fase contra-lateral relativa del pie igual, la cual permite medir la simetría del movimiento, II) igual factor de trabajo, que indica el porcentaje de la fase de postura dentro de un ciclo de marcha, y III) igual NÚMERO DE FROUDE.

Los métodos descritos anteriormente se aplican en general en condiciones no perturbadas; el escenario perturbado también puede ser útil para probar una característica típica del movimiento humano: el cumplimiento de las perturbaciones externas.

Una métrica simple utilizada también frecuentemente, a nivel cinemático, es la comparación por velocidad, calculada tomando la distancia recorrida por el bípedo en una medida fija de tiempo.

Análisis cinético

La locomoción humana se caracteriza por comportamientos dinámicos pasivos y activos intercalados a lo largo de un ciclo de marcha; se entiende por ciclo de marcha desde que hay un "evento" con un pie hasta que se vuelve a repetir el mismo "evento" con el mismo pie, lo que es equivalente a dos pasos. Un enfoque clásico para analizar la cinética es medir y luego comparar los perfiles de los momentos conjuntos. Recientemente, se han propuesto técnicas como la *medida de marcha pasiva* (PGM, por sus siglas en inglés) y la *medi-*

ción dinámica de la marcha (DGM, por sus siglas en inglés) para cuantificar las características pasivas y dinámicas de la locomoción humana [58].

Las fuerzas internas y externas en el caminar humano son variables repetibles, que podrían utilizarse como oráculos para la comparación con sus homólogos humanoides. En [2] se postulan dos criterios para dinámicas similares en cuanto a las fuerzas, además de los anteriores tres criterios basados en el movimiento: IV) las fuerzas sobre los pies son iguales a los múltiplos del peso corporal; y V) las salidas de potencia son proporcionales al producto del peso corporal por la velocidad [58].

Eficiencia Energética

La similitud con el movimiento humano trae asociado como beneficio la reducción de los costos energéticos [37, 42, 17, 63].

Un punto de referencia ampliamente utilizado para evaluar la eficiencia de la caminata, es el costo específico de transporte (c_t), definido como la relación de la energía consumida y el peso multiplicado por la distancia recorrida. El coste energético específico del transporte (c_{et}) comprende la energía total consumida, incluyendo el trabajo positivo y negativo de los actuadores y los costes de energía relacionados con la electrónica. El coste mecánico específico del transporte (c_{mt}) es necesario cuando se quiere aislar el trabajo mecánico positivo y comparar más fiablemente los costes energéticos de diferentes plataformas robóticas independientemente de los aspectos de control. La energía y el consumo de la misma también se pueden medir a nivel de actuador, midiendo el torque y velocidad de cada articulación, pudiendose calcular la potencia mecánica (p_{mec}) como el producto entre el torque y la velocidad, para cada actuador.

2.4. Conclusiones

En la Sección 2.1 se plantean las preguntas de investigación centrales del área del aprendizaje por imitación, en el Capítulo 3 veremos como estas preguntas sirvieron de guía en el diseño de la solución propuesta.

Una de las motivaciones en la utilización de la técnica de aprendizaje por imitación radica en que la forma en que funciona el proceso reduce el espacio de búsqueda de la solución.

Las funciones de mapeo g_R y g_E son definidas con el objetivo de conceptualizar el proceso de registro y correspondencia entre el instructor y el aprendiz, estas funciones crean un marco de trabajo utilizado para organizar las soluciones relativas al aprendizaje por imitación.

Argall plantea una categorización del mapeo entre el instructor y el aprendiz según la función de *Record Mapping* y la función de *Embodiment Mapping* que se utilice, donde se diferencia primeramente entre demostración e imitación, para luego entrar en detalle del tipo específico de mapeo.

Luego nos centramos en los desafíos que presenta dar respuesta a la pregunta de ¿Cómo imitar?, la cuál es una de las más importantes a la hora de implementar un comportamiento por imitación, debido a la dificultad que presenta el problema de la correspondencia.

El diseño manual de un robot móvil autónomo y adaptable es extremadamente difícil. Como alternativa, las computadoras pueden "evolucionar" poblaciones de robots en un simulador para exhibir un comportamiento útil y luego fabricar versiones físicas de las mejores.

Este enfoque, conocido como robótica evolutiva, cambia la manera en que vemos la robótica: en lugar de técnicas de aprendizaje automático mejorando los comportamientos de un robot diseñado a mano, el enfoque cambia a la creación de un sistema evolutivo que diseña y fabrica continuamente diferentes robots con capacidades que aumentan conforme avanzan las generaciones [11].

Los algoritmos evolutivos y los simuladores físicos de vanguardia permiten optimizar todos los aspectos del plan corporal y la política de control de un robot simultáneamente en un período de tiempo razonable [11].

No se encontró en el estado actual de la técnica de aprendizaje por imitación ningún trabajo que utilice técnicas evolutivas para la implementación de la función de mapeo g_E, su uso podría mejorar los resultados en el área en comparación con las soluciones existentes, ya que como se plantea en la sección 1.1 la mayoría de las habilidades cognitivas humanas de nivel superior pueden estar arraigadas evolutivamente a nuestra capacidad de observarnos e imitarnos unos a otros, por lo que implementar el mecanismo de aprendizaje por imitación utilizando técnicas evolutivas promete ser una fuente de soluciones también en los robots.

A diferencia de las técnicas evolutivas, las técnicas tradicionales para el diseño de marchas bípedas requieren de un modelo mecánico analítico, que muchas veces es implementado mediante el modelo de ZMP o el del péndulo

invertido, los cuales requieren de una especificación precisa de la dinámica del robot y su entorno para lograr una marcha estable [19]. Otros autores toman ventaja de la naturaleza periódica de la caminata y modelan cada articulación utilizando osciladores no lineales, resultando en una implementación más sencilla pero de apariencia artificial [61].

Dentro del conjunto de las tareas motoras necesarias para los robots bípedos, la marcha es un caso paradigmático, en parte debido a su complejidad, donde la estabilidad debe ser garantizada contando con una pequeña área de apoyo junto con las complicaciones inherentes al control de varios grados de libertad del bípedo [47]. Esta complejidad ha empujado a los investigadores a desarrollar soluciones que no toman en cuenta la importancia de imitar los atributos cualitativos de la marcha humana y tratan sólo de optimizar propiedades tales como la velocidad o la estabilidad del torso a la hora de implementar marchas en humanoides. Por lo general los patrones de caminata óptimos resultantes al utilizar este acercamiento no se parecen a la marcha humana. Sin embargo, la semejanza con el ideal humano debe tenerse en cuenta al generar patrones de marcha para robots humanoides. Como vimos en 1.1, estos robots están diseñados para interactuar de forma natural con los seres humanos y por lo tanto es importante que sus movimientos parezcan lo más humanos posibles [61].

Imitar a los humanos es el oráculo buscado cuando el fin es obtener el mejor desempeño, desde el punto de vista de la eficiencia energética, en caminatas bípedas. En gran parte de la bibliografía relevada [37, 42, 17, 63, 61] pueden verse trabajos relacionados con locomoción donde esta métrica es tenida en cuenta. Otros autores [61] también plantean inspirarse en las caminatas humanas como métrica bajo el argumento que los robots humanoides son diseñados con el objetivo de interactuar con humanos, por lo que es importante que sus movimientos también luzcan como humanos.

Minetti [42] plantea que la trayectoria en el espacio tridimencional del COM durante la marcha constituye la "firma de locomoción" para diferentes tipos de marcha y velocidades, tanto para humanos como para otras especies con piernas. Este estudio nos permite tener un mecanismo para medir similitud entre marchas; otra posibilidad para realizar esta medición es la aplicación del algoritmo DTW un ejemplo de su uso puede encontrarse en el trabajo de Wehner [61]. También fueron relevados en la bibliografía otros métodos para la medición de similitud entre caminatas [58, 4].

En el estado del arte es frecuentemente utilizada como métrica la distancia recorrida por el robot, en sentido recto hasta caerse [37, 25, 31, 47]; a pesar de ello trabajos de referencia en el área de robótica evolutiva como [25] muestran que el estado del arte actual en caminatas bípedas generadas por técnicas de aprendizaje automático es aún inmaduro, en [25] Eaton releva un trabajo de Miyashita del año 2003, donde la caminata generada soporta un máximo de 10 pasos antes de caerse; pese a la característica periódica de la locomoción bípeda, obtener una marcha en un robot que recorra largas distancias sin caerse o incluir un sistema que controle independientemente el equilibro, no es una tarea trivial.

El control de caminatas bípedas es un problema complejo donde intervienen muchos factores; su evaluación, como puede verse en lo relevado en este documento, no es ajena a esta dificultad, por lo que utilizar métricas que permitan evaluar diferentes atributos aportará un conocimiento más cabal acerca de su calidad.

Capítulo 3

Propuesta de la solución

En este trabajo, se presenta la arquitectura propuesta para la resolución del problema del aprendizaje de tareas motoras en robots humanoides, la misma se basa en el uso de la técnica del aprendizaje por imitación y no requiere de un modelo analítico de la tarea a ser aprendida.

El primer acercamiento al imitar a un humano realizar una tarea motora es utilizar soluciones mapeadas directamente desde el instructor al robot, pero al realizarlo no se obtienen soluciones factibles; esta dificultad es conocida en el área como el problema de la correspondencia, el cual introducimos en la Sección 2.1.2; en nuestra propuesta el mismo es formulado como un problema de optimización, donde su implementación se realiza mediante la utilización de técnicas evolutivas, ver figura 3.1 donde el problema de la correspondencia representado con la línea punteada, es calculado a partir de la aplicación del algoritmo evolutivo, representado por las líneas llenas.

Como fue mencionado en la Sección 2.2 las técnicas correspondientes al área de la robótica evolutiva requieren de una gran cantidad de evaluaciones, por lo que son realizadas en un ambiente virtual que simula las propiedades físicas del robot y su entorno.

El algoritmo evolutivo genera poblaciones de caminatas que son puestas a prueba dentro del ambiente virtual; en cada generación el algoritmo evolutivo construye soluciones que se desempeñan mejor, según la métrica definida, hasta obtener finalmente soluciones factibles en el robot simulado, ver figura 3.1.

Las soluciones factibles obtenidas para la plataforma virtual son seleccionadas para ser utilizadas en la plataforma real. Es aquí que nuevamente nos enfrentamos al problema de la correspondencia, donde se deben adaptar los

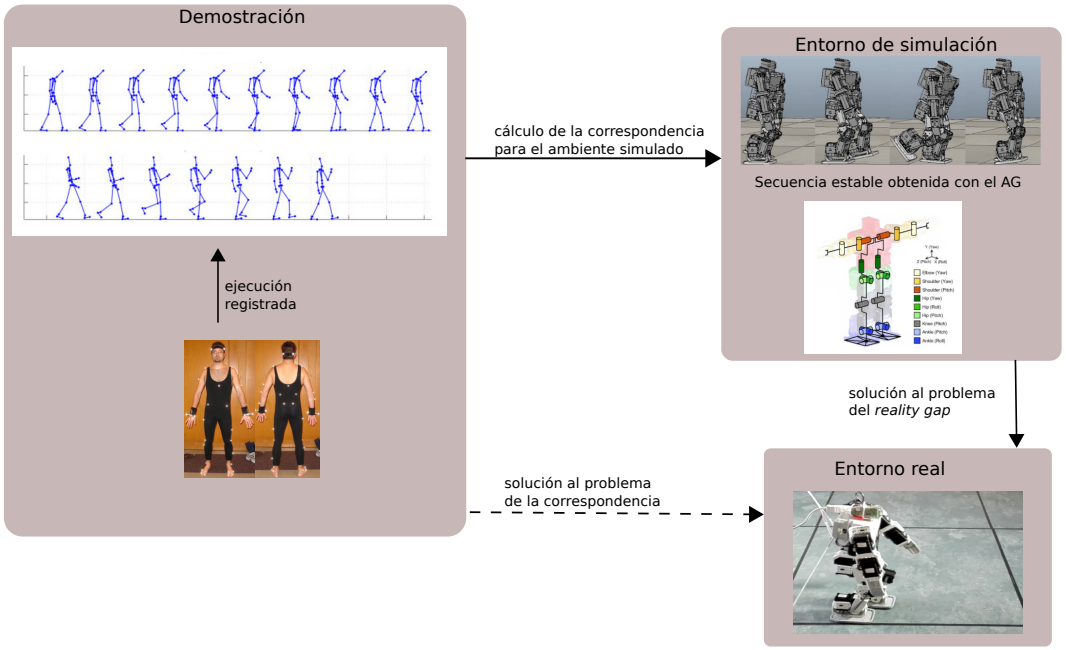

Figura 3.1: Componentes de la arquitectura propuesta para el aprendizaje de tareas motoras en robots humanoides.

valores que representan a la tarea motora ejecutada en la plataforma virtual a valores que también representen de forma exitosa la misma tarea en la plataforma real. Esto es debido a que las soluciones factibles en el ambiente virtual, adolecen de problemas inherentes a las simplificaciones de la física realizadas por el modelo implementado en el simulador. Este problema, presentado en la Sección 2.2.1, se conoce como el *reality gap* en el área de la robótica evolutiva, y para su solución se empleará el mismo algoritmo evolutivo utilizado, a excepción de la función de *fitness*, para resolver el problema de la correspondencia, ver figura 3.1. En el Capítulo 4 se presentan los resultados para al problema guía seleccionado.

La técnica evolutiva propuesta ha sido implementada mediante un algoritmo genético, el mismo utiliza como representación de las soluciones una SERIE TEMPORAL de poses que especifican el movimiento a imitar por el robot, el proceso de generación de esta serie será explicado en la Sección 3.4.

La comunidad científica ha sistematizado los problemas del área de aprendizaje por imitación en un conjunto de preguntas de investigación, presentadas en la Sección 2.1.1 las cuales sirvieron como guía para este trabajo y analizaremos a continuación.

3.1. ¿Cómo sabe el robot qué imitar?

Se utiliza un enfoque multi-maestro, tal como describe Calinon en [14], para permitirle al robot disponer de un conjunto suficientemente grande de información sensorial que le posibilite determinar que acciones imitar para generalizar la tarea. Esta información llega al robot como un flujo de datos sensoriales codificado mediante series temporales de poses.

El proceso evolutivo busca explotar los aspectos más relevantes en las demostraciones de tareas motoras ejecutadas por el instructor, desde el punto de vista cualitativo, al ser ejecutadas por el aprendiz. Intentando mediante este mecanismo dar respuesta a la pregunta de **¿qué imitar?**. Para determinar dicha respuesta es necesario descartar poses que no aportan a la ejecución de la tarea y seleccionar otras que permitan cumplirla. Es aquí donde un enfoque multi-instructor cobra importancia, permitiendo al aprendiz disponer de otros ejemplos a evaluar, los que pueden ser más cercanos a sus capacidades o limpios de elementos que no aportan a la tarea concreta a ser aprendida (vicios del instructor, u elementos circunstanciales que no corresponden con la ejecución de la tarea); para poder resolverlo es importante la definición de una función de *fitness* que permita evaluar las características deseables para la ejecución de una tarea motora y determinar mediante el operador de selección si son de relevancia para el problema.

3.1.1. Registro de movimientos

Para poder determinar qué imitar es necesario contar con una representación de la información a ser procesada.

Los datos fueron registrados mediante la técnica de sensores en el maestro utilizando equipamiento para la captura de movimientos (MOCAP, según sus siglas en inglés) mediante marcadores; el procedimiento consiste en colocarlos, de manera que generen un buen contraste respecto a la superficie de aplicación, en los puntos de interés del cuerpo humano que se quieren registrar; teniendo como referencia otros marcadores fijos de tamaño conocido y a distancia conocida, se puede calcular las distancias asociadas entre marcadores y tener una representación en el espacio tridimensional cartesiano de cada marcador para cada instante de tiempo.

En etapas iniciales el registro de movimientos se realizó con equipamiento

Figura 3.2: Marcadores utilizados por el Carnegie Mellon Graphics Lab, extraido de [29]

de la Unidad de Investigación de Biomecánica de la Locomoción Humana [1]; en estas etapas se tomó conocimiento con la tecnología de MOCAP y las herramientas para manipulación de los datos.

Para los resultados presentados en este trabajo la información del registro de movimientos utilizada se tomó de la base de movimientos libre de la Universidad de Carnegie Mellon (CMU, por sus siglas en inglés), MOCAP database[2], ya que se disponía de una gran cantidad y tipos de caminatas, simplificando el proceso de obtención de la información. La información de MOCAP fue registrada utilizando 41 marcadores, como puede verse en la Figura 3.2 y cuenta con un registro de aproximadamente 2500 movimientos variados, ejecutados por 140 instructores tomados a una frecuencia de 120 Hz. Una ventaja de esta base de datos y otras libres que se encuentran disponibles, es la posibilidad de tener los datos en formatos estándar para la captura de movimientos, como ser Biovision Hierarchy (BVH) donde se especifica la estructura del esqueleto generado por los marcadores y las series temporales de los mismos, como puede verse en la Figura 3.3. Estos formatos permiten utilizar PARSERS y programas de edición ya disponibles en la comunidad.

[1] Perteneciente a la Facultad de Medicina de la UdelaR
[2] http://mocap.cs.cmu.edu/

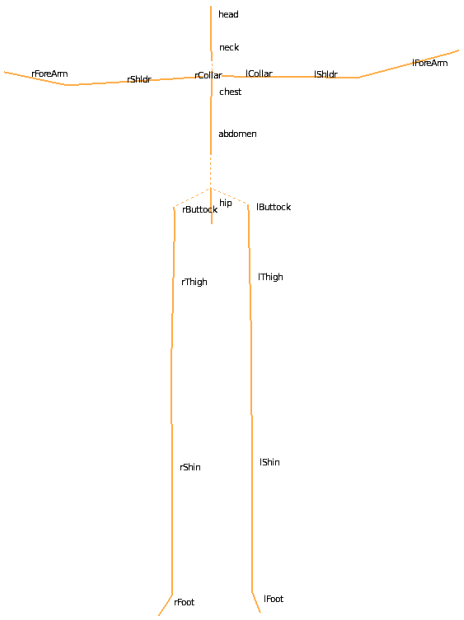

Figura 3.3: Representación jerárquica de los marcadores utilizados, para el cálculo de ángulos, correspondiente al modelo BVH utilizado.

3.2. ¿Cómo el robot asigna a las acciones observadas comportamientos de respuesta?

Basado en el marco teórico presentado en la Sección 2.1.2, las funciones *Record Mapping* y *Embodiment Mapping* son definidas para resolver el mapeo entre el instructor y el aprendiz.

La función de *Record Mapping* (g_R) nos permitirá registrar las ejecuciones de los instructores a partir de la técnica de sensores en el maestro, ver Sección 2.1.2. Por otro lado la función de *Embodiment Mapping* (g_E) será la encargada de resolver el problema de la correspondencia, dado que las demostraciones del instructor al ser ejecutadas en el agente aprendiz producen resultados muy alejados a lo exhibido por el instructor. Esto puede deberse a diversas razones, siendo las principales: interferencia en los medios de captura utilizados, aproximaciones al realizar el cambio de representación o la más importante, diferencias motrices entre el instructor y el aprendiz. Como fue mencionado en la Sección 2.1.2 la dinámica del cuerpo del instructor suele ser muy diferente a la del aprendiz, particularmente para las plataformas robóticas humanoides actuales; si el robot imita el movimiento cinemático exacto del ser humano, el movimiento resultante suele ser dinámicamente inestable.

Como veremos en la Sección 3.4.1, el algoritmo genético propuesto para

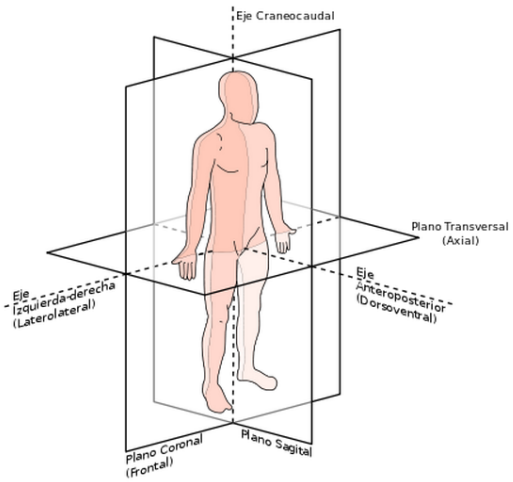

Figura 3.4: Planos corporales; imagen de dominio público con licencia (CC BY-SA 3.0).

resolver el problema de la correspondencia define implementaciones para los operadores clásicos evolutivos; particularmente mediante el uso del operador de mutación, se propone explorar variantes en las demostraciones, que permitan ajustar los movimientos imitados a las capacidades motrices del agente aprendiz. A partir de la serie temporal de poses calculadas el algoritmo genera una nueva serie que lleva a secuencias estables también en el aprendiz, aportando respuestas a la pregunta de **¿cómo imitar?**.

El robot seleccionado utiliza motores servo como actuadores para implementar las articulaciones de miembros. El control de dichos motores permite posicionarlo en un ángulo determinado entre 0 y 300 grados, por lo que se decidió realizar un mapeo de los ángulos de las articulaciones principales del instructor a las definidas en el agente aprendiz. Para esto se deben seleccionar los marcadores adecuados que permitan calcular los ángulos de interés para el movimiento del agente robot. Utilizando 3 marcadores quedan determinados 3 puntos del espacio cartesiano a partir de los cuales se pueden representar 2 vectores coincidentes en el punto correspondiente con la articulación, al proyectar los vectores sobre el plano de trabajo del actuador: sagital, frontal o transversal, ver Figura 3.4; al calcular el ángulo formado, por la proyección de los vectores, queda determinado el valor angular del actuador para un instante de tiempo. En la Figura 3.5 puede verse una representación de los grados de libertad controlables del robot, permitiendo identificar el plano en que trabaja cada actuador.

Al tomar un mapeo en base a los ángulos, es posible integrar varias bases de movimientos, aunque éstas manejen diferentes puntos donde colocar los marcadores. Solo es necesario tener marcadores que permitan determinar los ángulos de interés.

A partir de la serie temporal con las posiciones de los marcadores y el procesamiento para determinar los ángulos de interés, se obtiene un vector que especifica el ángulo de cada actuador; llamaremos a este vector pose, obteniendo una nueva serie temporal de poses, con la que el algoritmo propuesto trabajará, ver Figura 3.8.

3.2.1. Cálculo de ángulos

Para realizar el cálculo de los ángulos de la pose, es importante seleccionar la tripla de marcadores que mejor logre describir el rango de actuación que se quiere modelar. La construcción motora del robot es tal, que cada motor describe un rango de movimiento coincidente con alguno de los planos corporales de los ilustrados en la Figura 3.4. Por ejemplo si se quiere calcular el ángulo correspondiente a la rodilla izquieda (L_Knee_Pitch), ver Figura 3.5, utilizaremos los siguientes marcadores: el muslo izquierdo (L_Thigh), la tibia izquierda (L_Shin) y el talón del pie (L_Foot), ver Figura 3.3, los que describen en el plano sagital el campo de movimiento controlable por el motor Knee Pitch, como puede verse en la Figura 3.5

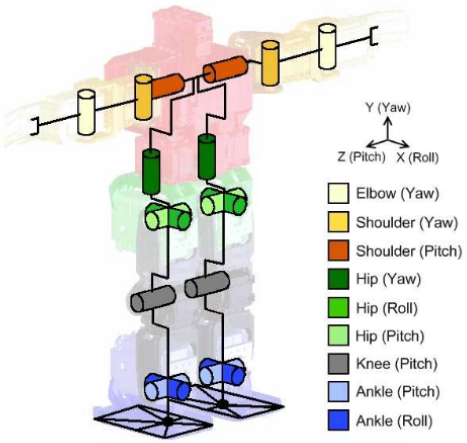

Figura 3.5: Grados de libertad controlables del robot aprendiz a utilizar; imagen de domino público extraida de [40].

Tabla 3.1: Marcadores utilizados para el cálculo del valor angular de cada articulación del robot.

articulación	**m1**	**m2**	**m3**
R_Shoulder_Yaw	rCollar	chest	rShldr
R_Shoulder_Pitch	rShldr	rForeArm	abdomen
R_Hip_Yaw	rShin	rThigh	rFoot
R_Hip_Roll	rThigh	abdomen	rShin
R_Hip_Pitch	rThigh	abdomen	rShin
R_Knee	rShin	rThigh	rFoot
R_Ankle_Pitch	rFoot	rShin	End Site12
R_Ankle_Roll	rFoot	rShin	End Site12
R_Elbow_Yaw	rForeArm	rShldr	rHand
L_Shoulder_Yaw	lCollar	chest	lShldr
L_Shoulder_Pitch	lShldr	lForeArm	abdomen
L_Hip_Yaw	lShin	lThigh	lFoot
L_Hip_Roll	lThigh	abdomen	lShin
L_Hip_Pitch	lThigh	abdomen	lShin
L_Knee	lShin	lThigh	lFoot
L_Ankle_Pitch	lFoot	lShin	End Site13
L_Ankle_Roll	lFoot	lShin	End Site13
L_Elbow_Yaw	lForeArm	lShldr	lHand

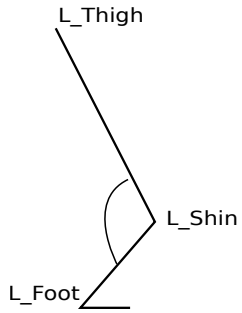

Figura 3.6: Representación del ángulo de la rodilla izquierda a partir de la tripla de marcadores correspondiente.

Si definimos al vector $\vec{v}$ formado por los puntos L_Foot y L_Shin y el vector $\vec{u}$ formado por los puntos L_Thigh y L_Shin, el ángulo entre ambos vectores visto desde el plano sagital en sentido anti-horario determina el valor angular para el motor asociado a L_Knee_Pitch, ver Figura 3.6.

El la Tabla 3.1, dado los marcadores: m1, m2, m3, los valores angulares el resto de las *articulaciones* son calculadas mediante el ángulo descripto por los vectores determinados de m3 a m1 y de m2 a m1 sobre uno de los planos posturales en el correspondiente sentido, como se ve en la Tabla 3.2.

Hubieron valores calculados de *articulaciones* que finalmente no se mapearon al robot por los siguientes motivos:

- los valores angulares alcanzados eran bruscos y presentaban más ruido que el resto de los marcadores, provocando movimientos que no podían ser ejecutados en el robot, llevando a inestabilidades que impedían al algoritmo evolutivo explotar soluciones en tiempos razonables; se tomó como criterio mapear estas la *articulaciones* al valor de reposo, definido por el modelo utilizado en el simulador, y dejar que el algoritmo evolutivo calcule los valores a tomar. Este problema ocurre con el tobillo, otros autores como [20] tomaron igual decisión, también debido a que no podían ser ejecutados por el robot;
- los marcadores utilizados por el equipo de la CMU para registrar movimientos fueron colocados en posiciones que en algunos casos no permitían obtener tres marcadores que describieran el ángulo buscado en el robot utilizado, en este caso también se tuvo en cuenta el criterio de mapear el valor de reposo y dejar que el algoritmo evolutivo calcule el valor a tomar.

El conjunto de ángulos que no fueron mapeados desde la imitación debido

a los problemas descriptos, pero si controlados por el genético a partir de la posición de reposo de la *articulación*, es el siguiente:

L_Elbow_Yaw, R_Elbow_Yaw, L_Shoulder_Yaw, R_Shoulder_Yaw, R_Hip_Yaw, L_Hip_Yaw, L_Ankle_Roll, R_Ankle_Roll, L_Hip_Roll, R_Hip_Roll, L_Ankle_Pitch, R_Ankle_Pitch.

Tabla 3.2: Sentido y plano utilizado para el cálculo del valor angular de cada articulación del robot.

articulación	**sentido**	**plano**
R_Shoulder_Yaw	antihorario	transversal
R_Shoulder_Pitch	horario	sagital
R_Hip_Yaw	antihorario	transversal
R_Hip_Roll	antihorario	frontal
R_Hip_Pitch	antihorario	sagital
R_Knee	antihorario	sagital
R_Ankle_Pitch	antihorario	sagital
R_Ankle_Roll	antihorario	frontal
R_Elbow_Yaw	antihorario	sagital
L_Shoulder_Yaw	antihorario	frontal
L_Shoulder_Pitch	horario	sagital
L_Hip_Yaw	antihorario	transversal
L_Hip_Roll	antihorario	frontal
L_Hip_Pitch	antihorario	sagital
L_Knee	antihorario	sagital
L_Ankle_Pitch	antihorario	sagital
L_Ankle_Roll	antihorario	frontal
L_Elbow_Yaw	antihorario	sagital

Para calcular la magnitud de los ángulos se hace uso de propiedades de los números complejos con el objetivo de lograr optimizar los cálculos. Podemos representar el ángulo mediante la proyección de los marcadores sobre el plano de trabajo complejo. Dados dos puntos en el plano complejo, llamemos w y z a dichos puntos. Se cumple la propiedad: $arg(w * z) = arg(w) + arg(z)$, siendo $arg()$ la función ARGUMENTO de los números complejos. Debido a que el argumento de un número complejo w, $arg(w)$ corresponde con el ángulo formado por el vector que pasa por el origen de los ejes complejos y w, respecto al eje de los reales y que el conjugado de un número tiene igual módulo pero argumento opuesto, $arg(a) = -arg(\bar{a})$. Se cumple que:

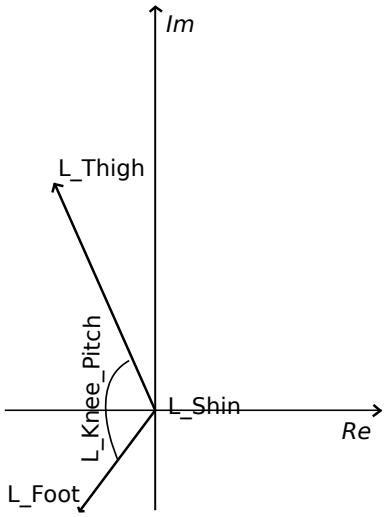

Figura 3.7: Representación del ángulo de la rodilla izquierda en el plano complejo.

$arg(L\_Foot \times \overline{L\_Thigh}) = arg(L\_Foot) - arg(L\_Thigh) = L\_Knee\_Pitch$, como muestra la Figura 3.7.

3.3. ¿Cómo el robot evalúa sus acciones, corrige errores y reconoce cuando ha logrado su objetivo?

Para cada solución generada, el algoritmo evolutivo evalúa aspectos cualitativos del desempeño de dicha solución en el robot simulado utilizando la función de *fitness* definida, la cual fue desarrollada tomando en cuenta las métricas acordes para el problema a resolver, para el caso del problema guía son tenidas en cuenta las métricas definidas en la Sección 2.3.2, y su implementación puede consultarse en la Sección 3.4.1.

Es el mismo procedimiento evolutivo del algoritmo propuesto que a medida que pasan las generaciones va a corregir los errores, respecto a la correspondencia, descartando las soluciones que obtienen un valor bajo de *fitness* y explorando nuevas mediante el operador de mutación.

El algoritmo evolutivo propuesto, maneja un criterio de parada, el cual puede activarse utilizando el valor 1 en el atributo *Convergence_criteria_enable*?, ver Apéndice 1; por lo que si trascurren más de 15 generaciones sin que haya una mejora en las soluciones el algoritmo finaliza.

3.4. Formulación del problema

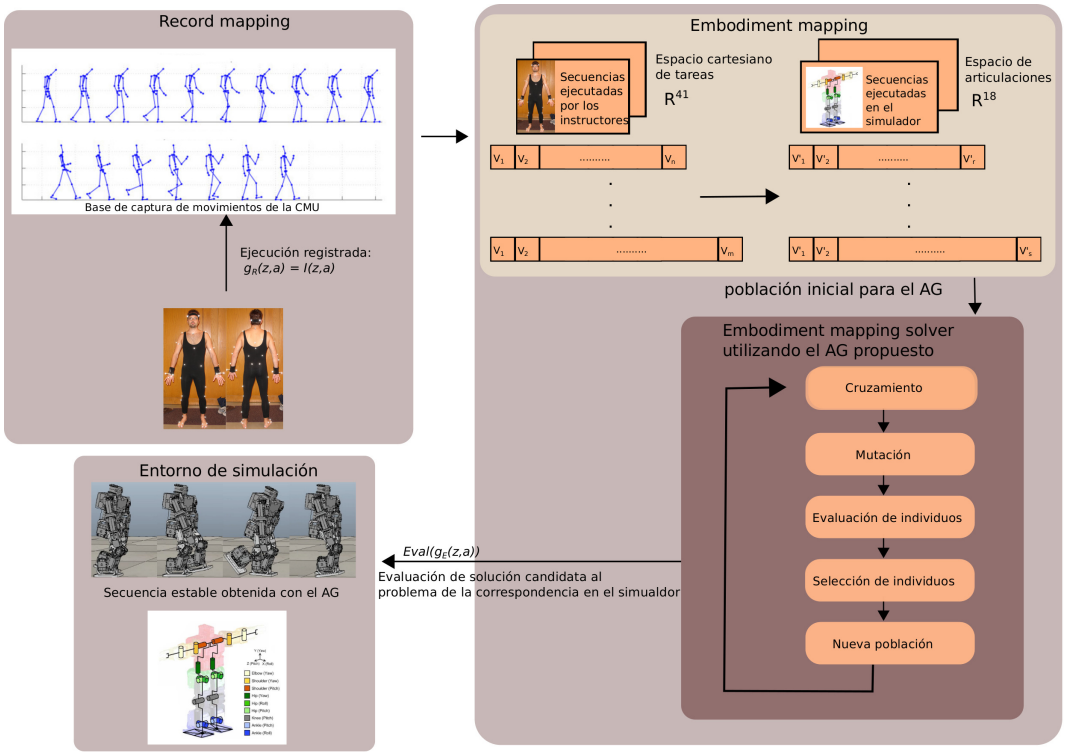

Figura 3.8: Solución propuesta al mapeo entre el instructor y el aprendiz.

Como fue señalado en la Sección 2.1.2 un conjunto de datos de PBD está compuesto por los pares de estado-acción registrados durante la ejecución por un grupo de instructores de la tarea motora a ser aprendida. La función de *Record Mapping* es equivalente en nuestra propuesta a la función identidad, como puede verse en la Figura 3.8, para este caso debido a la elección de utilizar sensores en el maestro como técnica para registrar los movimientos del instructor.

En esta implementación, los elementos del conjunto de *estados* toman valores enteros; existiendo un estado por cada medida de una posición postural del instructor, registrado a una tasa de muestreo de 120 Hz. Vamos a llamar a este conjunto T; su tamaño va a depender de la ejecución de la tarea motora.

Las *acciones* ejecutadas por el instructor corresponden a las posiciones capturadas por un conjunto de 41 marcadores que describen la posición postural del mismo.

Llamaremos $v_t \in R^{41}$ al vector cuyos elementos representan la posición de los marcadores en el espacio cartesiano en un instante dado de tiempo $t \in T$,

ver parte superior izquierda del bloque *Embodiment Mapping* de la Figura 3.8; en dicha figura se puede ver un conjunto de tiras de vectores, cada una representa una demostración realizada por un instructor, por lo que son de largo variable.

Los vectores v'_t son una representación del correspondiente v_t utilizando una dimensión acorde con los grados de libertad del robot a utilizar y con una frecuencia de muestreo menor, como se explicará más adelante en esta sección.

La función de *Record Mapping* puede ser descripta como: $g_R(z, a) \equiv g_R(t, v_t) \equiv I(t, v_t)$ para $t \in T$ y $v_t \in R^{41}$.

La función de *Embodiment Mapping* describe cómo los pares de estado-acción registrados se mapean finalmente en el aprendiz, ver Figura 3.8. Los estados y las acciones se representan de la misma manera que en el caso de la función de *Record Mapping*, una posición postural se describirá por el valor angular de 18 articulaciones presentes en el aprendiz.

Dado un estado temporal $t \in T$ y la acción asociada registrada del instructor v_t, la acción correspondiente para el aprendiz se calcula mediante g_E y es representada por el vector postural $v'_t \in R^{18}$, donde cada elemento de v'_t corresponde al valor angular de la *articulación* j para el instante de tiempo t para la tarea motora realizada. Para simplificar el problema, dividimos entre cuatro la frecuencia de registro de las acciones, por lo que el nuevo conjunto de las acciones T' y sus acciones, v'_t, correspondientes, se reduce a un cuarto del tamaño original.

La función de *Embodiment Mapping* puede ser descripta como: $g_E(t, v_t) = v'_t$ para $t \in T', v_t \in R^{41}$ y $v'_t \in R^{18}$; Los valores t y v_t deben coincidir con los registrados por g_R.

Una grabación de la ejecución de una tarea motora por el aprendiz se puede representar utilizando una serie temporal X, cada valor x_t de la serie corresponde a una instancia de vector postural $v'_t \in R^{18}$ para el instante de tiempo t. La función *Embodiment Mapping* se puede expresar usando la serie temporal definida como entrada:

$$g_E(X) \equiv g_E(t, x_t) \forall t \in T. \tag{3.1}$$

El aprendizaje por imitación se puede considerar como un método para reducir el espacio de búsqueda de un problema de optimización, ya que las ejecuciones realizadas por el instructor se acercan a las soluciones óptimas lo-

cales para el robot aprendiz. En este trabajo, proponemos una implementación genérica para la función g_E, basada en técnicas de optimización, que se puede utilizar en el aprendizaje de cualquier tarea motora. La implementación aprovecha la cercanía entre los estados y las acciones registradas desde el instructor y las que, si se asignan al aprendiz, describen la tarea del motor considerada.

La solución buscada para la función g_E se puede representar como una serie temporal Y que si se asigna al aprendiz describe la misma tarea motora. Si consideramos la serie temporal X construida usando las acciones y los estados registrados por la función g_R pero teniendo en cuenta solo las articulaciones que están presentes en el aprendiz; luego se describe un mapeo entre X e Y con

$$g_E(X) = Y \equiv y_t \forall t \in T \tag{3.2}$$

empleando la notación presentada de g_E en la Expresión 3.1.

El problema se puede reducir a: dada la serie temporal X, encontrar Y, otra serie del mismo tamaño, que pueda ejecutar en el aprendiz la misma tarea motora considerada, representado en la Expresión 3.2.

No conocemos la representación analítica de g_E ni el valor óptimo de Y, pero buscamos un óptimo local Y', donde la postura $y'_t \in Y'$ está cerca de $x'_t \in X' \forall t \in T$, por lo que podemos aplicar un método de optimización para encontrar Y, comenzando desde X y usando una métrica para la evaluación de la series temporales intermedias Y' calculadas. Si el instructor, o un grupo de ellos, proporciona más representaciones de la tarea motora, se dispone de más información utilizable para el método de optimización, resultando más simple encontrar cómo asignar a X una serie de tiempo Y que describe una posible solución para el problema de aprendizaje de tareas motoras.

3.4.1. El algoritmo genético implementando

Un algoritmo genético (AG) es formulado para la implementación del método de optimización descripto en la Sección 3.4. Operadores específicos de mutación y cruzamiento son definidos junto con la formulación de la representación de soluciones y la población inicial para sacar provecho de las características de este problema.

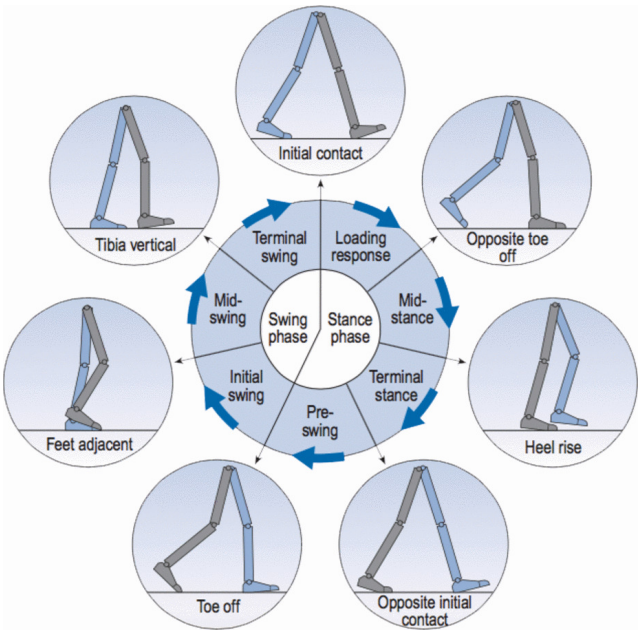

Figura 3.9: Ciclos de marcha, imagen extraída de [30].

Codificación de los individuos

Las soluciones del AG son representadas por una serie temporal de tamaño variable que simboliza a un cromosoma, para cada valor j de la serie un vector de R^n representa la pose del aprendiz en el instante de tiempo j y es equivalente a un gen de la representación cromosómica, cada valor angular de la pose representa un alelo en dicha representación. Consideramos el ciclo de marcha como el movimiento a ser codificado, dado que simboliza una unidad mínima que representa a una caminata en régimen y es una codificación estándar en el área de biomecánica, ver Figura 3.9.

En trabajos anteriores hemos recurrido a PCA como un método para la reducción dimensional de la codificación de individuos [55]. Las técnicas de reducción dimensional pueden simplificar la complejidad computacional del problema, pero, algunas características interesantes de las señales también se pueden perder durante el proceso. PCA podría fallar al extraer los pequeños movimientos circulares, ya que PCA conserva las dimensiones que presentan una gran variabilidad [14]. Debido a la baja variabilidad de algunas articulaciones, como el tobillo, que sin embargo tienen un gran impacto en el rendimiento de estabilidad del robot, optamos por no utilizar la reducción dimensional.

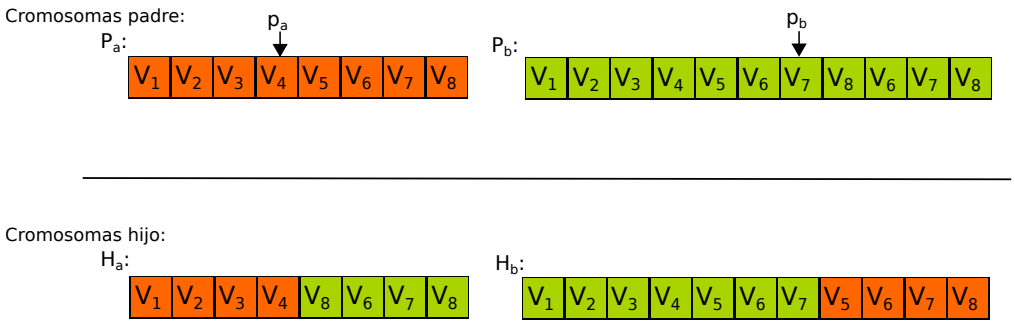

Figura 3.10: Ejemplo de aplicación del operador de cruzamiento propuesto; dados los individuos P_a y P_b, se obtienen como resultado de la aplicación del operador de cruzamiento los individuos H_a y H_b. Los genes p_a y p_b de los respectivos padres P_a y P_b corresponden con los seleccionados para realizar el cruzamiento.

Población inicial

Normalmente, la población inicial se genera aleatoriamente, pero como se describió en la Sección 3.4, queremos aprovechar la cercanía de las soluciones, que representan a la tarea motora para el aprendiz, con respecto a la ejecución del instructor. Un conjunto de individuos, que representa ejecuciones de diferentes maestros, se usa para crear la población inicial con el fin de generar diversidad que permita al algoritmo explotar las soluciones construidas.

Operador de cruzamiento

Se implementa una variante del cruzamiento en un punto, ver Figura 3.10, dado dos padres denominados P_a y P_b, se selecciona un gen aleatorio p_a en P_a. En lugar de seleccionar un valor aleatorio en P_b, para hacer el cruzamiento se selecciona un gen p_b donde la distancia euclidiana, según la Ecuación 3.3 es mínima; p_b intenta ser la postura de P_b que se asemeja más a p_a. Este enfoque se utiliza para mantener la coherencia y continuidad en las soluciones construidas. Para garantizar una transición suave entre las poses, se aplica una *spline* de interpolación cúbica para recalcular los valores angulares de las poses en el intervalo $(p_a - n, p_a)$ y $(p_a, p_a + n)$ para el hijo H_a y $(p_b - n, p_b)$ y $(p_b, p_b + n)$ para el hijo H_b, utilizando el spline que aproxima a los puntos del intervalo $[p_a - n\ ,\ p_a + n]$ y $[p_b - n,\ p_b + n]$.

Si no cumple un valor de similitud mínimo ($d(a, b) < \epsilon$, con ϵ calculado empíricamente) no se realiza la cruza, y se retorna como hijos a los mismos padres. El objetivo es evitar que el cruzamiento rompa con la coherencia del ciclo de marcha.

El operador de cruzamiento definido sigue un enfoque de hibridación [28], ya que a nivel de la representación cromosómica se realiza una búsqueda local, tratando de explotar localmente las soluciones, incorporando conocimiento específico sobre las mismas.

El operador propuesto se basa en la propiedad de que es baja la probabilidad de que en una secuencia motora se repita, si ese fuese el caso se debe segmentar y aplicar nuevamente el procedimiento de aprendizaje; por ejemplo, en el caso de los movimientos cíclicos como el del caso de estudio de la marcha, se debe representar el ciclo como la unidad motora a ser aprendida.

$$d(a,b) = \sqrt{\sum_{i=1}^{18}(a[i]-b[i])^2} \tag{3.3}$$

Función de *fitness*

El AG genera nuevas soluciones representadas por la serie temporal Y' que intenta alcanzar un óptimo local, o al menos una solución factible, Y; cada nueva solución se evalúa con una función de *fitness* que mide si la ejecución de la serie temporal construida representa la tarea de motora a implementar.

El problema guía de la caminata, es intrínsecamente multiobjetivo, debido a la necesidad de alcanzar una solución de compromiso entre varios objetivos conflictivos, como la velocidad y la estabilidad; como veremos en el Capítulo 4 para este problema se utilizaron cinco objetivos, la función de *fitness* es calculada como una suma ponderada de los mismos.

Operador de mutación

El objetivo del operador de mutación en los AG es guiar la búsqueda manteniendo alta la diversidad, con ello en mente un operador gaussiano clásico es utilizado con el anhelo de explorar soluciones cercanas, en términos de la distancia euclidea de sus genes, según la Expresión 3.3; los nuevos genes son calculados mediante el siguiente procedimiento: dado un gen, el cual corresponde a una pose, se seleccionan aleatoriamente, según la probabilidad de mutación definida, algunas *articulaciones* j_i para las que se calcula su nuevo valor mediante una distribución normal con media el valor de j_i y una desviación estándar $\theta = 3$, de esta manera la construcción de nuevas poses trata de mantener la distancia respecto a las ejecuciones de los instructores utilizados

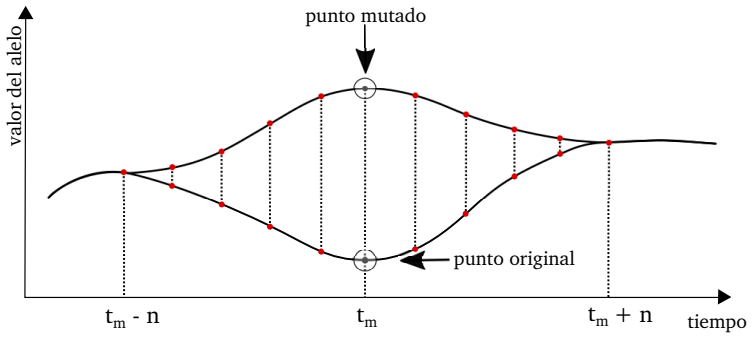

Figura 3.11: Descripción del procedimiento para el recalculo de los valores angulares, mediante *splines* cúbicos, utilizando una ventana de tamaño l.

como población inicial del algoritmo. Nuevos puntos, centrados en el gen mutado seleccionado y, en el peor de los casos, a una distancia de tres unidades del valor original del alelo son obtenidos para las *articulaciones* seleccionadas del gen. Dado que se utiliza una distribución normal, es más probable que los nuevos valores se encuentren cercanos al valor de j_i y menos probable los valores que se aproximan a los extremos del intervalo (j_i, θ).

Una dificultad que presenta el problema seleccionado como guía es que un pequeño cambio en el valor de un gen (una pose) puede generar un comportamiento inestable en el robot, provocando su caída y por consiguiente se obtiene un *fitness* muy bajo para esa solución, al no haber sido posible evaluar el resto de las poses del individuo. Este problema afecta a la búsqueda de soluciones por lo que se propone el siguiente procedimiento para mitigarlo: después de realizar la mutación gaussiana, se aplica una interpolación utilizando *splines* cúbicos para corregir los puntos cercanos al valor mutado, aportando una transición más suave, contribuyendo a que el movimiento resultante no presente saltos [55] tratando de reducir de esta manera al conjunto de las poses inestables generadas mediante mutación. En la Figura 3.11 se presenta un ejemplo al aplicar dicho procedimiento a nivel del alelo, el valor del mismo para el tiempo t_m es mutado, una spline cúbica que pasa por los puntos $t_m - n$, t_m y $t_m + n$ es calculada, luego los puntos en el intervalo $(t_m - n, t_m)$ y $(t_m, t_m + n)$ se vuelven a calcular utilizando el valor calculado por la spline.

Operador de selección

Este componente es responsable de seleccionar a los individuos para ser cruzados y los que prevalecerán en la próxima generación. Para la selección, se

aplica un enfoque de "selección de torneo", seleccionando a los individuos que participan en dicho torneo según una probabilidad de selección proporcional al *fitness* respecto los individuos que participarán en el torneo. Si f_i es el *fitness* del individuo i en la población, su probabilidad de ser seleccionado se rige por la Ecuación 3.4.

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \tag{3.4}$$

Una vez que se completa el cruce, se calcula la siguiente generación de individuos, utilizando el criterio elitista de inyectar el mejor 30 % de los individuos de la población actual en el conjunto de nuevos individuos obtenidos mediante el cruzamiento. Este nuevo conjunto se clasifica utilizando el *fitness* obtenido como criterio de ordenamiento y el subconjunto de los primeros n individuos, con n igual al tamaño de población utilizado, donde se seleccionan los mejores individuos para la próxima generación.

Capítulo 4

Evaluación experimental

Con el objetivo general, de validar la propuesta de imitación desarrollada en el Capítulo 3, es que en este capítulo se presentan los resultados obtenidos al aplicar el algoritmo propuesto para el aprendizaje de la tarea motora correspondiente a la caminata en línea recta, partiendo de las demostraciones ejecutadas por instructores humanos. Debido a que la complejidad de este problema es de renombre en el área de la robótica humanoide, se decide utilizarlo a modo de validación de la propuesta.

Como vimos en el Capítulo 3, la metodología propuesta para alcanzar una solución en nuestra plataforma, consiste en realizar una primera, pero importante, aproximación a la solución mediante la aplicación del algoritmo evolutivo en la plataforma simulada, utilizando como entrada los datos registrados a partir de la ejecución de un humano; como segunda aproximación, en este caso final, se utilizan como entrada los resultados obtenidos en la etapa de simulación, aplicando el mismo algoritmo evolutivo directamente sobre la plataforma física. Mediante la aplicación de estas dos etapas es que se obtiene una solución final al problema de la correspondencia, ver Figura 3.1. En las secciones 4.5 y 4.6 se presentan, respectivamente, los resultados obtenidos para la plataforma virtual y la real.

4.1. Plataforma robótica seleccionada

La plataforma robótica utilizada fue la Bioloid Expert Kit de Robotis, debido a ser la única plataforma robótica bípeda disponible en el laboratorio de Inteligencia Artificial, Robótica y Sistemas Embebidos del Instituto de

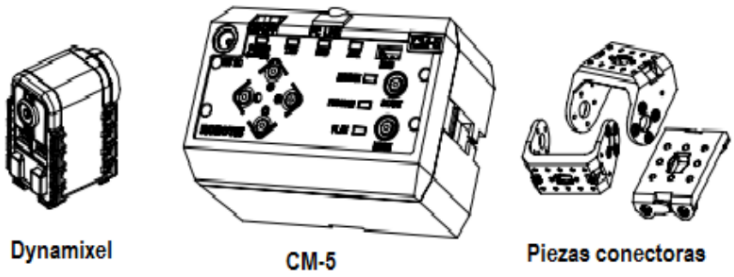

Figura 4.1: Componentes de *hardware* del Bioloid Expert Kit; imagen extraída de [54].

Computación [1], lugar donde se llevaron adelante los experimentos.

El kit está compuesto de tres tipos de componentes de hardware [54], ver Figura 4.1:

- Dynamixel: Es la unidad básica de Bioloid que actúa como una articulación o como sensor. Los servos Dynamixel AX-12 son actuadores que se utilizan como articulaciones. A su vez los sensores AX-S1 Dynamixel son unidades que sensan tanto distancia como sonido.
- CM-5: Este es el controlador principal del robot Bioloid. Las baterías que se encuentran dentro del CM-5 le proveen energía a los Dynamixel conectados.
- Piezas conectoras: Conectan a las unidades del robot. Los Dynamixel pueden ser conectados juntos con el uso de una de estas piezas. También conectan al Dynamixel con el CM-5.

La versión humanoide del Expert Kit consta de 18 servos Dynamixel AX-12 que implementan las articulaciones del robot, 1 Dynamixel AX-S1 para sensado de sonido y distancia, montado pero no utilizado en la propuesta, el CM-5 y las piezas conectoras correspondientes para ensamblar el robot, en la Figura 4.2 se puede ver una imagen del robot armado.

4.2. Acondicionamiento de datos

La unidad motora a ser aprendida corresponde con un ciclo de marcha, dado que representa toda la información necesaria para generar la caminata; mediante su concatenación se generan caminatas del largo deseado, la utilización de dicho ciclo es un estándar para el estudio de la locomoción bípeda en

[1] de la Universidad de la República

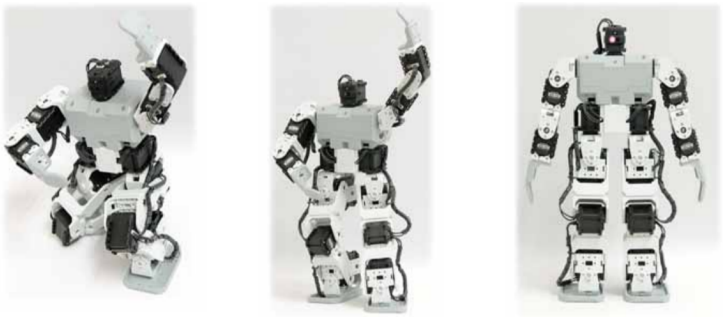

Figura 4.2: Robot Bioloid Expert humanoide ensamblado; imagen extraída de [53].

el área de la Biomecánica, en la Figura 3.9 se pueden apreciar las diferentes poses características de un ciclo de marcha.

El inicio del ciclo de marcha debe corresponder con la pose *initial contact* mostrada en la Figura 3.9, dado que esa configuración asume que el robot ya logró dar, al menos, un paso previo y no comienza del reposo, la codificación en el algoritmo genético de las soluciones debe ser equivalente, por lo que se requiere de un movimiento previo que lleve al robot desde la posición estable de reposo, con los dos pies juntos, a la posición de inicio del ciclo, a este movimiento lo llamaremos pre-ciclo; el mismo no participa de la codificación del algoritmo genético, por lo que no es afectado por los operadores evolutivos, sólo se utiliza para poner en régimen la caminata en el robot.

Para obtener el pre-ciclo se utiliza previamente el mismo algoritmo evolutivo, tomando como datos de entrada ejecuciones de la base de movimientos de la CMU que corresponden a dicho movimiento (comienza con los dos pies juntos y da un paso para culminar según la pose *initial contact*, mostrada en la Figura 3.9).

Antes de ser evaluadas las soluciones del AG en el simulador se le concatena la tira genética correspondiente con el pre-ciclo

4.3. Decisiones tecnológicas realizadas

Se trabaja sobre la plataforma robótica Bioloid Expert, con una configuración de *articulaciones* como la de la Figura 3.5, como plataforma virtual se utiliza el simulador V-REP en su versión 3.3.0 para 64 bits utilizando la API remota y un modelo externo del robot seleccionado, debido a que el mismo no es soportado nativamente por V-REP, resultado del trabajo de Al-Bahrani y colaboradores [51].

Como lenguaje de programación, para la implementación del algoritmo genético propuesto, se utiliza Python, el cual fue seleccionado debido a que posee un FOOTPRINT suficientemente pequeño como para ejecutar el algoritmo en plataformas de bajo recursos de hardware, las que se pueden embeber directamente sobre el robot. Además Python es muy poderoso en el manejo de estructuras de datos del tipo lista, las cuales son utilizadas para representar las soluciones.

Se utiliza la biblioteca evolutiva *pyevolve* en su versión 0.6rc1 para implementar el algoritmo genético propuesto; al momento de la decisión existían pocas bibliotecas disponibles, y *pyevolve* cumple con el requisito de permitir al usuario redefinir los operadores genéticos, por lo cual fue seleccionada.

4.4. Métricas empleadas

Para evaluar el desempeño de las soluciones obtenidas para el problema de la marcha bípeda, se definen las métricas a utilizar en esta etapa de experimentación. Las mismas están fundadas en el relevamiento realizado en la Sección 2.3.2.

La similitud con la marcha humana es una métrica importante, siendo uno de los objetivos de la propuesta y ha sido tenida en cuenta al diseñar la propuesta, pero debido a la dificultad para evaluar si las soluciones obtenidas cumplen con dicha métrica, su aplicación es dejada para trabajo futuro para la comparación con otras técnicas.

Como métrica se utilizarán dos que son ampliamente empleadas, la distancia recorrida por el robot y la cantidad de pasos ejecutados, hasta caerse o terminar la marcha.

4.5. Experimentación sobre plataforma simulada

En esta sección se presentan los experimentos realizados sobre la plataforma simulada, donde se obtiene una solución al problema de la correspondencia para la plataforma virtual y un acercamiento para la solución a dicho problema en la plataforma real.

El experimento consiste en enseñarle al robot la tarea motora de la marcha

a partir de un conjunto de diferentes demostraciones realizadas por distintos instructores. Para esto se propone resolver el problema de la correspondencia para un ciclo de marcha y luego mediante concatenación evaluar caminatas de largo variable.

Los individuos que conforman la población inicial, fueron seleccionados tratando de contemplar diferentes tipos de marcha tales como: triste, erguida, enérgica, entre otros disponibles en la base de MOCAP. Para simplificar la tarea de seleccionar únicamente las ejecuciones, de la base de movimientos de la CMU, que corresponden a caminatas, se automatizó mediante un SCRIPT que selecciona las caminatas, utilizando la METADATA de la base, generando un archivo XML con el primer ciclo de marcha, para cada demostración, para conformar la población inicial del algoritmo.

Para evaluar los resultados obtenidos se utilizará como métrica la función de *fitness* a ser presentada en la Sección 4.5.2, la cual aportará una medida de la calidad del ciclo de marcha generado por el AG. El procedimiento para evaluar un ciclo de marcha es el siguiente:

- se concatena una cantidad *gait_cycle_concatenation* de veces el ciclo de marcha a evaluar consigo mismo, con el objetivo de medir el ciclo de marcha calculado en una condición de régimen;
- se concatena el pre-ciclo por izquierda, para asegurar las condiciones esperadas por el primer ciclo.

Para los experimentos descriptos en este capítulo se utilizó un valor de $gait\_cycle\_concatenation = 4$, dado que permite probar en régimen la plataforma. El valor máximo obtenido para la plataforma virtual corresponde con un valor de $gait\_cycle\_concatenation = 5$, para valores mayores el robot no logra terminar el experimento sin caerse.

Los datos de la base de movimientos fueron registrados a una frecuencia de 120 Hz, dado que el entorno de simulación fue configurado a un paso de simulación de $50ms$, el sistema debe descartar 5 poses por cada una que toma, con el objetivo acompasar la cadencia de poses con la cadencia de ejecución del simulador. Esta propiedad es una de las expuestas por el framework de imitación propuesto y pueden consultarse en el Apéndice 1

Los resultados obtenidos son comparados contra una *línea base* correspondiente a aplicar el mismo algoritmo genético, pero sin tener en cuenta la com-

ponente de imitación y en lugar de partir de una población inicial conformada por las ejecuciones de un humano, se parte de una población aleatoria.

4.5.1. Tiempo de ejecución

Las pruebas fueron realizadas en un PC tipo Notebook con procesador Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz de 4 núcleos ejecutando un sistema operativo Ubuntu 15.04 de 64 bits en un disco duro SATA con interfaz de 3Gb/s de 5400 RPM. Los experimentos realizados con una población de tamaño 60 individuos tardan 8 horas en completar, se puede reducir este tiempo activando el criterio de convergencia, donde se pueden obtener resultados factibles en el entorno de las 3 horas, para un criterio de parada de 15 generaciones sin cambio del *fittest* (individuo con mejor valor de *fitness*).

4.5.2. Configuración del algoritmo

La plataforma propuesta es lo suficientemente genérica para utilizarse con cualquier tipo de movimiento; de todas maneras existen algunos componentes cuya especificación está muy ligada al tipo de movimiento a ser aprendido, por lo que necesitan un refinamiento antes de utilizar la plataforma para el aprendizaje de los mismos, a continuación entraremos en detalle en ellos.

Especificación de la función de *fitness*

Como fue mencionado en la Sección 3.4.1 el problema de la caminata bípeda es un problema multi-objetivo. En aras de la simplicidad, se reduce a uno mono-objetivo utilizando una suma ponderada de los mismos para la implementación de la función de *fitness*; la determinación de la frontera de Pareto es propuesta para su tratamiento en trabajos futuros.

La función de *fitness* es diseñada teniendo en cuenta por un lado, objetivos que aseguran se respete una construcción de soluciones coherente para el problema, y por otro objetivos basados en las métricas definidas en la Sección 4.4.

Para cada objetivo la variable *objetive* puede alcanzar un valor en el intervalo real [0..1], donde 1 corresponde al valor máximo que un objetivo particular podría alcanzar. Al multiplicar cada *objetive* por el correspondiente $objetive_{weight}$, definido en la Tabla 4.1, obtenemos el valor $objective_{weighted}$ pa-

ra usar en el cálculo de la función de *fitness*. Los objetivos más importantes tienen un valor $objetive_{weight}$ cercano a 1, la suma de todos los pesos debe ser 1 para garantizar que el valor máximo que puede alcanzar la función de *fitness* también sea 1.

Los objetivos utilizados para el cálculo de la función de *fitness* para este problema son:

- *distance*: corresponde a la distancia recorrida por el robot antes de caer o finalizar la ejecución, normalizada con respecto a la distancia recorrida por ciclo de caminata, empleando como distancia de referencia la implementación de Baliero y Pias [46], donde se utilizan técnicas de stop-motion con el mismo robot;
- *concGap*: corresponde a la medida de la distancia euclidiana entre el último cuadro de un ciclo de marcha y el primero, ya que la marcha es un movimiento periódico, es deseable que el final del ciclo se asemeje tanto como sea posible con el inicio, el valor 1 representa el par exacto de cuadros;
- $framesExec$: representa la cantidad de posturas ejecutadas, respecto del total, antes de que el robot se caiga o complete la tarea, el valor 1 representa la ejecución de todas las poses;
- *endCycleBalance*: medida de si el robot termina la ejecución con el torso erguido (minimizando la desviación del tronco de la postura erguida), el valor 1 corresponde a finalizar con la cabeza a la distancia del piso correspondiente a la altura del robot;
- *angle*: representa un valor correspondiente con el ángulo de desviación de la marcha del robot, un ciclo de caminata recta obtiene el valor 1.

La suma ponderada utilizada para el cálculo del *fitness* para el problema propuesto es:

$$\begin{aligned} fitness = distance_{weighted} + concGap_{weighted} + framesExec_{weighted} \\ + endCycleBalance_{weighted} + angle_{weighted} \end{aligned} \tag{4.1}$$

Cada individuo, generado por el algoritmo genético, se ejecuta en el entorno de simulación donde se obtiene la información necesaria para el cálculo de cada *objetive* y se asigna un valor de *fitness*.

Tabla 4.1: Pesos, calculados de forma empírica, utilizados para cada objetivo de la función de *fitness* propuesta.

$distance_{weight}$	0.10
$concGap_{weight}$	0.20
$framesExec_{weight}$	0.30
$endCycle_{weight}$	0.35
$angle_{weight}$	0.05

Para cada habilidad motora que se desee aprender se requiere la definición de una función específica de *fitness* en la arquitectura propuesta.

Especificación de parámetros

La calibración de estos parámetros fue realizada en dos etapas, utilizando un enfoque inspirado en el trabajo de [55]: la primera, llamada etapa de aproximación, en donde se calculan valores groseros de los parámetros, obteniendo una aproximación de los mejores valores; y la segunda etapa, llamada la etapa de refinamiento, donde en base al cálculo del valor de los parámetros de la etapa anterior se refinan para obtener los valores óptimos para el algoritmo genético. En cada etapa y para cada configuración de los parámetros de esa etapa (tamaño de la población inicial, taza de cruzamiento y tasa de mutación), se realizan cinco ejecuciones del algoritmo genético teniendo en cuenta esos parámetros. El *fitness* medio de los individuos de la población final se utiliza para calcular el rendimiento del algoritmo usando estos parámetros.

El criterio utilizado para seleccionar los mejores valores de parámetros tanto en las etapas de aproximación como de refinamiento fue el de *significación estadística*, que se define de la siguiente manera: dados los algoritmos A y B, determinados por el tamaño de la población inicial, tasa de cruzamiento y tasa de mutación, el algoritmo A es mejor que el algoritmo B si el fitness promedio de A es mayor que el de B, ver Ecuación 4.2; y si la diferencia promedio en el *fitness*, tomado para el mejor tercio de la población final, es mayor que la mayor desviación estándar de dicha subpoblación, como podemos observar en la Ecuación 4.3.

Para esta calibración se utiliza una población inicial diferente a la que se utiliza luego en la etapa de experimentación.

$$f_{\text{avg}}(A) > f_{\text{avg}}(B) \tag{4.2}$$

Tabla 4.2: Aproximación de parámetros

	Mínimo	Máximo	Incremento
mutation_rate	0.3	0.5	0.1
crossover_rate	80	100	10
population_size	20	40	10

Tabla 4.3: Refinamiento de parámetros

	Mínimo	Máximo	Incremento
mutation_rate	0.45	0.55	0.05
crossover_rate	90	100	5
population_size	25	35	5

$$|f_{\text{avg}}(A_{\text{best}}) - f_{\text{avg}}(B_{\text{best}})| > \text{máx}\left(\text{stdev}(A_{\text{best}}), \text{stdev}(B_{\text{best}})\right) \tag{4.3}$$

Cada algoritmo es comparado con los demás y clasificado; el algoritmo ganador se encuentra en una clase para la cual no existen mejores algoritmos luego de realizar las comparaciones. El parámetro correspondiente al número de generaciones, $number\_of\_generations$, es tomado en un valor suficientemente grande para estudiar la evolución del *fitness* para el ejemplo presentado.

Luego de la aproximación y el refinamiento, ver Tabla 4.2 y 4.3, los parámetros del algoritmo genético utilizado fueron: $mutation\_rate = 0.5$ (50 %), $crossover\_rate = 1$ (100 %), $population\_size = 30$ y $number\_of\_generations = 60$; los mismos fueron experimentalmente determinados con el mecanismo presentado, dejando como trabajo a futuro una calibración de mayor rigor de los mismos con el objetivo de obtener mayor provecho de la solución propuesta.

4.5.3. Comparación con la línea base

La aplicación del algoritmo descrito en la Sección 3.4 para el problema guía, muestra en los resultados que para nuestra propuesta de imitación las soluciones factibles se encuentran antes de la generación 35 en el caso promedio, para un conjunto de 20 experimentos realizados con la misma población inicial

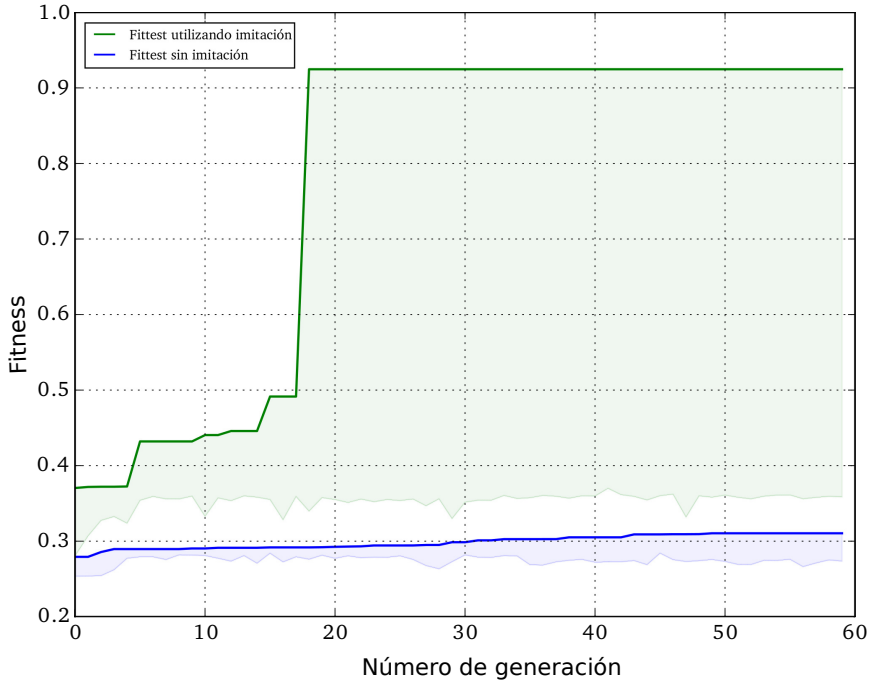

Figura 4.3: Comparación en la evolución del *fitness* a través de las generaciones con y sin el enfoque imitativo. En color azul se puede ver la evolución para el caso sin imitación y en verde el caso de la propuesta. Cada zona está delimitada por el *fitness* máximo y mínimo a través de las generaciones; figura de elaboración propia.

y parámetros calculados, según Sección 4.5.2.

En la Figura 4.4 se muestra un ejemplo representativo, donde el *fitness* máximo de la última generación mejora en un 151 % con respecto al mejor individuo de la población inicial; presentando el primer máximo local en la 5ª generación donde se alcanza la primera solución factible (un ciclo de caminata estable), a medida que pasan las generaciones se obtienen refinamientos del ciclo; en `https://www.fing.edu.uy/~aaguirre/maestria/lucy_evolution.avi` se puede ver la evolución obtenida. El *fitness* promedio mejora en un factor del 85 % en la generación final respecto a la inicial, donde un tercio de las soluciones encontradas son factibles.

Un resultado notable del algoritmo propuesto es que se obtienen mejoras considerable del *fittest* de una generación a la siguiente, presentando un aspecto escalonado, como ocurre en la generación 5, 15 y 18, donde el robot logra aprender: 2, 4 y 8 pasos respectivamente; obteniendo en la última generación mencionada una mejora del 86 % para la transición desde la generación 17 a la

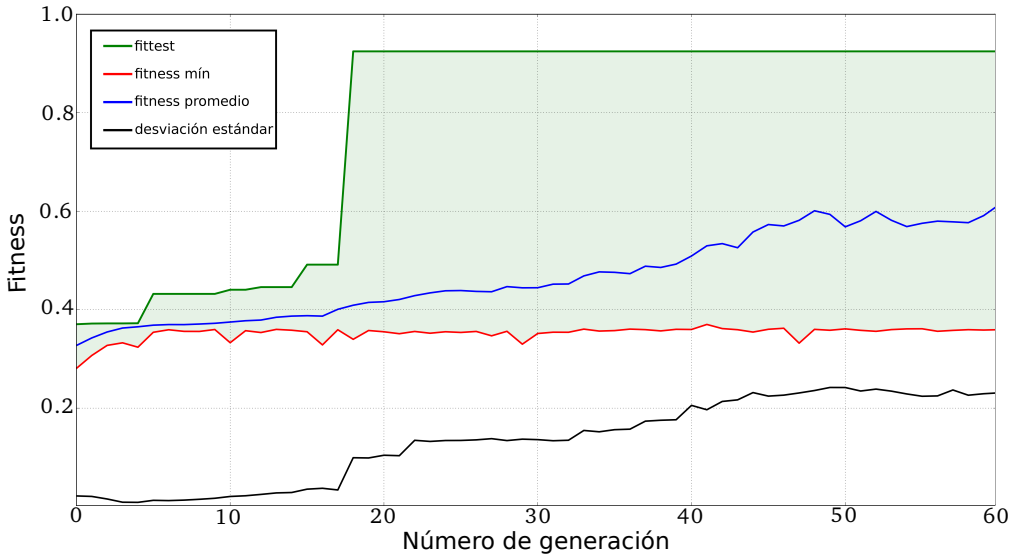

Figura 4.4: Evolución del *fitness* a través de las generaciones para el experimento que utiliza la propuesta de imitación. En color verde se puede observar los valores para el *fittest*, en rojo para el mínimo, en azul el *fitness* promedios y en negro la desviación estándar; figura de elaboración propia.

18 en el ejemplo de la Figura 4.4, este aumento repentino de generación a generación se mantiene en el caso promedio, para un conjunto de 20 experimentos realizados con la misma población inicial y parámetros calculados; esto refuerza la observación hecha en el Capítulo 3.4 sobre la proximidad, en el espacio de búsqueda, de las soluciones factibles entre el instructor y el aprendiz.

Una marcha estable de ocho pasos se obtiene mediante la cuarta concatenación de la solución de ciclo de caminata más apta, que se muestra en la Figura 4.5, como se muestra en este experimento; la marcha completa se puede ver en `http://www.fing.edu.uy/~aaguirre/maestria/ga_fittest.avi`.

Como línea base se utiliza un experimento utilizando la técnica de AG pero sin el enfoque de imitación. La población inicial se toma aleatoria y se utilizan los operadores clásicos de mutación y cruzamiento; se realiza una etapa de aproximación de parámetros como el de la Tabla 4.2 para este nuevo algoritmo. En la etapa de experimentación no se encuentran soluciones factibles luego de ejecutar 20 experimentos. Un ejemplo estadísticamente representativo, de la única clase de soluciones encontrada en los experimentos realizados se presenta en la Figura 4.3, donde la mejora del *fitness* máximo es del 11 % en el enfoque sin imitación. En la Figura 4.3 se comparan las dos propuestas, donde se puede ver que para el enfoque sin imitación no se alcanzan soluciones que mejoren la generación 0 del enfoque con imitación.

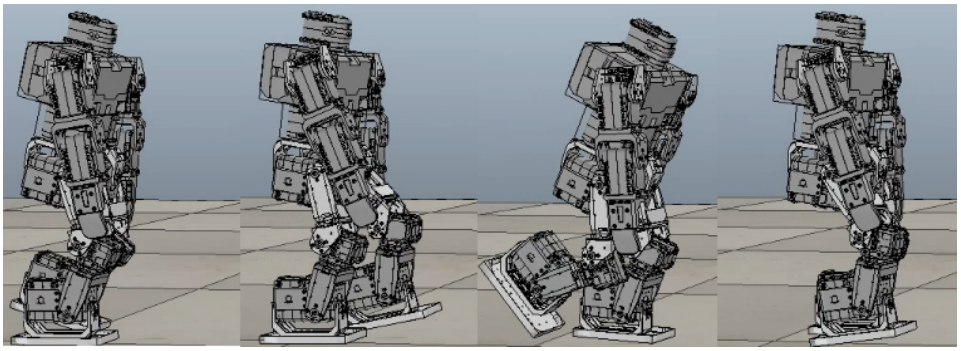

Figura 4.5: Instantáneas obtendas del ciclo de marcha generado en la plataforma virtual; figura de elaboración propia.

4.5.4. Análisis multimaestro

Para estudiar como impacta el uso de un enfoque multi-maestro a la solución se tomó la población inicial utilizada en el experimento de comparación contra la *línea base* presentado en la Sección 4.5. Se dividió dicha población inicial en dos conjuntos de 15 individuos cada uno y se realizaron pruebas utilizando como nueva población inicial el conjunto duplicado, para mantener poblaciones de 30 individuos como fué calibrado el algoritmo. De esta manera se obtiene el conjunto *new_initialpop_part*1 y *new_initialpop_part*2 ambos de 30 individuos.

Luego de 5 ejecuciones con cada nueva población se obtuvo que para el conjunto *new_initialpop_part*1 sólo 1 experimento logró obtener resultados factibles antes de la generación número 30, 2 lo hicieron entre la generación 30 y la 60, y los otros dos no obtuvieron resultado factibles para el problema. Para los 5 experimentos realizados con la población *new_initialpop_part*2 sólo 1 logró obtener una solucción factible para el problema en la generación 58.

Es importante recordar que para los 20 experimentos realizados con la población sin dividir todos obtuvieron soluciones factibles. Estos resultados dan más argumentos para pensar que un enfoque multi-maestro brinda mejores resultados a medida que aumenta la cantidad de ejecuciones diferentes a imitar, gracias a la diversidad que es introducida en el algoritmo. De todas maneras conviene realizar más experimentos para validar esta idea, los cuales son dejados para trabajo futuro.

4.5.5. Limitaciones de la plataforma virtual

La plataforma V-REP presenta algunas limitaciones, entre ellas se encuentra la imposibilidad de simular en la "plataforma virtual" a un paso de simulación más alto, de modo de ejecutar los experimentos a una velocidad mayor, esto hace mucho más lento el proceso de aprendizaje.

Otra limitación al utilizar el simulador es que por defecto viene configurado para ser utilizado en modo asincrónico, esto significa que si no recibe mensajes el entorno virtual se encarga de seguir simulando, eso provoca que si el equipo donde se corre la simulación está muy cargado, el resultado de ejecutar dos veces la misma simulación es diferente. Para evitar este indeterminismo que afecta al mecanismo evolutivo propuesto se configura para utilizar el modo sincrónico, donde se establece una ventana de tiempo donde el entorno simula la física y luego de transcurrido ese tiempo se procesa el siguiente mensaje o se detiene la simulación hasta que un mensaje sea recibido.

4.6. Experimentación sobre plataforma real

En esta sección se presentan los resultados sobre la plataforma real, donde se obtiene una solución al problema del *reality gap* presentado en la Sección 2.2.1, obteniendo, finalmente, una respuesta para el problema de la correspondencia en la plataforma real, ver Figura 3.1.

Para las pruebas sobre la plataforma real se utilizó el *framework* desarrollado, cambiando la propiedad *Lucy_simulated*? a 1. El mapeo de las *articulaciones* a los motores se realizó utilizando el archivo de configuración descripto en la Sección 1.2 del Apéndice 1.

Como población inicial para el algoritmo evolutivo, a ser ejecutado en la plataforma real, se utilizan los mejores seis individuos obtenidos en el experimento realizado sobre la plataforma virtual.

Para evaluar los resultados obtenidos, se utilizará como métrica la función de *fitness* presentada en la Sección 4.6.1, luego de concatenar el ciclo cuatro veces (*gait_cycle_concatenation*) consigo mismo.

Luego de la décima generación, el algoritmo llega al criterio de parada, establecido para la plataforma real en más de tres generaciones sin mejoras en el *fittest*.

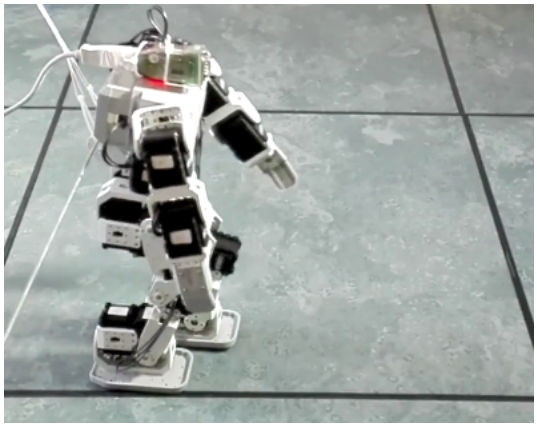

Figura 4.6: Ejecución del algoritmo genético en el robot físico; imágen de elaboración propia.

4.6.1. Configuración del algoritmo

Especificación de la función de *fitness*

El ambiente real, a diferencia del ambiente simulado, no dispone de una plataforma que permita calcular rápidamente atributos sobre el desempeño del robot; por lo que se se simplifica la función de *fitness* para la plataforma real, teniendo en cuenta los lineamientos definidos en la Sección 2.3.2. Para una evaluación rápida y sencilla de la función, sólo se utiliza la cantidad de pasos completados por el robot como métrica.

Se definió la siguiente función de *fitness* para evaluar la aptitud de los individuos ejecutados en el robot físico, no se considera el pre-ciclo:

- **1** si completa 4 ciclos de marcha
- **0.8** si completa 3 ciclos de marcha
- **0.6** si completa 2 y 1/2
- **0.4** si completa 1 y 1/2 ciclos de marcha
- **0.2** si completa 1 ciclo de marcha
- **0** si no completa 1 ciclo de marcha

El criterio de convergencia definido fue de tres generaciones sin mejoras, una vez obtenido el criterio de convergencia se detiene la ejecución del algoritmo.

Especificación de parámetros

El algoritmo genético en el robot físico, se ejecutó con una población inicial de 6 individuos en 30 generaciones; para el resto de los parámetros genéticos

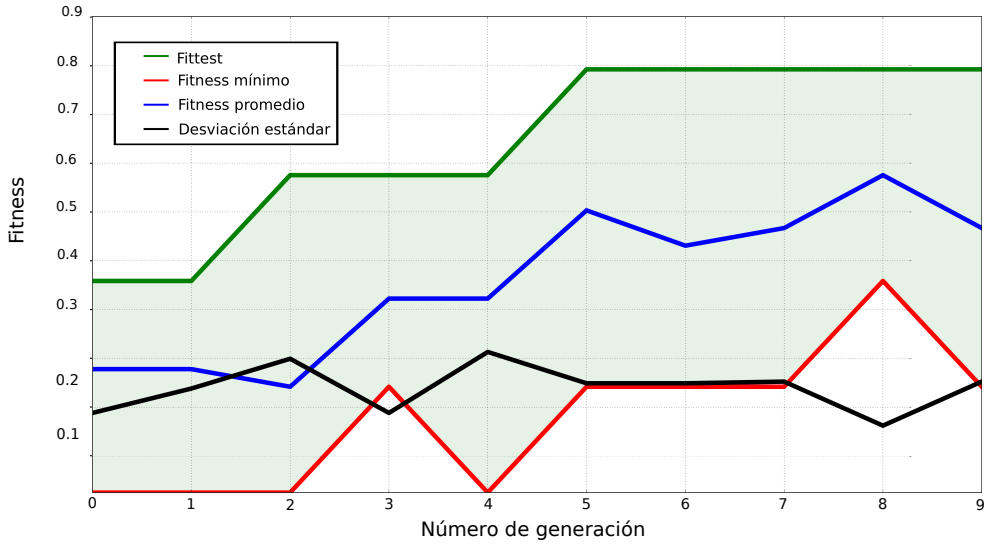

Figura 4.7: Evolución del *fitness* en cada generación para la plataforma real; gráfico de elaboración propia

se utilizaron los propuestos en la experimentación sobre la plataforma virtual, como se presentó en la Sección 4.5.2.

4.6.2. Comparación con la línea base

Se toma como línea base a individuos de la mejor generación luego de ejecutar el algoritmo genético en el simulador. El máximo desempeño de dichos individuos, obtenidos en la simulación, sobre el robot físico es de caminatas de sólo un ciclo y medio estables, y luego de ejecutar el algoritmo genético en el robot físico se logra obtener individuos que alcanzan una marcha estable de tres ciclos (sin contar el pre-ciclo), obteniendo una mejora del 100 %.

En la Figura 4.5 se ilustra una serie de capturas correspondientes al ciclo de marcha generado; la marcha completa obtenida para este experimento se puede ver en el video disponible en: http://www.fing.edu.uy/~aaguirre/maestria/tscf/walk_physic.mp4. La distancia recorrida por el robot para este experimento fue de 15 centímetros.

4.6.3. Configuración del entorno

El problema del equilibrio es sumamente complejo; con el objetivo de verificar incrementalmente la propuesta en la plataforma robótica real, se utiliza un apoyo que evita al robot caerse en hacia sus costados, no así hacia ade-

lante o atrás; reducir el problema mediante este cambio mejora el desempeño de las soluciones y permite una convergencia más rápida, a costa de obtener soluciones con un *reality gap* mayor. El apoyo consiste en una barra sujetada a la espalda del robot en un extremo y en el otro a un eje vertical, con un punto de rotación, que no impide el movimiento del robot, ver Figura 4.8. Como trabajo futuro se plantea el estudio de mejoras a la propuesta que permitan prescindir de esta estructura; un posible camino es trabajar con una población inicial más grande o con mayor diversidad; otra alternativa es implementar un controlador de equilibrio de lazo cerrado que ejecute en paralelo, pudiendo modificar las poses prescriptas de la secuencia en los momentos que se presenta una inestabilidad.

Por razones de seguridad, se utilizan hilos sujetados a la cadera del robot para evitar que el mismo impacte contra el piso, afectando la integridad del mismo.

En lugar de utilizar el controlador CM-5, descripto en la Sección 4.1, se utilizó un PC donde se ejecuta el algoritmo propuesto y calcula la salida para los actuadores, el cual comunica mediante un enlace USB-SERIAL con la plataforma robótica; el conversor USB-Serial del tipo *usb2dynamixel* fue montado sobre el hombro izquierdo del robot.

El cable USB y el conversor son elementos que introducen distorsión, ya que son artefactos necesarios únicamente por razones tecnológicas, y no fueron modelados en la plataforma virtual; por lo que los individuos utilizados como población inicial van a ser afectados negativamente por esta configuración. Aunque también estas diferencias son una oportunidad para explotar las capacidades del algoritmo al resolver el problema de la correspondencia. Como trabajo futuro se propone utilizar el módulo de comunicación inalámbrica ZIGBEE disponible en el CM-5.

4.6.4. Limitaciones de la plataforma física

A continuación se describen las limitaciones presentes en la plataforma implementada.

La función de *fitness* requiere de un usuario que la evalúe, enlenteciendo y haciendo engorrosa su aplicación; como trabajo futuro se propone automatizar este proceso.

La plataforma elegida posee varias desventajas: al ser un kit presenta pro-

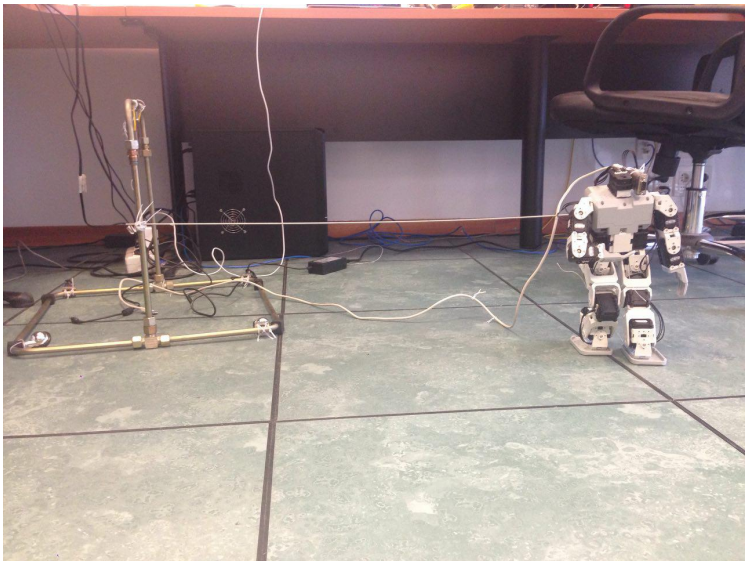

Figura 4.8: Entorno de trabajo montado para experimentación con la plataforma física.

blemas de robustez; es común que el cable del bus de motores se dañe por el movimiento de las piezas circundantes de las articulaciones; el ensamblado de las partes es complicado. Debido a estas razones, el mantenimiento del robot es muy engorroso, siendo un impedimento importante en la etapa de experimentación.

Capítulo 5

Consideraciones finales

5.1. Conclusiones y trabajo futuro

Este trabajo presenta una arquitectura para el aprendizaje de tareas motoras mediante la imitación, fundada sobre la formalización propuesta de Argall [6]. La misma ha sido validada en un entorno de simulación física, alcanzando avances promisorios en las pruebas realizadas sobre la plataforma real.

Para las pruebas se utilizó uno de los comportamientos más importantes y desafiantes necesarios en el campo de la robótica humanoide, obtener una caminata estable que se asemeje a la humana. El enfoque propuesto permite a un robot humanoide aprender a caminar basándose únicamente en la información de captura de movimiento humano, y una función de *fitness* que permita evaluar la calidad de las soluciones obtenidas, sin la necesidad de utilizar un modelo físico detallado de la dinámica del robot y su entorno.

Este Capítulo se divide en dos grandes secciones: la primera dedicada al estudio de las conclusiones y trabajo futuro para el problema guía de la marcha bípeda, y el segundo respecto a la propuesta general de aprendizaje por imitación.

5.2. Análisis del problema guía

El enfoque de aprendizaje por imitación ha mostrado un gran impacto en la búsqueda de soluciones factibles, tal como se presentó en la Sección 4.5, donde se obtuvo una caminata de 9 pasos; los experimentos de línea de base utilizando la configuración de población inicial clásica de un algoritmo genético

fallan.

El problema del *reality gap*, ha sido abordado utilizando la misma propuesta que la empleada en la plataforma simulada para enfrentar el problema de la correspondencia; los resultados obtenidos han sido positivos, permitiendo implementar un primer acercamiento a la marcha en el robot físico.

Como fue presentado en la Sección 4.6 una ejecución de muy pocas generaciones en la plataforma real, ha mejorado considerablemente el desempeño de los individuos obtenidos originalmente en el ambiente simulado, produciendo un movimiento, de 7 pasos, con las características deseadas. Más trabajo es necesario para resolver los problemas de estabilidad encontrados, los cuales han sido mitigados provisoriamente mediante el soporte utilizado para permitir estudiar el problema de forma incremental.

La función de *fitness* presentada en este trabajo es intrínsecamente multi-objetivo, la misma ha sido implementada como una suma ponderada de objetivos, pero se podría realizar una mejor exploración del espacio de búsqueda si se utilizara un enfoque multi-objetivo, por lo que la determinación de la frontera de Pareto se deja para su tratamiento en trabajos futuros.

El modelo utilizado del robot Bioloid en el ambiente de simulación no es oficial, y no pudieron relevarse registros documentados de que los resultados obtenidos en la plataforma virtual hayan sido utilizados en la plataforma real. De existir problemas en la simulación de la realidad proporcionada por el modelo, la utilización de otro puede reducir el *reality gap*.

A causa del problema con el simulador descripto en la Sección 4.5.5, la calibración de los parámetros del algoritmo genético insumió mucho tiempo y tuvo que ser simplificada, utilizando un conjunto de valores reducido del que debería de haberse utilizado; existen posibilidades de mejora del procedimiento al utilizar alguna de las propuestas que se describen más adelante en la Sección 5.3, para paliar el problema de la simulación a tiempo fijo, por lo que se propone como trabajo futuro refinar la calibración de parámetros para el problema guía.

5.2.1. Comparación con otras propuestas

Los resultados de la marcha obtenidos en el ambiente virtual presentan, a primera vista, un aspecto más humano que otras técnicas populares aplicadas al mismo modelo de robot, como ser: osciladores no lineales, o secuencias codificadas manualmente, donde la estrategia seguida es doblar las rodillas y

balancear el torso en dirección lateral para garantizar la estabilidad.

Con el objetivo de poder medir el desempeño de la solución en el problema de la marcha presentado en esta propuesta, respecto a otras técnicas, se realizó una prueba de concepto mediante la técnica de osciladores sinusoidales implementados con series truncadas de Fourier; esta técnica es ampliamente utilizada para la implementación de marchas en robots humanoides.

La prueba realizada consistió en intentar recrear la implementación del trabajo de Zamiri y colaboradores [64] en el ambiente virtual tomando como base el trabajo de [51], utilizando la misma escena y robot que los empleados en la validación de la propuesta por imitación presentada en la Sección 4.5; puede verse la ejecución del experimento en: https://www.fing.edu.uy/~aaguirre/maestria/fourier_experiment.avi.

Los resultados obtenidos superan en estabilidad a los presentados en la propuesta por imitación, no así en distancia recorrida por paso, donde la propuesta presentada es un 33 % mayor para el caso presentado en la Sección 4.5.

Mediante inspección visual, tomando como referencia las propiedades dinámicas de la locomoción humana, la marcha obtenida expone un comportamiento que difiere claramente del observado en el humano. Como se menciona en la Sección 2.3.2, la semejanza con la marcha humana es un punto importante a tener en cuenta para permitir el desarrollo de áreas como la robótica doméstica en los años venideros. Esta intuición necesita ser confirmada, en la búsqueda de mecanismos objetivos para evaluar dichas propiedades dinámicas, se propone utilizar la técnica de DTW, ver Sección 2.3.2, para evaluar la similitud respecto a la marcha humana de nuestra propuesta y la implementada utilizando el enfoque de osciladores sinusoidales.

Si c_human es la serie temporal que representa el movimiento de un humano promedio; $c\_imitation$ es el resultado producto de utilizar la presente propuesta de imitación y $c\_oscillators$ la obtenida mediante el enfoque de osciladores sinusoidales entonces se busca estudiar el valor de $dtw(c\_imitation, c\_human)$ respecto del de $dtw(c\_oscillators, c\_humano)$.

Otra forma de obtener un un análisis objetivo sobre la similitud respecto a la marcha humana puede obtenerse utilizando los estudios del centro de masa propuestos por Minetti, citados en la Sección 2.3.2.

Estos análisis quedaron fuera del alcance de esta etapa del proyecto, un mayor estudio es requerido, y por ello es planteado como un trabajo a futuro.

Existe una gran cantidad de técnicas que puede analizar para el problema guía seleccionado, un relevamiento completo y contemporáneo puede encontrarse en [48], como trabajo futuro se propone evaluar el desempeño obtenido por la técnica propuesta respecto a otras de las técnicas relevadas para generar marchas estables.

5.3. Análisis sobre la solución propuesta

El control motor se realiza sin retroalimentación sensorial. En los vertebrados, la retroalimentación sensorial de los Husos Musculares, del tendón, de los receptores de la articulación y de la piel, se utilizan como entrada para controlar los patrones de locomoción [8].

A modo de trabajo futuro se plantea explorar respuestas a la pregunta de ¿Cómo sabe el robot cuándo imitar?, presentada en la Sección 2.1.1, la cual fue dejada fuera del alcance del presente trabajo.

La solución propuesta, requiere de la especificación de una función de *fitness* que permita evaluar la calidad de las soluciones generadas por el mecanismo evolutivo; especificar las características de calidad del movimiento, es más simple que realizar una especificación exhaustiva del controlador, pero de todas maneras requiere de cierto conocimiento por parte del usuario, para identificar las características de calidad de una solución y lograr escribirlo como una función.

La solución propuesta fue validada mediante la exitosa generación de un controlador para la marcha bípeda, que se utilizó en un robot humanoide de 18 grados de libertad. No se realizaron pruebas en robots con más grados de libertad, por lo cual se plantea como trabajo futuro estudiar como impacta el escalar en grados de libertad al desempeño de la técnica.

Los resultados obtenidos en la plataforma física, permiten visualizar el potencial de la técnica al lograr atacar el problema del *reality gap*, introducido en la Sección 2.2.1, aunque más trabajo es necesario realizar para mejorar la plataforma de experimentación y los resultados.

También es importante señalar, que el uso de un algoritmo genético es un enfoque inexplorado pero prometedor para la solución del problema de correspondencia.

Utilizar aprendizaje por imitación como técnica, aporta una reducción del espacio de búsqueda respecto a otras técnicas, tal como fue mencionado en la

Sección 1.3, esta reducción permitió al algoritmo propuesto resolver exitosamente el problema del aprendizaje de la marcha bípeda, como pudo verse en el experimento contra una *línea base* aleatoria realizado en el Capítulo 4, a pesar de no haber utilizado métodos de reducción dimensional; otros trabajos de aprendizaje motor sobre tareas similares con cantidad de grados de libertad semejantes justifican el uso de técnicas de reducción dimensional para poder tratar el problema [19]. Como trabajo futuro se propone un estudio riguroso que permita comparar la calidad de las soluciones obtenidas por cada método, para estudiar si la reducción dimensional condiciona el espacio de soluciones.

Como trabajo futuro resulta interesante evaluar el desempeño de esta propuesta respecto a otras que utilicen técnicas alternativas para resolver el problema genérico del aprendizaje de tareas motoras.

Como fue comentado en la Sección 4.5.5, el simulador presenta una limitación muy grande al no permitir ejecutar los experimentos a un paso de simulación mayor, por lo que los mismos sobre la plataforma virtual tardan lo mismo que en la plataforma real. Esto es un problema para emplear técnicas evolutivas ya que se requiere ejecutar una cantidad muy grande de veces.

Como trabajo futuro para poder hacer frente a este problema se propone utilizar otro simulador que soporte dicho requerimiento, o emplear técnicas de computación de alta performance, de forma de paralelizar las ejecuciones de las evaluaciones en el simulador.

Los motores utilizados permiten regular el torque que ejercen, para simplificar el problema siempre se utilizó el valor de torque por defecto, como trabajo futuro se propone estudiar más a fondo las consecuencias que ello genera.

Se ha utilizado un enfoque de aprendizaje por demostración con varios instructores con el objetivo de generar diversidad, permitiendo al aprendiz reducir el espacio de búsqueda de soluciones. Los resultados experimentales refuerzan la intuición de que este enfoque tiene beneficios en comparación a otro de un único maestro, pero más trabajo es necesario para validar esta conclusión.

Se observó en los experimentos simulados que la generación de individuos factibles se da, con mucha más frecuencia, en las primeras 35 generaciones, luego la mejora es marginal; una posible explicación podría ser que el operador de mutación no genere poses útiles luego de alcanzar soluciones factibles debido a que valor de la desviación estándar(θ) sea muy grande y no le permita acercarse correctamente al óptimo local, otra explicación es que se haya

encontrado el óptimo local. Para el caso en que todavía no se haya alcanzado el óptimo local, modificar el operador de mutación para que el valor de θ vaya decrementando en función de que pasen generaciones podría aportar mejores resultados.

Los parámetros estimados, para el algoritmo genético en el problema de la marcha bípeda no han sido probados con otro tipo de movimientos, dado lo completo del movimiento de la marcha (que incluye movimiento de todo el cuerpo, además de mantener el equilibrio) es esperable que estos valores obtengan buenos resultados también para otro tipo de movimientos, como trabajo futuro se propone realizar un trabajo más formal que permita corroborar dicha intuición.

Referencias bibliográficas

[1] Aguirre, A. (2013). Lucy: Imitation learning on humanoid robot. https://github.com/aguirrea/lucy. Accessed: 2017-05-23.

[2] Alexander, R. M. (1984). The gaits of bipedal and quadrupedal animals. The International Journal of Robotics Research, 3(2):49–59.

[3] Alissandrakis, A., Nehaniv, C., and Dautenhahn, K. (2002). Imitation with alice: learning to imitate corresponding actions across dissimilar embodiments. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 32(4):482–496.

[4] Ames, A. D. (2012). First steps toward automatically generating bipedal robotic walking from human data. In Robot Motion and Control 2011, pages 89–116. Springer.

[5] Arbib, M. A. (2002). Imitation in animals and artifacts. chapter The Mirror System, Imitation, and the Evolution of Language, pages 229–280. MIT Press, Cambridge, MA, USA.

[6] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. Robotics and autonomous systems, 57(5):469–483.

[7] Berger, E., Amor, H. B., Vogt, D., and Jung, B. (2008). Towards a simulator for imitation learning with kinesthetic bootstrapping. In Workshop Proceedings of Intl. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR), pages 167–173.

[8] Billard, A. (2001). Learning motor skills by imitation: a biologically inspired robotic model. Cybernetics & Systems, 32(1-2):155–193.

[9] Billard, A., Calinon, S., Dillmann, R., and Schaal, S. (2008). Robot programming by demonstration. In Siciliano, B. and Khatib, O., editors, Handbook of Robotics, pages 1371–1394. Springer, Secaucus, NJ, USA.

[10] Billard, A., C. S. D. R. and Schaal, S. (2007). Robot Programming by Demonstration. Handbook of Robotics. MIT Press. (in Press).

[11] Bongard, J. C. (2013). Evolutionary robotics. Commun. ACM, 56(8):74–83.

[12] Breazeal, C., Brooks, A., Gray, J., Hoffman, G., Kidd, C., Lee, H., Lieberman, J., Lockerd, A., and Mulanda, D. (2004). Humanoid robots as cooperative partners for people. Journal of Humanoid Robots, 1:2004.

[13] Breazeal, C. and Scassellati, B. (2002). Imitation in animals and artifacts. chapter Challenges in Building Robots That Imitate People, pages 363–390. MIT Press, Cambridge, MA, USA.

[14] Calinon, S. (2009). Robot Programming by Demonstration: A Probabilistic Approach. EPFL/CRC Press. EPFL Press ISBN 978-2-940222-31-5, CRC Press ISBN 978-1-4398-0867-2.

[15] Calinon, S. and Billard, A. (2007). Learning of gestures by imitation in a humanoid robot. Technical report, Cambridge University Press.

[16] Calinon, S., Guenter, F., and Billard, A. (2005). Goal-directed imitation in a humanoid robot. In in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA.

[17] Castelán, M. and Arechavaleta, G. (2009). Approximating the reachable space of human walking paths: a low dimensional linear approach. In Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on, pages 81–86. IEEE.

[18] Chalodhorn, R., Grimes, D. B., Grochow, K., and Rao, R. P. (2007). Learning to walk through imitation. In IJCAI, volume 7, pages 2084–2090.

[19] Chalodhorn, R. and Rao, R. P. (2010). Learning to imitate human actions through eigenposes. In From Motor Learning to Interaction Learning in Robots, pages 357–381. Springer.

[20] Chalodhorn, R. and Rao, R. P. N. (2009). Using eigenposes for lossless periodic human motion imitation. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2502–2509.

[21] Chersi, F. (2012). Learning through imitation: a biological approach to robotics. IEEE transactions on autonomous mental development, 4(3):204–214.

[22] Collins, S., Ruina, A., Tedrake, R., and Wisse, M. (2005). Efficient bipedal robots based on passive-dynamic walkers. Science, 307(5712):1082–1085.

[23] Cully, A. and Mouret, J. B. (2016). Evolving a behavioral repertoire for a walking robot. Evol. Comput., 24(1):59–88.

[24] Dautenhahn, K. and Nehaniv, C. L., editors (2002). Imitation in Animals and Artifacts. MIT Press, Cambridge, MA, USA.

[25] Eaton, M. (2008). Evolving humanoids: Using artificial evolution as an aid in the design of humanoid robots. INTECH Open Access Publisher.

[26] Fitzpatrick, P., Harada, K., Kemp, C. C., Matsumoto, Y., Yokoi, K., and Yoshida, E. (2016). Humanoids, pages 1789–1818. Springer International Publishing, Cham.

[27] Gallese, V., Fadiga, L., Fogassi, L., and Rizzolatti, G. (1996). Action recognition in the premotor cortex. Brain, 119(2):593–609.

[28] Gen, M. and Cheng, R. (2000). Genetic algorithms and engineering optimization, volume 7. John Wiley & Sons.

[29] Graphics Laboratory, C. M. U. (2007). Graphics lab motion capture database. http://mocap.cs.cmu.edu/. Accessed: 2017-05-23.

[30] Guindy, M. and Elias, R. (2017). Happy feet maya plugin to generate customized gait cycles for 3d characters. In 2017 13th International Computer Engineering Conference (ICENCO), pages 372–377.

[31] Heinen, M. R. and Osório, F. S. (2006). Applying genetic algorithms to control gait of physically based simulated robots. In Evolutionary Computation, 2006. CEC 2006. IEEE Congress on, pages 1823–1830. IEEE.

[32] Husbands, P. (2010). Evolutionary Robotics, pages 373–382. Springer US, Boston, MA.

[33] Iosa, M., Fusco, A., Marchetti, F., Morone, G., Caltagirone, C., Paolucci, S., and Peppe, A. (2013). The golden ratio of gait harmony: repetitive proportions of repetitive gait phases. BioMed research international, 2013.

[34] Kulic, D. and Nakamura, Y. (2008). Incremental learning and memory consolidation of whole body motion patterns. In International conference on epigenetic robotics, pages 61–68.

[35] Kuniyoshi, Y., Inaba, M., and Inoue, H. (1994). Learning by watching: extracting reusable task knowledge from visual observation of human performance. IEEE Transactions on Robotics and Automation, 10(6):799–822.

[36] Lee, J.-Y. and Lee, J.-J. (2004). Optimal walking trajectory generation for a biped robot using multi-objective evolutionary algorithm. In Control Conference, 2004. 5th Asian, volume 1, pages 357–364. IEEE.

[37] Leitner, J. (2016). Malachy eaton: Evolutionary humanoid robotics - springer, 2015, ISBN 978-3-662-44598-3. Genetic Programming and Evolvable Machines, 17(1):75–76.

[38] Lemire, D. (2009). Faster retrieval with a two-pass dynamic-time-warping lower bound. Pattern recognition, 42(9):2169–2180.

[39] Lezama, D., Sklar, A., and Tejera, G. (2005). ahumandroid prototype: Primeros pasos en robótica bípeda workshop del cafr2005.

[40] Marion Fischer, Gwendolin Lepeltier, J. M. (2015). De la robotique à l'humanoïde... http://mariongwenjulie.wixsite.com/tpe-robothumanoide/blank-1. Accessed: 2017-05-23.

[41] Mataric, M., Williamson, M., Demiris, J., and Mohan, A. (1998). Behavior-based primitives for articulated control.

[42] Minetti, A. E., Cisotti, C., and Mian, O. S. (2011). The mathematical description of the body centre of mass 3d path in human and animal locomotion. Journal of biomechanics, 44(8):1471–1477.

[43] Nehaniv, C. L. and Dautenhahn, K. (2002). Imitation in animals and artifacts. chapter The Correspondence Problem, pages 41–61. MIT Press, Cambridge, MA, USA.

[44] Nguyen-Tuong, D. and Peters, J. (2011). Model learning for robot control: a survey. Cognitive processing, 12(4):319—340.

[45] Peters, J., Lee, D. D., Kober, J., Nguyen-Tuong, D., Bagnell, J. A., and Schaal, S. (2016). Robot Learning, pages 357–398. Springer International Publishing, Cham.

[46] Pias, G. and Baliero, M. (2011). Fútbol de robots para la liga humanoide de robocup. https://www.fing.edu.uy/inco/grupos/mina/pGrado/salimoo/. Accessed: 2017-05-23.

[47] Picado, H., Gestal, M., Lau, N., Reis, L. P., and Tomé, A. M. (2009). Automatic generation of biped walk behavior using genetic algorithms. In International Work-Conference on Artificial Neural Networks, pages 805–812. Springer.

[48] Prakash, C., Kumar, R., and Mittal, N. (2018). Recent developments in human gait research: parameters, approaches, applications, machine learning techniques, datasets and challenges. Artificial Intelligence Review, 49(1):1–40.

[49] Ramachandran, V. (2000). Mirror neurons and imitation learning as the driving force behind "the great leap forwardïn human evolution. Edge, 69.

[50] Rao, R. P. and Meltzoff, A. N. (2003). Imitation learning in infants and robots: Towards probabilistic computational models. In Proc. of Artificial Intelligence and Simulation of Behaviors, volume 559, page 560.

[51] Reda Al-Bahrani, Matt Derry, G. W. (2012). Genetic algorithm walker. https://sites.google.com/a/u.northwestern.edu/gawalker/system-design/simulator-and-robot-model. Accessed: 2017-05-23.

[52] Rizzolatti, G., Fadiga, L., Gallese, V., and Fogassi, L. (1996). Premotor cortex and the recognition of motor actions. Cognitive brain research, 3(2):131–141.

[53] Robotis, B. U. G. (2007). V 1.1. Bucheon City, Republic of Korea: ROBOTIS Co., LTD.

[54] Sander, M. (2015). Tesis de grado de la universidad de la república del uruguay. Proyecto PGILearn.

[55] Sander, M., Aguirre, A., and Benavides, F. (2016). Humanoid robot learning by demonstration based on visual bootstrapping technique. In 2016 XLII Latin American Computing Conference (CLEI), pages 1–8.

[56] Schaal, S., Ijspeert, A., and Billard, A. (2003). Computational approaches to motor learning by imitation. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 358(1431):537–547.

[57] Shafii, N., Khorsandian, A., Abdolmaleki, A., and Jozi, B. (2009). An optimized gait generator based on fourier series towards fast and robust biped locomotion involving arms swing. In 2009 IEEE International Conference on Automation and Logistics, pages 2018–2023.

[58] Torricelli, D. e. a. (2014). Benchmarking Human-Like Posture and Locomotion of Humanoid Robots: A Preliminary Scheme, pages 320–331. Springer International Publishing, Cham.

[59] van Dalen, S. (2012). A Linear Inverted Pendulum Walk Implemented on TUlip. PhD thesis, Eindhoven University of Technology.

[60] Wang, X., Kyrarini, M., Ristić-Durrant, D., Spranger, M., and Gräser, A. (2016). Monitoring of gait performance using dynamic time warping on imu-sensor data. In 2016 IEEE International Symposium on Medical Measurements and Applications (MeMeA), pages 1–6.

[61] Wehner, S. and Bennewitz, M. (2009). Optimizing the gait of a humanoid robot towards human-like walking. In ECMR, pages 277–282.

[62] Wieber, P.-B., Tedrake, R., and Kuindersma, S. (2016). Modeling and Control of Legged Robots, pages 1203–1234. Springer International Publishing, Cham.

[63] Wright, J. and Jordanov, I. (2015). Intelligent approaches in locomotion-a review. Journal of Intelligent & Robotic Systems, 80(2):255–277.

[64] Zamiri, A., Farzad, A., Saboori, E., Rouhani, M., Naghibzadeh, M., and Fard, A. M. (2008). An evolutionary gait generator with online parameter adjustment for humanoid robots. In 2008 IEEE/ACS International Conference on Computer Systems and Applications, pages 9–14.

Glosario

Argumento El argumento, abreviado como arg, de un número complejo z es el ángulo comprendido entre el eje real positivo del plano complejo y la línea que une z con el origen de dicho plano.

Footprint En informática, el footprint es la cantidad de espacio que una unidad particular de software o hardware ocupa.

Framework Un framework, entorno de trabajo o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Husos musculares Los husos neuromusculares son receptores sensoriales en el interior del vientre muscular que detecta cambios en la longitud del músculo.

Metadata Son datos que describen otros datos.

Número de Froude Es un número adimensional que relaciona el efecto de las fuerzas de inercia y la fuerzas de gravedad que actúan sobre un fluido.

Parsers Son programas informáticos que analizan una cadena de símbolos de acuerdo a las reglas de una gramática formal.

Script Es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

Serial Es el proceso de envío de datos de un bit a la vez, de forma secuencial, sobre un canal de comunicación o un bus.

Serie temporal Una serie temporal o cronológica es una secuencia de datos, observaciones o valores, medidos en determinados momentos y ordenados cronológicamente.

Stop-motion El stop motion es una técnica de animación que consiste en aparentar el movimiento de objetos estáticos por medio de una serie de imágenes fijas sucesivas.

Zigbee Zigbee es el nombre de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal.

APÉNDICES

Apéndice 1

Archivos de configuración

A continuación se describen los archivos de configuración más importantes de la plataforma desarrollada, los valores corresponden con los utilizados en la etapa de experimentación presentada en el Capítulo 4.

1.1. Configuración del entorno

```
<System>
    <!-- System paths-->
    <Directory>
        <Name>GAWalk Files</Name>
        <Value>/mocap/gawalk/xml</Value>
    </Directory>
    <Directory>
        <Name>ADHOC Files</Name>
        <Value>/mocap/tests/xml</Value>
    </Directory>
    <Directory>
        <Name>Transformed CMU mocap Files</Name>
        <Value>/mocap/cmu_mocap/xml/</Value>
    </Directory>
    <Directory>
        <Name>UIBLH mocap Files</Name>
        <Value>/mocap/uiblh_mocap/xml/</Value>
    </Directory>
```

```
<Directory>
    <Name>CMU mocap Files</Name>
    <Value>/mocap/cmu_mocap/bvh/</Value>
</Directory>
<Directory>
    <Name>Baliero transformed walk Files</Name>
    <Value>/mocap/baliero/xml/</Value>
</Directory>
<Directory>
    <Name>Lucy evolved walk cycles Files</Name>
    <Value>/mocap/lucy_walk_cycles/xml/</Value>
</Directory>
<Directory>
    <Name>Genetic Pool</Name>
    <Value>/gene_pool/</Value>
</Directory>
<File>
    <Name>Lucy vrep model</Name>
    <Value>/simulator/models/genetic_bioloid_without_texture.ttt</Value>
    <!--<Value>/simulator/models/genetic_bioloid_Bjorn_4_lucy.ttt</Value> -->
</File>
<Property>
    <!-- see the lucy.sh file to use the same name-->
    <Name>System Log</Name>
    <Value>out.txt</Value>
</Property>

<!-- Vrep connection parameters -->

<Property>
    <Name>Vrep IP</Name>
    <Value>127.0.0.1</Value>
</Property>
<Property>
    <Name>Vrep port</Name>
    <Value>19997</Value>
```

```
</Property>

<!-- Lucy vrep parameters -->

<Property>
    <Name>FALL_THRESHOLD_DOWN</Name>
    <Value>0.18</Value>
</Property>
<Property>
    <Name>FALL_THRESHOLD_UP</Name>
    <Value>0.30</Value>
</Property>
<Property>
    <Name>BALANCE_HEIGHT</Name>
    <Value>0.2226</Value>
</Property>

<!-- Values for Björn P Mattsson model
<Property>
    <Name>FALL_THRESHOLD_DOWN</Name>
    <Value>0.22</Value>
</Property>
<Property>
    <Name>FALL_THRESHOLD_UP</Name>
    <Value>0.40</Value>
</Property>
<Property>
    <Name>BALANCE_HEIGHT</Name>
    <Value>0.325</Value>
</Property>
-->

<Property>
    <Name>ConnectionTimeout</Name>
    <Value>5000</Value>
</Property>
```

```
<Property>
    <Name>threadingTime</Name>
    <Value>0.5</Value>
</Property>
<Property>
    <Name>Vrep not implemented joints</Name>
    <Value>R_Elbow_Yaw L_Elbow_Yaw L_Shoulder_Yaw R_Shoulder_Yaw</Value>
</Property>
<Property>
    <Name>Lucy render enable</Name>
    <Value>1</Value>
</Property>
    <Property>
    <Name>blank screen?</Name>
    <Value>0</Value>
</Property>
<Property>
    <Name>Lucy simulated?</Name>
    <Value>1</Value>
</Property>
<Property>
    <Name>synchronous mode?</Name>
    <Value>1</Value>
</Property>
<Property>
    <Name>simulation time step</Name>
    <Value>0.50</Value>
</Property>
<Property>
    <Name>physics enable?</Name>
    <Value>1</Value>
</Property>
<Property>
    <Name>re-evaluate fittest?</Name>
    <Value>1</Value>
</Property>
```

```
<Property>
    <Name>Concatenate walk cycles?</Name>
    <!-- 0 means not concatenate cycles, n > 1;
    means that the cycle will be repeated n times-->
    <Value>4</Value>
</Property>

<!-- Parser options -->

<Property>
    <Name>number of frames to skip</Name>
    <Value>5</Value>
</Property>

<!-- GA parameters -->

<Property>
    <Name>Population size</Name>
    <Value>30</Value>
</Property>
<Property>
    <Name>Number of generations</Name>
    <Value>60</Value>
</Property>
<Property>
    <Name>Crossover operator</Name>
    <Value>crossovers.G2DListCrossoverSingleNearHPoint</Value>
    <!-- <Value>Crossovers.G2DListCrossoverSingleHPoint</Value> -->
</Property>
<Property>
    <Name>Mutator operator</Name>
    <!-- <Value>mutators.G2DListMutatorRealGaussianSpline</Value> -->
    <Value>mutators.G2DListMutatorRealGaussianSpline</Value>
    <!-- <Value>Mutators.G2DListMutatorRealGaussianGradient</Value> -->

</Property>
```

```
    <Property>
        <Name>Selection operator</Name>
        <!--<Value>Selectors.GUniformSelector</Value>-->
        <!--<Value>Selectors.GRankSelector</Value>-->
        <Value>Selectors.GTournamentSelector</Value>
        <!-- pyevol uses RouletteWheel for the tournament selection-->
        <!--<Value>Selectors.GRouletteWheel</Value>-->
    </Property>
    <Property>
        <!-- probability that a couple get crossed -->
        <Name>CrossoverRate</Name>
        <Value>1.0</Value>
    </Property>
    <Property>
        <Name>MutationRate</Name>
        <!--value between 0 and 0.1, 0.02 represents 0.2 %-->
        <Value>0.05</Value>
    </Property>
    <Property>
        <!-- percentage of the best individuals that are
        copied to the next generation, p.e. 0.3
        correspond to the 30% of the individuals-->
        <Name>Elitism replacement percentage</Name>
        <Value>0.3</Value>
    </Property>
    <Property>
        <Name>Convergence criteria enable?</Name>
        <Value>1</Value>
    </Property>
</System>
```

1.2. Mapeo robot virtual - robot físico

```
<Robot>
    <Joint>
        <Name>L_Shoulder_Pitch</Name>
```

```
    <Id>2</Id>
</Joint>
<Joint>
    <Name>R_Shoulder_Pitch</Name>
    <Id>1</Id>
</Joint>
<Joint>
    <Name>L_Shoulder_Yaw</Name>
    <Id>4</Id>
</Joint>
<Joint>
    <Name>R_Shoulder_Yaw</Name>
    <Id>3</Id>
</Joint>
<Joint>
    <Name>L_Elbow_Yaw</Name>
    <Id>6</Id>
</Joint>
<Joint>
    <Name>R_Elbow_Yaw</Name>
    <Id>5</Id>
</Joint>
<Joint>
    <Name>L_Hip_Yaw</Name>
    <Id>8</Id>
</Joint>
<Joint>
    <Name>R_Hip_Yaw</Name>
    <Id>7</Id>
</Joint>
<Joint>
    <Name>L_Hip_Roll</Name>
    <Id>10</Id>
</Joint>
<Joint>
    <Name>R_Hip_Roll</Name>
```

```
        <Id>9</Id>
    </Joint>
    <Joint>
        <Name>L_Hip_Pitch</Name>
        <Id>12</Id>
    </Joint>
    <Joint>
        <Name>R_Hip_Pitch</Name>
        <Id>11</Id>
    </Joint>
    <Joint>
        <Name>L_Knee</Name>
        <Id>14</Id>
    </Joint>
    <Joint>
        <Name>R_Knee</Name>
        <Id>13</Id>
    </Joint>
    <Joint>
        <Name>L_Ankle_Pitch</Name>
        <Id>16</Id>
    </Joint>
    <Joint>
        <Name>R_Ankle_Pitch</Name>
        <Id>15</Id>
    </Joint>
    <Joint>
        <Name>L_Ankle_Roll</Name>
        <Id>18</Id>
    </Joint>
    <Joint>
        <Name>R_Ankle_Roll</Name>
        <Id>17</Id>
    </Joint>
</Robot>
```