UNIVERSIDAD DE COSTA RICA
SISTEMA DE ESTUDIOS DE POSGRADO

MAPEO PARA TRADUCCIÓN, DEL LENGUAJE DE ACCIÓN A INSTRUCCIONES DEL MODELO DE OBJETOS

Tesis sometida a la consideración de la Comisión del Programa de Estudios de Posgrado en Ingeniería Eléctrica para optar al grado de Maestría Académica

FÉLIX DAVID SUÁREZ BONILLA

Ciudad Universitaria Rodrigo Facio, Costa Rica
2018

"Esta tesis fue aceptada por la Comisión del Programa de Estudios de Posgrado en Ingeniería Eléctrica de la Universidad de Costa Rica, como requisito parcial para optar al grado y título de Maestría Académica en Ingeniería Eléctrica."

PhD. Jaime Cascante Vindas
Representante del Decano
Sistema de Estudios de Posgrado

Dr. rer. nat. Federico Ruiz Ugalde
Director de Tesis

Dr. rer. nat. Francisco Siles Canales
Asesor

Dr. Juan Luis Crespo Mariño
Asesor

Dr. Lochi Yu Lo
Representante del Director
Programa de Posgrado en Ingeniería Eléctrica

Félix David Suárez Bonilla
Candidato

Índice general

Resumen

En este trabajo de investigación se presenta una propuesta que intenta cerrar la brecha entre el razonamiento y la planificación, y los controladores que producen movimientos y acciones en el mundo real, esto para un robot humanoide. Para esto se traducen tareas mínimas a entradas que pueden ser entedidas por un sistema que controla objetos, el cual está compuesto compuesto por cinco partes: unidad de *affordability*, unidad de mapeo, unidad de referencia, unidad de propiedades de los objetos y unidad de factibilidad. Los experimentos realizados sobre el sistema indican que la utilización de tareas simples y el diseño de la unidad de *affordances* resultan prometedores para futuros enfoques a este problema.

Abstract

This thesis presents an architecture that helps to close the gap between reasoning and planning, and controllers that produce movements and actions on the real world for a humanoid robot. For this, simple tasks are translated into inputs that can be directly used by a system that can control objects, this system is composed of five parts: the offering unit, the mapping unit, the reference unit, the object properties unit and the feasibility unit. The results indicate that the use of simple tasks and the design of the offering unit are promising for future approaches to this problem.

Índice de cuadros

Índice de figuras

Lista de abreviaturas

ARCOS-Lab: Siglas en inglés de *Autonomous Robots and Cognitive Systems Laboratory* perteneciente a la Escuela de Ingeniería Eléctrica de la Universidad de Costa Rica.

LSTM: Siglas en inglés de *Long-short term memory* la cual es un tipo especial de red neuronal recurrente conformada por una celda de memoria que puede retener su valor durante un periodo de tiempo largo o corto.

NMT: Siglas en inglés de *Neural Machine Translation* el cual es un método de traducción automática que usa una red neuronal de tamaño considerable.

Lista de definiciones

Affordance: Es el cualquier característica de un objeto que le proporciona al sujeto información de cómo realizar una acción. Dicho de otra forma, es el conocimiento que una persona puede extraer acerca de las diferentes formas de utilizar un objeto (Gibson, 1977) (Moldovan, Moreno, van Otterlo, Santos-Victor, y Raedt, 2012).

Arquitectura robótica: Este término se refiere a como el sistema se divide en subsistemas y como estos interactúan, así como a los conceptos computacionales que sigue el sistema robótico en cuestión. En cuanto al interacción de los subsistemas, usualmente se usan flechas y bloques para representar estas interacciones (Siciliano y Khatib, 2007) (Coste-Maniere y Simmons, 2000).

Capa de bajo nivel: En el contexto de un sistema robótico, se refieren a los componentes de software que interactúan directamente con el hardware. Es la capa encargada de realizar el control, conectar actuadores y sensores, así como resolver toda la cinemática. En el contexto de este trabajo, también se considera que esta capa posee la biblioteca de objetos.

Capa de alto nivel: El sistema de alto nivel trabaja a nivel más simbólico y de significados, toma instrucciones más generales (en algún tipo de lenguaje) y las convierte a instrucciones más simples. Esta capa recibe además los siguientes nombres: capa de planeamiento simbólico,

capa de planeamiento y razonamiento, o sistema de alto nivel.

Modelo de objetos: Este componente se encarga de predecir y controlar el comportamiento del objeto. Un modelo de objeto es una descripción matemática del comportamiento del objeto, es decir, un sistema dinámico donde las entradas al sistema pueden ser cualquier cosa que el robot pueda transmitir al objeto y las salidas son cualquier cosa que el robot puede medir o inferir (Ruiz, Cheng, y Beetz, 2010).

Tarea simples: Es una instrucción, expresada en algún lenguaje, que consta de tres elementos: sujeto, acción y estado deseado; y que además puede ser ejecutada por un controlador ubicado en la capa de bajo nivel del sistema robótico.

UNIVERSIDAD DE COSTA RICA

SEP Sistema de Estudios de Posgrado

Autorización para digitalización y comunicación pública de Trabajos Finales de Graduación del Sistema de Estudios de Posgrado en el Repositorio Institucional de la Universidad de Costa Rica.

Yo, Félix Suárez Bonilla, con cédula de identidad 112930631, en mi condición de autor del TFG titulado Mapeo para traducción, del lenguaje de acción a instrucciones del modelo de objetos

Autorizo a la Universidad de Costa Rica para digitalizar y hacer divulgación pública de forma gratuita de dicho TFG a través del Repositorio Institucional u otro medio electrónico, para ser puesto a disposición del público según lo que establezca el Sistema de Estudios de Posgrado. **SI** [X] **NO** *[ ]

***En caso de la negativa favor indicar el tiempo de restricción:** _____ **año (s).**

Este Trabajo Final de Graduación será publicado en formato PDF, o en el formato que en el momento se establezca, de tal forma que el acceso al mismo sea libre, con el fin de permitir la consulta e impresión, pero no su modificación.

Manifiesto que mi Trabajo Final de Graduación fue debidamente subido al sistema digital Kerwá y su contenido corresponde al documento original que sirvió para la obtención de mi título, y que su información no infringe ni violenta ningún derecho a terceros. El TFG además cuenta con el visto bueno de mi Director (a) de Tesis o Tutor (a) y cumplió con lo establecido en la revisión del Formato por parte del Sistema de Estudios de Posgrado.

INFORMACIÓN DEL ESTUDIANTE:

Nombre Completo: Félix David Suárez Bonilla.

Número de Carné: A45276 Número de cédula: 112930631.

Correo Electrónico: felixdavidsuarezbonilla@gmail.com.

Fecha: 10/2/2020. Número de teléfono: 8783-4957.

Nombre del Director (a) de Tesis o Tutor (a): Federico Ruiz Ugalde.

Félix Suárez Bonilla.

FIRMA ESTUDIANTE

Nota: El presente documento constituye una declaración jurada, cuyos alcances aseguran a la Universidad, que su contenido sea tomado como cierto. Su importancia radica en que permite abreviar procedimientos administrativos, y al mismo tiempo genera una responsabilidad legal para que quien declare contrario a la verdad de lo que manifiesta, puede como consecuencia, enfrentar un proceso penal por delito de perjurio, tipificado en el artículo 318 de nuestro Código Penal. Lo anterior implica que el estudiante se vea forzado a realizar su mayor esfuerzo para que no sólo incluya información veraz en la Licencia de Publicación, sino que también realice diligentemente la gestión de subir el documento correcto en la plataforma digital Kerwá.

Introducción

La población mundial enfrenta un vertiginoso proceso de envejecimiento, que afecta a países desarrollados y en vías de desarrollo. El envejecimiento tiene importantes consecuencias y repercusiones para todas las facetas de la vida humana, la más importante es una mayor lentitud y una menor fluidez en la ejecución de algunas tareas, tanto desde el punto de vista físico como cognitivo (Karim, Lokman, y Redzuan, 2016a) (Karim, Lokman, y Redzuan, 2016b). La ejecución de una tarea puede ser díficil para un adulto mayor; por eso la importancia de un robot que pueda asistir, apoyar y colaborar y, de esta forma, mejorar la calidad de vida de estas poblaciones (Nava, 2011).

Las tareas a las que se enfrenta el ser humano en la vida diaria pueden requerir elevadas capacidades de comprensión del ambiente y altas destrezas manipulativas (Low y cols., 2017) (Feix, Bullock, y Dollar, 2014). La tarea de servir una taza de leche requiere de una gran cantidad de sub-tareas que por sí mismas se revisten de una gran complejidad como: encontrar el recipiente, empujar una caja, vertir líquido, sostener un recipiente y muchas otras. Por citar un ejemplo, el sostener un recipiente requiere de un conocimiento completo de como se comporta el objeto desde el punto de vista físico y mecánico, además de un entendimiento de la tarea u objetivo que se desea realizar con el recipiente.

Estas aplicaciones requieren de un robot amigable con el ambien-

te, que manipule objetos con suavidad y precisión, que cuente con una capacidad de percepción avanzada, que entienda las intenciones de los humanos y que colabore con dichas intenciones. Para que un robot cumpla con lo anterior debe tener ciertas capacidades físicas y cognitivas. En el aspecto físico el control suave puede resolver lo de la manipulación delicada. El problema de percepción se puede realizar con sensores con capacidad de detección en 3D. Finalmente, una fusión de sensores complementarios puede posibilitar la comunicación con el ser humano.

Bajo estas condiciones, las tareas que debe ejecutar un robot son complejas, las mismas usualmente se dividen en tareas más pequeñas que pueden ser ejecutadas por un controlador. Estas tareas simples pueden ser expresadas a través de una oración que tiene sujeto, verbo y predicado, las cuales se debe acoplar a los controladores del robot. El acople entre una tarea simple y los controladores del robot ha sido poco abordado, pero resulta fundamental para que un humano pueda expresar intenciones a un robot y que el mismo las ejecute en forma autónoma (Domingos, 2007).

Los campos de conocimiento de la psicología y de la neurociencia han realizado estudios relevantes sobre los humanos que pueden servir como insumo para el desarrollo de técnicas que se puedan incorporar en un robot. Este es el caso de la teoría de los *affordances* que permite explicar la relación que existe entre actores y objetos (Gibson, 1977). Este concepto se considera crucial para esta tesis y sirve como motivación e inspiración para desarrollar un mecanismo que permita acoplar

una tarea con controladores. Uno de los primeros acercamientos para manipular objetos, observando el problema desde la perspectiva de *affordances*, es la idea de modelos de objetos (Ruiz y cols., 2010) (Romay, Kohlbrecher, Conner, y von Stryk, 2015). Un modelo de objetos permite predecir y controlar un objeto abstrayendo el cuerpo del robot. Esto permite controlar los objetos directamente y no el robot.

El modelo de objetos permite abstraer la dinámica del cuerpo del robot, pero este todavía necesita ser conectado a un sistema de razonamiento y planificación para lograr transformar instrucciones a acciones sobre objetos. El sistema de alto nivel trabaja a nivel más simbólico y de significados, toma instrucciones más generales (en algún tipo de lenguaje) y las convierte a instrucciones más simples, mientras que la capa de modelos de objetos requiere de parametrizaciones y valores numéricos. Este trabajo de posgrado se enfoca en cerrar esta brecha entre el sistema de alto nivel y el modelo de objetos, a través de una nueva capa intermedia que incopora un sistema de mapeo a través de un lenguaje de acción orientado a manipulación de objetos.

1.1. Problema

Necesitamos que los robots pueden asistir, colaborar y ayudar a los seres humanos en tareas que son relevantes para estos, para ello necesitamos comunicar tareas y que las mismas sean entendidas correctamente. Por ejemplo, la tarea compleja “Hacer un desayuno” puede ser entendida por el sistema de planeamiento simbólico, pero para su

ejecución debe poder ser interpretada por los controladores del robot. Por tanto, el problema general es:

> ¿Como se comunica la interfaz entre un sistema de razonamiento y planificación y un sistema de control, en un robot humanoide?

En particular, la interfaz entre un sistema de alto nivel y un modelo de objetos. La solución propuesta es un sistema de mapeo y un lenguaje de acción orientado a manipulación de objetos, con la generalidad suficiente para desempeñar estas tareas (Ruiz y cols., 2010) (Ruiz, Cheng, y Beetz, 2011).

El sistema robótico que se investiga está orientado a manipulación de objetos, por lo que el lenguaje de acción y el mapeo del mismo van también en esa dirección. Es necesario analizar si esta solución es viable y si se puede extender a otras arquitecturas robóticas orientadas a otras tareas.

1.2. Hipótesis

En este trabajo de investigación, se parte de la conjetura o idea predeterminada de que un sistema de mapeo y un lenguaje de acción orientado a manipulación de objetos puede funcionar como capa de interfaz o capa intermedia para abstraer el sistema de alto nivel de la capa de bajo nivel (modelos de objetos) de la arquitectura robótica.

En este sentido, se considera que un lenguaje de acción, que representa tareas simples (tareas mínimas y bien definidas), es adecuado para representar la salida del sistema de alto nivel. También se parte de la premisa que un sistema que mapea la instrucción o tarea simple, para obtener los parámetros del controlador, podría en efecto cerrar la brecha entre el cuerpo (capa de bajo nivel) y la mente del robot (capa de alto nivel) (Ruiz y cols., 2010) (Ruiz y cols., 2011).

En la literatura existen lenguajes de acción, dichos lenguajes típicamente se diseñaron para representar o procesar conocimiento o ideas pero no para representar acciones de manipulación. Este es el caso de YAGI y GOLOG, que son dos lenguajes de acción utilizados en robótica, los cuales son más generales y orientados a procesamiento lógico (Ferrein y cols., 2016) (Levesque, Reiter, Lespérance, Lin, y Scherl, 1997). Por tanto podría ser necesario diseñar y formalizar un lenguaje de acción dirigido a manipular objetos rígidos en un determinado escenario.

La utilización de un lenguaje de acción plantea el reto de que el sistema de alto nivel debe poder generar oraciones en un lenguaje de acción. Esto podría añadir una complejidad adicional al sistema de planeamiento simbólico. Además, existen dos problemas al especificar y formalizar un lenguaje de acción: la expresividad del lenguaje y espacios de búsqueda intratables; debido al gran número objetos y acciones posibles sobre los objetos en el escenario.

En vista de lo anterior, se ha planteado la siguiente hipótesis para

el presente trabajo de investigación:

> Un sistema de mapeo entre un lenguaje de acción y un modelo de objetos, permite acoplar la capa de alto nivel con la capa de control de una arquitectura robótica orientada a manipulación de objetos, esto con una representatividad mayor al 92 % y un *affordability* del 100 %.

1.3. Justificación

En la actualidad existen dos grandes preguntas que muestran los problemas existentes en la robótica:

- ¿Como puede un robot controlarse a si mismo en forma autónoma?
- ¿Como pueden nuestras intenciones ser comunicadas a un robot?

Para ejemplificar esto, consideremos un robot cocinero al que le ordenamos que haga un desayuno sencillo de huevos revueltos y pan cuadrado. El robot debe ser capaz de hacer el huevo en forma **autónoma**, tener conocimiento del espacio de trabajo, ubicar los ingredientes y moverse por la cocina. El robot debe ser capaz de **entender** las instrucciones del ser humano, tomar la tarea encomendada y subdividirla en tareas más simples.

En este proyecto se va a trabajar ambos puntos, la comunicación y la autonomía. La arquitectura robótica que se toma como referencia es de tres capas, un capa de alto nivel, una capa intermedia (también llamada capa de interfaz) y una capa de bajo nivel (Ver figura 1.1). La capa de bajo nivel es la de control, el sistema de modelo de objetos también se considerá pertenenciente a la capa de bajo nivel; la capa intermedia es el sistema que se va a desarrollar y la capa de alto nivel corresponde al sistema de planeamiento simbólico.

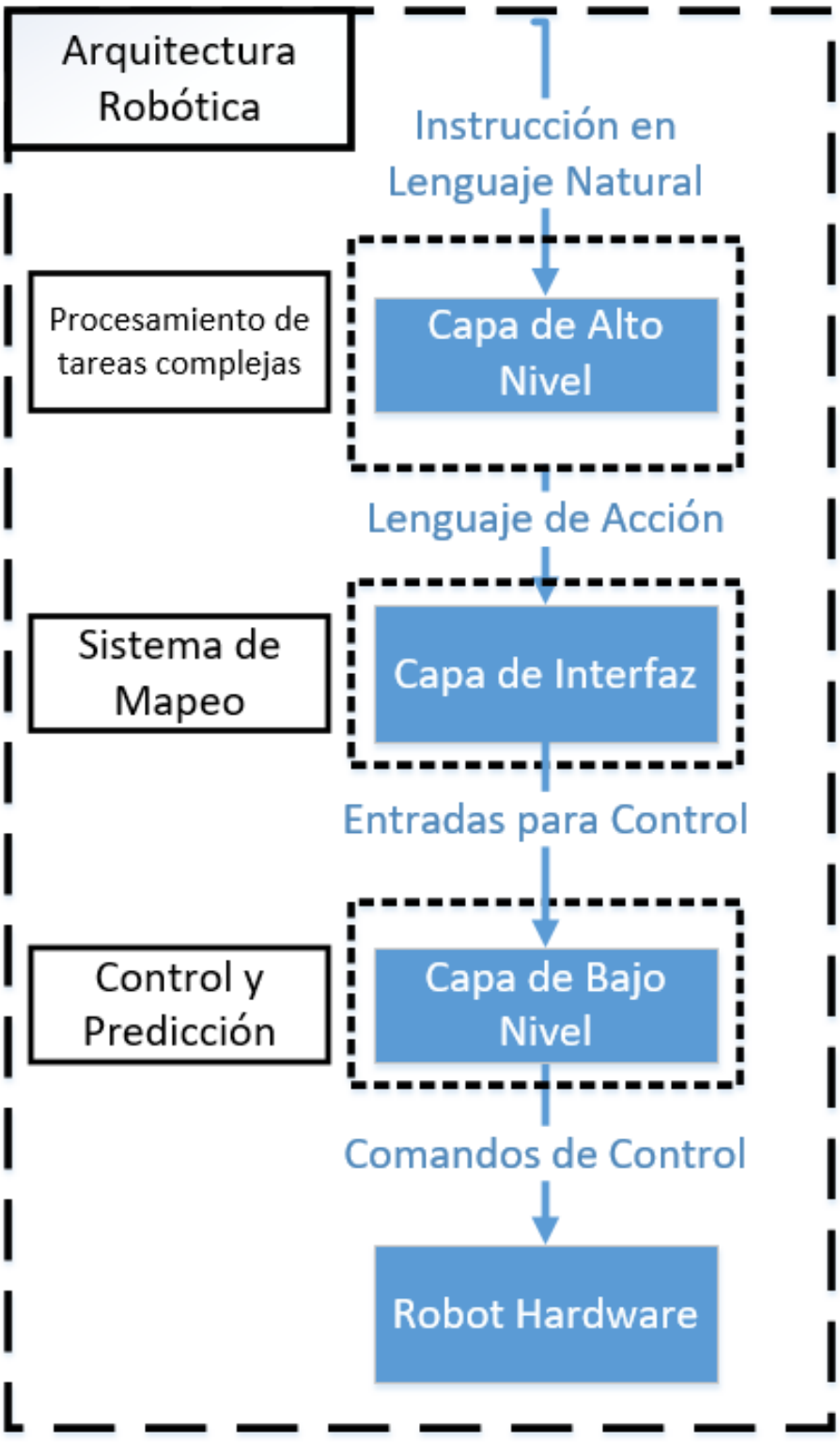

Figura 1.1: Arquitectura robótica de tres capas.

Existen experiencias previas en ciencias de la computación donde la creación de una capa de interfaz permite mayor sinergia, innovación y desarrollo; de tal forma que una mejora en una capa inferior tenga beneficios para las capas superiores y viceversa. En un inicio, la creación de una capa de interfaz puede ser recibida con escepticismo, pero cuando finalmente es aceptada, facilita un desarrollo más rápido de la capas contiguas (Domingos, 2007).

1.4. Metodología

El trabajo de esta investigación está dividida en dos etapas, primero se aborda la formalizacion de la semántica y la sintaxis del lenguaje de acción, de segundo la interfaz entre el modelo y el alto nivel. Nótese en la figura 1.2 que la primera parte implica la definición de un lenguaje y la segunda parte implica la implementación de este lenguaje en una arquitectura robótica.

La validación experimental se hará a través de simulaciones. La formalización del lenguaje de acción y su implementación será comprobada usando instrucciones simples las cuales deben ser mapeadas correctamente y entendidas por el modelo de objetos. El analizador léxico debe poder extraer el sujeto, verbo y predicado de la instrucción simple. Por otro lado, la base de datos de posiciones deberá poder proporcionar las coordenadas espaciales correspondientes a la posición. Por último, el modelo de objetos debe generar los comandos de movimiento de los actuadores.

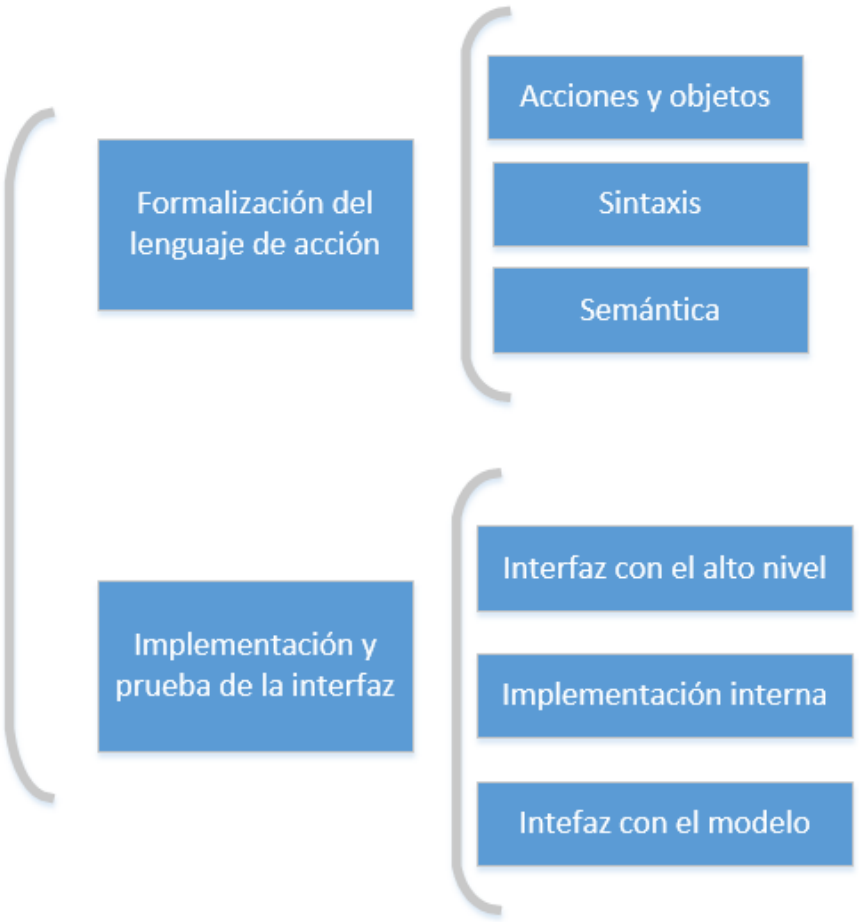

Figura 1.2: Metodología.

1.5. Objetivos

1.5.1. Objetivo general

Diseñar e implementar una interfaz entre el sistema de planeamiento simbólico y un modelo de objetos a partir de un lenguaje de acción orientado a manipulación de objetos.

1.5.2. Objetivos específicos

- Especificar formalmente la sintaxis y la semántica del lenguaje de acción orientado a manipulación de objetos.
- Determinar la interfaz de comunicación con la capa de planeamiento simbólico y el modelo de objetos.
- Implementar el sistema de mapeo entre el sistema de alto nivel y el modelo de objetos para un conjunto amplio de posibles acciones.
- Validar el sistema de mapeo con las otras capas de la arquitectura robótica a través de simulaciones.

1.6. Solución propuesta

Para solucionar el problema de como interconectar un sistema de alto nivel con los controladores del robot, en esta tesis se propone un **lenguaje de acción** especialmente diseñado para la manipulación de objetos. Una oración en este lenguaje de acción contendría elementos como: verbo, adverbio, objeto directo y predicado, que pueden servir como una instrucción que el robot debe ejecutar. A partir de este lenguaje de acción se podría hacer un **mapa asociativo**, donde, por ejemplo, se podría asociar: el verbo con el modelo y el controlador, el objeto directo con las propiedades del objeto y el predicado con el estado final deseado (Véase figura 1.3). El lenguaje de acción junto al mapa asociativo es una posible solución para interconectar ambas

capas (Guerra-Filho y Aloimonos, 2007).

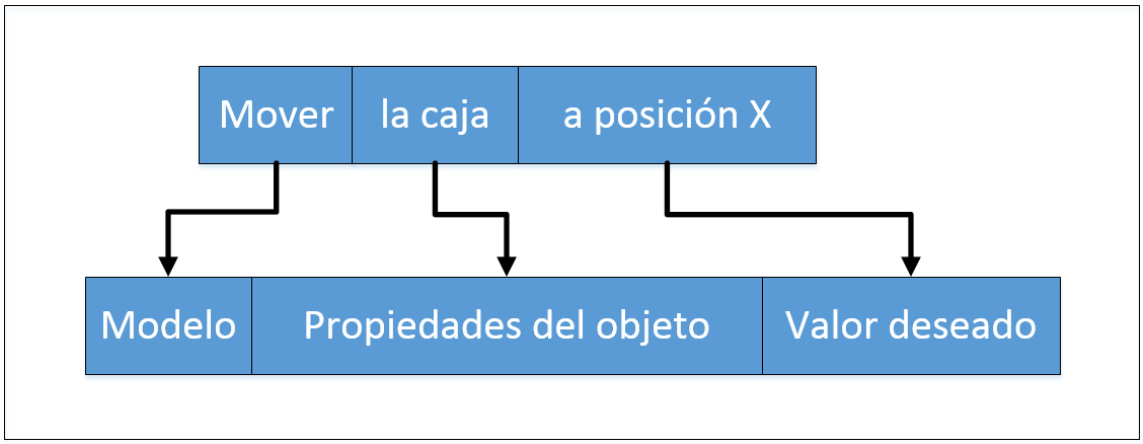

Figura 1.3: Mapeo de la instrucción en lenguaje de acción.

La forma de construir este mapa asociativo podría ser: 1) mediante la ayuda de un experto en la tarea a realizar y otro experto en las capacidades del robot, este tipo de asociatividad sería construida por humanos utilizando conocimiento experto previo; 2) utilizando "aprendizaje de máquina" donde a través de experimentos o ejemplos con seres humanos reales se enseña al robot a asociar elementos particulares de la oración con efectos o estados finales de los objetos, por citar un ejemplo, la asociatividad entre la frase "al frente" y las coordenadas espaciales correspondientes a la posición final de un objeto (Sugiura y Iwahashi, 2007).

El sistema que se va diseñar e implementar también necesita una **biblioteca de objetos** que es seleccionada y habilitada de acuerdo a las instrucciones en lenguaje de acción. La selección del objeto permite escoger el conjunto de propiedades o datos relativos al objeto, que serán necesarios para poder operar el modelo de objetos y el correcto funcionamiento de los controladores (Véase figura 1.6). Algunos

ejemplos de propiedades son: el coeficiente de fricción, las dimensiones físicas o la masa del objeto (Ruiz y cols., 2011).

La figura 1.4 muestra un diagrama del sistema que se va implementar. La primera parte, denominada "Sistema de razonamiento, planeamiento y conocimiento", traduce un lenguaje de alto nivel o un lenguaje natural, a un lenguaje de acción. A continuación el sistema de mapeo asocia esa instrucción generada por el alto nivel con las entradas del modelo de objetos. Por último, el"modelo de objetos" genera los comandos y las coordenadas de los movimientos a ser ejecutados por el robot.

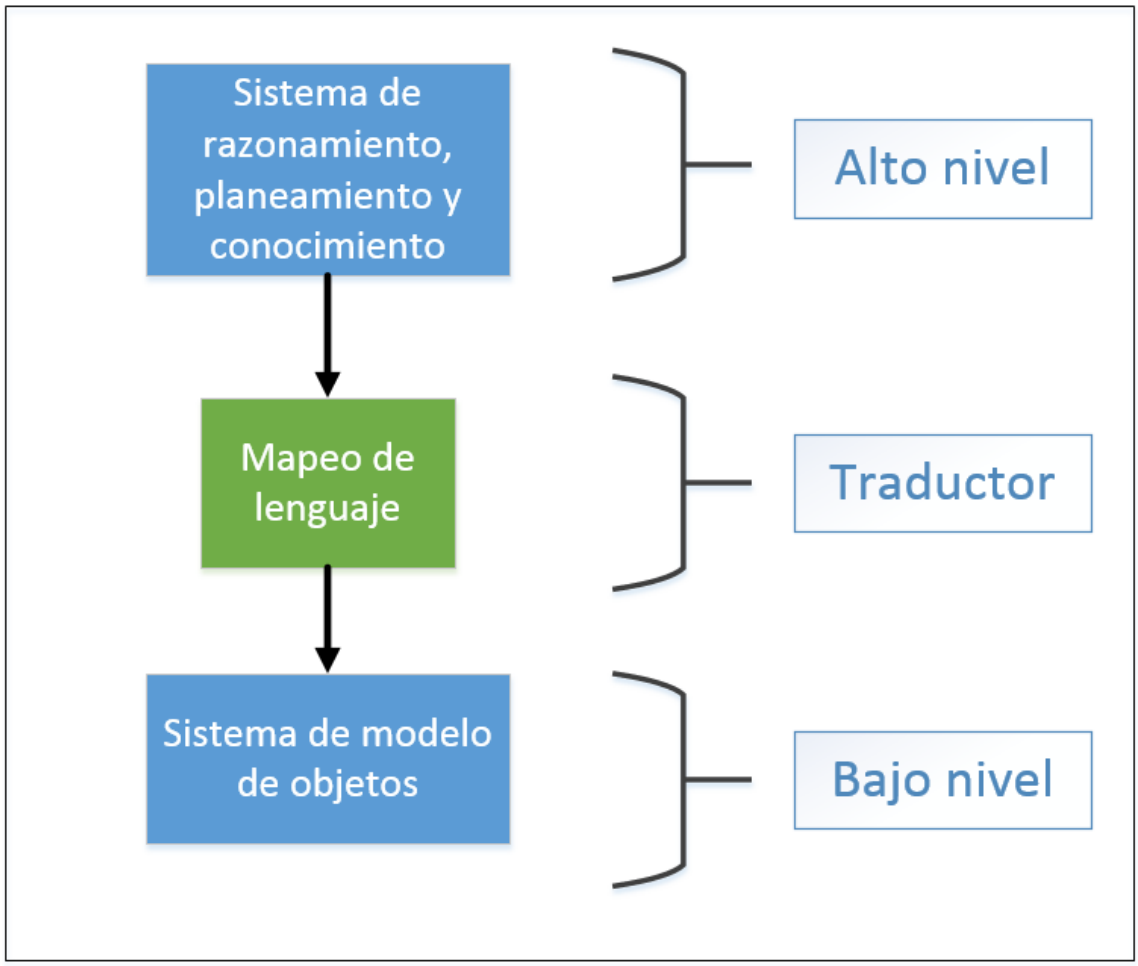

Figura 1.4: Diagrama de bloques del sistema.

La figura 1.5 muestra el proceso de traducción de la oración "Sírvame un vaso de leche". El sistema de alto nivel traduce la instrucción en lenguaje natural a un conjunto de instrucciones en lenguaje de acción. El sistema de razonamiento también determina las acciones que minimizan la cantidad de recursos necesarios para cumplir el objetivo. El componente de mapeo selecciona el modelo del objeto, las propiedades, la posición y el valor deseado, de cada instrucción del lenguaje de acción. Por último, el sistema de modelo de objetos comanda el sistema de control de la articulación robótica a partir de las instrucciones

proporcionadas.

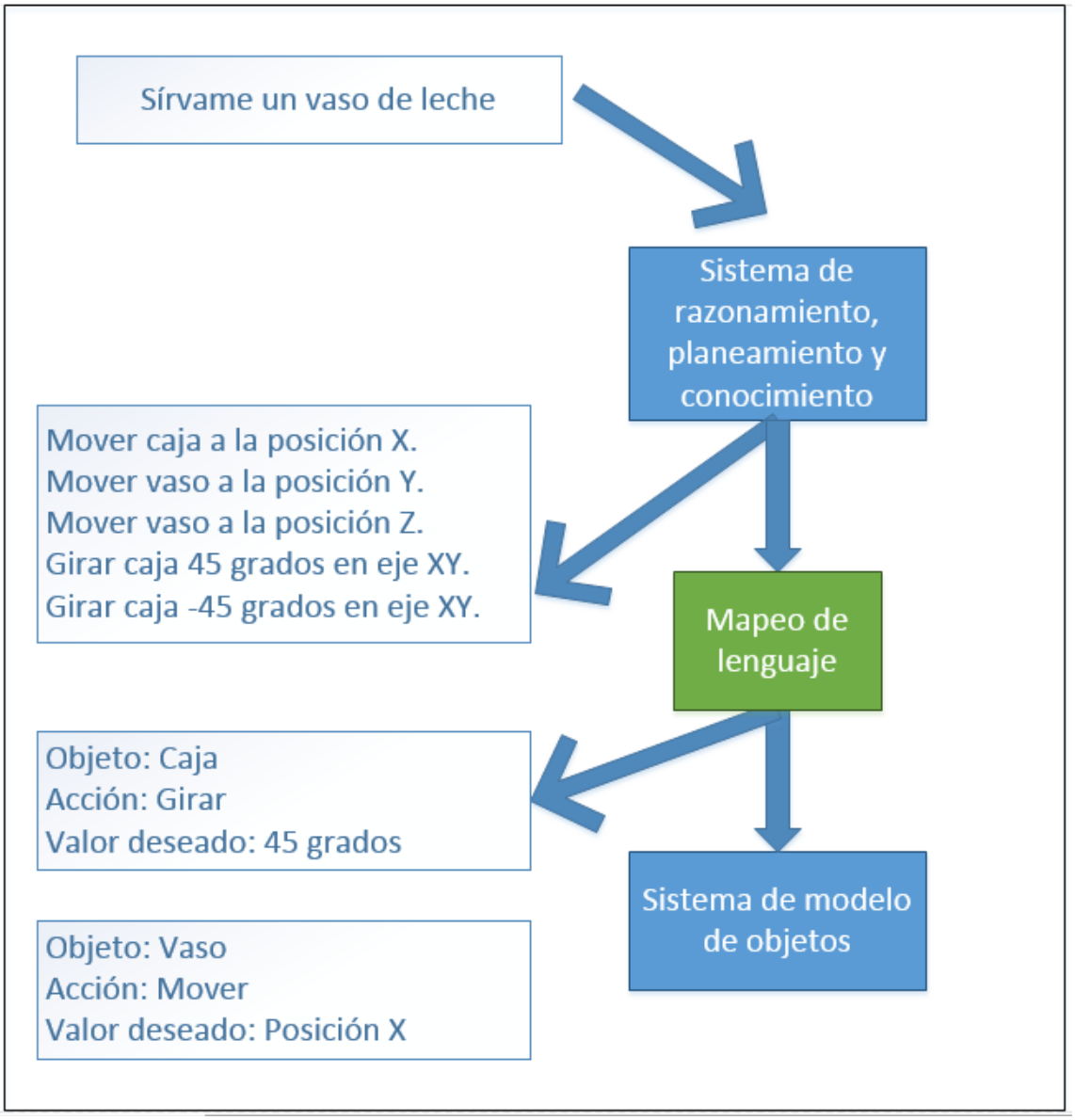

Figura 1.5: Ejemplo de mapeo de una instrucción en lenguaje natural.

Las instrucciones o comandos en lenguaje de acción son generados por el sistema de alto nivel. Estas instrucciones deben ser lo más reducidas y simples posibles para ser ejecutadas por el modelo de objetos. La forma es que esta instrucción simple se acopla al modelo de objetos es el principal tema de investigación de este proyecto, pero se parte

de que un nuevo lenguaje de acción, mapas o tablas asociativas y una librería de objetos es una solución factible a este problema.

La figura 1.6 describe el proceso de acoplar la instrucción en lenguaje de acción al modelo de objetos. El sistema de alto nivel produce una instrucción, la cual es mapeada en: selección de la posición (valor deaseado) y selección del objeto (acción y objeto). Las propiedades del objeto son seleccionadas una vez que se ha identificado el objeto. Nótese que se ha agregado una base de datos de posiciones al sistema para almacenar la posición de los objetos en el escenario de prueba y proporcionar los valores numéricos precisos a los objetos del sistema. El sistema posee un **analizador léxico**, el cual se encarga de analizar cada instrucción generada por el sistema de alto nivel y separarla en las partes correspondientes.

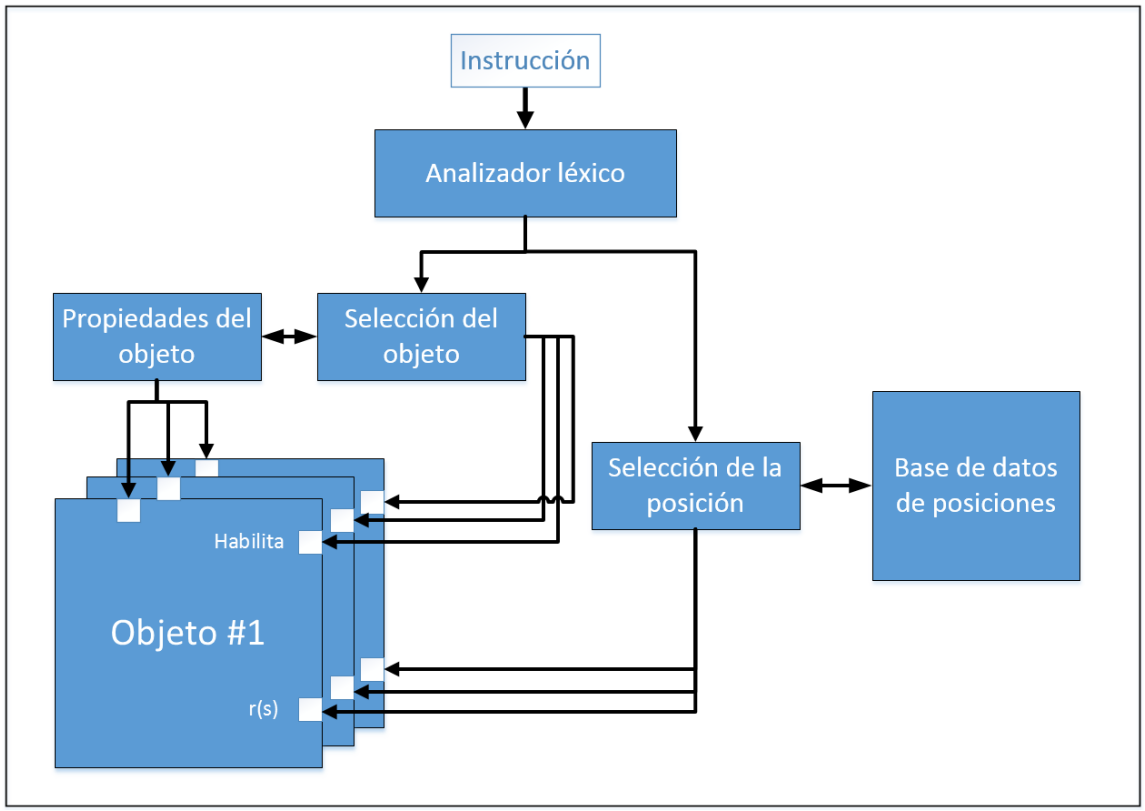

Figura 1.6: Diagrama de bloques del sistema de acople de la instrucción en lenguaje de acción con las entradas del modelo de objetos.

1.7. Aportes

En esta tesis se describen las siguientes contribuciones:

- Formalización de la semántica y sintaxis de un nuevo lenguaje de acción orientado a manipulación de objetos para dos escenarios: una cocina y una mesa de trabajo colaborativa (Suárez y Ruiz, 2018).
- Una arquitectura de software que me permita mapear oraciones del lenguaje de acción a parametrizaciones del controlador del sistema de objetos (Suárez y Ruiz, 2019).

- En una arquitectura robótica, una nueva capa de interfaz que permita cerrar la brecha entre los sistemas de planeamiento simbólico y la capa de control del robot.
- Una base de datos asociativa para ayudar a mapear las instrucciones simples y obtener las parametrizaciones que serán usadas por el controlador.

1.8. Guía del documento

En el capítulo 1 se presenta la propuesta de investigación correspondiente a esta investigación, lo cual incluye: el problema, la hipótesis, la justificación, la metodología, los objetivos, la solución propuesta y los aportes realizados.

La metodología se divide en dos partes: la primera parte corresponde a la generación de un lenguaje de acción; la segunda parte es el diseño y desarrollo del sistema de mapeo que se pretende utilizar como capa de interfaz en la arquitectura robótica.

En el capítulo 2 se presentan los antecedentes y el marco teórico, en el cual se abordan los aspectos más relevantes, como lo son: arquitectura robótica, modelos de objetos, lenguaje de acción, modelos de *affordances*, entre otros.

En el capítulo 3 se presenta el diseño general y detallado del sistema a implementar. Se presenta una descripción funcional y matemática

de cada uno de los bloques que componen el sistema, así como la especificaciones del lenguaje de acción. Por último, se detallan dos casos de uso a través de dos tareas simples: “Hágame un desayuno” y “Hágame una ensalada”.

En el capítulo 4, se diseñan los experimentos, lo cual incluye los experimentos para generar el lenguaje de acción; así como el diseño, desarrollo y prueba del software correspondiente a la capa de interfaz o sistema de mapeo.

En el capítulo 5 se muestran los resultados de los experimentos y se procede con la discusión de los mismos. Por último, en el capítulo 6 se muestran las conclusiones y las recomendaciones de este trabajo de investigación.

Antecedentes y marco teórico

Los robots, para que puedan trabajar apropiadamente con seres humanos, deben poder entender instrucciones, reaccionar a estímulos, conocer el entorno y anticipar su comportamiento. Sin embargo, en la actualidad, los robots se encuentran muy lejos de esta meta, esto se debe a las siguientes razones (Ruiz, 2014) (Cousins, 2011):

- La capacidad de un robot para detectar y reaccionar ante situaciones inesperadas del entorno es limitada. Por ejemplo, un robot programado para jugar fútbol podría no reaccionar adecuadamente ante la presencia de otro balón en el campo.
- La interacción robot-humano es un problema abierto. La interacción se refiere a la comunicación de acciones a ser ejecutadas por el robot y a la transferencia de conocimiento entre un humano y un robot. Esta comunicación se debe dar a través de un lenguaje de alto nivel.

El problema de comunicación entre un ser humano y un robot depende mucho de como el sistema de alto nivel entiende las instrucciones en lenguaje natural y las comunica al sistema de control. Existen referencias que apuntan a que agregar una capa adicional en una arquitectura robótica ayudaría a reducir la brecha entre la mente y el cuerpo del robot (Domingos, 2007) (Domingos y Lowd, 2009).

2.1. *Affordances*

La manipulación de objetos es una de las habilidades necesarias que debe desempeñar un robot. Los objetos se pueden ver como unidades que deben que ser entendidas y controladas (Ruiz, 2014) (Krivic, Ugur, y Piater, 2016). El concepto de ver un objeto como algo que tiene una función que depende de quién lo está manipulando se llama *affordance*. Este concepto tiene su origen en el diseño de productos y sugiere que la percepción del que manipula el objeto es la que determina su función (Norman, 2002).

El concepto *affordance* es muy útil para la manipulación de objetos y es medular dentro de la concepción de los modelos de objetos. La respuesta de un objeto está determinada por el modelo, a través del conjunto de fuerzas y torques aplicados. Por ejemplo, una lata de atún que se abre (acción especificada de acuerdo a la función que otorga el usuario), desde el punto de vista del robot, es un conjunto de fuerzas aplicadas en determinados puntos por una articulación robótica (Ruiz, 2014).

2.2. Planeamiento simbólico

La ejecución de tareas de la vida diaria, por parte de un operador humano, requiere de una serie de habilidades muy complejas y extensas. Adicionalmente, se necesita de una base de datos acerca del

contexto de las tareas y de los objetos a ser manipulados. Los sistemas de planeamiento simbólico realizan esta función tomando una acción compleja y desmenuzándola en tareas más sencillas, las cuales deben ser lo más simples posibles para ser ejecutadas por los controladores del robot (Quack, Wörgötter, y Agostini, 2015).

La segmentación de acciones complejas a acciones más simples dentro de una arquitectura robótica es un tema que en la actualidad se encuentra en investigación (Ruiken, Liu, Takahashi, y Grupen, 2016) (Müller, Kirsch, y Beetz, 2007) (Gorges, Schmid, Osswald, y Worn, 2007). El problema no se limita a tomar tareas y subdividirlas, si no que las tareas que se obtienen como resultado deben tener un orden tal que optimice la tarea desde el punto de vista de algún principio. Por ejemplo, un robot, en lugar de transportar los ingredientes de una receta de cocina, uno por uno, puede decidir utilizar un contenedor para transportarlos (C. Zhang y Shah, 2016).

El artículo de Muller, Kirsch y Beetz denominado *Transformational Planning for Everyday Activity* ha abordado este problemática a través de TRANER (Transformational Planner for Everyday Activity) el cual es un sistema de planificación, donde tareas complejas son divididas en tareas más sencillas. Esta investigación en particular tiene dos metas: obtener una biblioteca general para actividades diarias y un conjunto de reglas que implementan optimizaciones y revisiones de planes (Müller y cols., 2007).

El artículo de Tenorth y Beetz denominado *Priming Transfor-*

mational Planning with Observations of Human Activities plantea un enfoque diferente en la planificación de actividades donde el robot observa el comportamiento humano para encontrar el plan en su librería que mejor se ajusta a la forma en que el ser humano ejecuta la tarea. Esta investigación es un paso hacia adelante para que los robots puedan actuar con naturalidad en ambientes humanos (Tenorth y Beetz, 2010).

2.3. Lenguaje de acción

La parte que esta abordada pobremente es como estas instrucciones se acoplan a los controladores. En esa dirección el proyecto de Ruiz trabajó en la implementación de modelos matemáticos de los objetos para describir como se comportan (predición) y como controlarlos (Ruiz, 2014) (Ruiz y cols., 2010) (Stulp y cols., 2009). Estos modelos de objetos permiten, en lugar de comandar los motores, comandar el objeto directamente. El beneficio inmediato de comandar el objeto directamente es su cercanía con la forma en que expresan las tareas en una oración.

La salida del sistema de planeamiento simbólico son instrucciones en leguaje de acción. Algunos ejemplos de lenguajes usados en robótica son: PROLOG (Sterling y Shapiro, 1986), GOLOG (Levesque y cols., 1997), PREGO (Belle, 2014) y YAGI (Ferrein y cols., 2016). Los lenguajes citados están diseñados para realizar procesamiento lógico y razonamiento, con el fin de ser usados en la capa de planeamiento

simbólico, por lo que no están orientados a tareas de manipulación de objetos. Adicionalmente, los lenguajes de acción orientados a manipulación de objetos carecen de la generalidad suficiente para ser utilizados en tareas de propósito general. Este es el caso de un lenguaje de acción diseñado por Hopcroft exclusivamente para atar cordones de zapato (Hopcroft, Kearney, y Krafft, 1991).

2.4. Arquitectura robótica

El término "arquitectura robótica" puede referirse a dos conceptos diferentes: la estructura y el estilo. La **estructura** se refiere a como el sistema se divide en subsistemas y como estos interactúan, la cual es representada informalmente usando diagramas de caja y flechas. El **estilo** se refiere a los conceptos computacionales que sigue el sistema robótico en cuestión. Todos los sistemas robóticos poseen una estructura y un estilo, sin embargo, en un sistema dado es difícil determinar la arquitectura y el estilo, pues puede poseer varios estilos y las fronteras entre los diferentes subsistemas podrían no estar claras (Siciliano y Khatib, 2007) (Coste-Maniere y Simmons, 2000).

La primera arquitectura robótica comenzó con el robot **Shakey** en la Universidad de Stanford a finales de los años 60. La arquitectura de este primer robot se componía de tres elementos: sensado, planeamiento y ejecución. El subsistema de sensado traducía la imagen en un modelo interno. El subsistema de planeamiento tomaba este modelo y una meta para generar un plan, el cual luego era ejecutado por el ro-

bot. Este enfoque es denominado paradigma "sensar-planear-actuar" y su principal característica es que el plan es ejecutado sin usar directamente la información de sensado (Siciliano y Khatib, 2007).

En el año 2016 se diseñó y construyó una versión moderna de Shakey denominada "Shakey 2016", con el fin de evaluar los progresos en robótica. Los experimentos demostraron que, en las últimas décadas, ha habido un progreso importante en: planeamiento, navegación y detección de objetos. En la actualidad, el diseño y puesta en marcha de un robot es más rápido y fluido debido, entre otras cosas, a la gran disponibilidad de código de fuentes abiertas. Sin embargo, estas primeras ideas todavían están vigentes en la mayoría de arquitecturas robóticas y por eso es valioso traerlas a colación (Speck, Dornhege, y Burgard, 2017).

La arquitectura robótica de tres capas: planeamiento, ejecución y control en tiempo real; ha sido ampliamente usada en los últimos años. La capa de planeamiento almacena metas y actividades a largo plazo. La capa de ejecución es responsable de traducir planes de alto nivel en comportamientos de bajo nivel. La capa de control es el nivel más bajo de la arquitectura robótica, pues conecta actuadores y sensores. Estas tres capas necesitan comunicarse e intercambiar datos entre ellas (Fang, Zhao, He, y Lin, 2013).

2.5. Controlar objetos

La principal tarea de este componente es predecir y controlar el comportamiento del objeto. Un modelo de objeto es una descripción matemática del comportamiento del objeto, es decir, un sistema dinámico donde las entradas al sistema pueden ser cualquier cosa que el robot pueda transmitir al objeto y las salidas son cualquier cosa que el robot puede medir o inferir. El modelo de un objeto es un conocimiento de como se comporta un determinado objeto ante un determinado estímulo.

El modelo del objeto nos permite determinar que hará el objeto bajo ciertas condiciones conocidas. Por ejemplo, se puede determinar cómo se moverá el objeto al aplicar una fuerza en un determinado punto del mismo, para lo cual se deberían contemplar ciertos fenómenos físicos como: fricción, rotación y gravedad. El modelo del objeto será diferente si el objeto es diferente y deberá contemplar la información de interés en relación al problema que se intenta resolver.

Existe tres métodos diferentes que se utilizan para construir el modelo de objetos: aprendizaje de máquina, identificación del sistema e identificación de parámetros (Ruiz y cols., 2010). En caso de usar 'aprendizaje de máquina" el robot explora el objeto y realiza mediciones. Posteriormente, a través de un algoritmo de aprendizaje, se obtiene el modelo. En caso de usar "identificación de sistemas" el objeto se explora con diferentes entradas, se analizan las salidas y se obtiene

la función de transferencia. Este método es usado tradicionalmente en control automática para obtener el modelo de una planta o proceso.

Por último, el método de "identificación de parámetros" presenta ventajas importantes, esto debido a que un modelo bien realizado puede ser aplicado a una gran cantidad de objetos y, aún así, ser muy compacto. Este método primero hace uso de las ecuaciones obtenidas de la física y mecánica de los cuerpos, para luego identificar los parámetros del modelo obtenido de forma experimental. Por ejemplo: Un cuerpo en movimiento sobre una superfice plana presenta un coeficiente de fricción dinámica, el modelo en este caso está dado por:

$$\vec{a} = \frac{\vec{F}_a}{m} - \mu_d m g$$

La respuesta del objeto es la aceleración $\vec{a}$ y la entrada es la fuerza aplicada $\vec{F}_a$. Nótese que obtieniendo el modelo aplicando las leyes de la mecánica clásica (método de identificación de parámetros) es posible predecir la respuesta del objeto ante una determinada fuerza aplicada. El cálculo de la respuesta necesita la determinación de dos dos magnitudes físicas: la masa del cuerpo y el coeficiente de fricción. Ambos se obtienen realizando mediciones, en el primer caso a través de una váscula, en el segundo a través de un experimento más elaborado.

En un escenario compuesto de múltiples objetos, se deberá contar con una bibioteca de objetos, es decir, un conjunto de modelos matemáticos que describen el comportamiento de los objetos. Los modelos de objetos pueden ser analíticos o consistir en descripciones algorítmi-

cas, físicas o espaciales del objeto. En la actualidad se está realizando amplia investigación para generar bibliotecas de objetos y ponerlas a disposición de la comunidad de investigadores en robótica (Calli y cols., 2015).

2.6. Planeamiento

El concepto de planeamiento se refiere al proceso de tomar una tarea y descomponerla en partes más compactas que satisfacen las restricciones y los objetivos establecidos. Es decir, se refiere al establecimiento de la secuencia de eventos que son necesarios para realizar una tarea compleja. Para ambientes estáticos es muy sencillo establecer un plan, pero en robótica se busca que el robot sea capaz de construir planes en ambientes dinámicos y complejos. El robot debe poder reconocer su entorno y modificar sus acciones cuando el ambiente cambia inesperadamente.

El planeamiento o planificación es un área de la inteligencia artificial que tiene como fin la producción de planes, lo cuales son pensados para ser usados por un robot u otra clase de sistema. Este área se caracteriza por tratar con problemas muy complejos que deber ser resueltos y optimizados en un espacio de búsqueda multidimensional.

En robótica, el planeamiento, realizado por el sistema de inteligencia artificial de una arquitectura robótica, puede tener un alto componente de optimización matemática, donde dadas las condiciones

y características de un modelo, se pretende determinar el plan que minimiza la cantidad de recursos utilizados. Existen una gran variedad de formas de expresar este problema un de ellas es conocido como un "Proceso de Decisión de Markov" (Siciliano y Khatib, 2008).

2.7. Objetos

El modelo de un objeto tiene parámetros, como pueden ser: tamaño, peso, volumen, fricción, temperatura, entre otros. Esto es conocido como propiedades del objeto. Por ejemplo, en el caso de una caja que es empujada por una articulación robótica, los siguientes parámetros son considerados en el artículo de Ruiz, Cheng y Beetz (Ruiz y cols., 2010):

- Dimensiones del objeto: El robot, en lugar de explorar el objeto, toma un modelo CAD del mismo.
- Peso: Es necesario que se conozca para calcular la fricción y el torque, los cuales son parte del modelo.
- Coeficiente de fricción: Este valor se obtiene experimentalmente al ir aumentando la fuerza entre la superficie y la caja.

2.8. Lenguajes

El lenguaje es el proceso cognitivo que usualmente se expresa a través del habla usando palabras con una gramática, la cual contiene

los siguientes aspectos: fonológica, semántica, sintáctica, pragmática y contextual. En el contexto de esta investigación, se pueden distinguir tres tipos de lenguajes: lenguaje natural, lenguaje de acción y comandos de control (Ferrein y cols., 2016) (Guerra-Filho y Aloimonos, 2007).

2.8.1. Lenguaje natural

El lenguaje natural se refiere a cualquier oración expresada por un ser humano, la cual usualmente consiste en tareas de largo plazo y de mayor complejidad, por ejemplo:

- Hágame un desayuno.
- Prepáreme la cena.
- Hágame un emparedado.

Existe un campo específico en las ciencias de la computación que se encarga de todo lo referente a las interacciones entre computadores y el lenguaje humano, este se denomina "procesamiento de lenguaje natural". Este campo, a su vez, se subdivide en diferentes campos, por ejemplo:

- Recuperación de la información
- Traducción automática
- Reconocimiento de voz

- Síntesis de habla

El procesamiento del lenguaje natural presenta una serie de dificultades que complica su procesamiento, entre ellas:

- Ambiguedad
- Errores
- Detección

La ambiguedad es fundamental en el campo de la robótica, el lenguaje natural es inherentemente ambiguo a diferentes niveles:

- Nivel léxico: Una misma palabra puede tener diferentes significados que se deducen a partir del contexto.
- Nivel referencial: En lenguaje humano a veces se hace referencia a elementos anteriores o posteriores.
- Nivel estructural: Es necesario entender y reconocer la semántica del lenguaje para entender una oración.
- Nivel pragmático: El lenguaje humano no siempre significa lo que se está diciendo, por ejemplo: el sarcasmo y la ironía.

Por tanto, el lenguaje natural, para ser utilizado en una arquitectura robótica, debe ser traducido a una representación interna que no presente ambiguedad.

2.8.2. Lenguaje de acción

Un lenguaje de acción está diseñado para especificar la transición de estados de un sistema y es usualmente utilizado en la capa de planeamiento simbólico de una arquitectura robótica para describir como las acciones modifican los estados del sistema. Los lenguajes de acción son mayormente utilizados para planeamiento de tareas, pero en menor medida se han utilizado para manipulación de objetos. (S. Zhang, Yang, Khandelwal, y Stone, 2015).

El lenguaje de acción más simple, denominado "A", es un buen punto de partida para entender cualquier lenguaje de acción. Este lenguaje se compone de cuatro elementos: *action names*, *fluent names*, *effect propositions* (E) y *value propositions* (Nabeshima, Inoue, y Haneda, 2000). El componente *effect proposition* describe una relación causal entre de una acción y su efecto. Este se expresa de la siguiente forma:

$$a \ causes \ \phi \ if \ \psi \tag{2.1}$$

donde a es el nombre de la acción, ϕ es el efecto de la acción y ψ es un proposición condicional que usa conectivas lógicas. El componente *value proposition* describe un hecho observado; lo cual se expresa de la

siguiente forma:

$$\phi \ \ after \ \ a_1; \ldots; a_m \quad (m \geq 0) \tag{2.2}$$

donde a_i es el nombre de una acción y ϕ es una conjunción de fluentes.

Este tipo de lenguajes podrían ser utilizados para manipulación puesto que al considerar tareas de manipulación de objetos, las acciones podrían ser verbos como: deslizar, empujar y girar. Algunas fluentes podrían corresponder a la posición final del objeto o algún estado relacionado con la posición final del mismo.

2.8.3. Comandos de control

Los comandos de control se refieren a cualquier lenguaje específico utilizado para operar el hardware del robot directamente, por ejemplo, dos instrucciones en el lenguaje utilizado en *KUKA Robot Language*® podrían ser las siguientes:

- LIN {X 17.3, Y 26.0, Z 55.0}
- PTP {X 12.3, Y 30.0, Z 50.0}

La instrucción **LIN** mueve el objeto o efector final a una velocidad predefinida a lo largo de una línea recta hasta la posición especificada.

El comando **PTP** mueve el objeto, pero a través de la ruta más rápida, hasta la posición especificada.

2.9. Modelos de *affordances*

Los modelos de *affordances* fueron introducidos para modelar la interacción de un robot con su ambiente. Estos consisten en modelos probabilísticos que se pueden construir para correlacionar tres conjuntos: los objetos del escenario, dados por $O = \{o_1, \ldots, o_n\}$, las acciones que puede realizar el robot, dadas por $A = \{a_1, \ldots, a_n\}$ y los estados finales de los objetos, dadas por $E = \{E_1, \ldots, E_n\}$ (Moldovan y Raedt, 2014).

Esta definición, establecida por Moldovan (Moldovan y Raedt, 2014), resulta muy conveniente, debido a que, como más adelante veremos, es igual a la forma en que se han establecido las tareas simples en este trabajo. Las tareas simples se han establecido como: Verbo + Objeto + Estado Final, por tanto, la relación entre un tarea simple y un modelo de *affordances* es directa.

La construcción de estos modelos resulta muy útil, pues se pueden utilizar para inferir algunos de estos tres elementos: Verbo, Objeto o Estado Final; cuando uno de ellos es desconocido. Por ejemplo, se podría inferir la acción que fue manipulada si el objeto y el estado final son conocidos, esto se denomina *action recognition*; de forma similar se podría intentar inferir el estado final a partir del objeto y la acción

realizada, lo que se denomina *effect prediction*.

En este trabajo usamos estos modelos con el objetivo de determinar si una tarea simple es viable o factible de ser ejecutada, en particular, nos enfocamos en la acción y el objeto, pero también hay *affordances* entre el objeto y el estado final, o bien, entre la acción y el estado final. Por ejemplo, la tarea simple "Encienda la caja de leche" no es ejecutable porque no hay *affordability* entre el verbo (Encender) y el objeto (Caja de leche). Es decir, la caja de leche no es un objeto que posea el *affordance* de poder ser encendido, por lo que la tarea no es viable y no puede ser ejecutada.

2.10. Modelos secuencia a secuencia

Los modelos secuencia a secuencia son modelos de aprendizaje de máquina que han probado alcanzar un rendimiento significativo en problemas, tales como: reconocimiento de voz, resumen de textos, reconocimiento visual de objetos y traducción automática. A grandes rasgos, un modelo secuencia a secuencia posee dos componentes: un codificador y un decodificador (Ver figura 2.1).

En esta investigación los modelos secuencia a secuencia se pretenden utilizar para la traducción automática de instrucciones en lenguaje natural a un lenguaje intermedio de robótica móvil denominado *Robot Control Language* (RCL). Este campo del procesamiento en lenguaje natural se denomina *neural machine translation* (NMT).

Tradicionalmente, la traducción entre un lenguaje y otro se ha hecho al dividir las oraciones en múltiples piezas y traduciéndolas frase por frase. Este proceso tiene la desventaja de que el significado de una oración está determinado por toda la oración y no por porciones o frases. NMT resuelve esta desventaja al ser capaz de captura una gran cantidad de dependencias.

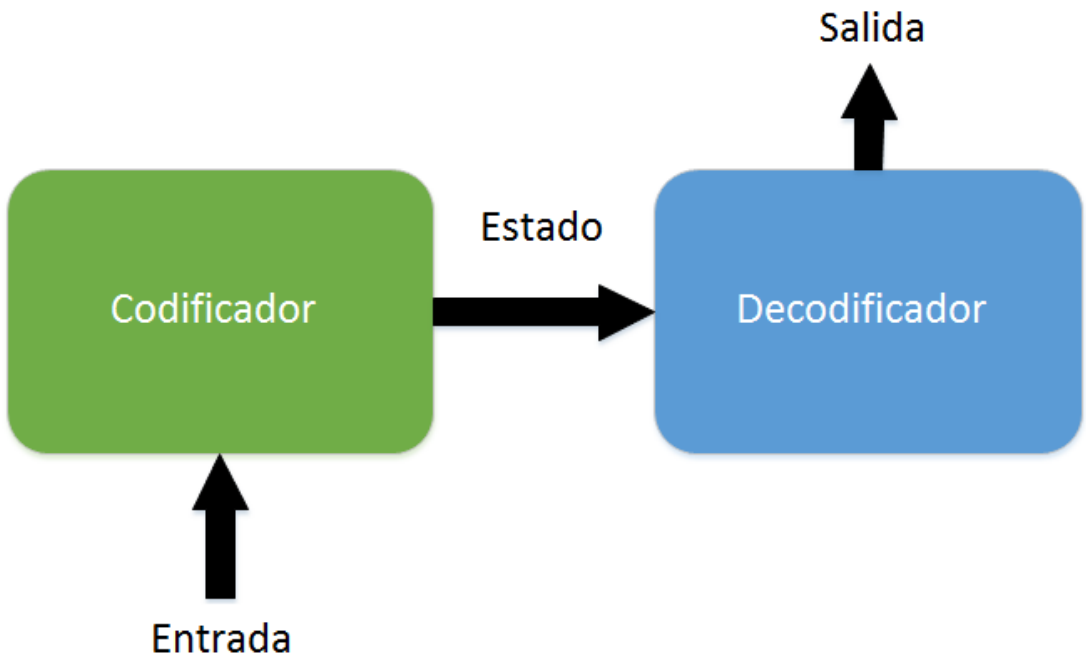

Figura 2.1: Arquitectura del modelo secuencia a secuencia que consiste en dos componentes principales: un codificador y un decodificador.

2.10.1. Red neuronal *LSTM*

Un tipo especial de red neuronal recurrente es *LSTM* (*Long short-term memory*). Un bloque *LSTM* común se compone de: una celda, una compuerta de entrada, una compuerta de salida y una compuerta

de olvido. Esto le permite a una red neuronal *LSTM* recordar valores, pero no únicamente el valor previo, si no que es capaz de recordar por mayores periodos de tiempo. En general, esta red neuronal se utiliza para clasificar, procesar y predecir series de tiempo.

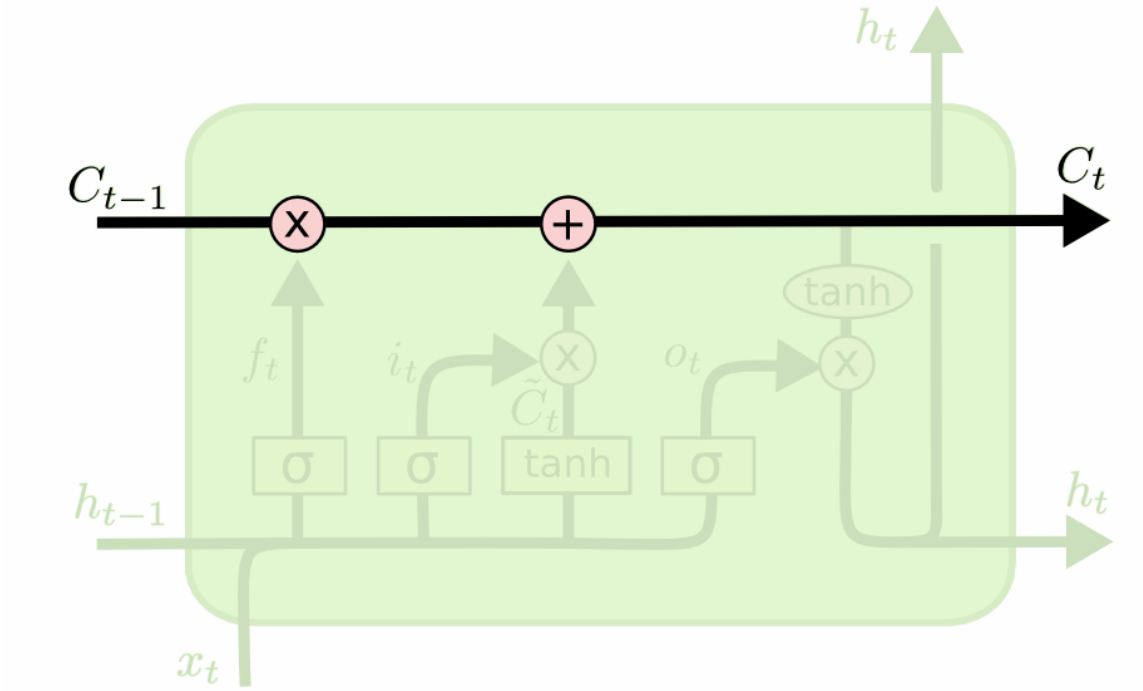

Figura 2.2: Estado de la celda *LSTM*.

La idea central detrás de *LSTM* es el estado de la celda, lo cual corresponde a la línea horizontal mostrada en la figura 2.2. La unidad puede remover o agregar información al estado de la celda, lo cual se regula a través de compuertas.

El primero elemento que se analizará es la compuerta de olvido, la cual se compone de una capa sigmoide y un operador de multiplicación. Esto permite multiplicar el estado por un número entre cero y uno, lo cual describe cuanto del estado se deja pasar.

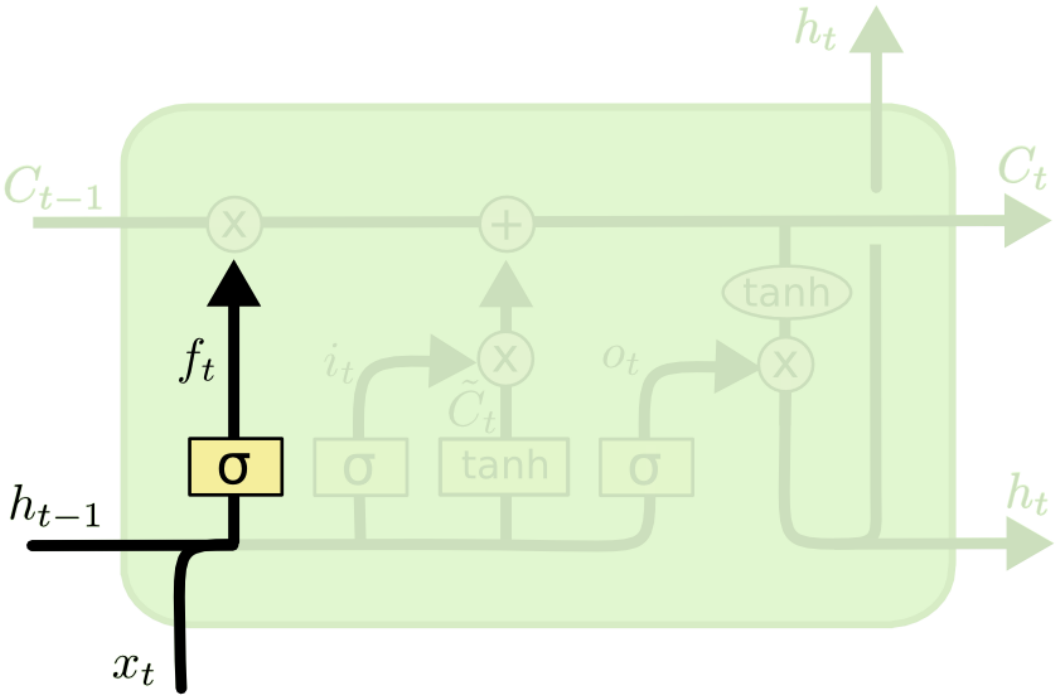

Figura 2.3: Compuerta de olvido en una celda *LSTM*

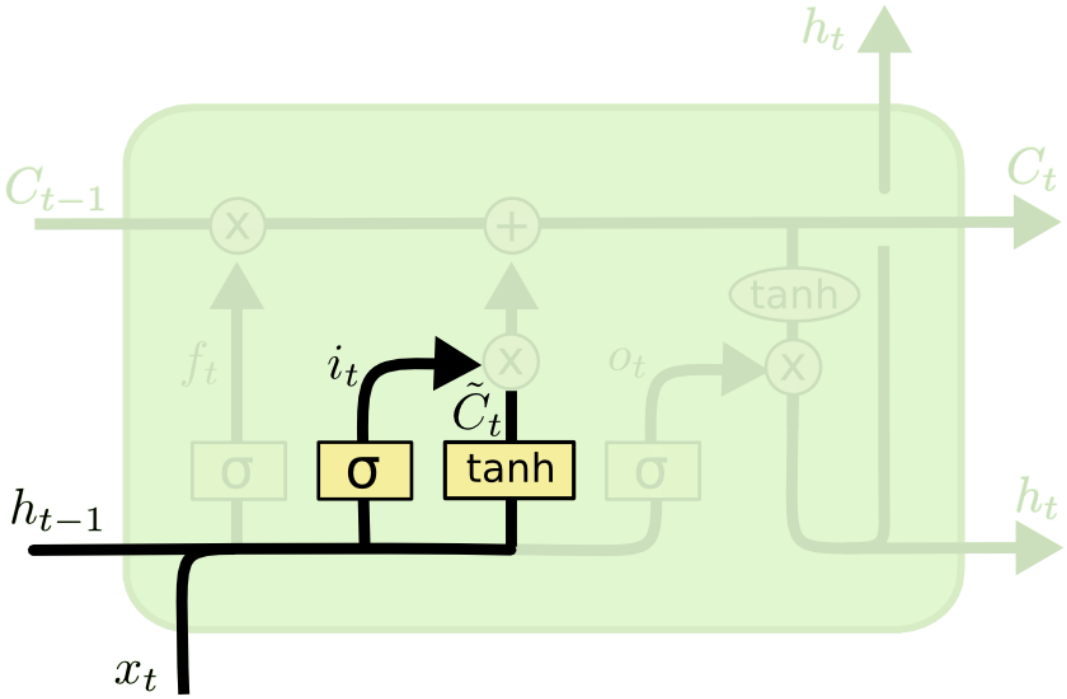

Figura 2.4: Compuerta de entrada de una celda *LSTM*

En el siguiente paso se decide que información será almacenada en el estado de la celda. Esta capa tiene dos partes: una capa sigmoide que es la compuerta de entrada y una capa tangente hiperbólica que genera un valor que es sumado al estado.

Por último, es necesario determinar la salida de la celda, la cual será una versión filtrada del estado de la celda. Primeramente una capa sigmoide decide que parte del estado irá a la salida, luego el estado de la celda va hacia una capa tangente hiperbólica y es multiplicada por el estado sigmoide, tal cual se observa en la figura 2.5.

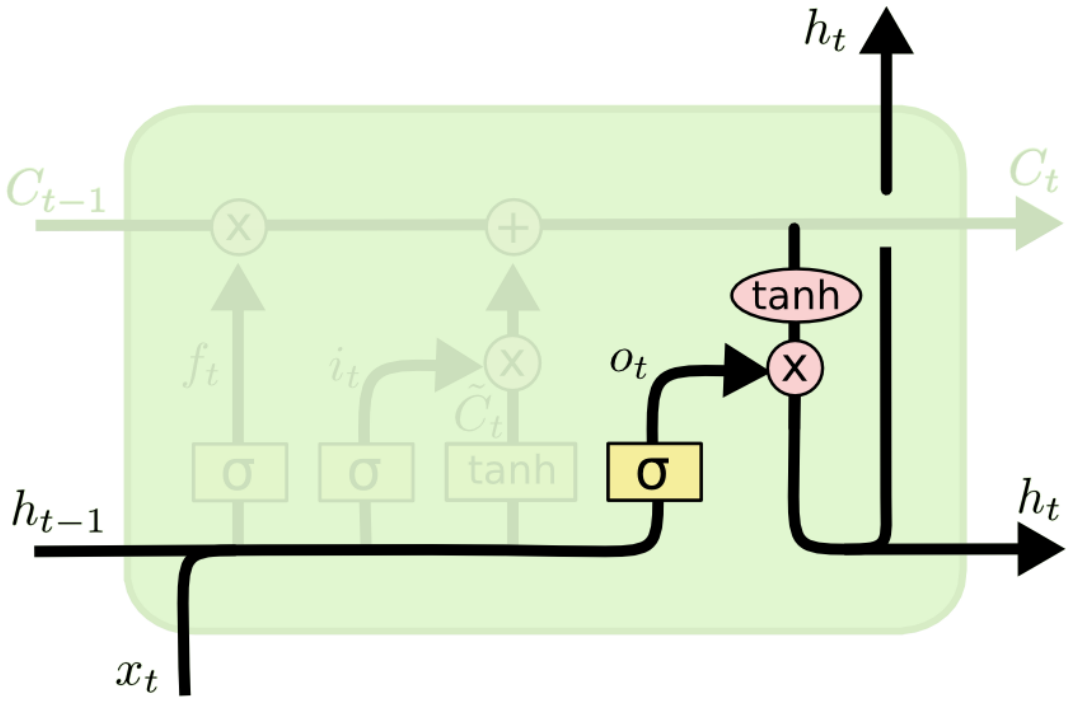

Figura 2.5: Compuerta de salida de una celda *LSTM*

Existen muchas variantes de la arquitectura de una celda *LSTM*, la que se acaba de describir es una celda estándar.

2.10.2. Traducción automática neuronal (NMT)

La celda *LSTM* es el elemento central de *NMT* debido a que es usada en ambos elementos principales: el codificador y el decodificador (Ver figura 2.6). El decodificador calcula una representación para cada oración fuente, a continuación, basado en esta representación, el decodificador genera una traducción palabra por palabra.

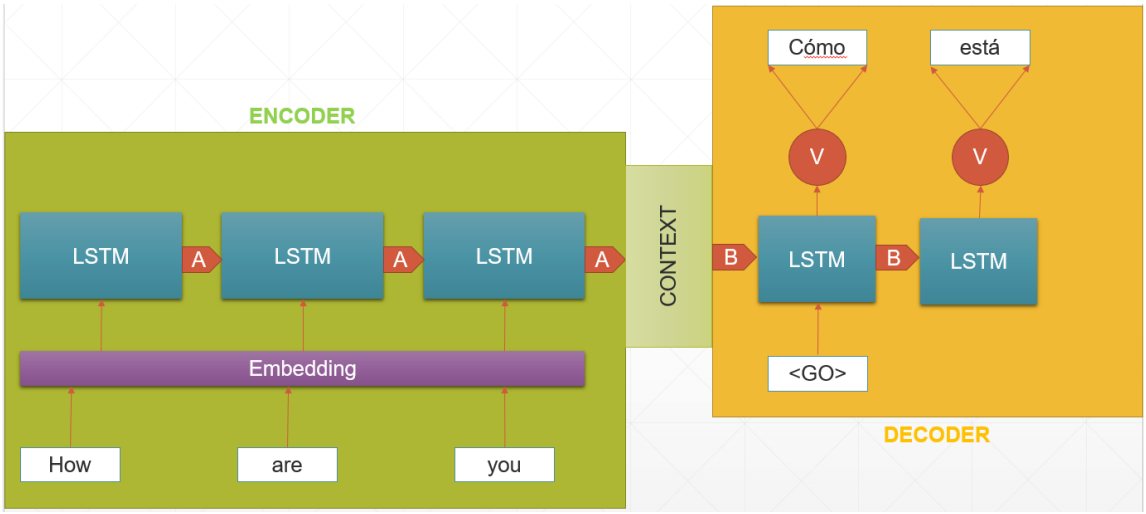

Figura 2.6: Arquitectura del modelo secuencia a secuencia que consiste en dos componentes principales: un codificador y un decodificador.

Arquitectura propuesta

En este capítulo se describen los elementos fundamentales del diseño del sistema: la arquitectura robótica, el diseño detallado del sistema de mapeo o capa de interfaz y las especificaciones del lenguaje de acción. Al final de esta sección también se describen dos casos de uso que muestran cómo la arquitectura puede resolver dos instrucciones de alto nivel.

3.1. Especificaciones generales del sistema

La arquitectura del sistema se muestra en la figura 3.1, la cual consiste de tres capas: la capa del alto nivel, la capa de interfaz y la capa de bajo nivel. La capa de alto nivel y la capa de bajo nivel no son el objeto de estudio de este proyecto de grado. Sin embargo, se considera importante presentar el estado del arte de estos dos sistemas.

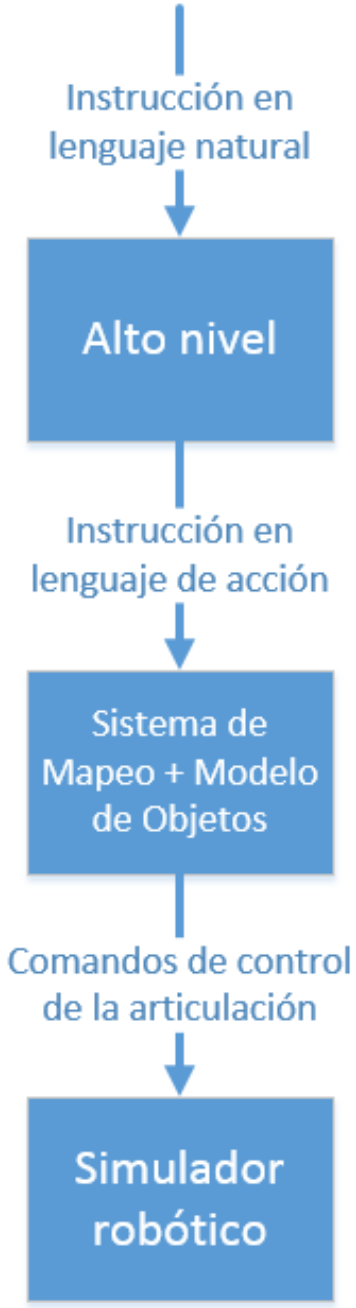

Figura 3.1: Arquitectura general del sistema a implementar compuesto por: la capa de alto nivel, el sistema de mapeo, el modelo de objetos y el simulador robótico.

En relación a la capa del alto nivel, la investigación de Thenmonzhi (Thenmozhi, Seshathiri, Revanth, y Ruban, 2017) se centra en tomar oraciones en lenguaje natural y generar los tokens respectivos donde se puede identificar: el verbo, la acción, las posiciones y los objetos. Luego, a través de un simulador, ejecuta estas instrucciones. El lenguaje usado está referido al escenario y la articulación robótica mostrada en la figura 3.2.

La investigación de Thenmonzhi se centra en la traducción de lenguaje natural a comandos de control y deja de lado cualquier referencia a la capa de control, pues los comandos son ejecutados directamente por un simulador. Como resultado, el robot carece de: autonomía, control automático y cognición. Esta debilidad está presente en la mayoría de investigaciones referentes a la capa de planeamiento simbólico de una arquitectura robótica, que es lo que se pretende abordar en este trabajo.

En las actualidad, las arquitecturas robóticas en general siguen un esquema de tres capas, las cuales se denominan: planeamiento, ejecución y control. La capa de control está conectada a sensores y actuadores. La capa de ejecución es responsable de seleccionar los comportamientos del robot. Por último, la capa de planeamiento asegura que el robot alcance los objetivos minimizando las variables objetivo (tiempo, distancia, energía, entre otros) (Siciliano y Khatib, 2008).

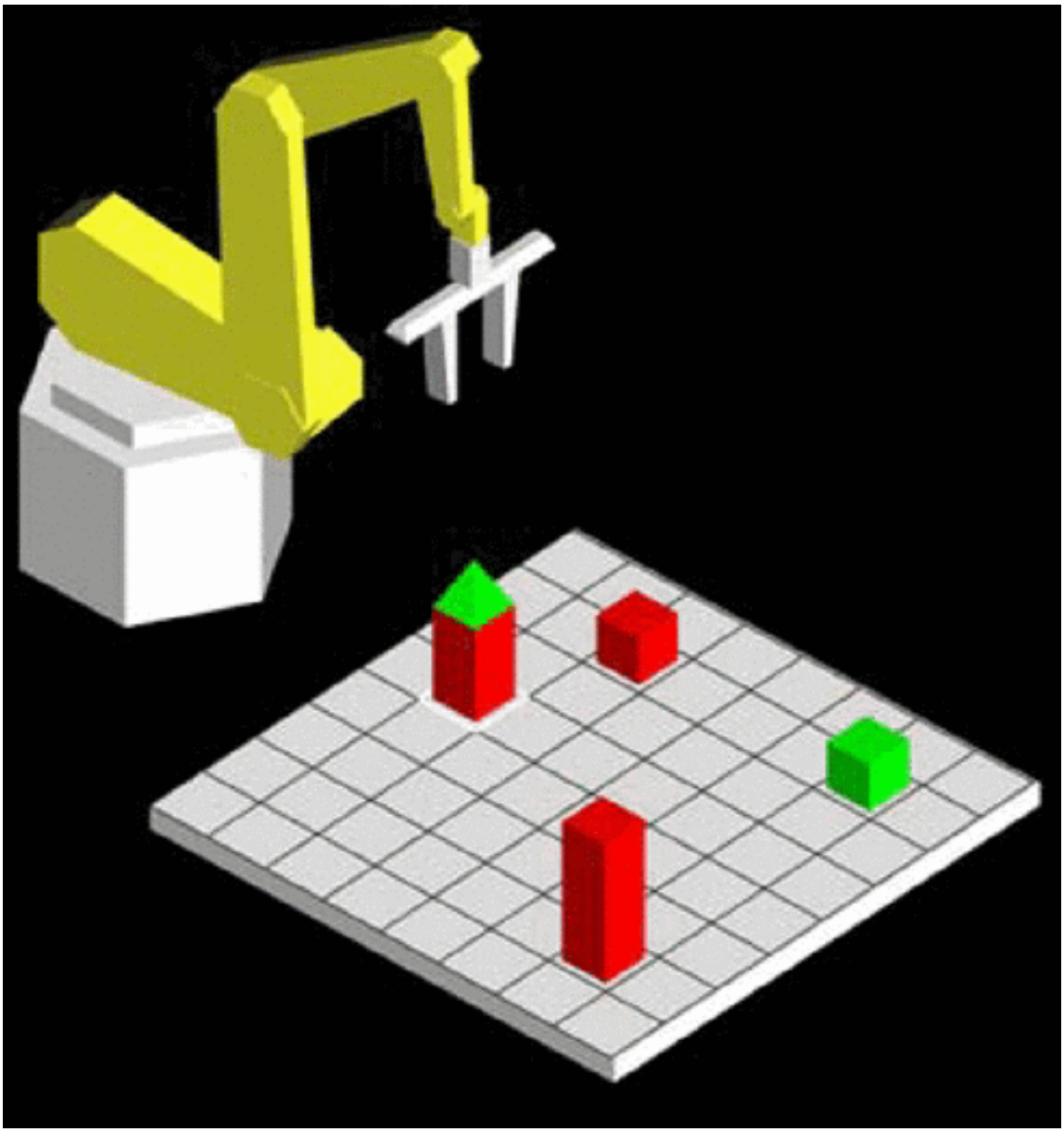

Figura 3.2: Escenario y articulación robótica utilizada por Thenmonzhi

En lugar de esta estructura de tres capas, en esta investigación se trabajan con dos capas: una capa de alto nivel y una capa de bajo nivel. Adicionalmente, para resolver el problema en cuestión, se ha colocado una capa adicional de interfaz. La capa de interfaz pretende hacer la conexión entre la capa de alto nivel y el sistema de modelos de objetos de la capa de bajo nivel. La capa de alto nivel procesa tareas de largo

plazo que le son presentadas en un lenguaje más natural; los modelos de objetos son modelos matemáticos que permite controlar y predecir el comportamiento de los objetos.

La arquitectura de tres capas del estado del arte resulta poco explicativa y descriptiva para representar el acople entre un modelo de objetos y una capa de alto nivel, por lo que se ha preferido un enfoque más simple para representar el sistema. El elemento fundamental que acopla ambas capas es el lenguaje de acción que genera el sistema del alto nivel, para luego ser procesado por la capa de interfaz, que a su vez indica al modelo de objetos que acciones realizar. Por último se tendrán los comandos de control de la articulación robótica.

3.2. Descripción detallada del sistema de mapeo

El sistema de mapeo (Vease figura 3.3), también denominado capa de interfaz, contiene cinco módulos: la unidad de *affordances*, la unidad de mapeo, la unidad de determinación de la referencia, la unidad de propiedades de los objetos, y la unidad de análisis de factibilidad. Estos cinco módulos constituyen el sistema de mapeo, el cual será implementado y validado experimentalmente en este trabajo de grado.

La unidad de *affordances*, a partir del verbo y el objecto, calcula una métrica denominada *affordability*. La unidad de mapeo, basado en el objeto y el verbo, determina la lista de controladores que pueden ser usados para ejecutar la acción. La unidad de determinación de la

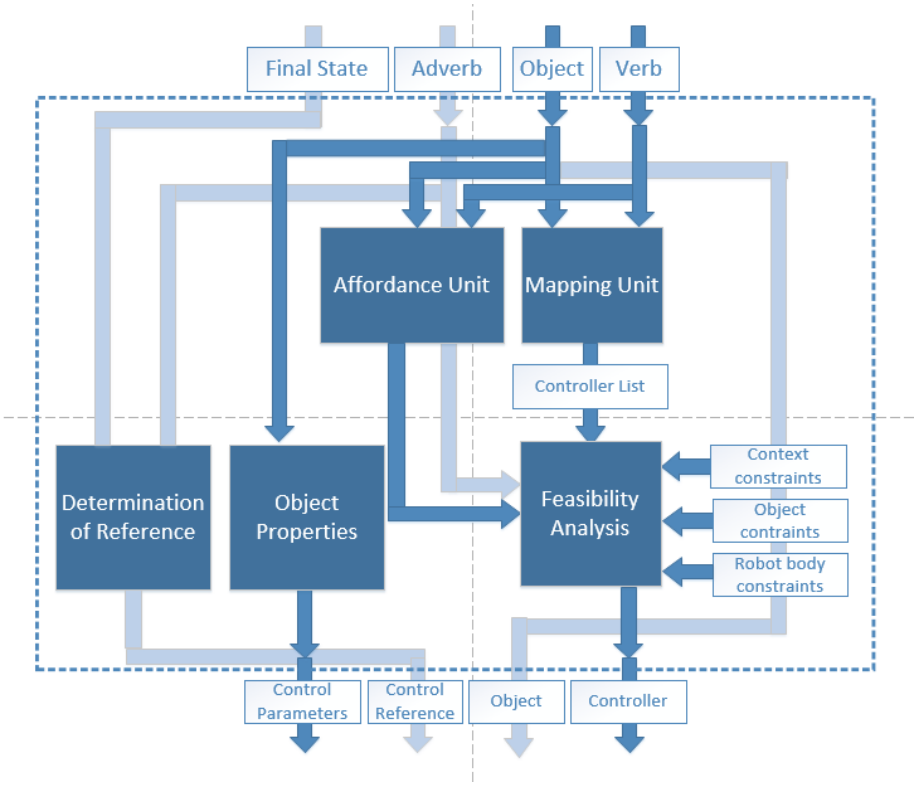

Figura 3.3: Sistema de mapeo, el cual contiene cinco módulos: *affordances*, mapeo, determinación de la referencia, propiedades de los objetos y análisis de factibilidad.

referencia calcula el valor deseado o de referencia a ser utilizado por los controladores de objetos. El módulo de propiedades de objetos es una base de datos que contiene las propiedades físicas de los objetos que son necesarias por los controladores. El módulo de análisis de factibilidad determina cuáles controladores son más viables para ejecutar la acción simple.

La capa de alto nivel genera tareas simples, en la forma: Objeto + Verbo + Estado Final. Estas instrucciones van al sistema de mapeo donde genera la siguientes salidas: parámetros de control, referencia de control, objeto y controlador. Estas señales pueden ir ahora a la capa de bajo nivel, donde se puede realizar el control y la predicción

a nivel físico de los objetos. Nótese en la figura 3.4 que el sistema de modelo de objetos requiere conocer el controlador a utilizar, el valor de referencia y las propiedades del objeto.

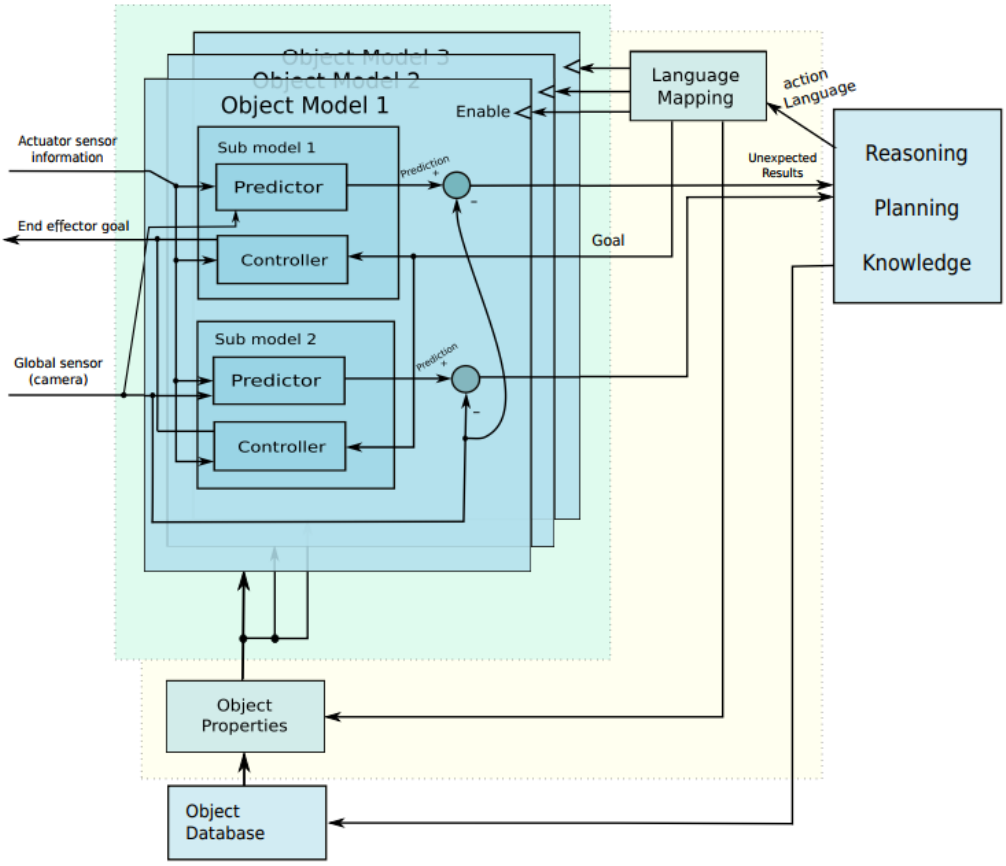

Figura 3.4: Sistema de modelo de objetos propuesto por Ruiz. El componente de razonamiento y planeamiento corresponde a la capa de alto nivel descrita en este documento.

A contiene se describen y analizan cada una de las unidades que componene el sistema de mapeo.

3.3. Unidad de *affordances*

Esta unidad, a partir del verbo y el objeto, determina si hay *affordability*. Por ejemplo, si el objeto es "estufa" y el verbo es "encender", hay *affordability* porque la estufa puede ser encendida. Por otro lado, si el objeto es "vaso" y el verbo es "encender", no hay *affordability*, esto porque el vaso no es un objeto proclive a ser encendido (la mayoría de las veces).

Para determinar el *affordability* entre el verbo y el objeto, calculamos la probabilidad condicional de tener un objeto dado que se ha tenido un verbo, para calcular esta probabilidad, usamos la ecuación (3.1) (Moldovan y Raedt, 2014) (Moldovan y cols., 2012):

$$P(objeto|verbo) = \frac{c(verbo, objeto)}{c(verbo)} \tag{3.1}$$

En la ecuación 3.1, la probabilidad condicional se calcula dividiendo la cantidad de ocurrencias del bigrama $Verbo + Objeto$ entre la cantidad de ocurrencias del unigrama $Objeto$. La cantidad de ocurrencias de cada uno se obtiene a partir de un corpus de tareas simples, que se denomina corpus de entrenamiento.

3.3.1. Matriz de *affordances*

En un escenario dado tenemos múltiples objetos $O = \{o_1, \ldots, o_n\}$, diferentes acciones de manipulación $A = \{a_1, \ldots, a_n\}$ y un conjunto

importante de estados finales $E = \{E_1, \ldots, E_n\}$. Al usar esta notación, podemos reescribir la ecuación 3.2 de la siguiente forma:

$$P(o_j|a_i) = \frac{c(a_i, o_j)}{c(a_i)} \tag{3.2}$$

Es posible definir una matriz de *affordances*, dada por A que refleje todas la combinaciones posibles de acciones y objetos en un escenario dado, es decir:

$$\mathbf{A} = \mathbf{P}(o_j|a_i) = \begin{pmatrix} P(o_0|a_0) & P(o_1|a_0) & \cdots & P(o_M|a_0) \\ P(o_0|a_1) & P(o_1|a_1) & \cdots & P(o_M|a_1) \\ \vdots & \vdots & \ddots & \vdots \\ P(o_0|a_L) & P(o_1|a_L) & \cdots & P(o_M|a_L) \end{pmatrix} \tag{3.3}$$

Al construir esta matriz se ha considerado que el lenguaje de acción, en el escenario dado, se compone de L verbos o acciones y M objetos. Es posible volver a escribir esta matriz de la siguiente forma:

$$\mathbf{A} = \begin{pmatrix} \frac{c(a_0,o_0)}{c(a_0)} & \frac{c(a_0,o_1)}{c(a_0)} & \cdots & \frac{c(a_0,o_M)}{c(a_0)} \\ \frac{c(a_1,o_0)}{c(a_1)} & \frac{c(a_1,o_1)}{c(a_1)} & \cdots & \frac{c(a_1,o_M)}{c(a_1)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{c(a_L,o_0)}{c(a_L)} & \frac{c(a_L,o_1)}{c(a_L)} & \cdots & \frac{c(a_L,o_M)}{c(a_M)} \end{pmatrix} \tag{3.4}$$

A partir de este valor numérico se busca estimar si hay *affordability* entre el verbo y el objeto. Si la probabilidad es cero y el corpus es lo

suficientemente grande, es posible concluir que no hay *affordability* entre el verbo y el objeto. Por tanto, la tarea simple no podrá ser ejecutada.

Los valores de la matriz **A** que tienen un valor relativamente bajo están asociados a tareas simples con baja frecuencia de ejecución, pero que cumplen con el "affordability" entre el verbo y el objeto. Los valores con probabilidad relativamente alta están asociadas a tareas simples con mayor probabilidad de ejecución respecto a los demás objetos de la fila.

La matriz **A** posee importantes cualidades que se pueden analizar. Si definimos dos vectores columna $\mathbf{c}(a_i)$, donde cada elemento es el número de veces que se repite determinada acción en el corpus, y $\mathbf{c}(o_i)$, donde cada elemento es el número de veces que se repite determinado objeto, de la siguiente forma:

$$\mathbf{c}(a_i) = \begin{pmatrix} c(a_0) \\ c(a_1) \\ \vdots \\ c(a_{L-1}) \\ c(a_L) \end{pmatrix} \tag{3.5}$$

$$\mathbf{c}(o_i) = \begin{pmatrix} c(o_0) \\ c(o_1) \\ \vdots \\ c(o_{M-1}) \\ c(o_M) \end{pmatrix} \tag{3.6}$$

A partir de la definición anterior, podemos derivar la siguiente relación algrebraica entre las matrices:

$$\mathbf{A}^t \cdot \mathbf{c}(a_i) = \begin{pmatrix} \sum_{i=0}^{L} c(a_i, o_o) \\ \sum_{i=0}^{L} c(a_i, o_1) \\ \vdots \\ \sum_{i=0}^{L} c(a_i, o_M) \end{pmatrix} \tag{3.7}$$

El vector resultante, dado por 3.7, corresponde a la suma de la frecuencia de todos los bigramas para un objeto dado, que es lo mismo que la frecuencia de aparición de este objeto, por tanto:

$$\mathbf{A}^t \cdot \mathbf{c}(a_i) = \mathbf{c}(o_i) \tag{3.8}$$

En la ecuación 3.8, es posible despejar $\mathbf{c}(a_i)$ si asumimos que $\mathbf{A}$ es una matriz cuadrada e invertible; por tanto:

$$\mathbf{c}(a_i) = (\mathbf{A}^t)^{-1} \cdot \mathbf{c}(o_i) = (\mathbf{A}^{-1})^t \cdot \mathbf{c}(o_i) \tag{3.9}$$

La ecuación 3.9 es un resultado interesante en relación a la matriz de affordability. Si la matriz en cuestión es cuadrada, la inversión y luego la transposición de la misma, multiplicada por el vector $\mathbf{c}(o_i)$, da como resultado el vector $\mathbf{c}(a_i)$.

El vector $\mathbf{c}(a_i)$ también pueden ser calculado, a partir de determinantes, usando la regla de Cramer:

$$\mathbf{c}(a_i) = \left(\frac{det(\mathbf{A}_i^t)}{det(\mathbf{A}^t)} \right) \tag{3.10}$$

En la ecuación 3.10, A_i^t es la matriz que se obtiene al reemplazar la columna i de A^t por el vector columna $\mathbf{c}(o_i)$.

El desarrollo realizado hasta este momento considera todas las acciones y objetos del escenario, sin embargo, si la cantidad de objetos y acciones en el escenario es muy significativa, la matriz es muy grande y casi inmanejable. En este caso, se puede trabajar con una matriz particionada, para la cual, los resultados obtenidos hasta este momento resultan igualmente válidos (Searle, 1982).

3.3.2. Entrenamiento

En lugar de crear tablas con la información de *affordances* para cada verbo y objeto, se construyó un corpus de tareas simples, a partir del cual se podía crear un modelo que permitiera calcular el *affordability*. Este método brinda la posibilidad de generar el modelo en forma

automática, una vez que se cuenta con un corpus de tareas simples. Esta forma de contruir el modelo también brinda mayor flexibilidad y eficiencia.

El sistema consiste en tres etapas: preprocesamiento, análisis léxico y lematización. La etapa de preprocesamiento borra comentarios y signos de puntuación, también transforma las tareas simples a minúscula. El analizador léxico reconoce el objeto y el verbo, y por último, la etapa de lematización identifica el lema de cada elemento (Ver figura 3.5).

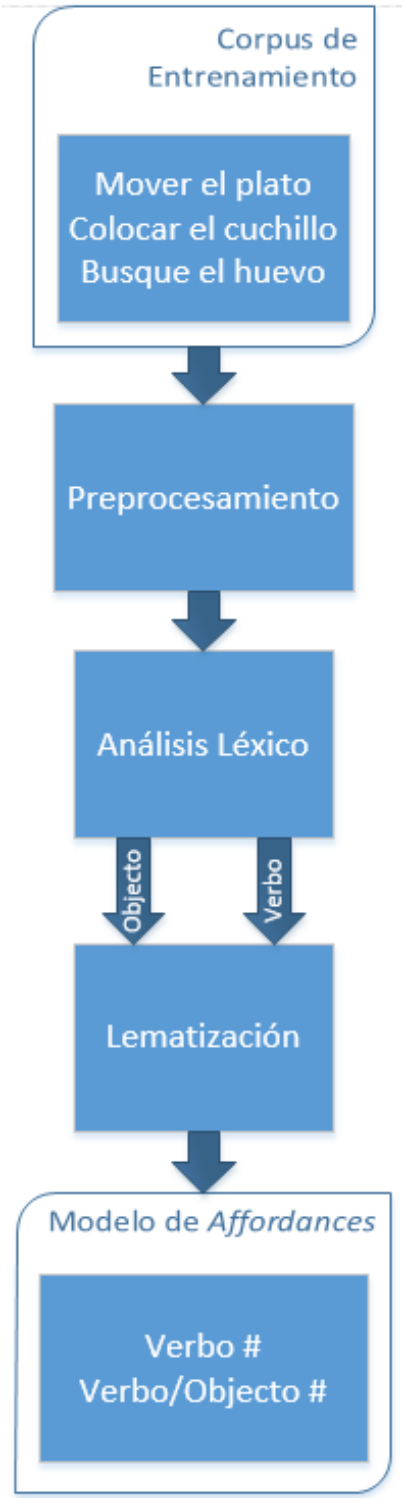

Figura 3.5: Entrenamiento de la unidad de *affordances* la cual consiste en tres etapas: preprocesamiento, análisis léxico y lematización.

La salida de la etapa de entrenamiento consiste en dos archivos de texto, uno para registrar el número de ocurrencias de cada verbo (unigramas) y otra registrar el número de ocurrencias del bigrama: verbo y objeto. Esta información será utilizada la calcular la probabilidad condicional indicada en la ecuación 3.1, esto para cualquier combinación de verbos y objetos.

3.3.3. Cálculo del *affordability*

Una vez generado el modelo de *affordances*, es posible calcular la probabilidad condicional. Este proceso consta de cuatro etapas: preprocesamiento, análisis léxico, lematización y cálculo de probabilidad (Ver figura 3.6). Las tres primeras etapas son idénticas a las utilizadas para generar el modelo de *affordances*. La última etapa es la nueva, y tiene el propósito de calcular el *affodability* a partir de la ecuación 3.1.

Una probabilidad de cero implica que el bigrama no tuvo ninguna ocurrencia, lo cual implica que no hay *affordability* entre el verbo y el objeto. Este resultado también puede deberse a que el corpus no es lo suficientemente grande, y el mismo no contiene el bigrama en cuestión. Para evitar este problema, el corpus debe ser los suficientemente grande.

Nótese que hay que determinar si entre el verbo y el objeto hay *affordability*, es decir, hay que tomar una decisión binaria, de acuerdo al valor de probabilidad calculado. Esta decisión se podría tomar utilizando algún estimador de decisión binaria, cómo por ejemplo (James

L. Melsa, 1978):

- Estimador de máxima verosimilitud
- Criterio *Neymar-Pearson*
- Criterio de probabilidad del error
- Criterio de riesgo de *Bayes*
- Criterio min-max

Estos cinco métodos o criterios se pueden resumir en el cálculo de la razón de verosimilitud dada por la ecuación 3.11 (James L. Melsa, 1978).

$$\frac{p(z|m_2)}{p(z|m_1)} = \Lambda(z) \underset{d_1}{\overset{d_2}{\gtrless}} T \tag{3.11}$$

En la ecuación 3.11, d_1 es tomar la decisión de que efectivamente, hay *affordability*, y d_2 es tomar la decisión opuesta, de que no hay *affordability* entre el verbo y el objeto. La observación z es el valor observado de probabilidad condicional que se calcula a partir de la ecuación 3.1.

Al analizar el problema en cuestión, nótese que $p(z|m_2)$ es en efecto una función indicatriz o función característica, lo cual resulta válido si el corpus de tareas simples es los suficientemente representativo. Esta

función se describe de la siguiente forma:

$$p(z|m_2) = i(z) = \begin{cases} 1 & \text{si } z = 0 \\ 0 & \text{si } z \neq 0 \end{cases} \tag{3.12}$$

Es decir, si se observa un valor z de cero, la función indicatriz sugiere que no hay *affordability*, esto porque la función de probabilidad tomaría un valor de 1. Por otro lado, la función $p(z|m_1)$ debería indicar lo mismo y dar un valor de cero, lo cual llevaría a la misma conclusión, en este caso se asume una función de probabilidad normal:

$$p(z) = \frac{1}{\sqrt{2\pi\sigma}} \cdot e^{\frac{-(z-\mu)^2}{2\sigma^2}} \tag{3.13}$$

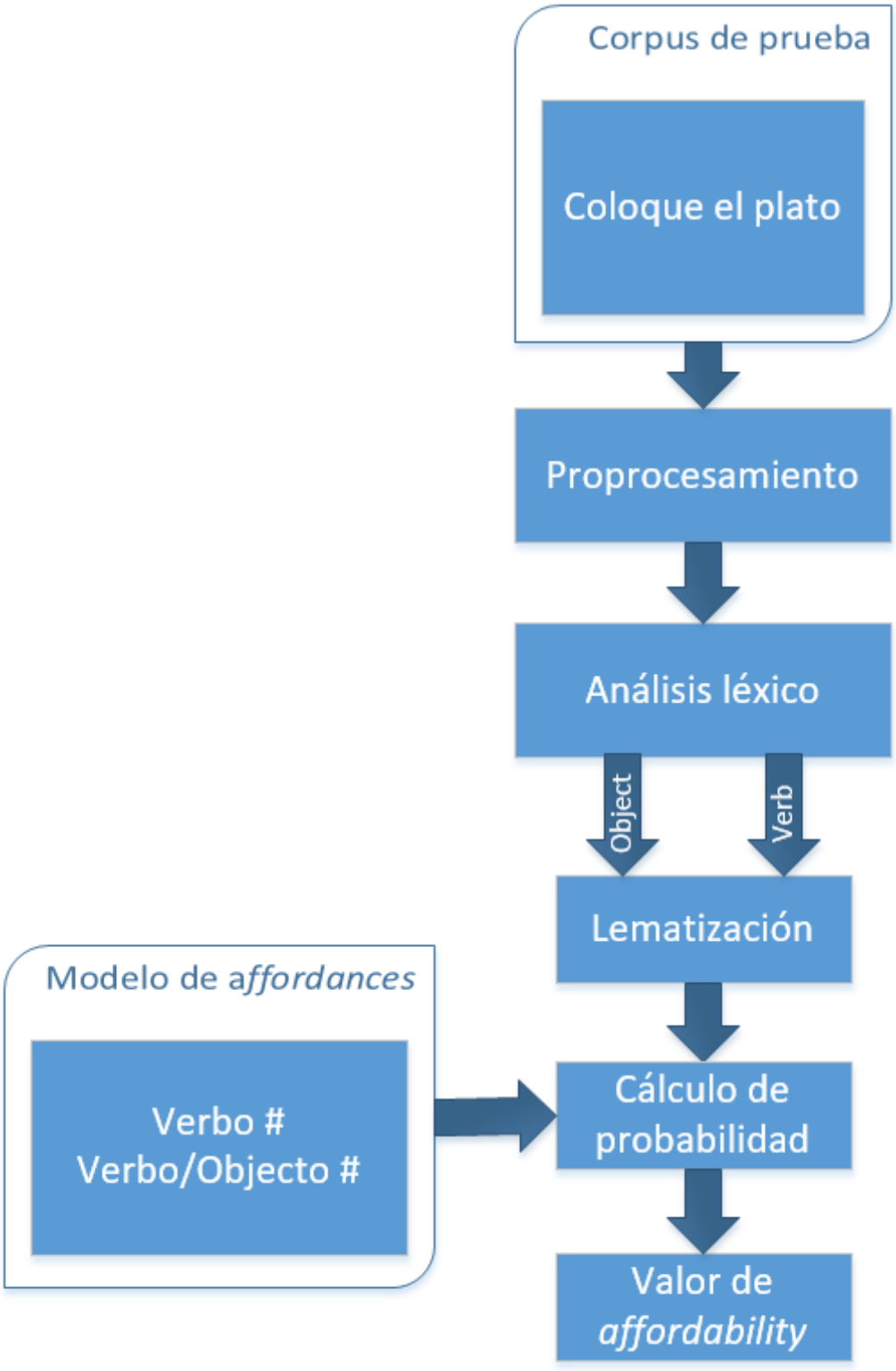

Figura 3.6: Prueba de la unidad de *affordances*, la cual consiste en cuatro etapas: preprocesamiento, análisis léxico, lematización y cálculo de probabilidad.

3.4. Unidad de mapeo

La unidad de mapeo, a partir del verbo y el objeto, genera un conjunto de controladores que pueden ejecutar las acciones con el objeto en cuestión. La implementación de este componente tiene el problema de que muy pocos modelos de objetos han sido desarrollados hasta ahora, por lo que su diseño e implementación será exploratoria.

La unidad de mapeo procesa el verbo y el objeto para generar una lista de controladores capaces de ejecutar las acciones con el objeto especificado. La unidad de mapeo también se encarga de tener un peso asociado a cada controlador, el cual es actualizado de acuerdo a los resultados en la unidad de análisis de factibilidad.

3.4.1. Categorización

Los verbos son categorizados en esta unidad, por ejemplo, los verbos: mover, colocar y poner, son básicamente la misma acción, pues los tres implican levantar un objeto de un punto y moverlo a otra ubicación. Estos verbos, denominados "levantar-poner", pueden utilizar el mismo controlador.

Esta unidad asocia el par $(verbo, objeto)$ a un conjunto de controladores de objetos dado por $\{c_0, \ldots, c_N\}$. Para cumplir esta función los controladores son etiquetados con los verbos que pueden ejecutar y los objetos que pueden manipular. Para implementar la categorización pa-

ra una tarea simple en específico, es necesario revisar estas etiquetas y determinar si el controlador en cuestión puede ejecutar la instrucción.

A cada controlador se le asigna un peso que será utilizado por la unidad de factibilidad para calcular el controlador que está mejor adaptado para ejecutar la tarea simple. Estos pesos son modificados de acuerdo a la formulación descrita en la siguiente sección.

3.4.2. Cálculo de los pesos

Para modelar las interacciones entre los objetos, acciones y controladores; definimos un conjunto de controladores dado por C, el cual incluye todos los controladores que pueden ejecutar la tarea de acuerdo a la categorización previa. El conjunto de controladores para N número de controladores es:

$$C = \{c_0, c_1, c_2, c_3, \ldots, c_N\} \tag{3.14}$$

En cada iteración, el peso asociado a cada controlador cambia. El conjunto de pesos correspondiente al conjunto de controladores está dado por:

$$W = \{w_0, w_1, w_2, w_3, \ldots, w_N\} \tag{3.15}$$

En cada iteración, un conjunto de controladores es procesado por la unidad de análisis de factibilidad, donde el peso de cada controlador

se modificará de acuerdo al siguiente criterio:

- Si el controlador es seleccionado por la unidad de análisis de factibilidad, se suma un valor A al peso ($+A$).
- Si el controlador no es seleccionado por la unidad en cuestión, restamos un valor P al peso ($+P$).

Después de j iteraciones, el peso del controlador tendrá el siguiente valor, donde: n es el número de veces que el controlador respectivo ha sido seleccionado, m el número de veces que el controlador ha sido rechazado, y w_i es el valor inicial asignada a cada peso. Al principio, el peso de los controladores es el mismo.

$$w_j = w_i + n \cdot A - m \cdot P \tag{3.16}$$

Los pesos cumplen un importante papel, ya que son utilizados en el módulo de análisis de factibilidad para determinar cuál controlador es más apto para ejecutar la tarea simple en cuestión.

3.5. Unidad de determinación de la referencia

Este componente calcula el valor deseado o valor de referencia a ser usado por los controladores del modelo de objetos. Este cálculo se realiza a partir del estado final y el adverbio de la tarea simple. Por ejemplo, en la tarea simple “mueva el vaso cerca del horno”, el adverbio

es “cerca” y el estado final está especificado por la posición del horno. Ambos elementos son usados para calcular donde ubicar el objeto.

Existen diferentes tipos de adverbios que expresan diferentes significados, tales como: adverbios de tiempo, adverbios de lugar y adverbios de modo. La traducción de un adverbio y un estado final a un valor deseado es un problema relevante, debido a que los adverbios pueden ser muy subjetivos y pueden estar definidos en conjunto con el contexto.

La implementación de esta unidad podría hacerse copiando el comportamiento humano frente a diferentes adverbios. Por ejemplo, nos podrían interesar que entiende un grupo de humanos con los adverbios: cerca, lejos, al frente y detrás. Esto se podría determinar haciendo experimentos y descubriendo la función de probabilidad que gobierna cada uno de estos adverbios, para luego replicar este comportamiento en esta unidad.

3.6. Unidad de propiedades de los objetos

Este módulo, a partir del nombre del objeto, determina las propiedades físicas del mismo, las cuales son utilizados para parametrizar los controladores del modelo de objetos, en la capa de bajo nivel. Por ejemplo, un vaso puede tener muchas propiedades, tales como: peso, coeficiente de fricción, forma y dureza. Estos datos pueden ser requeridos para efectuar el control y la predicción.

3.7. Unidad de análisis de factibilidad

Esta unidad determina cuál es el mejor controlador basado en: el *affordability*, la lista de controladores y las restricciones. En cuanto a las restricciones, estas pueden ser de tres tipos: del contexto, del cuerpo del robot y de los objetos.

Se considera que esta unidad debería hacer su análisis a partir de una función de costo, la cuál está dada por:

$$f(a_v, w_i, ct_0, ct_1, ct_2, \ldots) = a_v \cdot \sum_{i=0}^{N} [ct_i] + w_i \tag{3.17}$$

donde a_v es el valor de *affordability* previamente calculado en la unidad respectiva y ct_i son valores numéricos que representan tres tipos de restricciones: contexto, objeto y cuerpo del robot.

La función de costo propuesta en la ecuación 3.17 es el punto de partida para analizar la interacción entre la unidad de mapeo, la unidad de *affordances* y la unidad de cálculo de factibilidad. Por ahora, esta función es exploratoria, pero se considera un buen punto de partida para entender las interacciones que ocurre entre los diferentes bloques.

El *affordability* es parte de la función de costo, ya que este concepto está ligado a la viabilidad que presenta la ejecución de una tarea simple. Es por eso que la función toma en consideración esta variable en sus cálculos, de hecho, el valor de *affordability* multiplica la suma de

las restricciones, por lo que tiene un impacto significativo en el cálculo del factibilidad.

3.8. Especificaciones del lenguaje

El lenguaje es el proceso mental que se expresa a través del habla mediante el uso de palabras y una gramática que contiene aspectos: fonológicos, semánticos y sintácticos. En el contexto de esta investigación es importante diferenciar tres tipos de lenguaje: lenguaje natural, lenguaje de acción y comandos de control. El sistema de alto nivel procesa el lenguaje natural y el sistema de bajo nivel procesa el lenguaje de acción (véase figura 1.4).

El lenguaje natural se refiere a cualquier oración expresada por un humano, que generalmente consiste en instrucciones complejas, por ejemplo, “Sírvame un vaso de leche”. El lenguaje de acción consiste en instrucciones simples dirigidas a la realización de una tarea mínima y bien delimitada, por ejemplo, “Mueva la caja un metro a la derecha”. Los comandos de control corresponden al lenguaje específico utilizado para operar una articulación robótica.

El lenguaje humano está fuertemente determinado por la interacción social, la cual tiene como base: el sistema de producción, las creencias religiosas y los elementos culturales (Rodríguez Consuegra, 2003). Esto se denomina contexto lingüístico. Esto tiene una consecuencia importante en la especificación de un lenguaje, pues, en gene-

ral, es necesario un conocimiento detallado del contexto en que ocurre la comunicación para que el lenguaje sea totalmente entendido. Este es el principal reto en cuanto a la comunicación entre el ser humano y un sistema robótico.

Packard y Thenmozhi abordan la traducción de lenguaje natural a un lenguaje de acción, pues toman oraciones en lenguaje natural y las traducen a un lenguaje denominado *robot control language* (RCI) (Thenmozhi y cols., 2017). Estos dos autores consideran necesario un entendimiento y conocimiento del contexto para la traducción correcta de las oraciones. En cuanto al acople de un lenguaje de acción a un modelo de objetos, que es el objeto de estudio de investigación, también se requiere información de contexto.

La información de contexto, en el proceso de traducción de una instrucción simple a comandos de control, es proporcionada, en parte, por el modelo de objetos. Los modelos de objetos nos dicen como se comporta el objeto ante determinados estímulos. Sin embargo, esta información es suficiente, por lo que existen elementos adicionales que proporcionan información adicional acerca del entorno, uno de ellos es el lenguaje utilizado.

El contexto va más allá de proveer información adicional, si no que se considera que delimita el lenguaje que usamos y modifica los conceptos que pueden ser entendidos. La hipótesis de Sapir-Whorf establece que existe cierta relación entre las categorías gramaticales que una persona habla y la forma en que la persona entiende y conceptua-

liza el mundo (Hoijer, 1904). Esta hipótesis tiene dos versiones:

- Hipótesis whorfiana fuerte: El lenguaje de una persona determina completamente la forma en que conceptualiza, memoriza y clasifica la realidad.
- Hipótesis whorfiana débil: La lengua de un hablante tiene cierta influencia en la forma en que éste conceptualiza y memoriza la realidad.

La evidencia empírica muestra que el ser humano es capaz de categorizar y agrupar categorías de objetos a pesar de carecer de lenguaje, por lo que la hipótesis fuerte está descartada. Sin embargo, existe evidencia empírica a favor de la hipótesis débil, pues miembros de algunas tribus tienen problemas para reconocer colores para los cuales no existen palabras en el lenguaje.

Con base a ésto, una adecuada formalización del lenguaje de acción es fundamental para una adecuada manipulación de objetos en un escenario dado. Existe una relación entre el escenario y el lenguaje a utilizar en la arquitectura robótica. El escenario determina las acciones (verbos), los objetos y los posibles resultados deseados.

3.9. Casos de uso

En esta subsección se analiza como la arquitectura robótica propuesta puede tomar una tarea de alto nivel expresada en lenguaje

natural y procesarla en la capa de interfaz, de tal forma que la acción de manipulación pueda ser ejecutada por el robot.

3.9.1. "Hágame un desayuno"

Esta tarea compleja es usualmente llevada a cabo en una cocina. La receta puede variar en forma significativa de acuerdo a la geografía, el cocinero, e inclusive la disponibilidad de los ingredientes. Por tanto, el análisis se enfocará en un desayuno muy sencillo que se compone de una tostada con mantequilla y una taza de café.

La tarea compleja podría ser traducida a muchas tareas complejas, como lo son:

- Coloque la tostada sobre el plato.
- Unte la mantequilla sobre la tostada.
- Verter el café en la taza.

La primera tarea simple mueve la tostada al plato. La capa de interfaz debería poder traducir esta tarea simple a entradas hacia el modelo de objetos. A continuación se analizará como la capa de interfaz debería comportarse para esta tarea simple en la que se coloca la tostada sobre el plato.

La unidad de *affordances* verifica que la tostada puede ser movida, es decir, verifica si hay *affordability* entre el verbo y el objeto a ser

manipulado. La unidad de mapeo proporciona una lista de controladores que pueden ejecutar esta acción, por ahora, en este estudio de caso, supondremos que hay únicamente dos controladores que pueden ejecutar la acción respectiva, sobre el objeto en cuestión.

La unidad de determinación de referencia determina la ubicación actual del plato, la cual es el valor deseado. La únidad en cuestión conoce de antemano la ubicación de todos los objetos en la escena y proporciona la ubicación correspondiente a las capas inferiores de la arquitectura robótica. La unidad de propiedades del objeto proporciona la información respectiva, tales como: masa, fuerza de fricción, forma y dureza. Este módulo posiblemente ya conoce de antemano esta información.

Por último, la unidad de análisis de factibilidad analiza el contexto, el cuerpo del robot y el objeto a ser manipulado; con el fin de seleccionar el controlador más apto para ejecutar la tarea en cuestión. Para hacer esto, este módulo puede basar en experiencias previas, funciones de costo y modelos matemáticos.

3.9.2. "Hágame una ensalada"

Esta tarea compleja es muy común en una cocina y de igual forma que un desayuno, existe una gran cantidad variedad de ensaladas que pueden hacerse. En este estudio de caso nos enfocamos en una única ensalada con sólo dos ingredientes: lechuga y tomate. A partir de esta tarea complejo podemos derivar cientos o miles de tareas simples, tres

de ellas son:

- Corte el tomate en trozos pequeños.
- Rallar 20 gramos de lechuga.
- Mover la lechuga rallada a un plato.

Usaremos la segunda tarea simple para entender el comportamiento de la capa de interfaz. Una vez que iniciamos el procesamiento de esta tarea, la unidad de *affordances* confirma si hay *affordability* entre el verbo "rallar" y el objeto "lechuga", lo cual se cumple porque es bien conocido el hecho de que la lechuga puede ser rallada.

La unidad de mapeo proporciona la lista de controladores, por ahora supondremos que hay únicamente un controlador capaz de realizar esta tarea. La unidad de determinación de referencia proporciona la referencia del controlador. El valor deseado en este caso son "20 gramos", que es la cantidad de lechuga que debe ser rallada. Nótese que en este caso, el valor de referencia no es una ubicación o posición, si no que es una cantidad.

Posteriormente, la unidad de propiedades de los objetos da la información concerniente al objeto "lechuga". Por último, la unidad de análisis de factibilidad analiza si el controlador está preparado para ejecutar la tarea simple en cuestión.

Diseño de experimentos

En la presente sección se detallan los experimentos diseñados y desarrollados durante el proceso de implementación y prueba del sistema propuesto.

4.1. Definición de escenarios y tareas complejas

En este apartado se definen los escenarios y tareas complejas a ser utilizados en la construcción del corpus de tareas simples.

4.1.1. Escenario trivial binario

Para comenzar el análisis consideremos un universo donde un objeto puede tener solo dos posibles estados (ubicaciones espaciales), a las cuales denominaremos: A y B. Este escenario idealizado únicamente posee un único objeto al cual denominaremos "caja". Este contexto, a pesar de su simplicidad, puede tener una gran complejidad, pues las acciones posibles que una articulación robótica puede realizar son muy variadas, por ejemplo: arrastrar, empujar, mover, entre otros.

El conjunto de acciones posibles se determina a partir de las capacidades del robot y no a partir del universo posible de palabras existentes en el idioma. Esto se relaciona con el concepto de *affordances*, el cual es medular, ya que el conjunto de verbos dependerá de ambos ele-

mentos: las capacidades de la articulación robótica y las características del escenario (lo que incluye los objetos que se manipularán) (Glenberg y Kaschak, 2002).

En este escenario trivial se considera que el robot solo tiene capacidad para "mover" el objeto. Esta idealización permite generar un lenguaje que puede constar únicamente de dos instrucciones:

Mueva la caja a la posición A

Mueva la caja a la posición B

El escenario binario permite visualizar la complejidad del lenguaje de acción, pues la cantidad posible de instrucciones se relaciona con: el número de verbos, el número de estados finales posibles y el número de objetos presentes en el escenario. En este caso en particular son posibles únicamente dos instrucciones.

A partir del escenario del análisis del caso binario, podemos derivar la siguiente metodología para determinar el lenguaje a partir del escenario. Nótese que el escenario incluye la articulación robótica y los objetos físicos que se manipulan.

- Determinar los nombres de los objetos que ese encuentran en el escenario.
- Determinar todas las posibles acciones que puede realizar el robot sobre estos objetos.

- Determinar todos los posibles estados que pueden tener los objetos del escenario.

4.1.2. Escenarios de complejidad real

El referente más importante en relación a escenarios de complejidad real es el proyecto denominado *Yale-CMU-Berkeley Object and Model Set*, el cual es un esfuerzo entre varias universidades para estandarizar los objetos con lo que se que hace investigación en manipulación de objetos. Esta es una biblioteca de objetos de la vida diaria con diferentes formas, tamaños, texturas, peso y rigidez. La base de datos incluye: imágenes RGBD, propiedades físicas, modelos geométricos, entre otros (Calli y cols., 2015).

La librería contiene 76 objetos los cuales podrían formar parte del lenguaje de acción que se está diseñando. Estos están clasificados en cuatro grupos:

- Alimentos: *chips can*, *coffee can*, *cracker box*, *box of sugar*, *tomato soup can*, *middle row*, *mustard container*, *tuna fish can*, entre otros.
- Herramientas: *back row*, *power drill*, *wood block*, *middle row*, *scissors*, *padlock*, entre otros.
- Utensilios de cocina: *pitcher*, *bleach cleanser*, *abrasive sponge*, entre otros.

- Formas: *softball*, *tennis ball*, *golf ball*, *marbles*, *rope*, *cups*, *blank credit card*, entre otros.

Figura 4.1: Alimentos pertenencientes al estándar YCB.

El proyecto proporciona una lista de sugerencias de tareas de manipulación para cada grupo de objetos, la cuales se podrían tomar como referencia para generar la lista de verbos. Sin embargo, este proyecto presenta varios puntos débiles que imposibilitan su utilización en esta investigación:

- Las propiedades físicas de los objetos están limitadas a las dimensiones y el peso. Los objetos en cuestión carecen del modelo matemático.

- La articulación robótica del laboratorio no puede lidiar con la manipulación de algunos objetos pequeños como arandelas y tornillos.
- Existe la posibilidad de adquirir los objetos, pero se podrían tener problemas con la importación de recipientes de comida o químicos. Además de que estos objetos son ajenos a nuestra realidad nacional.

4.1.3. Definición de escenarios

A pesar de las debilidades arriba expuestas, algunos de estos objetos se podrían utilizar para construir el lenguaje de acción. El primer paso es definir dos escenarios, los cuales son:

- Cocina: Las nuevas herramientas, metodologías o conceptos, en el campo de la robótica, deben ser evaluados en un escenario real. Un escenario común para hacer estas validaciones es una cocina (Yamazaki, Watanabe, Nagahama, Okada, y Inaba, 2010) (Winkler y Beetz, 2015) (Lagrand, v. d. Meer, y Visser, 2016) (Blodow y cols., 2011). Las tareas de manipulación que se realizan en una cocina son complejas, diversas y de gran utilidad práctica; lo que justifica su escogencia como escenario.
- Mesa de trabajo: El robot debe poder realizar tareas sencillas relacionadas con el ensamblaje de componentes y materiales. Estas tareas se definirán en la siguiente sección, pero están relacionadas

con las unión o ensamblaje de componentes y materiales.

Los escenarios pueden variar en forma significativa, por ejemplo, una cocina en un restaurante es muy diferente a la cocina de una casa de una familia pequeña. La mesa de trabajo de un taller electromecánico podría ser muy diferente a la mesa de trabajo de un laboratorio de física de una universidad. En vista de lo anterior, la definición de escenarios debe hacerse en forma detallada y específica.

El escenario **cocina** consta de los siguientes elementos: un microondas, una pila o fregadero, una estufa, una refrigeradora, una licuadora, tres gabinetes y una mesa central (Véase figura 4.2). Es a partir de esta cocina en particular que se harán análisis posteriores.

El escenario **mesa de trabajo** consta de los siguientes elementos: dos gavetas, una sierra, un martillo, un juego de llaves, un juego de destornilladores, una llave francesa, un juego de alicates y una cinta métrica (Vease figura 4.3). Los análisis posteriores se basarán en esta mesa de trabajo.

El punto de partida para definir tareas complejas son los dos escenarios reales seleccionados. En cada uno de ellos se pueden establecer tareas complejas comunes, sobre las cuales se puede construir un corpus de tareas más simples, denominadas tareas simples.

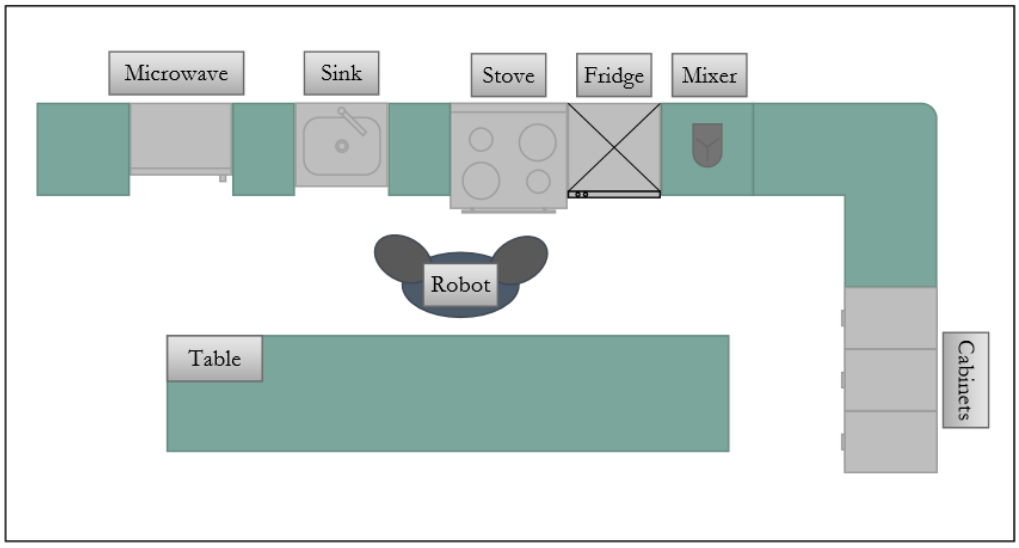

Figura 4.2: Escenario cocina el cual consiste de: un microondas, un fregadero, una estufa, una refrigeradora, una licuadora, tres gabinetes y una mesa central.

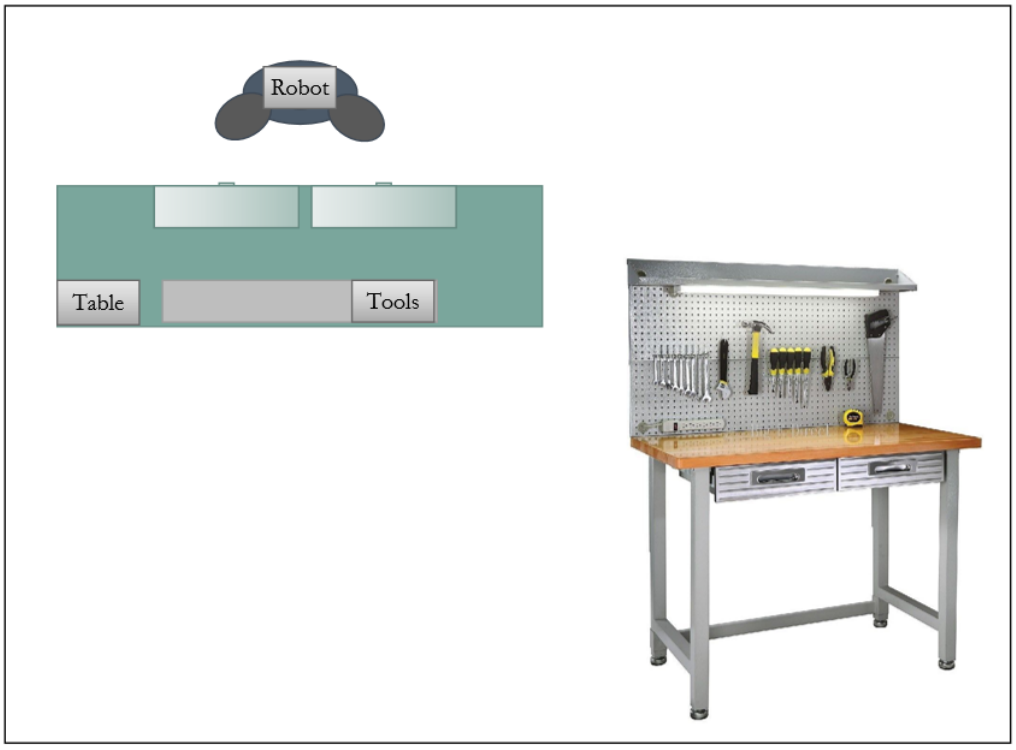

Figura 4.3: Mesa de trabajo colaborativa la cual consiste en los siguientes elementos: dos gavetas, un serrucho, un martillo, un juego de llaves, una llave francesa, un juego de desatornilladores, un juego de alicates y una cinta métrica.

4.1.4. Definición de tareas complejas

En cada escenario es posible definir una gran cantidad de tareas complejas. Por ejemplo, en una cocina, se podría establecer la tarea de "realizar un almuerzo". Esta tarea en sí misma se reviste de una gran complejidad, pues requiere cientos o miles de instrucciones simples para ser llevadas a cabo. Asociado a cada tarea simple existe un conjunto de controladores asociados que pueden desempeñar esa tarea.

Como punto de partida, para construir el lenguaje, se considerarán dos tareas complejas en el escenario "cocina" y dos tareas complejas en el escenario "mesa de trabajo". Las dos tareas en el escenario cocina, son las siguientes:

- Haga un emparedado.
- Haga un desayuno.

En el caso de la mesa de trabajo, las dos tareas seleccionadas son las siguientes:

- Haga una caja de madera.
- Limpie el teclado de la computadora.

Las tareas seleccionadas deben cumplir con una serie de requerimientos: ser realizables con la articulación robótica disponible, ser ricas en tareas simples, consistir en actividades comunes a realizar en la vida diaria y ser requerir una variedad importante de controladores.

4.2. Generación del corpus de tareas simples

Para generar el vocabulario del lenguaje de acción en una escena o ambiente determinado, es necesario generar la lista de tareas simples a partir de las tareas complejas que fueron seleccionadas. Nótese que el corpus generado consiste en tareas simples, las cuales se han definido de la siguiente forma:

> Una tarea es simple, si su ejecución requiere de único modelo de objetos en la capa de bajo nivel de la arquitectura robótica.

En la actualidad existen pocos modelos de objetos, por lo que el correcto establecimiento de una tarea simple depende, por mucho, del desarrollo de estos modelos en el futuro. Esto implica que el corpus generado deberá ser mejorado de acuerdo a los avances y desarrollos que se logren en este campo en particular.

El análisis de esta lista de tareas simples resulta ser muy relevante para el robot, pues permite establecer que acciones son requeridas, cuales objetos serán manipulados en la escena y cuales son las capacidades físicas que el robot posee. Los pasos a seguir para obtener la lista de tareas simples fueron los siguientes:

- Establecer el ambiente o escenario en el que el análisis será realizado. La descripción del mismo debe ser clara y concisa. Algunos

ejemplos son: una cocina, una mesa de trabajo, un taller de reparación de electrodomésticos, un laboratorio de química, entre otros.

- Determinar un conjunto de tareas complejas que son importantes en el escenario especificado. Por ejemplo, Beetz estableció las siguientes tareas complejas en una cocina: arreglar la mesa(*table setting*), cocinar (*cooking*) y limpieza (*house keeping*).(Beetz y cols., 2008).

- Una vez que las tareas complejas han sido definidas, se trabaja con expertos o con fuentes externas para generar la lista de tareas simples. Una alternativa es obtener las tareas simples de sitios como: *ehow.com* y *wikihow.com*, esta alternativa fue utilizada por Tenorth para transformar descripciones de tareas en una cocina en planes ejecutables por robots.

- La lista de tareas simples es analizada para determinar los requerimientos del robot en cuanto a objetos y acciones, lo cual deriva en requerimientos en cuanto al hardware y el software.

La lista de tareas simples puede ser construida por expertos en la tarea compleja, es decir, por personas que saben como ejecutar la tarea. La lista también se puede generar por voluntarios quienes, a partir de un sitio web como *wikihow.com* escriben la lista de tareas de simples. Por último, existen diversos investigadores que han obtenido las tareas utilizando alguna solución automática (Tenorth, Nyga, y Beetz, 2010) (Costa, Veiga, Sousa, y Nunes, 2017).

Entre estos enfoques más automatizados, tenemos por ejemplo, la generación de las tareas partir de la traducción de recetas de cocina, este es el caso del artículo de Moritz denominado *Understanding and executing instructions for everyday manipulation tasks from the World Wide Web* (Tenorth y cols., 2010). En el caso del artículo de denominado *Evaluation of Stanford NER for extraction of assembly information from instruction manuals* la generación se hizo a partir de manuales de ensamblaje (Costa y cols., 2017).

Para aplicar esta metodología, se seleccionaron dos escenarios: una cocina y una mesa de trabajo (Véase sección 4.1.3). En el caso de la cocina, dos tareas complejas fueron seleccionadas: "hacer un emparedado" y "hacer un desayuno". En el caso de la mesa de trabajo, las siguientes dos tareas complejas fueron seleccionadas: "haga una caja de madera" y "limpie el teclado de la computadora".

4.3. Cálculo de *affordability*

En esta subsección se presentan los experimentos que se hicieron para validar la propuesta de cálculo de los *affordances*.

4.3.1. Experimento A: Matriz de *affordances*

La unidad de *affordances* se valida utilizando los cinco verbos y los cinco objetos más utilizados en el escenario cocina. Esto para generar una matriz de *affordances*. Las tareas simples utilizadas se muestran

en la figura 4.4.

```
#####################################
Mueva el plato
Busque el plato
Coloque el plato
Introducir el plato
Verter el plato
#####################################
Mueva el cuchillo
Busque el cuchillo
Coloque el cuchillo
Introduzca el cuchillo
Verter el cuchillo
#####################################
Mueva el tomate
Busque el tomate
Coloque el tomate
Introducir el tomate
Verter el tomate
#####################################
Mueva el trozo de pan
Busque el trozo de pan
Coloque el trozo de pan
Introduzca el trozo de pan
Verter el trozo de pan
#####################################
Mueva el huevo
Busque el huevo
Coloque el huevo
Introduzca el huevo
Verter el huevo
#####################################
```

Figura 4.4: Experimento A. Construcción de matriz de *affordances*

La matriz de *affordances* presenta cualidades matemáticas muy interesantes que fueron analizadas en la sección anterior y tiene el potencial de ser utilizada para modelar el mundo (*world modelling*) en conjunto con los ya existentes modelos físicos.

La etapa de entrenamiento, con el fin de construir el modelo, consistió en 236 tareas simples. Este corpus se construyó a partir de cuatro tareas complejas: "Hacer un desayuno", "Hacer una ensalada", "Hacer un emparedado" y "Hacer huevos revueltos".

4.3.2. Experimento B: Verificación de *affordances*

Este segundo experimento utilizó el mismo modelo de *affordances*, el cual cuenta con 236 tareas simples. El conjunto de pruebas consistió en diez tareas simples, que se sabía previamente, no cumplían con el *affordability* entre el verbo y el objeto, esto para validar que en efecto, el sistema fuera capaz de hacer el análisis correctamente. El conjunto de pruebas utilizado se muestra en la figura 4.5.

Nótese que la oraciones utilizadas no son viables de ser ejecutadas, o al menos, su ejecucción en el mundo real es altamente improbable. Por ejemplo, la instrucción "quiebre el cuchillo" o "encienda el huevo", debería dar un *affordability* de cero, ya que en el mundo real, estas dos tareas no suelen ser ejecutadas, pues lo huevos no pueden ser encendidos y la tarea de quebrar un cuchillo no es muy poco común.

```
Quiebre el cuchillo sobre la mesa
Encienda el huevo
Corte el plato
Encienda el repollo
Corte la tostadora
Lave la tostadora
Apague la lechuga
Corte la refrigeradora
Encienda la tostada
Pique la mayonesa
```

Figura 4.5: Experimento B. Instrucciones simples que no cumplen con el *affordability*

4.4. Cálculo de factibilidad

En esta subsección se analiza la interacción entre dos unidades de la capa de interfaz: la unidad de mapeo y la unidad de análisis de factibilidad. La unidad de mapeo genera una lista de controladores que pueden ejecutar la tarea simple. En paralelo, la unidad de *affordances* calcula un valor numérico basado en el verbo y el objeto. En cada iteración, dos cosas ocurren: el valor de factibilidad es calculado basado en la función de costo, dado por la ecuación (3.17), y los pesos son calculados en base a la ecuación (3.15).

Con el propósito de estudiar la evolución de los pesos, se tomaron dos decisiones para simplificar el análisis:

- El *affordability* es generado en forma aleatoria, pero una vez que

se genera, se mantiene constante durante todo el experimento, es decir, para todas las iteraciones.

- Las restricciones son también generadas en forma aleatoria, pero en este caso, se generan valores diferentes en cada iteración.

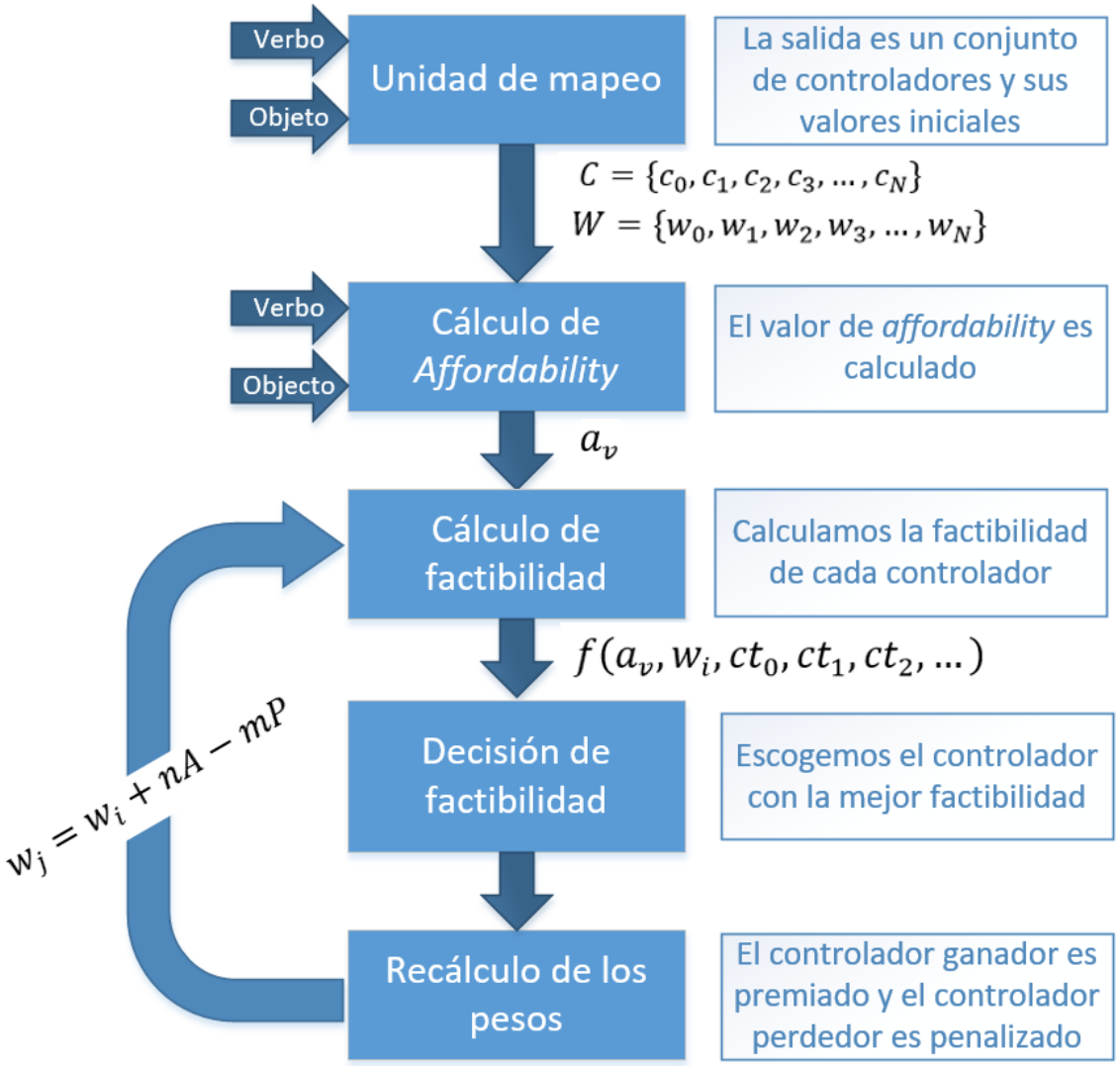

Figura 4.6: Diagrama de flujo que muestra la interacción entre la unidad de mapeo y la unidad de análisis de factibilidad.

Las variables de respuesta son los pesos de los controladores. El factor de diseño es el valor penalidad-incentivo. Los factores fijos

son: los pesos iniciales, la función de costo y el algoritmo penalidad-incentivo. Los factores de ruido son: el *affordability* y las restricciones; debido a que ambos son generados en forma aleatoria. Los valores numéricos correspondientes a las restricciones cambian en cada iteración, pero el valor de *affordability* es fijo, una vez que es generado.

El número de iteraciones es de 200, eso porque experimentos preliminares mostraban una tendencia clara, una vez que el número de iteraciones alcanzaba este valor. El número de niveles para el número de controladores es dos, pues se hacen las validaciones para dos y tres controladores. Para el valor penalidad-incentivo, se establecieron tres niveles: 0.05, 0.1 y 0.2.

El cuadro 4.1 detalla los valores numéricos para los que se realizó la simulación de dos controladores. El cuadro 4.2 muestra la misma información, pero para el caso de tres controladores. La utilización de dos y tres controladores se debe a que la disponibilidad de controladores para acciones de manipulación es aún muy limitada.

Cuadro 4.1: Experimento con dos controladores.

Factores	Símbolo	Valores
N. de controladores	-	2
Penalización	P	0.05/0.1/0.2
Incentivo	A	0.05/0.1/0.2
Pesos iniciales	w_i	10
Affordability	a_v	[0,1]
Restricciones	ct_j	[0,10]

Cuadro 4.2: Experimento con tres controladores.

Factores	Símbolo	Valores
N. de controladores	-	3
Penalización	P	0.05/0.1/0.2
Incentivo	A	0.05/0.1/0.2
Pesos iniciales	w_i	10
Affordability	a_v	[0,1]
Restricciones	ct_j	[0,10]

4.5. Experimentos finales

El sistema mostrado en la figura 3.3 se implementó en el lenguaje de programación Python. Los cinco módulos que lo constituyen se programaron en archivos separados y se usó JSON para el almacenamiento de varios tipos de información. La interconexión entre los módulos o subsistemas se hizo usando YARP (Metta, Fitzpatrick, y Natale, 2006).

A continuación se listan los archivos correspondientes al código fuente de la capa de interfaz:

- affordance_unit.py
- determination_of_reference.py
- feasibility.py
- lexical_analysis.py

- main.py
- mapping_system.py
- object_models.py
- properties.py
- tiny_DB.py

Los archivos tipo JSON se usaron para almacenar información relativa a: *affordability*, posiciones actuales de los objetos, pesos y asociatividad de los controladores. Estos archivos se listan a continuación:

- affordance_database.json
- position_database.json
- controllers_weight.json
- controller_database.json

El archivo *affordance_database.json* es una base de datos que almacena el *affordability* entre el verbo y el objeto. El archivo *position_database.json* asocia cada objeto con su posición actual en el escenario. El archivo *controllers_weight.json* almacena los pesos asociados a cada controlador. Por último, el archivo denominado *controller_database.json*, para cada controlador establece que acciones o verbos se pueden ejecutar y cuales objetos se pueden manipular.

El conjunto de tareas simples utilizado como conjunto de prueba se muestra en la figura 4.7. Este corpus consiste en 32 tareas simples con las que se desea probar el sistema que fue desarrollado.

```
#!/usr/bin/python

import main

main.main_system('Buscar la tostadora')
main.main_system('Buscar el pan')
main.main_system('Mover trozo de pan #1 cerca de la tostadora')
main.main_system('Introducir el trozo de pan #1 en la tostadora')
main.main_system('Mover trozo de pan #2 cerca de la tostadora')
main.main_system('Introducir el trozo de pan #2 en la tostadora')
main.main_system('Encender la tostadora')
main.main_system('Buscar un plato')
main.main_system('Mover el plato cerca de la tostadora')
main.main_system('Esperar que el pan este listo')
main.main_system('Colocar trozo de pan #1 en el plato')
main.main_system('Colocar trozo de pan #2 en el plato')
main.main_system('Mover plato a la mesa')
main.main_system('Mover mantequilla a la mesa')
main.main_system('Mover cuchillo a la mesa')
main.main_system('Untar trozo de pan #1 con mantequilla')
main.main_system('Untar trozo de pan #2 con mantequilla')
main.main_system('Mover trozo de pan #3 cerca de la tostadora')
main.main_system('Introducir el trozo de pan #3 en la tostadora')
main.main_system('Mover trozo de pan #4 cerca de la tostadora')
main.main_system('Introducir el trozo de pan #4 en la tostadora')
main.main_system('Encender la tostadora')
main.main_system('Buscar un plato')
main.main_system('Mover el plato cerca de la tostadora')
main.main_system('Esperar que el pan este listo')
main.main_system('Colocar trozo de pan #3 en el plato')
main.main_system('Colocar trozo de pan #4 en el plato')
main.main_system('Mover plato a la mesa')
main.main_system('Mover mantequilla a la mesa')
main.main_system('Mover cuchillo a la mesa')
main.main_system('Untar trozo de pan #3 con mantequilla')
main.main_system('Untar trozo de pan #4 con mantequilla')
```

Figura 4.7: Conjunto de tareas simples utilizadas para probar el sistema completo.

Resultados y discusión de experimentos

A continuación se discuten y analizan los resultados de los experimentos realizados en el capítulo anterior. Esta sección se divide en cuatro partes: los resultados referentes al corpus de tareas simples, la validación sobre el módulo de *affordances*, las simulaciones que se hicieron sobre la unidad de factibilidad, y los experimentos finales sobre la capa de interfaz.

5.1. Análisis del corpus

El corpus de tareas simples fue analizado; lo que permitió llegar a resultados muy interesantes en relación a los dos escenarios analizados y las tareas simples seleccionadas.

5.1.1. Verbos

En el escenario cocina, para las tareas complejas en cuestión, los verbos más comunes son: mover, buscar, colocar e introducir (Ver cuadro 5.1). Nótese que estas cinco acciones son requeridas por el robot para ejecutar el 72.4 % de las tareas simples. Cualquier robot que deba trabajar en este ambiente, posiblemente requiera ejecutar estas cinco acciones. Esto no sólo debe ser considerado por el modelo de objetos,

si no también, por la construcción mecánica de las articulaciones y la arquitectura software del robot.

Los verbos del tipo "agarrar y colocar" (*pick and place*) resultan ser muy comunes en este escenario. Los verbos: mover, ubicar e introducir; son acciones que corresponden a la misma acción de manipulación, tomar un objeto y colocarlo en otro sitio. Estos tres verbos corresponden al 49.28 % de las tareas simples. Es importante hacer notar que un único modelo de objetos podría ser usado para ejecutar estas tres acciones de manipulación.

Cuadro 5.1: Verbos más comunes en la cocina.

Verbo	Número de Ocurrencias	Porcentaje
Mover	44	31.88 %
Buscar	26	18.84 %
Colocar	16	11.59 %
Introducir	8	5.80 %
Verter	6	4.35 %
Otros	40	27.54 %

El corpus de tareas simples, que fue generado usando la cocina como escenario, consta de 20 verbos diferentes, de los cuales sólo siete verbos corresponden al %79.71 de las tareas simples. Es decir, un conjunto pequeño de siete verbos, es decir, el 35 % de la acciones de manipulación, permiten ejecutar el 80 % de las tareas simples. Nótese que el principio de pareto también se podría aplicar a la manipulación de objetos, pues son necesarias pocas acciones de manipulación para ejecutar la mayoría de tareas.

El verbo “buscar” se incluye en la tabla 5.1, sin embargo, no necesariamente, se trata de una acción de manipulación, ya que únicamente le indica al robot que debe encontrar la ubicación del objeto especificado. Esto significa que el robot debería saber cómo encontrar los objetos en la sala al ejecutar esta acción. Este verbo, por sí sólo, corresponde al 18.84 % de las tareas simples.

Un verbo que se reviste de especial importancia, de acuerdo al análisis realizado, es “verter”, el cual es una acción muy compleja que consiste en llenar un contenedor o recipiente con un líquido, fluido u otro material. Esta acción requiere de la manipulación de dos objetos: el contenedor que es llenado y el recipiente que es vaciado. Esta acción equivale al 4.25 % de las tareas simples.

Los verbos más utilizados en la mesa de trabajo difieren en forma significativa de los más utilizados en la cocina. En el caso de la mesa de trabajo, el verbo más frecuente no es una acción del tipo “agarrar y colocar”, sino un verbo más específico relacionado con una de las tareas complejas, el cual es el verbo “lijar”. Este verbo en particular aparece en el 17.01 % de las tareas simples. El segundo verbo que más aparece es “colocar”, el cual si es una acción del tipo “agarrar y colocar”, y aparece en el 16.33 % de las tareas simples (Ver tabla 5.2).

5.1.2. Objetos

En la cocina, los objetos más comunes son: plato, cuchillo, tomate, trozo de pan y huevo. Estos cinco objetos constituyen el 41.3 % de

Cuadro 5.2: Verbos mas comunes en la mesa de trabajo.

Verbo	Número de Ocurrencias	Porcentaje
Lijar	25	17.01 %
Colocar	24	16.33 %
Buscar	19	12.93 %
Limpiar	16	10.88 %
Taladrar	16	10.88 %
Otros	47	31.97 %

las tareas simples. De acuerdo a estos datos y las tareas complejas seleccionadas, el objeto más utilizado en la cocina es el plato, con un 10.14 %. El segundo objeto más utilizado es el cuchillo con un 9.42 %. El tercer objeto más usado es el tomate, con un 8.70 %.

El modelo de objetos de la capa de bajo nivel debe poder manipular estos objetos para el escenario y las tareas complejas en cuestión. Desde el punto de vista físico, en la construcción del robot, las articulaciones robóticas deben tener la capacidad de manipular un plato, un cuchillo y un tomate (los objetos más utilizados en la cocina).

En la mesa de trabajo colaborativa y para las tareas complejas seleccionadas, los objetos más utilizados son: pieza de madera, tecla, tornillo, borde de la caja de madera y cepillo. Esto cinco objetos corresponden al 78.2 % de las tareas simples. La pieza de madera es, por mucho, el objeto más referenciado en este corpus, con un 35.7 % de las tareas simples. El segundo objeto más referenciado es la tecla, con un 17.0 %.

5.1.3. Variación

De acuerdo a las tareas complejas seleccionadas y a las tareas simples analizadas, el verbo "mover" es más utilizado en la cocina, que en la mesa de trabajo colaborativa (Ver figura 5.1). Esto probablemente ocurre porque la cocina es un área más grande, donde hay que mover los objetos de un lugar a otro, en cambio, la mesa de trabajo es un lugar más pequeño donde la mayoría de materiales y herramientas están "a la mano". Los mismo ocurre con el verbo "buscar", el cuál es ligeramente más utilizado en la cocina que en la mesa de trabajo, posiblemente, por la misma razón.

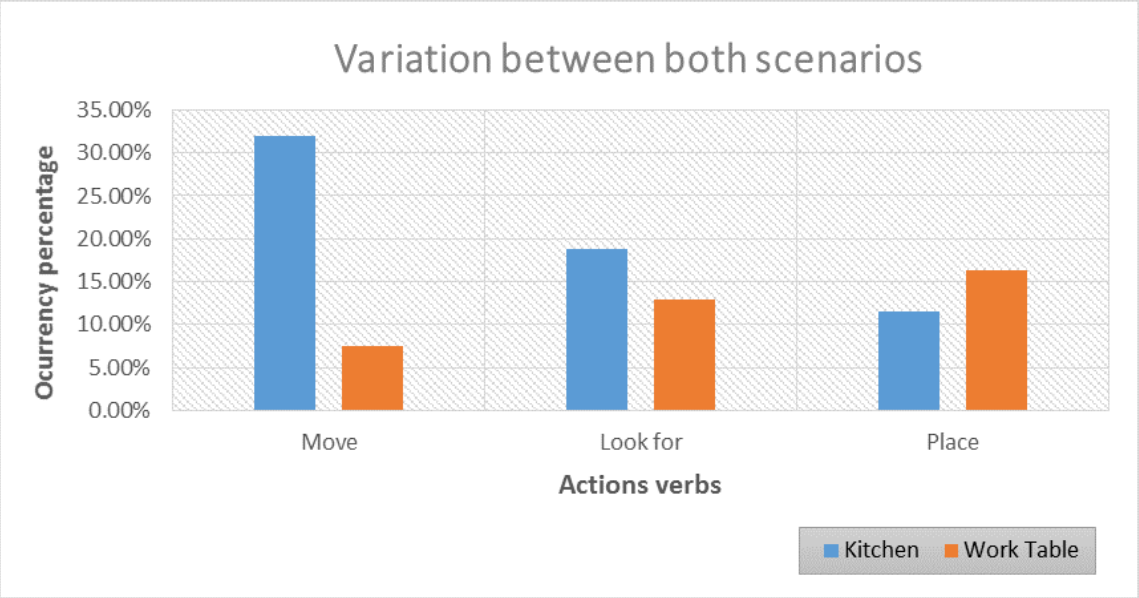

Figura 5.1: Porcentaje de ocurrencia entre escenarios para tres verbos: mover, buscar y colocar.

Otra aspecto relevante es la diferencia entre los dos escenarios para el verbo "colocar". Este verbo es más frecuentemente utilizado en la mesa de trabajo que en la cocina; lo cual posiblemente ocurre

debido a que en la mesa de trabajo se requiere constantemente quitar y colocar objetos tales como: herramientas y materiales, por otro lado, en el caso de la cocina, lo que se mueven son los ingredientes.

La variación entre los verbos fue analizada entre tareas complejas para ambos escenarios, la cual fue medida para los verbos más comunes en el escenario respectivo al calcular la raíz cuadrática media, tal como se muestra a continuación:

$$\sigma_{verbos} = \sqrt{(verbo(i)_{Tarea\,Compleja\,\#1} - verbo(i)_{Tarea\,Compleja\,\#2})^2} \cdot 100 \tag{5.1}$$

donde $verbo(i)$ es el porcentaje de ocurrencia para el mismo verbo en ambas tareas complejas, referido al mismo escenario. Al utilizar esta métrica en la cocina, se obtuvo un 3.18 % y en la mesa de trabajo colaborativa se obtuvo un 15.60 %. Basado en este análisis, las acciones en la mesa de trabajo colaborativa muestran mayor variabilidad que en el escenario cocina.

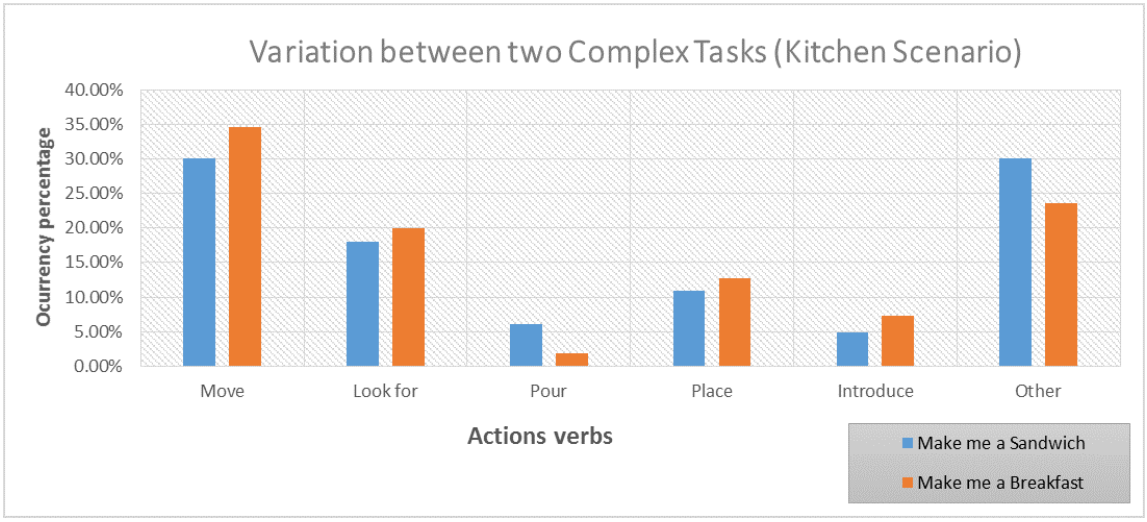

Figura 5.2: Variación entre tareas complejas para la cocina: Se muestra la variación entre tareas complejas, en el escenario cocina, para los siguientes verbos: mover, buscar, verter, colocar e introducir.

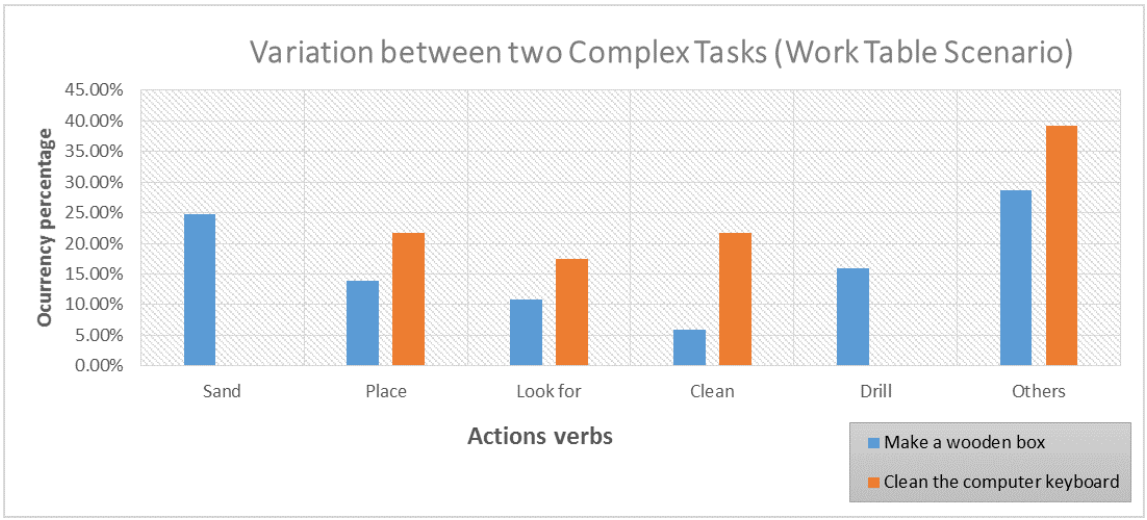

Figura 5.3: Variación entre tareas complejas para la mesa de trabajo colaborativa: Se muestra la variación entre tareas complejas, en la mesa de trabajo, para los siguientes verbos: lijar, colocar, buscar, limpiar y taladrar.

5.2. Affordances

5.2.1. Experimento A: Matriz de *affordances*

En la tabla 5.3 se muestra el cálculo de la probabilidad condicional para los cinco objetos más utilizados en la cocina y las cinco acciones (verbos) que son realizadas con mayor frecuencia en este escenario. Esta matriz se conoce como "matriz de *affordances*". La matriz de affordances resulta vital, pues permite describir y caracterizar un escenario a través de un objeto matemático muy compacto.

Cuadro 5.3: Matriz de *affordances*

Verbo/Objeto	Plato	Cuchillo	Tomate	Pan*	Huevo
Mover	0.12	0.14	0.10	0.07	0.03
Buscar	0.17	0.09	0.05	0.00	0.09
Colocar	0.12	0.03	0.12	0.15	0.06
Introducir	0.00	0.00	0.10	0.00	0.00
Verter	0.00	0.00	0.16	0.00	0.08

* Trozo de pan

Es importante comentar estos resultados. Esta matriz muestra la probabilidad condicional de que se manipule un determinado objeto dado que se tiene un verbo en específico. Por ejemplo, la probabilidad condicional dada por $P(Plato|Mover)$ es la probabilidad de que el objeto a manipular sea un plato, una vez que la acción seleccionada ha sido "mover".

En los casos en los que el valor calculado es cero, la conclusión obvia es que no hay *affordability*, sin embargo, podría deberse también al tamaño del corpus de entrenamiento. A diferencia de muchas otras áreas de la ingeniería, en robótica no existe un corpus de tareas simples para determinados escenarios, por lo que se tuvo que crear uno. El corpus utilizados para obtener estos datos consta de tareas simples.

5.2.2. Experimento B: Verificación de *affordability*

Este experimento utilizó un modelo de *affordances* generado con 236 tareas simples. El conjunto de pruebas consistió en diez tareas simples, que se sabía previamente, no cumplían con el *affordability* entre el verbo y el objeto (Ver figura 4.5).

Los resultados obtenidos mostraron una probabilidad condicional de cero para estas tareas simples. Por tanto, el sistema de mapeo pudo, para todo el conjunto de pruebas, determinar la métrica correctamente. Esto viene a confirmar la parte de la hipótesis planteada inicialmente en relación al *affordability*.

5.3. Factibilidad

La simulaciones se realizaron tal cual fueron descritas en la sección anterior. La variable de respuesta se graficó para dos y tres controladores (dos niveles) y para tres diferentes valores de penalidad-incentivo (tres niveles). En total se hicieron seis simulaciones las cuales corres-

ponden a las figuras: 5.4, 5.5, 5.6, 5.7, 5.8 y 5.9.

El factor de ruido tiene un papel predominante en los experimentos porque el *affordability* y las restricciones son generadas en forma aleatoria. A pesar del factor de ruido, un solo controlador tiende a tener una valor más alto, tal como se puede observar en todos los experimentos y gráficas realizadas.

Un peso más alto implica que el controlador en cuestión ejecuta la acción de manipulación de mejor manera. El peso de los controladores es la forma en que se han logrado considerar las experiencias pasadas en el cálculo de la factibilidad.

Los experimentos realizados también muestran una característica muy significativa en la función de costo y los factores fijos: la función de costo parecer ser más dependiente del peso que de las restricciones, esto después de un determinado número de iteraciones. Si la factibilidad está determinada por los pesos, el cálculo en cuestión podría estar descartando cualquier información en relación al objeto, el contexto y el cuerpo del robot, lo cual es una preocupación importante.

La ecuación 3.17 tiene el defecto de que multiplica las restricciones por un número entre cero y uno, lo que se denomina *affordability*, lo cual disminuye el impacto de los cambios en el ambiente, el robot y el objeto. La función de costo fue establecida como el punto de inicia, pero más análisis es necesario para determinar una función de costo que pueda seleccionar el controlador más adecuado para ejecutar la tarea simple.

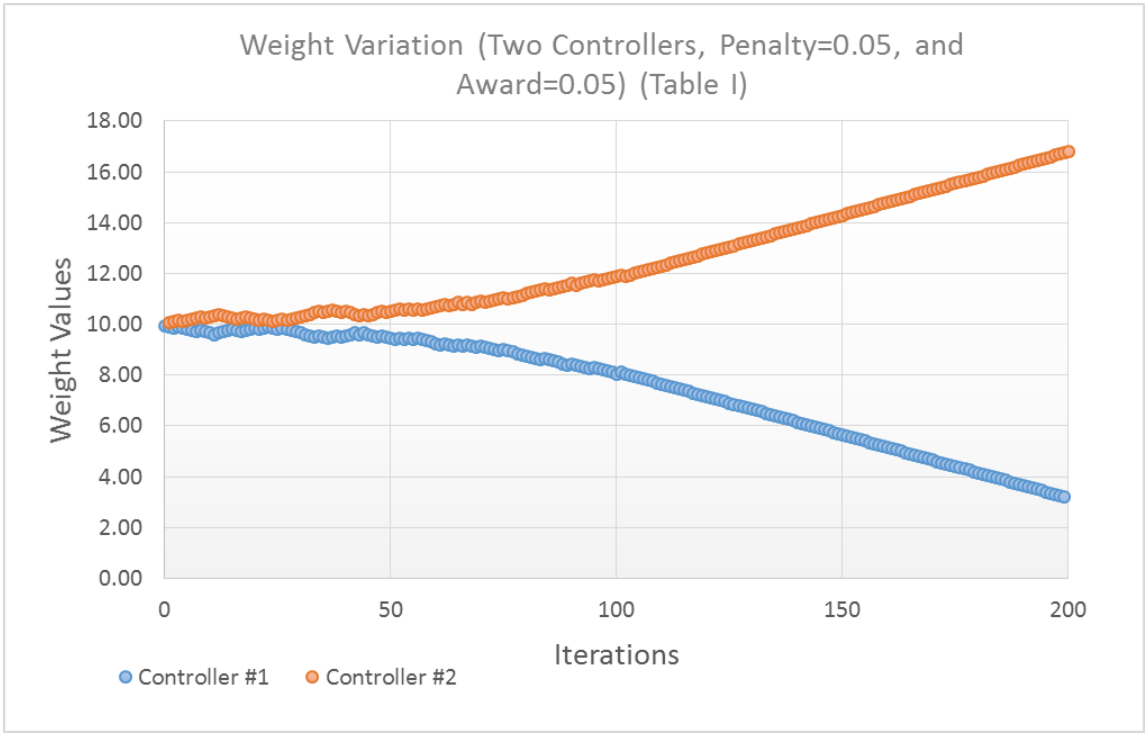

Figura 5.4: Variación de los pesos para dos controladores y un valor de penalización-incentivo de 0.05.

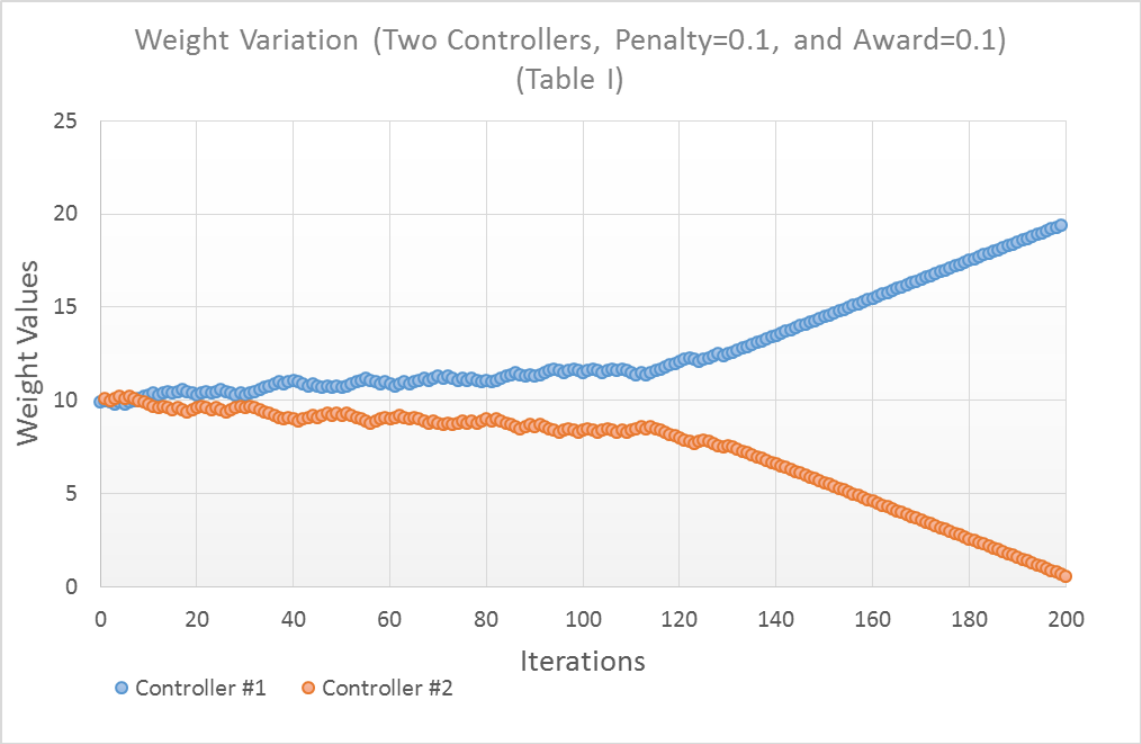

Figura 5.5: Variación de los pesos para dos controladores y un valor de penalización-incentivo de 0.1.

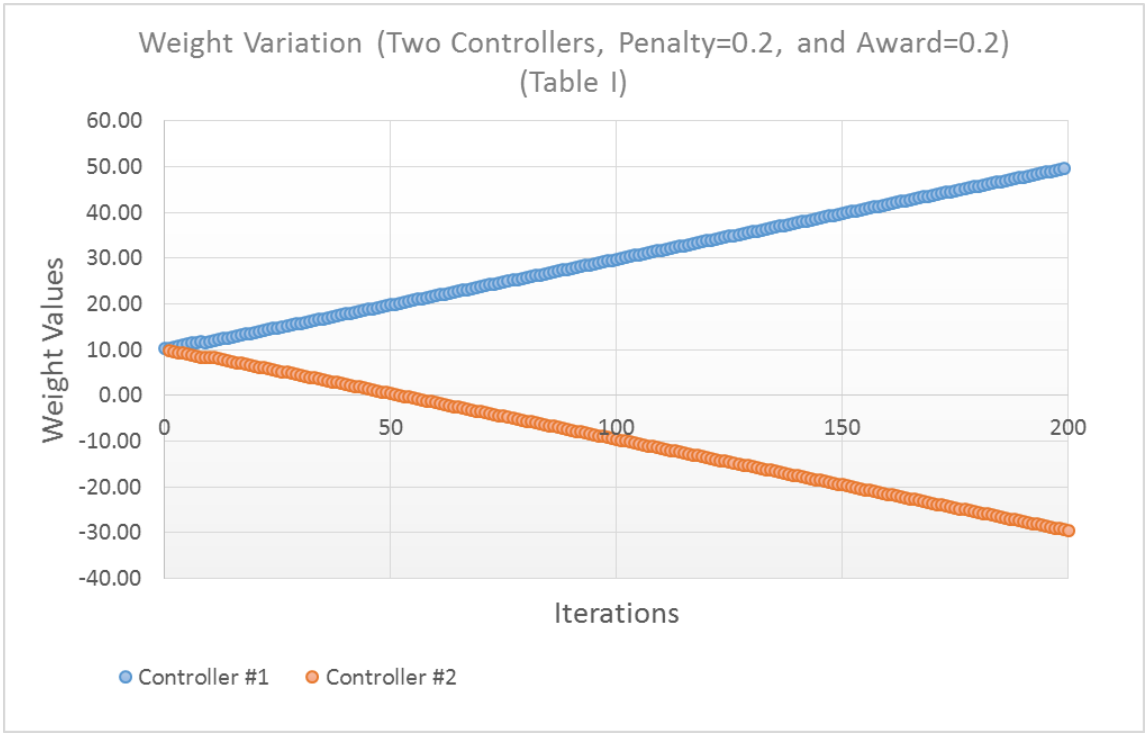

Figura 5.6: Variación de los pesos para dos controladores y un valor de penalización-incentivo de 0.2.

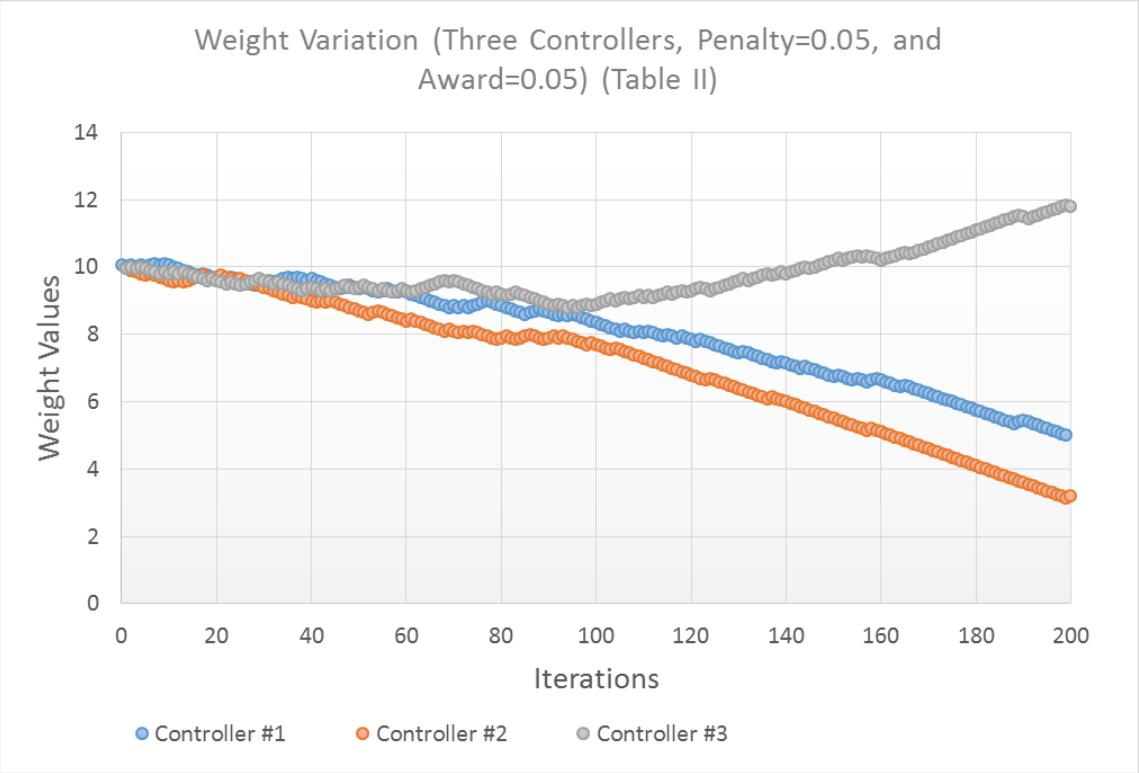

Figura 5.7: Variación de los pesos para tres controladores y un valor de penalización-incentivo de 0.05.

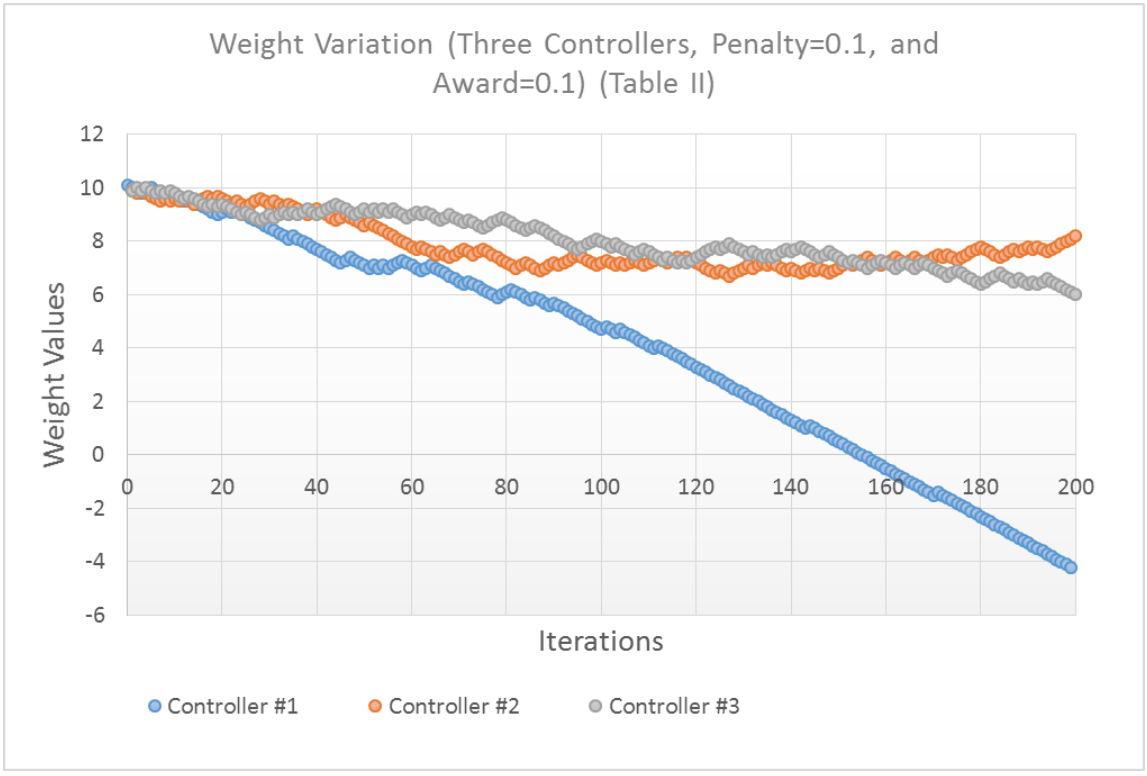

Figura 5.8: Variación de los pesos para tres controladores y un valor de penalización-incentivo de 0.1.

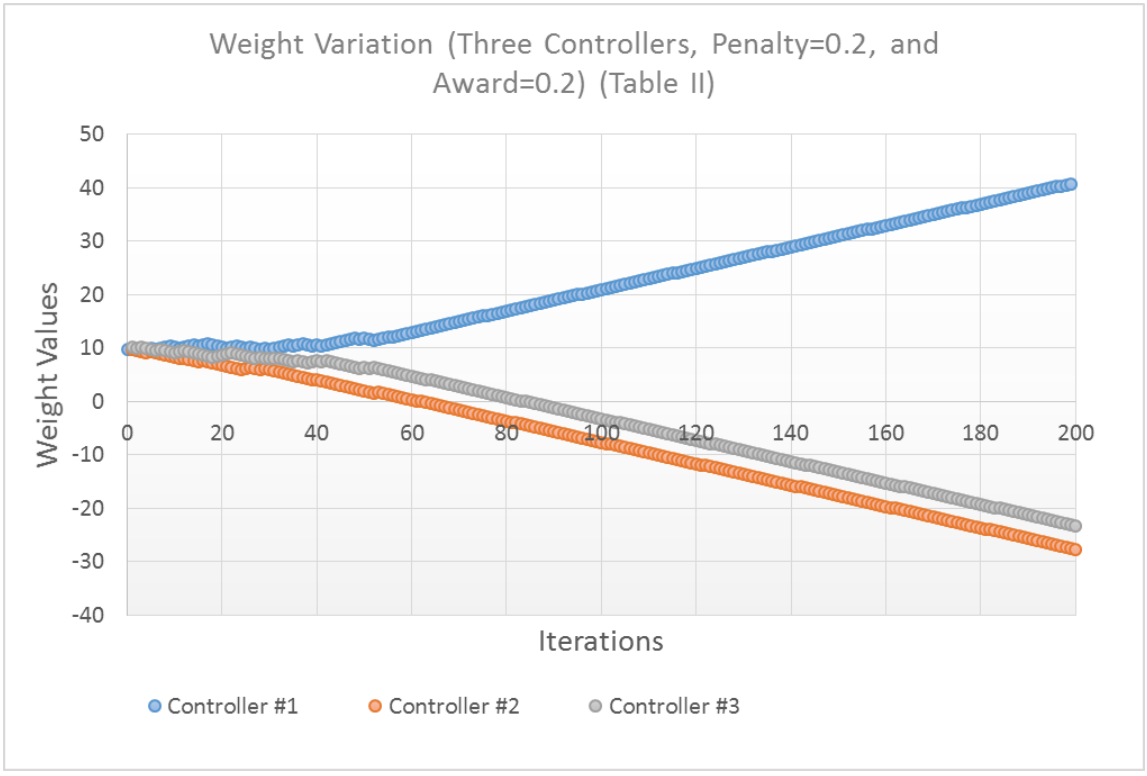

Figura 5.9: Variación de los pesos para tres controladores y un valor de penalización-incentivo de 0.2.

5.4. Experimentos finales

El conjunto de tareas simples utilizado como entrada al sistema se muestra en la figura 4.7. Este consiste de 16 tareas simples con las que se desea validar y probar la capa de interfaz. Este conjunto de pruebas se limitó al escenario correspondiente a la cocina.

La figura 5.10 muestra la ejecución de la tarea "Buscar la tostadora". El primero módulo que se ejecuta es el análizador léxico, el cual identifica la acción y el objeto, en este caso en particular, la acción es "buscar" y el objeto es "tostadora". Nótese que no se indentifica el adverbio ni el estado final porque no están dados en la instrucción simple, además de que los mismos no son necesarios.

La segunda unidad que se ejecuta es la unidad de mapeo, en este caso en particular esta unidad ha seleccionado un único controlador, el controlador número dos. A continuación también se ejecuta la unidad de *affordances*, la cual arroja un valor positivo, es decir, la unidad en cuestión ha confirmado que hay *affordability* entre la acción y el objeto.

A continuación, la unidad de factibilidad selecciona el controlador dos, esta escogencia resulta trivial cuando la unidad de mapeo, previamente, ha escogido un único controlador. La unidad de propiedades del objeto proporciona las características del objeto, en este caso en particular, el peso y el coeficiente de fricción.

Por último, se calcula la referencia a ser usada por el controlador,

en este caso en particula la referencia indica la ubicación espacial de la tostadora.

```
Performing lexical analysis ..................
Action: BUSCAR
Object: TOSTADORA
Adverb: dummy_adverb
Final state: dummy_state
..............................................
Performing mapping to select controllers .....
List of controllers: [u'2']
..............................................
Performing affordance analysis ...............
Affordability: TRUE
..............................................
Performing feasibility analysis ..............
Selected controller: 2
..............................................
Getting properties for the object .............
Object properties: Weigth: 5.00 kg, Friction:1.01
..............................................
Calculating control reference .................
Reference: [u'0.0', u'0.0', u'0.0']
..............................................
Outcome of the analysis.......................
..............................................
..............................................
Control parameters: Weigth: 5.00 kg, Friction:1.01
Control reference: [u'0.0', u'0.0', u'0.0']
Object: TOSTADORA
Controller: 2
```

Figura 5.10: Salidas generadas por la capa de interfaz para la tarea simple “Buscar la tostadora”.

```
Performing lexical analysis ..................
Action: INTRODUCIR
Object: TROZO_DE_PAN
Adverb: dummy_adverb
Final state: TOSTADORA
..............................................
Performing mapping to select controllers .....
List of controllers: []
..............................................
Performing affordance analysis ...............
Affordability: TRUE
..............................................
Performing feasibility analysis ..............
Selected controller: No controllers available.
..............................................
Getting properties for the object .............
Object properties: Weigth: 0.10 kg, Friction:0.01
..............................................
Calculating control reference ................
Reference: [u'1.25', u'7.88', u'7.42']
..............................................
Outcome of the analysis.......................
..............................................
..............................................
Control parameters: Weigth: 0.10 kg, Friction:0.01
Control reference: [u'1.25', u'7.88', u'7.42']
Object: TROZO_DE_PAN
Controller: No controllers available.
```

Figura 5.11: Salidas generadas por la capa de interfaz para la tarea simple “Introducir el trozo de pan #1 en la tostadora”.

Conclusiones y recomendaciones

Un robot, para que pueda ejecutar tareas de alta complejidad, debe estar equipado con una cantidad importante de conocimiento en la forma de modelos de objetos. De acuerdo al análisis realizado en el escenario cocina, las acciones más comunes son del tipo "agarrar y colocar". En el caso de la mesa de trabajo, las acciones de manipulación más comunes son verbos más específicos relacionados con las tareas complejas seleccionadas, como lo son: lijar, limpiar y taladrar. El poder implementar estos modelos de objetos tiene el potencial de permitir la construcción de robots más autónomos y flexibles, que pueda ejecutar actividades humanas cada vez más complejas (Calli y cols., 2015).

Debido a la limitada cantidad de tareas complejas y tareas simples, estos resultados no puede ser generalizados. Aún así, estos proporcionan información relevante que puede ser usada como guía para diseñar y construir software y hardware de robots humanoides. El porcentaje de uso de verbos y objetos puede ser usado para justificar investigación en modelos de objetos específicos y mejor las características mecánicas de las articulaciones robóticas.

Por ejemplo, la investigación futura en modelos de objetos debería enfocarse en la acción "verter", ya que es uno de los verbos más usados en la cocina, aparte de las acciones del tipo "agarrar y colocar". Las articulaciones robóticas, particularmente, las que estarían destinadas a una cocina, deberían poder realizar esta compleja acción de manipu-

lación, que requiere la manipulación de dos objetos al mismo tiempo: el contenedor donde es vertido el fluido y el contenedor que está siendo vaciado (Feix y cols., 2014).

Basado en el análisis que se realizó, la cocina muestra menos variabilidad entre tareas complejas que el escenario de la mesa de trabajo. Esto podría sugerir para la cocina, en el contexto de las tareas complejas analizadas, que podría ser más sencillo y barato construir un diccionario, un modelo de objetos y una arquitectura robótica, en comparación con la mesa de trabajo. En el caso de la mesa de trabajo, hay mayor diversidad de objetos y verbos, en comparación con la cocina.

Este análisis debería ser ampliado para incluir más escenarios y más tareas complejas. Una estudio más exhaustivo podría realizar planteamientos más generales que permitan construir robots para un conjunto más amplio de escenarios, tales como: una sala de una casa, un jardín, una oficina o un laboratorio (de cualquier tipo). Adicionamente, es aún desconocido como las características del escenario cambian el porcentaje de uso o la variabilidad de los verbos y objetos.

El análisis del corpus de tareas simples fue el punto de partida para desarrollar un lenguaje de acción que se pudiera utilizar como enlace entre el alto nivel y el bajo nivel de la arquitectura robótica. El lenguaje que se decidió utilizar, como entrada a la capa de interfaz, es un subconjunto del lenguaje natural, pues posee un limitado número de instancias y se organiza en trigramas.

Un diseño de una arquitectura robótica fue propuesto con el fin de procesar y dilucidar este lenguaje de acción, de tal forma que fuera inteligible para el sistema de modelo de objetos. El elemento más importante en la arquitectura es la capa de interfaz la cual consiste en cinco módulos: la unidad de *affordances*, la unidad de mapeo, la unidad de determinación de la referencia, la unidad de propiedades de objetos, y el analizador de factibilidad.

La arquitectura fue analizada para mostrar como puede resolver una instrucción en lenguaje natural, a través de sus diferentes capas, desde el alto nivel hasta el bajo nivel, a través de dos **casos de uso**: "Hágame un desayuno" y "Hágame una ensalada". En concreto, se analizaron dos tareas simples: "Coloque la tostada sobre el plato" y "Rallar 20 gramos de lechuga". Estas dos tareas mostraron la complejidad de la implementación de los módulos de la capa de interfaz y dieron una idea más clara del camino a seguir en su desarrollo.

Cada módulo de la capa de interfaz es un reto significativo ya que su diseño e implementación no está completamente acabada. En la capa de interfaz se obtuvieron resultados muy valiosos para dos módulos en particular: la unidad de *affordances* y la unidad factibilidad.

La unidad de *affordances* se implementó al crear un modelo de probabilidad condicional. Los datos de entrenamiento se basaron en 236 tareas simples obtenidas en el escenario cocina (Ver figura 4.2) a partir de un conjunto de tareas complejas. Estas 236 tareas simples fueron generadas por un ser humano, el cual usó como referencia el

escenario tal cual fue especificado.

Esta unidad permitió construir una tabla con los cinco verbos más utilizados y los cinco objetos más manipulados; a esta tabla se le denominó: matriz de *affordances* (Ver tabla III). En esta matriz, si la probabilidad condicional para un bigrama es cero, la conclusión inmediata es que no hay *affordability* entre la acción y el objeto, sin embargo, también podría deberse a que el corpus de entrenamiento no es los suficiente grande.

Esta matriz tiene el potencial de ser usada como una forma muy efectiva de modelar y comparar ambientes dinámicos. Es decir, de servir como un complemento semántico a las más tradicionales descripciones físicas del mundo.

Para analizar la interacción entre la unidad de mapeo y la unidad de análisis de factibilidad, se realizaron varias simulaciones. El diagrama de flujo correspondiente a estas simulaciones se muestra en la figura 4.6. Los gráficos obtenidos revelan las siguientes aspectos relevantes:

- A pesar de los factores de ruido, el peso asociado a un único controlador tiende a converger hacia un valor mayor y otro controlador converge hacia un valor menor. Esto se puede observar en la figuras 5.4, 5.5, 5.6, 5.6, 5.7, 5.8 y 5.9. Nótese que hay un momento después del cual los puntos comienzan a diverger el uno del otro.
- La función de costo, bajo las condiciones expuestas y después de

cierto número de iteraciones, está siendo mayormente determinada por el peso y no por las restricciones. Por tanto, el cálculo de factibilidad podrían estar descartando información relevante relacionada al contexto en el que debe operar el robot.

- Las oscilaciones en los gráficos se debe a los valores aleatorios de las restricciones. El comportamiento lineal se debe al incremento o decremento de los pesos; lo cual explica porqué una gráfica con un mayor valor incentivo-penalidad muestra menos variaciones.

La función de costo y los valores límites usandos en las simulaciones podrían no ser apropiados, así que se requiere investigación adicional para encontrar una ecuación o un algoritmo que pueda seleccionar el controlador más apto para ejecutar la tarea simple.

El trabajo futuro debería enfocarse en cómo diseñar, estudiar e implementar los diferentes subsistemas, así como todas las interacciones entre ellos. En relación a la unidad de *affordances*, se requiere un corpus más amplio, así como la incorporación de más escenarios. En relación a la unidad de factibilidad, es necesario evaluar varios algoritmos para encontrar el más adecuado.

Referencias

Beetz, M., Stulp, F., Radig, B., Bandouch, J., Blodow, N., Dolha, M., ... Tenorth, M. (2008, Aug). The assistive kitchen - a demonstration scenario for cognitive technical systems. En *Ro-man 2008 - the 17th ieee international symposium on robot and human interactive communication* (p. 1-8). doi: 10.1109/ROMAN .2008.4600634

Belle, H. J., Vaishak y Levesque. (2014). Prego: An action language for belief-based cognitive robotics in continuous domains. En C. E. Brodley y P. Stone (Eds.), *Aaai* (p. 989-995). AAAI Press.

Blodow, N., Goron, L. C., Marton, Z. C., Pangercic, D., Rühr, T., Tenorth, M., y Beetz, M. (2011, Sept). Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments. En *2011 ieee/rsj international conference on intelligent robots and systems* (p. 4263-4270). doi: 10.1109/IROS.2011.6094665

Calli, B., Walsman, A., Singh, A., Srinivasa, S., Abbeel, P., y Dollar, A. (2015). Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. En (Vol. 22, p. 36 - 52).

Costa, C. M., Veiga, G., Sousa, A., y Nunes, S. (2017, April). Evaluation of stanford ner for extraction of assembly information from instruction manuals. En *2017 ieee international conference on autonomous robot systems and competitions (icarsc)* (p. 302-309). doi: 10.1109/ICARSC.2017.7964092

Coste-Maniere, E., y Simmons, R. (2000). Architecture, the backbone of robotic systems. En *Proceedings 2000 icra. millennium conference. ieee international conference on robotics and automation. symposia proceedings (cat. no.00ch37065)* (Vol. 1, p. 67-72 vol.1). doi: 10.1109/ROBOT.2000.844041

Cousins, S. (2011, Agosto). Challenges of building personal robots. En *2011 ieee hot chips 23 symposium (hcs)* (p. 1-36). doi: 10.1109/HOTCHIPS.2011.7477513

Domingos, P. (2007). What's missing in AI: The interface layer. *University of Washington, Washington, USA*.

Domingos, P., y Lowd, D. (2009). *Markov logic: An interface layer for artificial*

intelligence (1.ª ed.). Morgan and Claypool Publishers.

Fang, J., Zhao, J., He, F., y Lin, X. (2013, Agosto). Design and research of three-layers open architecture model for industrial robot software system. En *2013 ieee international conference on mechatronics and automation* (p. 104-109). doi: 10.1109/ICMA.2013.6617901

Feix, T., Bullock, I. M., y Dollar, A. M. (2014, Octubre). Analysis of human grasping behavior: Correlating tasks, objects and grasps. *IEEE Transactions on Haptics*, *7*(4), 430-441. doi: 10.1109/TOH.2014.2326867

Ferrein, A., Maier, C., Muehlbacher, C., Niemueller, T., Steinbauer, G., y Vassos, S. (2016). Controlling Logistics Robots with the Action-based Language YAGI. En *Proc. of 9th international conference on intelligent robotics and applications (icira2016).* Tokyo, Japan.

Gibson, J. (1977). The theory of affordances. En J. Shaw Robert y Bransford (Ed.), *Perceiving, acting, and knowing: Toward and ecological psychology* (pp. 62–82). Hillsdale, NJ: Erlbaum.

Glenberg, A. M., y Kaschak, M. P. (2002, 01 de Sep). Grounding language in action. *Psychonomic Bulletin & Review*, *9*(3), 558–565. Descargado de https://doi.org/10.3758/BF03196313 doi: 10.3758/BF03196313

Gorges, N., Schmid, A. J., Osswald, D., y Worn, H. (2007, Noviembre). A framework for creating, coordinating, and executing skills on a humanoid robot. En *2007 7th ieee-ras international conference on humanoid robots* (p. 385-391). doi: 10.1109/ICHR.2007.4813898

Guerra-Filho, G., y Aloimonos, Y. (2007, Mayo). A language for human action. *Computer*, *40*(5), 42-51. doi: 10.1109/MC.2007.154

Hoijer, H. (Ed.). (1904). *Language in culture : conference on the interrelations of language and other aspects of culture.* University of Chicago Press.

Hopcroft, J. E., Kearney, J. K., y Krafft, D. B. (1991). A case study of flexible object manipulation. *The International Journal of Robotics Research*, *10*(1), 41-50. doi: 10.1177/027836499101000105

James L. Melsa, D. L. C. (Ed.). (1978). *Decision and estimation theory.* McGraw-Hill.

Karim, H. A., Lokman, A. M., y Redzuan, F. (2016a, Agosto). Older adults pers-

pective and emotional respond on robot interaction. En *2016 4th international conference on user science and engineering (i-user)* (p. 95-99). doi: 10.1109/IUSER.2016.7857941

Karim, H. A., Lokman, A. M., y Redzuan, F. (2016b, Agosto). Older adults perspective and emotional respond on robot interaction. En *2016 4th international conference on user science and engineering (i-user)* (p. 95-99). doi: 10.1109/IUSER.2016.7857941

Krivic, S., Ugur, E., y Piater, J. (2016, Agosto). A robust pushing skill for object delivery between obstacles. En *2016 ieee international conference on automation science and engineering (case)* (p. 1184-1189). doi: 10.1109/COASE.2016.7743539

Lagrand, C., v. d. Meer, M., y Visser, A. (2016, May). The roasted tomato challenge for a humanoid robot. En *2016 international conference on autonomous robot systems and competitions (icarsc)* (p. 341-346). doi: 10.1109/ICARSC.2016.63

Levesque, H. J., Reiter, R., Lespérance, Y., Lin, F., y Scherl, R. B. (1997). Golog: A logic programming language for dynamic domains. *The Journal of Logic Programming*, *31*(1), 59 - 83. (Reasoning about Action and Change) doi: http://dx.doi.org/10.1016/S0743-1066(96)00121-5

Low, J. H., Lee, W. W., Khin, P. M., Thakor, N. V., Kukreja, S. L., Ren, H. L., y Yeow, C. H. (2017, Abril). Hybrid tele-manipulation system using a sensorized 3-d-printed soft robotic gripper and a soft fabric-based haptic glove. *IEEE Robotics and Automation Letters*, *2*(2), 880-887. doi: 10.1109/LRA.2017.2655559

Metta, G., Fitzpatrick, P., y Natale, L. (2006). Yarp: Yet another robot platform. *International Journal of Advanced Robotic Systems*, *3*(1), 8. Descargado de https://doi.org/10.5772/5761 doi: 10.5772/5761

Moldovan, B., Moreno, P., van Otterlo, M., Santos-Victor, J., y Raedt, L. D. (2012, May). Learning relational affordance models for robots in multi-object manipulation tasks. En *2012 ieee international conference on robotics and automation* (p. 4373-4378). doi: 10.1109/ICRA.2012.6225042

Moldovan, B., y Raedt, L. D. (2014, Sept). Learning relational affordance models for two-arm robots. En *2014 ieee/rsj international conference on intelligent robots and systems* (p. 2916-2922). doi: 10.1109/IROS.2014.6942964

Müller, A., Kirsch, R., y Beetz, M. (2007). Transformational planning for everyday activity. En *In 17th international* (pp. 248–255).

Nabeshima, H., Inoue, K., y Haneda, H. (2000). Implementing an action language using a sat solver. En *Proceedings 12th ieee internationals conference on tools with artificial intelligence. ictai 2000* (p. 96-103). doi: 10.1109/TAI.2000.889852

Nava, N. E. (2011). *Advanced mechanics in robotic systems*. Springer London.

Norman, D. A. (2002). *The design of everyday things*. New York, NY, USA: Basic Books, Inc.

Quack, B., Wörgötter, F., y Agostini, A. (2015, Setiembre). Simultaneously learning at different levels of abstraction. En *2015 ieee/rsj international conference on intelligent robots and systems (iros)* (p. 4600-4607). doi: 10.1109/IROS.2015.7354032

Rodríguez Consuegra, F. (2003). La filosofía del lenguaje: su naturaleza y su contexto. *Dianoia*, *48*(50), 41-68.

Romay, A., Kohlbrecher, S., Conner, D. C., y von Stryk, O. (2015, Noviembre). Achieving versatile manipulation tasks with unknown objects by supervised humanoid robots based on object templates. En *2015 ieee-ras 15th international conference on humanoid robots (humanoids)* (p. 249-255). doi: 10.1109/HUMANOIDS.2015.7363543

Ruiken, D., Liu, T. Q., Takahashi, T., y Grupen, R. A. (2016, Noviembre). Reconfigurable tasks in belief-space planning. En *2016 ieee-ras 16th international conference on humanoid robots (humanoids)* (p. 1257-1263). doi: 10.1109/HUMANOIDS.2016.7803431

Ruiz, F. (2014). *Compact models of objects for skilled manipulation* . Universidad Técnica de Múnich, Universidad Técnica de Múnich, Intelligent Autonomous Systems, Garching, Germany.

Ruiz, F., Cheng, G., y Beetz, M. (2010, Octubre). Prediction of action outcomes using an object model. En *2010 ieee/rsj international conference on intelligent robots and systems* (p. 1708-1713). doi: 10.1109/IROS.2010.5649552

Ruiz, F., Cheng, G., y Beetz, M. (2011, Octubre). Fast adaptation for effect-aware pushing. En *2011 11th ieee-ras international conference on humanoid robots*

(p. 614-621). doi: 10.1109/Humanoids.2011.6100863

Searle, S. R. (Ed.). (1982). *Matrix algebra useful for statistics*. Wiley Series.

Siciliano, B., y Khatib, O. (2007). *Springer handbook of robotics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Siciliano, B., y Khatib, O. (Eds.). (2008). *Springer handbook of robotics*. Springer.

Speck, D., Dornhege, C., y Burgard, W. (2017, Abril). How much does it take to redo shakey the robot? *IEEE Robotics and Automation Letters*, *2*(2), 1203-1209. doi: 10.1109/LRA.2017.2665694

Sterling, L., y Shapiro, E. (1986). *The art of prolog: Advanced programming techniques*. Cambridge, MA, USA: MIT Press.

Stulp, F., Kresse, I., Maldonado, A., Ruiz, F., Fedrizzi, A., y Beetz, M. (2009). Compact models of human reaching motions for robotic control in everyday manipulation tasks. En *Proceedings of the 8th international conference on development and learning (icdl).*

Sugiura, K., y Iwahashi, N. (2007). Learning object-manipulation verbs for human-robot communication. En *Proceedings of the 2007 workshop on multimodal interfaces in semantic interaction* (pp. 32–38). New York, NY, USA: ACM. doi: 10.1145/1330572.1330577

Suárez, F., y Ruiz, F. (2018, July). Analysis of atomic manipulation tasks in human-shared scenarios (a kitchen and a collaborative workshop table) for a humanoid robot. En *2018 ieee international work conference on bioinspired intelligence (iwobi)* (p. 1-9). doi: 10.1109/IWOBI.2018.8464220

Suárez, F., y Ruiz, F. (2019, Feb). Automatic translation of spanish natural language commands to control robot comands based on lstm neural network. En *2019 third ieee international conference on robotic computing (irc)* (p. 125-131). doi: 10.1109/IRC.2019.00026

Tenorth, M., y Beetz, M. (2010). Priming transformational planning with observations of human activities. Universidad Técnica de Múnich, Intelligent Autonomous Systems, Garching, Germany.

Tenorth, M., Nyga, D., y Beetz, M. (2010, May). Understanding and executing instructions for everyday manipulation tasks from the world wide web. En *2010 ieee international conference on robotics and automation* (p. 1486-1491).

doi: 10.1109/ROBOT.2010.5509955

Thenmozhi, D., Seshathiri, R., Revanth, K., y Ruban, B. (2017, Jan). Robotic simulation using natural language commands. En *2017 international conference on computer, communication and signal processing (icccsp)* (p. 1-4). doi: 10 .1109/ICCCSP.2017.7959814

Winkler, J., y Beetz, M. (2015, Sept). Robot action plans that form and maintain expectations. En *2015 ieee/rsj international conference on intelligent robots and systems (iros)* (p. 5174-5180). doi: 10.1109/IROS.2015.7354106

Yamazaki, K., Watanabe, Y., Nagahama, K., Okada, K., y Inaba, M. (2010, Dec). Recognition and manipulation integration for a daily assistive robot working on kitchen environments. En *2010 ieee international conference on robotics and biomimetics* (p. 196-201). doi: 10.1109/ROBIO.2010.5723326

Zhang, C., y Shah, J. A. (2016, Octubre). Co-optimizing task and motion planning. En *2016 ieee/rsj international conference on intelligent robots and systems (iros)* (p. 4750-4756). doi: 10.1109/IROS.2016.7759698

Zhang, S., Yang, F., Khandelwal, P., y Stone, P. (2015). Mobile robot planning using action language. En F. Calimeri, G. Ianni, y M. Truszczynski (Eds.), *Logic programming and nonmonotonic reasoning: 13th international conference, lpnmr 2015, lexington, ky, usa, september 27-30, 2015. proceedings* (pp. 502–516). Cham: Springer International Publishing. doi: 10.1007/978-3-319-23264-5_42