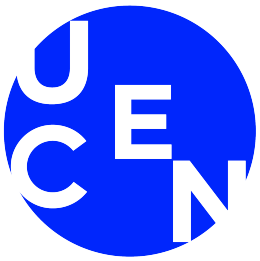


UNIVERSIDAD CENTRAL DE CHILE
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA

Agente de aprendizaje por refuerzo con retroalimentación interactiva.

Memoria para optar al título profesional de
Ingeniero Civil en Computación e Informática.

Profesor Guía: **Francisco Javier Cruz Naranjo**
Profesor Informante: **Claudio Alex Henríquez Berroeta**
Profesor Informante: **Alejandro Antonio Sanhueza Olave**

Rubén Alejandro Contreras Ortiz

Santiago, Chile
2020

UNIVERSIDAD CENTRAL DE CHILE
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA

Agente de aprendizaje por refuerzo con retroalimentación interactiva.

Memoria para optar al título profesional de
Ingeniero Civil en Computación e Informática.

Profesor Guía: **Francisco Javier Cruz Naranjo**
Profesor Informante: **Claudio Alex Henríquez Berroeta**
Profesor Informante: **Alejandro Antonio Sanhueza Olave**

QUIENES RECOMIENDAN QUE SEA ACEPTADA PARA COMPLETAR
LAS EXIGENCIAS DEL TÍTULO DE INGENIERÍA CIVIL EN
COMPUTACIÓN E INFORMÁTICA

Rubén Alejandro Contreras Ortiz

Santiago, Chile
2020

Esta tesis está dedicada a mi familia
Rubén Contreras O.

Resumen

A medida que pasa el tiempo, surge una dependencia clara de la tecnología, la cual está inmersa en la vida diaria de la mayoría de las personas, algunos han llamado este fenómeno como una revolución digital (Carmona, 1972). Esta dependencia, sucede con distintos tipos de tecnologías. Una de las que ha dado que hablar las últimas décadas, ha sido el aprendizaje de agentes de forma autónoma.

El aprendizaje por refuerzo (Sutton and Barto, 2018) es una estrategia de entrenamiento de un agente autónomo, el cual obtiene conocimiento en base de la táctica "prueba-error". A su vez ésta tiene ciertas variantes, pero la que será ocupada en el proyecto es aprendizaje por refuerzo combinado con una retroalimentación interactiva (Cruz et al., 2018).

El proyecto consiste en crear un agente de aprendizaje por refuerzo con retroalimentación interactiva, el cual será entrenado para aprender a cruzar un ambiente simulado doméstico, como por ejemplo dentro de un laberinto, con obstáculos, etc. Posterior al entrenamiento el agente será capaz de manejar una unidad aérea no tripulada (Valavanis and Vachtsevanos, 2015), o también conocido como drone. La retroalimentación interactiva será a través de órdenes por medio de comandos de voz. A su vez se aplicarán métodos nuevos, utilizando simultáneamente reward-shaping y policy-shaping.

Debido a lo mencionado antes, durante el transcurso de esta tesis se hablará de dos grandes experimentos, el primero está relacionado al reconocimiento de comandos de voz para ejecutar órdenes de movimiento en un dron, donde se ocuparon técnicas de coincidencia de fonemas tanto en inglés y español. El reconocimiento de la voz a la acción mejoró para ambos idiomas con coincidencia de fonemas en comparación con el uso exclusivo del algoritmo basado en la nube sin instrucciones basadas en el dominio. Al usar entradas de audio sin procesar, el enfoque basado en la nube logra una precisión del 74,81 % y del 97,04 % para las instrucciones en inglés y español, respectivamente. Sin embargo, con nuestro enfoque de coincidencia de fonemas,

los resultados mejoran, con una precisión del 93,33 % en inglés y del 100,00 % en español.

El segundo experimento está relacionado con el aprendizaje por refuerzo interactivo, enfocándose más en los resultados de la combinación de policy-shaping con reward-shaping, obteniendo resultados positivos a otras tecnicas de aprendizaje por refuerzo. Considerando el tiempo de ejecución de los algoritmos, se puede concluir que la combinación de las dos técnicas de aprendizaje por refuerzo interactivo ayudan al aprendizaje del agente, incluso más que éstas implementadas por separado.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En los últimos años el aprendizaje automático (machine learning en inglés) se ha apoderado del mercado de las ciencias de datos, todo con el fin de resolver problemas pertenecientes a la vida cotidiana de las personas, ya no como instrucciones programadas, sino con la máquina aprendiendo de forma autónoma una tarea específica (Dans, 2019).

Desde aproximadamente unos 10 años, los vehículos aéreos no tripulados (del tipo drone) han sido más solicitados por el mercado en general. Aunque éstos realmente se crearon hace muchas décadas atrás, aproximadamente en el año 1916 fue el profesor Archibald Low trabajando para el Ministerio del Aire británico realizó un proyecto el cual tenía como fin desarrollar defensas contra los dirigibles alemanes (Haluani, 2014). Actualmente en el siglo XXI los drones tienen un rol importante en muchas áreas, como por ejemplo uso militar, agricultura, recreación, etc.

1.2. Definición del Problema

Durante muchos años, la tecnología relacionada a robots fue enfocada solamente en el área industrial. En las últimas décadas el gran crecimiento tecnológico, ya sea en ámbitos de capacidad de hardware, o en software, han logrado expandir el abanico de aplicación de la robótica como por ejemplo en la industria de la agricultura (Sánchez, 2018).

Según lo mencionado anteriormente, el avance tecnológico va de la mano con el

crecimiento de conocimiento en el área, ya que cada cierto tiempo van apareciendo ideas nuevas de cómo resolver problemas. Hace algunos años que se está mencionando la estrategia de aprendizaje por refuerzo para la enseñanza autónoma de agentes robóticos, ejemplos de agentes de aprendizaje por refuerzo son (Mnih et al., 2013) o (Yu et al., 2018).

A su vez, un tema que está en la palestra el último tiempo es el uso de vehículos aéreos no tripulados o UAV's (unmanned aerial vehicle proveniente del inglés), ya sea por razones de uso personal, policial, militar, etc (Valavanis and Vachtsevanos, 2015). Este aparato es muy práctico, ya que permite sobrevolar un área mientras se graba una imagen de ésta (en algunos casos con imágenes transmitidas en tiempo real). Por lo tanto es un artefacto que genera un grado de recreación tal, que permite que el usuario se sienta libre, porque sin mayor esfuerzo puede recorrer grandes superficies de terreno por medio de un control remoto (Valavanis and Vachtsevanos, 2015), pero ¿Qué sucede con las personas que tienen amplios problemas de movilidad? o las personas que no sienten comodidad a la hora de utilizar un control, en este sentido, utilizar comandos de voz es mucho más natural para una persona.

El objetivo de este proyecto se basa en enseñar a un agente de aprendizaje por refuerzo a manejar un drone dentro de un ambiente simulado, complementando técnicas de aprendizaje por refuerzo interactivo como lo son reward shaping y policy shaping, midiendo su eficiencia en el entrenamiento, y comparando estos resultados con los métodos utilizados por separado. De esta forma, la gente que no tiene la capacidad para manejar directamente un dron, tendrá una posibilidad novedosa de hacerlo.

1.3. Objetivos

1.3.1. Objetivo General

Implementar un agente de aprendizaje por refuerzo integrando técnicas de aprendizaje por refuerzo interactivo, con el fin de entrenar un agente de aprendizaje por refuerzo que maneje un drone en un ambiente simulado.

1.3.2. Objetivos Específicos

En el presente trabajo, se han definido los siguientes objetivos específicos:

- Investigar sobre la actualidad del aprendizaje por refuerzo, a su vez de sus respectivas variantes.

- Diseñar un entorno simulado relacionado al problema, el cual consiste en un vehículo aéreo no tripulado capaz de recibir y ejecutar las instrucciones entregadas por el agente.

- Desarrollar agentes utilizando la técnica aprendizaje por refuerzo interactivo con la finalidad de que manejen un drone en un ambiente simulado a través de instrucciones básicas.

- Analizar los resultados obtenidos por los agentes, según su recompensa promedio y otras herramientas estadísticas.

- Medir mejora en el entrenamiento al utilizar policy y reward shaping (tiempos de entrenamiento, y recompensas promedio).

1.4. Hipótesis

Como anteriormente se mencionó, el aprendizaje por refuerzo tiene distintas variantes, y la estrategia que se ocupará será la mezcla de distintas sub-áreas de RL. Por otro lado está el aprendizaje por refuerzo con retroalimentación interactiva (Cruz et al., 2018), el cual consiste en que cada vez que el agente autónomo interactúe o intente interactuar con el entorno, un agente externo va a tener cierta probabilidad de intervención, ya sea para guiar qué acción debe tomar o para indicar si la decisión que tomó fue acertada o no.

La hipótesis de trabajo se centrará en sí un agente de aprendizaje por refuerzo interactivo es capaz de mejorar el rendimiento de aprendizaje de un agente autónomo, a la hora de tratar de identificar qué instrucciones debe tomar para manejar en un entorno determinado. Todo esto con el fin de controlar una UAV en un ambiente simulado. Con la intención de validar esta suposición surge unas interrogantes ¿El agente será capaz de realizar su recorrido dentro del entorno simulado ocupando distintas estrategias de aprendizaje por refuerzo interactivo (específicamente la combinación de "Policy-shaping" y "Reward-shaping")?, dependiendo de la respuesta anterior, ¿Es confiable la interpretación del agente?

1.5. Metodología de Trabajo

Conforme a la hipótesis planteada anteriormente, la metodología de trabajo será la del método científico (Nola and Sankey, 2014) tal como se muestra en la figura 1.1.

- **Planteamiento del Problema**: Un UAV simulada, a su vez un agente de RL interactivo con feedback a través de la voz para entrenar un agente que maneje instrucciones básicas para manejar dicha unidad aérea ¿Será capaz el agente de interpretar de manera correcta las órdenes? si es así, ¿Qué grado de efectividad tiene al deducirlas?
- **Examen y análisis de los enfoques existentes**: Se realizará un estudio bibliográfico respecto a los enfoques existentes relacionados con el proyecto, o también conocido como estado del arte.
- **Construcción del escenario experimental**: Para el desarrollo de este punto se utilizará un software capaz de simular robots en un ambiente controlado, esto con el fin de poner a prueba las órdenes entregadas al drone por medio del agente. También se desarrollarán algoritmos para interpretar dispositivos de entrada para el feedback que recibirá el agente.
- **Revisión de resultados y análisis**: A partir de una serie de pruebas que se harán durante el desarrollo del proyecto, se generará una serie de resultados, los cuales serán sometidos a un análisis exhaustivo, principalmente relacionado a la precisión de predicción que tiene el agente.
- **Resultados del informe**:Toda la información recopilada durante el desarrollo del proyecto, será plasmada en la tesis, la cual será en el formato solicitado por la escuela.

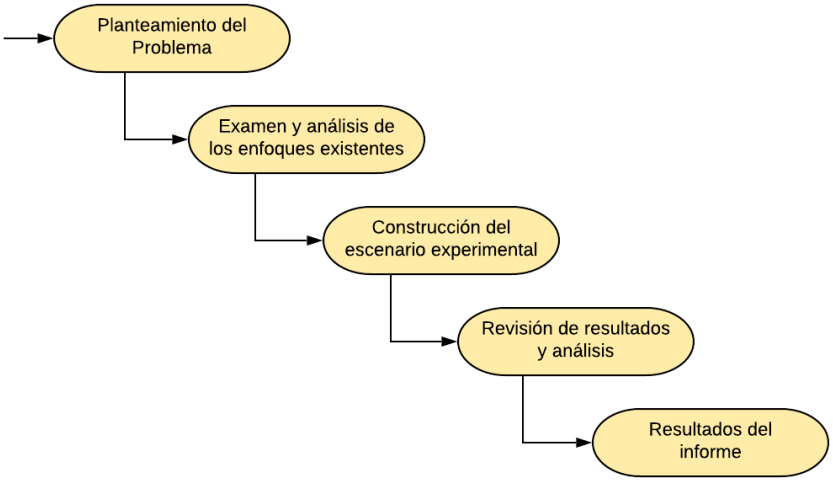


Figura 1.1: Los cinco pasos llevados a cabo en el método científico. Adaptada desde (Nola and Sankey, 2014).

Capítulo 2

Marco Teórico y Estado del Arte

En este apartado, se explicará la documentación y conocimientos previos necesarios para abordar la investigación. El tema a tratar, puede ser separado en al menos cuatro puntos, los cuales son aprendizaje por refuerzo, aprendizaje por refuerzo interactivo, interfaces naturales de usuarios y control de voz en drones.

2.1. Aprendizaje por Refuerzo

Aprendizaje por refuerzo (RL proveniente del inglés Reinforcement Learning) es un enfoque de la ciencia del machine learning. Este es un modelo de aprendizaje conductual, por el cual el algoritmo aprende a través de la prueba y error, a raíz de esto se la otorga una retroalimentación sobre sus acciones lo cual permite identificar si el agente está tomando buenas decisiones o no. Este tipo de algoritmos busca simular la naturaleza del aprendizaje y cómo se pone en práctica ésta en seres cognitivos. Uno de los puntos más importantes a destacar, es cómo interactúa el agente con su entorno, a medida que lo hace va generando información o mejor dicho conocimiento, el cual lo ayuda para la toma de decisiones más adelante, todo esto se puede apreciar en la figura 2.1. Por ejemplo, si un humano (o agente) interactúa con el entorno para cumplir el objetivo de martillar un clavo, el individuo al realizar una acción tendría su retroalimentación, ya sea porque lo hizo de buena manera, lo cual le generaría cierta sensación de placer, o por el contrarío, si su objetivo no es logrado pegándose en el dedo, obteniendo una retroalimentación negativa que sería el dolor. De tal manera el agente va rescatando información de todas sus acciones para cada vez lograr de mejor forma su cometido.

Hablando de un agente de aprendizaje reforzado se pueden identificar cuatro ele-

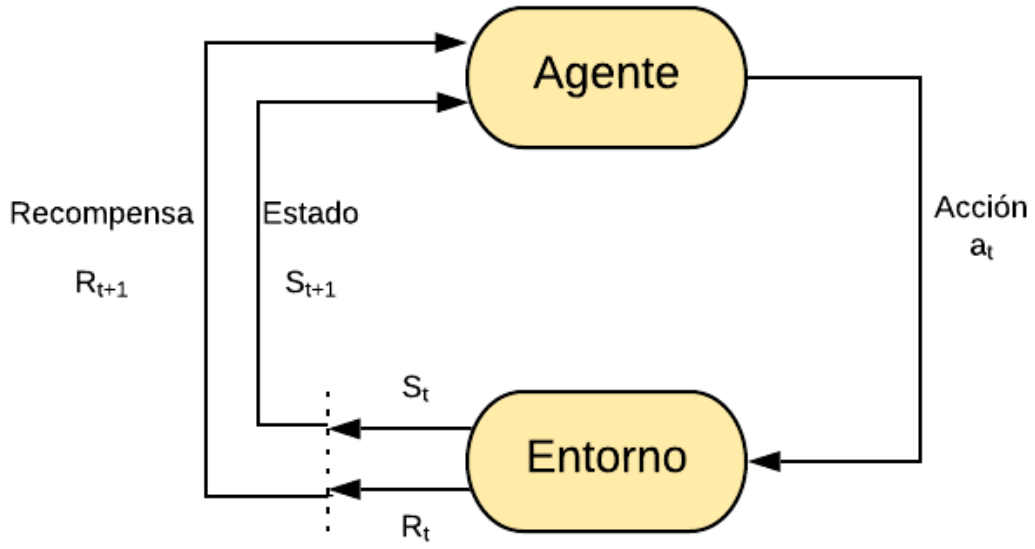


Figura 2.1: Interacción entre agente y entorno. El agente realiza una acción A^t, encontrándose en un estado S^t, logrando un nuevo estado S^{t+1} y una recompensa numérica para el agente respecto a la acción tomada (Sutton and Barto, 1998).

mentos básicos para que se pueda desarrollar de buena forma, los cuales serán expuestos a continuación:

1. **Política**: Ésta define el comportamiento del agente, esto sin importar el estado en que se encuentre. Puede ser representada a través de un método tabular o por métodos computacionales más avanzados como redes neuronales artificiales.

2. **Señal de recompensa**: La señal de recompensa es un valor numérico entregado al agente, inmediatamente después de que éste realice una acción. Esta señal tiene que definir la meta del agente y uno de los objetivos principales de éste es maximizar la acumulación de recompensa.

3. **Función de valor**: Ésta define qué tan buena fue la acción realizada por el agente en determinado estado y/o acción, es representada a través de un valor numérico, el cual es calculado utilizando distintos parámetros.

4. **Modelo del entorno (opcional)**: Éste puede ayudar a predecir estados y recompensas futuras. Todo esto a través de la imitación del entorno y con la finalidad de maximizar las recompensas futuras al tomar las mejores

decisiones.

2.1.1. Proceso de Decisión Markoviano

Los procesos de decisión de Markov o MDP (Markov Decision Processes en inglés) son los cimientos de las tareas de aprendizaje por refuerzo. En un MDP, las transiciones y recompensas dependen solo del estado actual y de la acción seleccionada por el agente(Puterman, 2014). Escrito de otra forma, un estado de Markov contiene toda la información relacionada con la dinámica de una tarea, esto quiere decir que una vez que se conoce el estado actual, el historial de transiciones que llevaron al agente a esa posición es irrelevante en términos del problema de toma de decisiones.

Un MDP puede ser finito, este es aquel en donde los estados, las acciones y recompensas (S ,A y R) tienen un número limitado de posibilidades, estas se pueden definir matemáticamente dentro de un tiempo (t), en este caso las variables R_t y S_t dependen solo del estado del agente y la acción anterior.

Hablando en términos de MDP, éste puede ser representado en una tupla de cuatro elementos, los cuales son:

1. **Estado (S)**: Se refiere a la cantidad finita de estados en los que se puede encontrar el agente.
2. **Acciones (A)**: Las acciones que puede tomar el agente autónomo.
3. **Función de transición (δ)**: La función de transición a la cual será sometido el agente durante su entrenamiento.
4. **Función de recompensa (r)**: Función importantísima dentro del aprendizaje del agente, ya que define el objetivo de éste.

2.1.2. Diferencia-Temporal

Diferencia Temporal (TD proveniente del inglés temporal difference) es una metodología de aprendizaje centrales y más novedosas en RL. Proveniente de mezclar otras metodologías como Monte Carlo (MC) y programación dinámica (PD). Una diferencia de suma importancia en comparación con las metodologías anteriores, es que TD no necesita el modelo del entorno para su funcionamiento, ya que el agente aprende con cada acción, y no cuando finaliza un episodio (Sutton and Barto, 1998). Un ejemplo de TD, es el tiempo que demora una persona en ir a

comprar a un supermercado, ya que ésta trata de predecir cuánto se demora, es una predicción sin tener el conocimiento del tráfico, de la cantidad de personas que estén en el supermercado, etc. A medida de que la persona realiza su camino , éste va conociendo nuevos estados (esto conlleva a aprender del entorno), a su vez el tiempo se actualiza haciendo una predicción más exacta.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right] \tag{2.1}$$

Para calcular el valor-estado TD se requiere una predicción. Para realizar éstas existen dos categorías principales que son on-police y off-police. On-policy estima el valor de la acción sobre un estado para una política actual, un ejemplo de esto es el algoritmo SARSA, la fórmula de actualización del par estado-acción, que se puede observar en la ecuación (2.2). Off-policy aproxima la acción-valor aprendida al par acción-valor óptimo, esto pasando completamente de la política que se siga (Sutton and Barto, 1998), un ejemplo es el algoritmo Q-learning cuya fórmula se puede observar en la ecuación (2.3).

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right] \tag{2.2}$$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right] \tag{2.3}$$

2.1.3. Explotación y Exploración

Dentro del mundo del aprendizaje por refuerzo existen otros dilemas a resolver, uno de estos es cómo balancear la exploración y la explotación, esto es de suma importancia, ya que un agente que no tenga un método para explorar nuevos caminos, repetirá la acción aprendida que le otorgue la mejor recompensa (cabe destacar que ésta puede ser negativa en algunos casos), por eso es importante y necesario que cada ciertas acciones que tome el agente, éste explore nuevas soluciones, a esto se le llama exploración, que busca nuevas acciones que le generen una mayor recompensa. Lograr un balance entre la exploración y la explotación puede traer diversos beneficios, ya sea en tiempo que se demore en aprender el agente, y la calidad de las políticas. A su vez un agente que se dedique a explorar demasiado obstaculiza que incremente la recompensa a corto plazo, ya que la mayoría de las veces intentará tomar nuevos caminos. En su contraparte, que un agente se dedique mucho a la explotación impide que las recompensas a largo plazo aumenten,

esto debido a que el agente casi siempre tratará de repetir los caminos que sabe, independiente si éstos llegan a una buena solución o no.

Volviendo a un ejemplo similar al del apartado anterior, una persona que va todos los días al trabajo, siempre camina hasta el mismo paradero para tomar el bus y así llegar a su lugar de destino luego de caminar un par de minutos, después de que repitió su recorrido durante unos días (está relacionado a la explotación), éste intenta una nueva ruta, la cual consiste en caminar hacia un paradero más lejano (relacionado a la exploración), pero el bus que pasa por ahí lo deja afuera de su trabajo. De esta misma manera el agente intenta tener un equilibrio para buscar nuevos caminos hacia su lugar de trabajo.

Alguno de los métodos más conocidos que tratan de resolver el dilema son ϵ-*greedy* y $softmax$ (Tokic, 2010).

1. ϵ-*greedy*: Este método consiste en que el agente tiene una probabilidad de exploración (ϵ) entre cero y uno, todo con el fin de que cada vez que el agente tenga que tomar una decisión, éste tenga la capacidad de poder optar por nuevos caminos en vez de la acción que maximiza su recompensa. Cabe destacar que este algoritmo puede tener algunas variantes, una de ellas puede ser que al comienzo del aprendizaje, la probabilidad de exploración sea más elevada, y a medida que se desarrolla el agente ϵ vaya disminuyendo.

2. $softmax$: Algoritmo el cual selecciona una acción basándose en probabilidades de la función valor-acción. El método softmax utiliza un parámetro T (denominado temperatura) para determinar el nivel de exploración. Por un lado, si $T \to \infty$, todas las acciones disponibles son igualmente probable. Por otro lado, si $T \to 0$, entonces el método softmax se vuelve codicioso (Szepesvári, 2010).

2.2. Aprendizaje por Refuerzo Interactivo

Aprendizaje por refuerzo interactivo o IRL (proveniente del inglés Interactive Reinforcement Learning) es un método que se utiliza con el fin de optimizar el tiempo de aprendizaje del agente, esto por medio de otros agentes (los cuales se denominan entrenadores), éstos pueden ser humanos propiamente tal, otro agente entrenado previamente o simplemente un robot el cual tiene mecanizado un movimiento.

Hablando en términos de IRL se distinguen dos formas de relación entre el entrenador y el agente. El primer método consiste en que el entrenador modifique la

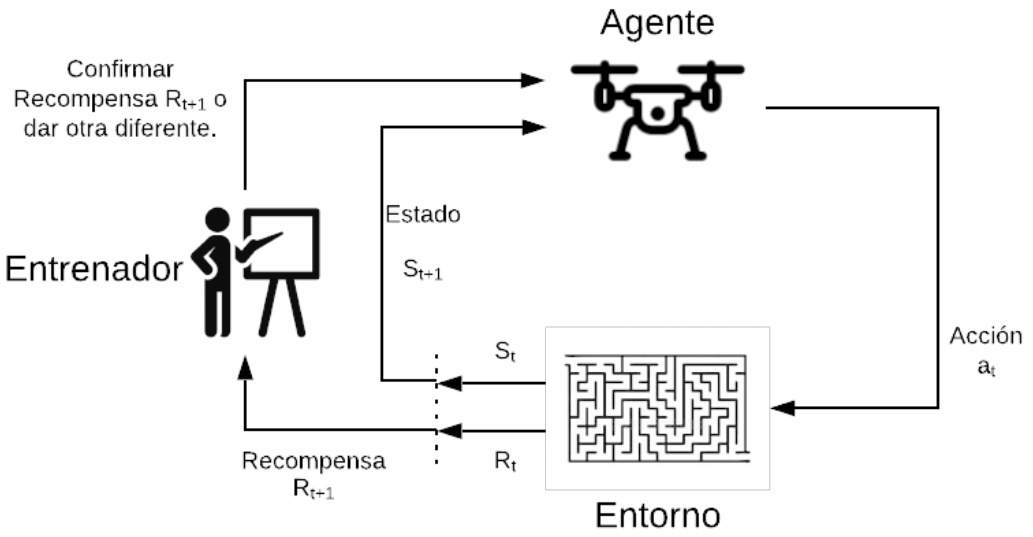


Figura 2.2: Interacción agente-entorno, en el cual un entrenador externo interviene en la política de recompensa del agente.

recompensa que se le otorgue al agente (reward-shaping), con tal de que el agente se percate de posibles errores, esto se puede apreciar en la figura 2.2 .El segundo consiste en que el entrenador le da consejos al agente en su toma de decisiones (policy-shaping), esto con tal de guiar a éste para que no cometa tantos errores, esto se puede apreciar en la figura 2.3 (Cruz et al., 2016). En IRL también existen algunas problemáticas, una de ellas es la cantidad de veces que interviene el entrenador, ya que si interactúa demasiado con el agente, éste estaría casi replicando lo que sabe el agente externo, o también el caso contrario que éste dé su apreciación muy poco, logrando que su entrenamiento no sirva de mucho (Taylor et al., 2014). Otra problemática es la calidad de las intervenciones del entrenador, ya que a veces un mal consejo del agente externo, puede fastidiar todo lo que sabe el agente (Cruz et al., 2016).

2.3. Interfaz Natural de Usuario

El término interfaz de usuario natural es una metodología emergente de interacción con la computadora que se centra en las habilidades humanas como el tacto, la visión, la voz, el movimiento y las funciones cognitivas superiores, como la expresión, la percepción y el recuerdo tal como se muestra en la figura 2.4. Una interfaz de usuario natural o NUl (proveniente de sus siglas en inglés) busca aprovechar el

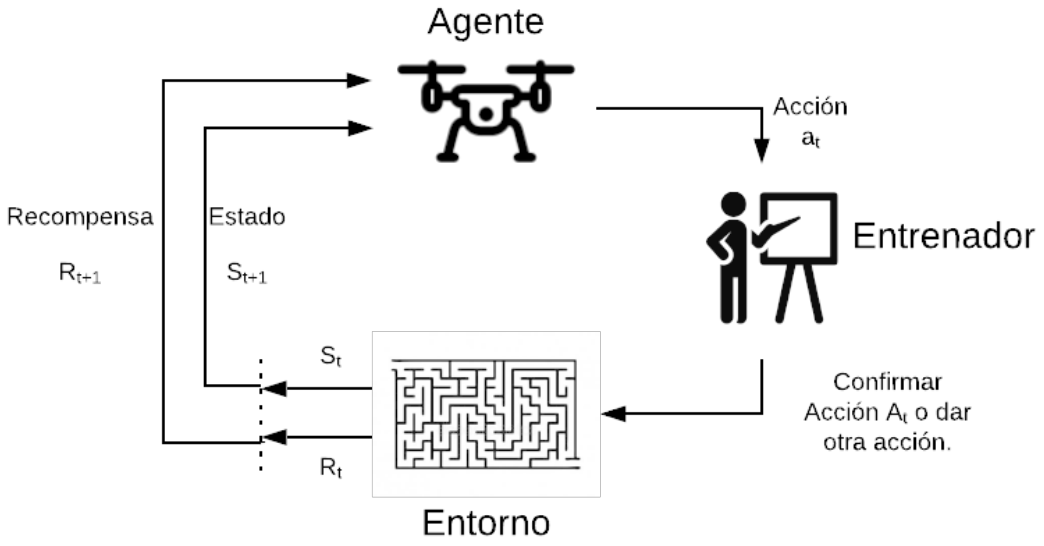


Figura 2.3: Interacción agente-entorno, en el cual un entrenador externo interviene en la de acciones del agente.

poder de una gama mucho más amplia de modalidades de comunicación que aprovechan las habilidades que las personas adquieren a través de la interacción física tradicional (Liu, 2010). De la misma manera, la interfaz gráfica de usuario (GUI) fue un gran avance para los usuarios de computadoras desde las interfaces de línea de comandos, las interfaces de usuario naturales en todas sus diversas formas se convertirán en una forma común de interactuar con las computadoras. La capacidad de las computadoras y los seres humanos para interactuar de formas diversas y sólidas, adaptadas a las habilidades y necesidades de un usuario individual, nos liberará de las limitaciones actuales de la informática, lo que permitirá una interacción compleja con los objetos digitales en nuestro mundo físico (Kaushik et al., 2014).

2.4. Control de Voz en Drones

Actualmente, algunos investigadores abordaron el control de UAV utilizando interfaces más naturales para las personas, por ejemplo, a través del reconocimiento automático de voz. Por ejemplo, (Lavrynenko et al., 2016) propusieron un sistema de control remoto por radio, en el que se utilizó un método de identificación semántica basado en coeficientes cepstrales de frecuencia mel (MFC). El audio

Figura 2.4: Ejemplos de interfaces naturales del usuario, como habla, gestos, etc.

capturado se tradujo en una acción que, a su vez, fue trasladada al UAV para su ejecución. Cada vez que un micrófono capturaba un nuevo comando de voz, el sistema calculaba el coeficiente cepstral. El coeficiente se comparó con una base de datos de coeficientes cepstrales, utilizando el criterio de distancia mínima para igualar el comando deseado. Sin embargo, la base de datos de coeficientes cepstrales solo comprendía cuatro comandos de voz, correspondientes a cada dirección del UAV. (Fayjie et al., 2017) presentaron un método de reconocimiento de voz más preciso. En ese estudio, los autores utilizaron un modelo de Markov oculto para el reconocimiento de voz con adaptación de voz para controlar el UAV. Su propuesta se basó en un motor de decodificación de voz llamado Pocketsphinx, utilizado con ROS en el simulador Gazebo. La decodificación de voz trabajó con la herramienta de base de conocimientos CMU Sphinx, implementada con siete acciones para controlar la altitud, la dirección, la guiñada y el aterrizaje. Sin embargo, la herramienta de base de conocimientos CMU Sphinx no se está desarrollando activamente y se considera obsoleta en comparación con los enfoques modernos basados en redes neuronales. Otro enfoque similar fue creado por (Landau and van Delden, 2017), donde los autores utilizaron el servicio de reconocimiento de voz Nuance. Propusieron un control UAV manos libres con comandos de voz, para actuar sobre un dron DJI Phantom 4, desarrollado con DJI Mobile SDK para iOS. La arquitectura propuesta estaba compuesta por un manos libres Bluetooth para captura de voz, y los comandos de voz se tradujeron y evaluaron utilizando

expresiones regulares. Las expresiones regulares se dividieron en tres grupos. El primer grupo contenía posibles palabras para mover el dron en cualquier dirección. El segundo consistió en posibles palabras para mover el dron en cualquier dirección, pero con una distancia establecida. El tercer grupo estaba compuesto por palabras para despegar y aterrizar el dron. La implementación de este trabajo se limitó a ser utilizada a través de un dispositivo manos libres Bluetooth conectado a un teléfono inteligente Android para controlar solo drones fabricados por DJI. (Chandarana et al., 2017) presentaron un software desarrollado a medida que utiliza el reconocimiento de voz y gestos para la planificación de rutas de UAV. En su investigación, los autores realizaron una comparación de interfaces de lenguaje natural utilizando una interfaz basada en mouse como línea de base y evaluando a los usuarios individuales que debían completar una ruta de vuelo compuesta por trayectorias o acciones de vuelo. Los autores propusieron un software en el que los usuarios interactuaban con el mouse, el gesto o el habla para construir tres rutas de vuelo específicas. La fase de reconocimiento de voz se manejó utilizando el software de conversión de voz a texto CMU Sphinx 4-5prealpha, usado con gramática basada en reglas. Esto permitió que el sistema escuchara formaciones de nombres compuestos, por ejemplo, trayectorias adelante-izquierda y atrás-derecha, entre otras. Su trabajo también presentó una evaluación de la respuesta del usuario a las interfaces de lenguaje natural. Aunque el rendimiento más alto se logró con la interfaz basada en mouse, los usuarios informaron preferencia al usar el habla para la planificación de la misión.

Además, un enfoque multimodal que considera la interacción de voz con drones utilizó un diccionario de palabras para el reconocimiento de voz (Fernandez et al., 2016). Sin embargo, solo se permitieron 15 comandos diferentes en un idioma para controlar el UAV. Quigley y col. propuso un controlador de voz para reconocer los comandos enviados a un UAV semiautónomo (Quigley et al., 2004). En sus experimentos, se llevaron a cabo pruebas de vuelo que revelaron que el ruido ambiental y la conversación podrían afectar considerablemente la confiabilidad del sistema de reconocimiento de voz. (Jones et al., 2010) llevaron a cabo un estudio de interfaces de voz y gestos para la interacción con drones en entornos simulados. Los resultados mostraron que los sujetos que participaron generalmente prefirieron usar comandos de nivel inferior, como izquierda o derecha para controlar el dron.

Uno de los estudios más recientes fue una extensión de (Lavrynenko et al., 2016) desarrollado por (Lavrynenko et al., 2019). En esta extensión, los autores presentaron un sistema de control por radio similar con análisis cepstral. Sin embargo, en

este proyecto también agregaron comunicación encriptada. La arquitectura propuesta utilizó un panel de control de voz que manejaba el cifrado, incluidos los coeficientes cepstral y wavelet. Además, se realizó el proceso inverso de cuantificación del coeficiente, comparándolo con la base de datos cepstral utilizando el criterio de distancia mínima. Ambas partes, el cifrado y el descifrado, presentaron una clave de cifrado, que trabaja con filtros de señal adquiriendo las características del habla.

Capítulo 3

Interpretación de Comandos de Voz

Este primer experimento presentado en el proyecto, consta de un algoritmo que pueda reconocer comandos de voz con una alta precisión, con el fin de poder manejar una unidad aérea no tripulada en un ambiente simulado.

3.1. Escenario Experimental

El escenario en el cual se trabajará será en V-REP (Rohmer et al., 2013), que es un software de simulación de código cerrado disponible gratuitamente con una licencia educativa para varios sistemas operativos, como Linux, Windows e iOS, para simular diferentes tipos de robots en entornos realistas. Además, cuenta con una amplia gama de bibliotecas API para comunicarse con el simulador a través de diferentes lenguajes de programación (Ayala et al., 2020). Para este experimento se construyó un escenario simulado compuesto por diferentes tipos de mobiliario que se utilizan a diario en un ambiente doméstico. Se usará el controlador de estabilización de vuelo provisto por el simulador para enfocarnos en la ejecución de los comandos a través de instrucciones de voz. El escenario experimental se ilustra en la Figura 3.1.

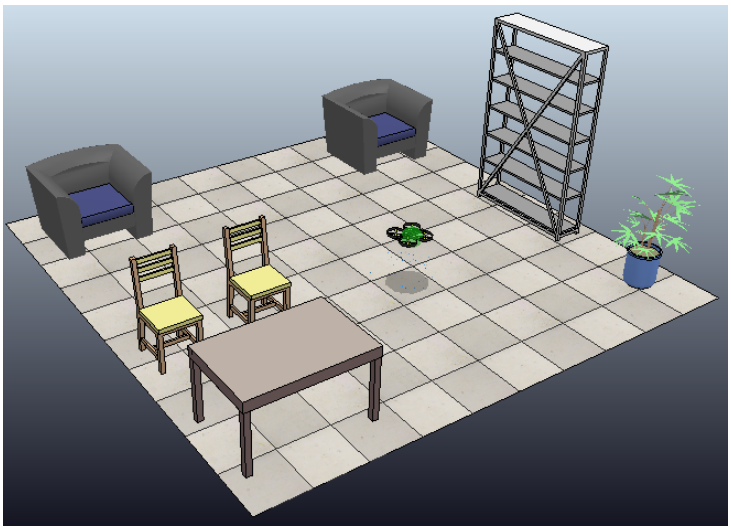


Figura 3.1: El entorno doméstico simulado en V-REP con un quadrotor y muebles de uso diario, como sofás, sillas, una mesa, un estante y una planta.

En este escenario habrá nueve acciones posibles definidas que se muestran en la tabla 3.1. Cabe destacar que las instrucciones se podrán dar en ambos idiomas: español e inglés. Una vez que se le dio una instrucción al dron, se tiene que ejecutar continuamente hasta que se instruya otra acción. Por lo tanto, para detener el vehículo, sera necesario proporcionar instrucciones explícitas para la acción "detener". La única excepción a la regla anterior era la ejecución de la acción "abajo", que podía detenerse automáticamente en caso de que el dron alcanzara 0,5 m de distancia del suelo. En tal caso, el movimiento se detuvo para evitar una colisión.

Para la implementación del escenario experimental, se utilizará el lenguaje de programación Python y se conectará al simulador a través de la API V-REP, con el fin de pasar las instrucciones entre el algoritmo de reconocimiento automático de voz y el simulador. Como se mencionó, los usuarios podrán pronunciar palabras y frases en dos idiomas, español e inglés. La selección y los beneficios de utilizar estos idiomas son dobles. Por un lado, la lengua materna de los participantes es el español. Por lo tanto, por otro lado, dado que el inglés se usa a nivel mundial, en este estudio será necesario realizar una comparación tanto del español como del inglés con mayor precisión.

Tabla 3.1: Descripción de los comandos permitidos para producir una acción para controlar el UAV.

No.	Clases de Acción	Descripción
1	Up	Aumenta la altitud del UAV
2	Down	Disminuye la altitud del UAV
3	Go right	Mover UAV a la derecha
4	Go left	Mover UAV a la izquierda
5	Go forward	Mover UAV para delante
6	Go back	Mover UAV para atrás
7	Turn right	Girar UAV 90° setido horario
8	Turn left	Girar UAV 90° antihorario
9	Stop	Parar movimiento del UAV

3.2. Solución Propuesta

En la arquitectura propuesta, la interacción con el dron no se hará a través de control remoto, sino que utilizará una Interfaz de Usuario en Lenguaje Natural (NLUI), interpretando instrucciones de humanos a través del reconocimiento automático de voz. Para la interpretación de las instrucciones, la voz de la persona fue capturada por un micrófono conectado a una computadora que ejecutó los algoritmos para procesar la señal de audio recibida. El micrófono puede ser incorporado desde una computadora portátil o cualquier otro dispositivo externo. Sin embargo, la calidad de la señal capturada puede variar considerablemente y, a su vez, afectar la precisión de la interpretación (Cruz et al., 2015). Para transformar la señal de audio en texto, se utilizó Google Cloud Speech (GCS) en combinación con un lenguaje basado en dominios.

Cada clase de acción tendrá más de una forma de ejecutar un movimiento, por ejemplo, la acción “down” se puede realizar diciendo la palabra “baja” o la oración “disminuir altura” en español, o también en la forma “bajar” o simplemente “down” en inglés. Es importante señalar que no todas las frases eran necesariamente correctas gramaticalmente, ni en español ni en inglés. Como resultado, no se asume aquí todo el tiempo que un usuario final daría una instrucción usando oraciones gramaticalmente correctas. Además, es ampliamente reconocido que en

muchas ocasiones el lenguaje hablado está menos estructurado. Por tanto, carece de formalidad ya que los usuarios no siguieron las reglas gramaticales.

En este sentido, definimos un diccionario basado en dominios que comprende 48 frases pertenecientes a las nueve clases de acción. Es importante señalar que las clases "go" y "turn" se diferenciaron ya que la primera movió el dron hacia la izquierda o hacia la derecha en las coordenadas x, y manteniendo la orientación del dron, y la segunda cambió el ángulo de guiñada del dron en 90 grados en el sentido de las agujas del reloj. o en sentido antihorario.

Las transmisiones de audio se reciben desde el micrófono y se envían al servicio GCS basado en la nube a través de la API Web Speech, de donde obtuvimos una oración reconocida como hipótesis. A continuación, se comparó la hipótesis con nuestro diccionario basado en dominios realizando una coincidencia de fonemas utilizando la distancia de Levenshtein (Levenshtein, 1966).

La distancia de Levenshtein $\mathcal{L}$, también conocida como distancia de edición, es la cantidad mínima de operaciones necesarias para transformar una oración s_x en otra oración s_y. Comparamos los caracteres dentro de s_x con los que están dentro de s_y. Las operaciones consideradas para transformar la oración comprenden sustituciones, inserciones y eliminaciones. El costo de cada operación de edición fue igual a 1. La distancia se calculó recursivamente como $\mathcal{L}_{s_x,s_y}(|s_x|,|s_y|)$ con$|s_x|$y$|s_y|$ como la longitud de las oraciones s_x y s_y respectivamente, y donde el i -ésimo segmento de la oración se calculó como se muestra en la Ecuación (3.1). En la ecuación, $c_{s_{x_i},s_{y_j}}$ es 0 si $s_{x_i} = s_{y_j}$ y 1 en caso contrario. Por lo tanto, el costo de transformar la oración s_1 = "a la izquierda" en s_2 = "ir a la izquierda" fue igual a $\mathcal{L}_{s_1,s_2} = 3$ ya que involucraba inserción de 3 nuevos caracteres. Además, el costo de transformar la oración s_3 = "ir a la derecha" a s_4 = "ir a la izquierda" fue igual a $\mathcal{L}_{s_3,s_4} = 4$ ya que el número de operaciones necesarias era 3 sustituciones y 1 deleción.

$$\mathcal{L}_{s_x,s_y}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0 \\ \min \begin{cases} \mathcal{L}_{s_x,s_y}(i-1,j)+1 \\ \mathcal{L}_{s_x,s_y}(i,j-1)+1 \\ \mathcal{L}_{s_x,s_y}(i-1,j-1)+c_{s_{x_i},s_{y_j}} \end{cases} & \text{if } \min(i,j) \neq 0 \end{cases} \tag{3.1}$$

Para realizar la coincidencia de fonemas, se calculó la distancia de Levenshtein entre la hipótesis reconocida y el diccionario basado en dominios. Posteriormente, se

seleccionó la instrucción que muestra la distancia mínima. Una vez que el comando de voz se convirtió en texto, la señal se procesó y clasificó como una instrucción para el UAV. En nuestro escenario, el dron estaba dentro del simulador de robot V-REP. La figura 3.2 muestra la arquitectura propuesta. Además, el algoritmo 3.1 retrata las operaciones realizadas para el control del dron a través de comandos de voz considerando con y sin coincidencia de fonemas.

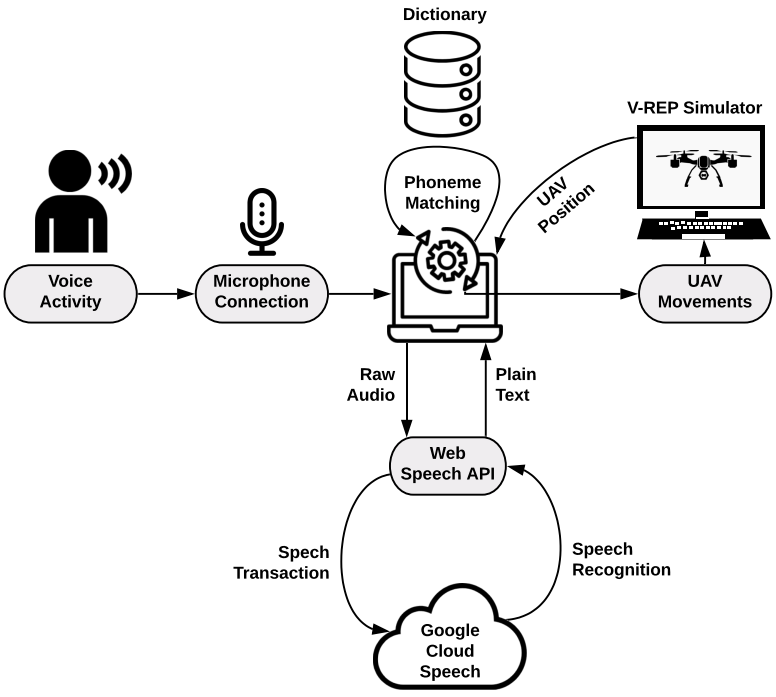


Figura 3.2: La arquitectura propuesta para el control de UAV a través del habla. En esta representación, una persona dice las instrucciones en un micrófono, y nuestro algoritmo las procesa. Luego, la instrucción se clasifica utilizando el diccionario basado en el dominio y se ejecuta para el UAV.

Algorithm 3.1. Algoritmo propuesto

para la interpretación de una señal de audio en una instrucción para el dron. El algoritmo comprende dos secciones para el reconocimiento de voz con y sin coincidencia de fonemas.

Initialize dictionary D with instructions i and classes C.
repeat
    Wait for microphone *audio* signal.
    Send *audio* signal to Google Cloud Speech.
    Receive *hypothesis* h.
    if phoneme matching is activated **then**
        for each instruction $i \in D$ **do**
            Compute $\mathcal{L}_{h,i}$.
        end for
        Choose instruction as mín $\mathcal{L}_{h,D_i}$.
        Match chosen instruction to action class $a \in C$.
        Execute action class $a \in C$ in the scenario.
    else
        for each instruction $i \in D$ **do**
            Compare h to D_i.
            if $h \in D$ **then**
                Match instruction $i \in D$ to action class $a \in C$.
                Execute action class $a \in C$ in the scenario.
                Exit loop.
            end if
        end for
    end if
until an exit instruction is given

Los experimentos se realizaron en una computadora con las siguientes características: procesador Intel Core i7-8750H, 8GB DDR4 2666MHz RAM, NVIDIA GeForce GTX 1050Ti con 4GB de GDDR5 y Windows 10 Home. La conexión a Internet utilizada fue una fibra óptica con una velocidad de carga / descarga de 300/100 Mbps.

3.3. Resultados Experimentales

En esta sección, presentamos los principales resultados que obtuvimos al probar el algoritmo propuesto. En nuestros experimentos, además de probar con instrucciones en línea pronunciadas por diferentes personas, también usamos grabaciones de diversos lugares, como espacios abiertos, oficinas y aulas. Las grabaciones presentaron una relación señal / ruido (SNR) promediada de num -3,09E-04 dB, mostrando una relación ligeramente mejor para las oraciones en español. Esto también puede atribuirse al idioma nativo de los participantes. Los valores de SNR se muestran en la Tabla 3.2 para cada clase de acción en ambos idiomas.

Tabla 3.2: SNR de entrada sin procesar (dB) para cada clase de acción tanto en español como en inglés.

Clase	**Inglés**	**Español**	**Promedio**
Up	-4,21E-04	5,33E-04	5,57E-05
Down	-9,06E-04	-2,37E-05	-4,65E-04
Go Right	-6,93E-04	-2,09E-04	-4,51E-04
Go Left	-8,03E-04	-3,16E-05	-4,18E-04
Go Forward	-3,85E-04	-1,36E-04	-2,60E-04
Go Back	-6,86E-04	-9,70E-06	-3,48E-04
Turn Left	-9,54E-04	-5,55E-05	-5,05E-04
Turn Right	-7,79E-04	2,68E-04	-2,55E-04
Stop	-2,94E-04	3,02E-05	-1,32E-04
Promedio	-6,58E-04	4,06E-05	-3,09E-04

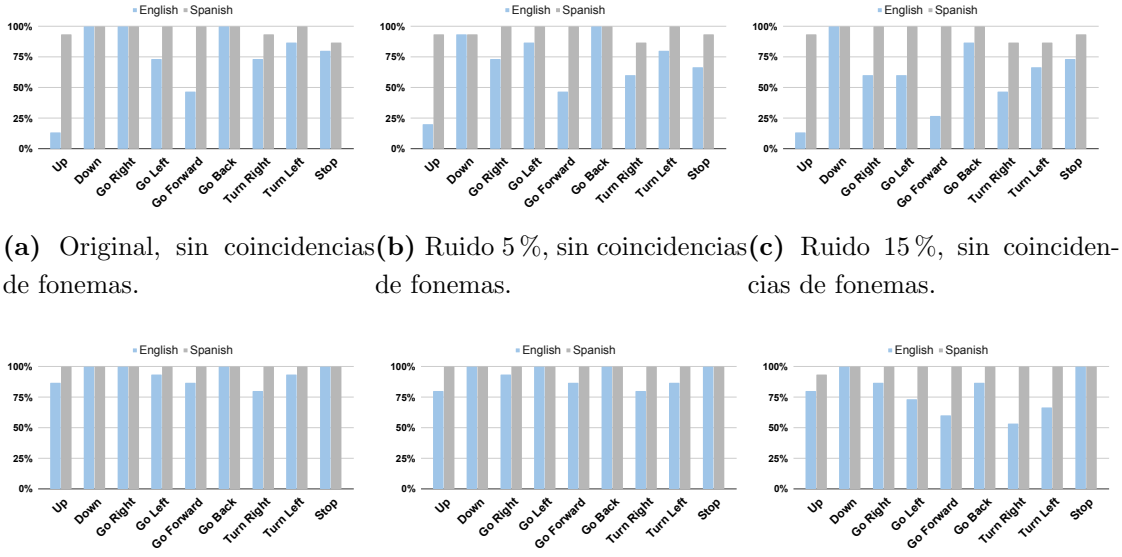
Para determinar la precisión del algoritmo propuesto, se ejecutaron pruebas en dos idiomas con y sin coincidencia de fonemas utilizando tres configuraciones diferentes, es decir, entrada sin procesar, entrada con ruido del 5 % y entrada con ruido del 15 %. Las dos configuraciones ruidosas se agregaron para probar la robustez del algoritmo en presencia de ruido e incluyeron ruido uniforme $n_1 = 0{,}05$ (uniformemente distribuido $U(-n_1, n_1)$) y $n_2 = 0{,}15$ (uniformemente distribuido $U(-n_2, n_2)$) equivalente a 5 % y 15 % con respecto a la entrada sin procesar original. Para cada configuración, cada clase de acción se realizó 15 veces para cada idioma. Por lo tanto, cada clase fue convocada un total de 30 veces, 15 para inglés

y 15 para español. En total, se probaron 270 instrucciones para cada configuración, 135 para cada idioma. Un total de 5 personas participaron en esta prueba experimental. Aunque sabíamos que el número de participantes era bastante pequeño, pudimos sacar conclusiones importantes para experimentos futuros. Además, esta investigación incluyó a personas de diferentes grupos de edad, desde los 19 hasta los 56 años (media $M = 35,4$, desviación estándar $SD = 18,45$, 3 mujeres, 2 hombres).

La figura 3.3 ilustra la precisión obtenida usando instrucciones en inglés y español para todos los niveles de ruido. Las figuras 3.3a, 3.3b y 3.3c muestran la precisión sin utilizar la coincidencia de fonemas, es decir, el algoritmo comparó el texto recibido de GCS directamente con nuestro diccionario basado en dominios. tratando de encontrar una coincidencia exacta. De lo contrario, no fue reconocido o etiquetado como "sin clase". Cuando no se utilizó la coincidencia de fonemas, se produjo una diferencia de precisión considerable entre los comandos en español e inglés, y el primero presentó los valores de reconocimiento más altos. En este sentido, los usuarios que instruían en español, es decir, su lengua materna, lograron un mejor reconocimiento de acciones en comparación con los comandos en inglés, probablemente debido a las diferencias de acento y pronunciación al hablar las palabras en un idioma extranjero. Las figuras 3.3d, 3.3e y 3.3f demuestran la precisión de reconocimiento obtenida utilizando el lenguaje basado en dominios para la coincidencia de fonemas. Al utilizar la coincidencia de fonemas, la diferencia en el reconocimiento logrado entre ambos idiomas fue atenuada por nuestro algoritmo que busca la instrucción más similar para clasificar la entrada de audio.

En términos de entradas ruidosas, como se mencionó, realizamos experimentos usando la entrada sin procesar, una entrada ruidosa del 5 % y la entrada ruidosa del 15 %. Las figuras 3.3a y 3.3d ilustran los resultados obtenidos sin y con la técnica de coincidencia de fonemas cuando se utiliza la entrada de audio sin procesar. Cuando no se aplicó ninguna coincidencia de fonemas, el algoritmo reconoció 232 de 270 instrucciones utilizando ambos idiomas, logrando una precisión del 85,93 % en el reconocimiento de voz a acción. En particular, el uso del idioma español logró un 97,04 % de precisión, mientras que el uso del inglés alcanzó un 74,81 % de precisión. Sin embargo, cuando se utilizó la coincidencia de fonemas, el algoritmo mejoró considerablemente la precisión del reconocimiento para ambos idiomas, logrando una precisión del 96,67 %. Si bien el uso del español logró una precisión del 100,00 %, el reconocimiento de los comandos en inglés mejoró significativamente en comparación con el enfoque sin coincidencia de fonemas, alcanzando una precisión

del 93,33 %.



(a) Original, sin coincidencias de fonemas. **(b)** Ruido 5 %, sin coincidencias de fonemas. **(c)** Ruido 15 %, sin coincidencias de fonemas.

(d) Original, con coincidencias de fonemas. **(e)** Ruido 5 %, con coincidencias de fonemas. **(f)** Ruido 15 %, con coincidencias de fonemas.

Figura 3.3: Precisión de reconocimiento promedio para cada clase de acción en los idiomas español e inglés con diferentes niveles de ruido en la señal de entrada. Sin utilizar la coincidencia de fonemas, el texto recibido del servicio basado en la nube se transfirió directamente al escenario. Esta implementación mostró una diferencia considerable entre los idiomas debido al idioma nativo del usuario. Usando la coincidencia de fonemas, el texto recibido del servicio basado en la nube se comparó con las instrucciones dentro del diccionario basado en el dominio. El uso de la correspondencia de fonemas demostró una mejora en el reconocimiento del habla a la acción para ambos idiomas, disminuyendo la diferencia de precisión entre ellos.

Para probar la robustez del método propuesto, aplicamos un 5 % de ruido a la entrada de audio. Los resultados obtenidos sin y con coincidencia de fonemas se muestran en las Figuras 3.3b y 3.3e respectivamente. En promedio, sin aplicar la coincidencia de fonemas, el algoritmo reconoció 214 de 270 instrucciones considerando ambos idiomas, logrando una precisión del 82,96 % en el reconocimiento de voz a acción. En particular, las instrucciones en español lograron una precisión del 96,30 %, mientras que las instrucciones en inglés tuvieron una precisión del 69,63 %. Al utilizar la correspondencia de fonemas, el algoritmo logró una precisión del 95,93 %, es decir, una precisión del 100,00 % para los comandos en español

y del 91,85 % para los comandos en inglés. Al comparar la precisión del reconocimiento con una entrada ruidosa del 5 % con la entrada en bruto, los resultados obtenidos fueron ligeramente peores, especialmente cuando se utilizó la coincidencia de fonemas. Esto demostró la solidez del enfoque propuesto en presencia de entradas de audio ruidosas.

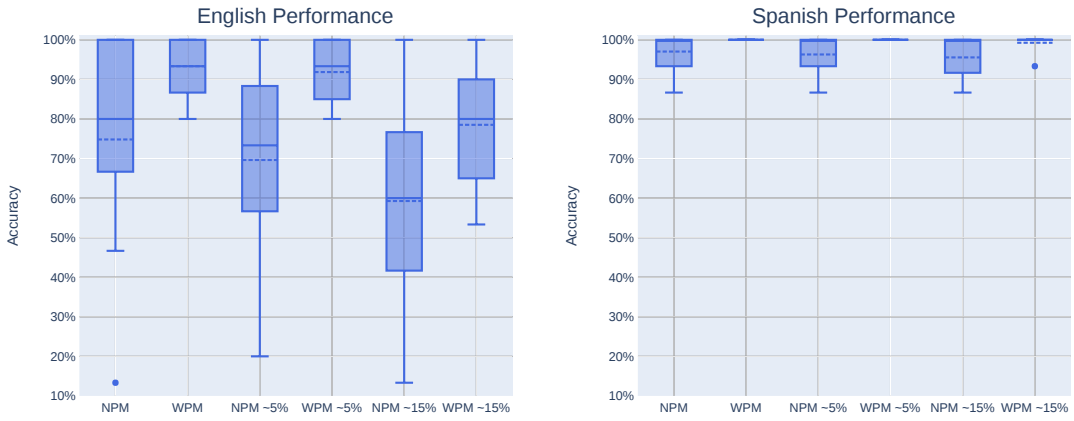
Finalmente, usamos una señal de entrada de audio con un 15 % de ruido. Los resultados se muestran en las Figuras 3.3c y 3.3f sin y con coincidencia de fonemas, respectivamente. Sin aplicar la coincidencia de fonemas, el algoritmo reconoció 198 de 270 instrucciones, logrando una precisión del 77,41 % en el reconocimiento del habla a la acción en promedio para ambos idiomas. El uso de las instrucciones en español logró una precisión del 95,56 %, mientras que la instrucción en inglés fue del 59,26 %. Cuando se introdujo la coincidencia de fonemas en esta configuración, el algoritmo logró un 88,89 % de precisión, es decir, un 99,26 % de precisión para las instrucciones en español y un 78,52 % de precisión para las instrucciones en inglés. Aunque similar al caso anterior, la introducción de ruido afectó la precisión de reconocimiento obtenida. Esto se esperaba debido a la distorsión de la señal de entrada. El uso de la correspondencia de fonemas mitigó este problema considerablemente. La mitigación de la caída de la precisión del reconocimiento fue especialmente importante considerando que el uso del inglés era un idioma extranjero para los participantes de los experimentos. Esto resultó en expresiones defectuosas o instrucciones mal pronunciadas. La tabla 3.3 resume los resultados antes mencionados para todas las configuraciones con ambos enfoques.

Tabla 3.3: Precisión de reconocimiento de audio obtenida con y sin coincidencia de fonemas en los idiomas español e inglés.

Objetivo	**Lenguaje**	**Original**	**Ruido 5 %**	**Ruido 15 %**
Sin coincidencia de fonemas	Español	97.04 %	96.30 %	95.56 %
	Inglés	74.81 %	69.63 %	59.26 %
	Ambos	85.93 %	82.96 %	77.41 %
Con coincidencia de fonemas	Español	100.00 %	100.00 %	99.26 %
	Inglés	93.33 %	91.85 %	78.52 %
	Ambos	96.67 %	95.93 %	88.89 %

Las figuras 3.4a y 3.4b ilustran el rendimiento del sistema como diagramas de caja

para instrucciones en inglés y español respectivamente. Las casillas se agrupan considerando seis conjuntos, es decir, entradas sin procesar sin coincidencia de fonemas (NPM), entradas sin procesar con coincidencia de fonemas (WPM), 5 % entradas ruidosas sin coincidencia de fonemas (NPM ˜5 %), 5 % de entradas ruidosas con coincidencia de fonemas (WPM ˜5 %), 15 % entradas ruidosas sin coincidencia de fonemas (NPM ˜15 %) y 15 % de entradas ruidosas con coincidencia de fonemas (WPM ˜15 %). El uso de instrucciones en inglés resultó en una mayor variabilidad entre los participantes durante los experimentos debido al idioma nativo de los participantes, como se señaló anteriormente. Aunque el uso de comandos en español obtuvo mejores resultados en general, la técnica de coincidencia de fonemas mejoró el reconocimiento automático de voz para el escenario propuesto utilizando instrucciones en inglés o en español.



(a) Precisión de reconocimiento usando instrucciones en inglés.

(b) Precisión de reconocimiento usando instrucciones en español.

Figura 3.4: Precisión de reconocimiento de audio para todas las configuraciones experimentales que utilizan ambos idiomas. NPM y WPM significan sin coincidencia de fonemas y con coincidencia de fonemas, respectivamente, y el porcentaje junto al enfoque en el eje x representa el valor de ruido de cada configuración. Las líneas continuas y segmentadas representan los valores de la mediana y la media en cada cuadro. El uso de la coincidencia de fonemas mejoró significativamente el reconocimiento de la voz a la acción incluso en presencia de entradas ruidosas.

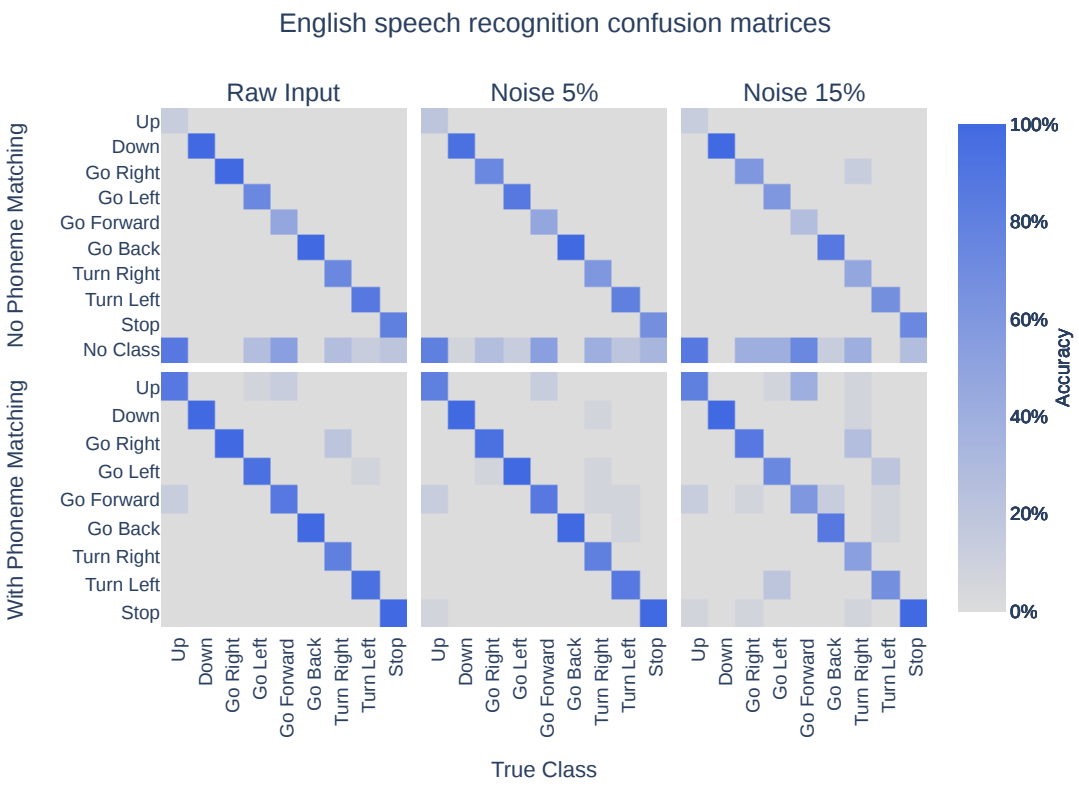


Figura 3.5: Distribución de clases predecible y real para cada configuración experimental utilizando instrucciones en inglés. La coincidencia de fonemas mejoró los resultados generales obteniendo menos clasificaciones erróneas para todas las clases de acción. Aunque una entrada ruidosa empobreció la clasificación de la clase de acción en ambos enfoques, el uso de la correspondencia de fonemas permitió una mejor precisión de reconocimiento para todos los niveles de ruido.

La figura 3.5 muestra las matrices de confusión para el reconocimiento de acciones de clase usando instrucciones en inglés en todas las configuraciones experimentales. Cuando no se utilizó ninguna coincidencia de fonemas, la etiqueta "no class" se refirió a la no coincidencia entre la hipótesis obtenida de GCS y las instrucciones dentro del lenguaje basado en el dominio. Los resultados obtenidos demostraron muchos casos en los que la hipótesis no coincidía con ninguna oración del diccionario, lo que llevó a una clasificación errónea de la instrucción. La implementación de la correspondencia de fonemas, es decir, el algoritmo que calcula la distancia entre la hipótesis recibida de GCS y cada instrucción en el diccionario basado en dominios, condujo a un mejor reconocimiento de la clase de acción. La mejora se logró para todos los comandos que el enfoque propuesto podría utilizar independientemente de la capacidad de lenguaje del usuario. Además, la Figura 3.6

ilustra las matrices de confusión para el reconocimiento de acciones de clase para instrucciones en español en todas las configuraciones experimentales. En este sentido, cuando se utilizó el idioma nativo del individuo, se produjeron menos errores de clasificación en comparación con las instrucciones en inglés. Esto siguió siendo cierto incluso cuando se emitió una señal de audio más ruidosa.

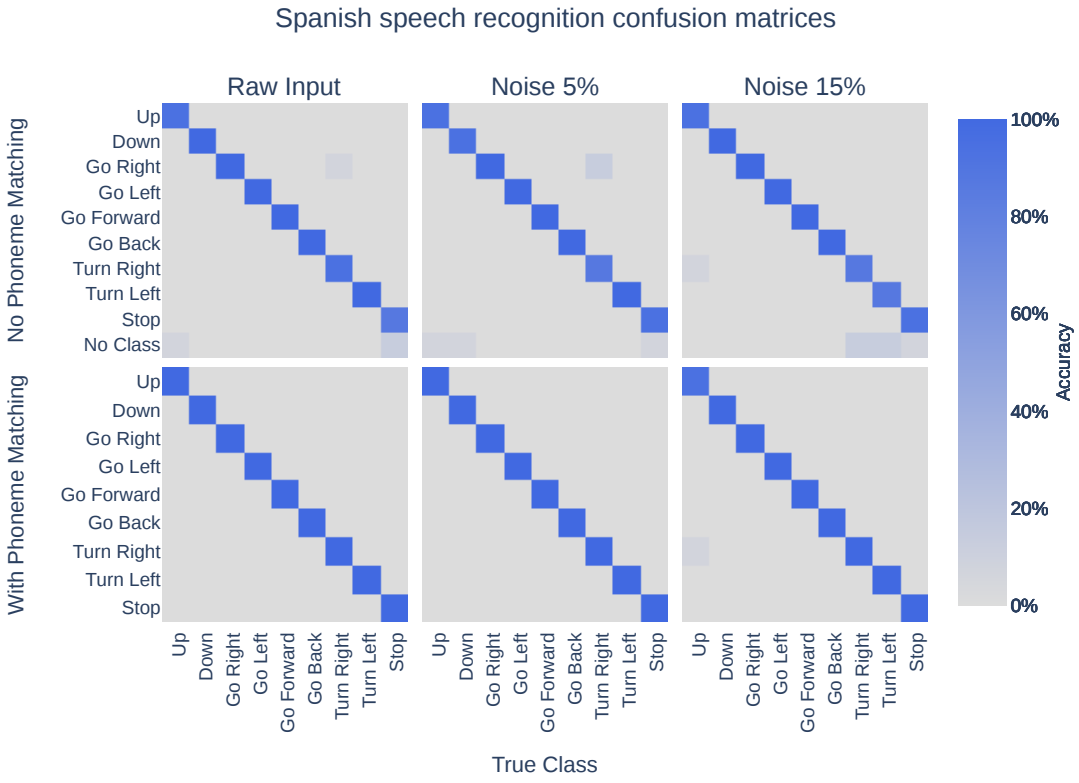


Figura 3.6: Distribución de clases predicha y verdadera para cada configuración experimental utilizando instrucciones en español. Cuando se utilizó el idioma español, se produjeron menos errores en la clasificación de acciones en comparación con las instrucciones en inglés. Sin embargo, la coincidencia de fonemas aún permitía una mejor precisión de reconocimiento para todos los niveles de ruido en comparación con el enfoque que no lo usaba.

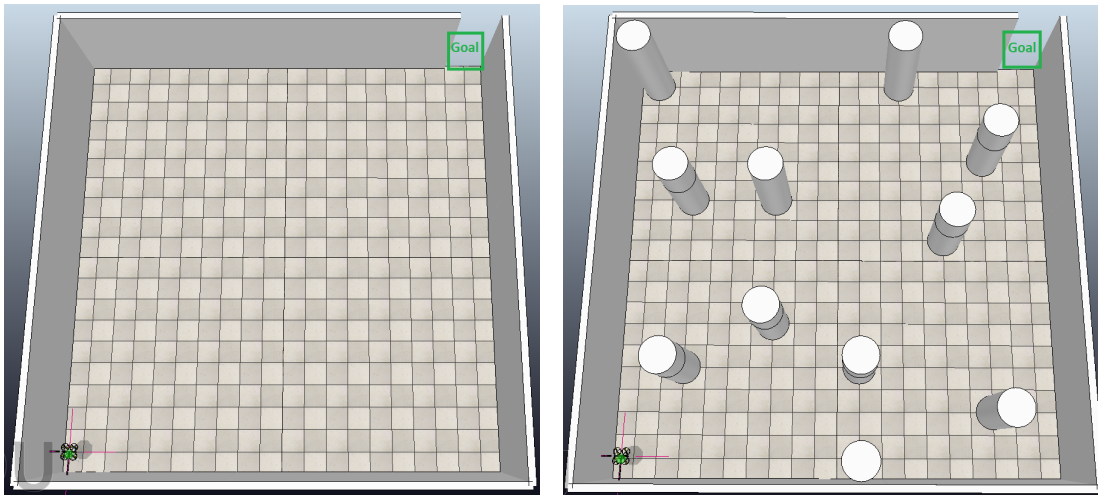
Capítulo 4

Aprendizaje por Refuerzo Interactivo del Dron

Este segundo experimento que se presenta en el documento, va directamente relacionado con el objetivo general de éste, centrándose en implementar distintas configuraciones de agentes de aprendizaje por refuerzo, entre ellas una que mezcla "policy-shaping" y "reward-shaping", todo esto implementado en un dron en ambientes simulados.

4.1. Escenario Experimental

Al igual que en el capítulo anterior, se utilizará el simulador de robótica V-REP, montando dos escenarios experimentales. Ambos poseen una superficie de 10 metros de ancho y 10 metros de largo, a su vez, esta superficie tiene una muralla perimetral, con una pequeña apertura de un metro de ancho en el lado nor-este de ésta. Todo esto se puede apreciar en la figura 4.1.



(a) Primer escenario experimental, sin obstáculos que impidan el movimiento del dron.

(b) Segundo escenario experimental, con obstáculos que impidan el movimiento del dron.

Figura 4.1: Entornos simulados en V-REP con una unidad aérea cada uno.

Estos escenarios tendrán un dron al comenzar la simulación, el cual podrá moverse por toda la superficie, con el fin de salir por la apertura del lado superior derecho, cabe destacar que la unidad aérea acepta 9 tipos de comandos o acciones, las cuales son idénticas a las mostradas en el capítulo anterior (éstas se pueden apreciar en la tabla 3.1), las que implican un tipo de traslación en el dron, lo hacen con un alcance de 1 metro, o sea, si a este se le solicita la acción "Up" o "Subir" (que se muestra en la tabla 3.1), significa que el robot aumentará su altura en un metro.

La unidad aérea posee 4 sensores de proximidad con un alcance de 1 mt de distancia, los cuales se encuentran en los 4 puntos cardinales de ésta, por lo tanto si existe algún tipo de objeto que se encuentre frente alguno de estos sensores, se puede apreciar un cambio de estados en éstos.

En el caso del entorno mostrado en la figura 4.1b, posee pilares de unos 0,80 mts de diámetro, los cuales obstruyen el paso del UAV en algunas trayectorias.

4.1.1. Límites

Los escenarios presentados anteriormente, poseen algunas limitantes, como por ejemplo:

- La altitud máxima de la unidad aérea no puede ser mayor a 2,50 mts, por lo tanto si el dron se encuentra a esa altitud, y se le solicita subir, éste no

efectuará la acción.

- La altitud mínima de la unidad aérea no puede ser menor a 0,50 mts, por lo tanto si el dron se encuentra a esa altura, y se le solicita bajar, éste no efectuará la acción.

- Si al dron se le solicita una acción o comando que implique moverse en una dirección, y a su vez justo existe un objeto o muralla que obstruya el paso del UAV en ésta, el sensor que apunta en esa dirección se activará, lo cual implica que no se efectuará el movimiento del dron.

4.2. Solución Propuesta

En esta sección se plantean distintas técnicas de aprendizaje que se aplicarán en el dron, con el fin de que éste llegue al objetivo final, el cual es salir por la apertura presente en la parte superior derecha de los entornos mostrados en la figura 4.1. Para lograr el aprendizaje de la unidad aérea se mezclaran distintas técnicas de entrenamiento, las cuales fueron mencionadas en capítulos anteriores.

Específicamente esta sección se hablará de dos experimentos principales, los cuales están relacionados cada uno a cada escenario experimental, por lo tanto para que sea más fácil la referencia, se aplicará una abreviación, diciendo "Experimento sin obstáculos" se referirá al experimento de aprendizaje de la unidad aérea en el primer entorno simulado, o sea el que se muestra en la figura 4.1a, y cuando se hable de "Experimento con obstáculos" se referirá al experimento de aprendizaje del UAV en el segundo entorno simulado, o sea el que se muestra en la figura 4.1b.

Para los siguientes experimentos, surgió una extensión del experimento que fue mencionado en el capítulo 3, ya que a esté, en vez de que sea para solo reconocer acciones, se le agregaron 4 clases nuevas (las cuales se pueden ver en la tabla 4.1), éstas corresponden a consejos de recompensa, éstas pueden ser representadas en un diccionario de unas 15 instrucciones, ya sea en español como en inglés. Por cada clase de recompensa, se grabaron 15 audios en español y 15 en inglés, provenientes de un universo de 5 personas. Todo con el fin de utilizar el algoritmo del capítulo anterior para el aprendizaje de agentes interactivos, tanto como de "policy-shaping" y "reward-shaping".

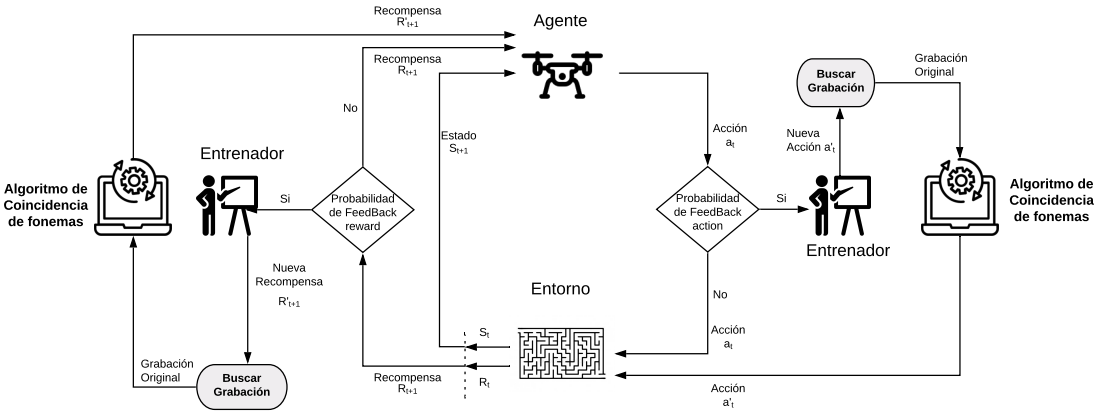


Figura 4.2: Arquitectura propuesta para agente de IRL.

Tabla 4.1: Comandos de recompensas admitidos por el algoritmo reconocedor de instrucciones por medio de la voz.

No.	Clase	**Descripción**
1	Very bad	Recompensa negativa cuando va a colisionar.
2	Bad	Recompensa negativa cuando se aleja del obj.
3	Well	Recompensa positiva cuando se acerca al obj.
4	Perfect	Recompensa positiva cuando se llega al obj.

Respecto al experimento sin obstáculos, o sea, el relacionado al escenario experimental No 1, ya que el dron solo puede moverse en distancias de 1 mt (esto sin tomar en cuenta el desplazamiento en el eje Z), puede ser representado por una grilla de 10 x 10, eso es respecto a la posición del UAV, pero también tiene una variable que influye en éste, la cual es la orientación del dron, ya que puede ser "Norte","Sur","Este" y "Oeste", por lo tanto si quisiéramos representar la cantidad de estados posibles en este escenario sería de unos 10 x 10 x 4, lo cual serían unos 400 estados posibles.

Ya que el fin de este proyecto es ver la efectividad de combinar "Reward-shaping" con "Policy-shaping", ser ocuparan diversas combinaciones de técnicas de aprendizaje por refuerzo, específicamente de aprendizaje por refuerzo interactivo. Por lo tanto se procederá a realizar 4 sub-experimentos, los cuales tienen en común ciertos aspectos, como:

- Para todos se utilizará la política de epsilon-greedy, con un valor de epsilon de 0,1 , o más bien lo que significa que el 10 % de las veces que el agente tenga que tomar una decisión, ésta será de forma aleatoria para explorar nuevos caminos.

- Se ocupará técnicas de Temporal-Differences, específicamente de Q-Learning, con el parámetro de gamma de 0,95, esto significa que cada vez que el agente tenga que actualizar el valor de su par estado-acción, a éste le importará demasiado el valor del estado siguiente. Debido a que se ocupará Q-Learning, se ocupará como referencia el par estado-acción, por lo tanto significa que por cada estado, existen 9 acciones posibles, o sea el entorno se complejiza a unas 3.600 posibilidades de par estado-acción, lo cual puede hacer un poco demoroso el aprendizaje del agente.

- Para guardar el valor de cada par estado-acción, se hará uso de una matriz, la cual tiene unas 3.600 combinaciones como se habló anteriormente.

- El valor de la variable alpha, es de un 0,1, ya que al revalorizar el valor del estado-acción actual, no se quiere caer en mínimos locales.

- Cada vez que se necesite un consejo de acción por parte del agente entrenador, o algún tipo de recompensa por el mismo, primero será procesada por éste, luego, dependiendo del valor obtenido, se buscará alguna grabación que corresponda a éste consejo, y se le entregará al algoritmo de el capítulo 3, con el fin de que este consejo, ya sea tanto de acción o como de recompensa al nuevo agente, sea simulado como si viniera de una persona. Esto implica, que en algunas ocasiones, como el experimento de el capítulo 3 tiene una efectividad de un 96,67 % (asumiendo que el audio de la grabación es el original, ya sea tanto en español o inglés) al reconocer las instrucciones, puede que se dé el caso de que la acción o recompensa sugerida no sea la que quiera dar el agente entrenador.

Hablando más específicamente de los 4 sub-experimentos del Experimento sin obstáculos, serán los siguientes:

- 20 agentes de aprendizaje por refuerzo, los cuales aprenderán autónomamente, los cuales seguirán los parámetros mencionados anteriormente.

- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de policy-shaping por parte del último agente entrenado autónomamente.

- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de reward-shaping por parte del último agente entrenado autónomamente.

- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de policy-shaping y reward-shaping por parte del último agente entrenado autónomamente.

Cabe destacar que cuando se habla de probabilidad de reward y policy shaping, es de un valor de un 0,15 o más bien de un 15 %. En la figura 4.2 se puede apreciar la arquitectura propuesta para el aprendizaje de los agentes, ya sean de aprendizaje autónomo o por aprendizaje por refuerzo interactivo. Si hablamos respecto de la cantidad de episodios por agente, estos serían unos 20, finalizando cada uno de estos cuando el dron llega a la apertura que se encuentra en la parte superior-derecha del escenario. Cabe destacar que para los agentes entrenadores y los implementados con policy-shaping ocupan la misma función de reward, pero los implementados con con alguna variante que contenga reward-shaping ocupan una distinta (las cuales se pueden ver en la tabla 4.1), la cual es menos castigadora, y a su vez en la recompensa final es menor, por lo tanto, cuando se obtiene un consejo de reward por parte del entrenador (o sea el 15 % de las veces), ésta recompensa va a ser más castigadora que una entregada por la función de reward del agente actual, por ejemplo, cuando el agente actual va a chocar, se le otorga un -10 de recompensa, pero si fuera por un consejo del agente entrenador, este castigo sería de unos -20. También sucede cuando el agente aprendiz termina un episodio este recibe una recompensa de 800, no obstante si la recompensa viene por parte del agente entrenador será de unos 1.000. En general cuando el agente aprendiz toma una buena acción, la recompensa entregada por el agente entrenador será mayor a la por defecto que tiene el aprendiz, y cuando éste toma una mala decisión, el castigo dado por el entrenador será mayor.

Tabla 4.2: Valores de la función recompensa, para los distintos tipos de agentes de aprendizaje por refuerzo.

Clase	**Entrenadores y P-S**	**R-S, P-S y R-S**
Very bad	-20	-10
Bad	-1	-0.5
Well	1.5	1
Perfect	1000	800

Cuando hablamos del experimento con obstáculos, o sea, el escenario experimental relacionado cambia, independiente de que el dron solo puede moverse en distancias de 1 mt (esto sin tomar en cuenta el desplazamiento en el eje Z), puede ser representado por una grilla de 10 x 10, eso es respecto a la posición del UAV, pero a ese resultado hay que restar la cantidad de obstáculos presentes el éste, que vienen a ser unos 11 pilares, o sea, que a diferencia del escenario experimental A, éste tiene 11 posiciones en la cual el dron no puede estar, eso vendría a ser unos 89 estados respecto a posición, en este experimento también influye la orientación del dron, ya que puede ser "Norte","Sur","Este" y "Oeste", por lo tanto si quisiéramos representar la cantidad de estados posibles en este escenario sería de unos (10 x 10 - 11) x 4, lo cual serían unos 356 estados posibles.

Al igual que en experimento sin obstáculos, el fin de este proyecto es ver la efectividad de combinar "Reward-shaping" con "Policy-shaping", por lo tanto se procederá a realizar 4 sub-experimentos, los cuales tienen en común ciertos aspectos, como:

- Para todos se utilizará la política de epsilon-greedy, con un valor de epsilon de 0,1 , o más bien lo que significa que el 10 % de las veces que el agente tenga que tomar una decisión, ésta será de forma aleatoria para explorar nuevas posibilidades.
- Se ocupará técnicas de Temporal-Differences, específicamente de Q-Learning, con el parámetro de gamma de 0,95, esto significa que cada vez que el agente tenga que actualizar el valor de su par estado-acción, a éste le importará demasiado el valor del estado siguiente. Debido a que se ocupará Q-Learning, se ocupará como referencia el par estado-acción, por lo tanto significa que por cada estado, existen 9 acciones posibles, o sea el entorno se complejiza a unas 3.204 posibilidades de par estado-acción, número el cual es menor al del ex-

perimento sin obstáculos, por lo tanto al proceder a tomar los experimentos, el tiempo de ejecución de estos debe ser menor.

- Para guardar el valor de cada par estado-acción, se hará uso de una matriz, la cual tiene unas 3.204 combinaciones como se habló anteriormente.
- El valor de la variable alpha, es de un 0,1, ya que al revalorizar el valor del estado-acción actual, no se quiere caer en mínimos locales.
- Cada vez que se necesite un consejo de acción por parte del agente entrenador, o algún tipo de recompensa por el mismo, primero será procesada por éste, luego, dependiendo del valor obtenido, se buscará alguna grabación que corresponda a éste consejo, y se le entregará al algoritmo de el capítulo 3, con el fin de que este consejo, ya sea tanto de acción o como de recompensa al nuevo agente, sea simulado como si viniera de la voz de una persona. Esto implica, que en algunas ocasiones, como el experimento de el capítulo 3 tiene una efectividad de un 96,67 % (asumiendo que el audio de la grabación es el original, ya sea tanto en español o inglés) al reconocer las instrucciones, puede que se dé el caso de que la acción o recompensa sugerida no sea la que quiera dar el agente entrenador.

Hablando más específicamente de los 4 sub-experimentos del Experimento con obstáculos, serán los siguientes:

- 20 agentes de aprendizaje por refuerzo, los cuales aprenderán autónomamente, los cuales seguirán los parámetros mencionados anteriormente.
- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de policy-shaping por parte del último agente entrenado autónomamente.
- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de reward-shaping por parte del último agente entrenado autónomamente.
- 20 agentes de aprendizaje por refuerzo interactivo, los cuales tendrán una probabilidad de consejo de policy-shaping y reward-shaping por parte del último agente entrenado autónomamente.

Cabe destacar que cuando se habla de probabilidad de reward y policy shaping, es de un valor de un 0,15 o más bien de un 15 %. Al igual que en el experimento sin obstáculos, en la figura 4.2 se puede apreciar la arquitectura propuesta para

el aprendizaje de los agentes, ya sean de aprendizaje autónomo o por aprendizaje por refuerzo interactivo. Si hablamos respecto de la cantidad de episodios por agente, estos serían unos 20, finalizando cada uno de estos cuando el dron llega a la apertura que se encuentra en la parte superior-derecha del escenario.

Al igual que en el experimento sin obstáculos, las funciones de recompensa son distintas para agentes entrenadores y los implementados con policy-shaping, comparados con los que tengan implementada alguna variante que contenga reward-shaping (las cuales se pueden ver en la tabla 4.1), ésta cual es menos castigadora, y a su vez en la recompensa final es menor, por lo tanto, cuando se obtiene un consejo de reward por parte del entrenador (o sea el 15 % de las veces), ésta recompensa va a ser más castigadora que una entregada por la función de reward del agente actual.

Los experimentos A y B se realizaron en una computadora con las siguientes características: procesador Intel Core i7-8750H, 8GB DDR4 2666MHz RAM, NVIDIA GeForce GTX 1050Ti con 4GB de GDDR5 y Windows 10 Home. La conexión a Internet utilizada fue una fibra óptica con una velocidad de carga / descarga de 300/100 Mbps.

4.3. Resultados Experimentales

Después de terminar la serie de experimentos mencionados en el capítulo anterior, se pueden apreciar datos de gran relevancia de los entrenamientos de los diversos agentes de aprendizaje por refuerzo.

Para empezar, en la figura 4.3 se pueden apreciar las recompensas promedios de los agentes de aprendizaje por refuerzo, ya sea los relacionados al experimento sin obstáculos o el B.

Siendo más específicos, en la figura 4.3a, se muestra un gráfico, el cual tiene las recompensas promedios de los agentes de aprendizaje (aprendizaje autónomo, en el gráfico se refiere a ellos como Entrenador) por refuerzo por los 20 episodios (relacionados al Experimento sin obstáculos) que duraba el entrenamiento de cada uno de estos. A su vez esto se repite para las 3 configuraciones siguientes, o sea al aplicado con tecnicas de “Policy-shaping” (en el gráfico se refiere a ellos con la abreviación P-S), “Reward-shaping” (en el gráfico se refiere a ellos con la abreviación R-S) y “Policy-shaping con Reward-shaping” (en el gráfico se refiere a ellos con la abreviación P-S y R-S).

En la gráfica se logra ver que los agentes con una curva de aprendizaje más lenta fueron los agentes entrenadores. Los siguientes agentes con mejor curva de aprendizaje que los entrenadores son los que fueron aplicados con la técnica de Reward-shaping, siendo levemente superior a los entrenadores. Luego los siguientes agentes que siguen con mejor recompensa promedio son los que fueron aplicados con ambas técnicas, o sea Policy-shaping y Reward-shaping a la vez. Por último los agentes con mejor recompensa promedio son los que fueron aplicados con solamente Policy-shaping. También en la tabla 4.3, a parte de ver la recompensa promedio por cada experimento, se puede apreciar la desviación estándar de la curva de aprendizaje, siendo la configuración de agentes entrenadores con la mayor desviación estándar, y la con menor la combinación de técnicas de P-S y R-S.

Para apreciar la recompensa promedio por agente del Experimento con obstáculos, nos dirigimos a la figura 4.3b, en ésta se muestra un gráfico, el cual tiene las recompensas promedios de los agentes entrenadores, por los 20 episodios que duraba el entrenamiento de cada uno de estos. A su vez esto se repite para las 3 configuraciones que son las mismas apreciadas en el Experimento sin obstáculos.

En la gráfica se logra ver que los agentes con una recompensa promedio menor fueron los agentes implementados con reward-shaping, pero partiendo con una recompensa más alta que los agentes entrenadores en los primeros episodios. Los que siguen en la lista son los que fueron aplicados con ambas técnicas, o sea Policy-shaping y Reward-shaping a la vez, no obstante no presenta una curva de aprendizaje con mucha pendiente. Los siguientes en la lista son los agentes entrenadores, logrando tener una recompensa promedio más alta que los otros dos casos explicados anteriormente, pero, en los primeros episodios de aprendizaje, fueron los que tuvieron menor recompensa. Para finalizar, los agentes que tuvieron una mejor recompensa promedio fueron los implementados con policy-shaping. No obstante, si bien los agentes entrenadores no fueron los con menor recompensa promedio, en la tabla 4.3 se puede apreciar que la desviación estándar de la curva de aprendizaje de éstos es la mayor, y nuevamente la con menor desviación estándar fue la combinación de técnicas de P-S y R-S.

En términos de recompensa promedio, los agentes implementados con alguna variación de reward-shaping son los que tienen menores resultados, esto es debido a que su función de recompensa es distinta a los agentes entrenadores y los implementados con policy-shaping. Uno de los cambios más significativos es cuando el agente llega al objetivo final, otorgando una recompensa de 800 unidades, en comparación con la otorgada a los otros agentes que tiene un valor de 1.000 unidades,

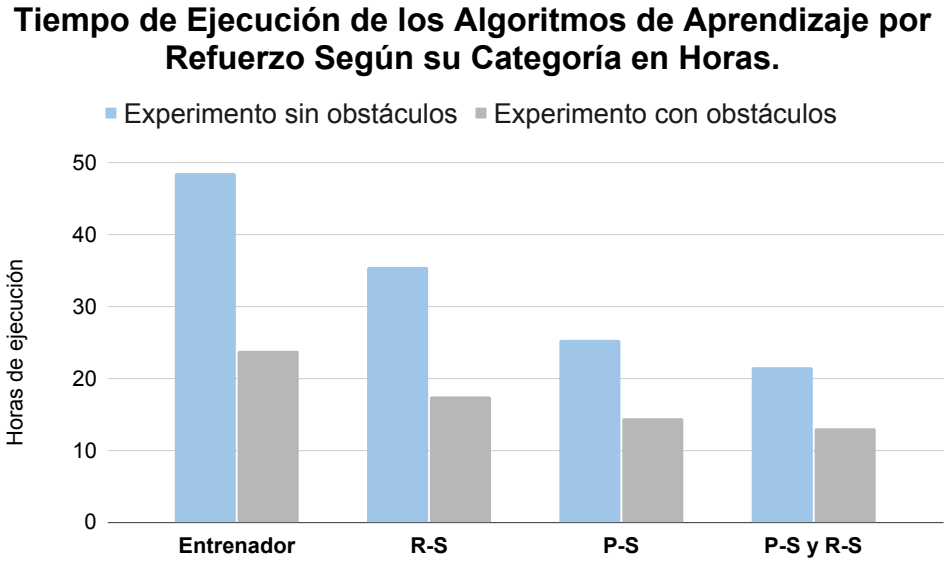
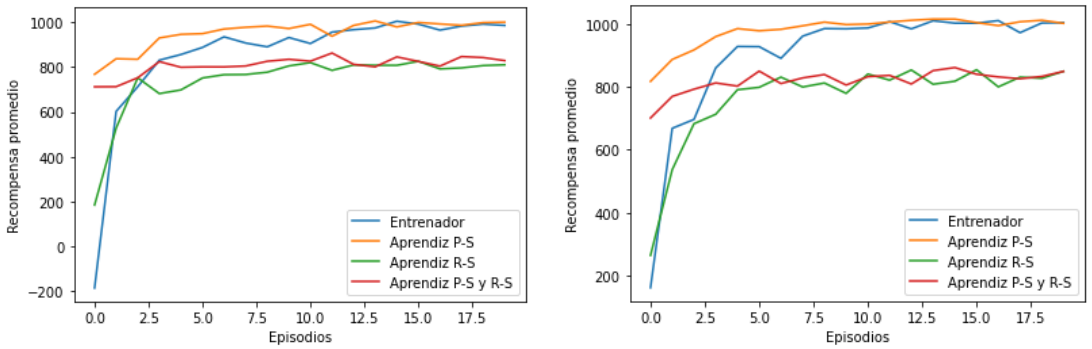


Figura 4.4: Tiempos de ejecución por cada algoritmo de aprendizaje por refuerzo, tomando como unidad de medida 1 hora.

siendo una clara diferencia de 200 puntos la cual influye de forma negativa en las gráficas de las recompensas promedio para los agentes de reward-shaping.



(a) Gráfico de recompensas promedio de los 20 agentes de aprendizaje por refuerzo según la técnica utilizada para aprender en el primer entorno sin obstáculos.

(b) Gráfico de recompensas promedio de los 20 agentes de aprendizaje por refuerzo según la técnica utilizada para aprender en el segundo entorno con obstáculos.

Figura 4.3: Gráficos de recompensas promedio de los 20 agentes de aprendizaje por refuerzo según la técnica utilizada para aprender en su respectivo entorno simulado.

Tabla 4.3: Recompensa promedio de los agentes entrenados, tanto como del Experimento sin obstáculos y B con sus respectivas configuraciones.

Experimento	**Configuración**	**Promedio**	**Desviación Estándar**
Experimento sin obstáculos	Autónomo	854.7	264.6
	P-S	952.2	64.8
	R-S	738.6	147.1
	P-S y R-S	808.2	40.5
Experimento con obstáculos	Autónomo	902.4	200.0
	P-S	979.6	50.4
	R-S	765.5	139.2
	P-S y R-S	819.1	35.8

También, en la figura 4.4 se pueden apreciar los tiempos de ejecución de cada experimento sin obstáculos y B, con sus 4 configuraciones respectivamente, que fueron abreviadas de igual manera que en la gráfica 4.3. De esta gráfica se obtiene los siguientes resultados.

- El tiempo de ejecución más demoroso en la etapa de entrenamiento fue de los agentes entrenadores, ya sea para el experimento sin obstáculos y B (con un tiempo de ejecución de 48,5 y 23,8 horas respectivamente, valores aproximados).
- La siguiente configuración de entrenamiento que se demoró menos fue la de Reward-shaping, con un tiempo de ejecución de unas 35,5 y 17,5 horas(valores aproximados) para cada experimento sin obstáculos y B respectivamente. Siendo un 26,89 % y un 26,47 % menos que los agentes entrenadores para el experimento sin obstáculos y B.
- La siguiente configuración de entrenamiento que se demoró menos fue la de la técnica de Policy-shaping, con un tiempo de ejecución de unas 25,4 y 14,4 horas(valores aproximados) para cada experimento sin obstáculos y B respectivamente. Siendo un 47,69 % y un 39,50 % menos que los agentes entrenadores para el experimento sin obstáculos y B.
- La configuración de entrenamiento que se demoró menos fue la de la técnica de Policy-shaping y Reward-shaping combinadas, con un tiempo de ejecución

de unas 21,6 y 13,1 horas(valores aproximados) para cada experimento sin obstáculos y B respectivamente. Siendo un 55,52 % y un 44,96 % menos que los agentes entrenadores para el experimento sin obstáculos y B.

Capítulo 5

Conclusiones

A lo largo de todo este documento se pudo apreciar dos grandes experimentos, los cuales hacen referencia al capítulo 3 y el 4, el primero trata sobre el manejo de una unidad aérea por medio de la voz, y técnicas adecuadas para reconocer de mejor manera las instrucciones que se le da a éste. En el segundo se habla de la reutilización del algoritmo con mejor resultado que se generó en el primer experimento, ocupándolo para dar consejos en agentes de aprendizaje por refuerzo, simulando que el consejo lo diera una persona. A partir de los dos experimentos mencionados anteriormente, se pueden deducir distintas conclusiones.

En el trabajo del capítulo 3, se presentó una arquitectura para controlar un dron simulado a través de comandos de voz interpretados a través de un sistema de reconocimiento de voz automático basado en la nube y un diccionario de idiomas basado en dominios. El uso de la correspondencia de fonemas mejoró el nivel de precisión en el reconocimiento de instrucciones. Las entradas sin procesar sin coincidencia de fonemas dieron como resultado un 97,04 % y un 74,81 % de precisión en el reconocimiento de acciones en español e inglés, respectivamente. En promedio, el reconocimiento de comandos de voz sin coincidencia de fonemas logró una precisión del 85,93 %. Después de probar el método de reconocimiento de voz complementado con un lenguaje basado en dominio para operar el UAV en un entorno doméstico, se obtuvieron mejores resultados.

En general, el rendimiento en el reconocimiento de instrucciones mejoró con la correspondencia de fonemas, obteniendo una precisión del 93,33 % y 100,00 % en inglés y español, respectivamente. De media, obtuvimos un 96,67 % de precisión cuando las instrucciones se interpretaron utilizando la correspondencia de fonemas. Además, probamos nuestro enfoque con un 5 % y un 15 % de ruido en la entrada.

En general, al utilizar la correspondencia de fonemas, nuestro método logró buenos resultados mostrando la robustez del algoritmo propuesto contra el ruido.

Al terminar los experimentos del capítulo 4, que son relacionados a agentes de aprendizaje por refuerzo, con distintas configuraciones, se lograron resultados bastante interesantes. Tanto en el experimento sin obstáculos como en el B, las curvas de aprendizaje de los distintos tipos de agentes son bastante parecidas en términos de progresión, logrando la mejor curva de recompensa la configuración de los agentes con Policy-shaping, seguido por la técnica de Policy y Reward shaping combinadas (recordemos que el objetivo de este documento era ver el comportamiento de ésta configuración y compararlo con las demás) en el experimento sin obstáculos, y en el B los agentes entrenadores, a continuación los agentes con mejor curva de recompensa son los que están configurados con Reward-shaping para el experimento sin obstáculos y con la combinación de las dos técnicas para el experimento con obstáculos. Por último los agentes entrenadores en el experimento sin obstáculos, los cuales no presentaban ningún tipo de consejo durante su entrenamiento, y en el B los agentes implementados con reward-shaping.

Si bien todo indicaría que la mejor técnica de aprendizaje por refuerzo interactivo es policy-shaping, los tiempos de ejecución de los algoritmos de entrenamiento no indican eso, situando a la configuración de policy y reward shaping combinadas con el menor tiempo, específicamente unas 17,35 horas promediando los tiempos del experimento sin obstáculos y B. Ésta configuración es seguida por la técnica de policy-shaping, con un tiempo promedio de 19,9 horas. Luego el algoritmo implementado con reward-shaping lo sigue con unas 26,5 horas en promedio, para terminar con los agentes más lentos que fueron los entrenadores.

La diferencia que se menciona en el párrafo anterior, se puede deber a que policy-shaping solo aconseja para tomar acciones, por lo tanto obviamente ayuda en el aprendizaje del dron o el agente, pero al combinar p-s y r-s éste demoró menos en su aprendizaje, esto se puede deber a que también daba consejos de reward al agente, por lo tanto cuando este tomaba una acción equivocada, el consejo de castigo era mayor al que tenía el agente por defecto, afectando negativamente en su recompensa, pero no así en su tiempo de aprendizaje. En el experimento con obstáculos, los agentes que tenían implementado reward-shaping (ya sea en combinación con policy-shaping o no) tuvieron menor recompensa promedio, esto se puede deber a que el escenario experimental tenía muchos obstáculos en el camino del dron, cosa que no pasaba con el primer escenario, por lo tanto, si se solicitaba un consejo de recompensa (como se dijo en el capítulo anterior), éste

iba a ser más "castigador" que el que viene por defecto. También afecta que la recompensa por terminar el episodio era de unas 800 unidades, en comparación de las 1.000 que tenían los agentes entrenadores y los implementados con p-s. Es más si nos enfocamos en la curva de aprendizaje de los dos experimentos, la que tuvo una mayor desviación durante los episodios fue la de los agentes entrenadores, por lo tanto, se puede intuir que su aprendizaje fue más "tardío" o con más "malos movimientos", cosa que se corrobora con la figura 4.4.

Por otra parte, si nos fijamos en la tabla 4.3, nos damos cuenta que tanto para el experimento sin obstáculos y B hay ciertas similitudes, más específicamente en los valores de desviación estándar de las curvas de aprendizaje de los agentes, siendo los agentes entrenadores los con mayor valor, seguidos por los agentes implementados con solo reward-shaping, luego por los implementados con policy-shaping, y por ultimo con los agentes que combinaron las dos técnicas. Esto puede significar que el entrenamiento más "expedito" fue el de los agentes de p-s y r-s a la vez, teniendo menos inconvenientes a la hora de tomar decisiones.

La hipótesis del párrafo anterior puede ser respaldada por los datos de tiempo de ejecución de los algoritmos (véase en la figura 4.4), que mantienen una cierta relación proporcional a los valores de desviación estándar en las recompensas de los agentes entrenados.

Si tomamos en cuenta el tiempo de ejecución de los algoritmos y la desviación estándar de la recompensa de los agentes entrenados, se puede concluir que la combinación de las dos técnicas de aprendizaje por refuerzo interactivo ayudan al aprendizaje del agente, incluso más que éstas implementadas por separado. No obstante tampoco se obtiene una mejora considerable en términos de tiempo de ejecución comparado a una implementación de solo policy-shaping, pero si habláramos de algún otro tipo de escenario experimental, uno que implique más números de estados o par estado-acción, ésta pequeña ventaja podría significar una reducción de tiempo considerable. La misma ventaja aplicaría al aumentar el número de agentes, ya que en este experimento solo se ocuparon 20 agentes por cada configuración para cada experimento.

Para terminar se concluye que los objetivos de este proyecto fueron cumplidos, logrando encontrar implementar técnicas novedosas para manejar un dron con una interfaz natural de usuario (específicamente comandos de voz), y con grandes resultados a la hora de interpretar los movimientos del dron. A su vez se lograron implementar distintas configuraciones de agentes de aprendizaje por refuerzo interactivo, las cuales dieron resultados bastante positivos, sobre todo para la hipótesis

planteada al principio del documento, la cual buscaba estudiar la efectividad de de la combinación de "Policy-Shaping" y "Reward-Shaping".

5.1. Trabajos Futuros

En un futuro se espera realizar una extensión de esta tesis, complejizando el escenario experimental para que se parezca más a un entorno doméstico. A su vez implementarlo en un entorno real y que el agente aprenda a manejar el vehículo aéreo, pero esta vez ocupando técnicas de aprendizaje profundo, para que el agente de aprendizaje por refuerzo pueda ser expuestos a escenarios distintos, pero de igual forma que pueda realizar la tarea.

Apéndice A

Publicaciones Originadas de este Trabajo

A raíz de esta memoria, se ha generado la publicación de un artículo científico, el cual es mostrado a continuación:

- Contreras, Ruben; Ayala, Angel; Cruz, Francisco. Unmanned aerial vehicle control through domain-based automatic speech recognition. *Journal Multidisciplinary Digital Publishing Institute, Computers*, 2020, vol. 9, no 3, p. 75.

Apéndice B

Lista de Acrónimos

AI – Artificial Intelligence.

ANN – Artificial Neural Network.

HRI – Human-robot Interaction.

IRL – Interactive Reinforcement Learning.

P-S – Policy Shaping.

R-S – Reward Shaping.

MDP – Markov Decision Process.

NN – Neural Network.

RL – Reinforcement Learning.

DNN – Deep Neural Network

CNN – Convolutional Neural Network

SARSA – State, Action, Reward, State, Action.

MC – Monte Carlo

TD – Temporal Difference

SNR – Signal-To-Noise Ratio

NPM – sin coincidencias de fonemas(Sin Coincidencia de Fonemas)

WPM – With Phoneme Matching(Con Coincidencia de Fonemas)

Apéndice C

Agradecimientos

En primer lugar quiero agradecer a mi tutor Francisco Javier Cruz Naranjo , quien con sus conocimientos y apoyo me guió a través de cada una de las etapas de este proyecto para alcanzar los resultados que buscaba. A su vez, el agradecimiento a mis profesores informantes Claudio Alex Henríquez Berroeta y Alejandro Antonio Sanhueza Olave, quienes siempre estuvieron para brindar una mano hacia mi persona cuando lo necesité durante mi proceso académico.

También quiero agradecer a la Universidad Central de Chile, por brindarme todos los recursos y herramientas que fueron necesarios para llevar a cabo el proceso de investigación. No hubiese podido arribar a estos resultados de no haber sido por su incondicional ayuda.

También agradecer a amigos, ya sea de lazos creados en la infancia o en la universidad, los cuales me otorgaron apoyo cuando lo necesitaba.

Por último, quiero agradecer a mi familia, por apoyarme en todo momento. En especial a mi madre Gloria Ortiz Navea y padre Claudio Contreras González, que siempre estuvieron ahí para darme palabras de apoyo y ánimos para seguir adelante.

Muchas gracias a todos.

Bibliografía

Ayala, A., Cruz, F., Campos, D., Rubio, R., Fernandes, B., and Dazeley, R. (2020). A comparison of humanoid robot simulators: A quantitative approach. In *Proceedings of the IEEE International Joint Conference on Development and Learning and Epigenetic Robotics ICDL-EpiRob*, page 6.

Carmona, F. (1972). Profundizacion de la dependencia tecnologica. *Problemas del Desarrollo*, 3(12):19–22.

Chandarana, M., Meszaros, E. L., Trujillo, A., and Allen, B. D. (2017). 'fly like this': Natural language interface for UAV mission planning. In *Proceedings of the 10th International Conference on Advances in Computer-Human Interactions (ACHI 2017)*, pages 40–46, Nice, France. IARIA XPS Press.

Cruz, F., Parisi, G. I., and Wermter, S. (2018). Multi-modal feedback for affordance-driven interactive reinforcement learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Cruz, F., Twiefel, J., Magg, S., Weber, C., and Wermter, S. (2015). Interactive reinforcement learning through speech guidance in a domestic scenario. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1341–1348, Killarney, Ireland. IEEE.

Dans, E. (2019). El machine learning y sus sesgos. Nombre - Ruby on Rails; Goldman Sachs Group Inc; Copyright - Copyright Newstex Nov 12, 2019; Última actualización - 2019-11-12.

Fayjie, A. R., Ramezani, A., Oualid, D., and Lee, D. J. (2017). Voice enabled smart drone control. In *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 119–121, Milan, Italy. IEEE.

Fernandez, R. A. S., Sanchez-Lopez, J. L., Sampedro, C., Bavle, H., Molina, M., and Campoy, P. (2016). Natural user interfaces for human-drone multi-modal

interaction. In *Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1013–1022, Arlington, VA, USA. IEEE.

Haluani, M. (2014). La tecnología aviónica militar en los conflictos asimétricos: historia, tipos y funciones de los drones letales. *Cuestiones Políticas*, 30(52):46–89.

Jones, G., Berthouze, N., Bielski, R., and Julier, S. (2010). Towards a situated, multimodal interface for multiple UAV control. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pages 1739–1744, Anchorage, Alaska USA. IEEE.

Kaushik, D., Jain, R., et al. (2014). Natural user interfaces: Trend in virtual interaction. *arXiv preprint arXiv:1405.0101*.

Landau, M. and van Delden, S. (2017). A system architecture for hands-free UAV drone control using intuitive voice commands. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '17, page 181–182, New York, NY, USA. Association for Computing Machinery.

Lavrynenko, O., Konakhovych, G., and Bakhtiiarov, D. (2016). Method of voice control functions of the uav. In *Proceedings of the 2016 IEEE 4th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC)*, pages 47–50, Kyiv, Ukraine. IEEE.

Lavrynenko, O., Taranenko, A., Machalin, I., Gabrousenko, Y., Terentyeva, I., and Bakhtiiarov, D. (2019). Protected voice control system of uav. In *Proceedings of the 2019 IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD)*, pages 295–298, Kyiv, Ukraine. IEEE.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710.

Liu, W. (2010). Natural user interface-next mainstream product user interface. In *2010 IEEE 11th International Conference on Computer-Aided Industrial Design & Conceptual Design 1*, volume 1, pages 203–205. IEEE.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Nola, R. and Sankey, H. (2014). *Theories of scientific method: an introduction*. Routledge.

Puterman, M. L. (2014). *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Quigley, M., Goodrich, M. A., and Beard, R. W. (2004). Semi-autonomous human-uav interfaces for fixed-wing mini-uavs. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2457–2462, Chicago, IL USA. IEEE.

Rohmer, E., Singh, S. P., and Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems IROS*, pages 1321–1326, Tokyo, Japan. IEEE.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: Bradford Book.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Sánchez, M. (2018). En fronteras, aeropuertos y agricultura, los mil usos de los drones en américa latina.

Valavanis, K. P. and Vachtsevanos, G. J. (2015). *Handbook of unmanned aerial vehicles*. Springer.

Yu, Y., Soung, C. L., and Wang, T. (2018). Carrier-sense multiple access for heterogeneous wireless networks using deep reinforcement learning.