Universidad ORT Uruguay

Facultad de Ingeniería

Optimización de sistemas hidropónicos basados en Inteligencia Artificial

Entregado como requisito para la obtención del título Master en Ingeniería

Mateus Würth - 147799

Tutores: Sergio Yovine, Franz Mayr

2023

Declaración de autoría

Yo, Mateus Würth, declaro que el trabajo que se presenta en esta obra es de mi propia mano. Puedo asegurar que:

- La obra fue producida en su totalidad mientras realizaba el proyecto final del Master en Ingeniería;
- Cuando he consultado el trabajo publicado por otros, lo he atribuido con claridad;
- Cuando he citado obras de otros, he indicado las fuentes. Con excepción de estas citas, la obra es enteramente mía;
- En la obra, he acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, he explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Mateus Würth
28 de febrero de 2023

Abstract

Este proyecto en la modalidad de trabajo de investigación se encuentra fuertemente vinculado a los temas abordados en los cursos y proyectos de investigación de la cátedra de Inteligencia Artificial y Big Data. Concretamente, en este trabajo se busca extender y brindar inteligencia a un prototipo de hardware desarrollado previamente (realizado para la tesis de Ingeniería en sistemas). Como insumo generado por la tesis de ingeniería se obtuvo un ambiente(prototipo) de control de plantas en un medio hidropónico, el cual contempló la integración de sensores y actuadores electrónicos que interactúan con el fin de optimizar el crecimiento de las mismas. A lo largo de este trabajo se profundizó en la virtualización de dicho ambiente, además de la generación de modelos de inteligencia artificial con *Reinforcement Learning* que logren optimizar el crecimiento de las plantas, a través de la búsqueda de políticas óptimas en la ejecución de acciones.

Palabras Clave

Aprendizaje por Refuerzo; DQN; PPO; A2C Hidroponia; NPK; Calidad de Agua; Calidad de aire; Arduino; Raspberry Pi; Serial; Baseline3

Índice general

Capítulo 1

Introducción

Según Arely Nohemí López [1] hidroponia, agricultura hidropónica o "soilless culture", es el cultivo de plantas en ausencia de tierra. La palabra hidroponia proviene del lenguaje griego hidro que significa agua y ponos cuyo significado es labor o trabajo (RAE, 2001). En esta técnica de cultivo, los nutrientes que la planta requiere para su crecimiento y que normalmente encuentra en la tierra; son suministrados en cantidades mínimas en el agua, lo que conlleva al desarrollo de cultivos sanos en espacios relativamente menores en comparación con la agricultura tradicional (Antillón, 2004).

Las plantas se denominan organismos autótrofos, esto quiere decir que fabrican sus propios alimentos. Para poder realizarlo requieren materias primas y energía. La energía la obtienen de la luz del sol y/o artificial, mientras que las materias primas del medio en el cual son alojadas (tierra, agua o algún otro sustrato) .

Este trabajo tiene como objetivo extender la investigación realizada previamente para la tesis de ingeniería [2], profundizando en la generación de agentes inteligentes que logren optimizar el crecimiento de las lechugas en un ambiente hidroponico, además de generar modelos virtuales de crecimiento de las mismas, con el fin de acelerar los procesos de entrenamiento y prueba. En la etapa anterior se construyó un ambiente capaz de medir N (Nitrógeno), P (Fósforo), K (Potasio); de forma independiente, pH, CE (Conductividad Eléctrica), luminosidad, gases (C02 y otros), temperatura, humedad en el aire. Por otro lado, tener la capacidad de controlar actuadores (calefactores, extractores, aireadores, bombas peristálticas) para regular los niveles antes mencionados. Y con dicho ambiente, se realizó una prueba de concepto en la generación de un agente inteligente centrado en el control de temperatura optimizado por consumo [2].

Un agente inteligente es una entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en su entorno de manera racional.
Se usa el término `percepts` (percepción) para referirse al contenido que perciben los sensores de un agente. En general, la elección de acción de un agente en un instante dado puede depender de su conocimiento incorporado y de toda la secuencia de percepción observada hasta la fecha, pero no de nada que no haya percibido.

En la figura 1.1 se puede visualizar el flujo de comunicación entre el ambiente y el agente. En este caso particular los sensores y actuadores pertenecen al ambiente y los mismos son gestionados por el agente.

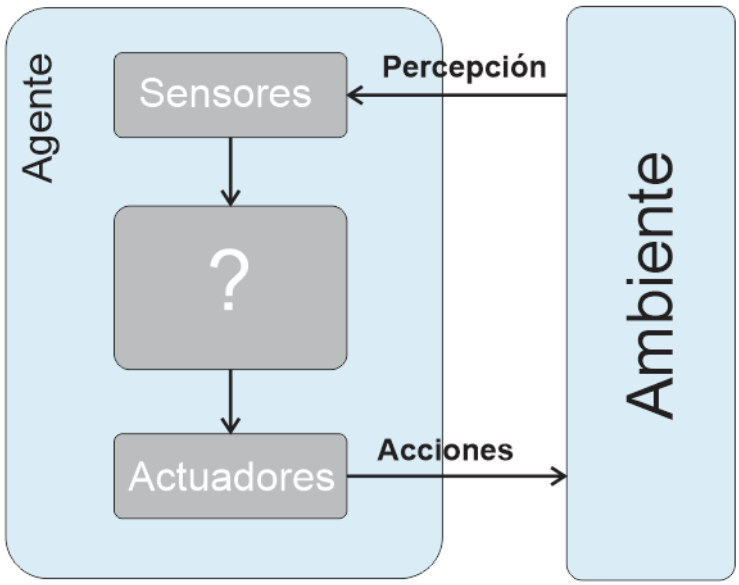


Figura 1.1

A lo largo del avance del proyecto se fueron identificando otros proyectos y artículos entorno a la optimización de crecimientos de lechugas en ambientes controlados. En concreto Mokhtar A et. al [3] estudiaron la predicción del rendimiento de la lechuga (peso fresco) utilizando cuatro modelos de aprendizaje automático (ML), a saber, regresor de vector de soporte (SVR), aumento de gradiente extremo (XGB), bosque aleatorio (RF) y red neuronal profunda (DNN). Indicaron que el potencial del modelo DNN para predecir el rendimiento de la lechuga fresca es prometedor y se puede aplicar a gran escala como una herramienta rápida para que los responsables de la toma de decisiones gestionen el rendimiento de los cultivos. Tom Mitchell define el machine learning (ML) en uno de sus libros como: "El estudio de algoritmos de computación que mejoran automáticamente su rendimiento gracias a la experiencia".

Ahsan, M et. al [4] afirman que las aplicaciones de aprendizaje profundo (DL) y visión por computadora en agricultura de precisión tienen un gran potencial para identificar y clasificar especies de plantas y vegetación. En su caso prepararon y utilizaron cuatro concentraciones de nutrientes diferentes (soluciones de nitrógeno de 0, 50, 200, 300 ppm) para cultivar lechugas en invernadero. Capturaron imágenes Red-Green-Blue (RGB) de hojas de lechuga. Implementaron modelos de redes neuronales de convolución para identificar los niveles de nutrientes de las lechugas estudiadas con fines comparativos y lograron una precisión del 87,5 al 100 % realizándolo.

Wouter van den Bemd en su estudio [5] comparó las adaptaciones de dos algoritmos de aprendizaje por refuerzo de última generación, PPO (Proximal Policy

Optimization)[6] y SAC (Soft Actor-Critic), para optimizar las ganancias de los productores en invernadero hidropónico de lechuga. En su análisis, entrenó los algoritmos en un solo invernadero y los evaluó en muchos invernaderos diferentes. Si bien la versión simple de PPO y SAC afirmaron funciona mejor que las líneas de base, observó severa degradación del rendimiento cuando se modificó el invernadero de evaluación.

Alipio et. al [7] presentan un sistema controlado mediante Arduino en el cual comparan los resultados obtenidos entre el sistema hidropónico clásico y la gestión mediante redes bayesianas, siendo el último caso el que logra optimizar los resultados. Cabe destacar que tanto en este trabajo como en otros la aplicación de inteligencia artificial se ve limitada a algunos controles y se realiza mediante aprendizaje pasivo .

Dbritto y Hamdare [8] propusieron preparar un sistema de Inteligencia Artificial para realizar cultivos hidropónicos en un ambiente cerrado que automáticamente entregará una mezcla de agua y una solución nutritiva, directamente a las raíces de las plantas usando sensores.

Ruizhe Yang et. al [9] se concentraron en detectar hojas amarillas, ya que estas son los principales tipos de hojas anormales en la lechuga hidropónica. Este tipo de anomalías se corresponde con deficiencias en las condiciones óptimas. Utilizaron Regresión lineal múltiple (MLR), K-Nearest Neighbor (KNN) y Support Vector Machine (SVM) con el fin de detectar estas anomalías. MLR obtuvo precisiones de detección de 89,48 % y 99,29 % para hojas amarillas y podridas, respectivamente, mientras que SVM alcanzó 98,33 % y 97,91 %, respectivamente.

Sidinei José Lopes et. al [10] propusieron modelos para estimar la variación temporal de la acumulación de materia seca, donde los mejores resultados para el desarrollo relativo se obtuvieron utilizando grados-día efectivos, caracterizando la importancia de la temperatura del aire para la fase vegetativa y la radiación solar para la reproductiva. Estos modelos, como se describirá más adelante fueron los utilizados para entrenar dos de los agentes inteligentes propuestos en este trabajo.

Si bien se identificaron diversos artículos en el área, ninguno de estos disponibilizó modelos virtuales del crecimiento de la lechuga ni datos suficientes para entrenar modelos, con excepción del último descripto.

Para lograr cumplir con los objetivos de este trabajo se plantearon las siguientes tareas:

- Puesta a punto del ambiente anterior, prueba con DQN(Deep Q-Network) y otros algoritmos como PPO(Proximal Policy Optimization)[6] y A2C (Advantage Actor Critic) [11].
- Modelado del ambiente real a virtual con todas sus características.
- Investigar de la existencia de modelos virtuales del crecimiento de las lechugas.

- Experimentar con alguno de los modelos encontrados. Esto permitirá implementar un sistema base, el cual al cambiar el modelado de la lechuga permita encontrar nuevos agentes que optimicen el crecimiento en base a diferentes parámetros.
- Crear un nuevo ambiente capaz de incrementar la velocidad de pruebas y generar modelados virtuales del crecimiento de las lechugas. Además de determinar valores óptimos de todos los parámetros para el crecimiento de las mismas.
- Generar ambientes Gym [12] capaces de comportarse como el ambiente real e interactuar con el mismo.
- Entrenar y evaluar modelos de Inteligencia Artificial con reinforcement learning.

La estructura del documento es la siguiente:

En el Capítulo (2) (**Virtual Lettuce Generator**), se detallan aspectos relativos al diseño, construcción, programación, integración y prueba del nuevo ambiente.

En el Capítulo (3) (**Agentes Inteligentes**), se detalla el diseño de 5 agentes inteligentes, especificando acciones posibles para cada uno, modelado del ambiente, descripción de steps, descripción de episodios, funciones de recompensas y pruebas.

Finalmente las conclusiones (Capítulo 4) y anexos (Capítulo 6).

Capítulo 2

Virtual Lettuce Generator (VLG)

2.1. Objetivos

El objetivo principal de este trabajo es aplicar técnicas, métodos y prácticas de ML e IA con el fin de entrenar agentes inteligentes que logren optimizar el crecimiento de las lechugas en un ambiente hidropónico. Estos agentes logran automatizar los procesos, reducir el error humano y mejoran la toma de decisiones en diferentes situaciones. La interacción para entrenar estos agentes puede ser realizada directamente con plantas o modelos virtuales de las mismas. Las primeras pruebas se realizaron utilizando plantas reales, pero fue rápidamente descartado principalmente por dos razones; la necesidad de la interacción humana constante para cambiar la planta en caso de ser necesario y el costo de reducir los nutrientes en el agua, cuando por acciones incorrectas se contamina el medio. Dada la escasez de modelos virtuales empíricos del crecimiento de las lechugas en ambientes hidropónicos (en base a datos sensibles como nutrientes, luminosidad y temperatura), se fijo un nuevo objetivo, desarrollar un nuevo ambiente. El propósito de dicho ambiente es acelerar los procesos de recolección de información valiosa(imágenes, datos de los sensores y actuadores) y de esa forma construir modelos virtuales de crecimiento utilizando técnicas de Machine Learning en base a todos los parámetros disponibles (temperatura, humedad, luminosidad y nutrientes). A este nuevo ambiente se le denominó Virtual Lettuce Generator y no es un reemplazo del anterior (GreenLab) [2], sino un complemento.

2.2. Diseño y Funcionalidades

A continuación en la figura 2.1 se puede visualizar el modelado 3D del mismo. En el anexo 6.2 se provee un resumen del ambiente desarrollado en la tesis de ingeniería.

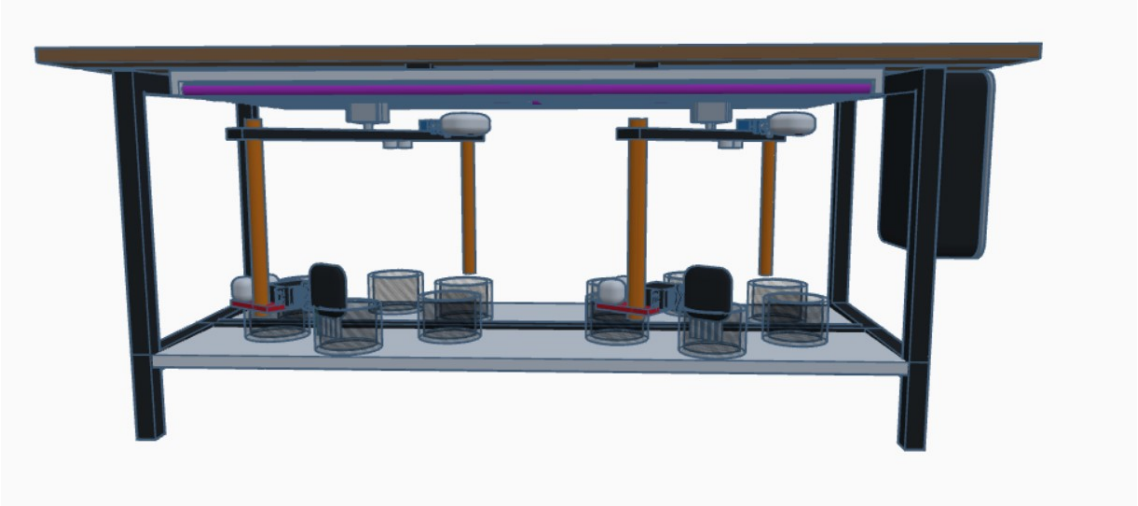


Figura 2.1: Diseño del dispositivo

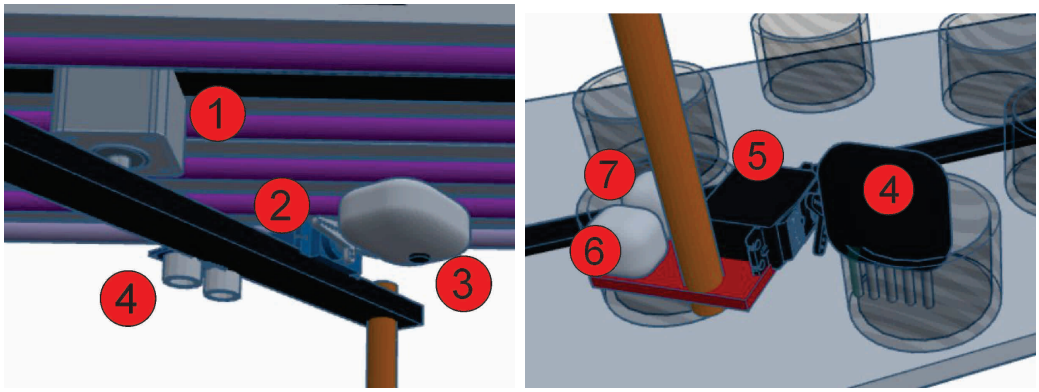
Este presenta varias diferencias de diseño respecto del anterior:

- Dos hélices de comportamiento idéntico que giran en la zona superior del dispositivo para obtener información de las plantas.
- Recipientes individuales, brindando la posibilidad de aumentar la variabilidad en la nutrición de las lechugas.
- Capacidad de medir la evolución de los nutrientes en cada estación de las 12 disponibles. Por estación se hace referencia al lugar de cada planta.
- Capacidad de medir luminosidad, temperatura y humedad de cada estación de las 12 disponibles.
- Capacidad de extraer la dimensión de las plantas a través de información visual con cámaras y sensores en cada estación de las 12 disponibles.

Con las funcionalidades descriptas anteriormente, es posible etiquetar cada foto obtenida con su respectivo contexto y utilizar dicha información para entrenar modelos virtuales del crecimiento de las mismas.

A continuación se describirán las distintas partes del ambiente desarrollado.

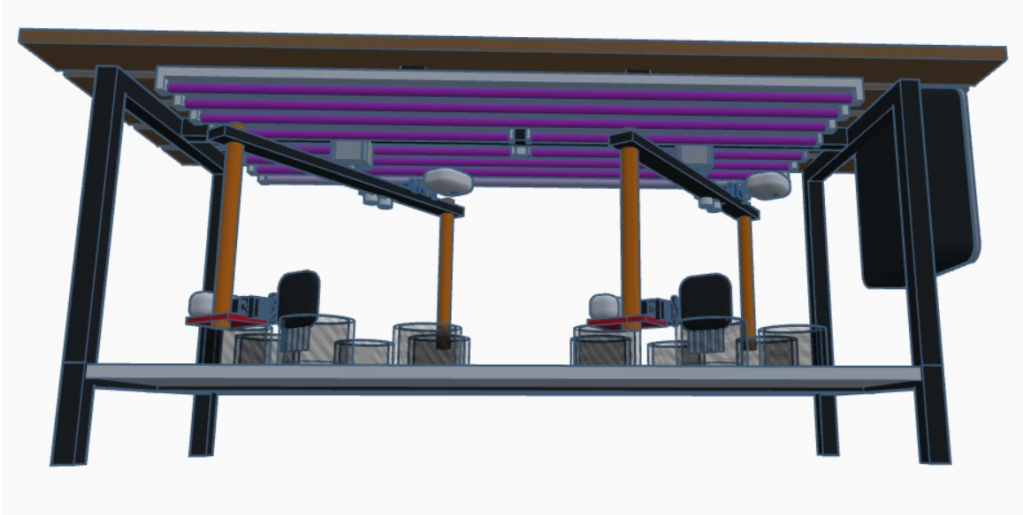
1. Motor paso a paso que brinda el giro correcto a la hélice para ubicar los sensores y la cámara en cada planta.
2. Servo motor que regula el ángulo de enfoque de la cámara.
3. Cámara FTP(File Tranfer Protocol) que envía foto al servidor. La cual es luego tratada para extraer información del tamaño de la lechuga.
4. Sensor de distancia para determinar la altura de la planta.
5. Sensor NPK(Nitrógeno , Potasio y Fósforo).



(a) Información Visual (b) Sensores

Figura 2.2: Sistema de control

6. Servo motor que varia su posición de 0 a 90° con el fin de insertar o extraer el sensor NPK de cada recipiente.
7. Sensor de temperatura y humedad.
8. Sensor de luminosidad.



(a) Versión inicial

Figura 2.3: Diseño del mueble - Vista inferior

En la figura 2.3a se puede visualizar en violeta los 10 tubos de luz de amplio espectro con mando independiente. Dependiendo de los que se enciendan varia la luminosidad en los diferentes puntos donde se ubiquen de lechugas.
Por último el ambiente tiene la capacidad de controlar un aire acondicionado preconfigurado con el fin de variar la temperatura del exterior y en consecuencia la de las plantas. A continuación se presentan consideraciones del ambiente y su funcionamiento.

2.2.1. Funcionamiento del ambiente

Consideraciones:

- El centro de todos los recipientes esta ubicado sobre una circunferencia con centro en el eje del motor y en la base del ambiente.
- El sensor de distancia que mide la altura de la lechuga y la cámara están ubicados sobre la misma circunferencia pero a nivel de la hélice.
- La distancia angular entre los recipientes en los cuales se ubican las lechugas es idéntica.

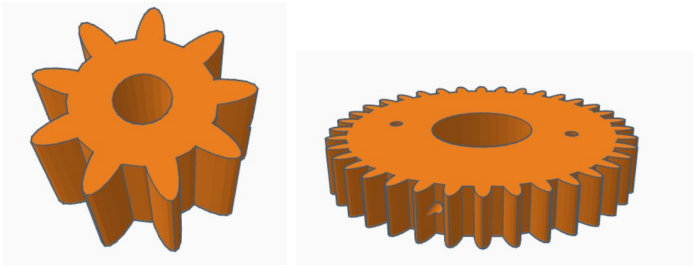
Funcionamiento:

1. Ambas hélices giran en sentido antihorario visualizado desde abajo hasta presionar el botón de corte de giro que marca la posición 0°. Este botón es necesario para tener una referencia de hardware del grado 0. En caso de no tenerlo el sensor de nutrientes podría no ubicar los recipientes y desfasarse.
2. La hélice derecha gira 180° y se ubica en posición de descanso para que la izquierda pueda girar libremente y no choquen los sensores.
3. La hélice derecha gira hasta encontrar el primer recipiente con una lechuga y acciona el servo-motor realizando un giro de 90° con el fin de introducir el sensor de nutrientes. El sensor pasa de estar horizontal a estar vertical y listo para medir.
4. Realiza la medición de nutrientes.
5. Realiza la medición de temperatura, humedad y luminosidad.
6. Se almacena dicha información a la espera de etiquetar la foto.
7. El sensor de nutrientes vuelve a estado horizontal y continua con el giro pronto para el siguiente recipiente.
8. Continua con el proceso de medición en todas las estaciones.
9. Al finalizar vuelve a la posición 0^{o}.
10. La hélice derecha se reubica girando 180^{o} para comenzar con la toma de fotografías. Esto se debe a que el sensor de distancia y la cámara están ubicados de forma opuesta a los sensores que antes realizaron las mediciones.
11. Al finalizar vuelve a la posición 0^{o}.
12. Comienza un giro lento para censar las altura de las lechugas.

13. Se envían las fotografías y los datos de los sensores al servidor.
14. La hélice derecha realiza todos los mismos pasos.
15. Ambas hélices vuelven a su posición inicial.

2.2.2. Piezas 3D generadas para la construcción del ambiente

Estas piezas fueron diseñadas y quedan disponibles en el repositorio 6.

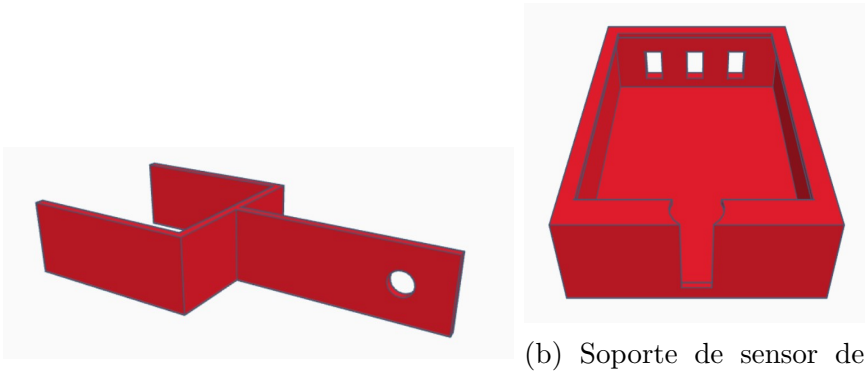


(a) Polea al motor (b) Polea de aumento de torque

Figura 2.4: Engranajes

- La polea al motor visualizada en la figura 2.4a es ajustada al eje para trasladar con los dientes del engranaje su movimiento.
- La polea de aumento de torque visualizada en la figura 2.4b es para anclar a la hélice y aumentar la fuerza para el giro de la misma.

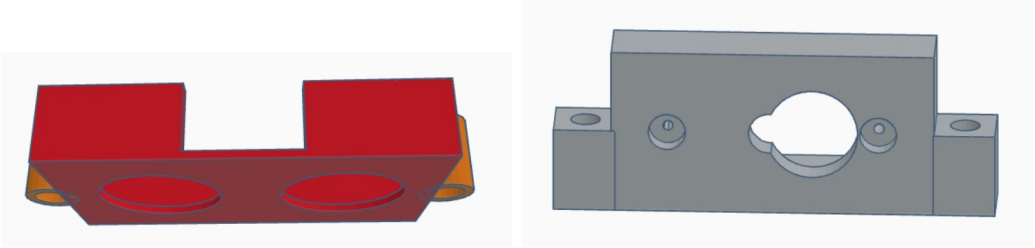
Las dos piezas descriptas anteriormente finalmente no fueron utilizadas dado que no aportaban ventajas considerables. Si bien incrementaban el torque del motor, reducía la precisión de giro por la dimensión de los dientes de las poleas. Al realizar la prueba sin estas piezas, se logró un funcionamiento acorde a lo esperado y se descarto su utilización.



(a) Soporte del botón limitador (b) Soporte de sensor de nutrientes móvil

Figura 2.5: Soportes

- El soporte del botón visualizado en la figura 2.5a ubica al botón en una posición adecuada del giro para presionarse cuando la hélice se ubica en los 0^{o}.
- El soporte del sensor de nutrientes visualizado en la figura 2.5b ancla el sensor al servo motor, con el objetivo de copiar el movimiento del eje del motor.



(a) Soporte de sensor de distancia (b) Soporte de servomotor

Figura 2.6: Soportes

- El soporte del sensor de distancia visualizado en la figura 2.6a ancla al sensor de distancia sobre la hélice en la posición correcta.
- El soporte del servo motor visualizado en la figura 2.6b ancla al servo motor sobre la hélice en la posición correcta, para controlar el movimiento de la cámara superior.

2.3. Variables Observadas

Las variables observadas en VLG son un subconjunto de las variables de GreenLab [2], esto es debido a la dificultad de implementación y el costo económico que requería.
Dichas variables son:

1. **Temperatura**, la medición es realizada en °C.
2. **Humedad**, la medición es realizada en %.
3. **Luminosidad**, la medición es realizada en Luxes.
4. **Nitrógeno**, la medición es realizada en ppm(partes por millón).
5. **Fósforo**, la medición es realizada en ppm(partes por millón).
6. **Potasio**, la medición es realizada en ppm(partes por millón).
7. **Largo, ancho y altura de la planta**, la mediciones son realizadas en cm. Para el largo y el ancho se utiliza la cámara mientras que para la altura el sensor de distancia.

2.4. Actuadores

Los actuadores son:

1. **Luces**, 10 tubos de luz con encendido independiente.
2. **Temperatura**, emisor infrarrojo con el fin de controlar un aire acondicionado.

3. **Motores de control**, dos motores para el control de las hélices, 2 servomotores para el control de las cámaras y 2 servomotores para el control del sensor de nutrientes.

2.5. Hardware

2.5.1. Descripción de Materiales utilizados

A continuación se describen todos los componentes de hardware que componen a VLG.

1. **Arduino Mega 2560**, es una tarjeta de desarrollo open-source construida con un microcontrolador modelo Atmega2560 que posee pines de entradas y salidas (E/S), analógicas y digitales. Esta tarjeta es programada en un entorno de desarrollo que implementa el lenguaje Processing/Wiring. Arduino puede utilizarse en el desarrollo de objetos interactivos autónomos o puede comunicarse a un PC a través del puerto serial (conversión con USB) [13].

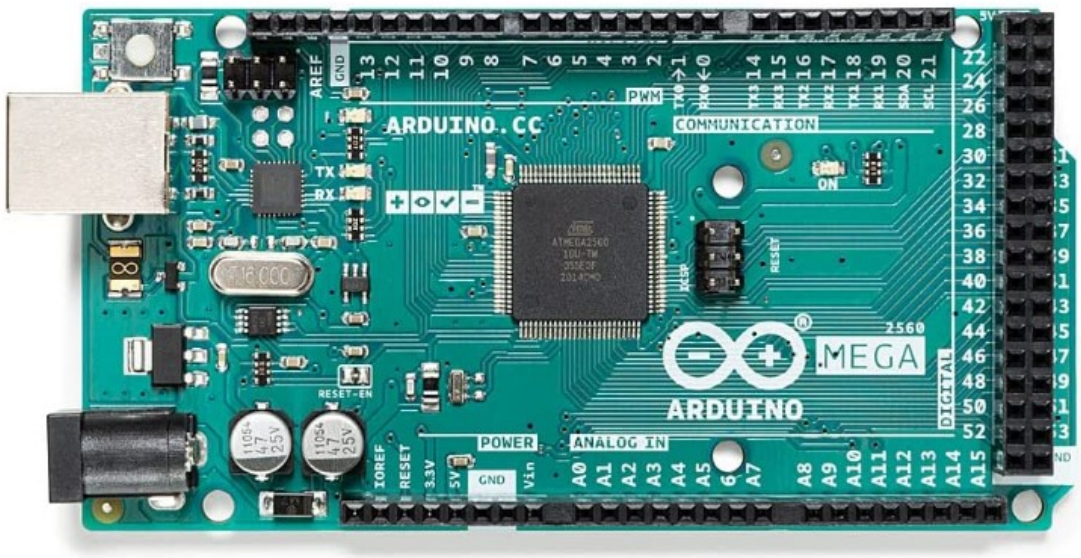


Figura 2.7: Arduino Mega

2. **Cámara D-Link DCS-930l**, es una cámara que permite visualizar en tiempo real el video y obtener imágenes para luego enviarlas por FTP a un servidor.

3. **DHT22 Sensor de Temperatura y Humedad**, permite monitorear temperatura y humedad relativa de forma precisa. La salida suministrada es de tipo digital.

4. **BH 1750 Sensor de Luminosidad**, es un sensor de luz ambiental de 16 bits, el cual retorna la luminosidad en luxes. El sensor se comunica con un microcontrolador mediante el protocolo de comunicación I2C.

Figura 2.8: Cámara

Figura 2.9: DHT22

Figura 2.10: BH-1750

5. **Servo SG90**, es un servomotor de 9 gramos que puede girar de 0 a 180 grados (aproximadamente) a una velocidad de aproximadamente 0,3 segundos.

Figura 2.11: Servo SG90

6. **Servo MG995**, es un servomotor de 58 gramos que puede girar de 0 a 180 grados (aproximadamente) a una velocidad de aproximadamente 0,17 segundos/60°, con mayor fuerza que el SG90.

Figura 2.12: Servo MG995

7. **Solid State relés**, una placa de 8 relés de estado sólido que permite controlar la fase de 220v de diferentes dispositivos.

Figura 2.13: Placa de relés de estado sólido

8. **Motor Nema 23**, es un motor paso a paso con una placa frontal de 2,3 × 2,3 pulgadas (58,4 × 58,5 mm) y un ángulo de paso de 1,8° (200 pasos/revolución). Cada fase consume 2,8 A a 3,2 V, lo que permite un par de retención de 19 kg-cm.

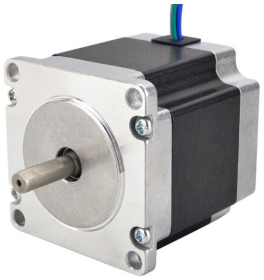


Figura 2.14: Motor Nema 23

9. **HC-SR40**, es un sensor de distancias por ultrasonido capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 2 a 450 cm.

Figura 2.15: Sensor de distancia

10. **Soil meter**, es un sensor que mide nitrógeno, potasio y fósforo del medio en el cual se coloque.

Figura 2.16: Soil Meter

11. **Max RS485**, es un conversor de RS485 a TTL que interpreta la salida del sensor y la comunica a la Arduino.

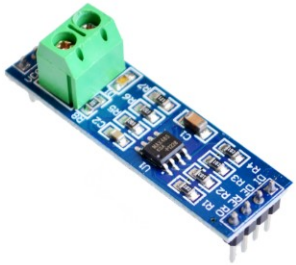


Figura 2.17: Motor Nema 23

2.5.2. Motores

En esta sección se describirá a alto nivel las conexiones realizadas entre los componentes para el funcionamiento adecuado de VLG. En la imagen 2.18 se puede visualizar las conexiones realizadas entre la Arduino Mega, los drivers de los motores y los motores. Con el fin de simplificar el diagrama y su entendimiento sólo se indica al pin al cual se conectaron. En rojo se puede visualizar los pines que ocuparon en el microcontrolador.

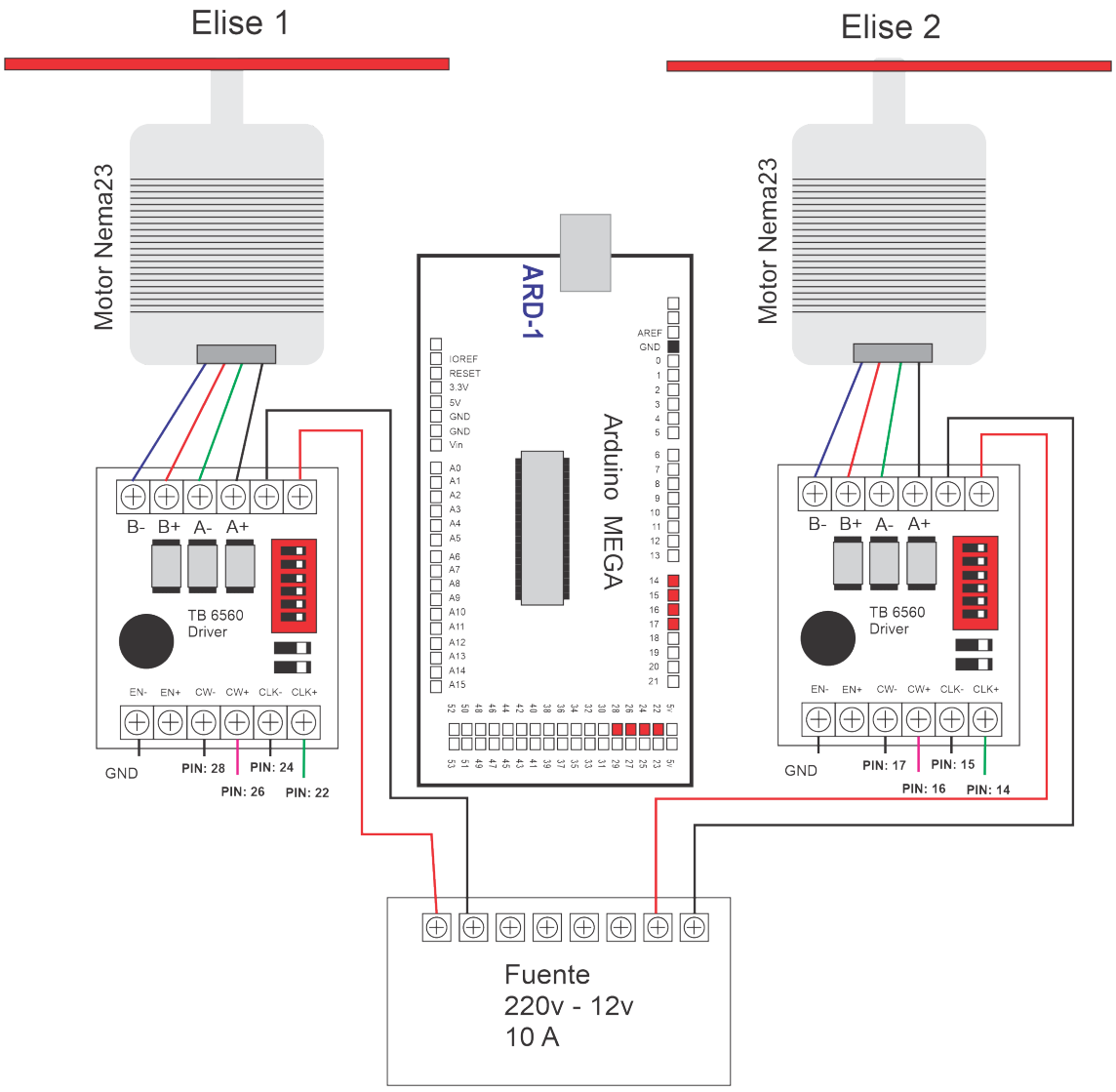


Figura 2.18: Conexión Motores y Drivers

2.5.3. Luces y Cámaras

En la imagen 2.19 se pueden visualizar las conexiones realizadas entre la Arduino Mega y las placas de relés que controlan las luces y cámaras de VLG. Los cilindros violetas en la parte inferior de la imagen son los tubos de luz de amplio espectro.

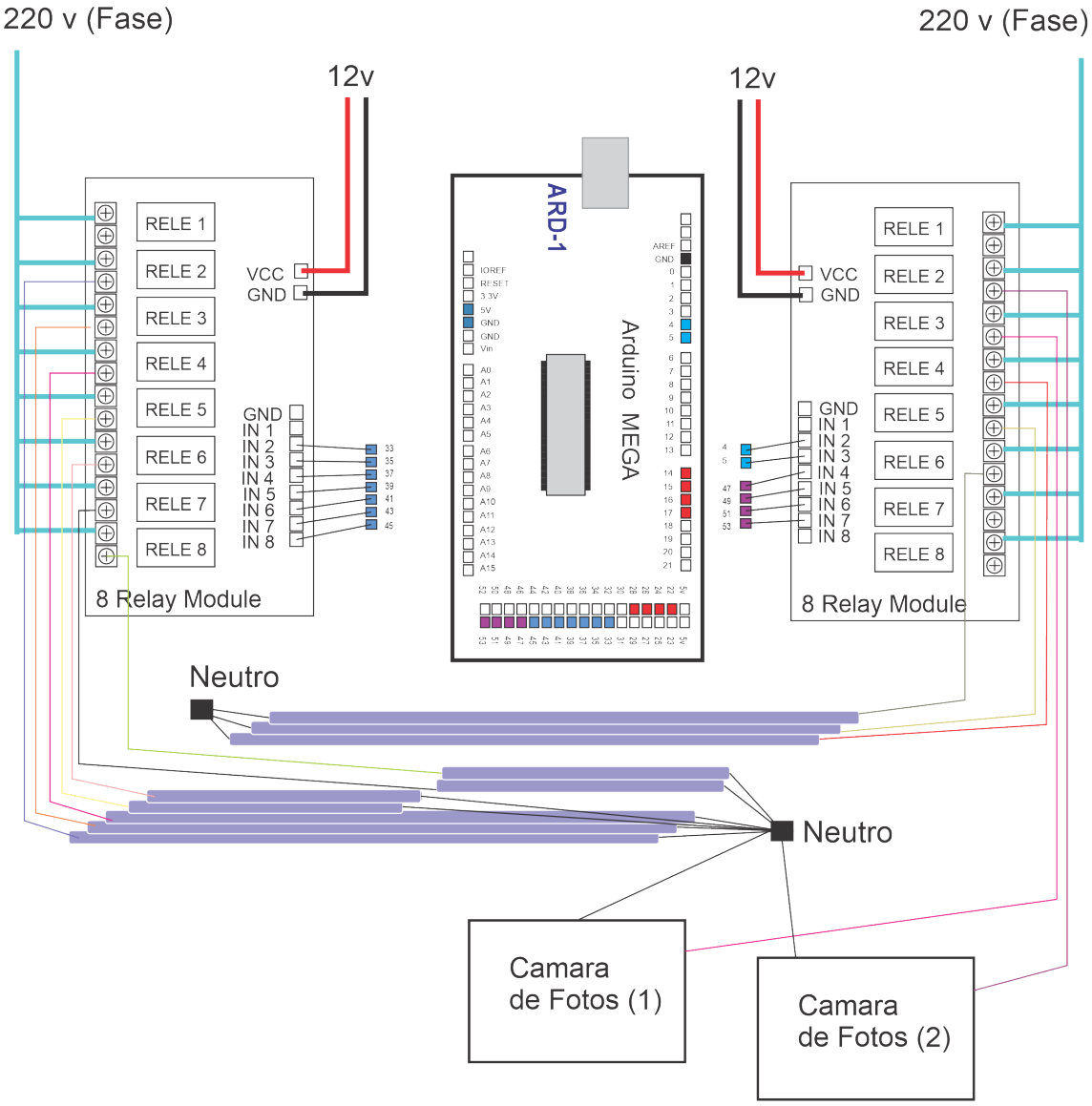


Figura 2.19: Conexión de luces de amplio espectro y cámaras

2.5.4. Sensores y actuadores inferiores

A continuación en la figura 2.20 se pueden visualizar las conexiones realizadas de la Arduino Mega con sensores y actuadores que miden nutrientes del agua, temperatura, humedad y luminosidad de las plantas.

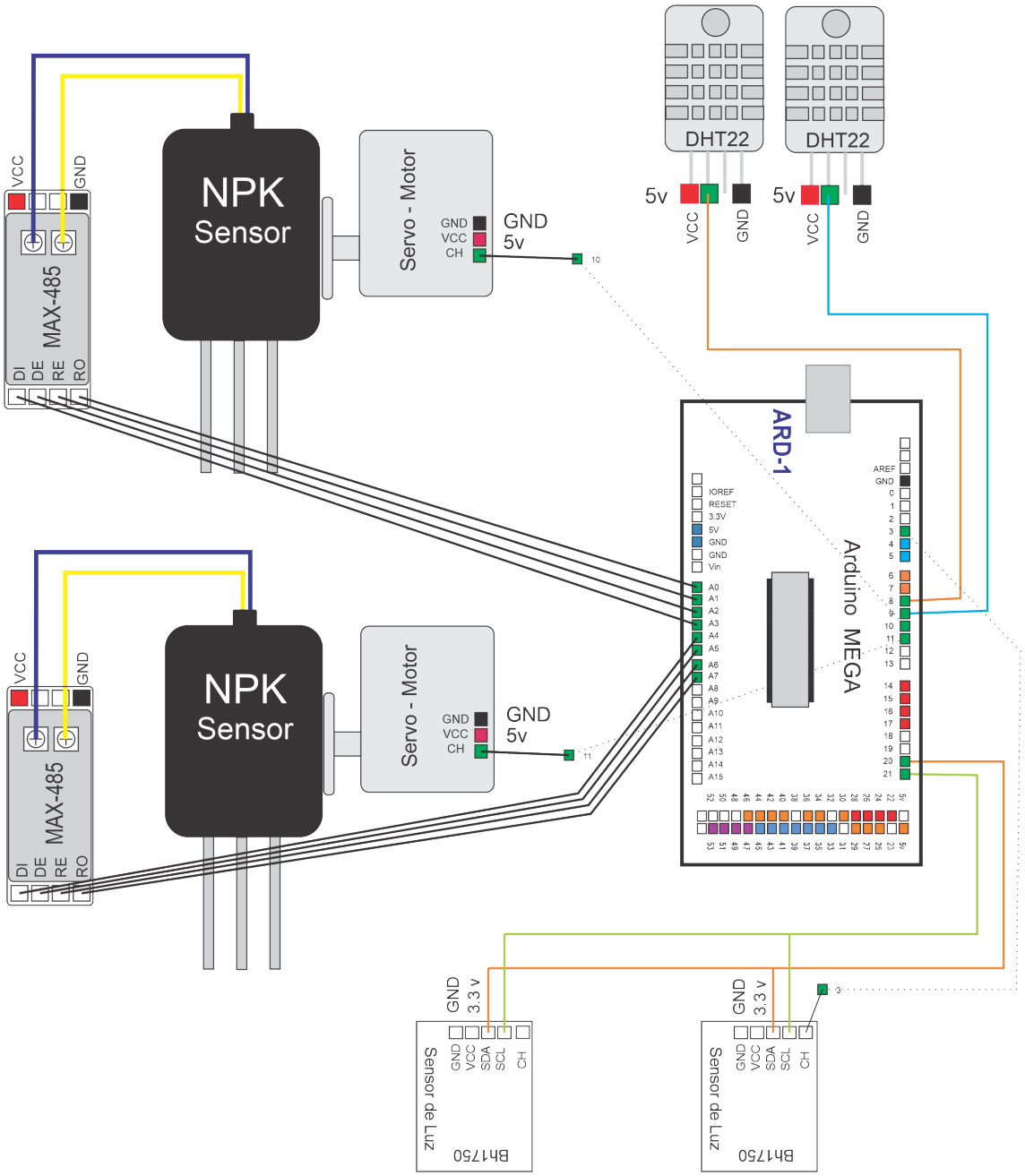


Figura 2.20: Sensores y actuadores de base

2.5.5. Sensores y actuadores de la hélice

Con el fin de tener un feedback claro del tamaño de las plantas, en las hélices se ubicaron cámaras y sensores de distancia que logran extraer la información deseada. En la imagen 2.21 se pueden visualizar todos los actores en el proceso de recolección de información de las plantas.

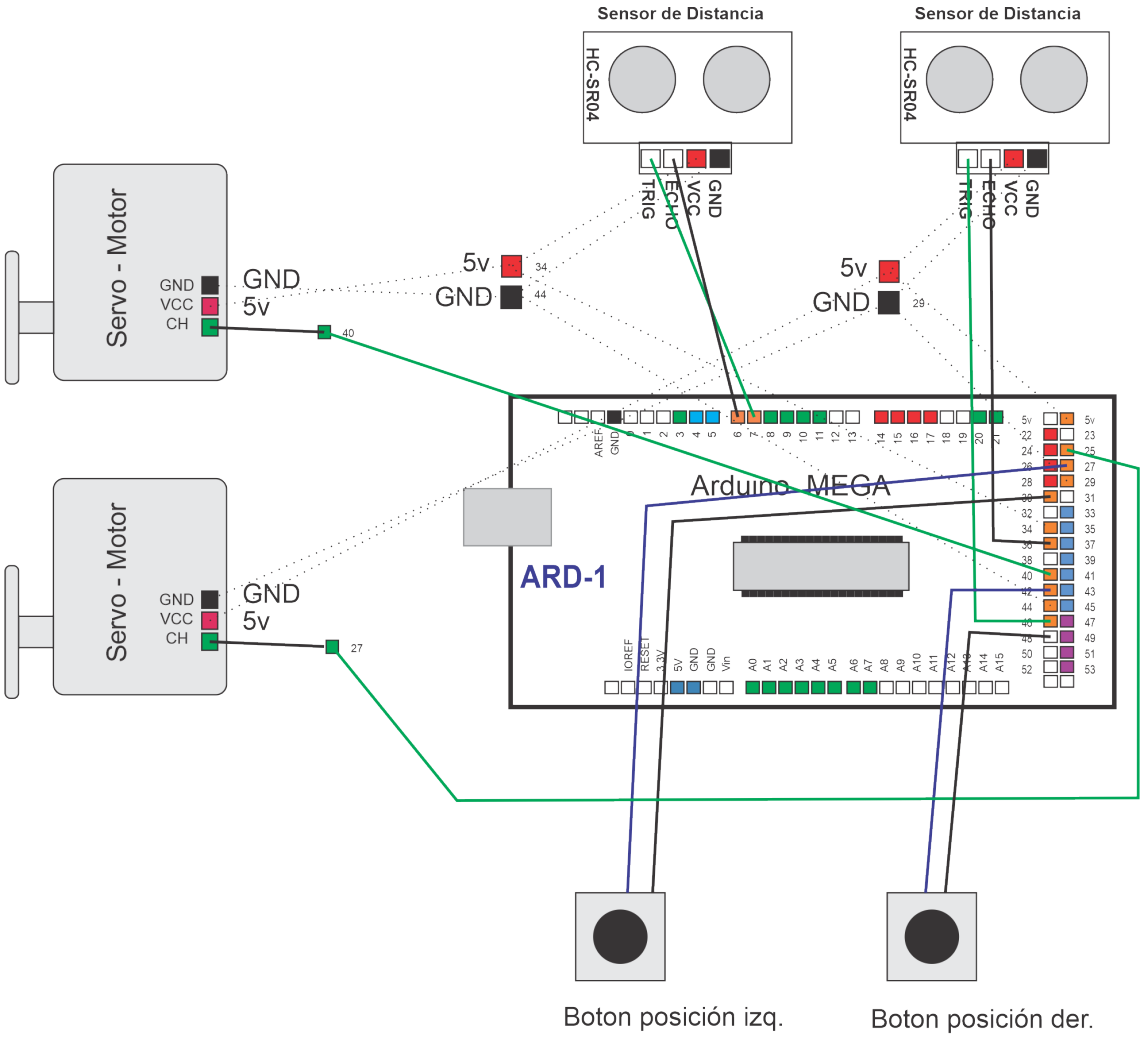


Figura 2.21: Sensores y actuadores de hélice

2.5.6. Comunicación

Una vez recolectada la información, la misma debe ser enviada al servidor. Esto es realizado en una primera instancia por Serial desde la Arduino Mega a un ESP-8266 y luego a una Raspberry Pi por Wifi. En la imagen 2.22 se puede visualizar la interacción entre los dispositivos.

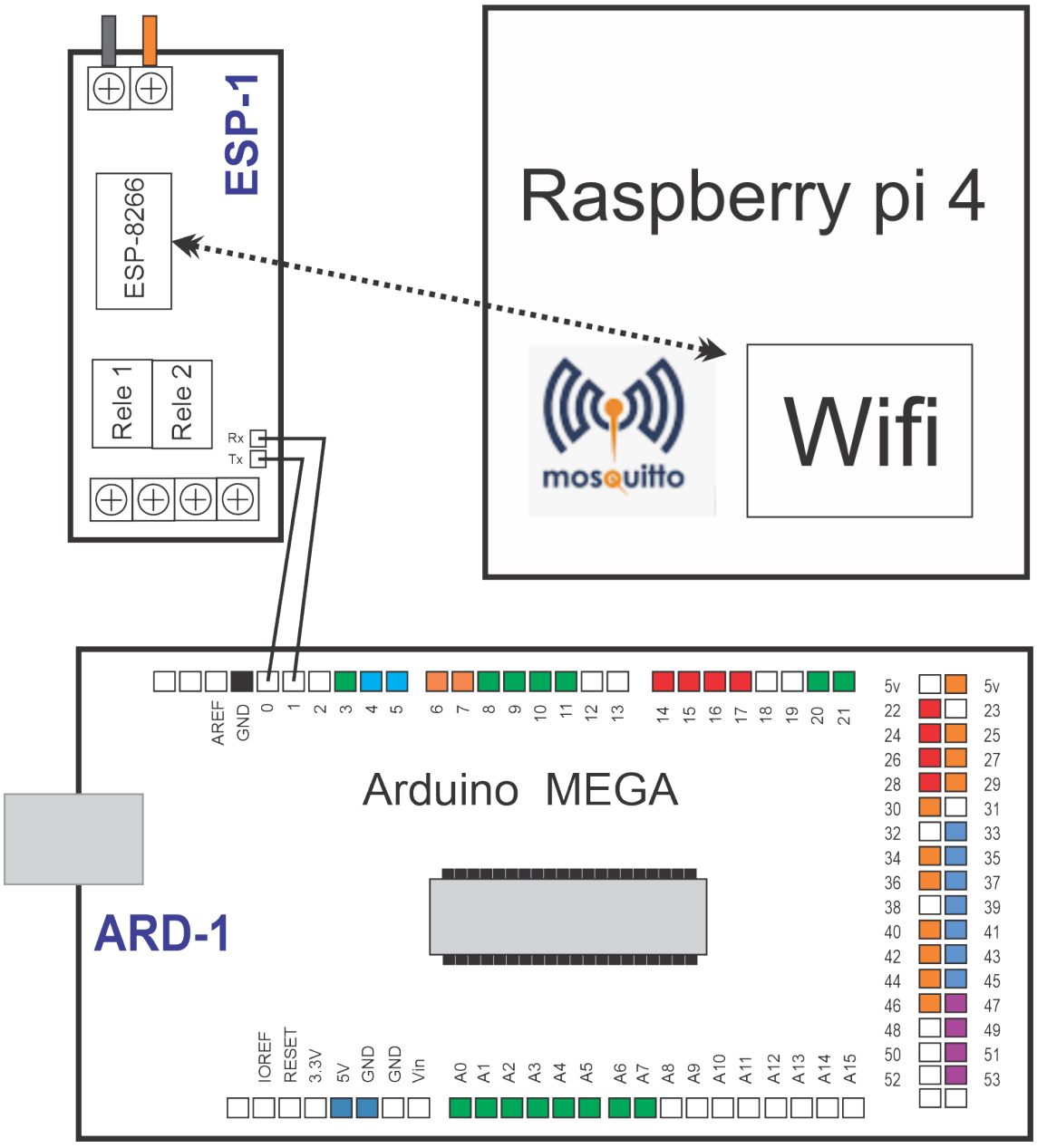


Figura 2.22: Comunicación

2.6. Software

2.6.1. Programación del microcontrolador

Para programar la Arduino Mega se utilizó el IDE de Arduino [13].

2.6.2. Extracción de información de la cámara

En la figura 2.23 a continuación se puede visualizar una imagen tomada por una de las cámaras de VLG. La imagen tomada a las lechugas tiene como objetivo extraer el ancho y largo de la planta para tener información del crecimiento. Para esto en primer lugar se trabajó con la imagen al natural para intentar extraer la información antes mencionada. Al tener diferentes objetos, colores y formas en el fondo de la misma, esto dificultó el trabajo con la imagen por lo cual se optó por tratarla antes de extraer la información deseada. Para intentar minimizar el ruido de la imagen de fondo, inicialmente se acondicionó alterando el ambiente. Posteriormente luego de que no funcionara adecuadamente en todas las oportunidades se procedió a alterar la imagen convirtiendo el fondo a color negro. Es importante mencionar que todas las imágenes deben pasar por este proceso automático el cual no requiere interacción humana.

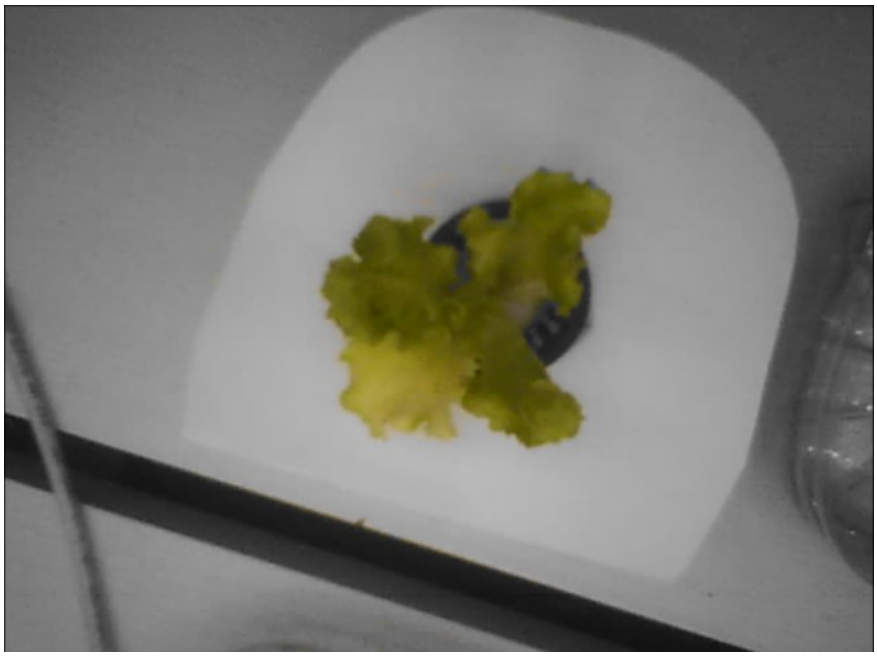


Figura 2.23: Lechuga natural

Para realizar todo el procesamiento de imágenes se utilizó OpenCV [14], la cual es una biblioteca multiplataforma con la que podemos desarrollar aplicaciones de Computer Vision en tiempo real. Se enfoca principalmente en el procesamiento de

imágenes, la captura y el análisis de videos, incluyendo funciones como la detección de rostros y la detección de objetos[15].
A continuación se detalla el proceso seguido:

1. En primer lugar se determinó experimentalmente el rango de colores con el cual se identifican a las lechugas, desde suaves hasta fuertes (verdes).

2. En segundo lugar se paso la imagen a escala de grises, como se puede visualizar en la figura 2.24.

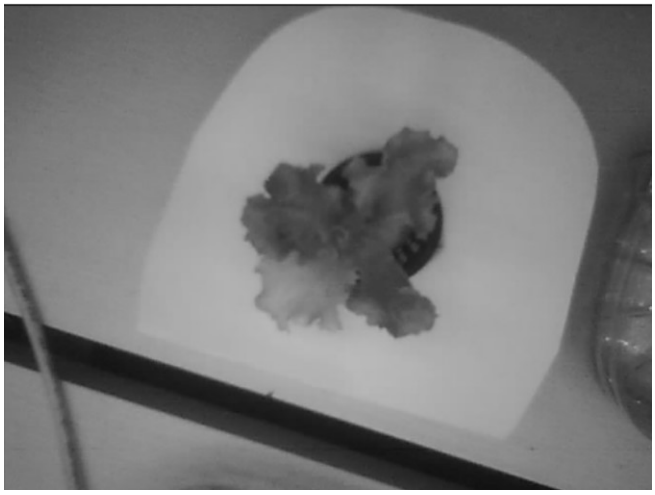


Figura 2.24: Sensores y actuadores de hélices

3. La imagen original se convirtió de formato RGB a HSV como se puede visualizar en el figura 2.41. El objetivo de este paso es clasificar los colores por tres características matiz, saturación, valor.

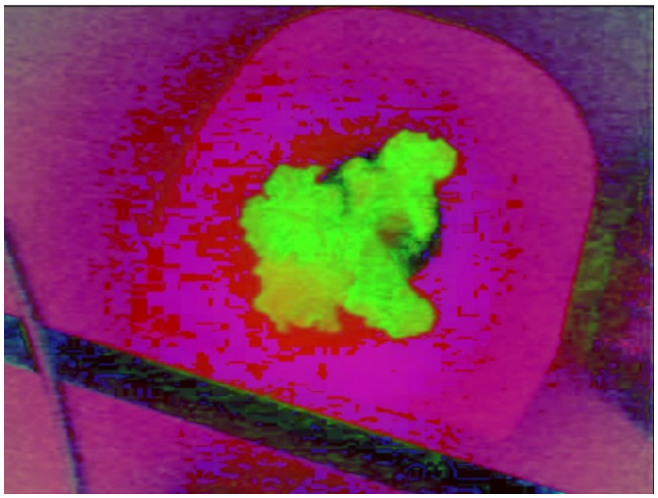


Figura 2.25: Imagen en HSV

4. Luego se procedió a aplicar un filtro que traduce los verdes hallados experimentalmente de la imagen a blancos y todos los demás colores a negro.

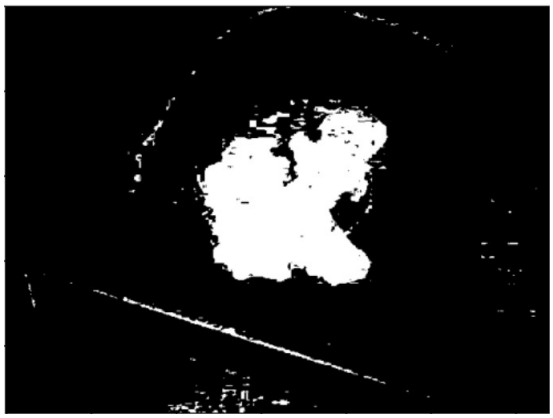


Figura 2.26: Imagen resultante del filtro

5. Se realizó una adición de la figura 2.26 con la imagen 2.24 obteniendo fondo negro con la lechuga en gris, como se puede visualizar en la figura 2.27

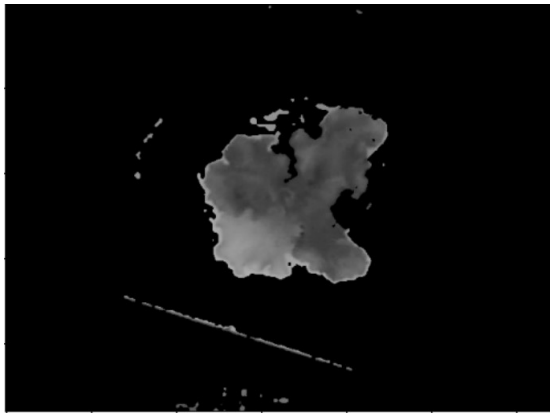


Figura 2.27: Adición de imágenes

6. En este paso se utiliza la función Canny de OpenCv la cual identifica borde y convierte en negro los píxeles que no son bordes. Los resultado están disponibles en la imagen 2.28.

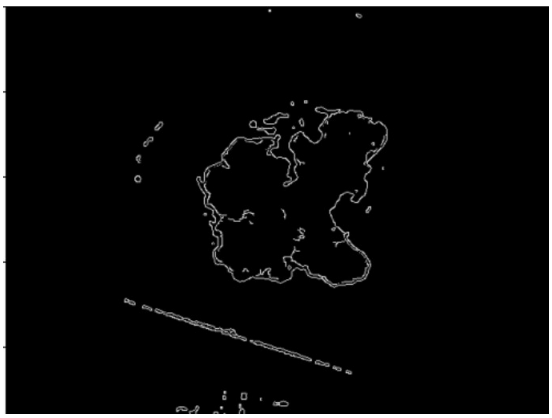


Figura 2.28: Identificando bordes

7. El siguiente paso es engrosar los bordes para que identificar el objeto sea mas sencillo. El resultado s puede visualizar en la imagen 2.29

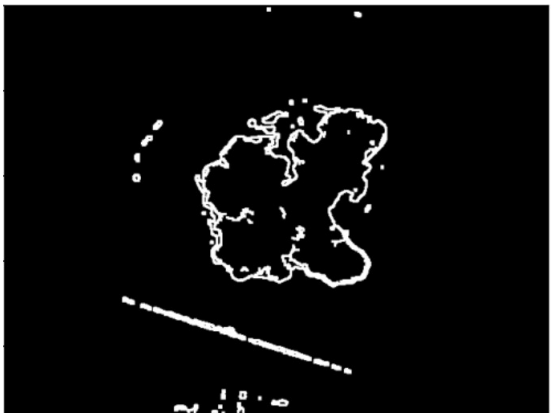


Figura 2.29: Engrosando Bordes

Finalmente en la figura a 2.30 se pueden observar los resultados de la identificación y su posterior medición. Todas la mediciones son realizadas sobre las imágenes tratadas y visualizadas sobre la lechuga color. Los valores son almacenados en variables para su posterior comunicación. Todo el código esta disponible en el repositorio. Si bien se podrían haber utilizado modelos propietarios de Machine Learning para realizar estos estudios. OpenCv ofrece un conjunto de librerías y funciones prediseñandas que ofrecen la solución deseada.

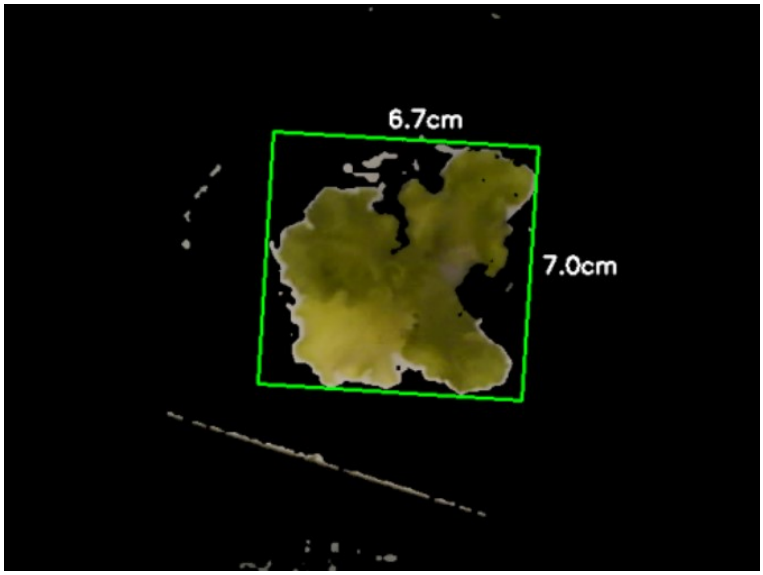


Figura 2.30: Midiendo la lechuga

2.7. Ambiente real

A continuación se podrán visualizar imágenes que corresponden con la construcción real del ambiente VLG.

Figura 2.31: Toma superior

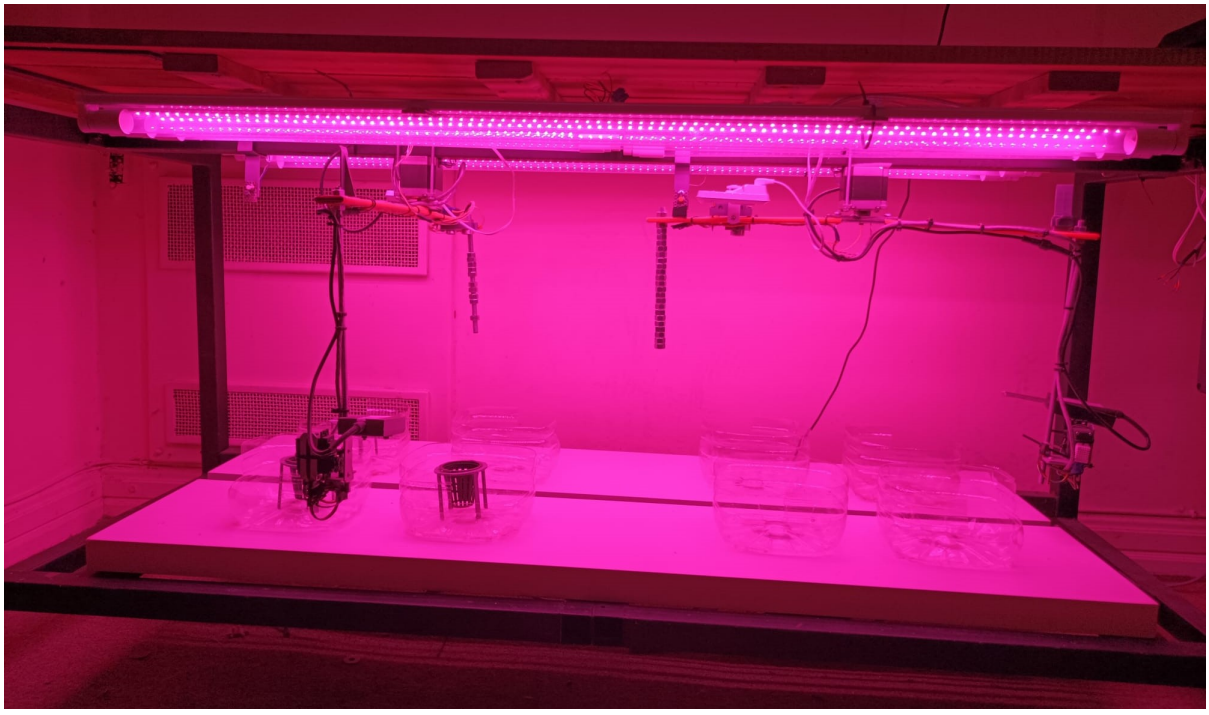


Figura 2.32: Toma frontal

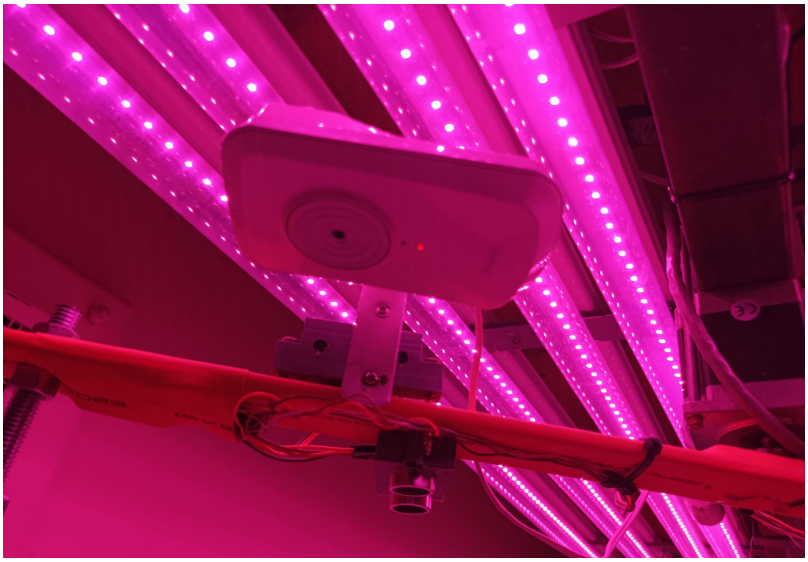


Figura 2.33: Cámara + Sensor de distancia (Toma frontal)

Figura 2.34: Cámara + Sensor de distancia (Toma Posterior)

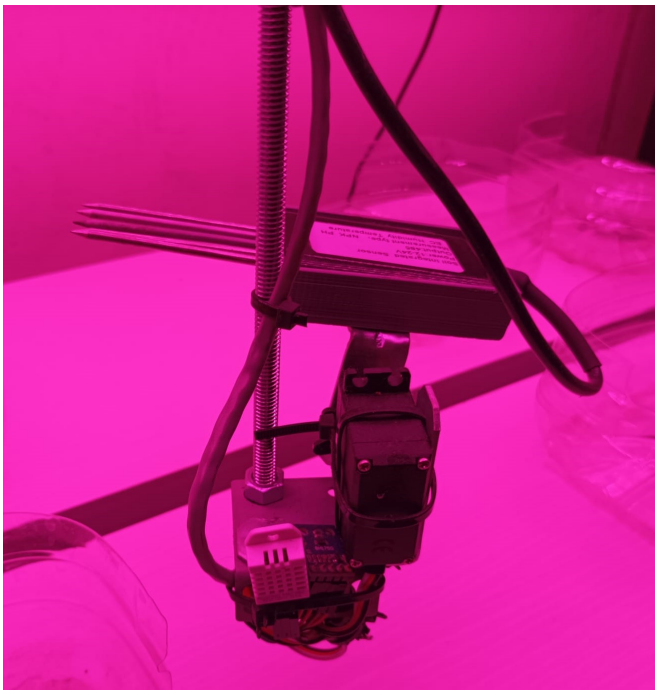


Figura 2.35: Sensores: NPK, temperatura, humedad y luminosidad (Lateral)

Figura 2.36: Sensores: NPK, temperatura, humedad y luminosidad (Superior)

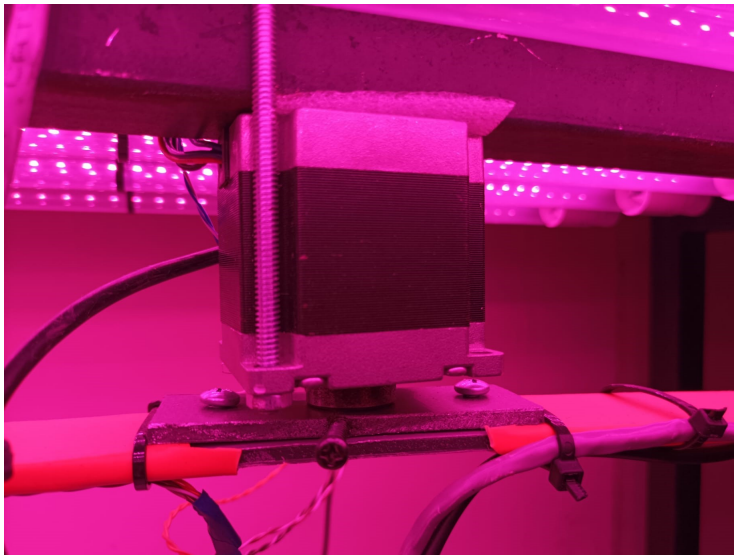


Figura 2.37: Motor Paso a paso + Rodamiento de giro

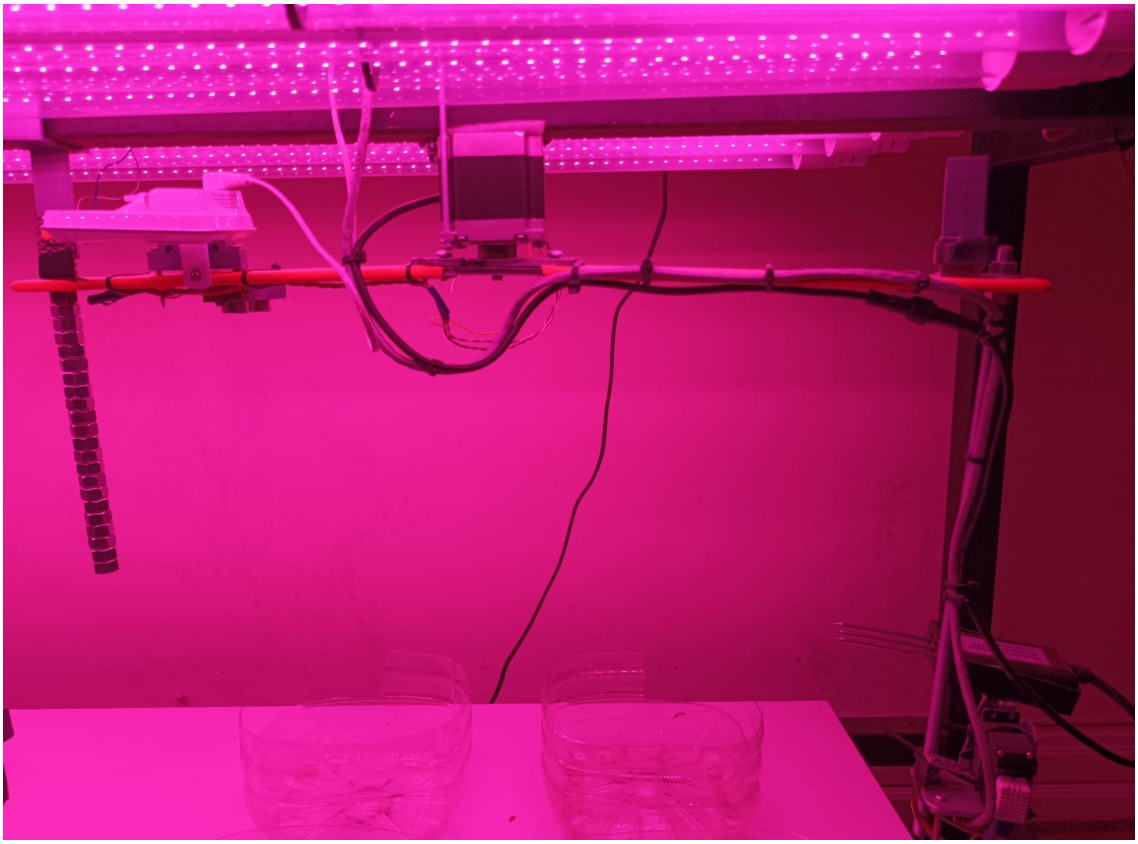


Figura 2.38: Hélice completa

Figura 2.39: Botón de parada

Figura 2.40: Soporte de Lechuga

Figura 2.41: Soporte + Recipiente de agua

Capítulo 3

Agentes de Inteligencia Artificial

3.1. Introducción

Un agente inteligente es una entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en su entorno de manera racional, es decir, de manera correcta y tendiendo a maximizar un resultado esperado. Es capaz de percibir su contexto con la ayuda de sensores y actuar en ese medio utilizando actuadores (elementos que reaccionan a un estímulo realizando una acción) [16].

Durante todo este capítulo para entrenar los agentes se utilizó en su mayoría Deep Q-Learning. Si bien se realizaron pruebas con PPO (Proximal Policy Optimization) [6] y A2C (Advantage Actor Critic) [11] fueron puntuales.

A continuación se describirá el funcionamiento de Deep Q-Learning. Varias de las palabras claves para el entrenamiento de un agente con Deep Q-learning son: acción, recompensa, ambiente, step y episodio. Básicamente un agente ejecuta una cierta cantidad de steps sobre el ambiente los cuales son organizados en episodios. En general el episodio puede ser de largo fijo o variable dependiendo de lo deseado. En caso de ser variable debe tener una condición de parada (la cual lleva a un estado final). Cada step parte de un estado inicial (s) y ejecuta una acción (a) dentro del dominio de la mismas, la cual tiene repercusión en el ambiente. Dicha repercusión es observada y genera un nuevo estado (s'). En resumen el agente parte de un estado s, ejecuta una acción a y llega a un estado s' y para cuantificar dicho proceso le asigna una recompensa (r) a la transición. Digamos que conocemos la recompensa esperada de cada acción en cada paso. Nuestro agente sabrá exactamente qué acción realizar. Realizará la secuencia de acciones que eventualmente generará la máxima recompensa total [17].

Esta recompensa total también se denomina valor Q y se formaliza como:

$$Q(s, a) = r(s, a) + \gamma \cdot max_a \, Q(s', a)$$

La ecuación anterior establece que el valor Q producido por estar en el estado s y realizar la acción a es la recompensa inmediata r(s,a) más el valor Q más alto posible del siguiente estado s'. Gamma aquí es el factor de descuento que controla

la contribución de las recompensas en el futuro [17]. En consecuencia la función de valor Q dependerá de los futuros estados y sus recompensas como se describe a continuación.

$$Q(s,a) \Rightarrow \gamma \cdot Q(s',a) + \gamma^2 \cdot Q(s'',a) + \gamma^3 \cdot Q(s''',a) + ... + \gamma^n \cdot Q(s^n,a)$$

En Deep Q-learning, se usa una red neuronal para aproximar la función de valor Q. El estado se proporciona como entrada y el valor Q de todas las acciones posibles se genera como salida.

3.2. Vista Jerárquica

Dado que este proyecto cuenta con ambientes virtuales y físicos, se decidió jerarquizar la forma en que se realiza el aprendizaje, con el fin de poder acelerar los procesos de entrenamiento y reducir la dificultad de los agentes y sus estados. Los agentes se clasifican en "Agentes de alto nivel" y "Agentes de bajo nivel". Los agentes de alto nivel involucran acciones abstractas por ejemplo "Fijar temperatura o luminosidad en un valor fijo alcanzable", mientras que las acciones de bajo nivel son aquellas que se deben realizar en el ambiente virtual o físico para alcanzar dicho valor, estas podrían ser "Calentar", "Enfriar", "Ventilar", "Subir", "Bajar", todo dependerá del estado del ambiente. A continuación en la sección 3.3 se describirán todos los ambientes virtuales Gym, sus acciones y forma del estado.

3.3. Agentes Inteligentes

En esta sección se describirá la arquitectura, diseño, implementación y prueba de 5 agentes inteligentes.

Arquitectura básica de todos los ambientes:

Con el fin de poder interactuar con el ambiente real, se creó un ambiente virtual en OpenAI [18] el cual encapsula el conocimiento de las acciones; y sus efectos, tiene acceso a la recolección de datos para conformar un nuevo estado del mismo y posee predefinido el sistema de recompensas acorde a cada estado.

Cada ambiente se encapsuló en una clase llamada GreenLabEnv. Dicha clase en todas sus versiones posee todo este conocimiento empírico del ambiente real. La misma tiene 4 métodos:

- **_init_**: es donde se inicializa el ambiente, se define el conjunto de acciones, el estado inicial y el largo del step.
- **step**: es el que recibe una acción y dado el estado actual ejecuta dicha acción en el ambiente. Como resultado obtiene un estado nuevo, recompensa y si alcanzo un estado final.

- **reset**: vuelve a posicionar al ambiente en un estado inicial. En este caso sólo toma la temperatura actual del ambiente.
- **render**: el cual no es utilizado en este caso.

Arquitectura básica de todas las redes de los agentes

Luego de probar otras arquitecturas se decidió utilizar la descripta a continuación en la figura 3.1:

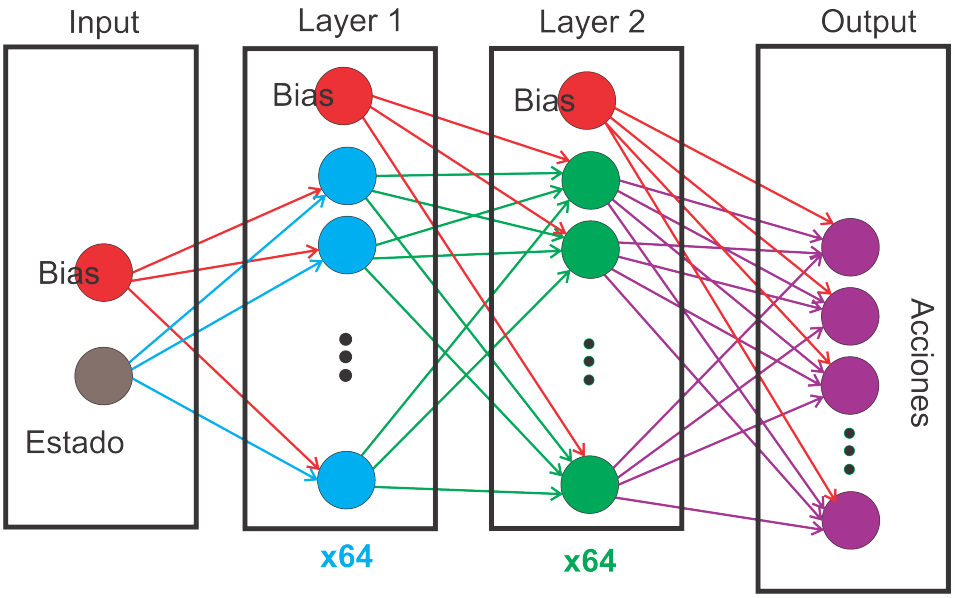


Figura 3.1: Modelo

Como se puede visualizar tiene como entrada el estado del ambiente. Es importante recordar que cada uno de los 5 agentes que se describirán en esta sección tiene estados distintos. Luego la capa de entrada esta conectada a la Capa 1(primer capa oculta) la cual esta compuesta por 64 neuronas(por lo cual tendrá 64 parámetros). Posterior a esta dicha capa esta conectada a la Capa 2 con igual dimensión que la primera. Ambas capas ocultas tienen como función de activación una ReLu. La función de activación ReLu aplica una transformación no lineal muy simple en la cual activa la neurona solo si la entrada está por encima de cero. Por último la capa de salida la cual esta determinada por la cantidad de acciones de cada agente, las cuales varían en los 5 agentes.

3.3.1. Agente 1

En esta subsección se describirá el agente inteligente entrenado previamente en la tesis de ingeniería [2]. Dado que se requería para la tesis de maestría y parte de las librerías que se utilizaron previamente dejaron de funcionar por actualizaciones de Keras [19], se reimplemento con PyTorch [20]. El objetivo de este agente es aprender a calibrar automáticamente la temperatura optimizándola por consumo,

en un ambiente que tiene la capacidad de censar temperatura y consumo además de aumentar y reducir la temperatura del mismo.

3.3.1.1 Descripción del estado del ambiente

El estado en esta versión del ambiente es únicamente representado por la temperatura instantánea del ambiente. Esto es como primer paso para comenzar a generar agentes que lo extiendan y permitan incorporar más variables. Se consideró que la temperatura era uno de los factores cruciales dado que optimizar el consumo es crucial para que este tipo de modelos sea rentable.

3.3.1.2 Modelado del ambiente real

Con el objetivo de reducir los tiempo de entrenamiento, se decidió modelar cuatro funciones representativas del comportamiento del ambiente. Estas funciones pretenden imitar el comportamiento de ambiente frente a la realización de una acción. Es decir al ejecutar una acción se produce una variación en la temperatura, estas variaciones puede ser forzadas o alcanzadas por equilibrio térmico con el ambiente. A continuación se describen todos los casos estudiados.

- **Calentamiento Forzado (Acción 1):** Se da cuando se quiere incrementar la temperatura con la acción de los caloventiladores.
- **Enfriamiento Forzado (Acción 0):** Se da cuando se quiere disminuir la temperatura por acción de la refrigeración.
- **Calentamiento Ventilado (Acción 2):** Se da cuando la temperatura interior del ambiente es inferior a la temperatura exterior y por equilibrio térmico la temperatura interior se ajusta.
- **Enfriamiento Ventilado (Acción 2):** Se da cuando la temperatura interior del ambiente es superior a la temperatura exterior y por equilibrio térmico la temperatura interior se ajusta.

Para modelar estas cuatro funciones se midió experimentalmente la variación que sufre el ambiente al ejecutar una acción. Por ejemplo para el calentamiento forzado, una vez que el ambiente estaba en equilibrio térmico con el exterior, se ejecuto la acción 1 (enciendo los caloventiladores) y se hicieron mediciones continuas cada 2 minutos simulando el tiempo por step. Se calcularon los incrementos entre estado actual y posterior a la acción. De esta forma para cada temperatura ejecutando la misma acción, se pudo modelar una función la cual recibe como parámetro la temperatura actual y retorna el incremento aproximado. Con lo puntos obtenidos en forma discreta se generó un función continua utilizando el método de mínimos cuadrados. De igual forma se realizó con las demás acciones obteniendo funciones acordes al comportamiento del ambiente. Es importante hacer notar que a diferentes temperaturas exteriores dichas funciones podrían variar. En este caso se asumió

constante a 29°C.

A continuación se puede visualizar los gráficos y las expresiones analíticas de las funciones.

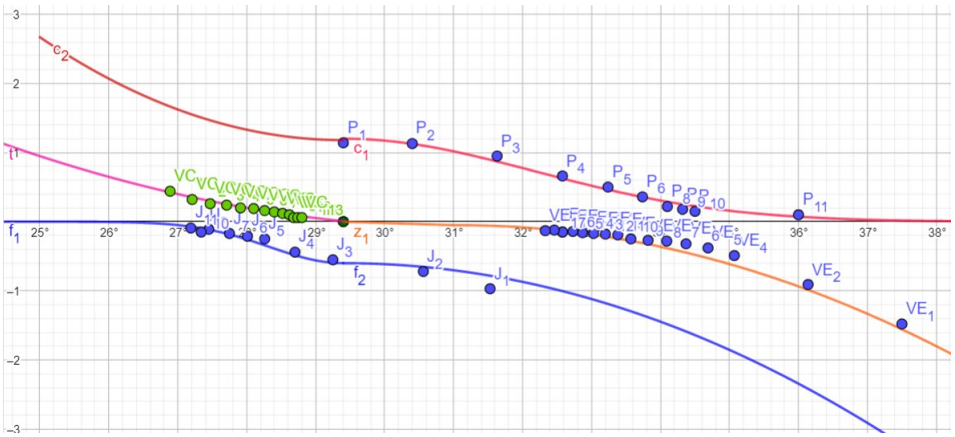


Figura 3.2: Diferentes gráficos

La curva roja bosquejada en la figura 3.2 representa la función de calentamiento forzado, la curva azul la de enfriamiento forzado, la rosada curva de calentamiento ventilado y por último la naranja la de enfriamiento ventilado.

3.3.1.3 Descripción de la acciones

Se resolvió encapsular el comportamiento en tres acciones posibles:

- **Acción 0** (Enfriar): Una vez ejecutada en un step, el ambiente virtual se comunica con el ambiente real y enciende la refrigeración durante el tiempo de duración del step.
- **Acción 1** (Calentar): Una vez ejecutada en un step, el ambiente virtual se comunica con el ambiente real y enciende los caloventiladores durante el tiempo de duración del step.
- **Acción 2** (Neutral): Una vez ejecutada en un step, el ambiente virtual se comunica con el ambiente real deshabilitando todos los dispositivo de control de temperatura y ventilando el ambiente durante el step. Esta acción permite equiparar la temperatura exterior con la interior con bajo consumo de energía.

3.3.1.4 Función de recompensa

Dadas las características y restricciones del ambiente, se definió un intervalo máximo en el cual puede operar. Claramente dependiendo de la temperatura exterior(no se logró suficiente aislación térmica para alcanzar dicha independencia). Por otro lado

se definió un intervalo óptimo de pertenencia para el confort de las plantas. Este último esta muy vinculado a la función de recompensa. Ya que por fuera de él, las recompensas obtenidas serán negativas mientras que por dentro son positivas con determinado formato.

El intervalo de confort para las plantas fue definido entre 27°C y 32°C con una temperatura objetivo en el promedio del mismo. Por lo cual en 29,5°C es donde debe alcanzar el máximo de recompensa. En consecuencia se uso dicho valor como media para la función gaussiana, además de ajustar la desviación de la misma con el fin de que las recompensas en los extremos interiores del intervalo fueran notoriamente más bajas. Luego de tener como base dicha función se optimizó la amplitud de la misma con dos parámetros:

- **Consumo**: se definió un coeficiente de consumo(CAPC) el cual reduce la recompensa cuanto más grande es el consumo. Este se describe más adelante.
- **Tiempo de convergencia al promedio**: Se lleva registro de la temperatura antes y después de ejecutar una acción. Realizando la diferencia de ambas y asumiendo un escenario discreto se obtiene la velocidad media de la variación de los incrementos. Conociendo el tiempo por *step* y la cantidad de *steps* necesarios determinados por la velocidad de convergencia, se calcula el tiempo de convergencia. Este tiempo es un indicador muy importante para obtener un balance entre consumo del ambiente y confort.

  El mejor escenario se da cuando la temperatura interior se ajusta por la exterior y el consumo es el más bajo posible. Pero si no se calculara este tiempo el sistema podría priorizar estar en el intervalo sin converger al promedio. Por esta razón se definió 30 minutos como punto de quiebre para aumentar o reducir el *reward*. Para tiempo menores a 30 minutos incrementa el *reward*, mientras que para tiempo mayores a 30 minutos disminuye. Para poder comprender el comportamiento en detalle de esta función se desarrollo un archivo en Geogebra [21] el cual posee varios deslizadores que simulan el comportamiento del sistema.

A continuación $f(x)$ describe la función de *reward* para el intervalo de valores positivos.

CVC (**C**oeficiente de **V**elocidad de **C**onvergencia): encapsula el cálculo de la velocidad de convergencia y tiempo de convergencia al promedio del intervalo. A mayor tiempo de convergencia menor *REWARD*, mientras que a menor tiempo mayor *REWARD*.

CAPC (**C**oeficiente de **A**juste **P**or **C**onsumo): encapsula el conocimiento de la variación del *reward* de acuerdo al consumo. A mayor consumo menor *REWARD*.

tmax y **tmin** determinan al intervalo de confort del ambiente y su promedio el objetivo (**prom**). Mientras que **hCampana** calcula la imagen en los extremos del

intervalo, con el fin de desplazar a la función hacia abajo, restando su valor y así logrando una función continua con las que están por fuera del intervalo.

$$f(x) = \ CVC \cdot CAPC \cdot e^{-\frac{(x-prom)^2}{3}} - hCampana, \quad tmin \leq x \leq tmax$$

Las funciones por fuera del intervalo son lineales y sus coeficientes principales se ajustan en tiempo real por el consumo, con el fin de mejorar el reward a menor consumo.

$$p(x) = \ \frac{-Consumo}{100}(x - tmax), \quad x \geq tmax$$

$$g(x) = \ \frac{Consumo}{100}(x - tmin), \quad x \leq tmin$$

A continuación se ejemplifica la función de recompensa completa para valores específicos de temperatura, consumo y tiempo de convergencia al promedio, que se utilizó para el entrenamiento.

$$REWARD(x) = \begin{cases} \frac{59}{100}(x - 27) & si \quad x \leq 27 \\ 2{,}0854 \cdot 4{,}42 \cdot e^{-\frac{(x-29{,}5)^2}{3}} - 1{,}1477 & si \quad 27 < x < 32 \\ \frac{-59}{100}(x - 32) & si \quad x \geq 32 \end{cases}$$

Y por último un bosquejo de dicha función.

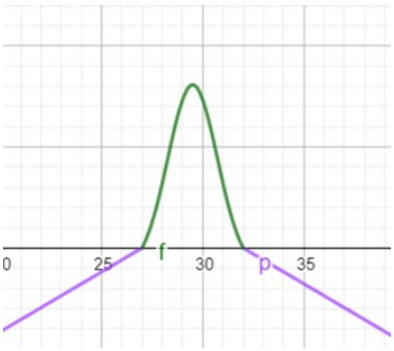


Figura 3.3: Gráfico completo de la función de recompensa.

3.3.1.5 Descripción del step

A continuación en la imagen 3.5 se puede visualizar el proceso de la ejecución de un step del agente. Como se puede observar, el primero paso es que el ambiente este inicializado en un estado correcto, en segundo lugar el agente ejecuta una acción sobre el ambiente; luego el mismo se encarga de decodificar dicha acción, ejecutarla y calcular la recompensa con la función que se describió en la sección 3.3.2. Por último el ambiente retorna la recompensa la cual es almacenada, con el objetivo de poder bosquejar posteriormente los resultados.

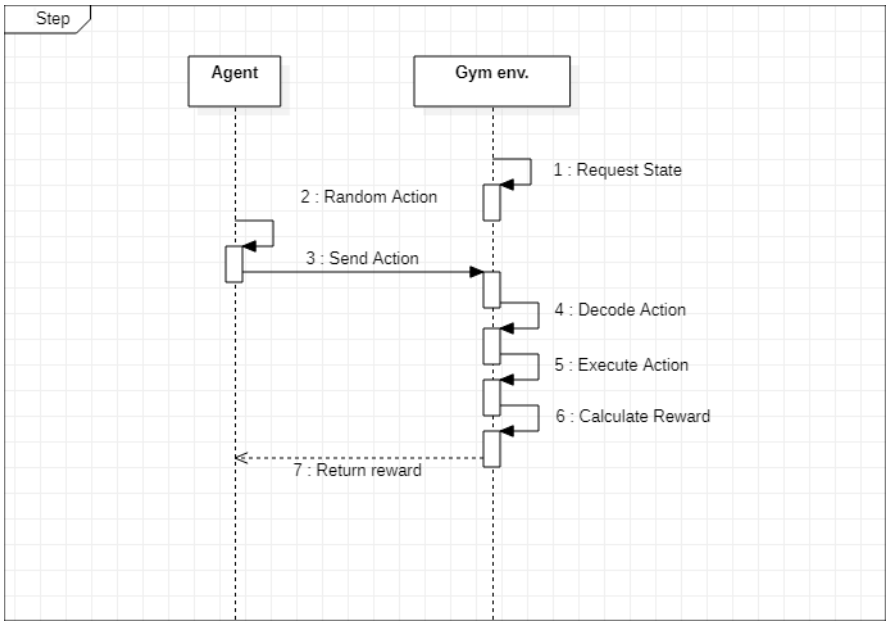


Figura 3.4: Step del agente

3.3.1.6 Descripción de un episodio

Un episodio es una agrupación de pasos, cada uno de estos pasos ejecuta una acción (las cuales fueron descriptas anteriormente) y el objetivo principal es llegar de un estado inicial a uno final. En este caso el estado inicial es variable y corresponde con la temperatura inicial del ambiente. El estado final corresponde con la temperatura fija de 29.4°C. Para este agente el largo de un episodio es fijo y se corresponde a 60 *steps*. Dado que cada *step* en el ambiente real tiene una duración de 2 minutos, la ejecución de un episodio culminaría luego de 120 minutos, lo cual se reduce a segundos con el ambiente simulado.

3.3.1.7 Prueba del agente entrenado con DQN

Según Christopher J. C. H. Watkins [22], Q-learning es una forma sencilla para que los agentes aprendan a actuar de manera óptima en dominios markovianos

controlados. Equivale a un método incremental para la programación dinámica que impone demandas computacionales limitadas.
En este caso para la ejecución del algoritmo se fijaron los siguientes parámetros, los cuales se encontraron experimentalmente:

1. Learning rate: es un parámetro de ajuste en un algoritmo de optimización que determina el tamaño del paso en cada iteración mientras se mueve hacia un mínimo de una función de pérdida. En este caso fue fijado en 0,001.

2. Gamma: el parámetro gamma define cuanto influye un solo ejemplo de entrenamiento, donde los valores bajos significan 'poco' y los valores altos significan 'mucho'. Para este caso particular se fijo en 0,999.

3. TIME_STEPS: es la cantidad de steps con la que se entrenó al algoritmo. En este caso se fijo en 288000.

Para entrenar este agente se utilizó una librería llamada Stable Baselines3 (SB3)[23]. Esta librería es un conjunto de implementaciones confiables de algoritmos de aprendizaje por refuerzo en PyTorch[20].

Figura 3.5: Step del agente

A continuación en la tabla 3.1, se puede visualizar el proceso de selección de acciones del agente partiendo de una temperatura de 38°C (estado del ambiente).
En dicha tabla se observa, el número del *step*, la acción que seleccionó el agente, la temperatura anterior a la ejecución de la acción, el incremento o decremento de temperatura que se produce por la ejecución de la acción, la nueva temperatura y la recompensa asociada a cada *step*. Analizando los valores de acciones seleccionados por el agente, 0 hace referencia a la acción 0, la cual corresponde con el enfriamiento forzado y la acción 2 que en este caso corresponde con el enfriamiento ventilado. Es muy importante que el agente ejecute la acción 2, dado que la temperatura exterior al ambiente es inferior a la del interior. Es importante recordar que la temperatura objetivo fija en este caso es de 29.5ºC y que si no la alcanza absolutamente es porque el agente prioriza reducir el consumo ejecutando acciones de ventilación. En este

caso particular el 80 % de las veces elige optimizar el consumo (Acción 2) y prioriza enfriar (Acción 0) reiteradamente al inicio, cuando la velocidad de convergencia al estado del ambiente(temperatura) es muy baja.

Step	Acción	T. Anterior	Incremento	T. Actual	Recompensa
1	0	38	-3.5584	34.4416	-5.37152
2	0	34.4416	-1.6167	32.8248	-1.81475
3	0	32.8248	-1.0691	31.7556	0.230033
4	0	31.7556	-0.8219	30.9337	1.591802
5	0	30.9337	-0.6940	30.2396	3.944640
6	0	30.2396	-0.6281	29.6114	12.19082
7	0	29.6114	-0.6017	29.0096	5.355126
8	0	29.0096	-0.5615	28.4481	2.309814
9	2	28.4481	0.12827	28.5763	6.551416
10	2	28.5763	0.10935	28.6857	6.973351
11	2	28.6857	0.09388	28.7796	7.290686
12	2	28.7796	0.08108	28.8607	7.525308
13	2	28.8607	0.07039	28.9311	7.695169
14	2	28.9311	0.06137	28.9924	7.814536
15	2	28.9924	0.05372	29.0462	7.894531
16	0	29.0462	-0.5682	28.4779	2.407578
17	2	28.4779	0.12379	28.6017	6.653908
18	2	28.6017	0.10571	28.7075	7.050895
19	2	28.7075	0.09088	28.7983	7.348400
20	2	28.7983	0.07858	28.8769	7.567455
21	2	28.8769	0.06829	28.9452	7.725168
22	2	28.9452	0.05959	29.0048	7.835069
23	0	29.0048	-0.5606	28.4442	2.297458
24	2	28.4442	0.12886	28.5731	6.537993
25	2	28.5731	0.10983	28.6829	6.963175
26	2	28.6829	0.09427	28.7772	7.283096
27	2	28.7772	0.08141	28.8586	7.519751
28	2	28.8586	0.07066	28.9292	7.691199
29	2	28.9292	0.06160	28.9909	7.811803
30	2	28.9909	0.05391	29.0448	7.892763
31	0	29.0448	-0.5679	28.4768	2.403736
32	2	28.4768	0.12396	28.6008	6.650002
33	2	28.6008	0.10585	28.7066	7.047945
34	2	28.7066	0.09099	28.7976	7.346208
35	2	28.7976	0.07868	28.8763	7.565858
36	2	28.8763	0.06837	28.9447	7.724035
37	2	28.9447	0.05966	29.0043	7.834297
38	2	29.0043	0.05226	29.0566	7.907180
39	2	29.0566	0.04592	29.1025	7.950765

Step	Acción	T. Anterior	Incremento	T. Actual	Recompensa
40	2	29.1025	0.04047	29.1430	7.971186
41	2	29.1430	0.03575	29.1787	7.973059
42	2	29.1787	0.03165	29.2104	7.959808
43	2	29.2104	0.02807	29.2385	7.933942
44	2	29.2385	0.02494	29.2634	7.897246
45	2	29.2634	0.02220	29.2856	7.850950
46	2	29.2856	0.01978	29.3054	7.795839
47	2	29.3054	0.01765	29.3231	7.732348
48	2	29.3231	0.01576	29.3388	7.660630
49	2	29.3388	0.01409	29.3529	7.580607
50	2	29.3529	0.01260	29.3655	7.492011
51	2	29.3655	0.01128	29.3768	7.394416
52	2	29.3768	0.01011	29.3869	7.287259
53	2	29.3869	0.00906	29.3960	7.169859
54	2	29.3960	0.00813	29.4041	7.041437
55	2	29.4041	0	29.4041	2.091813
56	2	29.4041	0	29.4041	2.091813
57	0	29.4041	-0.6000	28.8041	3.820868
58	2	28.8041	0.07781	28.8819	7.580129
59	2	28.8819	0.06764	28.9496	7.734147
60	2	28.9496	0.05904	29.0086	7.841169

Tabla 3.1: Tabla del episodio

En la figura 3.6, se puede observar como varia la temperatura a lo largo de la ejecución de un episodio. Es importante recordar que si bien la ejecución de este episodio toma tan solo segundos, ejecutado sobre el ambiente real demandaría 2 horas de entrenamiento.

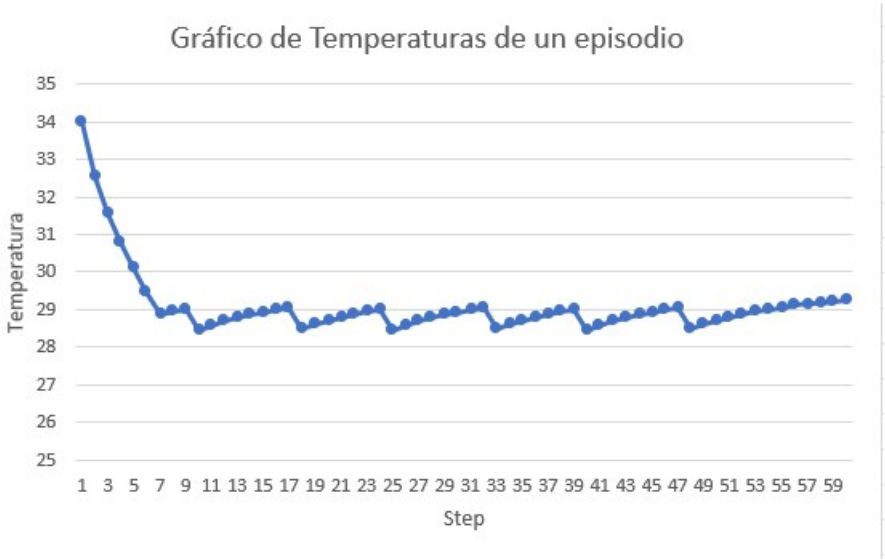


Figura 3.6: Curva de enfriamiento

A continuación en la tabla 3.2 se puede visualizar el proceso de selección de acciones del agente partiendo de una temperatura de 19°C (Estado del ambiente).

La distribución de los datos es exactamente igual que en la tabla 3.1. Analizando los valores de acciones seleccionados por el agente, 1 hace referencia a la acción 1, la cual corresponde con el calentamiento forzado y la acción 2 que en este caso corresponde con el calentamiento ventilado. Es muy importante que el agente ejecute la acción 2 dado que la temperatura exterior al ambiente es superior a la del interior. Es importante recordar que la temperatura objetivo fija en este caso es de 29.5ºC y que si no la alcanza absolutamente es porque el agente prioriza reducir el consumo ejecutando acciones de ventilación. En este caso particular aproximadamente el 80 % de las veces elige optimizar el consumo y prioriza calentar al inicio. Es importante mencionar que el ambiente logra aumentar más rápido la temperatura de lo que logra reducirla. El 80 % indica que 8 de cada 10 acciones que ejecuta el agente son de ventilación las cuales reducen el consumo. Ese mismo valor se obtiene al compararlo contra un agente que solo cuente con dos acciones, calentar o enfriar, en las cuales el consumo siempre es alto y no se ejecutan acciones que busquen utilizar la temperatura exterior como acción de control.

Step	Acción	T. Anterior	Incremento	T. Actual	Recompensa
1	1	19	9.45999	28.46	12.51972
2	2	28.46	0.12648	28.5864	6.592476
3	2	28.5864	0.10790	28.6943	7.004450
4	2	28.6943	0.09269	28.7870	7.313857
5	2	28.7870	0.08009	28.8671	7.542254
6	2	28.8671	0.06955	28.9367	7.707254
7	2	28.9367	0.06067	28.9974	7.822834
8	2	28.9974	0.05311	29.0505	7.899872
9	0	29.0505	-0.5689	28.4815	2.419480
10	2	28.4815	0.12326	28.6048	6.665945
11	2	28.6048	0.10527	28.7100	7.059984
12	2	28.7100	0.09052	28.8006	7.355150
13	2	28.8006	0.07828	28.8789	7.572371
14	2	28.8789	0.06804	28.9469	7.728653
15	2	28.9469	0.05938	29.0063	7.837439
16	0	29.0063	-0.5609	28.4454	2.301265
17	2	28.4454	0.12868	28.5741	6.542141
18	2	28.5741	0.10968	28.6838	6.966320
19	2	28.6838	0.09415	28.7779	7.285442
20	2	28.7779	0.08131	28.8592	7.521469
21	1	28.8592	1.16249	30.0217	10.91278
22	0	30.0217	-0.6154	29.4063	12.65944
23	0	29.4063	-0.6000	28.8063	3.832942
24	1	28.8063	1.16711	29.9734	11.71948

Step	Acción	T. Anterior	Incremento	T. Actual	Recompensa
25	0	29.9734	-0.6131	29.3602	11.31904
26	0	29.3602	-0.5995	28.7606	3.584495
27	2	28.7606	0.08363	28.8443	7.481038
28	2	28.8443	0.07253	28.9168	7.663456
29	2	28.9168	0.06318	28.9800	7.792608
30	2	28.9800	0.05526	29.0352	7.880241
31	0	29.0352	-0.5662	28.4690	2.377753
32	2	28.4690	0.12513	28.5941	6.623329
33	2	28.5941	0.10680	28.7009	7.027788
34	2	28.7009	0.09178	28.7927	7.331225
35	2	28.7927	0.07933	28.8720	7.554933
36	2	28.8720	0.06892	28.9409	7.716276
37	2	28.9409	0.06013	29.0011	7.829007
38	0	29.0011	-0.5598	28.4412	2.287889
39	2	28.4412	0.12931	28.5705	6.527524
40	2	28.5705	0.11020	28.6807	6.955236
41	2	28.6807	0.09458	28.7753	7.277172
42	2	28.7753	0.08166	28.8569	7.515412
43	2	28.8569	0.07087	28.9278	7.688096
44	2	28.9278	0.06178	28.9896	7.809664
45	2	28.9896	0.05407	29.0437	7.891377
46	0	29.0437	-0.5677	28.4759	2.400756
47	2	28.4759	0.12410	28.6000	6.646966
48	2	28.6000	0.10596	28.7060	7.045652
49	2	28.7060	0.09108	28.7971	7.344504
50	2	28.7971	0.07875	28.8758	7.564616
51	2	28.8758	0.06843	28.9442	7.723154
52	2	28.9442	0.05971	29.0040	7.833697
53	0	29.0040	-0.5604	28.4435	2.295273
54	2	28.4435	0.12896	28.5725	6.535608
55	2	28.5725	0.10991	28.6824	6.961367
56	2	28.6824	0.09434	28.7767	7.281747
57	2	28.7767	0.08147	28.8582	7.518763
58	2	28.8582	0.07071	28.9289	7.690493
59	2	28.9289	0.06164	28.9906	7.811316
60	2	28.9906	0.05395	29.0445	7.892448

Tabla 3.2: Tabla del episodio

En la figura 3.7 se puede observar como varía la temperatura a lo largo de la ejecución de un episodio. En un escenario en el cual la temperatura del ambiente era inferior a la objetivo.

Figura 3.7: Curva de calentamiento

3.3.1.8 Prueba del agente entrenado con A2C

Según Isaac Kargar [11] en el campo del aprendizaje por refuerzo, el algoritmo Advantage Actor Critic (A2C) combina dos tipos de algoritmos de aprendizaje por refuerzo (basado en políticas y basado en valores). Afirma que los agentes basados en políticas aprenden directamente una política (una distribución de probabilidad de acciones) asignando estados de entrada a acciones de salida. Los algoritmos basados en valores aprenden a seleccionar acciones en función del valor previsto del estado de entrada o la acción.

Segun Mike Wang [24] A2C consta de dos redes (el actor y el crítico) que trabajan juntas para resolver un problema en particular. En un nivel alto, la función Advantage calcula el error TD o el error de predicción del agente. La red de actores elige una acción en cada paso de tiempo y la red crítica evalúa la calidad o el valor Q de un estado de entrada dado. A medida que la red crítica aprende qué estados son mejores o peores, el actor usa esta información para enseñar al agente a buscar buenos estados y evitar malos estados. A continuación en la figura 3.8 se puede visualizar gráficamente el comportamiento del algoritmo.

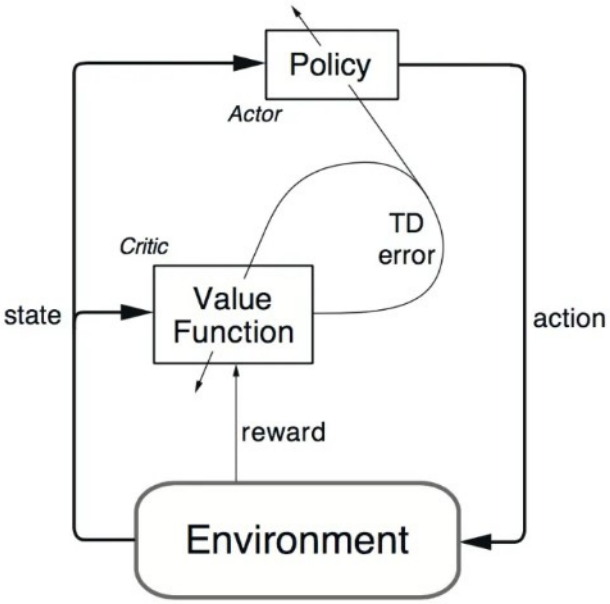


Figura 3.8: Descripción de A2C

Para el entrenamiento de este algoritmo se fijaron los parámetros en los mismos valores que en la sección 3.3.2.

En la imagen 3.9 a continuación se pueden observar los resultados del entrenamiento utilizando A2C.

Figura 3.9: Rewards con A2C

Sin necesidad de plotear ambos bosquejos (el experimento anterior y este) en los mismo pares de ejes coordenados, se puede observar que obtiene recompensas más bajas y en consecuencia acciones que optimizan menos el consumo o no son adecuadas. En este caso no se observará la elección de acciones como en en el entrenamiento de DQN.

3.3.1.9 Prueba del agente entrenado con PPO

El algoritmo de optimización de políticas proximales combina ideas de A2C (tener varios trabajadores) y TRPO (utiliza una región de confianza para mejorar el actor). La idea principal es que después de una actualización, la nueva política no debe estar muy alejada de la política anterior. Para eso, PPO usa el recorte para evitar una actualización demasiado grande.

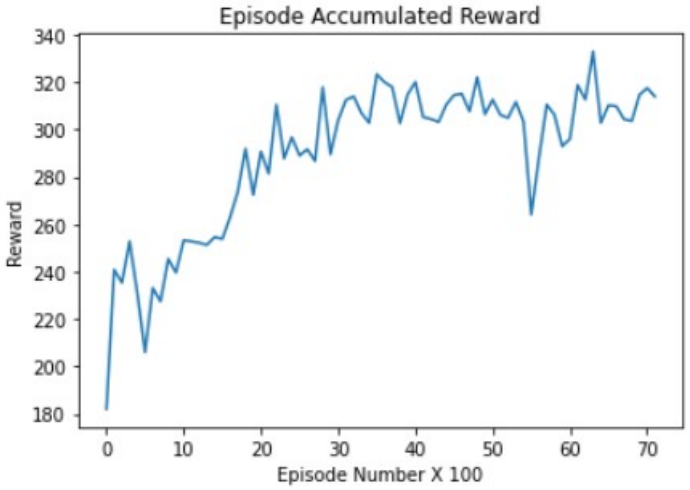


Figura 3.10: Rewards con PPO

3.3.1.10 Conclusiones de las pruebas

Analizando los resultados arrojados por los diferentes algoritmos, se puede concluir que Deep q-learning es el algoritmo que logra maximizar las recompensas, además de encontrar políticas óptimas de ejecución de acciones. Por dicha razón se continuó utilizando este algoritmo para el resto de los agentes inteligentes.

3.3.2. Agente 2

De aquí en adelante todos los agentes fueron desarrollados con el propósito de cumplir los objetivos de la tesis de Maestría. En esta subsección se describirá un agente inteligente, el cual extiende el Agente 1, brindándole la posibilidad de determinar la política óptima de ejecución de acciones, para alcanzar cualquier temperatura en el ambiente.

3.3.2.1 Descripción del estado del ambiente

Para lograr lo antes descripto, el estado no solo debe estar compuesto por la temperatura actual del ambiente sino también por la temperatura objetivo. Por lo cual el estado es un array con dos componentes, la temperatura instantánea y la objetivo. El modelado del ambiente y las acciones son exactamente las mismas que en el Agente 1.

3.3.2.2 Función de recompensa

La función de recompensa extiende a la mencionada en el Agente 1. Generalizando el valor a alcanzar y brindando un intervalo variable entorno a la temperatura objetivo. Este intervalo con centro en la temperatura objetivo tiene un radio de 2.5°C en el cual las recompensas comienzan a ser positivas. Por fuera de dicho intervalo serán negativas.
A continuación $f(x)$ describe la función de *reward* para el intervalo de valores positivos.

$$f(x) = \ CVC \cdot CAPC \cdot e^{-\frac{(x-tObj)^2}{3}} - hCampana, \quad tObj - 2{,}5 \leq x \leq tObj + 2{,}5$$

Las funciones por fuera del intervalo son lineales y sus coeficientes principales se ajustan en tiempo real por el consumo, con el fin de mejorar el reward a menor consumo.

$$p(x) = \ \frac{-Consumo}{100}(x - (tObj + 2{,}5)), \quad x \geq tObj + 2{,}5$$

$$g(x) = \frac{Consumo}{100}(x - (tObj - 2{,}5)), \quad x \leq tObj - 2{,}5$$

A continuación se ejemplifica la función de recompensa completa para valores específicos de temperatura, consumo y tiempo de convergencia al promedio.

$$R(x, tObj) = \begin{cases} \frac{59}{100}(x - (tObj - 2{,}5)) & si \quad x \leq (tObj - 2{,}5) \\ 2{,}0854 \cdot 4{,}42 \cdot e^{-\frac{(x-tObj)^2}{3}} - 1{,}1477 & si \quad tObj - 2{,}5 < x < tObj + 2{,}5 \\ \frac{-59}{100}(x - tObj + 2{,}5) & si \quad x \geq tObj + 2{,}5 \end{cases}$$

Esta función instanciada con $tObj = 29{,}5$ es idéntica a la función de recompensa del agente anterior.

3.3.2.3 Descripción del agente

Este agente se comporta de la misma forma que el Agente 1. Con la variación del estado y la función de recompensa.

3.3.2.4 Prueba del agente entrenado con DQN

En este caso para la ejecución del algoritmo se fijaron los siguientes parámetros, los cuales se encontraron experimentalmente:

1. Leraning rate: 0,001.
2. Gamma: 0,999.
3. Time steps: 576000.

Para entrenar este agente se utilizó la misma librería. En la figura 3.11 se observan los resultados de la ejecución de los 576000 steps en sus respectivos episodios. Se ve como progresivamente las recompensas son mayores.
A continuación en la tabla 3.3 se puede visualizar el proceso de selección de acciones del agente partiendo de una temperatura de 30°C (estado del ambiente).
En dicha tabla se puede observar el número del *steps*, la acción que seleccionó el agente, la temperatura anterior a la ejecución de la acción, el incremento o decremento de temperatura que se produce por la ejecución de la acción, la temperatura objetivo (variable) y la recompensa asociada a cada *step*. Analizando los valores de acciones seleccionadas por el agente, 0 hace referencia a la acción 0, la cual corresponde con el enfriamiento forzado y la acción 2 que en este caso corresponde con el enfriamiento ventilado.

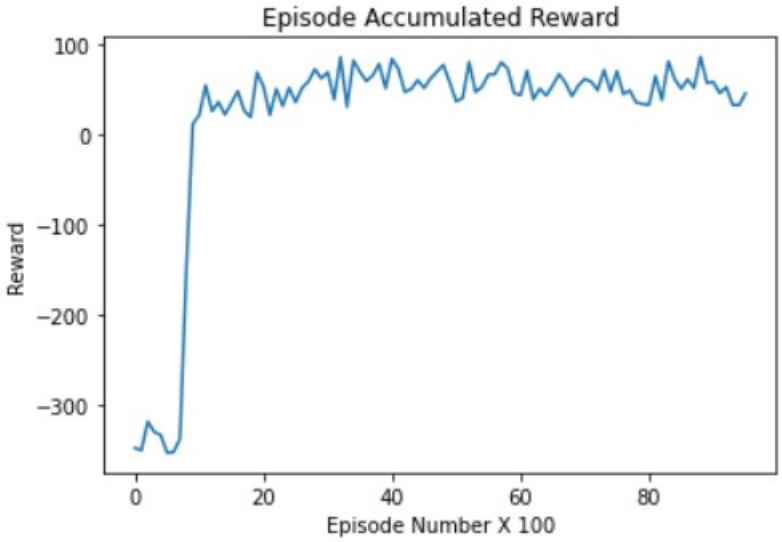


Figura 3.11: Resultados

En este caso particular las acciones son ejecutadas buscando una temperatura objetivo de 28°C, temperatura que varía en cada episodio.

Step	Acción	T. Actual	Incremento	T. Objetivo	Recompensa
1	0	30	-0.6144	28	5.564728
2	0	29.3856	-0.5999	28.0	3.565066
3	0	28.7856	-0.5091	28.0	3.199519
4	0	28.2764	-0.3465	28.0	2.875430
5	0	27.9298	-0.2344	28.0	2.360393
6	2	27.6954	0.25605	28.0	9.691340
7	0	27.9514	-0.2409	28.0	2.402977
8	0	27.7105	-0.1734	28.0	1.839253
9	0	27.5370	-0.1326	28.0	1.318311
10	0	27.4043	-0.1062	28.0	0.880173
11	0	27.2981	-0.0878	28.0	0.540430
12	0	27.2102	-0.0746	28.0	0.526282
13	2	27.1356	0.36967	28.0	9.131928
14	2	27.5053	0.29286	28.0	9.667535
15	2	27.7982	0.23692	28.0	9.627911
16	0	28.0351	-0.2669	28.0	2.553759
17	2	27.7681	0.24245	28.0	9.652017
18	0	28.0106	-0.2592	28.0	2.511906
19	2	27.7514	0.24556	28.0	9.663478
20	2	27.9969	0.20142	28.0	9.352680
21	0	28.1984	-0.3202	28.0	2.788254
22	2	27.8781	0.22240	28.0	9.541257
23	0	28.1005	-0.2879	28.0	2.656767
24	2	27.8126	0.23428	28.0	9.614712
25	0	28.0468	-0.2706	28.0	2.573213
26	2	27.7762	0.24097	28.0	9.646023

Step	Acción	T. Actual	Incremento	T. Objetivo	Recompensa
27	0	28.0171	-0.2612	28.0	2.523269
28	2	27.7559	0.24473	28.0	9.660551
29	0	28.0006	-0.2560	28.0	2.494271
30	2	27.7445	0.24684	28.0	9.667763
31	2	27.9914	0.20239	28.0	9.363139
32	0	28.1937	-0.3187	28.0	2.782616
33	2	27.8750	0.22295	28.0	9.545196
34	0	28.0980	-0.2871	28.0	2.653025
35	2	27.8108	0.23459	28.0	9.616329
36	0	28.0454	-0.2702	28.0	2.570931
37	2	27.7752	0.24114	28.0	9.646749
38	2	28.0164	0.19805	28.0	9.315191
39	0	28.2144	-0.3256	28.0	2.807410
40	2	27.8888	0.22049	28.0	9.527222
41	0	28.1093	-0.2907	28.0	2.669616
42	2	27.8185	0.23320	28.0	9.608990
43	0	28.0517	-0.2722	28.0	2.581079
44	2	27.7795	0.24036	28.0	9.643471
45	0	28.0198	-0.2621	28.0	2.527890
46	2	27.7577	0.24438	28.0	9.659322
47	0	28.0021	-0.2565	28.0	2.496930
48	2	27.7455	0.24665	28.0	9.667136
49	2	27.9922	0.20224	28.0	9.361582
50	0	28.1944	-0.3189	28.0	2.783464
51	2	27.8755	0.22287	28.0	9.544609
52	0	28.0983	-0.2872	28.0	2.653587
53	2	27.8111	0.23454	28.0	9.616088
54	0	28.0457	-0.2702	28.0	2.571274
55	2	27.7754	0.24111	28.0	9.646641
56	0	28.0165	-0.2610	28.0	2.522132
57	2	27.7554	0.24481	28.0	9.660850
58	0	28.0002	-0.2559	28.0	2.493617
59	2	27.7443	0.24689	28.0	9.667916
60	2	27.9911	0.20242	28.0	9.363521

Tabla 3.3: Tabla del episodio

En la figura 3.12 se puede observar como varia la temperatura a lo largo de la ejecución de un episodio. Es importante recordar que si bien la ejecución de este episodio toma tan solo segundos, ejecutado sobre el ambiente real demandaría 2 horas de entrenamiento.

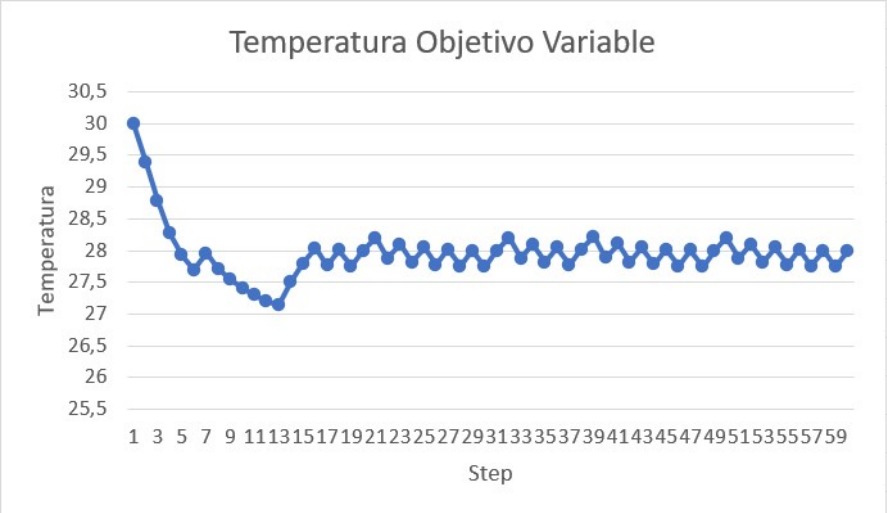


Figura 3.12: Curva de enfriamiento

3.3.3. Agente 3

Para poder entender este agente, primero se debe describir como es controlada la luminosidad del ambiente real. Dicho ambiente cuenta con dos plafones de luces de amplio espectro, las cuales son desplazadas por un motor de cadena que determina la altura a la que se ubicarán de las plantas. La luminosidad que reciben las plantas es inversamente proporcional a la distancia a la cual se encuentran de las mismas. En resumen cada altura se corresponde con un valor de luminosidad. El objetivo de este agente es aprender a regular la luminosidad del ambiente automáticamente. Para poder realizar mayor cantidad de pruebas en menos tiempo se modeló su comportamiento, lo cual reduce riesgo de roturas en el ambiente real. Si bien en esta caso sería sencillo optar por programar un autómata que la regulara, se optó por que todo el comportamiento del ambiente sea dirigido por políticas de RL.

3.3.3.1 Descripción del estado del ambiente

El estado del ambiente en este caso se corresponde con un *array* de dos posiciones compuesto por:

- La luminosidad actual
- La luminosidad objetivo

3.3.3.2 Modelado del ambiente real

En la figura 3.13 a continuación se puede visualizar la distancia máxima y mínima entre las luces y las lechugas. Si bien el movimiento del motor es continuo a lo largo de todo el recorrido y controlado por tiempo, se optó por discretizar las posiciones de las luces con el fin poder fijar una apreciación clara y reducir el error mecánico.

En este caso la distancia mínima de traslación de las luces es 1 cm. Eso genera 31 posiciones y cada una de ellas otorga una luminosidad constante. Cabe recordar que con el fin de verificar la correcta posición de las luces, se les integró un sensor de distancia.

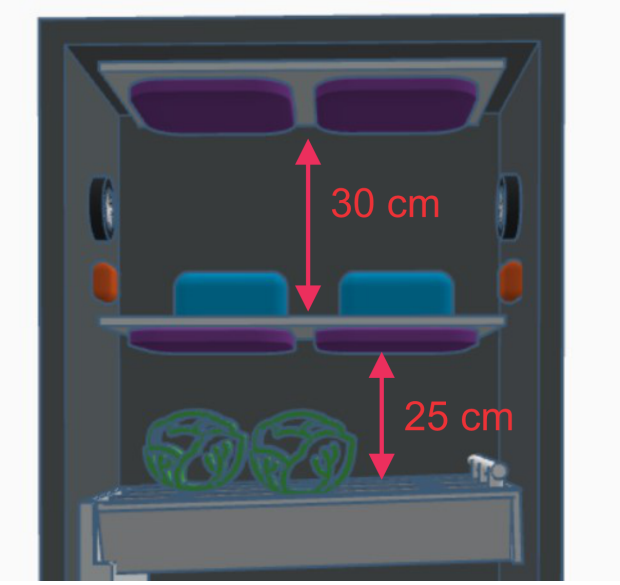


Figura 3.13: Luces del ambiente

Se generó una función continua capaz de indicar el valor de la luminosidad a cada distancia en centímetros del techo el ambiente. Para modelarla primero se registraron las luminosidades en 11 posiciones distintas, las cuales se pueden observar a continuación:

$\{(0, 395), (5, 442), (8, 474), (10, 498), (13, 529), (15, 563), (20, 619), (23, 658), (25, 693), (28, 750), (30, 780)\}$

La función resultante luego de aplicar el método de mínimos cuadrados para determinar los coeficientes de una función cuadrática fue:

$$L(x) = 396{,}9402 + 8{,}3911x + 0{,}1438x^2 \quad si \quad 0 \leq x \leq 30$$

Donde la variable x representa la distancia en centímetros de arriba hacia abajo.

En la figura 3.14 se pueden visualizar los puntos tomados experimentalmente y la función aproximada.

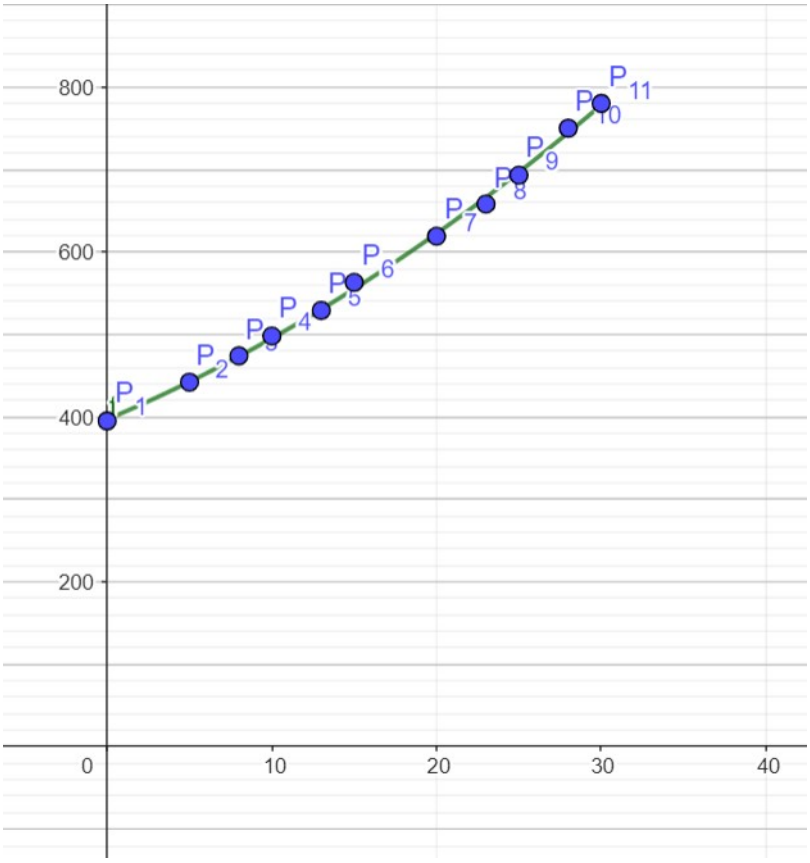


Figura 3.14: Gráfico de Luminosidades

3.3.3.3 Descripción de la acciones

Se resolvió encapsular el comportamiento en dos acciones posibles:

- **Acción 0** (Bajar): Una vez ejecutada en un *step*, las luces bajan $1\,cm$ incrementando la luminosidad.
- **Acción 1** (Subir): Una vez ejecutada en un *step*, las luces suben $1\,cm$ decrementando la luminosidad.

3.3.3.4 Función de recompensa

En este caso la función de recompensa es bastante sencilla. El objetivo es alcanzar el valor máximo cuando la luminosidad del ambiente alcance la objetivo, considerando la cantidad de *steps* que requiere para llegar a un estado final. A mayor cantidad de *steps* la recompensa es más baja. De esta forma se busca premiar a las políticas de acciones mas cortas.
A continuación se describe dicha función:

$$R(X, lumObj, steps) = -steps.|x - lumObj| + 200$$

donde la variable x corresponde a la luminosidad del ambiente. El valor 200 es simbólico y representa el valor máximo cuando se alcanza un estado final. lumObj es la luminosidad objetivo deseada y *steps* la cantidad de *steps* que se ejecutaron del episodio hasta ese instante.

3.3.3.5 Descripción de un episodio

A diferencia de los agentes anteriores, el largo de los episodios es variable, dado que una vez encontrada la luminosidad objetivo la misma no va a variar. El objetivo es determinar la menor cantidad de acciones a ejecutar para lograr que la luminosidad del ambiente sea la deseada. En este caso el estado inicial es variable y corresponde con un array de dos componentes, el cual posee la luminosidad instantánea y objetivo. Ambas luminosidades son inicializadas randomicamente. Un estado final es alcanzado cuando la luminosidad del ambiente y la objetivo coinciden o alcanza los 60 *steps* (Es importante mencionar que la luminosidad objetivo no varia a lo largo del episodio). La recompensa siempre será 0 con la excepción que el agente alcance un estado final. En ese caso se utiliza la función de recompensa descripta anteriormente.

3.3.3.6 Prueba del agente entrenado con DQN

Este agente se entreno con Q-learning y se fijaron los siguientes parámetros, los cuales se encontraron experimentalmente:

1. Leraning rate: 0,001.
2. Gamma: 0,999.
3. TIME_STEPS: 52500. Aumentando la cantidad de *steps* desmejora la política de ejecución de acciones.

A continuación en la figura 3.20 se puede observar la función de recompensas promedio. La misma describe las recompensas obtenidas a lo largo del entrenamiento. Es evidente que las recompensas aumentan a medida que aumenta la cantidad de episodios. Para verificar que el agente esta aprendiendo se procederá a visualizar el proceso de selección de acciones del agente, partiendo de diferentes luminosidades. En referencia a los datos visualizados a continuación, la primer componente representa la acción seleccionada, la segunda la luminosidad antes de realizar la acción, la tercera la luminosidad objetivo y por último la recompensa. En los datos visualizados se pueden observar 4 episodios en las tablas 3.4, 3.5, 3.6, 3.7.

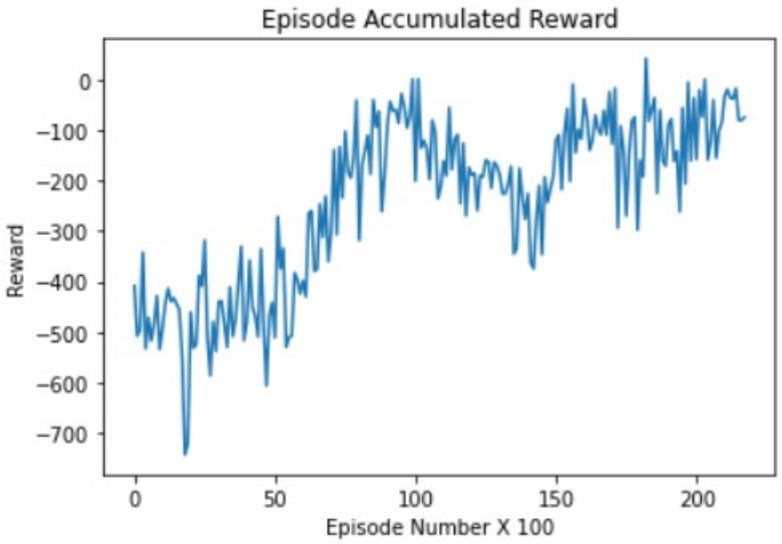


Figura 3.15: Recompesas del Agente

Episodio 1:
Este episodio luego de 3 *steps* logra alcanzar la luminosidad deseada.

Step	Acción	Luminosidad Actual	Luminosidad Objetivo	Recompensa
1	0	**473**	**452**	0
2	0	462	452	0
3	0	**452**	452	200.

Tabla 3.4: Tabla del episodio 1

Episodio 2:
Este episodio luego de 17 *steps* logra alcanzar la luminosidad deseada. Es importante destacar que en este caso realizar siempre la mejor acción en cada *step*.

Step	Acción	Luminosidad Actual	Luminosidad Objetivo	Recompensa
1	1	**405**	**581**	0
2	1	414	581	0
3	1	423	581	0
4	1	432	581	0
5	1	442	581	0
6	1	452	581	0
7	1	462	581	0
8	1	473	581	0
9	1	484	581	0
10	1	495	581	0
11	1	506	581	0
12	1	518	581	0
13	1	530	581	0
14	1	542	581	0
15	1	555	581	0
16	1	568	581	0
17	1	**581**	581	200.

Tabla 3.5: Tabla del episodio 2

Episodio 3:
Este episodio luego de 31 *steps* logra alcanzar la luminosidad deseada. En este caso al inicio no selecciona las mejores acciones pero a partir del *step* 13 se corrige y selecciona todas de forma óptima.

Step	Acción	Luminosidad Actual	Luminosidad Objetivo	Recompensa
1	1	**712**	**473**	0
2	1	728	473	0
3	0	712	473	0
4	1	728	473	0
5	0	712	473	0
6	1	728	473	0
7	0	712	473	0
8	1	728	473	0
9	0	712	473	0
10	1	728	473	0
11	0	712	473	0
12	1	728	473	0
13	0	712	473	0
14	0	696	473	0
15	0	681	473	0
16	0	666	473	0
17	0	651	473	0
18	0	636	473	0
19	0	622	473	0
20	0	608	473	0
21	0	594	473	0
22	0	581	473	0
23	0	568	473	0
24	0	555	473	0
25	0	542	473	0
26	0	530	473	0
27	0	518	473	0
28	0	506	473	0
29	0	495	473	0
30	0	484	473	0
31	0	**473**	473	200.

Tabla 3.6: Tabla del episodio 3

Episodio 4:
Este episodio luego de 12 *steps* logra alcanzar la luminosidad deseada. En este caso siempre selecciona la acción óptima.

Step	Acción	Luminosidad Actual	Luminosidad Objetivo	Recompensa
1	1	432	555	0
2	1	442	555	0
3	1	452	555	0
4	1	462	555	0
5	1	473	555	0
6	1	484	555	0
7	1	495	555	0
8	1	506	555	0
9	1	518	555	0
10	1	530	555	0
11	1	542	555	0
12	1	555	555	200.

Tabla 3.7: Tabla del episodio 4

En las figuras 3.16, 3.17, 3.18, 3.19 se puede observar como varia la luminosidad a lo largo de la ejecución de un episodio. En todos los casos visualizados la luminosidad del ambiente logra alcanzar la luminosidad previo a los 60 steps, valor máximo para el largo de un episodio.

Figura 3.16: Episodio 1

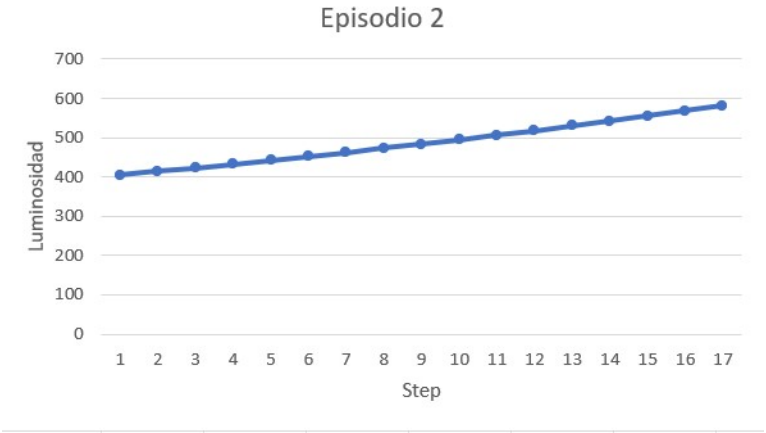


Figura 3.17: Episodio 2

Figura 3.18: Episodio 3

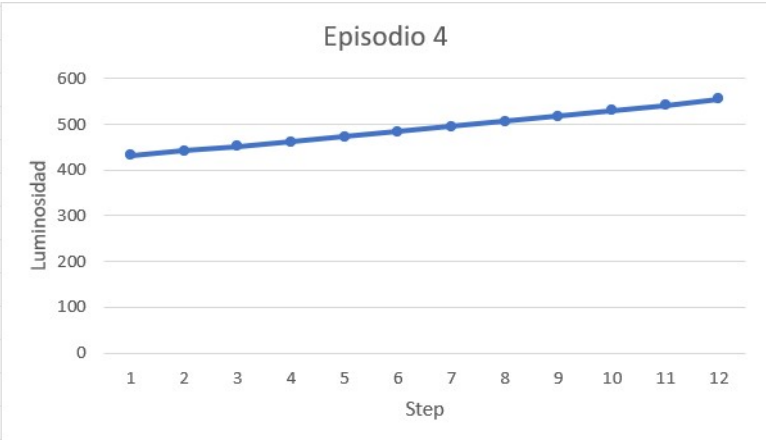


Figura 3.19: Episodio 4

3.3.4. Agente 4

Previo a esta subsección, todos los agentes anteriores son de bajo nivel, esto implica que sus objetivos son identificar las acciones necesarias(en el ambiente) para garantizar los valores deseados. La pregunta que surge es, ¿cuáles son los valores deseados?, o mejor dicho ,¿cuáles son los valores de temperatura y luminosidad que optimizan el crecimiento?. En consecuencia el objetivo de este agente es identificar cual es la temperatura óptima en cada etapa del crecimiento de la lechuga. Una vez determinada dicha temperatura, la misma se transforma en la temperatura objetivo para el Agente 2, la cual indica las acciones al ambiente para garantizar su estado óptimo.
Se experimentó en el ambiente real, pero el tiempo requerido para obtener suficiente información y entrenar un modelo se estimó era muy alto. Por dicha razón se optó por investigar de la existencia de modelos virtuales de la lechuga, los cuales pudiesen predecir el crecimiento en diferentes etapas de la misma, por la incidencia de la temperatura y la luminosidad.

Durval Dourado-Neto et. al [10] realizaron un estudio acerca de la acumulación de materia seca de las plantas en función de la radiación solar, la temperatura y la humedad relativa en ambientes de invernadero utilizando hidroponia. Propusieron modelos matemáticos para estimar la variación temporal de la acumulación de materia seca, caracterizando con funciones la incidencia de la temperatura y la radiación solar diaria. A partir de su articulo se extrajeron funciones de recompensas con las cuales se entrenaron al agente. En la subsección 3.3.4 se describirán dichas funciones.

3.3.4.1 Descripción del estado del ambiente

El estado del ambiente en este caso se corresponde con un array de dos posiciones compuesto por:

- La temperatura del ambiente.
- El día de evolución de la lechuga. Este dato es fundamental en los modelos propuestos por Durval Dourado-Neto et. al.

3.3.4.2 Descripción de la acciones

En este caso las acciones son de alto nivel, las mismas se corresponden con fijar una temperatura en un rango entre 10°C y 38°C, por lo cual hay 29 acciones. Ejemplo:

- **Acción 0**: fijar temperatura en 10°C.
- **Acción 1**: fijar temperatura en 11°C.
- **Acción 2**: fijar temperatura en 12°C.

  .
  .
  .

- **Acción 28**: fijar temperatura en 38°C.

3.3.4.3 Función de recompensa

Como se describió en la introducción de este agente, la función de recompensa fue extraída del paper de Durval Dourado-Neto et. al. Para poder comprenderla a continuación se definirán varios conceptos.

$$\hat{Tj} = \frac{T_{maxj} + T_{minj}}{2}$$

$\hat{Tj}$ refiere a la temperatura media en el día $j - esimo$. Mientras que T_{minj} y T_{maxj} refieren a la temperatura mínima y máxima en el día $j - esimo$.

$$DD_n = \sum_{j=1}^{n} (\hat{Tj} - T_{BI})$$

DD_n refiere al índice térmico o suma calórica (Gilmore & Rogers 1958), mientras que T_{BI} es la temperatura base más baja (TBI = 10°C).

$$RD_{DD_n} = \frac{\sum_{j=1}^{n} (\hat{Tj} - T_{BI})}{DD_{fmp}}$$

RD_{DD_n} refiere al desarrollo relativo de la lechuga, mientras que DD_{fmp} es la suma calórica del cultivo de lechuga en el punto de madurez fisiológica en grados celcius. Para la evaluación de la variación relativa de la masa seca vegetal en función del tiempo ($RTDM_j$), se utilizó un modelo empírico propuesto por los autores, el cual depende de las temperatura diaria. Esta función es utilizada como función de recompensa.

$$RTDM_j = cos(\frac{\pi}{2}(1 - RD_j))$$

Se tomó por defecto la cantidad de días (35) en que la planta se va a evaluar.Es importante destacar que para el cálculo de la recompensa hace falta conocer las temperaturas de los 35 días del desarrollo para poder calcular DD_{fmp}.

$$RTDM_j = cos^{\alpha}(\frac{\pi}{2}(1 - RD_j))$$

El valor α se utiliza para ajustar la curva a los datos tomados experimentalmente.

3.3.4.4 Descripción de steps y episodios

Un step de este agente tiene como objetivo fijar la temperatura de un día y observar sus repercusiones. Como se describió en la función de recompensa, se fijó por defecto la cantidad de días en que la planta se va a evaluar. Esto tiene como consecuencia que un episodio estará conformado por 35 steps. Dado que en cada step se desea obtener la recompensa de ejecutar una de las acciones y el cálculo de la recompensa requiere la temperatura de los 35 días, se decidió que cualquiera de las acciones

ejecutadas en el día j, no solo modifica la temperatura de ese día, sino que también actualizará todas las posteriores hasta el día 35. El primer step del episodio fija la temperatura para los 35 días en el mismo valor, el segundo step fija la nueva temperatura del día 2 hasta el día 35 en el mismo valor, este proceso se repite hasta que en el step 35 el agente culmina de fijar todas las temperaturas (las cuales no tiene porque ser iguales).

3.3.4.5 Prueba del agente entrenado con DQN

Este agente se entreno con Q-learning y se fijaron los siguientes parámetros, los cuales se encontraron experimentalmente:

1. Leraning rate: 0,001.
2. Gamma: 0,999.
3. TIME_STEPS: 59500. Aumentando la cantidad de steps desmejora la política de elección de acciones.

A continuación en la figura 3.20 se puede observar la evolución de las recompensas promedios máximas a medida que aumentan la cantidad de episodios. Es claro que a partir del episodio 1300 comienza a seleccionar acciones que optimizan el crecimiento de la planta según la función de recompensa descripta anteriormente.

Figura 3.20: Recompesas del Agente

Dado que en este caso la función de recompensa no fue diseño propio, se procederá a analizar los resultados desde el punto de vista matemático y comparar dicho análisis con los resultados arrojados por el agente.

Desde el punto de vista matemático, al ser una función sinusoidal acotada entre 0 y 1 estrictamente creciente, da a entender que maximizará la recompensa cuando RD_{DD_n} alcance el valor máximo. Analizando a RD_{DD_n} se puede entender alcanzará dicho valor, cuando la temperatura acumulada sea lo más grande posible. En

consecuencia se espera que el agente luego de ser entrenado indique que todas las temperaturas deberían ser próximas a la máxima en el dominio (38°C). A continuación se detallan dos episodios y las acciones que el agente seleccionó.

En la tabla 3.8 se pueden visualizar los últimos dos episodios del entrenamiento de este agente. En ambos casos se puede comprobar la hipótesis matemática indicada anteriormente. Las acciones seleccionadas por dicho agente en todos los steps son la 27 o 28. Esto se traduce en fijar la temperatura en 37°C o 38°C, lo cual hace pensar en que si aumentamos aún más el dominio de la temperatura seguirá eligiendo temperaturas más altas. Si bien es lo óptimo para esta función de recompensa , no lo es así para la planta.

Episodio 1				**Episodio 2**		
Step	Acción	Recompensa		Step	Acción	Recompensa
1	27	0,013		1	28	0,012
2	28	0,021		2	28	0,022
3	28	0,056		3	28	0,057
4	27	0,093		4	28	0,092
5	27	0,127		5	28	0,126
6	27	0,161		6	28	0,161
7	27	0,195		7	28	0,195
8	27	0,229		8	28	0,229
9	27	0,262		9	28	0,263
10	27	0,295		10	28	0,296
11	27	0,328		11	28	0,329
12	27	0,36		12	28	0,362
13	27	0,392		13	28	0,394
14	27	0,423		14	28	0,426
15	27	0,454		15	28	0,457
16	27	0,484		16	28	0,488
17	27	0,514		17	28	0,518
18	27	0,543		18	28	0,547
19	27	0,572		19	28	0,576
20	27	0,6		20	28	0,604
21	27	0,627		21	28	0,632
23	27	0,653		23	28	0,658
23	27	0,679		23	28	0,684
24	27	0,704		24	28	0,709
25	28	0,724		25	28	0,733
26	27	0,751		26	3	0,851
27	28	0,771		27	28	0,77
28	28	0,793		28	28	0,792
29	28	0,814		29	28	0,813
30	28	0,834		30	28	0,833
31	28	0,853		31	28	0,852
32	28	0,871		32	28	0,87

Step	Acción	Recompensa		Step	Acción	Recompensa
33	28	0,888		33	28	0,887
34	28	0,903		34	28	0,903
35	28	0,903		35	28	0,903

Tabla 3.8: Episodios finales

Para poder comprender mejor lo que logra este agente, a continuación en la figura 3.21 se pueden visualizar 4 series de datos. Cada una corresponde a un episodio y sus recompensas. Es importante recordar que la recompensa no solo esta asociada a la temperatura fijada en cada step, sino también a la de todos los días del episodio. La curva azul corresponde con una etapa del agente entrenado, mientras que las demás a otros episodios previos del entrenamiento. En términos de desarrollo y en base a este escenario particular, se puede concluir que la materia seca de la lechuga (en el mejor escenario) es aproximadamente un 20 % superior al resto de los casos. Este resultado es relativo al modelo matemático descripto anteriormente.

Figura 3.21: Recompensas del Agente

3.3.5. Agente 5

Este agente es una extensión del Agente 4 involucrando la radiación solar además de la temperatura diaria. Retomando la pregunta planteada anteriormente, ¿cuáles son los valores de temperatura y luminosidad que optimizan el crecimiento?. En el articulo publicado por Durval Dourado-Neto et. al también propusieron una función que modela el desarrollo relativo de la materia seca de la lechuga en base a la temperatura y la luminosidad. Esta función será descripta en la subsección 3.3.5

3.3.5.1 Descripción del estado del ambiente

El estado del ambiente en este caso se corresponde con un array de tres posiciones compuesto por:

- La temperatura del ambiente.
- La luminosidad del ambiente en luxes.
- El día de evolución de la lechuga.

3.3.5.2 Descripción de la acciones

Este agente contiene todas las acciones del Agente 4, pero además todas las luminosidades discretas descriptas en el Agente 3. En consecuencia habrán 29 acciones correspondientes a fijar temperaturas y 31 acciones encargadas de fijar luminosidades.

Acciones de temperatura:

- **Acción 0**: fijar temperatura en 10°C.
- **Acción 1**: fijar temperatura en 11°C.
- **Acción 2**: fijar temperatura en 12°C.
  .
  .
- **Acción 28**: fijar temperatura en 38°C.

Acciones de Luminosidad:

- **Acción 0**: fijar luminosidad en 396 luxes. Corresponde a colocar las pantallas de luces en la parte superior del ambiente.
- **Acción 1**: fijar luminosidad en 405 luxes. Corresponde a colocar las pantallas de luces a 1 cm de la parte superior del ambiente.

- **Acción 2**: fijar luminosidad en 414 luxes. Corresponde a colocar las pantallas de luces a 2 cm de la parte superior del ambiente.
  .
  .
- **Acción 30**: fijar luminosidad en 778 luxes. Corresponde a colocar las pantallas de luces en la parte inferior del ambiente.

Dado que el espacio de acciones en este caso no puede ser un vector o par de acciones que modelen la temperatura y la luminosidad fijada cada día. Se decidió modelar una matriz $M \in M_{TxL}$, donde T es el número de acciones de temperatura y L la cantidad de acciones de luminosidad. En la cual el número de la fila representa la acción que fija la temperatura y el número de la columna la acción que fija la luminosidad, mientras que el valor dentro de ella la acción con la cual se corresponde el par.

Siendo

$$M = ((m_{ij})) \ / \ m_{ij} = i \cdot \#AL + j, \ 0 \leq i \leq \#AT, \ 0 \leq j \leq \#AL$$

donde #AL es la cantidad de acciones de luminosidad y #AT es la cantidad de acciones de temperatura. La misma matriz se puede observar a continuación en la ecuación 3.1.

$$M = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & ... & 29 & 30 \\ 31 & 32 & 33 & 34 & 35 & ... & 60 & 61 \\ 62 & 63 & 64 & 65 & 66 & ... & 91 & 92 \\ . & . & . & & & & . & . \\ . & . & . & & & & . & . \\ . & . & . & & & & . & . \\ 868 & 869 & 870 & 871 & 872 & ... & 897 & 898 \end{pmatrix} \tag{3.1}$$

Si comenzamos a nombrar desde la fila 0 y la columna 0 (desde el punto de vista de la programación facilita su generalización). Por ejemplo la acción 63 esta ubicada en la fila 2 y columna 1, en consecuencia si se ejecuta dicha acción la temperatura se fijara en $12C$ mientras que la luminosidad lo hará en 405 luxes.

Para no tener que almacenar con qué par de acciones de temperatura y luminosidad se correspondía cada valor de la matriz, se reconoció un patrón que simplifica la identificación de acciones. La parte entera de dividir el valor de la matriz entre la cantidad de columnas corresponde con la fila (acción de temperatura) y el módulo de dividir al mismo valor por la cantidad de columnas corresponde con la columna (acción de la luminosidad). A continuación se pueden visualizar las operaciones realizadas por el lenguaje de programación.

```
def parAccionesTL(cantAccionesTemp,cantAccionesLuz,valorMatriz):
  accionTemp=math.trunc(valorMatriz/cantAccionesLuz)
  accionLuz=valorMatriz % cantAccionesLuz
  return accionTemp,accionLuz
```

3.3.5.3 Función de recompensa

Dado que esta función de recompensa es una extensión de la descripta en el Agente 4, solo se mencionarán aquellos cálculos no introducidos previamente.

$$RD_{EDD_n} = \left[\frac{1}{\frac{1}{\sum_{j=1}^{n}(\hat{Tj} - T_{BI})} + g * \frac{1}{IPAR_n}} \right] \frac{1}{EDD_{fmp}}$$

$IPAR_n$ es la radiación fotosintéticamente activa entrante en $MJ\,m^{-2}\,d^{-1}$ en el día n, considerando que los sensores miden la luminosidad en lux se debe multiplicar por 0,0079 para realizar la conversión.

EDD_{fmp} es la temperatura efectiva acumulada hasta el punto de madurez fisiológica (ºC día); g es una constante para la especie, referida al efecto relativo de la luz y la temperatura del aire para lechuga; RD_{EED_n} refiere al desarrollo relativo de la lechuga hasta el día n.

Para la evaluación de la variación relativa de la masa seca vegetal con el tiempo ($RTDM_n$) se utilizó la misma función que en el Agente 4 pero ajustada a utilizar RD_{EDD_n} y no RD_{DD_n}. Esta función es utilizada como función de recompensa del agente.

$$RTDM_n = cos(\frac{\pi}{2}(1 - RD_{EED_n}))$$

3.3.5.4 Descripción de steps y episodios

Un step de este agente tiene como objetivo fijar la temperatura y la luminosidad de un día y observar sus repercusiones. Este agente vuelve a tener 35 steps de largo, dado que es el tiempo en que se quiere evaluar la lechuga. Se reitera la política de la ejecución de acciones (cualquiera de las acciones ejecutadas en el día j, no solo modifica la temperatura y luminosidad de ese día, sino que también actualiza todas las posteriores hasta el día 35).

3.3.5.5 Prueba del agente entrenado con DQN

Este agente se entreno con Q-learning y se fijaron los siguientes parámetros, los cuales se encontraron experimentalmente:

1. Leraning rate: 0,001.
2. Gamma: 0,999.
3. TIME_STEPS: 65100.

A continuación en la figura 3.22 describe las recompensas obtenidas al día 35 a lo largo del entrenamiento.

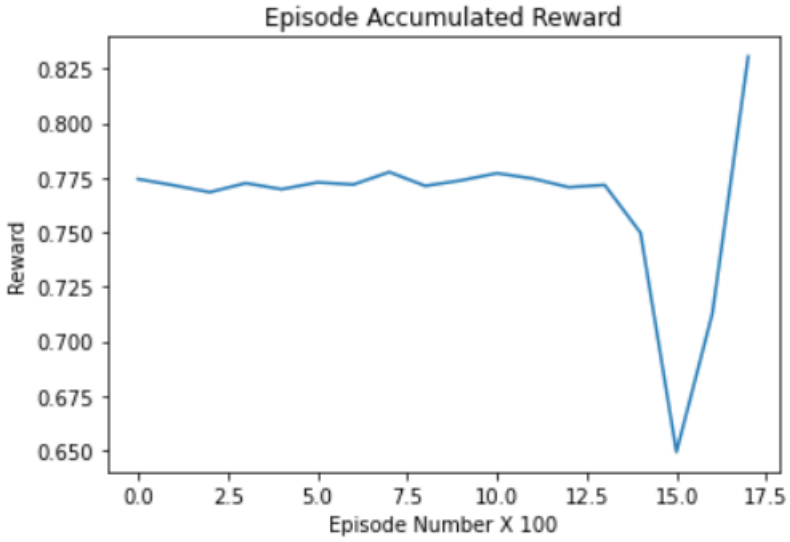


Figura 3.22: Recompesas del Agente

Como se puede observar la recompensa aumenta considerablemente sobre el final del entrenamiento. Si bien se realizaron otras pruebas con mayor cantidad de steps. Aumentar dicho parámetro sólo logra empeorar las políticas de selección de acciones. Para poder comprender mejor las recompensas alcanzadas y no analizar tan solo su valor numérico, se procederá a comparar episodios intermedios contra finales en el entrenamiento además de las acciones que el agente selecciona.

La curva naranja y azul de la figura 3.23 son dos episodios finales, mientras que la gris y la amarilla son dos episodios intermedios. Nuevamente se repite que comparando el mejor contra el peor caso de los estudiados, el agente logra incrementar el desarrollo relativo de la materia seca en un 20 % aproximadamente.

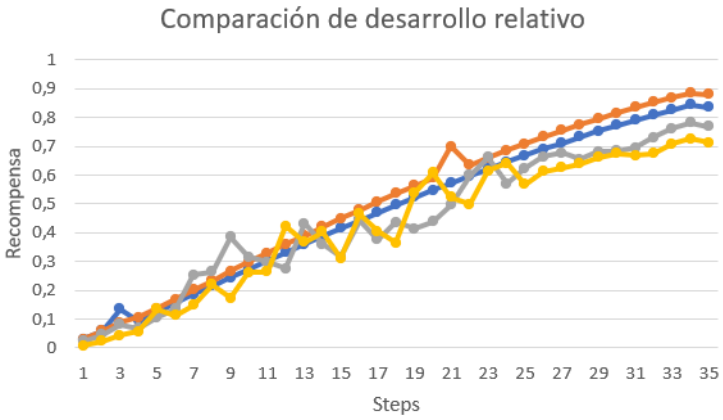


Figura 3.23: Recompesas del Agente

En las tablas 3.9 se puede visualizar en detalle las acciones seleccionadas y las recompensas bosquejadas anteriormente en la curva azul.

Step	Acción de Temperatura	Acción de Luminosidad	Recompensa
1	20	26	0,0297545935584992
2	20	26	0,0594727622811489
3	20	26	0,0891382559959802
4	27	18	0,106344252444749
5	28	13	0,136688545251865
6	28	13	0,169317749541877
7	28	13	0,201728173138851
8	28	13	0,233896416463081
9	28	13	0,265794744676823
10	28	13	0,297393188936764
11	28	13	0,328660621993018
12	28	13	0,359565356984914
13	28	13	0,390075506334447
14	28	13	0,420159211910509
15	28	13	0,449784802295391
16	28	13	0,478920906811156
17	28	13	0,507536542821513
18	28	13	0,535601185883388
19	28	13	0,563084828492989
20	28	13	0,589958030977287
21	13	3	0,698972282139361
22	28	13	0,634727461672797
23	28	13	0,660091345113703
24	28	13	0,684723115089447
25	28	13	0,708595669433298
26	28	13	0,731682723984141
27	28	13	0,753958845585989
28	28	13	0,775399483341237
29	28	13	0,795980998270507
30	28	13	0,815680691485248
31	28	13	0,834476830946196
32	28	13	0,852348676857049
33	28	13	0,869276505725699
34	28	13	0,885241633113007
35	28	13	0,879884348251962

Tabla 3.9: Episodio final

Nuevamente una de las soluciones que ofrece el agente es maximizar la temperatura como se vio en el agente 4. Se recuerda que la acción 28 corresponde con fijar la temperatura en 38°C.

3.3.6. Resumen

En resumen en este capitulo se describieron 5 agentes, sus acciones, funciones de recompensas y ambientes simulados. En los primeros 3 se lograron encontrar políticas que logran dar soluciones a los problemas planteados. El primer agente teniendo 3 acciones disponibles(calentar, enfriar y ventilar) logra alcanzar una temperatura fija de 29.5°C optimizando las acciones ejecutas por el consumo del modelo de los dispositivos. Aprende que cuando se desea incrementar la temperatura debe ejecutar la acción calentar, que cuando quiere disminuir la temperatura debe ejecutar la acción enfriar y dependiendo de la temperatura exterior del ambiente cuando ventilar para lograr equilibrio térmico. Se pudo observar que el 80 % de las veces que elige una acción prioriza minimizar el consumo y cuando no lo hace es por el tiempo que le podría insumir llegar a un estado final ejecutando acciones de ventilación. En general las acciones forzadas(calentar o enfriar) se ejecutan sobre todo al inicio con el fin de acercarse a los 29.5°C. El segundo agente surge con la necesidad de probar

si otras temperaturas podrían ser aún mejores que la planteada anteriormente. Para esto el sistema tiene que tener la capacidad no sólo de alcanzar una temperatura fija sino también cualquier temperatura del dominio. Por dicha razón se incorpora al estado del ambiente la temperatura objetivo. El propósito del agente es encontrar la política óptima de ejecución de acciones, que logre reducir la distancia entre la temperatura del ambiente y la objetivo en cada episodio(la temperatura objetivo en cada episodio es variable). Las acciones son las mismas que en el agente inmediatamente anterior, pero la función de recompensa se ve adaptada a una generalización de la misma. Este agente logra aprender lo deseado pero con un *rate* inferior (50 %) al del agente anterior. Esto quiere decir que ejecuta más veces acciones forzadas en comparación al agente anterior, pero logra llegar a estados finales optimizando el consumo. El tercer agente involucra a la luminosidad, con la necesidad de determinar la luminosidad óptima que las plantas requieren, el sistema primero debe aprender a regular la luminosidad en el ambiente real. El objetivo de este agente es aprender a regular automáticamente la luminosidad en luxes dentro del ambiente. No busca optimizar recursos, en consecuencia no sería necesario entrenar un agente, dado que la mejor solución al problema es muy clara, cada altura de las luces en el ambiente refleja una única luminosidad. Independientemente de esto, se decidió verificar si el agente lograba aprender dichas políticas. Las acciones de este agente son subir o bajar (las luces mecánicamente) y el estado esta conformado por la luminosidad del ambiente y la luminosidad óptima. Su objetivo es encontrar la política óptima de ejecución de acciones, que logre reducir la distancia entre la luminosidad del ambiente y la objetivo, ejecutando la menor cantidad de acciones. En varios episodios finales se puede observar que el agente aprende cuales son las acciones que logran incrementar y reducir la luminosidad, además de alcanzar estados finales. En algunos episodios se puede observar que no todas las acciones son óptimas(lo cual es menor dado que cada acción insume muy poco tiempo en el ambiente), pero logra llegar a estados finales. El largo del episodio es solo un indicador porque si la distancia entre la luminosidad inicial y la objetivo es muy grande seguro le va a tomar mayor cantidad de *steps*. Hasta aquí se describieron los agentes de bajo nivel. Los de alto

nivel son los agentes 4 y 5. Su objetivo principal es aprender a fijar temperaturas y luminosidades óptimas dentro del dominio de las mismas (en diferentes etapas del crecimiento de las lechugas). Estos agentes desconocen como hacer para que el ambiente real alcance dichos valores, de esto se encargan los agente 2 y 3. El agente 4, solo se enfoca en aprender a determinar la temperatura óptima en cada etapa del crecimiento. Este agente puede ejecutar 29 acciones, cada una representa fijar una temperatura entera entre 10°C y 38°C. Las métricas y funciones de recompensas utilizadas son las recomendadas por Sidinei José Lopes en su articulo [10]. El estado es representado por la temperatura del ambiente y el día de desarrolla acumulado. Se pudo observar que una vez entrenado el agente, el mismo eligió maximizar la temperatura para maximizar las recompensas. Este resultado no es deseado dado que si se aumenta el dominio de las temperaturas, seguirá aumentando la temperatura óptima. El último y quinto agente es una extensión del agente 4, en el cual además de considerar la temperatura, involucra a la radiación solar. El conjunto de acciones aumenta considerablemente y se pueden observar resultados similares a los del agente anterior.

Capítulo 4

Conclusiones

Como conclusiones del presente proyecto, se logró con éxito extender la investigación realizada en la tesis de ingeniería, ahondando en los dos pilares principales planteados: El desarrollo de un entorno virtual para la generación de datos de crecimiento de lechugas y el desarrollo de agentes más complejos en su capacidad de toma de decisiones.
Esto incluyó por una parte el diseño, modelado y fabricación del *Virtual Lettuce Generator*. Por otro lado, se desarrollaron 5 agentes inteligentes que resuelven diversos problemas en la gestión de recursos en el ambiente hidróponico. El primer agente logró mejorar los resultados conseguidos en un estudio previo seleccionando el 80 % de las veces acciones que minimizan el consumo. El segundo agente, no solo logró optimizar el consumo, sino también aprendió a hacerlo independiente de la temperatura objetivo que se fijara. El tercer agente aprendió a controlar el ambiente para fijar en el mismo una luminosidad deseada. El cuarto agente logró en base a su función de recompensa optimizar las temperaturas en cada etapa del desarrollo de la lechuga. Se comporta de forma similar al cuarto no dando indicio de la importancia de maximizar la luminosidad. Si bien los últimos dos agentes logran obtener resultados de interés, considero que no son óptimos para cultivos hidropónicos reales puesto que las funciones de recompensa utilizadas son casos particulares de la realidad en otro contexto, sin considerar que la temperatura y la luminosidad puedan ser variables controlables.

4.1. Mejoras a Futuro

Dado que ya se dispone de un sistema de base para entrenar agentes inteligentes y además un entorno para comenzar a extraer datos (VLG); el siguiente paso es comenzar a generar modelos virtuales de lechugas, con el objetivo de que los agentes interactúen con estos modelos. De esta forma no solo se podrán entrenar agentes que optimicen la temperatura y la luminosidad, sino también los nutrientes. Para poder lograr lo antes mencionado, se deberán almacenar datos y diferentes escenarios con VLG los cuales permitan a modelos de Machine Learning entrenarse con cada objetivo particular.

Capítulo 5

Referencias Bibliográficas

[1] A. N. López-Anchondo, C. E. López-Ortiz, C. M. Mejía-Hernández, and D. L. de-la Cruz1, "Hidroponia una alternativa sustentable para el cultivo sin suelo: características y aspectos básicos." [Online]. Available: url@hyphttps://www.researchgate.net/profile/Cristina-Garcia-De-La-Pena/publication/320472196_Productos_organicos_y_fitohormonas_efecto_en_la_concentracion_de_aminoacidos_en_tuberculos_de_Caladium_bicolor_en_dos_etapas_fonologicas/links/59eaaa360f7e9bfdeb6cce58/Productos-organicos-y-fitohormonas-efecto-en-la-concentracion-de-aminoacidos-en-tuberculos-de-Caladium-bicolor-en-dos-etapas-fonologicas.pdf#page=31

[2] M. Wurth, "Diseño y construcción de un sistema biológico, hidropónico y autónomo basado en inteligencia artificial." [Online]. Available: url@hyphttps://dspace.ort.edu.uy/handle/20.500.11968/4827

[3] A. Mokhtar, W. El-Ssawy, H. He, N. Al-Anasari, S. S. Sammen, Y. Gyasi-Agyei, and M. Abuarab, "Using machine learning models to predict hydroponically grown lettuce yield," *Frontiers in Plant Science*, vol. 13, 2022. [Online]. Available: url@hyphttps://www.frontiersin.org/articles/10.3389/fpls.2022.706042

[4] M. Ahsan, S. Eshkabilov, B. Cemek, E. Küçüktopcu, C. W. Lee, and H. Simsek, "Deep learning models to determine nutrient concentration in hydroponically grown lettuce cultivars (lactuca sativa l.)," *Sustainability*, vol. 14, no. 1, 2022. [Online]. Available: url@hyphttps://www.mdpi.com/2071-1050/14/1/416

[5] W. J. G. M. van den Bemd, "Robust deep reinforcement learning for greenhouse control and crop yield optimization." [Online]. Available: url@hyphttps://research.tue.nl/en/studentTheses/robust-deep-reinforcement-learning-for-greenhouse-control-and-cro

[6] OpenIA, "Proximal policy optimization." [Online]. Available: url@hyphttps://openai.com/blog/openai-baselines-ppo/

[7] M. I. Alipio, A. E. M. Dela Cruz, J. D. A. Doria, and R. M. S. Fruto, "A smart hydroponics farming system using exact inference in bayesian network," in *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, 2017, pp. 1–5.

[8] G. Dbritto and S. Hamdare, "An ai based system design to develop and monitor a hydroponic farm," in *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, 2018, pp. 1–5.

[9] R. Yang, Z. Wu, W. Fang, H. Zhang, W. Wang, L. Fu, Y. Majeed, R. Li, and Y. Cui, "Detection of abnormal hydroponic lettuce leaves based on image processing and machine learning," *Information Processing in Agriculture*, vol. 10, no. 1, pp. 1–10, 2023. [Online]. Available: url@hyphttps://www.sciencedirect.com/science/article/pii/S2214317321000834

[10] S. Lopes, D. Dourado-Neto, P. Manfron, and L. Jasniewicz, "Models to estimate phytomass accumulation of hydroponic lettuce," *Scientia Agricola*, vol. 61, pp. 392–400, 12 2003.

[11] I. Kargar, "Rl series-a2c and a3c." [Online]. Available: url@hyphttps://kargarisaac.medium.com/rl-series-a2c-and-a3c-in-pytorch-6e9edf5c8788

[12] OpenIA, "Gym documentation." [Online]. Available: url@hyphttps://www.gymlibrary.dev/content/basic_usage/

[13] Arduino, "Getting started with arduino products." [Online]. Available: url@hyphttps://www.arduino.cc/en/Guide

[14] OpenCv, "Opencv." [Online]. Available: url@hyphttps://opencv.org/releases/

[15] TutorialsPoint, "Overview opencv." [Online]. Available: url@hyphttps://www.tutorialspoint.com/opencv/opencv_overview.htm#:~:text=OpenCV%20is%20a%20cross%2Dplatform,face%20detection%20and%20object%20detection.

[16] Wikipedia, "Agente inteligente." [Online]. Available: url@hyphttps://es.wikipedia.org/wiki/Agente_inteligente_(inteligencia_artificial)

[17] A. Vidhya, "A hands-on introduction to deep q-learning using openai gym in python." [Online]. Available: url@hyphttps://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

[18] OpenIA, "Openai." [Online]. Available: url@hyphttps://openai.com/

[19] M. Plappert, "keras-rl," 2016. [Online]. Available: url@hyphttps://github.com/keras-rl/keras-rl

[20] PyTorch, "Pytorch." [Online]. Available: url@hyphttps://pytorch.org/

[21] Geogebra, "Geogebra." [Online]. Available: url@hyphttps://www.geogebra.org/?lang=es-UY

[22] P. D. Christopher J. C. H. Watkins, "Q-learning." [Online]. Available: url@hyphttps://link.springer.com/article/10.1007/BF00992698

[23] Baselines3, “Stable-baselines3 docs - reliable reinforcement learning implementations.” [Online]. Available: url@hyphttps://stable-baselines3.readthedocs.io/en/master/

[24] M. Wang, “Advantage actor critic tutorial: mina2c.” [Online]. Available: url@hyphttps://towardsdatascience.com/advantage-actor-critic-tutorial-mina2c-7a3249962fc8#:~:text=In%20the%20field%20of%20Reinforcement, input%20states%20to%20output%20actions.

Capítulo 6
Anexos

6.1. Link del repositorio

Link del repositorio: `url@hyphttps://gitlab.com/castelli.m/MasterEnIngenieria`

6.2. Ambiente anterior

Esta anexo muestra los resultados de la construcción del ambiente desarrollado previamente en la tesis de ingeniería.
Como se puede observar en la figura 6.1 sobre la imagen se pueden distinguir rectángulos de colores, los cuales ubican la mayoría de los componentes del mismo. Continuando en la misma figura, el color:

- naranja; representa el sensor de distancia que mide la temperatura a la base de la bandeja para incrementar o reducir la intensidad de luz en las plantas.
- rojo; representa la entrada de aire de los caloventiladores.
- azul; representa la entrada de aire frío del sistema de refrigeración.
- verde; representa la ubicación de todos los sensores interiores al ambiente.
- negro; representa la cámara de recolección de imágenes.
- violeta; representa la luces de amplio espectro móviles.

Continuando con la figura 6.2, el color:

- naranja; representa uno de los huecos en los cuales se pueden colocar plantas.
- azul; una zona de comunicación entre el interior y el exterior de la bandeja a través de la cual el humidificador ultrasónico entrega humedad al ambiente.
- rojo; representa un espacio no utilizable a través del cual se logra medir la altura real de las luces de amplio espectro además de la cámara. Esta distancia permita establecer métricas para identificar el tamaño de las plantas.

La figura **??** representa la zona inferior del ambiente, se observan los siguientes colores:

- celeste; identifica el sensor de nivel de agua.
- amarillo; identifica el sensor de pH.
- rosado; identifica el sensor de nutrientes del agua.

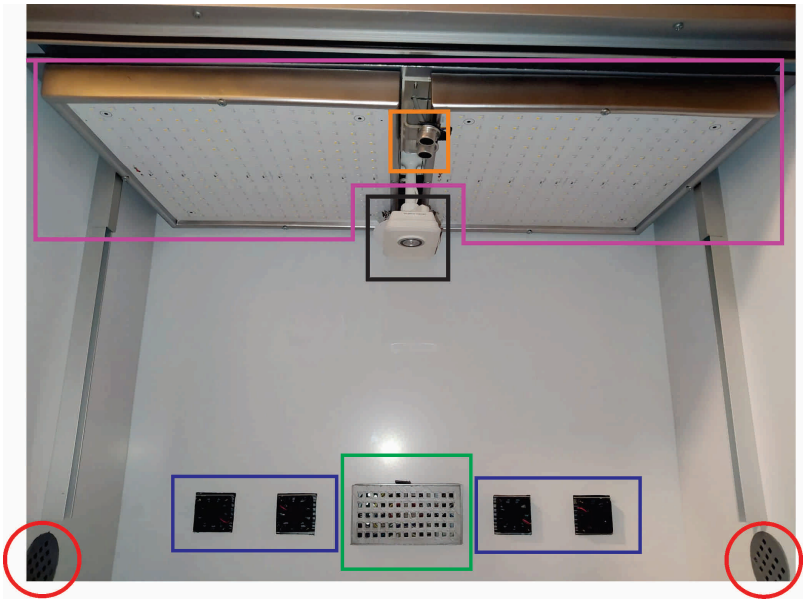


Figura 6.1: Zona superior

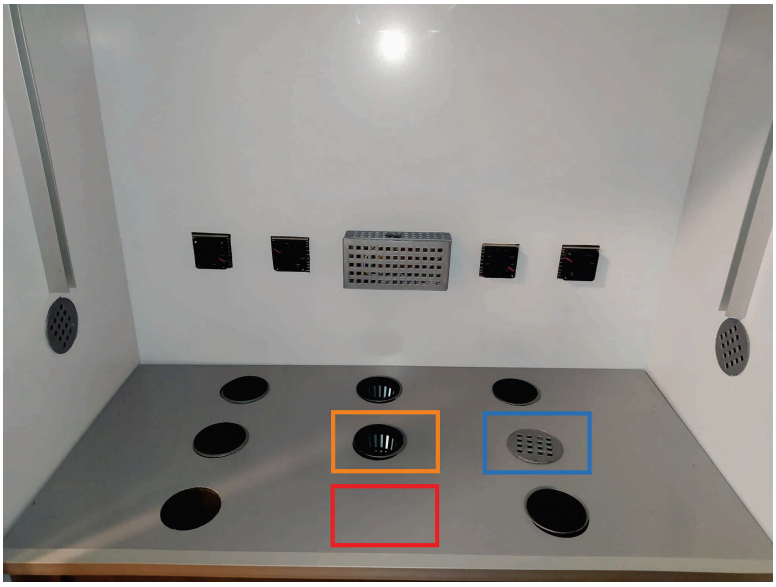


Figura 6.2: Zona media

- rojo; identifica la bomba de agua.
- verde; identifica la pieza de centralización de administración de nutrientes.
- naranja; identifica un receptáculo de nutrientes con su correspondiente bomba peristáltica de precisión.

En la figura **??** Se puede visualizar el interior completo de la bandeja, en la cual en color naranja se identifica el humidificador ultrasónico con su respectivo accesorio con el fin de funcionar correctamente.