



Universidad  
Tecnológica  
del Perú

Facultad de Ingeniería  
Ingeniería de Sistemas e Informática

**Tesis:**

**“Desarrollo de una Interfaz Gráfica para la Definición de Áreas de Podado de Césped Utilizando Visión Artificial para la Interacción con un Robot Autónomo”**

**LEANDRO GUSTAVO ORTEGA PALACIOS**

para optar el Título Profesional de:  
**Ingeniero de Sistemas e Informática**

Asesor:

**MSc. Hugo Angel Barreda de la Cruz**

Arequipa – Perú

2022

## **AGRADECIMIENTOS**

Agradezco a mi madre, por su amor incondicional; a mi padre, por que pude contar con su apoyo en todo momento; a Jenny, porque está conmigo, hasta para corregirme un punto y coma; a mis tíos, por su aliento para ayudarme encontrar la tranquilidad en momentos de estrés; a Rayza, por su preocupación incansable; a mi gran amigo Diego, por su apoyo, guía y cooperación; y a mi asesor, por su dedicación y compromiso en la elaboración de esta tesis.

El desarrollo de esta tesis fue financiado por el Programa Nacional de Innovación para la Competitividad y Productividad (Innovate Perú), de acuerdo al contrato N°320INNOVATEPERUPIEC12019.

## **RESUMEN**

La automatización de procesos se ha transformado en una praxis cada vez más común, esto es principalmente aplicado en tareas manuales de naturaleza repetitiva o que representan un peligro para la integridad del personal o que presenten riesgo a su calidad de vida, en estos escenarios la automatización representa una opción viable y preferible. Esta automatización de procesos, es realizada por máquinas controladas y/o robots autónomos, son usualmente aplicados al campo aeroespacial, marítimo, seguridad entre muchos otros. La tecnología moderna hace posible que estos robots operen con supervisión mínima.

Como otro ejemplo podemos mencionar el uso de robots, en la agricultura tanto en el campo industrial como doméstico, enfocándose en precisión agrícola, insumos, metrología, proceso de información, siembra, riego y cosecha, liberando al agricultor de estas pesadas actividades. Más específicamente queremos mencionar el uso de estos autómatas para el podado de césped, con el fin de cortarlo con más regularidad y mantener más tiempo su aspecto; se espera también una mayor uniformidad del terreno establecido.

La empresa Opciones Ingeniería y Tecnología Ambiental S.R.L., ha desarrollado un robot podador autónomo con capacidades importantes que son necesarias para el podado de césped que incluye, sensores físicos para detectar obstáculos dentro de su trayecto, GPS para el rastreo de su posición, entre otros sensores. Esta investigación, se enfoca en desarrollar e integrar una Interfaz gráfica web para que los administradores del robot

autónomo especifiquen el área de trabajo en el que el robot se va a desenvolver. Esto conlleva a la necesidad de un buen servidor que pueda atender a los requerimientos online que este tipo de sistemas necesitan.

Dada la necesidad del proyecto, se debe contar con una interfaz gráfica que integre un sistema de geolocalización y manejo de coordenadas geográficas, para que estos datos puedan ser aprovechados por el robot y aportar en la autonomía del autómeta; en el desarrollo de este proyecto se utilizó las APIs de geolocalización de Google, el sistema es bastante completo, obviamente el requerimiento de cada proyecto necesita de algunas personalizaciones que deben ser implementadas por el programador encargado, a lo largo del texto el lector podrá fácilmente leer las necesidades realizadas en este proyecto. Es también necesaria la comprensión del sistema mecánico-electrónico para poder hacer que el servidor y la interfaz con la que los usuarios traten sea la más intuitiva, didáctica y de fácil adaptación. Es en este escenario, donde son necesarios los criterios de interacción persona-computadora, considerados en el diseño y creación de las interfaces de usuario, para así mejorar la experiencia que se tenga el usuario con el sistema. Malos criterios considerados al momento de diseñar una interfaz gráfica, disminuyen la eficiencia de trabajadores y/u operadores del sistema.

**PALABRAS CLAVE:** Automatización de procesos, Visión artificial, OpenCV, Google Maps JavaScript API, Google Maps Static API, EPSG:4326 WGS 84, EPSG:900913 Google Maps Global Mercator, NumPy, Matplotlib, UI, UX.

## **ABSTRACT**

Process automation has become an increasingly common praxis, they are mainly applied to manual tasks of a repetitive nature or that represent a danger to the integrity of the personnel or that present a risk to their quality of life, in these scenarios automation represents a viable and preferable option. This process automation is carried out by controlled machines and/or autonomous robots, they are usually applied to the aerospace, maritime, and security fields, among many others. Modern technology makes it possible for these robots to operate with minimal supervision.

As another example we can mention the use of robots, in agriculture both in the industrial and domestic fields, focusing on agricultural precision, inputs, metrology, information processing, planting, irrigation, and harvesting, freeing the farmer from these heavy activities. More specifically, we want to mention the use of these automata for mowing grass, to cut it more regularly and maintain its appearance for longer, a greater uniformity of the established terrain is also expected.

The company Opciones Ingeniería y Tecnología Ambiental S.R.L. has developed an autonomous mowing robot with important capabilities that are necessary for lawn mowing includes physical sensors to detect obstacles within its path, GPS to track its position, among other sensors. This research focuses on developing and integrating a graphical web interface for autonomous robot administrators to specify the work area in which the robot is going to

operate. This leads to the need for a good server that can meet the online requirements that this type of system needs.

Given the need for the project, a graphic interface must be taken into account that has a geolocation system and management of geographic coordinates, so that these data can be used by the robot and contribute to the autonomy of the automaton, in the development of this project the Google geolocation APIs were used, the system is quite complete the requirement of each project needs some customizations that must be implemented by the programmer in charge, throughout the text the reader will be able to easily read the necessary ones made in this draft.

It is also necessary to understand the mechanical-electronic system to make the server and the interface with which the users deal the most intuitive, didactic, and easy to adapt. It is in this scenario that human-computer interaction criteria considered in the design and creation of user interfaces are necessary, to improve the user's experience with the system. Bad criteria considered when designing a graphical interface decrease the efficiency of workers and/or system operators.

**KEYWORDS:** Process automation, Computer vision, OpenCV, Google Maps JavaScript API, Google Maps Static API, EPSG:4326 WGS 84, EPSG:900913 Google Maps Global Mercator, NumPy, Matplotlib, UI, UX.

## ÍNDICE

AGRADECIMIENTOS.....	ii
RESUMEN.....	iii
ABSTRACT .....	v
ÍNDICE .....	vii
ÍNDICE DE FIGURAS.....	xii
ÍNDICE DE TABLAS .....	xv
INTRODUCCIÓN.....	xvii
CAPÍTULO 1.....	1
GENERALIDADES .....	1
1.1. Descripción de la Problemática.....	1
1.1.1. Pregunta Principal de Investigación.....	3
1.2. Objetivos de la Investigación .....	3
1.2.1. Objetivo General.....	3
1.2.2. Objetivos Específicos .....	3
1.3. Justificación e Importancia.....	3
1.4. Alcance y Limitaciones .....	4
1.4.1. Alcance.....	4
1.4.2. Limitaciones.....	5
CAPÍTULO 2.....	6
GENERALIDADES .....	6

2.1.	Automatización de Procesos .....	6
2.2.	Geoposicionamiento .....	7
2.2.1.	Sistema Geodésico Mundial .....	8
2.2.2.	API de Maps JavaScript.....	8
2.2.3.	API de Maps Static .....	9
2.3.	Imágenes Digitales .....	9
2.3.1.	Definición.....	9
2.3.2.	Imágenes de Color .....	10
2.3.3.	Modelos de Colores.....	11
2.4.	Procesamiento de Imágenes .....	14
2.4.1.	Algoritmos en el Dominio Espacial.....	14
2.4.2.	Algoritmos de Extracción de Características.....	17
2.5.	Visión Artificial .....	18
2.6.	Python .....	20
2.6.1.	Flask.....	20
2.6.2.	OpenCV.....	20
2.6.3.	Scikit-Image.....	21
2.6.4.	NumPy.....	21
2.7.	Interacción Persona-Computadora .....	21
2.7.1.	Interfaz de Usuario .....	22
2.7.2.	Interfaz Gráfica de Usuario .....	22
2.7.3.	Experiencia de Usuario.....	22
2.8.	Robots Móviles .....	23



CAPÍTULO 3.....	24
ESTADO DEL ARTE.....	24
3.1. Interacción Persona - Computadora .....	24
3.2. Robot GrassBot .....	27
3.3. Visión Artificial .....	27
CAPÍTULO 4.....	29
METODOLOGÍA DE LA INVESTIGACIÓN .....	29
4.1. Tipo y Método de la Investigación.....	30
4.2. Diseño de la Investigación .....	30
4.3. Técnicas e instrumentos de colecta de datos .....	30
4.3.1. Técnicas .....	30
4.3.2. Instrumentos.....	31
4.3.3. Población.....	31
4.3.4. Muestra .....	31
4.4. Estudio de Casos .....	31
4.5. Operacionalización de Variables.....	32
CAPÍTULO 5.....	33
DESARROLLO DE LA TESIS.....	33
5.1. Desarrollo de la Interfaz Gráfica .....	34
5.1.1. Estructuración.....	35
5.1.2. Reconocimiento.....	36
5.1.3. Exploración.....	37
5.1.4. Modelado.....	37

5.1.5.	Ideación y Prototipado .....	46
5.1.6.	Formalización .....	48
5.1.7.	Implementación .....	49
5.1.8.	Validación en Contexto .....	49
5.1.9.	Despliegue .....	49
5.2.	Desarrollo del Sistema de Visión Artificial.....	51
5.2.1.	El Objetivo .....	52
5.2.2.	Adquisición de Imagen .....	53
5.2.3.	Procesamiento de Imagen .....	54
5.2.4.	Análisis de Imagen .....	56
5.2.5.	Toma de Decisiones .....	61
CAPÍTULO 6.....		62
ANÁLISIS E INTERPRETACIÓN DE RESULTADOS .....		62
6.1.	Funcionalidades de la Interfaz Gráfica.....	62
6.2.	Experiencia de Usuario de la Interfaz Gráfica .....	66
6.2.1.	Dimensión Simplicidad .....	66
6.2.2.	Dimensión Comunicación .....	68
6.2.3.	Dimensión Estructura .....	69
6.2.4.	Dimensión Navegación .....	71
6.2.5.	Dimensión Utilidad.....	73
6.2.6.	Evolución de las Dimensiones .....	76
6.3.	Sistema de Visión Artificial.....	81
6.3.1.	Base de Datos .....	81

6.3.2.	Consideraciones.....	82
6.3.3.	Métricas.....	82
6.3.4.	Resultados .....	83
6.3.5.	Otro Algoritmo Aplicado en el Sistema de Visión Artificial.....	93
	CONCLUSIONES .....	95
	RECOMENDACIONES.....	97
	ANEXO A.....	99
	REQUERIMIENTOS FUNCIONALES .....	99
	ANEXO B.....	102
	REQUERIMIENTOS NO FUNCIONALES.....	102
	ANEXO C .....	106
	ESPECIFICACIÓN DE CASOS DE USO.....	106
	ANEXO D .....	112
	CÓDIGO FUENTE.....	112
	ANEXO E.....	115
	DOCUMENTOS.....	115
	BIBLIOGRAFÍA.....	118

## ÍNDICE DE FIGURAS

Figura 1: Imagen en escala de grises y representación numérica de un fragmento 8x8. .	10
Figura 2: Composición de un píxel en una imagen a color. ....	11
Figura 3: Representación gráfica del modelo RGB. ....	12
Figura 4: Representación gráfica del modelo HSV. ....	13
Figura 5: Estructura de sistema de visión artificial. ....	19
Figura 6: Patrón de diseño modelo-vista-controlador mediado. ....	34
Figura 7: Fases de la metodología utilizada. ....	35
Figura 8: Diagrama de casos de uso – Listado de robots. ....	39
Figura 9: Diagrama de casos de uso – Definición de área de podado. ....	39
Figura 10: Diagrama de casos de uso - Definición de área de podado por visión artificial.	40
Figura 11: Diagrama de secuencia – Listado de robots. ....	41
Figura 12: Diagrama de secuencia – Definición de área de podado. ....	42
Figura 13: Diagrama de secuencia - Detección de área de podado por visión artificial. ...	43
Figura 14: Diagrama de clases - Modelo. ....	43
Figura 15: Diagrama de clases - Controlador. ....	44
Figura 16: Diagrama de clases - Vista. ....	44
Figura 17: Boceto - Lista de Robots. ....	45
Figura 18: Boceto - Información del Robot. ....	45
Figura 19: Boceto - Herramientas para definición de área de podado. ....	46
Figura 20 :Prototipo - Lista de Robots. ....	47
Figura 21: Prototipo - Información del Robot. ....	48
Figura 22: Prototipo - Herramientas para definición de área de podado. ....	48
Figura 23: Diagrama de despliegue de la interfaz gráfica. ....	51
Figura 24: Diagrama de actividades del sistema de visión artificial. ....	52

Figura 25: Conjunto de las diferentes áreas de aplicación del robot GrassBot.....	53
Figura 26: Mapa e imagen satelital obtenido a través de las APIs de Google. ....	54
Figura 27: Segmentación y binarización de imagen satelital. ....	55
Figura 28: Post-procesamiento de la imagen satelital. ....	56
Figura 29: Detección de bordes de la imagen satelital. ....	56
Figura 30: Conversión de píxeles a coordenadas de la proyección mundial. ....	60
Figura 31: Representación de las coordenadas resultantes en un mapa. ....	61
Figura 32: Visualización del listado de robots. ....	63
Figura 33: Vista previa de la geoposición de un robot en un mapa. ....	63
Figura 34: Visualización de sensores del robot. ....	64
Figura 35: Herramientas de dibujo para trazar el área de podado sobre un mapa. ....	64
Figura 36: Detección de área de podado a través de visión artificial. ....	65
Figura 37: Herramienta para guardar los datos generados en el mapa. ....	65
Figura 38: Muestra de información guardada desde la consola. ....	66
Figura 39: Simplicidad pregunta 1.....	66
Figura 40: Simplicidad pregunta 2.....	67
Figura 41: Simplicidad pregunta 3.....	68
Figura 42: Comunicación pregunta 4. ....	68
Figura 43: Comunicación pregunta 9. ....	69
Figura 44: Estructura pregunta 5.....	70
Figura 45: Estructura pregunta 8.....	70
Figura 46: Navegación pregunta 6.....	71
Figura 47: Navegación pregunta 7.....	72
Figura 48: Navegación pregunta 10.....	72
Figura 49: Navegación pregunta 11.....	73
Figura 50: Utilidad pregunta 12.....	74
Figura 51: Utilidad pregunta 13.....	75

Figura 52: Utilidad pregunta 14.....	76
Figura 53: Prototipo del modelo 1.....	77
Figura 54: Resultados de entrevista del modelo 1.....	77
Figura 55: Prototipo del modelo 2.....	78
Figura 56: Resultados de entrevista del modelo 2.....	78
Figura 57: Prototipo del modelo 3.....	79
Figura 58: Resultados de entrevista del modelo 3.....	79
Figura 59: Prototipo del modelo 4.....	80
Figura 60: Resultados de entrevista del modelo 4.....	80
Figura 61: Muestra gráfica de los casos para el cálculo de acierto.....	82
Figura 62: Ubicación que no contiene área de podado.....	83
Figura 63: Ubicación donde se detecta una posible área de podado.....	84
Figura 64: Mapas del caso 01 a analizar y la detección manual del área de podado.....	85
Figura 65: Resultado de la visión artificial en el caso 01.....	86
Figura 66: Comparación de resultados del caso 01.....	86
Figura 67: Muestra de acierto del sistema de visión artificial desde la consola del caso.....	87
Figura 68: Mapas del caso 02 a analizar y la detección manual del área de podado.....	88
Figura 69: Resultado de la visión artificial del caso 02.....	89
Figura 70: Comparación de resultados del caso 02.....	89
Figura 71: Muestra de acierto de la visión artificial desde la consola del caso 02.....	90
Figura 72: Mapas del caso 03 a analizar y la detección manual del área de podado.....	91
Figura 73: Resultado de la visión artificial en el caso 03.....	91
Figura 74: Comparación de resultados del caso 03.....	92
Figura 75: Muestra de acierto de la visión artificial desde la consola del caso 03.....	92

## ÍNDICE DE TABLAS

Tabla 1:Tabla de variables.....	32
Tabla 2: Estructura del trabajo. ....	36
Tabla 3: Especificaciones del subsistema.....	36
Tabla 4: Comparativa de resultados entre diferentes algoritmos.....	93
Tabla 5: Requerimiento funcional RF-001.....	99
Tabla 6: Requerimiento funcional RF-002.....	99
Tabla 7: Requerimiento funcional RF-003.....	100
Tabla 8: Requerimiento funcional RF-004.....	100
Tabla 9: Requerimiento funcional RF-005.....	100
Tabla 10: Requerimiento funcional RF-006.....	101
Tabla 11: Requerimiento no funcional RNF-001. ....	102
Tabla 12: Requerimiento no funcional RNF-002. ....	102
Tabla 13: Requerimiento no funcional RNF-003 ....	102
Tabla 14: Requerimiento no funcional RNF-004. ....	103
Tabla 15: Requerimiento no funcional RNF-005. ....	103
Tabla 16: Requerimiento funcional RNF-006. ....	103
Tabla 17: Requerimiento funcional RNF-007. ....	104
Tabla 18: Requerimiento funcional RNF-008. ....	104
Tabla 19: Requerimiento funcional RNF-009. ....	105
Tabla 20: Actor AT-001.....	106
Tabla 21: Caso de uso ID001. ....	106
Tabla 22: Caso de uso ID002. ....	106
Tabla 23: Caso de uso ID003. ....	107
Tabla 24: Caso de uso ID004. ....	107

Tabla 25: Caso de uso ID005. ....	107
Tabla 26: Caso de uso ID006. ....	107
Tabla 27: Caso de uso ID007. ....	108
Tabla 28: Caso de uso ID008. ....	108
Tabla 29: Caso de uso ID009. ....	108
Tabla 30: Caso de uso ID010. ....	109
Tabla 31: Caso de uso ID011. ....	109
Tabla 32: Caso de uso ID012. ....	109
Tabla 33: Caso de uso ID013. ....	109
Tabla 34: Caso de Uso ID014. ....	110
Tabla 35: Caso de uso ID015. ....	110
Tabla 36: Caso de uso ID016. ....	110
Tabla 37: Caso de uso ID017. ....	111
Tabla 38: Caso de uso ID018. ....	111
Tabla 39: Caso de uso ID019. ....	111



## **INTRODUCCIÓN**

La automatización de los procesos mediante máquinas autónomas, tiene como objetivo facilitar tareas repetitivas que son llevadas a cabo de manera manual por seres humanos. Muchas veces dichas actividades causan dificultades en la calidad de vida de las personas, y lo repetitivo de la actividad, a futuro, puede conllevar a una deficiencia en los procesos. En este escenario, la automatización se ha transformado en una opción factible con el objetivo de facilitar y perfeccionar las actividades con herramientas, que, dependiendo de la actividad, se usan programas, máquinas controladas y/o robots autónomos con la supervisión mínima posible. Como ejemplo de lo expuesto, tenemos el uso de robots en el podado del césped, con el fin de cortarlo con más regularidad y mantener por más tiempo un buen aspecto; se espera también una mayor uniformidad del terreno establecido.

Los robots en la agricultura se han enfocado en la precisión agrícola, lo que se entiende como la optimización de los recursos naturales e insumos, usando metodologías para medir y procesar información, así como actuadores para la siembra, riego y cosecha, ayudando al agricultor en estas actividades. Con el pasar del tiempo, los robots cada vez han ido interviniendo más en diferentes labores dentro de la agricultura. Por ejemplo, los drones que permiten evaluar los daños después de un desastre gracias a su capacidad aérea, hasta los que pueden medir parámetros ambientales para diversos fines; robots

recolectores que se basan en un sistema de cámaras para analizar y detectar si un fruto es cosechable e inclusive recolectar el mismo; robots construidos bajo mecanismos hidráulicos y neumáticos permiten adaptarse a los árboles y realizar la labor del desramado en tiempos reducidos, comparados a los tiempos de desramado manuales; robots podadores permiten cortar el césped de manera remota y/o automática debido a que están equipados con sensores que les permite trazar su nueva ruta de podado al encontrarse con un obstáculo; entre otros, vea [1].

La empresa Opciones Ingeniería y Tecnología Ambiental S.R.L., ha desarrollado un robot podador autónomo con capacidades importantes que son necesarias para el podado como: sensores ópticos para detectar obstáculos dentro de su trayecto, GPS para el rastreo de su posición, sensor de inclinación para obtener la gradiente del terreno, cable perimetral, el cual es el responsable de limitar el área de podado alcanzada por el robot y una estación de carga solar que servirá para la recarga energética del robot y la actualización de información con el sistema informático en la nube. Esta investigación se enfoca en desarrollar e integrar una interfaz gráfica web para que los administradores del robot autónomo especifiquen el área de trabajo en el que el robot se va a desenvolver. Esta interfaz es un programa informático compuesto por un conjunto de componentes gráficos que sirven para representar diferentes acciones posibles y todo tipo de información. Se desea utilizar visión artificial como ayuda o asistencia para definir el área de podado, reconociendo el área verde de imágenes satelitales a través de la posición terrestre del robot obtenida mediante su sensor de geoposición.

## **CAPÍTULO 1**

### **GENERALIDADES**

#### **1.1. Descripción de la Problemática**

En los años recientes, la ciencia de la robótica y el conjunto de tecnologías que conforman el internet de las cosas (IoT) se han utilizado en una gran gama de artefactos tecnológicos que se desenvuelven en los entornos domésticos, comerciales e industriales. Estos dispositivos trabajan de forma autónoma e incluyen gran parte de las tareas diarias, reuniendo datos, procesándolos y transmitiéndolos a internet para ponerla a disposición de sus usuarios, aunque no se encuentren de manera presencial. Si bien existen herramientas autónomas para el podado de césped a nivel internacional [1], en Perú contamos con limitantes económicas que dificultan la compra/importación de esta tecnología, adicionalmente nombrar limitaciones técnicas tecnológicas al no contar con personal capacitado en su operación y mantenimiento y menos personal profesional para su diseño, desarrollo y posterior comercialización. Habitualmente, la poda de áreas verdes se realiza con una periodicidad de 14 días y se obtiene material de desecho que al ser procesado puede ser compostado para ser reaprovechado. Cuando hablamos sobre la conservación de campos verdes de gran extensión, el césped es considerado como una plantación, a través del cual se realiza una inversión en insumos, como por ejemplo la mano de obra, convirtiéndose en costos significativos y, además, sumamos

las herramientas de podado habituales que consumen derivados del petróleo, lo que produce contaminación ambiental y gastos más prominentes [2].

Es en este contexto que, la empresa Opciones, conjuntamente con investigadores de una universidad privada en Arequipa, desarrollaron un robot podador, que permite reducir tiempos y costos, y facilita las labores de la conservación del césped. El robot podador autónomo, GrassBot, es utilizado para la conservación del césped decorativo, el uso de GrassBot, no solamente permite reducir costos en el capital humano, sino que puede operar todo el día, brindando al cliente un dispositivo con la capacidad de realizar los procesos de manera eficaz. Hablamos de un robot que realiza, juntamente con el podado de césped de manera autónoma, el sensado del terreno, el monitoreo, recopilación de datos sobre la salud del césped, y además, se autoabastece de energía solar almacenada en una estación de carga [2].

Todos los datos que GrassBot recopila, son enviados a un servidor privado con acceso a Internet para ser procesados y mostrados a través de una interfaz web hacia los usuarios. Así mismo, es necesario un apartado, dentro de esta interfaz web, donde se muestra la posición actual del robot y el área de podado en la que está trabajando, si así fuese requerido. Debido a que GrassBot, puede trabajar 24 horas al día, es requisito visualizar la geolocalización del robot, para permitir un mejor rastreo y seguimiento del mismo. Todo esto permitirá un mayor monitoreo sobre las diferentes características que maneja GrassBot.

El desarrollo de esta tesis, se basa en diseñar e implementar una interfaz gráfica que cumpla con los requisitos exigidos por el proyecto y que se complemente con el sistema informático que está siendo desarrollado en paralelo por el equipo técnico de la empresa Opciones Ingeniería y Tecnología Ambiental, esta interfaz gráfica deberá permitir al usuario administrador, observar, definir y editar el área de podado, así como mostrar los datos de los sensores, relacionados al campo, NPK (Nitrógeno, Fósforo y Potasio), humedad y temperatura, y los sensores de ubicación, inclinación,

colisión y conductividad del GrassBot, para el monitoreo del mismo. Adicionalmente se implementó un algoritmo de visión artificial para la detección automática de áreas verdes que este robot deba podar, que se basa en la geoposición del robot e imágenes satelitales.

### **1.1.1. Pregunta Principal de Investigación**

¿Cómo el desarrollo de una interfaz gráfica, permite monitorear sensores y definir un área de podado, utilizando visión artificial, para el proyecto de investigación que desarrolla el autómata GrassBot?

## **1.2. Objetivos de la Investigación**

### **1.2.1. Objetivo General**

Desarrollar una interfaz gráfica que, permita monitorear sensores y definir el área de podado con asistencia de visión artificial, para un proyecto de investigación que desarrolla el autómata GrassBot.

### **1.2.2. Objetivos Específicos**

- Desarrollar una interfaz gráfica que, permita monitorear los sensores y estados del autómata GrassBot.
- Integrar a la interfaz gráfica, la capacidad de mostrar el mapa satelital de acuerdo al geoposicionamiento del autómata GrassBot.
- Desarrollar un entorno que, permita la definición del área de podado.
- Integrar un sistema de visión artificial, para determinar un área de podado; basándose en el geoposicionamiento del autómata GrassBot.
- Validar el sistema de visión artificial.

## **1.3. Justificación e Importancia**

El país presenta una deficiencia en el desarrollo de tecnología, el Programa Innóvate Perú, financia diferentes proyectos que serán desarrollados con participación de

empresas que se envuelvan en proyectos de Investigación, Desarrollo e Innovación (IDI). Este trabajo se realizó conjuntamente con la empresa Opciones Ingeniería y Tecnología Ambiental S.R.L., para desarrollar un robot podador de césped autónomo diseñado para ser utilizado en clubes, parques cementerio, canchas de fútbol y césped ornamental. Este autómata, opera en campo de forma totalmente independiente, comandado desde internet de forma remota.

El desarrollo de este proyecto, presenta una justificación económica debido a que nos permite la reducción de costos en el mantenimiento del césped, combustible, esfuerzo, horas hombre y tiempo empleado; una justificación práctica, porque se automatizan las tareas que el proceso del podado de césped conlleva, y una justificación social, porque permite la integración de la tecnología conjuntamente con las actividades humanas, teniendo por consecuente que, el agrupamiento de estos factores, permite el desarrollo de actividades de manera sostenible.

La implementación de una interfaz gráfica, puede mejorar la eficacia con la que los operarios de un sistema rinden, en cuanto a la elaboración del trabajo. Para el proyecto planteado por Opciones, se espera tener una basta red de robots en diferentes campos de operación, es en este escenario que una red operada desde la nube presenta una opción viable.

Integrar visión artificial, permite detectar características particulares, como en la detección de bordes. Del mismo modo, se comprende que, gracias a la visión artificial, podemos obtener la información adecuada para resolver la problemática en la que nos encontramos.

## **1.4. Alcance y Limitaciones**

### **1.4.1. Alcance**

➤ La presente investigación está basada en el plan de investigación N° 03 - 2020 del proyecto "Mejora e ingeniería optimizada para desarrollo de un prototipo autónomo

y autosostenible de diagnóstico y mantenimiento de áreas verdes en la región Arequipa” con código de convenio N° 320- INNOVATEPERU-PIEC1-2019 de la empresa Opciones Ingeniería y Tecnología Ambiental S. R. L., que consiste en la integración de una interfaz gráfica que muestre el área de podado al sistema desarrollado por el equipo técnico de desarrollo del sistema informático en la nube.

➤ Se implementará una interfaz gráfica con la integración de la API “Maps JavaScript API” de Google, para mostrar mapas gráficos donde se indique la geoposición del robot GrassBot.

➤ Se implementará un sistema de visión artificial, integrado a la interfaz gráfica, enfocado en el procesamiento y análisis de imágenes satelitales basado en la geoposición del robot GrassBot.

➤ Las imágenes utilizadas en el sistema, serán obtenidas a través de la API “Maps Static API” de Google.

#### **1.4.2. Limitaciones**

➤ El sensor de GPS del robot GrassBot, tiene una precisión de  $\pm 5$  metros a la redonda.

➤ Las imágenes satelitales que nos brinda “Maps Static API” no son en tiempo real.

➤ En diferentes casos, las imágenes que se obtienen desde la API JavaScript Maps, difieren en tonalidades con las imágenes que se obtienen desde la API Static Maps.

➤ La validación del sistema de visión artificial, se realizará basándose en la relación del resultado obtenido por dicho programa con el sistema geodésico EPSG:4326 WGS 84 debido a que, el proyecto GrassBot, se encuentra en una etapa prematura para una validación real.

➤ Las imágenes obtenidas desde la API Maps Static de Google, tienen una resolución 640 x 640 píxeles y una calidad de imagen alta.

## **CAPÍTULO 2**

### **GENERALIDADES**

En este capítulo se detallan los diferentes fundamentos teóricos relacionados a las herramientas, modelos y procedimientos que se han utilizado en las diferentes etapas de esta investigación.

#### **2.1. Automatización de Procesos**

Entendemos por automatización de procesos que, es el conjunto de métodos que nos permiten simplificar tareas repetitivas. El hecho de que estas tareas sean repetitivas, implican problemas en el nivel de vida de las personas, y debido a procesos poco eficientes, en [3], se expone el porqué de la automatización de procesos. Trabajos como [4, 5], muestran el desarrollo de máquinas para automatizar el proceso de cortar el césped.

Una ventaja muy importante que se resalta en [3] acerca de la automatización de procesos, es el feedback, esto nos permite realizar una evaluación completa para luego ejecutar una comparación y correcciones de los procesos en tiempo real pero dejando flujos de trabajo (workflows) sobre tareas realizadas con anterioridad. Además, posibilita generar informes de todo el proceso o parte de ello, si se guardan registros del mismo. Así mismo, la programación de tareas que la automatización de procesos permite, nos brinda la capacidad de planificarlas sin supervisión. A



continuación, se detallan los beneficios más importantes en la automatización de procesos:

- Permite realizar controles y seguimiento sobre el proceso en todo momento de manera específica e íntegra, para conocer su estado de manera inmediata, evitando el efecto “caja negra” de las inmensas hojas de cálculo, en las que se suelen desarrollar.
- Reducción del tiempo en las etapas del proceso, es decir, desde que empieza hasta que finaliza.
- Obtener, en tiempo real, indicadores de desempeño (KPI) actualizados.
- Tener conocimiento, con exactitud, sobre lo acontecido en cada fase del proceso, mediante el registro o rastro que producen las actividades.
- Identificar, en las fases del proceso, dónde se aletargan las tareas y, las tareas innecesarias que no tienen un impacto en la mejoría del proceso.
- Recibir alertas automáticas de lo que acontece, en cualquier parte del proceso.
- Aplicar a los diversos procesos, el control mediante reglas de forma uniforme.
- Obtener resultados con el esfuerzo y coste mínimo, o que por lo menos, no sean tan variables.

## **2.2. Geoposicionamiento**

Una parte importante de esta investigación, es poder mostrar ubicaciones en mapas satelitales, por lo que, el sistema de geoposicionamiento, es un concepto fundamental para poder entender cómo convertir píxeles de una determinada imagen satelital en coordenadas esféricas para señalar una posición geográfica.

Geoposicionar significa situar un punto, que puede ser un objeto o una persona, en un plano cartográfico. En otras palabras, es poder ubicar dicho punto lo más exacto posible. Sin embargo, actualmente debido al limitado acceso a tecnologías de imágenes satelitales de las que disponemos, la geolocalización exacta al 100 % no

es alcanzable. Es por esto, que durante todos los años en los que se ha hablado del geoposicionamiento, siempre hemos contado con un margen de error. Dentro del geoposicionamiento se cuenta con 3 tipos de planos: planos de gran extensión, planos cortos y microlocalizaciones. El efecto inmediato, es pérdida de precisión cuando el usuario se encuentra en movimiento, debido a que hay una amplia probabilidad de que, al momento de realizar una acción, el usuario se encuentre en otra posición y fuera del radio de alcance [6].

### **2.2.1. Sistema Geodésico Mundial**

El WGS (*por sus siglas en inglés World Geodesic System*), es un marco de referencia que establece la latitud, longitud y altura para la navegación, posicionamiento y orientación a través de coordenadas tridimensionales. El WGS84, es un estándar, que representa una referencia geodésica global para aplicaciones como el geoposicionamiento, navegación, entre otras. Este estándar, incluye la definición de las constantes fundamentales y derivadas del sistema de coordenadas, el Modelo Gravitacional Terrestre (EGM) elipsoidal (normal), una descripción del Modelo Magnético Mundial (WMM) asociado y una lista actual de transformaciones de datum locales [7].

### **2.2.2. API de Maps JavaScript**

Es una API brindada por Google, que nos permite obtener un mapa de Google, siempre y cuando le enviemos parámetros necesarios para el mismo, los cuales son: latitud y longitud de la posición, tipo de visualización del mapa y nivel de acercamiento, con esto tendremos el componente necesario para poder mostrar en nuestra aplicación, la posición del robot que se seleccione a través de un mapa [8].

Esta herramienta contiene amplias funcionalidades que nos permiten manejar diferentes acciones como: dibujar, mover, editar líneas, círculos, rectángulos, controlar los diferentes eventos de interacción con estos mapas, entre otras. Lo que

nos lleva a poder definir el área de podado sobre un mapa con el uso adecuado de estas funcionalidades.

### **2.2.3. API de Maps Static**

Esta API de Google, funciona semejante a la API de Maps JavaScript pero con la diferencia que el componente retornado por la API, es un mapa en formato de imagen PNG, el cual nos permite ponerlo dentro de un componente de edición y de esta manera nos da la capacidad de dibujar sobre la imagen del mapa y representar el área de podado gráficamente, a través de las propias funcionalidades del propio lenguaje para el desarrollo de la página web, como por ejemplo, a través de un *canvas* [9].

## **2.3. Imágenes Digitales**

### **2.3.1. Definición**

Se entiende como imagen digital, al conjunto de píxeles ubicados en una posición dentro de un plano bidimensional determinado, en donde representan un color a través de un valor discreto y finito. Los píxeles son los puntos elementales de la imagen, y es la terminación habitual que se usa para indicar la unidad básica de medida, en una imagen digital [10].

En la Figura 1, se muestra la representación en escala de grises de una imagen. Cada píxel tiene el valor de un número entero entre 0 y 255 que interpreta la intensidad luminosa. En la misma figura, se ha obtenido un fragmento de una pequeña zona de la imagen, y se muestran los valores enteros en forma de matriz, que corresponden a cada elemento de la matriz  $P_{ij}$ , con las coordenadas en el plano  $x = i, y = j$ .

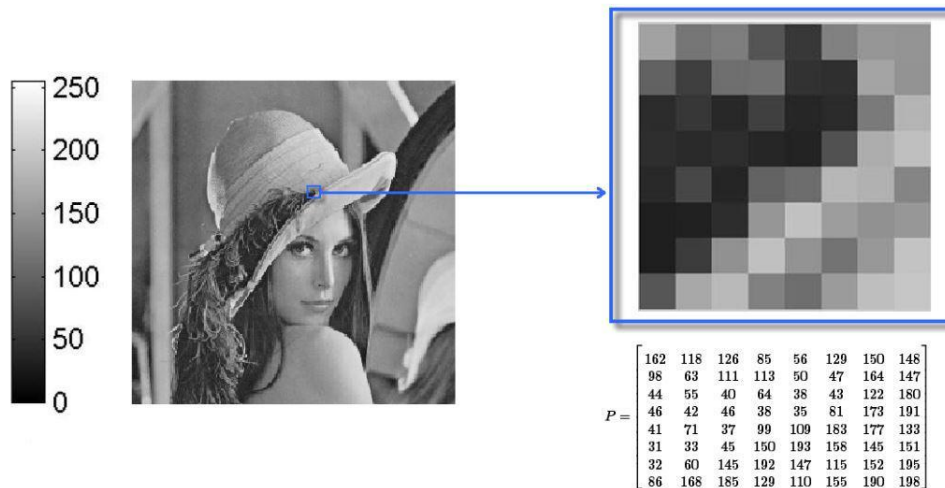


Figura 1: Imagen en escala de grises y representación numérica de un fragmento 8x8.

Fuente: Implementación de un sistema de detección de señales de tráfico mediante visión artificial basado en FPGA [11, p. 42].

Una de las propiedades de mayor valor en las imágenes digitales, es la resolución. La resolución indica la cantidad de píxeles que contendrá una imagen y representa la calidad de la misma, mientras más píxeles tenga una imagen, su calidad visual es mayor. La resolución está conformada por dos parámetros, el primer parámetro indica, cuantas columnas de píxeles (ancho de la imagen) y el segundo parámetro indica, cuantas filas de píxeles (alto de la imagen).

### 2.3.2. Imágenes de Color

Para fundamentar una imagen digital en color, es el mismo concepto descrito en el apartado 2.3.1, con la única diferencia que, los píxeles son codificados y descritos según el espacio de color que utiliza la imagen. Por ejemplo, para un espacio de color RGB, que generalmente es el más usado para representar imágenes, cada píxel representa un color creado a partir de ciertas cantidades de los colores rojo, verde y azul [12]. Esta representación se compone de tres matrices numéricas, una para cada canal de color cuyos valores son diferentes basados en el modelo de color que se utilice, como se muestra en la Figura 2.

En las imágenes RGB, cada píxel está conformado por un único valor de intensidad para cada color. La combinación de la intensidad de estos tres componentes, dará como resultado el color final dentro de este espacio de color. Así, el color blanco se formará al maximizar la intensidad de color de los componentes y, por el contrario, el color negro será el resultado de minimizar la intensidad de los componentes [11].

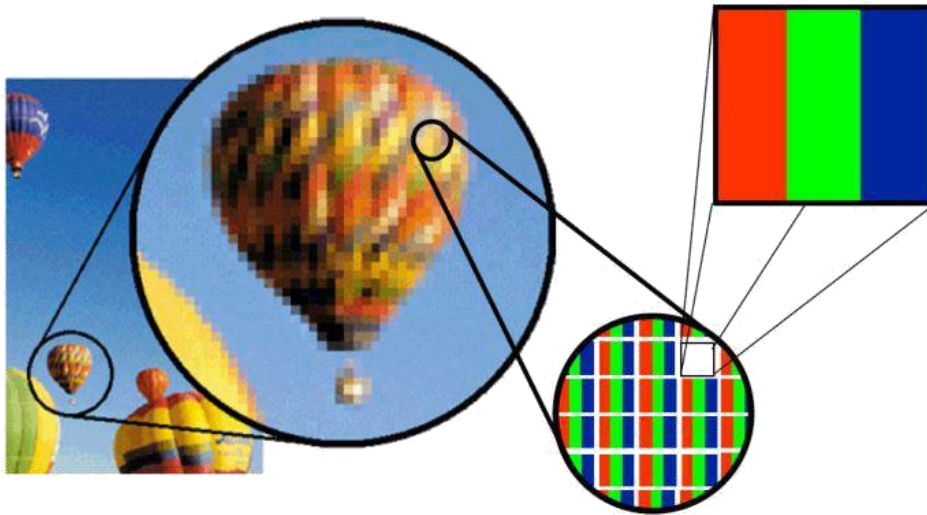


Figura 2: Composición de un píxel en una imagen a color.

Fuente: Implementación de un sistema de detección de señales de tráfico mediante visión artificial basado en FPGA [11, p. 43].

### 2.3.3. Modelos de Colores

Los modelos de colores son modelos matemáticos abstractos que permiten la representación de colores a través de números enteros, los cuales son de vital importancia en el procesamiento de imágenes digitales, debido a que nos permite enfocarnos de una manera distinta al analizar cada píxel, de tal modo que, podemos obtener la toda la totalidad de la información en la imagen. Los sistemas no lineales, trabajan constantemente en los espacios de color, debido a que estos buscan resaltar específicas peculiaridades de una imagen [13].

#### Modelo de colores: RGB

El modelo RGB, es el modelo que todos los sistemas informáticos utilizan para generar los colores en diferentes tipos de pantallas. Este modelo se basa en la “síntesis aditiva”, en ella se combinan las intensidades de los colores rojo, verde y azul para conseguir los distintos colores, incluyendo el negro y el blanco [14], como se muestra en la Figura 3.

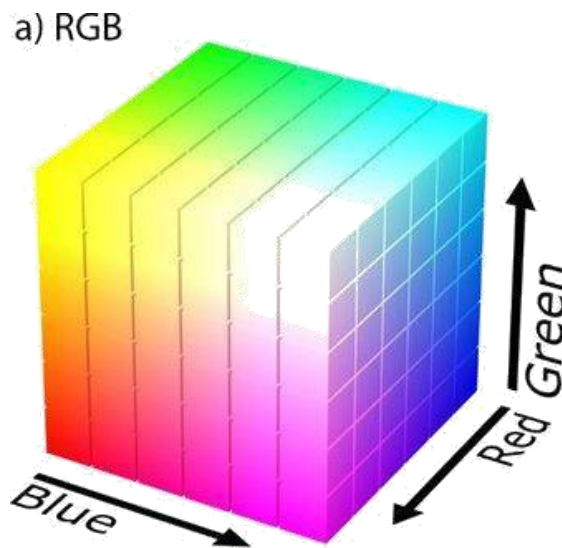


Figura 3: Representación gráfica del modelo RGB.

Fuente: Practices and pitfalls in inferring neural representations [14, p. 346].

En la Figura 3 podemos observar la representación gráfica del modelo RGB, la forma de este modelo lo convierte en un modelo relevante para el procesamiento de imágenes.

### **Modelo de colores: HSV**

A diferencia del modelo RGB, que trabaja con los colores rojo, verde y azul, este modelo utiliza el tono (*Hue*), saturación (*Saturation*) y valor (*value*) del color, respectivamente. En algunos casos también es llamado HSB, en donde el valor se reemplaza por el brillo (*brightness*).

En comparación al modelo RGB, la representación gráfica del modelo HSV es cilíndrico, y el subconjunto de este espacio donde se define el color, es una pirámide de base hexagonal. En el modelo HSV, los colores se obtienen desde el tono a partir

de 0° hasta los 360°, el brillo del color está dentro del área hexagonal desde 0 a 1 y la saturación, al igual que el brillo, se comprende en el eje de la pirámide hasta el final del área hexagonal, entre 0 y 1 [15], como se muestra en la Figura 4.

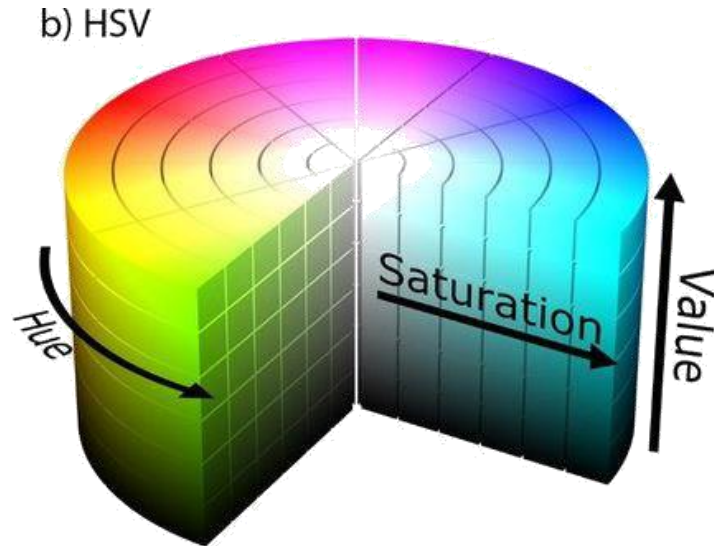


Figura 4: Representación gráfica del modelo HSV.

Fuente: Practices and pitfalls in inferring neural representations [14, p. 346].

Este espacio, se obtiene a partir de una transformación no lineal del espacio RGB, usando las siguientes relaciones demostradas en [15]:

$$H = \arccos \frac{\frac{1}{2}((R - G) + (R - B))}{\sqrt{(R - G)^2 + (R - B)(G - B)}} \quad (2.1)$$

$$S = 1 - 3 \frac{\min(R, G, B)}{R + G + B} \quad (2.2)$$

$$V = \frac{1}{3} (R + G + B) \quad (2.3)$$

Utilizando las relaciones (2.1), (2.2) y (2.3), se puede obtener una transformación no lineal del modelo de color RGB, y así obtener el tono, saturación y valor, respectivamente.

Las condiciones de luz en el modelo HSV es invariante, por lo que este modelo tiene una ventaja en comparación del modelo RGB, que depende fuertemente de la

intensidad; debido a esto, para trabajar una mejor umbralización, trabajaremos con el espacio de color HSV.

## **2.4. Procesamiento de Imágenes**

El procesamiento digital de imágenes, es el uso de algoritmos computacionales que toman una imagen como entrada y entregan una imagen como salida. El procesamiento de imágenes tiene como objetivo, mejorar el aspecto de las imágenes y enfatizar características que nosotros deseemos hacer notar [16].

Para poder resaltar una característica en particular desde una imagen, se requiere aplicar diversas operaciones a dicha imagen, para así adaptar la imagen a una aplicación en particular. Esto implica que el resultado del procesamiento, depende fuertemente del problema que se esté abordando [17].

El procesamiento de imágenes tiene un conjunto de algoritmos que están divididos en tres grupos:

- Algoritmos en el dominio espacial.
- Algoritmos en el dominio de frecuencia.
- Algoritmos de extracción de características.

A continuación, explicamos los algoritmos en el dominio espacial, que son los algoritmos abordados en nuestra investigación.

### **2.4.1. Algoritmos en el Dominio Espacial**

Son aquellos algoritmos que procesan píxel por píxel o tomando en cuenta un conjunto de píxeles vecinos de una imagen. Los píxeles vecinos, son aquellos píxeles que se encuentran contiguos si, y sólo si, tienen en común por lo menos uno de sus lados o esquinas [18].

Dentro de los algoritmos de dominio espacial encontramos tipos de transformaciones, que son detalladas en [11]:



- Las transformaciones puntuales, que son operaciones que dependen solamente del valor del píxel de entrada para obtener un píxel resultante, como por ejemplo, binarización, segmentación, entre otras.
- Las transformaciones locales, se basan en el píxel de entrada y la contribución de píxeles vecinos en la operación a realizar. Dentro de estas transformaciones encontramos las operaciones morfológicas, realce de bordes, filtros lineales, no lineales, etc.
- Las transformaciones geométricas, que se realizan tomando en cuenta la posición de los píxeles de la imagen, aplicando operaciones de rectificación, cambios de escala, rotación, traslación, entre otras.

### **Operaciones Morfológicas**

Dentro del procesamiento de imágenes, las operaciones morfológicas se aplican sobre imágenes binarias que han sido afectadas por una segmentación, separando los puntos de interés (normalmente identificado por el '1') sobre el fondo (normalmente identificado por el '0'). Una imagen binaria se puede interpretar como una matriz de dos dimensiones, donde cada posición representa un píxel con dos posibles valores, '1' o '0'. De este modo la identificación de estructuras es más sencilla.

Las operaciones morfológicas, son aplicadas a imágenes binarizadas enfocándose en los objetos de interés. Por lo general, tienen una imagen binaria como parámetro de entrada y salida. El valor de cada píxel en la imagen de salida se obtiene con operaciones no lineales sobre el píxel de entrada y sus vecinos. Estas operaciones no lineales se llevan a cabo a través de un componente llamado máscara, a la cual le damos una forma específica antes de operar sobre la imagen y nos permite obtener un resultado basado en el tipo de operación. Las operaciones morfológicas se usan para [11]:

- Supresión de ruidos.

- Simplificación de formas.
- Destacar la estructura de los objetos (detección de envolvente, ampliación, reducción).
- Descripción de objetos (área, perímetro).

Las operaciones morfológicas más usuales son la dilatación y la erosión, pero en esta investigación también haremos uso del gradiente morfológico y el rellenado, con el objetivo de simplificar la detección de la posible área de podado.

### **Dilatación**

Dada una imagen binaria  $A$ , y una máscara  $B$ , la dilatación de  $A$  por  $B$  se define como:

$$A \oplus B = \{x | (\hat{B})_x \cap A \neq \emptyset\} \quad (2.4)$$

Esto se entiende como todos píxeles  $x$  tales que la intersección de la máscara  $B$  situada en la posición  $x$  y la imagen  $A$  es diferente al conjunto vacío.

### **Erosión**

Dada una imagen binaria  $A$ , y una máscara  $B$ , la erosión de  $A$  por  $B$  se define como:

$$A \ominus B = \{x | B_x \subseteq A\} \quad (2.5)$$

Lo que se puede entender como todos los píxeles  $x$  tales que la estructura  $B$  situada sobre  $x$  pertenezcan en su totalidad a la imagen  $A$ . Por ende, todos los píxeles dentro de la máscara  $B$  situada en la posición  $x$  deben ser los objetos marcados con '1'.

### **Gradiente Morfológico**

Esta operación es muy importante para obtener los contornos de nuestros puntos de interés. Siendo de este modo que, para obtener un contorno externo, restamos la dilatación de una imagen  $A$  con la imagen original:

$$\text{Contorno}_{\text{externo}} = (A \oplus B) - A \quad (2.6)$$

Y para obtener un contorno interno, restamos la imagen original con la erosión de la misma:

$$\text{Contorno}_{\text{interno}} = A - (A \ominus B) \quad (2.7)$$

Por lo tanto, se define el gradiente morfológico de una imagen, como la resta entre la dilatación de una imagen  $A$  con la erosión de la misma imagen  $A$ :

$$Y = (A \oplus B) - (A \ominus B) \quad (2.8)$$

### **Relleno de Región**

Dado un píxel “semilla”, en una posición determinada, este se irá dilatando hasta encontrar un borde limitante.

$$X_k + 1 = (X_k \oplus B) \cdot \bar{E} \quad (2.9)$$

En esta operación es muy importante la posición de nuestro píxel “semilla”, porque si este se encuentra dentro de nuestro borde limitante, rellenará la región interna de nuestra silueta; en cambio si este píxel “semilla” se encuentra fuera de nuestro borde limitante, rellenará la región externa.

### **2.4.2. Algoritmos de Extracción de Características**

Los algoritmos de extracción de características, se enfocan en el análisis de las imágenes, para obtener atributos y regiones de interés, separación de objetos del fondo, detección de bordes o formas, entre otros [18]. A continuación, detallaremos los algoritmos más importantes dentro de este grupo, que intervienen en nuestra investigación.

### **Segmentación**

La segmentación permite subdividir una imagen en regiones u objetos constituyentes, de tal modo que, los píxeles subdivididos poseen propiedades o atributos similares.

La mayoría de algoritmos se basan en la similitud y la discontinuidad, ambas propiedades básicas de la imagen. Cuando se utilizan algoritmos basados en la similitud, se puede lograr una partición de la imagen en zonas que poseen características similares con base a criterios previamente definidos. Con respecto a la utilización de algoritmos basados en la intensidad, podemos dividir la imagen en zonas donde se detectan cambios abruptos en la intensidad, esto comúnmente se utiliza en la detección de bordes [19].

La segmentación se encarga de recorrer una imagen píxel por píxel y evalúa la información que esta contiene, es así como puede resaltar los píxeles con características que nos interesan. Al aplicar esta operación sobre una imagen de entrada, obtenemos como resultado una imagen binarizada, donde los píxeles que pertenecen a la región de interés son representados con el valor '1' y, contrariamente, son representados con '0' las del fondo [10]. Los algoritmos de segmentación se pueden dividir en [11]:

- Segmentación basada en características del píxel.
- Segmentación basada en transiciones.
- Segmentación basada en modelos.
- Segmentación basada en homogeneidad.
- Segmentación basada en morfología Matemática.

Los algoritmos de segmentación dependen estrictamente del objetivo que deseamos obtener a través de nuestra aplicación, así como también depende el tipo de imagen y sus características. Por lo tanto, es muy importante tener claro los objetos de interés y características que poseen.

## **2.5. Visión Artificial**

La visión artificial, también llamada visión por computador o visión computacional, es una inteligencia artificial con la capacidad para ver visualmente el entorno circundante en el que se encuentra, mediante el análisis y la extracción de información útil de una sola imagen o secuencia de imágenes. La visión artificial también puede convertir imágenes y videos en datos o información, que se modelan en patrones, predicciones y planes como conocimiento o sabiduría mediante técnicas de aprendizaje automático; esta estructura de la visión artificial depende del campo en la que se aplicaría y cómo se diseñe. Esto quiere decir que, la funcionalidad de este tipo de inteligencia artificial es afectada por los componentes y la organización de un sistema de visión artificial; un sistema de visión artificial se compone básicamente de un

objeto, adquisición de imágenes, procesamiento de imágenes, análisis de imágenes y finalmente la toma de decisiones como se puede ver en la Figura 5, esta estructura puede tener otros componentes como la múltivista y la base de conocimiento, que son agregados para obtener información más detallada para objetivos más complejos [20, 21].

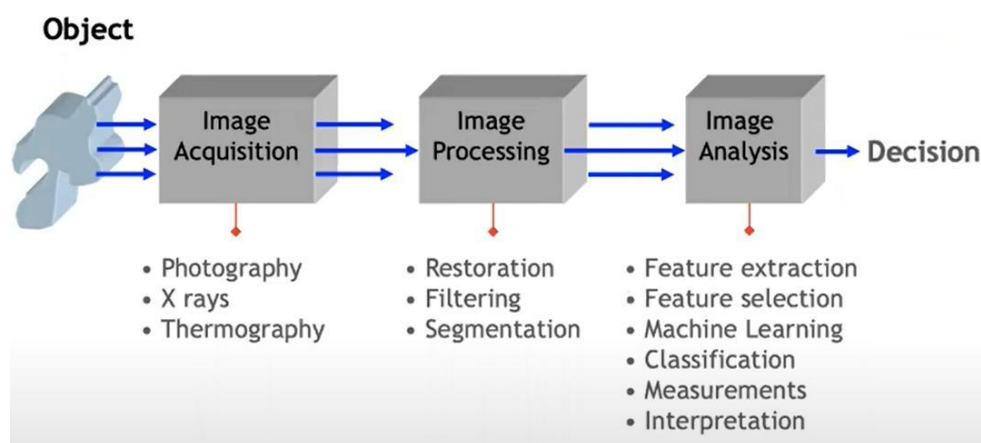


Figura 5: Estructura de sistema de visión artificial.

Fuente: Domingo Mery Curso Visión por Computador [22].

Donde:

- El objeto, representa lo que vamos a estudiar.
- Adquisición de imágenes, métodos con los cuales obtenemos la imagen o el conjunto de imágenes que contienen al objeto.
- Procesamiento de imágenes, algoritmos por los que logramos obtener el resalte de las características del objeto.
- Análisis de imágenes, algoritmos que, a través de una imagen de entrada, nos permite obtener una medición, decisión o interpretación.
- Toma de decisiones, es la información relevante que nos ayude a interpretar lo que pasa con el objeto.

## **2.6. Python**

Python, es un lenguaje de programación de código abierto, por lo que nos brinda capacidades como ser de uso gratuito, tiene una gran comunidad que está constantemente activa, desarrollando cada vez nuevas características y capacidades para los diferentes ámbitos en los que Python puede desarrollarse [23]. Siendo de este modo que, a través de Python, podemos utilizar la biblioteca Flask, la cual es un micro framework que nos permite realizar páginas web dinámicas bajo el Modelo Vista-Controlador (MVC).

### **2.6.1.Flask**

Flask, al ser un framework, puede darnos la capacidad de crear una página web “full stack”, esto quiere decir que podemos crear una página web completa, controlando simultáneamente la parte encargada de darle la lógica de funcionamiento a nuestra página web (lo que se conoce como BackEnd) y la parte en la que el usuario interactúa con nuestra página web (lo que se conoce como FrontEnd) [24].

Así mismo, Flask, puede trabajar conjuntamente con la biblioteca Flask-RESTful, la cual nos brinda la capacidad de integrar API REST a nuestra aplicación web para darle una mayor seguridad, estandarización y libertad de adaptarlo a diferentes entornos. Debido a que esta aplicación va a ser integrada a un sistema, API REST nos permite integrarlo con mayor facilidad y rapidez.

### **2.6.2.OpenCV**

OpenCV, es una biblioteca de código abierto que contiene diversas funcionalidades para apoyar a la resolución de tareas de visión artificial. En 1999, apareció por primera vez y desde entonces ha sido utilizado en múltiples aplicativos, siendo mencionada hasta el día de hoy, como la biblioteca más popular de visión artificial [25]. En esta investigación se hará uso de esta biblioteca, porque aplicaremos las funciones para el reconocimiento de objetos aplicados a la identificación de las áreas de podado en imágenes satelitales.

### **2.6.3. Scikit-Image**

Scikit-Image, es una biblioteca de procesamiento de imágenes, bajo la licencia BSD para el lenguaje de programación Python. Incluye algoritmos para segmentación, transformaciones geométricas, manipulación del espacio de color, análisis, filtrado, morfología, detección de características y otros algoritmos que nos permiten realizar un buen procesamiento de imagen. Está diseñado para operar internamente con las bibliotecas numéricas y científicas de Python, NumPy y SciPy, respectivamente [26].

### **2.6.4. NumPy**

NumPy, es una biblioteca para el lenguaje de programación Python, bajo la licencia BSD, que permite crear vectores y matrices multidimensionales de gran escala, junto con una gran colección de funciones matemáticas de alto nivel para operar con ellas. Gracias a sus características, esta biblioteca es ideal para poder manipular los datos de las matrices que representan las imágenes [27].

## **2.7. Interacción Persona-Computadora**

Interacción persona - computadora (HCI), estudia la interacción entre el usuario y una interfaz de usuario (UI). HCI es explícito o implícito. La HCI explícita maneja la manipulación directa de una interfaz por parte del usuario, como por ejemplo, al clicar en un botón en la interfaz gráfica. El sistema reacciona en función de la acción realizada directamente por el usuario. Por otro lado, el análisis de la información social y contextual, permite una interacción implícita entre el hombre y la computadora. Esto se hace principalmente, mediante el uso de los diferentes sensores con los que está equipado el dispositivo, como por ejemplo determinando la ubicación [28].

Alrededor de quince años atrás, la idea del diseño de la Interacción Persona - Computadora (HCI) fue dada por sentada bajo los conceptos del modelado de los sistemas, quienes se encargaban de cumplir las tareas requeridas, y el modelado de los usuarios, quienes especificaban si la interacción de estos con los sistemas, tenían

un comportamiento medible relacionado a la facilidad de uso, facilidad de aprendizaje y la eficiencia al desarrollar las tareas necesarias [29].

Conforme el tiempo ha ido transcurriendo, el usuario, quien era un factor para el desarrollo de un sistema, pasó a ser un actor dentro del modelado, lo cual generó un giro al concepto en lo que a “usuario” se refiere, siendo de esta manera, que más allá de buscar el desarrollo de tareas en un sistema, se añade el diseño para la conformidad del usuario, debido a que ya no era un engranaje más como parte del sistema, sino se convirtió en un consumidor de nuestro sistema [29].

### **2.7.1. Interfaz de Usuario**

Se entiende como interfaz de usuario, a la interacción de comunicación entre un humano y una computadora, mediante un dispositivo. También se considera como la manera en la que un usuario interactúa con un aplicativo o página web. La creciente dependencia de muchas empresas, por las aplicaciones web y las aplicaciones móviles, han llevado a muchas empresas a dar mayor prioridad a la interfaz de usuario, en un esfuerzo por mejorar la experiencia general del mismo [30].

### **2.7.2. Interfaz Gráfica de Usuario**

Interfaz gráfica de usuario (*GUI*), acrónimo en inglés de *Graphical User Interface*. La interfaz gráfica de usuario, es un programa o entorno que gestiona la interacción con el usuario, basándose en relaciones visuales como iconos, menús o un puntero [30].

### **2.7.3. Experiencia de Usuario**

La experiencia de usuario (*UX*), es el proceso mediante el cual un usuario interactúa con un entorno o dispositivo concreto, resultando en una percepción positiva o negativa sobre dicho entorno. Esta percepción depende de todos los elementos que intervienen en toda la interacción del usuario, desde los dispositivos que usa hasta las emociones que, las características de este último, puedan transmitir al usuario. “La experiencia usuario, por su enfoque en la relación sistema hombre-máquina, aparece como una evolución de la ergonomía (física y psicológica) y mantiene al



diseño en el papel protagónico de la búsqueda incansable de satisfacción de las experiencias” [31].

La experiencia de usuario, también puede entenderse como un conjunto de factores y elementos, que permiten lograr la generación de un producto o servicio de forma satisfactoria para el usuario, ya sea desde componentes de hardware hasta sistemas informáticos. Este concepto se desarrolló a partir de los años 90's en Silicon Valley, en las mejores compañías de software y poco a poco definió un proceso importante para lograr su objetivo [32]:

- Definir.
- Investigar.
- Ideación.
- Prototipar.
- Elegir.
- Implementar.
- Aprender.

Como podemos ver, la experiencia de usuario desarrolla su objetivo a corto plazo para lograr una primera satisfacción con el usuario, pero con el tiempo, el usuario y el producto evolucionan y de esta manera el UX deberá poder ser medible para verificar nuevamente la satisfacción y, en caso de ser necesario, evolucionar este UX con base a las nuevas necesidades de nuestro producto y/o usuario [33].

## **2.8. Robots Móviles**

Se define a los robots móviles como aquellos dispositivos que tienen como objetivo principal restringir, en la medida de lo posible, la interacción humana en la resolución de tareas previamente realizadas por humanos; estos sufren debido al problema de la toma de decisiones, para lo cual se le integran diversos tipos de sensores que cubran las necesidades, según la finalidad del robot [4].

## **CAPÍTULO 3**

### **ESTADO DEL ARTE**

En este capítulo, se mencionan antecedentes e investigaciones relevantes relacionadas a esta investigación, para una mejor comprensión de los temas abordados, las investigaciones de cómo se han resuelto y los enfoques que se toman para los diversos problemas similares al de esta investigación.

#### **3.1. Interacción Persona - Computadora**

Jakob Nielsen [34], en 1993 publicó un artículo donde explica los beneficios que trae consigo aplicar la iteratividad en el diseño de una interfaz de usuario, siendo uno de estos beneficios, la corrección de errores o defectos que pueda tener dicha interfaz de usuario en su etapa de desarrollo, así como mejorar la usabilidad del mismo por parte de los usuarios. Estos beneficios fueron demostrados a través de pruebas, en donde el usuario hacía uso del sistema y calificaba los diferentes aspectos del mismo, a través de preguntas cuantitativas, lo cual permitió demostrar lo importante que era el involucrar al usuario final en el diseño de una interfaz de usuario.

Así mismo, en el 2006 A. Dillon [35] concluyó que, el diseño de la interfaz de usuario es un proceso complicado, debido a que requiere un análisis detallado del desempeño y preferencias humanas. Además de la comprensión de aspectos emocionales que para ese entonces, no habían sido estudiados por los científicos cognitivos, pero que

en un futuro serían importantes para darle al usuario la facilidad de interacción con estas y nuevas tecnologías que se desarrollen.

S. Sridevi, en [36], enfatiza lo que denomina como “The golden rules”, que son una serie de principios que forman las bases para el diseño de la interfaz de usuario como una guía importante para el diseño de software. Estos principios se enfocan en desarrollar una interfaz gráfica consistente y que tenga una buena usabilidad para el usuario bajo ciertos parámetros como, intuitividad, interactiva, sencillez, que cumpla con las expectativas, entre otras. Para llegar al correcto desarrollo de estos parámetros, propone un modelo espiral el cual analiza, valida, implementa y genera de forma recurrente el diseño, para lograr la interfaz adecuada que brinde la sinergia entre usuario y funcionalidades.

S. Raheel [28], demostró que la integración de una interfaz adaptativa de usuario, puede obtener mejores resultados conforme a la usabilidad, aceptación y la experiencia de usuario. Mediante las funciones adaptativas y mecanismos de adaptabilidad, se presentan grandes ventajas y una personalización adaptable exitosa para los diferentes usuarios.

M. Hashemi y J. Herbet, en [37], propusieron un marco de trabajo para reconocer los comportamientos de los usuarios, mientras interactúan con una interfaz de usuario, obteniendo como resultado que, el seguimiento de las actividades de los usuarios de la interfaz de usuario, no solo está relacionado con la calidad de los elementos de esta interfaz, sino que también se puede considerar como la calidad de software en sí y el rendimiento que proporciona. El análisis de la actividad del usuario en tiempo real, también puede brindar la oportunidad de ajustar la calidad del servicio en función de las necesidades de los usuarios al predecirlas. También cambiar los elementos de la interfaz de usuario, en función de lo que el usuario necesita e interactúa más; ocultar aquellas partes que un usuario no suele usar, también podría ayudar a tener una mejor experiencia con la interfaz de usuario.

C. Córdoba, en su investigación doctoral [38], presenta un modelo de Experiencia de Usuario Extendido, basado en tres tipos de experiencias: estética, significativa y afectiva. Con este modelo pudo comprobar, con ecuaciones estructurales por medio de modelos anidados y componentes jerárquicos, que la aceptación tecnológica por parte de los usuarios, se puede obtener a través de la experiencia de usuario que ofrezca en el sistema, mediante características como: confianza, calidad, eficacia, eficiencia, entre otras.

M. Hassenzahl y N. Tractinsky [39], en el 2006 presentaron un trabajo que contenía la recopilación de diversos artículos basados en la experiencia de usuario (UX), con la finalidad de enriquecer la definición y entendimiento del UX, debido a que la experiencia de usuario se tomaba más como un parámetro secundario en el desarrollo de un sistema y no como un apartado que puede generar riqueza en nuestro sistema, ya que el usuario finalmente lo consume.

Hussein, Idyawati and Mahmud, Murni and Tap, Abu Osman Md, en [40], demostraron, con una encuesta a profesionales instruidos en HCI y la relación entre HCI y el desarrollo de proyectos web, que los conocimientos y la aplicación de la HCI en proyectos de desarrollo web, pueden fortalecer la política de integración de los principios de UX, para lograr diseñar tecnologías informáticas más útiles y agradables para el usuario.

Adikari, y sus colegas en [41], destacan un nuevo enfoque contextual del desarrollo ágil del UX, teniendo como antecedente la literatura de la HCI, en la cual se muestra el amplio interés de la integración del UX, el desarrollo ágil de software y el pensamiento del diseño. Estos tres enfoques de diseño diferentes del marco, se complementan entre sí, para beneficiar la derivación efectiva de los requisitos contextuales que incluyen la funcionalidad del sistema, así como aspectos de la experiencia total del usuario, basados en el entendimiento compartido, obtenido de las partes interesadas en el contexto.

### **3.2. Robot GrassBot**

La empresa Opciones Ingeniería y Tecnología Ambiental, conjuntamente con Innovate Perú y la Universidad Tecnológica del Perú, han desarrollado un robot podador autómatas que puede realizar un excelente trabajo de podado de césped. Este robot fue elaborado con el objetivo de proveer ventajas, como la reducción de la huella de carbono e hídrica, disminución de costos y monitoreo del terreno. El robot se comunica a través de internet, con una aplicación web que nos muestra toda la información que el robot va recopilando, mientras realiza la tarea de podado de césped [2].

### **3.3. Visión Artificial**

Jinsong Deng, y sus colegas, en [42], desarrollaron una metodología para monitorear la expansión urbana y el cambio de espacios verdes en la metrópolis central de Yangtze desde 1996 al 2016. Para lograr su objetivo, tuvieron que hacer un gran esfuerzo en la detección del cambio sobre el terreno no urbano, a través de imágenes satelitales, para posteriormente realizar una clasificación híbrida con la cual se pueden obtener resultados con una gran precisión sobre este cambio, para así poder tomar decisiones importantes requeridas.

En el 2012, Himan Shahabi, Hasan Zabihian y Afsaneh Shikhi, trabajaron en una aplicación para detectar la devastación de áreas verdes por construcciones urbanas en la ciudad de Boukan, a través de imágenes satelitales dentro de los años 1991 a 2008. Utilizando la comparación NDVI, el análisis de componentes principales y la clasificación de puestos, lograron obtener resultados que indicaban una devastación poco mayor al 50 % de las áreas verdes desde 1991 hasta el 2008, destacando el NDVI por devolver datos más precisos [43].

En el libro [20], publicado en el 2019 por Mohammad Ali Nematollahi, y sus colegas, nos explican conceptos, aplicaciones y técnicas de alto nivel para la aplicación de visión y audición por computador, para lograr un futuro análisis urbano inteligente. Dentro de estos conocimientos, encontramos un gran detalle de diversos métodos para la detección de verdes, de agua, de redes de caminos, entre otros, en imágenes satelitales e imágenes panorámicas a nivel de calle.

Nicolás Dobernack, en [11], implementó un sistema de detección de señales de tráfico, mediante visión artificial en el 2013, detallando las diversas técnicas que se utilizan para el procesamiento de imágenes y video, que aplica a través de un sensor de imagen con el cual puede aplicar la visión artificial con gran porcentaje de acierto, para la detección de las más de 15 señales de tráfico utilizadas.

En [44], podemos encontrar las diversas técnicas que aplicó Christina Ludwig, para el mapeo de espacios verdes urbanos públicos, en el área central de Dresden. Uno de sus dos grandes problemas, fue poder distinguir los espacios públicos de los privados, para lo cual aplicó un modelo jerárquico bayesiano y datos de OpenStreetMap; y para cubrir el otro gran problema, sobre la detección de pequeñas áreas de vegetación, fusionó el NVDI con la teoría de Dempster-Shafer, sobre los datos de OpenStreetMap. Logrando una precisión final, para la detección de espacios públicos, de un 95 %.

Isabel Riomoros, en [45], realiza una investigación para, la segmentación automática de texturas en imágenes agrícolas, en el cual demuestra los diferentes resultados que obtenemos al aplicar diversas técnicas de segmentación, para colores y texturas de la vegetación. Con estos resultados, podemos analizar los diferentes márgenes de éxito y error, para los diferentes casos de detección en las imágenes, y así poder encontrar una técnica apropiada para las diferentes necesidades que tengamos.

## **CAPÍTULO 4**

### **METODOLOGÍA DE LA INVESTIGACIÓN**

En este capítulo se desarrolla la metodología de esta investigación, así como la definición de variables e indicadores.

Esta investigación, seguirá el proceso metodológico que presenta Sebastián Sastoque, Cristian Narváez y Germán Garnica, en el artículo [46], sobre una metodología para la construcción de interfaces gráficas centradas en el usuario, la cual está conformada por las siguientes fases:

- Fase de estructuración, en esta fase se establecen los procesos y elementos de trabajo.
- Fase de reconocimiento, donde se estudian las necesidades y el entorno.
- Fase de exploración, aquí analizaremos las actividades que realiza el usuario.
- Fase de modelado, consiste en construir un modelo de interfaz, basándose en el procesamiento de la información recopilada.
- Fase de ideación y prototipado, donde se construye el prototipo ideal de interfaz.
- Fase de formalización, donde se formaliza el diseño final de la interfaz.
- Fase de implementación, en esta fase se implementa la solución final.
- Fase de validación en contexto, donde validamos la interfaz de usuario.
- Fase de despliegue, donde desplegamos e integramos la interfaz a la solución final.

#### **4.1. Tipo y Método de la Investigación**

El tipo de esta investigación es aplicada, porque se estudiarán las diferentes maneras de procesar los datos de un sistema geodésico mundial, para así poder aplicar la visión artificial, con la finalidad de detectar áreas de podado con base a una posición dentro de este sistema geodésico, y de este modo poder integrar dicho resultado en la interfaz gráfica desarrollada.

Hernández, Fernández y Baptista, en [47], nos indican que, el método para esta investigación sería hipotético deductivo, debido a que comenzaremos desarrollando una interfaz que podrá automatizar la tarea de definir un área de podado de césped, basándose en la geoposición del robot GrassBot.

#### **4.2. Diseño de la Investigación**

Hernández, Fernández y Baptista, en [47] y Kerlinger, en [48], nos indican que, el enfoque de esta investigación es mixta, porque tendrá una variable cuantitativa experimental, en la cual realizaremos experimentos que nos permitirán medir el porcentaje de acierto de nuestra detección de área verde, y una variable cualitativa de tipo investigación-acción, donde desarrollaremos una interfaz gráfica, que será retroalimentada a través del operario del sistema hasta lograr la interfaz gráfica deseada, esta retroalimentación será hecha con base a entrevistas.

Además, posee un alcance descriptivo, debido a que proponemos una solución utilizando herramientas tecnológicas, que permitan mejorar el proceso de definición de áreas de podado de césped desde una interfaz gráfica para un robot podador autónomo.

#### **4.3. Técnicas e instrumentos de colecta de datos**

##### **4.3.1. Técnicas**

Para la variable cualitativa, utilizaremos la encuesta.



Para nuestra variable cuantitativa, utilizaremos técnicas de ubicación geográfica global y las técnicas de visión artificial.

#### **4.3.2. Instrumentos**

Los instrumentos utilizados en esta investigación son:

- El cuestionario, usando la escala de Likert para la medición respectiva.
- La base de datos del sistema geodésico EPSG:4326 WGS 84 y EPSG:900913 Google Maps Global Mercator.

#### **4.3.3. Población**

Nuestra población, es toda aquella área verde de gran extensión donde el robot GrassBot puede aplicarse, debido al corte específico que este proporciona. Estas áreas pueden ser áreas verdes de clubes, parques cementerio, canchas de fútbol y áreas con césped ornamental, dentro de la provincia de Arequipa.

#### **4.3.4. Muestra**

Para esta investigación, se tomó la geoposición 10 zonas conformadas por 1 parque cementerio, 1 área verde de un club, 1 área de césped ornamental y 7 canchas de fútbol de la provincia de Arequipa.

### **4.4. Estudio de Casos**

Se contará con el apoyo de la empresa Opciones Ingeniería y Tecnología Ambiental S.R.L., para hacer las validaciones de los indicadores respectivos al desarrollo del prototipo de la interfaz gráfica, en relación a la experiencia de usuario. Del mismo modo, se contará con el acceso a una API, que nos brinda información del robot GrassBot y las diferentes geolocalizaciones que serán usadas para calcular el porcentaje de acierto del sistema de visión artificial, para la detección de áreas de podado.

#### 4.5. Operacionalización de Variables

VARIABLES	DIMENSIONES	INDICADORES
Interfaz Gráfica	Experiencia de Usuario	Simplicidad
		Comunicación
		Estructura
		Navegación
		Utilidad
Visión Artificial para Detección de Áreas de Podado	Métrica en la Base de Datos de sistemas geodésicos	Acierto

Tabla 1:Tabla de variables.

Fuente: Autoría Propia

## **CAPÍTULO 5**

### **DESARROLLO DE LA TESIS**

En este capítulo se realizará el desarrollo de la interfaz gráfica, basándose en la metodología de Sebastián Sastoque, Cristian Narváez y Germán Garnica, en [46], que consta de las fases de estructuración, reconocimiento, exploración, modelado, ideación y prototipado, formalización, implementación, validación en contexto y despliegue, que serán detalladas a profundidad. Del mismo modo, se realizará la implementación del sistema de visión artificial, enfocado en el procesamiento y análisis de imágenes satelitales con base a una geoposición, para ello tendremos como primer paso, la obtención de una imagen satelital que se basará en la geoposición del robot GrassBot, a través de la latitud y longitud que el sensor de éste último transmita, seguidamente, realizaremos el preprocesamiento y segmentación de la imagen, para detectar si en la ubicación del robot, existe un área de podado, y de ser el caso que se detecte un área, se calcularán los píxeles perimetrales de dicha área, a continuación, se analizará la conversión de estos píxeles perimetrales en datos de latitud y longitud, que son necesarios para el funcionamiento del robot, y a su vez serán plasmados en la API de Google Maps para la representación adecuada.

## 5.1. Desarrollo de la Interfaz Gráfica

Como se ha explicado en el alcance de esta investigación, en el capítulo 1, el desarrollo de esta interfaz gráfica, debe integrarse al sistema que se ha desarrollado por parte del equipo técnico de la empresa Opciones, de tal modo se entiende que, el trabajo de ambas partes es independiente, es por esto que se tomaron en cuenta las características de dicho sistema informático, con el fin de desarrollar una interfaz gráfica que tenga un emparejamiento, lo más cercano posible, con el sistema ya desarrollado y conserve la experiencia de usuario para el operario del sistema. Debido a que el desarrollo de la interfaz gráfica para mostrar gráficamente el área de podado es independiente, se solicitó al equipo técnico de desarrollo de la empresa Opciones, implementar una API que nos permita obtener la estructura de los datos que se manejarán para el envío y recepción de información, es por esto que se planteó una arquitectura MVC de tipo controlador mediador, como se muestra en la Figura 6, para el desarrollo de esta interfaz gráfica que, según nos explica James Bucanek, en [49], es un patrón importante que puede aplicarse tanto en sistemas completos como en aplicaciones aisladas y que nos permite un manejo de información transparente a través de una API, entre nuestra interfaz gráfica y el sistema informático de la empresa Opciones.



Figura 6: Patrón de diseño modelo-vista-controlador mediado.

Fuente: ModelViewController Pattern [49, p. 355].

En vista de que nuestra interfaz gráfica, se integrará a un sistema informático web para control y monitoreo interno del robot autónoma GrassBot, nuestra interfaz gráfica debe enfocarse en la experiencia de usuario, para que el operario del sistema tenga

un desempeño fluido y práctico; es por esto que aplicamos la metodología indicada en el capítulo 4, que consta de las fases que se muestran en la siguiente imagen:

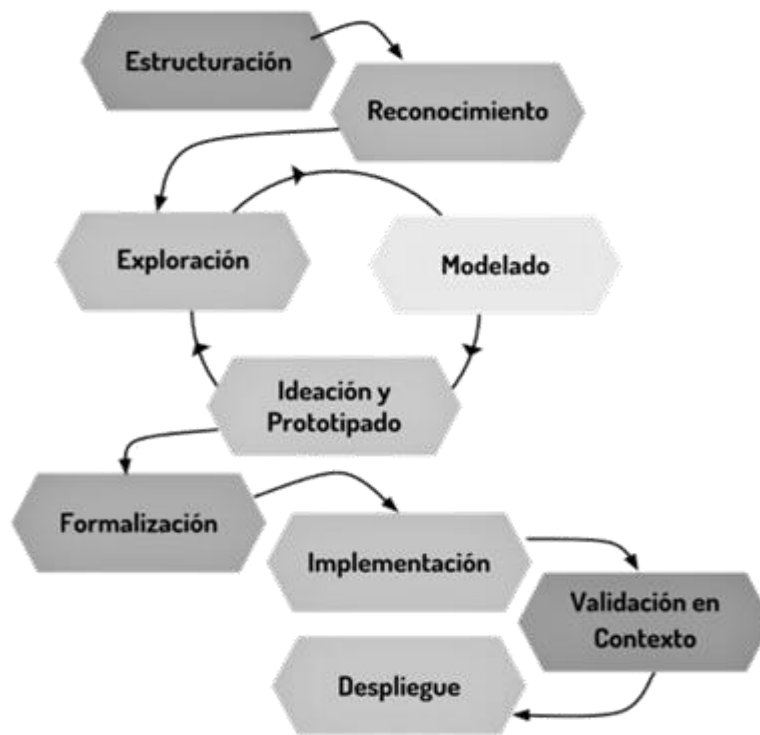


Figura 7: Fases de la metodología utilizada.

Fuente: Metodología para la construcción de Interfaces Gráficas Centradas en el Usuario

[46, p. 320]

### 5.1.1. Estructuración

En esta fase inicial, establecemos los procesos de gestión y los elementos de trabajo, que conformarán las bases para la ejecución de las diferentes fases de la metodología, los cuales son descritos en las Tablas 2 y 3:

ESTRUCTURA DEL TRABAJO				
TAREA	TIEMPO	EQUIPO TRABAJO	RECURSOS	ESPACIO TRABAJO
Automatizar el proceso y facilitar las operaciones del mantenimiento del césped	6 meses	Leandro Gustavo Ortega Palacios, Alberth Ronal Tamo Calla, Danitza Aruquipa	Computador Portátil y conexión a Internet	Home Office

Tabla 2: Estructura del trabajo.

Fuente: Autoría Propia.

ESPECIFICACIONES DEL SUBSISTEMA				
ACTIVIDAD	MIEMBROS	LENGUAJES	FRAMEWORK	PROPS.
Diseño y desarrollo de los módulos Front-End con enfoque en el UX	Leandro Gustavo Ortega Palacios, Alberth Ronal Tamo Calla, Danitza Aruquipa	HTML5 CSS3 JavaScript Python	Flask	UX simple Amigable Ligero Dinamico Multiplataforma

Tabla 3: Especificaciones del subsistema.

Fuente: Autoría Propia

### 5.1.2.Reconocimiento

Como siguiente fase de esta metodología, debemos identificar al usuario encargado que operará nuestra interfaz gráfica, siendo la responsable del desarrollo del sistema informático, la ingeniera Danitza Aruquipa, con quien realizamos la identificación de la necesidad de una interfaz gráfica, con la capacidad de poder definir un área de podado para el robot autómatas GrassBot, con herramientas que permitan realizar trazos sobre un mapa, con el objetivo de almacenar y mostrar la información gráfica de dicha área de podado para luego, en una etapa posterior del proyecto, delimitar al robot a través de esta información generada.

### **5.1.3.Exploración**

Al llegar a esta fase, hemos conocido al usuario operario, teóricamente a través de sus necesidades y problemática, pero también necesitamos conocer el punto de vista, ideas, sugerencias y lo que espera el usuario con respecto a la interfaz gráfica, es por esto que recopilamos las siguientes consideraciones:

- Previa visualización de la geoposición de los robots.
- Muestreo de la información de los diferentes sensores del robot.
- Definición del área de podado en un mapa de manera rápida y sencilla.
- Se debe poder distinguir la información entre los diferentes robots.

### **5.1.4.Modelado**

En esta etapa, realizamos el procesamiento y síntesis de la información recopilada, para realizar el levantamiento de los diferentes requerimientos y diagramas que nos permitirán modelar la interfaz gráfica, a través de bocetos.

#### **Requerimientos Funcionales**

- La interfaz gráfica debe permitir visualizar un listado de los robots disponibles.
- La interfaz gráfica debe mostrar una vista previa de la ubicación del robot seleccionado en un mapa.
- La interfaz gráfica debe permitir visualizar la información de los diferentes sensores que el robot registra en la API de la nube.
- La interfaz gráfica debe tener herramientas de dibujo sobre un mapa para indicar el área de podado.
- La interfaz gráfica debe tener una herramienta que aplica visión artificial para la detección de áreas de podado.
- La interfaz gráfica debe permitir guardar la información del área de podado dibujada en el mapa.

#### **Requerimientos No Funcionales**

- La interfaz gráfica será desarrollada en la arquitectura MVC.

- La interfaz gráfica debe tener UX simple.
- La interfaz gráfica debe tener las características de simplicidad, comunicación, estructura, navegabilidad y utilidad.
- La interfaz gráfica debe ser amigable, ligera, dinámica y multiplataforma.
- La interfaz gráfica está implementada en HTML5, CSS3, JavaScript y Python.
- La interfaz gráfica debe funcionar con CentOS 7.
- El sistema de visión artificial debe estar implementado en Python.
- El sistema de visión artificial debe trabajar con una API gratuita que nos permita obtener mapas (API Maps JavaScript de Google).
- El sistema de visión artificial debe trabajar con una API gratuita que nos permita obtener imágenes estáticas de los mapas (API Maps Static de Google).

### **Diagrama de Casos de Uso**

En las Figura 8, 9 y 10, se modeló el diagrama de caso de uso de la interfaz gráfica, donde el usuario hace la interacción con esta interfaz, empezando con la selección de un robot, para visualizar su información correspondiente a ubicación y sensores, así como también, tiene un botón que lo enviará a un vista donde podrá hacer uso de las herramientas de dibujo para determinar el área de podado sobre un mapa, del mismo modo, tendrá una herramienta para detectar un área de podado mediante visión artificial.



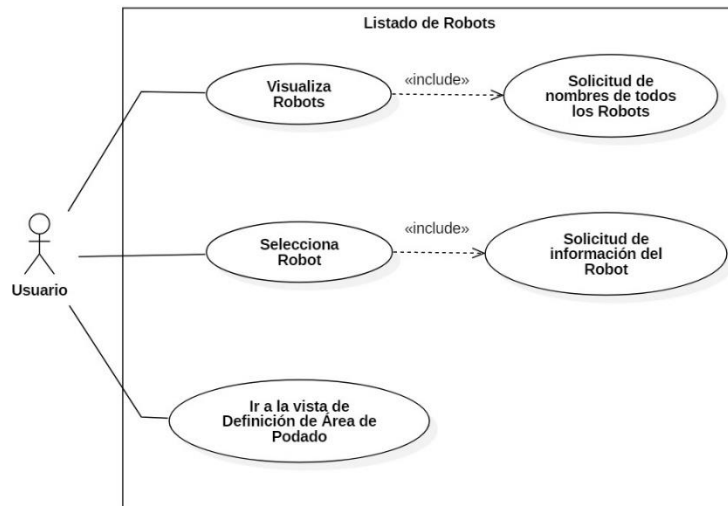


Figura 8: Diagrama de casos de uso – Listado de robots.

Fuente: Autoría Propia.

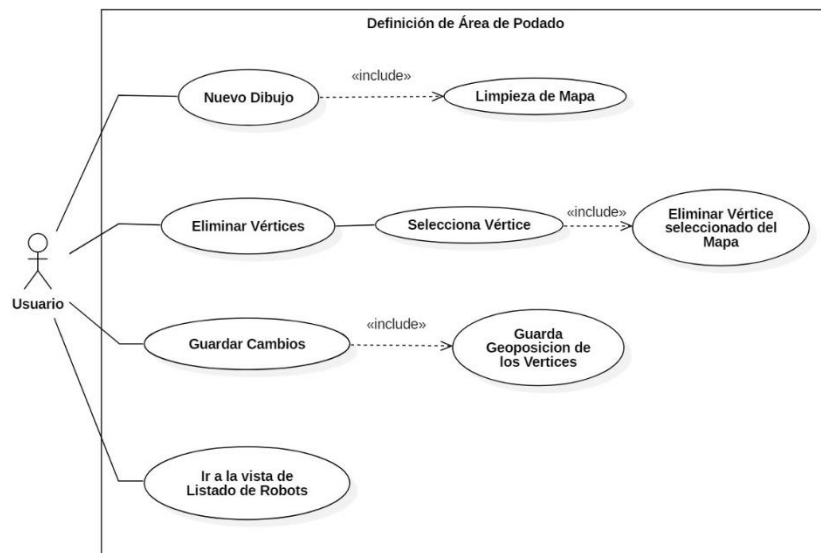


Figura 9: Diagrama de casos de uso – Definición de área de podado.

Fuente: Autoría propia

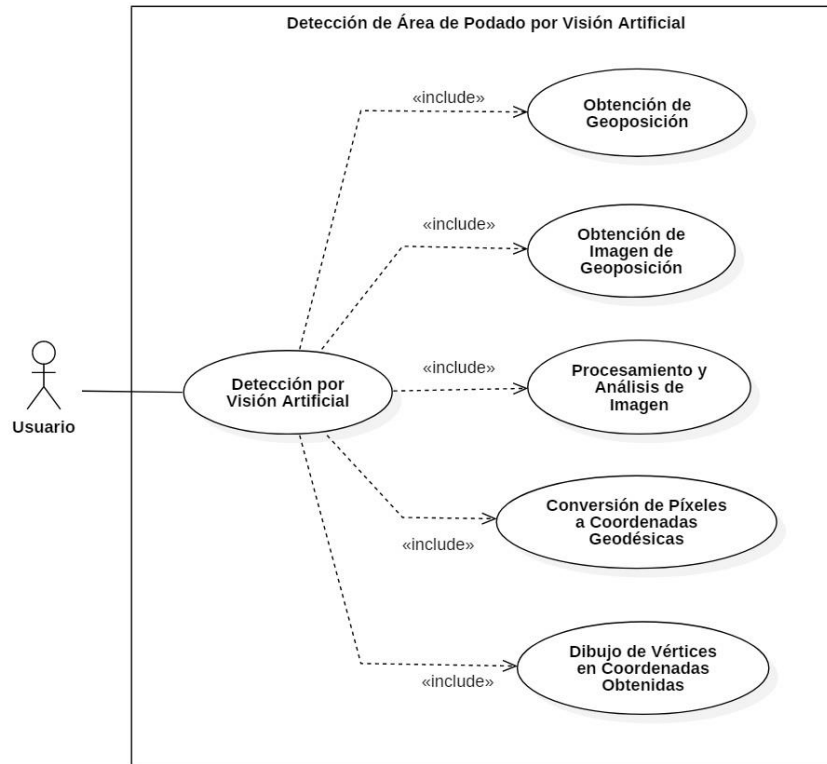


Figura 10: Diagrama de casos de uso - Definición de área de podado por visión artificial.

Fuente: Autoría propia.

### Diagrama de Secuencia

En la Figura 11, se presenta el diagrama de secuencia de la vista de listado de robots de la interfaz gráfica, en el cual podemos observar el proceso en el que el usuario solicita la información de la lista de robots, y es solicitada a la API de Opciones a través del controlador, para ser visualizados desde la interfaz, seguidamente el usuario podrá seleccionar un robot de la lista y se ejecutará la acción de solicitud de información a la API de Opciones, basándose en el identificador (ID) del robot seleccionado. La API de Opciones retorna el conjunto de datos al controlador, y este a la vista para que sea mostrada al usuario y, a su vez, se le habilite el botón para direccionarse al área de dibujo de área de podado.

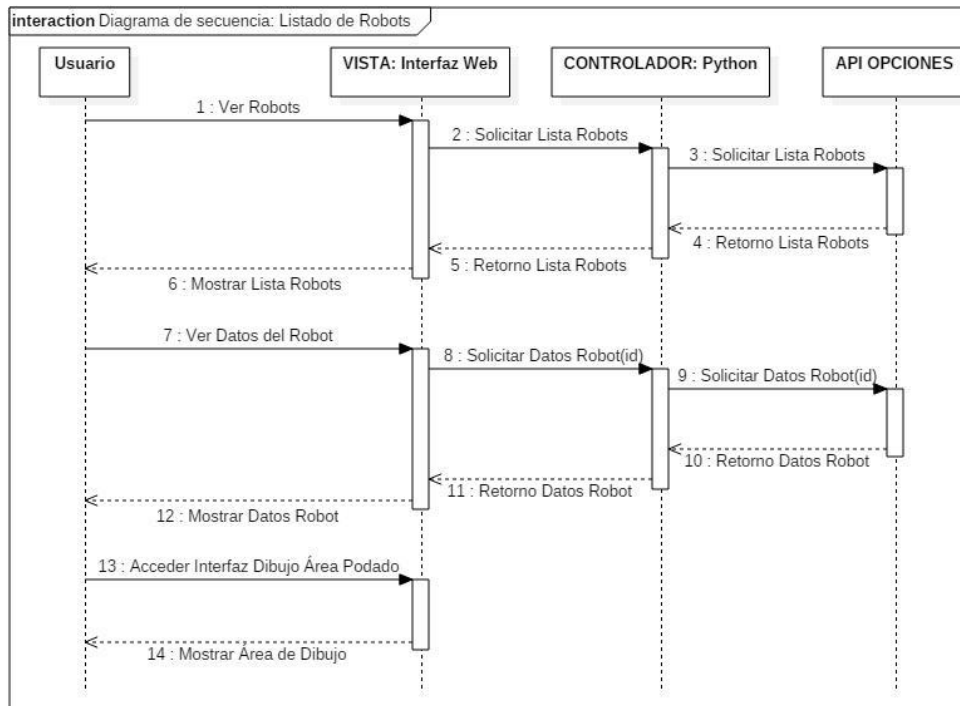


Figura 11: Diagrama de secuencia – Listado de robots.

Fuente: Autoría Propia.

En la Figura 12, se muestra el proceso de interacción entre el usuario y la vista de la interfaz gráfica, para poder realizar la tarea de definir el área de podado con las diferentes herramientas de dibujo, que le permitirán al usuario dibujar, eliminar y editar vértices, de tal manera que, pueda tener un desempeño ordenado y con la mayor precisión posible. Finalmente, cuando se haya dibujado el polígono que indicará el área de podado correspondiente, podrá hacer uso de la herramienta de guardado, que realiza una comunicación desde la vista hasta la API de Opciones, a través del controlador, indicando el ID del robot seleccionado en la vista del listado de robots, para poder almacenar esta información y retornarlo a la vista de listado de robots con una respuesta exitosa de guardado.

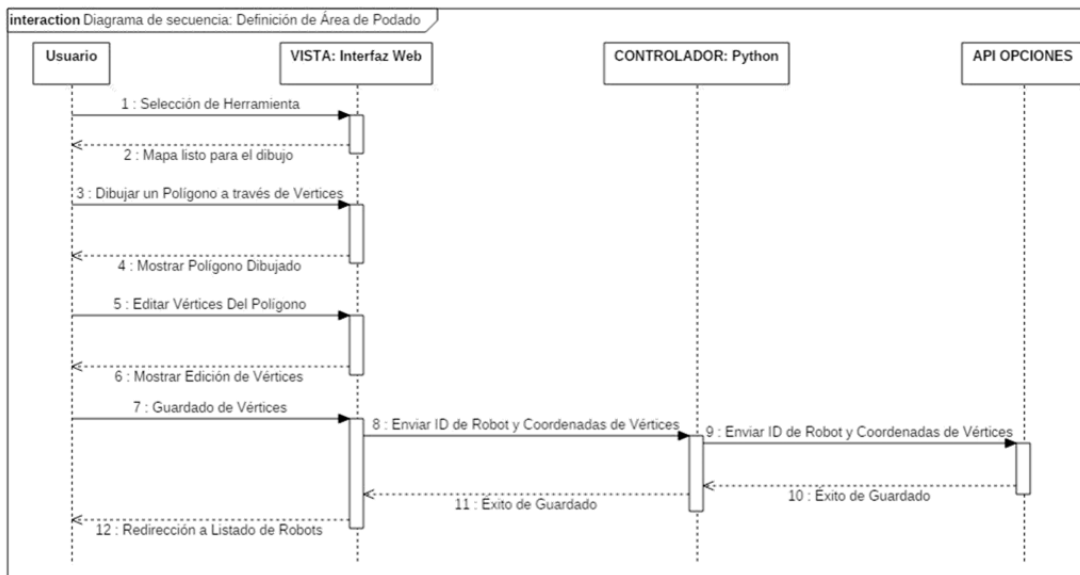


Figura 12: Diagrama de secuencia – Definición de área de podado.

Fuente: Autoría Propia.

En la Figura 13, se presenta el proceso del uso de la herramienta de detección de área de podado por visión artificial, en el cual el controlador primero obtiene la posición del robot en latitud y longitud, para posteriormente solicitar una imagen de mapa a través de la posición del robot a la API de Google, esta última retorna dicha imagen al controlador, para así realizar el procesamiento y análisis de la imagen, extraer los píxeles delimitadores del área de podado detectada y convertirlos a latitud y longitud, para enviar todas estas coordenadas a la vista y representarlos en el mapa que visualiza el usuario.

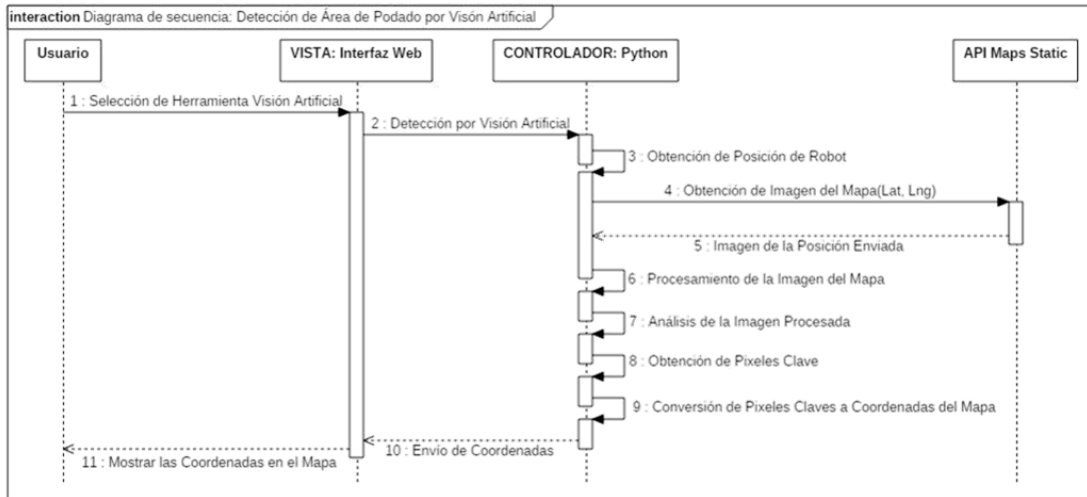


Figura 13: Diagrama de secuencia - Detección de área de podado por visión artificial.

Fuente: Autoría Propia.

### Diagrama de Clases

En la Figura 14, se presenta un diagrama donde se detallan las clases correspondientes al paquete del modelo, y los atributos y métodos respectivos de cada uno de ellos.

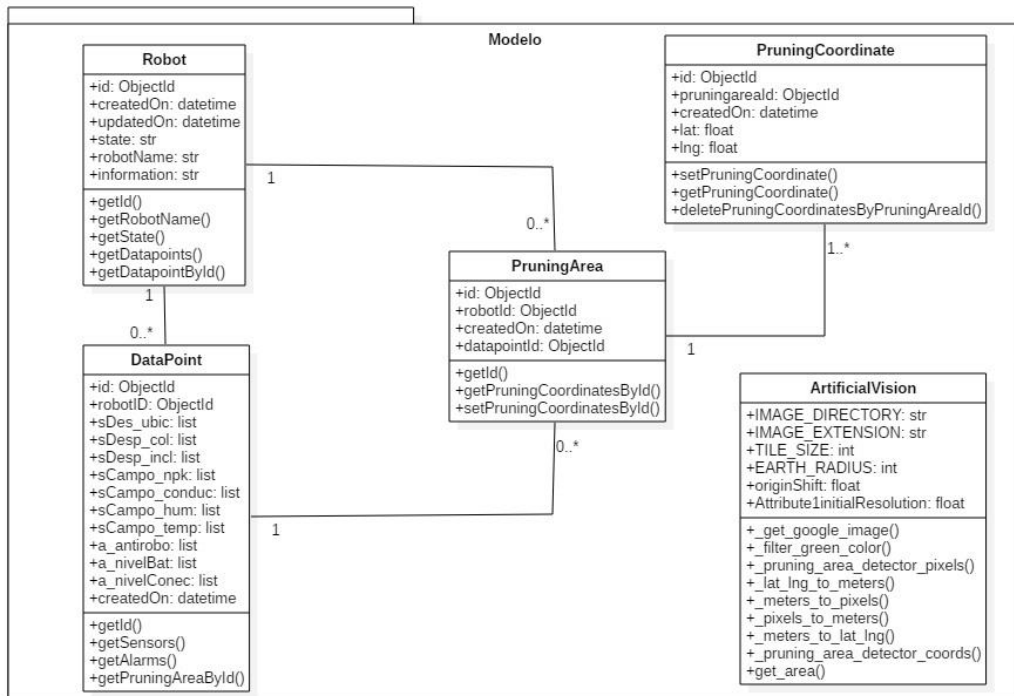


Figura 14: Diagrama de clases - Modelo.

Fuente: Autoría Propia.

En la Figura 15, se muestra el diagrama de clases correspondientes al paquete del controlador, conformado por el controlador de la vista del listado de robots y el controlador de la vista de la definición de área de podado, incluyendo los métodos de cada uno de estos.

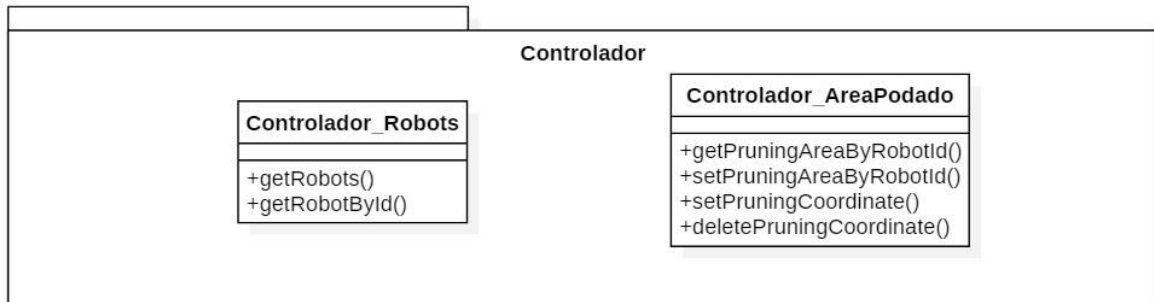


Figura 15: Diagrama de clases - Controlador.

Fuente: Autoría Propia.

En la Figura 16, se muestra el diagrama de clases correspondientes al paquete de la vista, que está conformado por la vista de listado de robots y la vista de definición de área de podado.

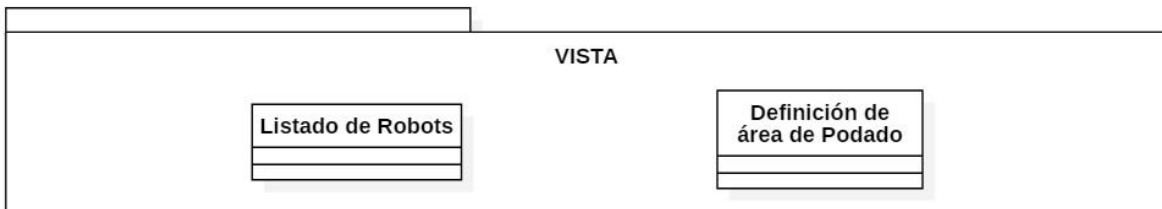


Figura 16: Diagrama de clases - Vista.

Fuente: Autoría Propia.

### Bocetos

Después de haber realizado los diagramas adecuados, podemos modelar bocetos básicos para graficar la información previamente recopilada y explorada.

En la Figura 17, mostramos la pantalla principal que, desde un primer vistazo, tendrá el listado de robots, que se obtendrán a través de la API de Opciones, disponibles para su selección.

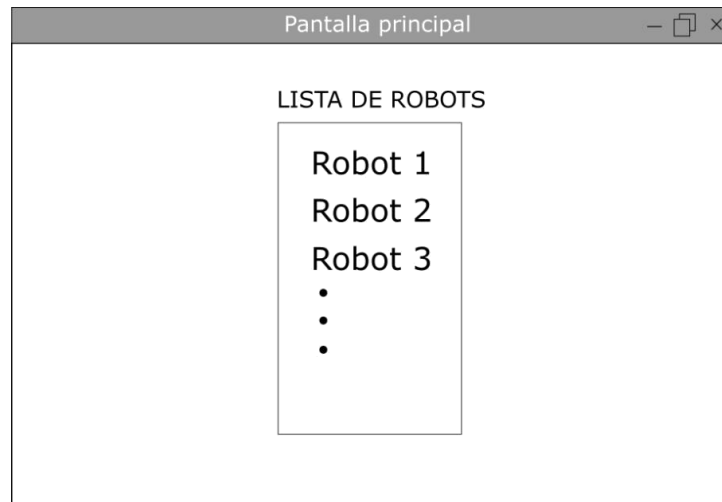


Figura 17: Boceto - Lista de Robots.

Fuente: Autoría Propia.

En la Figura 18, se muestra dos pequeños recuadros que aparecen al seleccionar un robot, uno de ellos contiene la imagen de la geolocalización y el botón para dirigirnos a la pantalla de definición de área de podado, y otro que mostrará los datos de los sensores de dicho robot; estos también serán obtenidos a través de la API de Opciones.



Figura 18: Boceto - Información del Robot.

Fuente: Autoría Propia.

En la Figura 19, se muestra la pantalla donde el usuario usará las diferentes herramientas para definir el área de podado y guardar los cambios realizados. Para esto contamos con las siguientes herramientas:

- La herramienta Dibujar, nos permitirá realizar trazos con el objetivo de dibujar el área que podará el robot.
- La herramienta Borrar, nos permite eliminar el área dibujada previamente.
- La herramienta Aceptar, nos permitirá guardará el área que hemos dibujado en el sistema informático a través de la API de Opciones.
- La herramienta Cerrar, nos permitirá salir de esta pantalla y regresar a la pantalla principal.



Figura 19: Boceto - Herramientas para definición de área de podado.

Fuente: Autoría Propia.

### 5.1.5. Ideación y Prototipado

La fase de ideación y prototipado es muy importante, porque es aquí donde podemos entrar a un ciclo repetitivo de mejora con el usuario, por las fases de exploración y modelado, para alcanzar su mayor productividad a través de una interfaz gráfica que se ajuste lo mejor posible al cumplimiento de sus necesidades y objetivos. De esta manera nos reunimos constantemente con el usuario encargado para revisar las diferentes versiones modeladas y prototipadas, lo cual nos permitió obtener la



información necesaria para mejorar dichas versiones, hasta lograr un prototipo final. En total, se realizaron 4 prototipos, incluyendo el prototipo final, en los cuales se aplicó la entrevista no estructurada para obtener una retroalimentación de sus características y poder mejorarlas.

### Prototipo Final

En las Figuras 20, 21 y 22, podemos ver el desarrollo del prototipo final de las dos vistas que conforman la interfaz gráfica. En este prototipo, podemos identificar la relación y mejora que tiene respecto a los bocetos modelados inicialmente, cumpliendo las métricas adecuadas para lograr la aceptación y satisfacción del usuario encargado.



Figura 20 :Prototipo - Lista de Robots.

Fuente: Autoría Propia.

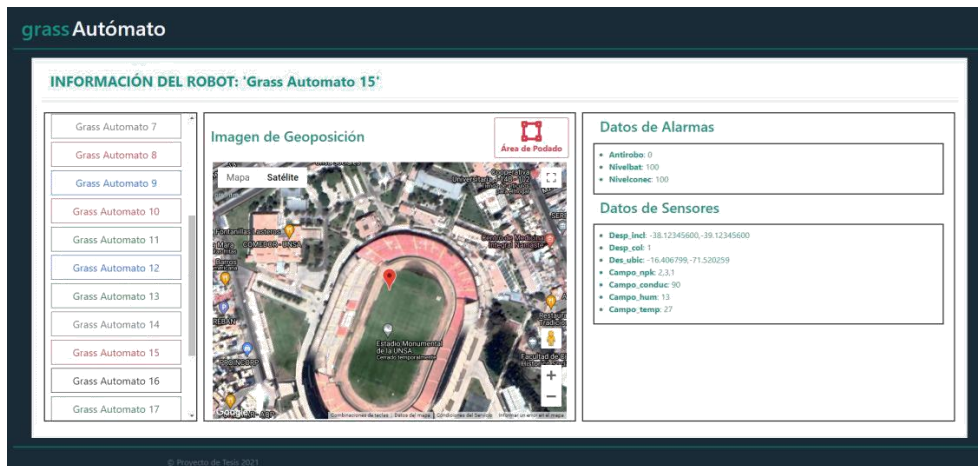


Figura 21: Prototipo - Información del Robot.

Fuente: Autoría Propia.



Figura 22: Prototipo - Herramientas para definición de área de podado.

Fuente: Autoría Propia

### 5.1.6. Formalización

Para esta fase, evaluamos la viabilidad de nuestro prototipo final, con el objetivo de precisar los elementos de diseño que va a poseer nuestra solución final. Para ello, en la etapa de estructuración, tenemos las Tablas 2 y 3, que nos indican el alcance de nuestra interfaz gráfica y los lineamientos tecnológicos a contemplar, respectivamente. Además de esto, evaluamos que nuestra solución cumpla con los requerimientos indicados en la fase de modelado. Para todas estas evaluaciones y

validaciones, nos reunimos con la ingeniera Danitza Aruquipa y formalizamos el prototipo final de la interfaz gráfica.

#### **5.1.7. Implementación**

En esta fase, desarrollamos tecnológicamente la interfaz gráfica, con el objetivo de hacer la validación de contexto con el usuario encargado. Siendo de esta manera, que desarrollamos la interfaz con la arquitectura MVC que planteamos al inicio de este apartado, y finalmente, hacer las validaciones funcionales de los lineamientos descritos en fases anteriores.

#### **5.1.8. Validación en Contexto**

Una vez desarrollada la interfaz gráfica, en esta fase, nos dedicamos al cumplimiento de requerimientos y necesidades especificadas, por parte de la interfaz gráfica. Para ello, nos reunimos con el usuario encargado y realizamos las validaciones correspondientes a través de pruebas funcionales, obteniendo retroalimentación de mejora y/o cambios que podemos desarrollar a la interfaz, y aplicamos una entrevista para validar los indicadores de este proyecto.

#### **5.1.9. Despliegue**

En esta etapa final, recopilamos toda la retroalimentación de la fase anterior y, en caso de requerir algún cambio o mejora, estos son desarrollados en la interfaz gráfica y posteriormente planteamos una reunión con el usuario encargado para la validación de estos nuevos cambios y/o mejoras requeridas. Finalmente, preparamos la interfaz gráfica para la integración con el sistema informático desarrollado.

#### **Diagrama de Despliegue**

En la Figura 23, se muestra el diagrama de despliegue, que nos permite modelar la arquitectura al momento de desplegar la interfaz gráfica, junto al sistema informático, y está compuesta por los siguientes nodos:

➤ Nodo Vista; está compuesto por la interfaz gráfica, que es la encargada de interactuar con el usuario, y Java Script para poder hacer una conexión con la API

Maps JavaScript de Google y mostrar los mapas necesarios en las diferentes vistas de la interfaz.

➤ **Nodo Controlador;** es el encargado de ejecutar las acciones y realizar los cambios que el usuario solicita a través de la interfaz gráfica, está compuesta por el framework Flask, que nos permite la creación de la aplicación web, y por las bibliotecas OpenCV, Scikit-Image y NumPy, que son los elementos claves que se están utilizando para llevar a cabo el manejo y procesamiento adecuados sobre las imágenes, que son adquiridas gracias a la conexión con la API Maps Static de Google, y poder hacer la detección de área de podado por visión artificial de manera autónoma.

➤ **Nodo APIs Google;** está compuesto por la API Maps JavaScript, que nos permite obtener un mapa navegable de una posición enviada, y de la API Maps Static, que nos brinda una imagen de una posición y resolución enviada (código disponible en [8] y [9]).

➤ **Nodo Sistema Informático;** representa el sistema informático desarrollado por la empresa Opciones, compuesta por una pequeña aplicación que nos posibilita la capacidad de conectarnos con el sistema informático para la obtención y guardado de información que se requiera desde nuestra interfaz gráfica.

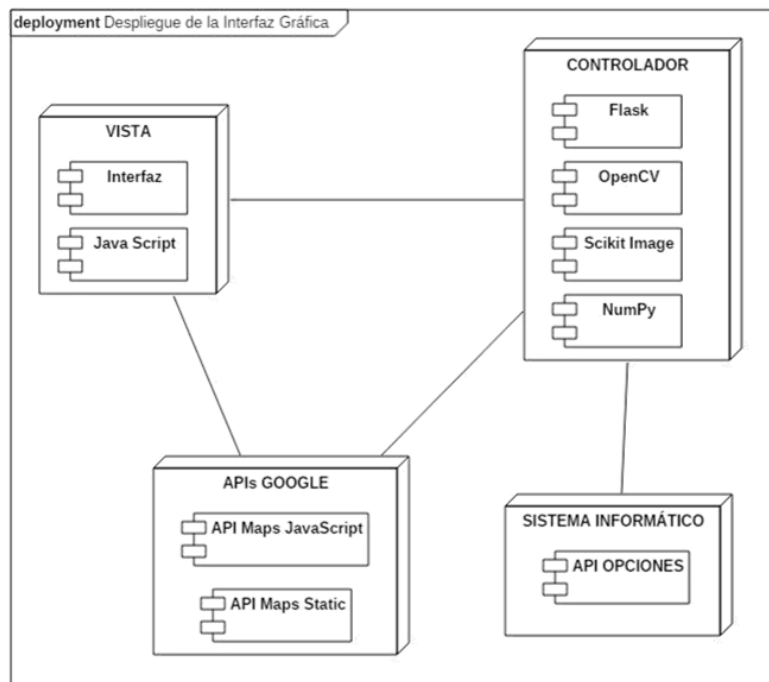


Figura 23: Diagrama de despliegue de la interfaz gráfica.

Fuente: Autoría Propia

## 5.2. Desarrollo del Sistema de Visión Artificial

Como hemos visto en el capítulo 2, la visión artificial nos permite obtener información numérica, que puede ser tratada por un ordenador, a través de la adquisición, procesamiento y análisis de imágenes, además sabemos que nuestra interfaz gráfica trabaja con mapas, lo que son un conjunto de imágenes móviles, podemos integrar un sistema de visión artificial a nuestra interfaz, con el objetivo de facilitar y automatizar el proceso de definición de área de podado para las diferentes aplicaciones en las que el robot GrassBot puede desenvolverse.

Nuestro sistema de visión artificial, está compuesto por un objeto, adquisición de imagen, procesamiento de imagen, análisis de imagen y la toma de decisiones; estos componentes son suficientes para lograr nuestro objetivo a través de las actividades indicadas en la Figura 24.

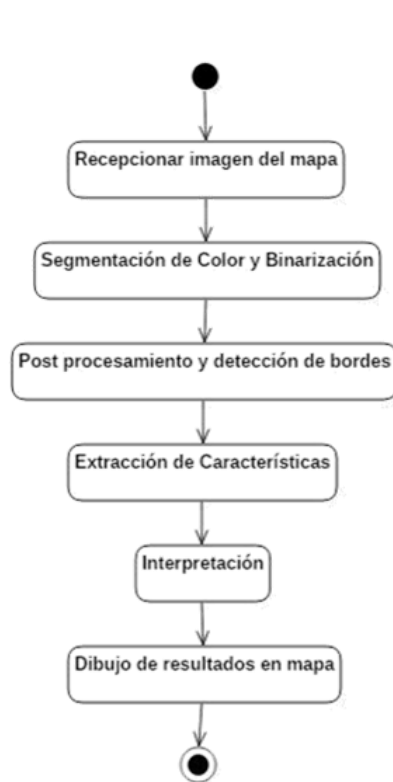


Figura 24: Diagrama de actividades del sistema de visión artificial.

Fuente: Autoría Propia

### 5.2.1.El Objetivo

Nuestro principal componente del sistema de visión artificial, será representado por las diferentes áreas de podado donde el robot GrassBot puede trabajar, estos lugares son clubes, parques cementerio, canchas de fútbol y césped ornamental, además, se requieren diversas imágenes satelitales de estas áreas.





Figura 26: Mapa e imagen satelital obtenido a través de las APIs de Google.

Fuente: Autoría Propia

### 5.2.3. Procesamiento de Imagen

Una vez que nuestra imagen llega a este componente, nuestro objetivo es obtener una imagen con únicamente nuestro objeto en cuestión, a través de diferentes procesos. Para llegar a una imagen que contenga únicamente nuestro objeto, desde la imagen satelital, realizamos como primer paso, la segmentación del color verde, a través del modelo de color HSV, debido a que es más sencillo encontrar los diferentes estados del verde mediante el tono, saturación y valor; seguidamente, procedemos a binarizar todo el conjunto de píxeles que comparten esta cualidad, para lograr una imagen que representa todos las áreas verdes, a través de píxeles blancos en la imagen obtenida, como se puede observar en la Figura 27.



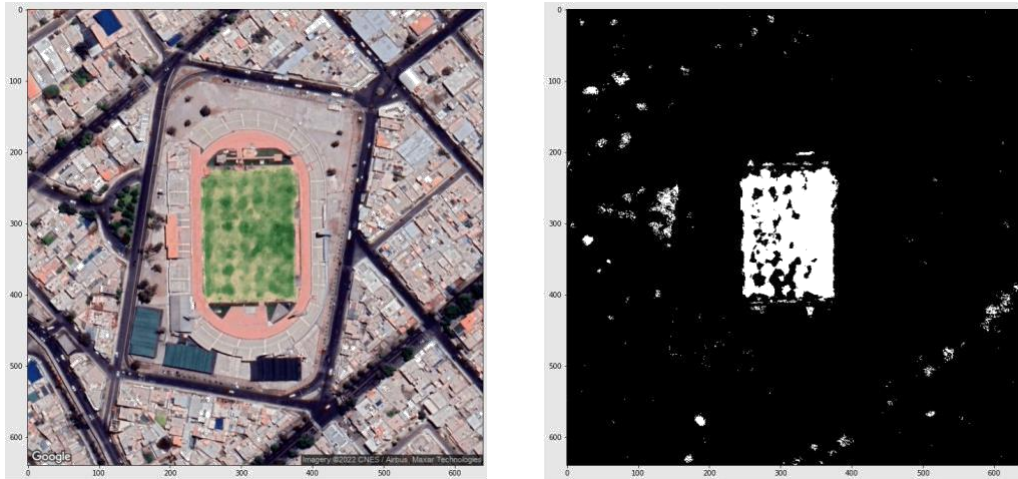


Figura 27: Segmentación y binarización de imagen satelital.

Fuente: Autoría Propia

Una vez que sabemos que nuestro objeto está contenido dentro de nuestra imagen binarizada, procedemos a realizar el post-procesamiento, por medio de diversas operaciones morfológicas, comenzando por la limpieza de ruido, mediante la apertura y cierre; rellenado de espacios faltantes, por medio de la dilatación, erosión y rellenado; finalmente la eliminación de otros objetos diferentes a nuestro objeto en cuestión, a través de la gradiente morfológica y erosión, resultando en una imagen con únicamente nuestro objeto, como se muestra en la Figura 28. Luego del post-procesamiento de la imagen, se pasa a realizar la detección de bordes del objeto, para luego aplicar el casco convexo, que nos permitirá encontrar los píxeles claves que englobarán nuestro objeto, a través de un polígono y su centroide, como se muestra en la Figura 29.

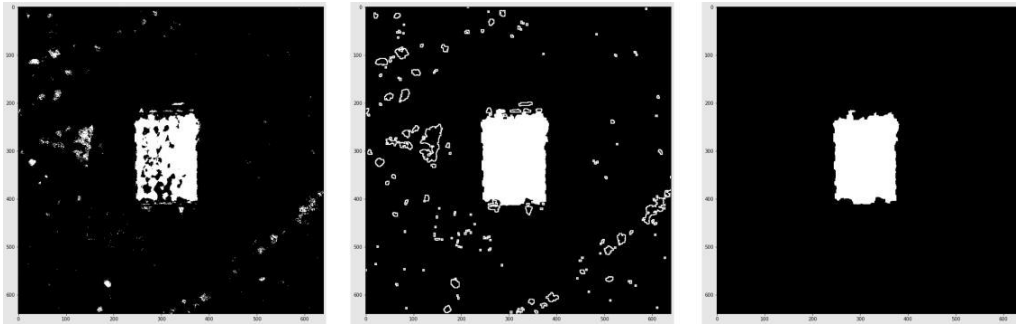


Figura 28: Post-procesamiento de la imagen satelital.

Fuente: Autoría Propia

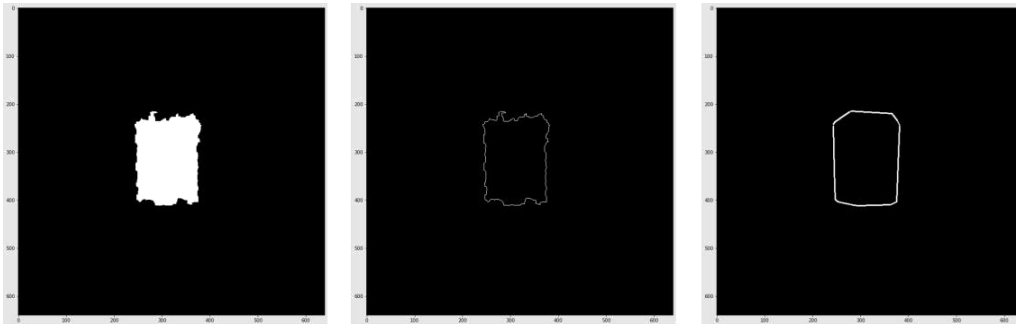


Figura 29: Detección de bordes de la imagen satelital.

Fuente: Autoría Propia

#### 5.2.4. Análisis de Imagen

Este componente, es el último que trabajará con nuestra imagen procesada, debido a que nuestra finalidad es obtener uno o varios valores que se puedan interpretar para la toma de decisiones desde dicha imagen. Es por esto que obtenemos los píxeles característicos del casco convexo y los utilizamos para convertirlos en coordenadas geográficas, a través de la proyección de los sistemas geodésicos mundiales EPSG:4326 WGS 84 y EPSG:900913 Google Maps Global Mercator. Para representar dichas coordenadas, en el mapa de nuestra interfaz gráfica, realizamos diversas conversiones requeridas de varias fórmulas obtenidas desde MapTiler, en

[50]. Es importante señalar tres puntos que nos ayudan a comprender el motivo de las conversiones realizadas:

- El sistema geodésico mundial EPSG:4326, está basado en un sistema de coordenadas elipsoidal, sus ejes son la latitud y longitud, la orientación es el norte y este, y la unidad de medida son los grados.
- El sistema geodésico mundial EPSG:900913, está basado en un sistema de coordenadas esférico, sus ejes son el este y el norte, la orientación es el este y norte, y la unidad de medida son los metros.
- La proyección de los mosaicos ráster de la tierra, tienen un tamaño de 256x256 píxeles, desde un zoom igual a 0, y son contados desde la esquina inferior-izquierda, para el Sistema de Gestión de Transporte (TMS), mientras que para Google, son contados desde la esquina superior-izquierda.

Para comenzar con la conversión de estos píxeles claves, nuestro primer paso es identificar los píxeles, a nivel mundial, que representa la geoposición del robot, esto significa que debemos convertir la latitud y longitud (EPSG:4326 WGS 84) a metros en las coordenadas X y Y (EPSG:900913 Google Maps Global Mercator) para luego convertirlo en píxeles  $x$  y  $y$ . Estas transformaciones, requieren que tengamos las variables *originShift* (5.1) e *initialResolution* (5.2) previamente definidas:

$$originShift = \frac{2 * \pi * EARTH\_RADIUS}{2} \quad (5.1)$$

Donde:

- EARTH\_RADIUS: es el radio de la tierra en metros, que es igual a 6378137 metros.

$$initialResolution = \frac{2 * originShift}{TILE\_SIZE} \quad (5.2)$$

Donde:

➤ **TILE\_SIZE**: es el tamaño del mosaico, que es igual a 256 píxeles. Comenzamos haciendo la transformación de la longitud a metros en la coordenada X a través de (5.3):

$$lng\_meters\_x = \frac{LNG * originShift}{180} \quad (5.3)$$

Donde:

➤ **LNG**: es la longitud de la geoposición del robot GrassBot.

Y la transformación de la latitud a metros en la coordenada Y, mediante (5.4):

$$lat\_meters\_y = \frac{\frac{\ln(\tan(\frac{(90 + LAT) * \pi}{360}))}{\frac{\pi}{180}} * originShift}{180} \quad (5.4)$$

Donde:

➤ **LAT**: es la latitud de la geoposición del robot GrassBot.

En este punto, ya tenemos la representación en metros para las coordenadas X y Y de la latitud y longitud de las coordenadas del robot GrassBot, por lo que ahora convertimos los metros (X, Y) en píxeles (x, y), y para esto necesitamos conocer el *zoom* utilizado en nuestra imagen satelital, para aplicarlo en (5.5), y con esta última, conseguir el píxel en la coordenada x, a través de (5.6), y el píxel en la coordenada y, por medio de (5.7):

$$resolution = \frac{initialResolution}{2^{zoom}} \quad (5.5)$$

$$meters\_pixel\_x = \frac{meters\_x + originShift}{resolution} \quad (5.6)$$

$$meters\_pixel\_y = \frac{meters\_y + originShift}{resolution} \quad (5.7)$$

Ahora que ya sabemos el píxel que representa nuestra geoposición del robot GrassBot en los píxeles de las coordenadas de la proyección mundial, podemos hacer

el cálculo para encontrar el píxel que representa el centroide de nuestro casco convexo en los píxeles de las coordenadas de la proyección del mundial, y esto lo hacemos mediante (5.8), para el píxel x, y (5.9), para el píxel y:

$$pixel\_centroid\_x = pixel\_x + hull\_centroid\_x - \frac{IMAGE\_WIDTH}{2 * IMAGE\_SCALE} \quad (5.8)$$

Donde:

- hull\_centroid\_x: es el píxel en la coordenada x del centroide del casco convexo.
- IMAGE\_WIDTH: es el ancho de nuestra imagen satelital.
- IMAGE\_SCALE: es la escala de nuestra imagen satelital.

$$pixel\_centroid\_y = pixel\_y - (hull\_centroid\_y - \frac{IMAGE\_HEIGHT}{2 * IMAGE\_SCALE}) \quad (5.9)$$

Donde:

- hull\_centroid\_y: es el píxel en la coordenada y del centroide del casco convexo.
- IMAGE\_HEIGHT: es el alto de nuestra imagen satelital.

Una vez obtenido los píxeles (x, y) del centroide de nuestro casco convexo representado en los píxeles coordenadas de la proyección mundial, podemos obtener fácilmente la misma representación de los píxeles del casco convexo que encierra nuestra área detectada, como se muestra en la siguiente imagen:

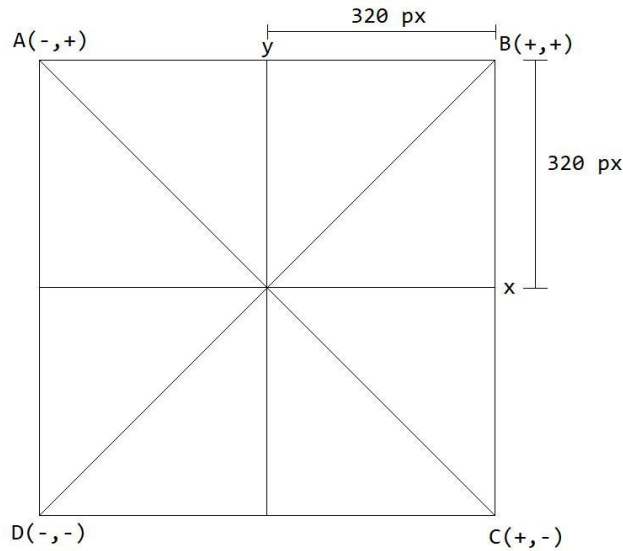


Figura 30: Conversión de píxeles a coordenadas de la proyección mundial.

Fuente: Autoría Propia

Para este punto, ya tenemos todos los píxeles de nuestro casco convexo representados en los píxeles coordenadas de la proyección mundial, por lo que ahora debemos convertir cada uno de estos píxeles en metros  $X$  y  $Y$ , haciendo uso de (5.5) y las siguientes ecuaciones:

$$pixel\_meters\_x = pixel\_x * resolution - originShift \quad (5.10)$$

$$pixel\_meters\_y = pixel\_y * resolution - originShift \quad (5.11)$$

Después de obtener la posición en metros de cada píxel de nuestro casco convexo, solo nos falta transformar cada uno de ellos en latitud y longitud para, que se pueda representar en el mapa de nuestra interfaz gráfica, para ello aplicamos (5.12) y (5.13), para las coordenadas  $X$  y  $Y$ , respectivamente:

$$meters\_x\_lng = \frac{pixel\_meters\_x}{originShift} * 180 \quad (5.12)$$

$$meters\_y\_lat = \frac{180}{\pi} * \frac{\left( 2 * \arctan \left( \frac{pixel\_meters\_x}{\epsilon} \frac{originShift * 180 * \pi}{180} \right) - \pi \right)}{2} \quad (5.13)$$

Y es así como, hemos conseguido el conjunto de datos que nos indica la latitud y longitud de cada píxel de nuestro casco convexo, siendo este el resultado final del componente de análisis de imagen.

### 5.2.5. Toma de Decisiones

Finalmente, tenemos el conjunto de valores que ha proporcionado el componente de análisis de imagen, y con eso podemos dibujar estos valores dentro del mapa de nuestra interfaz gráfica. En la Figura 31, se muestra el resultado de utilizar la API JavaScript Maps de Google, indicándole el conjunto de valores obtenidos, representados por las coordenadas de latitud y longitud.



Figura 31: Representación de las coordenadas resultantes en un mapa.

Fuente: Autoría Propia

## **CAPÍTULO 6**

### **ANÁLISIS E INTERPRETACIÓN DE RESULTADOS**

En este capítulo se muestra la validación y análisis de resultados relacionados a las diferentes características de la interfaz gráfica y el sistema de visión artificial, para la detección de áreas de podado, con base a la geoposición del robot autónomo GrassBot. Para la interfaz gráfica, se validan los requerimientos que fueron recabados en el desarrollo de la misma, y se analizaron los resultados obtenidos, respecto a las características de experiencia de usuario (UX) que presenta la interfaz, a través de una entrevista aplicada a la empresa Opciones, evaluando diferentes cualidades que nos permitirán indicar la experiencia de usuario lograda.

Para el sistema de visión artificial, se analizaron los resultados obtenidos, comparando una definición manual, con la detección del sistema de visión artificial de un área de podado, en diferentes ubicaciones donde el robot puede aplicarse.

#### **6.1. Funcionalidades de la Interfaz Gráfica**

A continuación, se presenta la validación de los requerimientos funcionales del prototipo de la interfaz gráfica desarrollada.

- La interfaz gráfica debe permitir visualizar un listado de los robots disponibles.



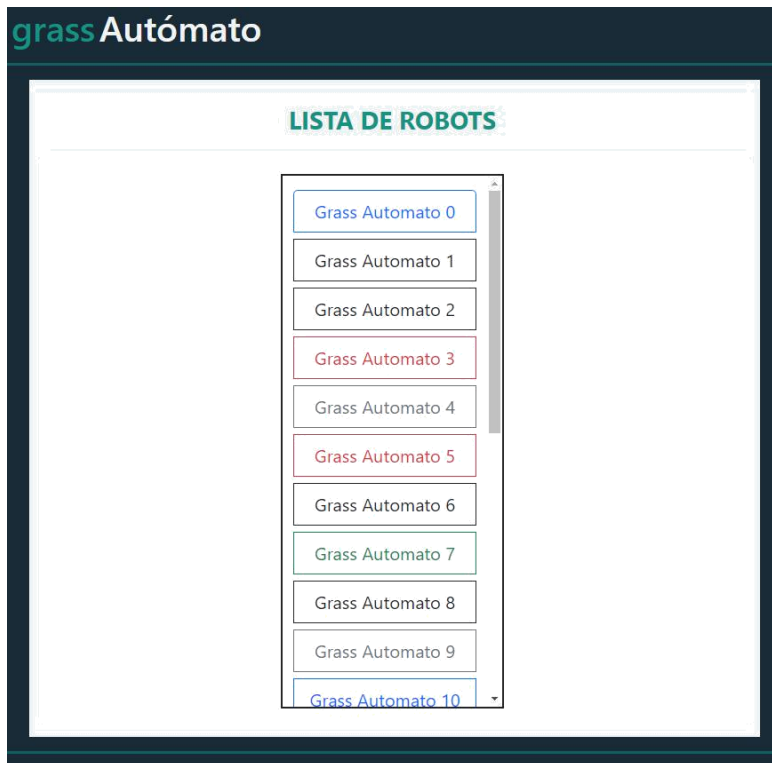


Figura 32: Visualización del listado de robots.

Fuente: Autoría Propia

- La interfaz gráfica debe mostrar una vista previa de la ubicación del robot seleccionado en un mapa.



Figura 33: Vista previa de la geoposición de un robot en un mapa.

Fuente: Autoría Propia

- La interfaz gráfica debe permitir visualizar la información de los diferentes sensores que el robot registra en la API de la nube.



Figura 34: Visualización de sensores del robot.

Fuente: Autoría Propia

- La interfaz gráfica debe tener herramientas de dibujo sobre un mapa, para indicar el área de podado.



Figura 35: Herramientas de dibujo para trazar el área de podado sobre un mapa.

Fuente: Autoría Propia

- La interfaz gráfica debe tener una herramienta que aplica visión artificial, para la detección de áreas de podado.



Figura 36: Detección de área de podado a través de visión artificial.

Fuente: Autoría Propia

- La interfaz gráfica, debe permitir guardar la información del área de podado dibujada en el mapa.



Figura 37: Herramienta para guardar los datos generados en el mapa.

Fuente: Autoría Propia

```

127.0.0.1 - - [02/Feb/2022 21:07:52] "GET /api/robotPruningArea/6192c5699a3659e49986eb71?datapoint_id=61eeda886c3c7657239c6a83 HTTP/1.1" 200 -
127.0.0.1 - - [02/Feb/2022 21:07:52] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [02/Feb/2022 21:07:58] "GET /api/automaticPruningAreaDetection/?lat=-16.408615&lng=-71.530987&zoom=18 HTTP/1.1" 200 -
*** COORDENADAS DE TRAZO ***
[{'lat': -16.40805924, 'lng': -71.53093336}, {'lat': -16.40807468, 'lng': -71.53085825}, {'lat': -16.40810555, 'lng': -71.53076169}, {'lat': -16.4081107, 'lng': -71.53075097}, {'lat': -16.40813128, 'lng': -71.53071341}, {'lat': -16.40815701, 'lng': -71.53068123}, {'lat': -16.40816216, 'lng': -71.53067586}, {'lat': -16.40823935, 'lng': -71.53064904}, {'lat': -16.40829081, 'lng': -71.53063295}, {'lat': -16.40829595, 'lng': -71.53063295}, {'lat': -16.40833197, 'lng': -71.53063831}, {'lat': -16.40899065, 'lng': -71.53078852}, {'lat': -16.40910386, 'lng': -71.53081534}, {'lat': -16.40914503, 'lng': -71.5308368}, {'lat': -16.40918105, 'lng': -71.53086898}, {'lat': -16.4091862, 'lng': -71.53087435}, {'lat': -16.40920164, 'lng': -71.53090117}, {'lat': -16.4092428, 'lng': -71.53103528}, {'lat': -16.40924795, 'lng': -71.53108356}, {'lat': -16.40924795, 'lng': -71.53111575}, {'lat': -16.4092428, 'lng': -71.5311372}, {'lat': -16.40922222, 'lng': -71.53120158}, {'lat': -16.40921193, 'lng': -71.53121767}, {'lat': -16.40911415, 'lng': -71.53134105}, {'lat': -16.40909872, 'lng': -71.53135714}, {'lat': -16.40907813, 'lng': -71.53137324}, {'lat': -16.40894949, 'lng': -71.53137324}, {'lat': -16.40860471, 'lng': -71.53129277}, {'lat': -16.4081673, 'lng': -71.53118012}, {'lat': -16.40814157, 'lng': -71.53116403}, {'lat': -16.4081107, 'lng': -71.53113184}, {'lat': -16.40806438, 'lng': -71.53097627}, {'lat': -16.40805924, 'lng': -71.53095481}]
127.0.0.1 - - [02/Feb/2022 21:08:00] "PUT /api/robotPruningArea/6192c5699a3659e49986eb71 HTTP/1.1" 200 -
127.0.0.1 - - [02/Feb/2022 21:08:00] "GET / HTTP/1.1" 200 -

```

Figura 38: Muestra de información guardada desde la consola.

Fuente: Autoría Propia

## 6.2. Experiencia de Usuario de la Interfaz Gráfica

Se desarrolló una entrevista de 14 preguntas, con la finalidad de evaluar la experiencia de usuario de la interfaz gráfica desarrollada, mediante los indicadores de simplicidad, comunicación, estructura, navegación y utilidad. Esta entrevista fue aplicada a los encargados del sistema informático de la empresa Opciones (N=10), de los cuales se obtuvo los siguientes resultados para cada dimensión.

### 6.2.1. Dimensión Simplicidad

**Pregunta 1:** ¿Puede orientarse fácilmente dentro de la interfaz?

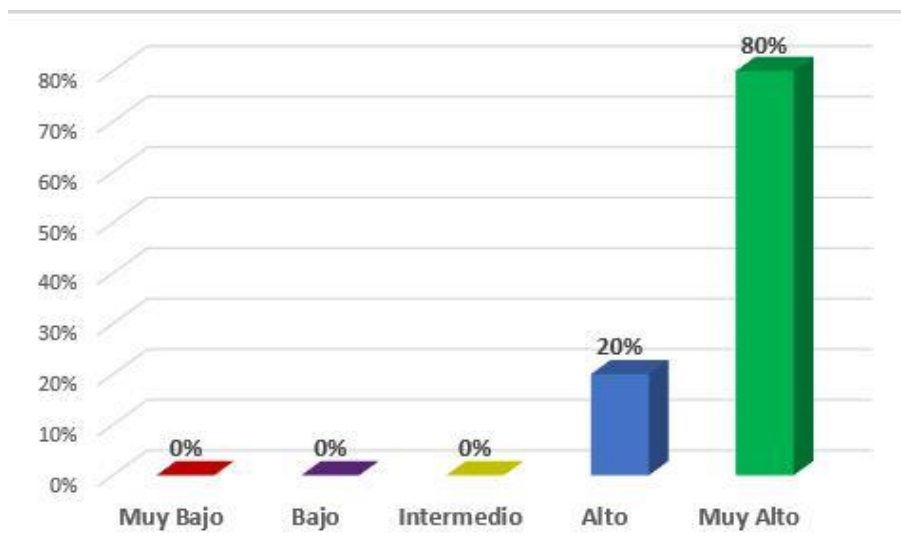


Figura 39: Simplicidad pregunta 1.

Fuente: Autoría Propia

De la Figura 39, observe que un 80 % de los encuestados clasifican la interfaz dentro de la categoría muy alto, y un 20 % dentro de la categoría alto, entonces los operarios se ubican rápidamente dentro de las diferentes vistas de la interfaz.

**Pregunta 2:** ¿Puede ubicar la información requerida fácilmente?

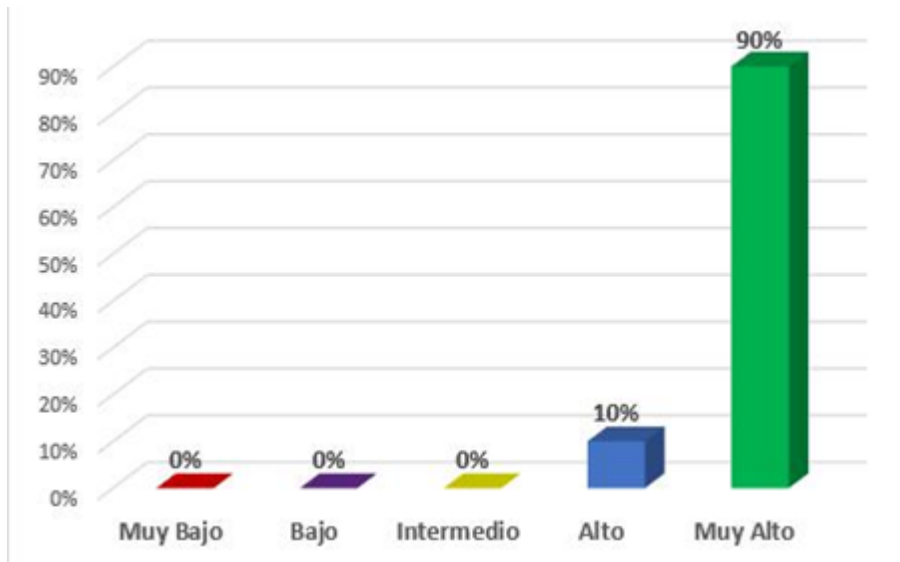


Figura 40: Simplicidad pregunta 2.

Fuente: Autoría Propia

Según la Figura 40, se muestra que un 90 % dentro de la categoría muy alto, y un 10 % dentro de la categoría alto. Entonces, los usuarios localizan fácilmente los datos del robot que le sean deseados.

**Pregunta 3:** ¿La interfaz detalla toda la información necesaria para el cumplimiento de sus operaciones?

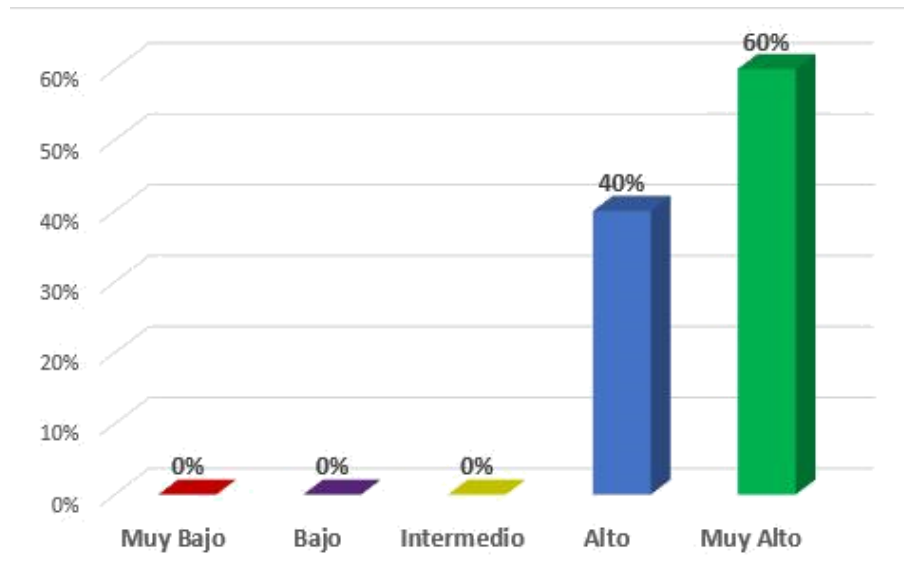


Figura 41: Simplicidad pregunta 3.

Fuente: Autoría Propia

Se muestra en la Figura 41, que un 60 % es calificada dentro de la categoría muy alto, y un 40 % dentro de la categoría alto. Entonces, la información mostrada es entendible para el operario del sistema y determinar el estado actual del robot.

### 6.2.2. Dimensión Comunicación

**Pregunta 4:** ¿Se le informa correctamente sobre cada acción a realizar?

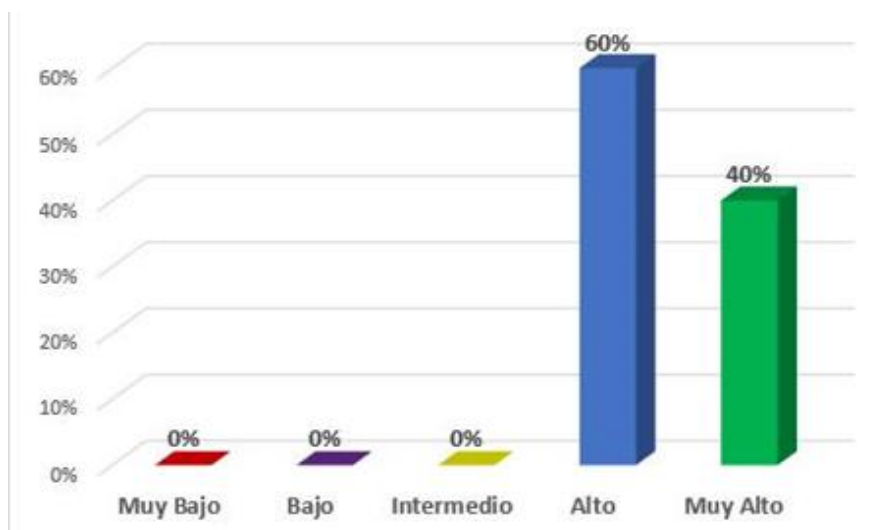


Figura 42: Comunicación pregunta 4.

Fuente: Autoría Propia

Como se muestra en la Figura 42, encontramos un 40 % en la categoría muy alto y un 60 % en la categoría alto. Entonces, se interpreta que los cuadros de diálogo, de la interfaz gráfica, transmiten la información correcta cuando el operario del sistema realiza alguna acción.

**Pregunta 9:** ¿Considera entendibles los cuadros de diálogo desplegados por la interfaz?

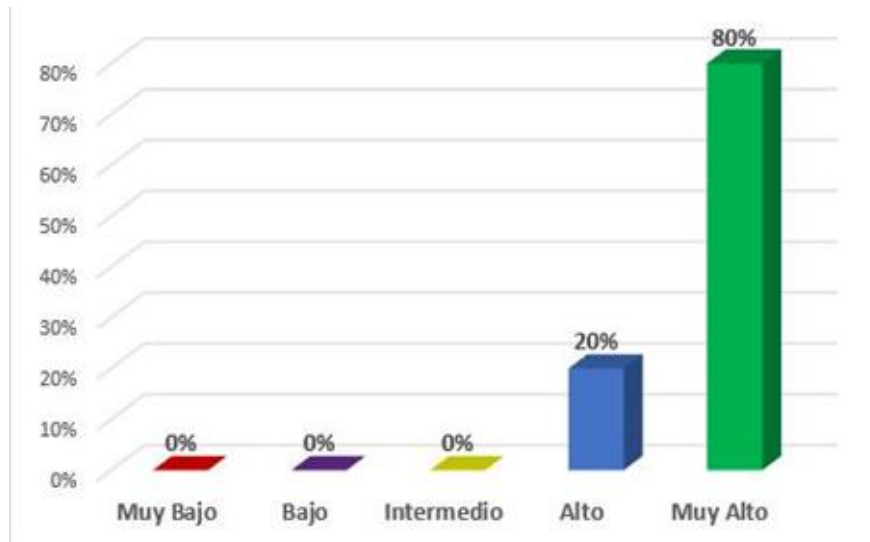


Figura 43: Comunicación pregunta 9.

Fuente: Autoría Propia

Según la Figura 43, se muestra un 80 % dentro de la categoría muy alto, y un 20 % dentro de la categoría alto. Entonces, los textos incluidos dentro de los cuadros de diálogo, de la interfaz gráfica, se han redactado de manera adecuada, permitiendo al operario del sistema entender lo que el sistema le quiere informar a través de estos textos.

### 6.2.3. Dimensión Estructura

**Pregunta 5:** ¿El diseño de la interfaz es atractivo?

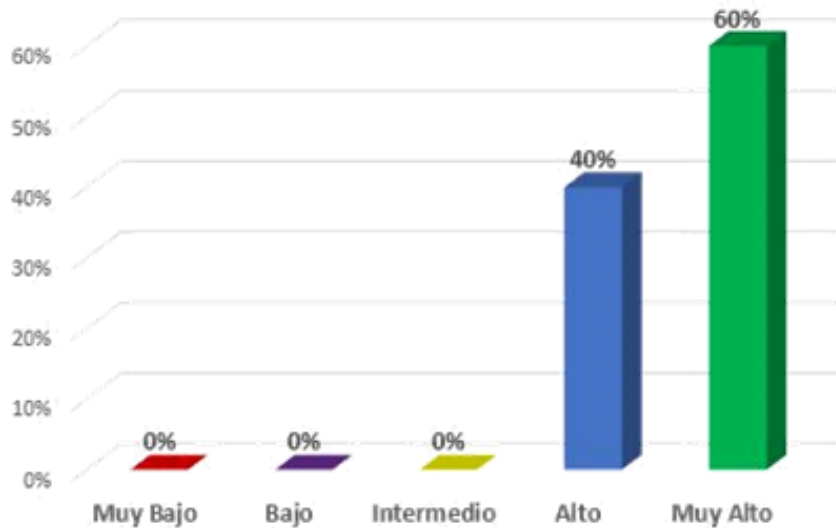


Figura 44: Estructura pregunta 5.

Fuente: Autoría Propia

Como se muestra en la Figura 44, un 60 % está dentro de la categoría muy alto y un 40 % dentro de la categoría alto. Entonces, la interfaz gráfica es amigable para el operario del sistema resultando en una interfaz gráfica aliciente.

**Pregunta 8:** ¿El diseño de la interfaz está distribuido correctamente?

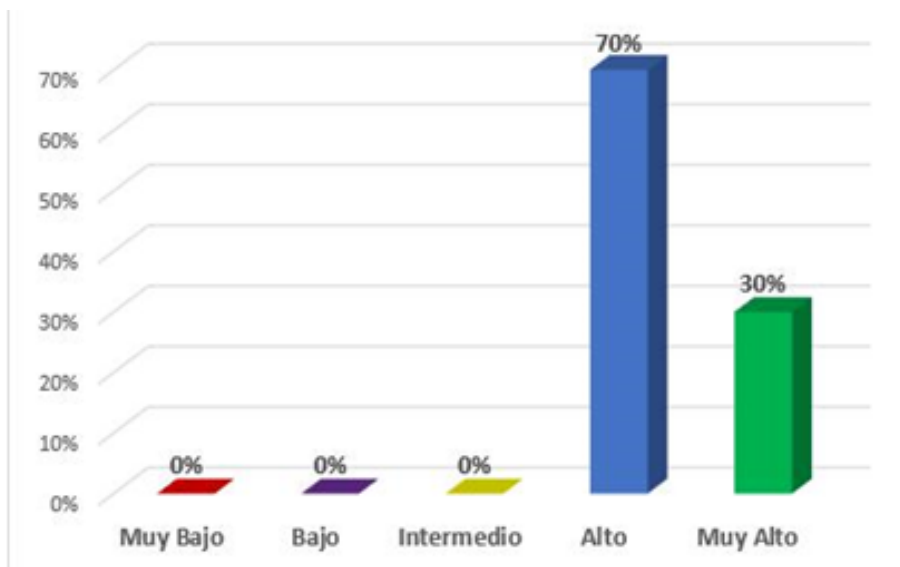


Figura 45: Estructura pregunta 8.

Fuente: Autoría Propia



Como se muestra en la Figura 45, sobre la distribución del diseño de la interfaz gráfica, es muy alto en un 30 %, y alto en un 70 %. Entonces, los diversos componentes de la interfaz gráfica, fueron ubicados lo suficientemente correctos a la vista los operarios.

#### 6.2.4. Dimensión Navegación

**Pregunta 6:** ¿Le resulta fácil navegar por la interfaz?

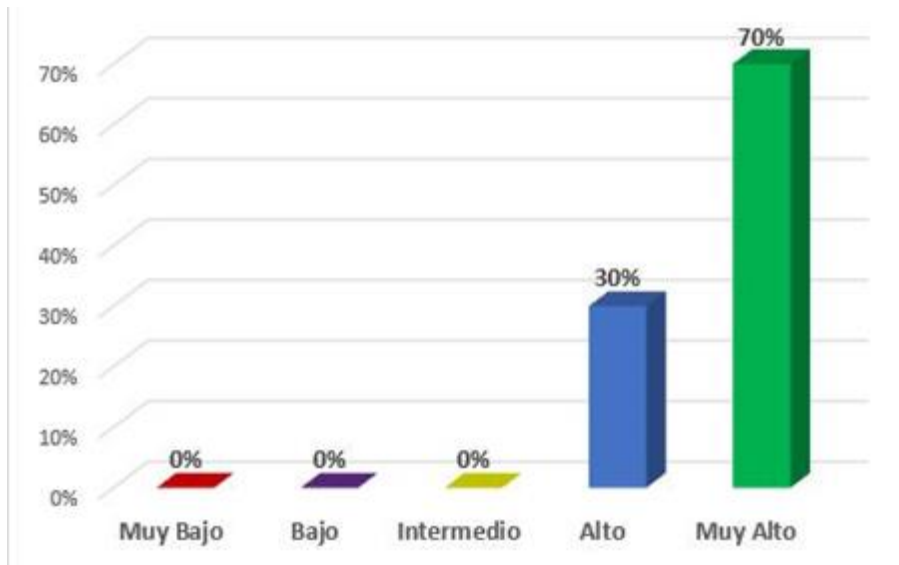


Figura 46: Navegación pregunta 6.

Fuente: Autoría Propia

Según la Figura 46, un 70 % se encuentra dentro de la categoría muy alto, y un 30 % dentro de la categoría alto. Entonces, la sencillez de la interfaz es suficiente para la rápida navegación del operario en el sistema.

**Pregunta 7:** ¿Encuentra usted dinámica la interfaz para desplazarse entre las páginas?

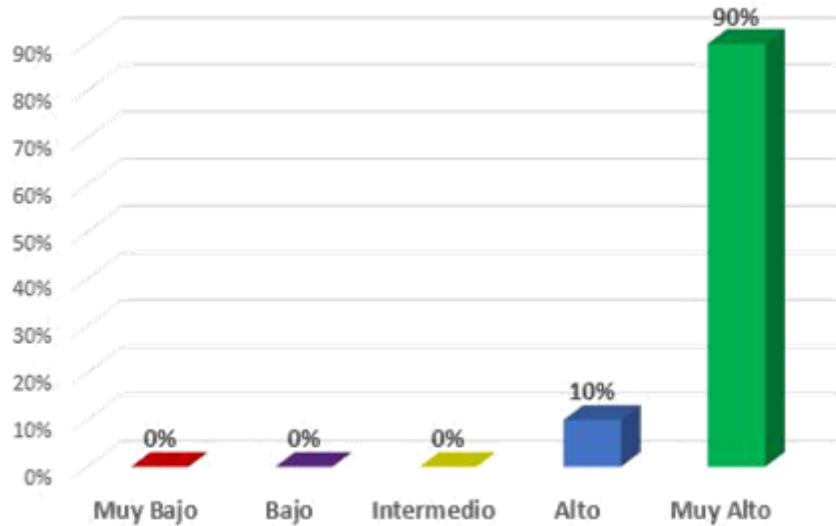


Figura 47: Navegación pregunta 7.

Fuente: Autoría Propia

Como se muestra en la Figura 47, el 90 % de los entrevistados están dentro de la categoría muy alto, y el otro 10 %, están dentro de la categoría alto. Entonces, la interfaz consta de botones que le permite, al operario del sistema, interactuar dinámicamente entra las páginas de la interfaz.

**Pregunta 10:** ¿La interfaz ofrece las funcionalidades necesarias para cubrir las operaciones a realizar?

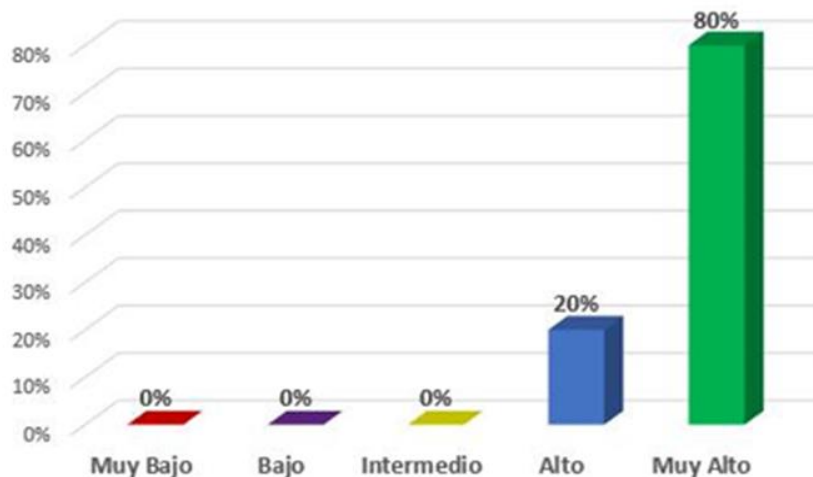


Figura 48: Navegación pregunta 10.

Fuente: Autoría Propia

Como se muestra en la Figura 48, un 80 % se encuentra en la categoría muy alto, y un 20 % en la categoría alto. Entonces, la interfaz contiene las herramientas necesarias para que, el operario del sistema, pueda desarrollar sus funciones con normalidad.

**Pregunta 11:** ¿Considera que la interfaz es intuitiva?

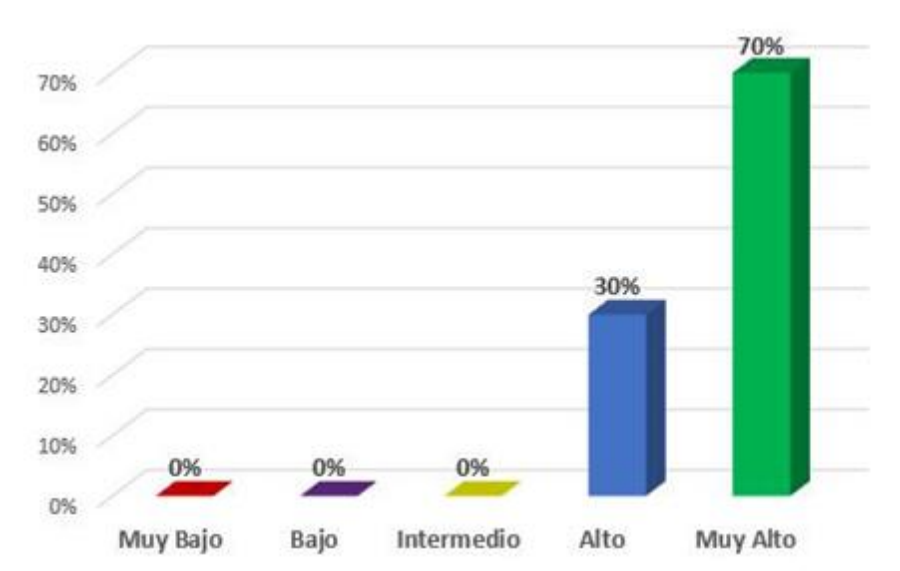


Figura 49: Navegación pregunta 11.

Fuente: Autoría Propia

Según la Figura 49, un 70 % se encuentra en la categoría muy alto, y un 30 % en la categoría alto. Entonces, la interfaz contiene elementos de rápida percepción, lo que permite al operario del sistema, comprender y navegar fácilmente en el entorno en que se encuentra.

#### 6.2.5. Dimensión Utilidad

**Pregunta 12:** ¿Considera que la información mostrada es veraz y actual?

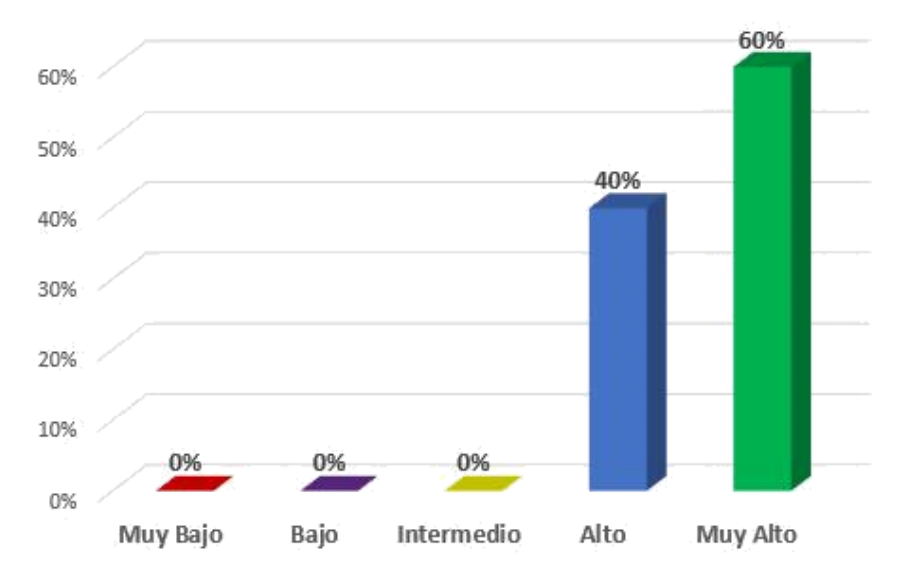


Figura 50: Utilidad pregunta 12.

Fuente: Autoría Propia

Según la Figura 50, un 60 % se encuentra en la categoría muy alto, y un 40 % se encuentra en la categoría alto. Entonces, la información que se obtiene a través de la API desde el sistema informático de Opciones, puede ser vigente en gran parte, pero en otras situaciones no, esto se debe a que el robot GrassBot, debe conectarse a la estación de carga para enviar su información actual, lo cual es realizado cada cierto intervalo de tiempo.

**Pregunta 13:** ¿Considera que la herramienta de visión artificial brindada automatiza la operación de definición de área de podado?

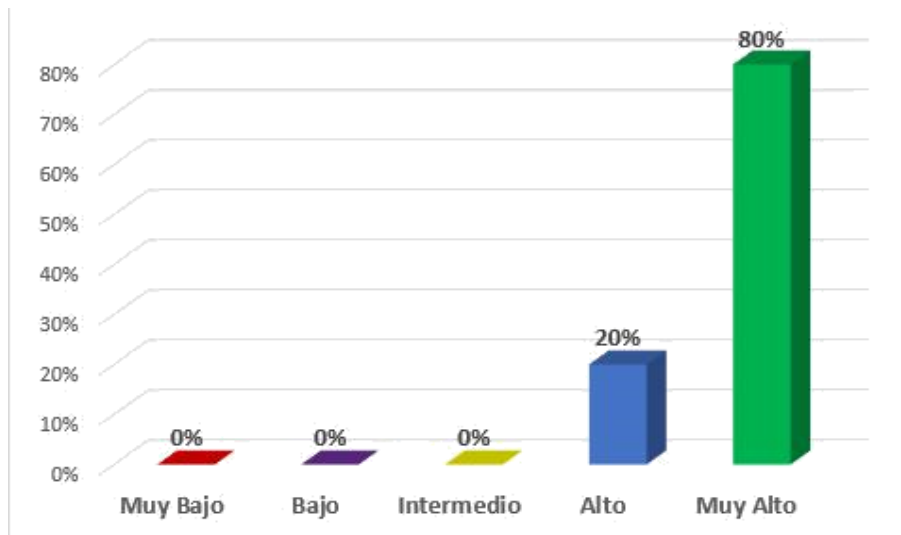


Figura 51: Utilidad pregunta 13.

Fuente: Autoría Propia

Como se observa en la Figura 51, un 80 % se encuentra en la categoría muy alto y un 20 % se encuentra en la categoría alto. Entonces, la herramienta de visión artificial es percibida, en la mayoría de casos, que automatiza el proceso de definición de área de podado suficientemente precisa.

**Pregunta 14:** ¿Considera que son útiles las diferentes herramientas que brinda la interfaz para el desarrollo de sus operaciones?

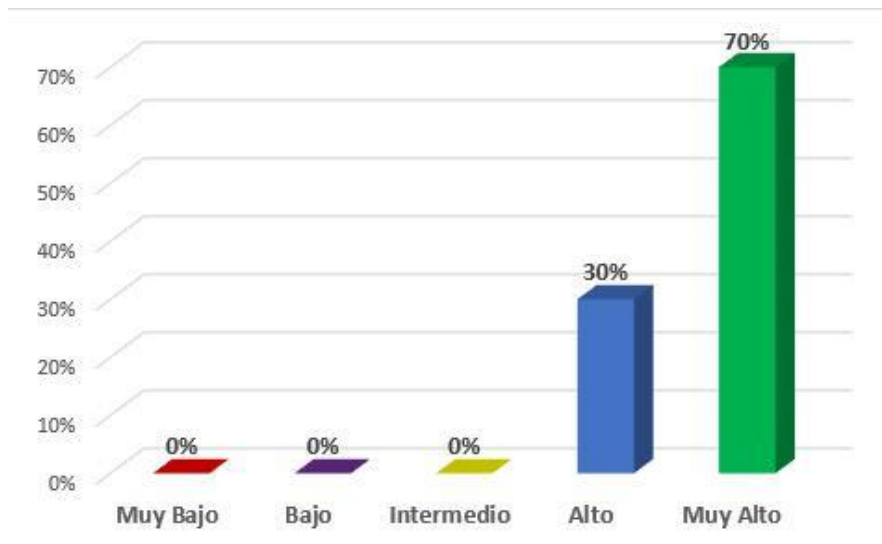


Figura 52: Utilidad pregunta 14.

Fuente: Autoría Propia

Según la Figura 52, un 70 % se encuentra en la categoría muy alto y un 30 % en la categoría alto. Entonces, las herramientas desarrolladas en la interfaz gráfica cumplen gran parte de las operaciones que el operario del sistema requiere, tanto manualmente como de manera asistida.

#### 6.2.6. Evolución de las Dimensiones

Durante todo el proceso de desarrollo de la interfaz gráfica, se desarrollaron cuatro prototipos de manera iterativa, mostrados en las Figuras 53, 55, 57 y 59. Esto se realizó a lo largo del tiempo del proyecto, a través de la aplicación del instrumento para cada modelo de nuestra interfaz gráfica, y se realizaron las mejoras correspondientes con base a los resultados obtenidos por la encuesta, véase Figuras 54, 56, 58 y 60.

LISTA DE ROBOTS



HERRAMIENTAS DE PODADO



Figura 53: Prototipo del modelo 1.

Fuente: Autoría Propia

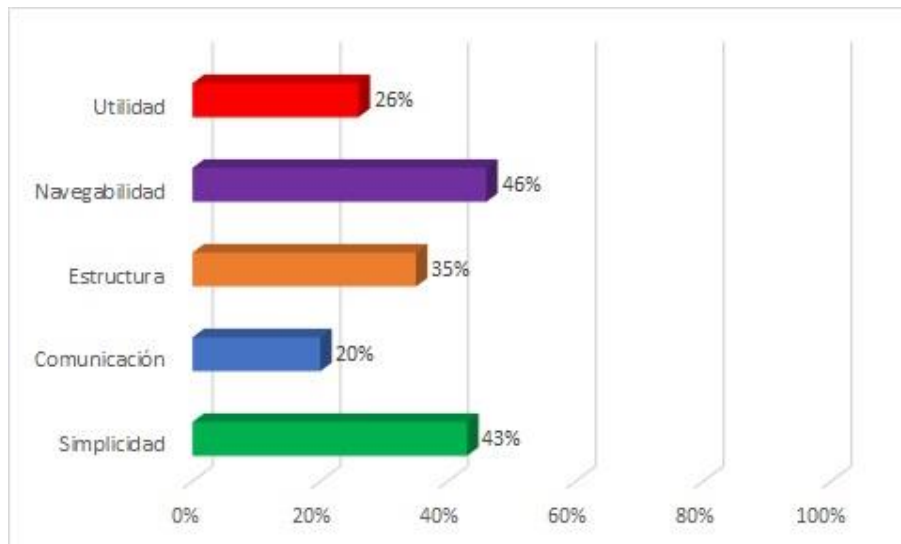


Figura 54: Resultados de entrevista del modelo 1.

Fuente: Autoría Propia

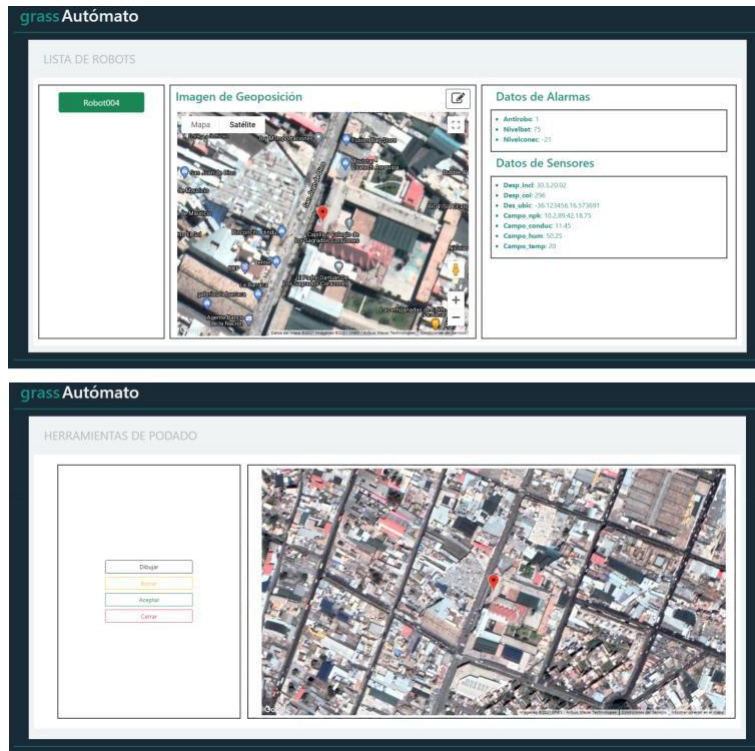


Figura 55: Prototipo del modelo 2.

Fuente: Autoría Propia

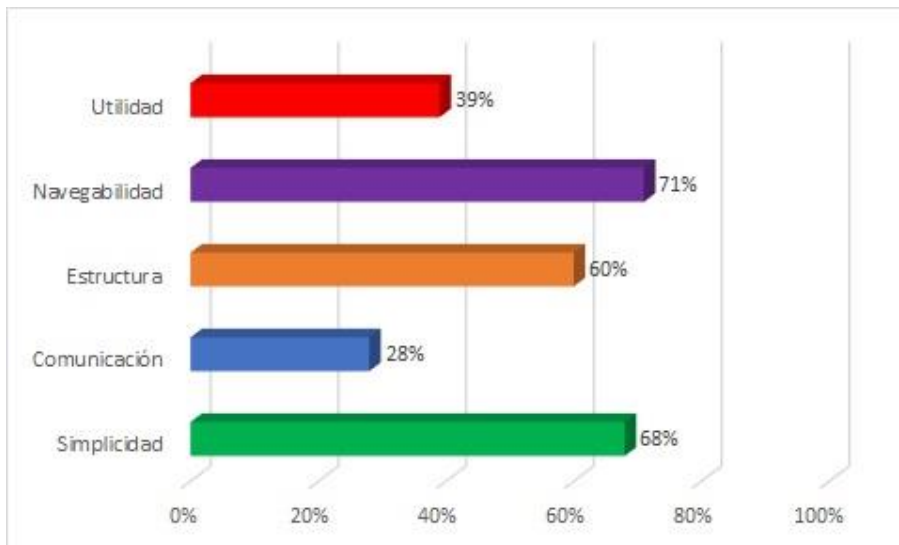


Figura 56: Resultados de entrevista del modelo 2.

Fuente: Autoría Propia





Figura 57: Prototipo del modelo 3.

Fuente: Autoría Propia

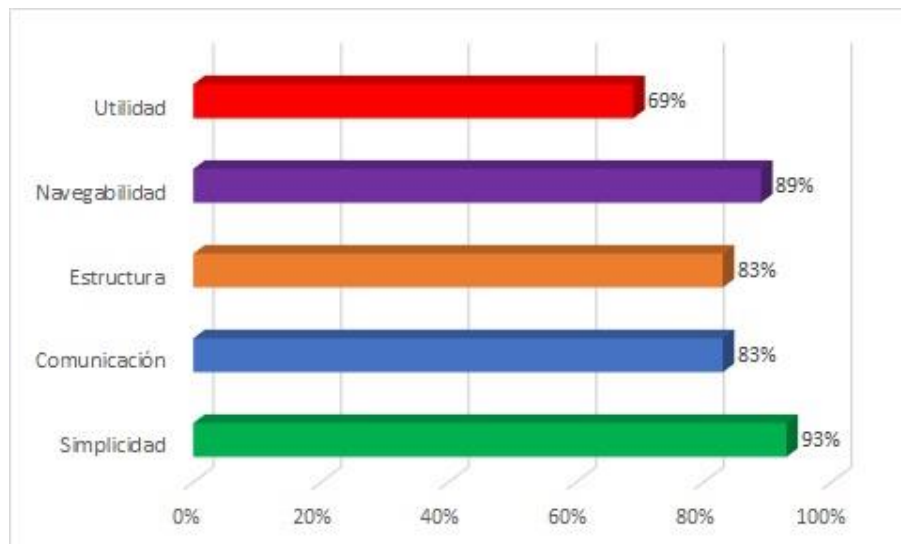


Figura 58: Resultados de entrevista del modelo 3.

Fuente: Autoría Propia

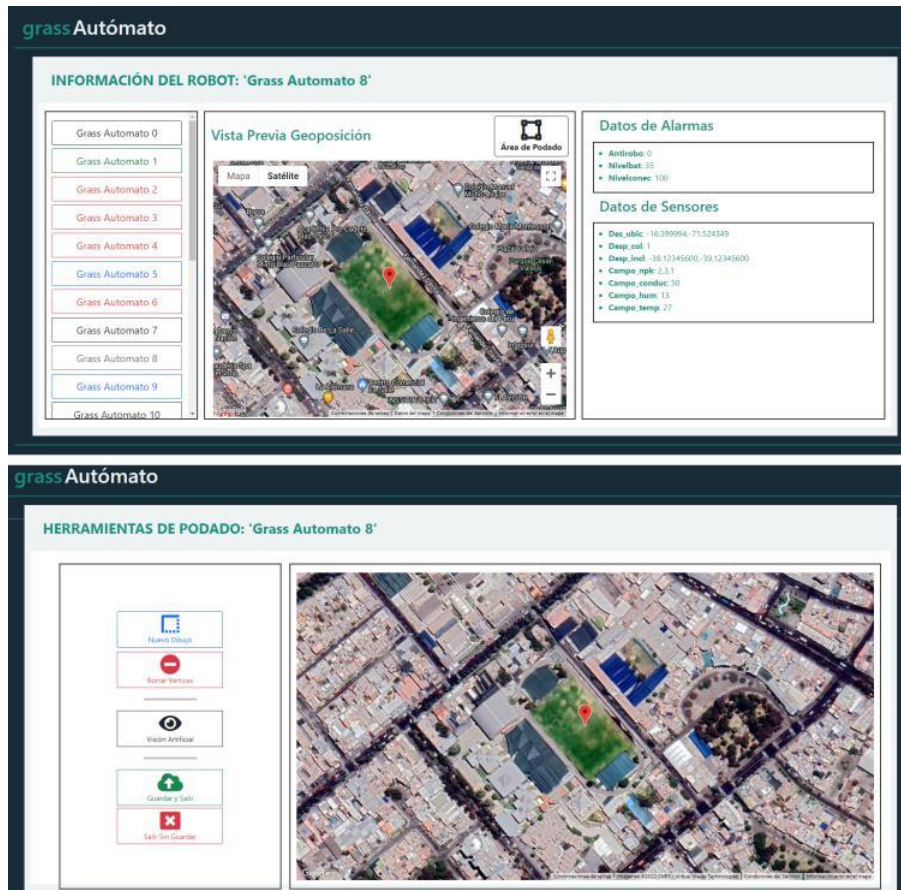


Figura 59: Prototipo del modelo 4.

Fuente: Autoría Propia

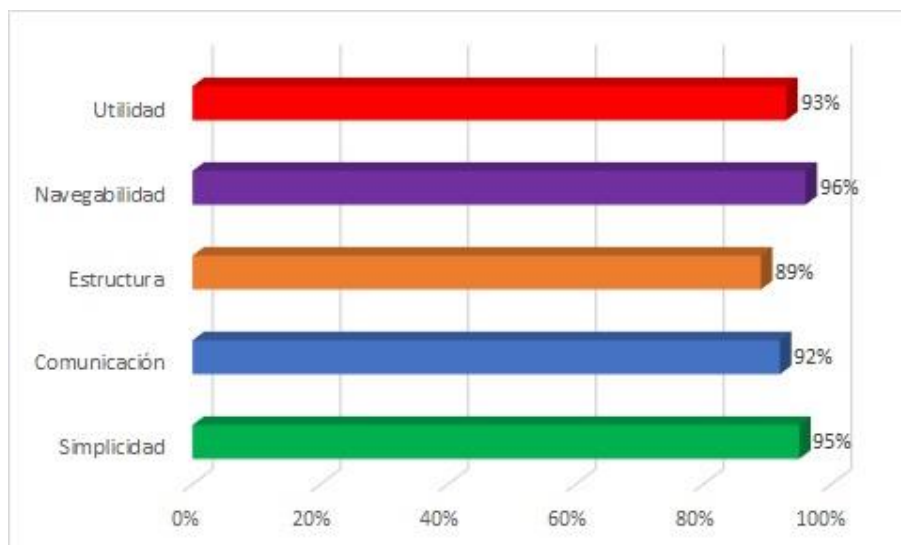


Figura 60: Resultados de entrevista del modelo 4.

Fuente: Autoría Propia

Como podemos observar, nuestros modelos han obtenido mejoras a lo largo de las iteraciones, siendo el primer modelo, la Figura 53, una interfaz monocromática, sin acceso a la información del robot, y la definición de área de podado se trabaja sobre una imagen de un mapa. Seguido de este, tenemos la Figura 55, que representa a nuestro segundo modelo, modelo en el cual, se sincronizan los colores del sistema informático de Opciones con nuestra interfaz gráfica, y se realiza la conexión mediante la API, para el acceso a la información del robot. Después esta nuestro tercer modelo, el cual se puede ver en la Figura 57, en este modelo integramos algunos cuadros de diálogo en las acciones, reemplazamos la imagen del mapa por un mapa satelital y, un botón que permite utilizar la primera versión del sistema de visión artificial, sistema mediante el cual generaba una detección rectangular englobando el área de podado y detección excedente. Finalmente llegamos al último modelo, la Figura 59, donde se integra la última versión del sistema de visión artificial y cuadros de diálogos en todas las acciones para una mejor experiencia de usuario por parte del operario del sistema. Todos estos cambios, en cada iteración, nos permitieron lograr un porcentaje aceptable en los resultados para la utilidad, navegabilidad, estructura, comunicación y simplicidad de nuestra interfaz gráfica, lo cual nos dice, que la experiencia de usuario que brinda nuestra interfaz, es adecuada para el uso en el cumplimiento del monitoreo y definición de áreas de podado sobre el robot GrassBot.

### **6.3. Sistema de Visión Artificial**

#### **6.3.1. Base de Datos**

Para poder aplicar el sistema de visión artificial, se tomaron 10 lugares diferentes que contengan áreas de césped de gran extensión donde el robot GrassBot puede aplicarse.

Para poder hacer una comparativa de la operación manual con la herramienta de visión artificial de la definición de área de podado, se solicitó al operario del sistema, delimitar manualmente el área verde contenida dentro de cada uno de estos diez lugares para, posteriormente, realizar la comparación respectiva.

### 6.3.2. Consideraciones

Una vez ingresados los datos requeridos al sistema, debemos saber cómo obtener el valor de acierto que representará la detección del sistema de visión artificial. El objetivo es lograr una detección lo más próxima al área objetivo, es por esto que, debemos tener en cuenta los casos donde nuestro porcentaje de acierto disminuye:

1. Cuando el área detectada excede la zona a detectar.
2. Cuando el área detectada presenta ausencias de la zona a detectar.
3. Una mezcla de los puntos anteriores, cuando el área detectada excede la zona a detectar en ciertos puntos y, en otros puntos, se presenta ausencia de la zona a detectar.

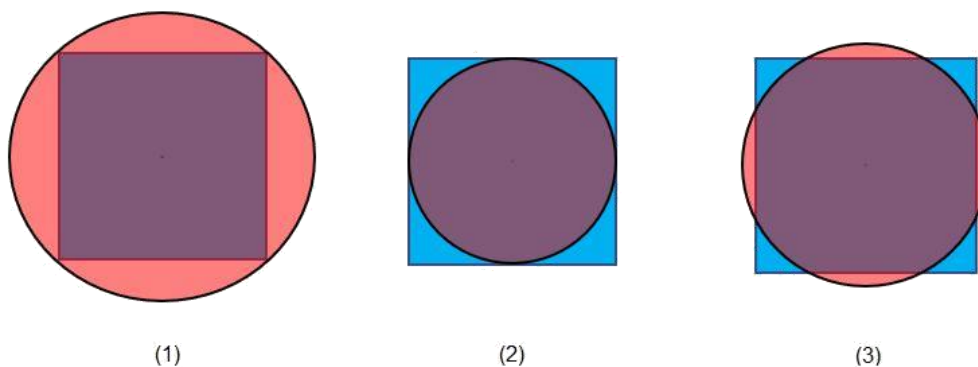


Figura 61: Muestra gráfica de los casos para el cálculo de acierto.

Fuente: Autoría Propia

### 6.3.3. Métricas

Conociendo estos casos, podríamos decir que el porcentaje de acierto del sistema disminuye con las detecciones excedentes y las detecciones faltantes. Para lo cual

convertimos estos dos factores en una sola variable, llamada *detección errónea*, y se representaría a través de (6.1).

$$deteccion\ erronea = \frac{(cantidad\ pixeles\ excedentes + cantidad\ pixeles\ faltantes)}{cantidad\ pixeles\ a\ detectar} * 100\% \quad (6.1)$$

$$Acierto = 100\% - deteccion\ erronea \quad (6.2)$$

Es por esto que, en la medida que el porcentaje de la zona detectada no exceda, por más de un 100 %, el porcentaje de zona a detectar, entonces se podrá calcular la efectividad del sistema de visión artificial.

#### 6.3.4. Resultados

Dentro de las pruebas realizadas, el sistema de visión artificial, identifica las situaciones cuando la geoposición de nuestro robot no se encuentra dentro de una zona de podado, como se muestra en la Figura 62.



Figura 62: Ubicación que no contiene área de podado.

Fuente: Autoría Propia

El sistema de visión artificial, es capaz de detectar zonas de área de podado cuando el robot se encuentra ubicado dentro de dicha zona, como muestra la Figura 63, debido a que la geoposición del robot es de vital importancia para poder realizar la detección.



Figura 63: Ubicación donde se detecta una posible área de podado.

Fuente: Autoría Propia

Posteriormente, probamos el sistema de visión artificial con la información de los 10 lugares diferentes previamente ingresados al sistema, obteniendo resultados de acierto entre un 80 % y 98 %. Este porcentaje de acierto varia porque depende del algoritmo que aplica nuestro sistema de visión artificial y la forma de zona de podado a detectar.

Dentro de nuestras pruebas realizadas, tenemos un caso donde identificamos una zona perteneciente a un parque cementerio, y está representada en la Figura 64, en ella observamos la zona a detectar y la detección manual realizada, respectivamente.



Figura 64: Mapas del caso 01 a analizar y la detección manual del área de podado.

Fuente: Autoría Propia

Al aplicar nuestro sistema de visión artificial, obtenemos un resultado, representado en la Figura 65, y a simple vista es evidente que el resultado contiene diferencias en algunos puntos, en comparativa a la detección manual, esto indica que tenemos una “detección errónea”.



Figura 65: Resultado de la visión artificial en el caso 01.

Fuente: Autoría Propia

Para calcular el porcentaje de la detección errónea, procedemos a realizar la comparativa entre el resultado de ambas detecciones, como se muestra en la Figura 66. En esta comparación, obtenemos la cantidad precisa para la detección errónea y luego calculamos la precisión del sistema de visión artificial con la ecuación (6.2), como se muestra en la Figura 67.

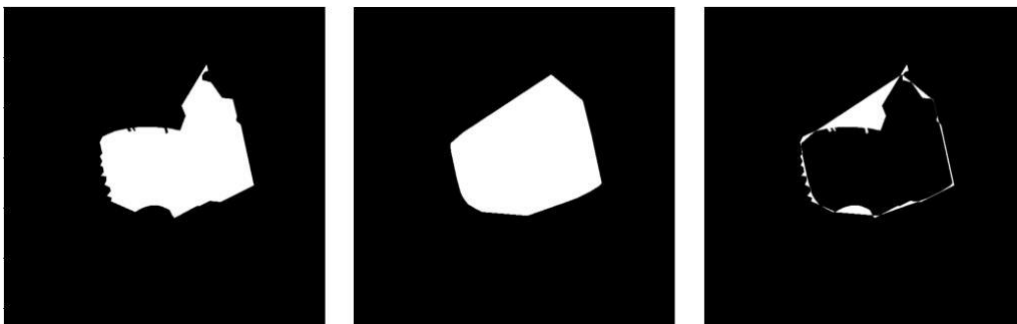


Figura 66: Comparación de resultados del caso 01.

Fuente: Autoría Propia



El porcentaje de Error es: 17.01 %.  
El porcentaje de Acierto es: 82.99 %.



Figura 67: Muestra de acierto del sistema de visión artificial desde la consola del caso.

Fuente: Autoría Propia

Como hemos visto, el porcentaje de acierto que obtuvo nuestro sistema es de 82.99 %, este porcentaje se ve reducido por detecciones excedentes, en su mayoría, excedentes que se relacionan con el algoritmo que aplicamos y la superficie de la zona.

Como siguiente caso, tenemos la Figura 68, en la cual identificamos una cancha de fútbol, y a simple vista identificamos algunos puntos de la superficie donde no hay convexidad, lo cual es desfavorable para nuestro sistema de visión artificial.



Figura 68: Mapas del caso 02 a analizar y la detección manual del área de podado.

Fuente: Autoría Propia

En la Figura 69, observamos que, el resultado de la detección por parte del sistema de visión artificial, en ciertos puntos, ubicamos detección excedente y en otros puntos encontramos detección faltante. En esta situación, al igual que en el caso anterior, la detección excedente se da por el algoritmo aplicado por nuestro sistema, y la detección faltante se da, porque ciertos píxeles dentro del área de podado, no cumplen con las características adecuadas para considerarse verde dentro del espectro.



Figura 69: Resultado de la visión artificial del caso 02.

Fuente: Autoría Propia

Para comprobar la existencia de píxeles excedente y faltante, realizamos la comparación entre los resultados, y conseguimos una pequeña cantidad de píxeles que no forman parte del área a detectar, la cual podemos observar en la Figura 70.



Figura 70: Comparación de resultados del caso 02.

Fuente: Autoría Propia

A pesar de encontrar píxeles excedentes y faltantes, la detección errónea representa un menor porcentaje de error, en comparación al caso anterior. Siendo de este modo, que obtenemos un porcentaje de acierto del 90.83 %, como se muestra en la Figura 71, y esto se debe, a que la superficie del área a detectar, tiene menor cantidad de puntos no convexos.



Figura 71: Muestra de acierto de la visión artificial desde la consola del caso 02.

Fuente: Autoría Propia

En un siguiente caso, en la Figura 72, tenemos una ubicación en donde podemos identificar un estadio y el área verde que lo conforma. Según lo descrito en los casos anteriores, la superficie del área verde presenta una convexidad por todos sus puntos.



Figura 72: Mapas del caso 03 a analizar y la detección manual del área de podado.

Fuente: Autoría Propia

En la Figura 73, mostramos el resultado al aplicar nuestro sistema de visión artificial, y en ella podemos percibir una gran similitud con la detección manual vista en la Figura 72, la diferencia es que, la detección por parte de nuestro sistema, muestra una mayor cantidad de vértices en el dibujo.

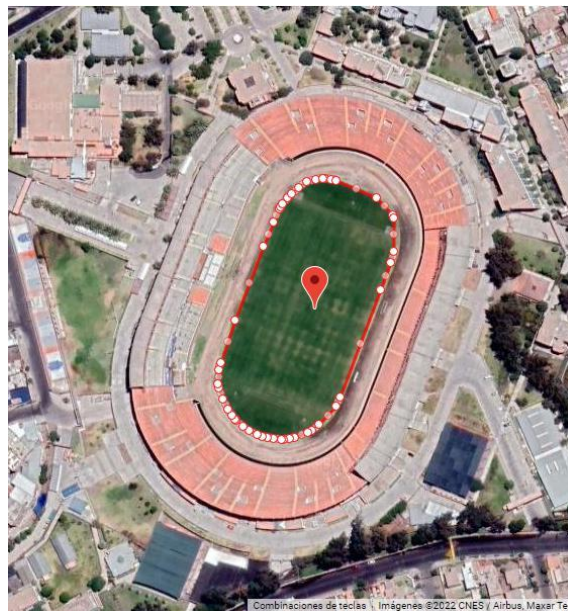


Figura 73: Resultado de la visión artificial en el caso 03.

Fuente: Autoría Propia

Los resultados de nuestro sistema de visión artificial son comparados con la detección manual, como se muestra en la Figura 74. Como podemos observar, el área a detectar presenta una convexidad en su superficie, lo cual se ajusta adecuadamente con el algoritmo aplicado por el sistema, y por ello obtenemos un porcentaje de acierto del 97.25 %, indicado en la Figura 75.

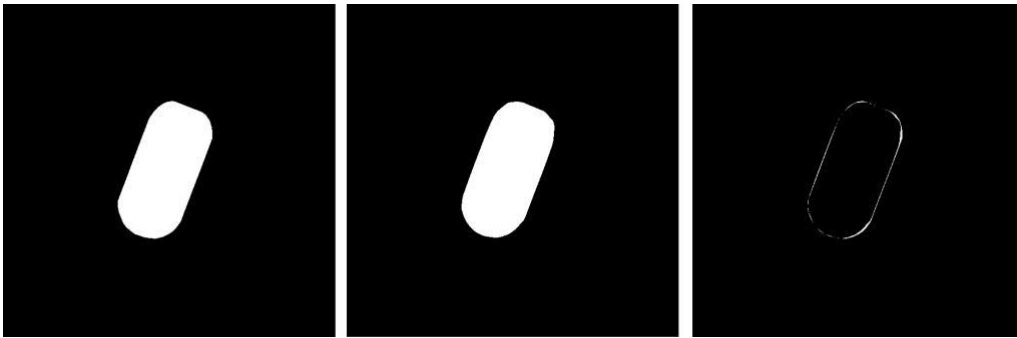


Figura 74: Comparación de resultados del caso 03.

Fuente: Autoría Propia



Figura 75: Muestra de acierto de la visión artificial desde la consola del caso 03.

Fuente: Autoría Propia

### 6.3.5. Otro Algoritmo Aplicado en el Sistema de Visión Artificial

Debido a las diversas formas que puede tener un área de podado, hay casos en los que el algoritmo del casco convexo, no es el más óptimo para obtener un porcentaje de detección alto. Por esta razón, probamos en nuestro sistema de visión artificial, la implementación del algoritmo “aproximación de contorno”, y procedimos a realizar una comparativa de los resultados arrojados, con el algoritmo de casco convexo y la detección temprana realizada por el operario del sistema, obteniendo los resultados que se detallan en la Tabla 4.

Caso	Detección Temprana		Casco Convexo		Aprox. de Contorno	
	% de Error	% de Acierto	% de Error	% de Acierto	% de Error	% de Acierto
01	17.28 %	82.72 %	06.07 %	93.93 %	17.63 %	82.37 %
02	07.84 %	92.16 %	02.69 %	97.31 %	07.79 %	92.21 %
03	06.27 %	93.73 %	02.75 %	97.25 %	06.55 %	93.45 %
04	08.49 %	91.51 %	17.01 %	82.99 %	09.62 %	90.38 %
05	12.97 %	87.03 %	07.56 %	92.44 %	11.80 %	88.20 %
06	23.40 %	76.60 %	08.17 %	91.83 %	23.37 %	76.63 %
07	11.65 %	88.35 %	10.71 %	89.29 %	11.47 %	88.53 %
08	18.43 %	81.57 %	18.48 %	81.52 %	17.91 %	82.09 %
09	17.73 %	82.27 %	12.50 %	87.50 %	17.03 %	82.97 %
10	10.99 %	89.01 %	09.17 %	90.83 %	10.97 %	89.03 %

Tabla 4: Comparativa de resultados entre diferentes algoritmos.

Fuente: Autoría Propia.

Los resultados tras aplicar un algoritmo dentro de nuestro sistema, nos arroja una mayor probabilidad de acierto, en la mayoría de nuestros casos y, a su vez, podemos ver que el algoritmo de casco convexo nos permite obtener probabilidades de acierto más altas relativas a la convexidad de las superficies de las áreas de podado, que suelen tener las zonas donde el robot autónomo GrassBot se puede aplicar. Por otro lado, el algoritmo de aproximación de contorno, nos

devuelve mayores resultados en los casos donde la superficie del área de podado no presenta una convexidad.



## **CONCLUSIONES**

**Primera**, se desarrolló una interfaz gráfica, donde se muestran los valores de los diferentes sensores y estados del robot autómatas GrassBot, esto a través de una conexión por API a un sistema brindado por la empresa Opciones, de tal manera que se puede continuar con el monitoreo del robot autómatas mientras se realiza la tarea de definición de área de podado.

**Segunda**, considerando, que tomamos en cuenta los recursos de acceso libre que nos ofrece Google, se integró “Maps Static API”, para obtener las imágenes de mapas satelitales que se utilizan en el sistema de visión artificial, y “Maps JavaScript API”, para obtener los mapas satelitales de la geoposición del robot GassBot, y que son mostrados en la interfaz gráfica, con la finalidad de ubicar al robot y las áreas de podado.

**Tercera**, se desarrolló un apartado exclusivo para la definición del área de podado sobre mapas, gracias a la integración de “Maps Javascript API”, en donde se le habilita al operario del sistema, la capacidad de realizar el trazado de dicha área, conformado por diversos puntos geográficos; siendo así, que se permite guardar la información de la ubicación geográfica de cada punto dibujado y poder consultarlo a lo largo del tiempo.

**Cuarta**, se desarrolló e implementó un sistema de visión artificial a la interfaz gráfica, como una herramienta adicional para la definición de área de podado, logrando la automatización

de esta tarea al operario del sistema que, de acuerdo a la forma del área de podado que se requiera definir, puede significar un mayor tiempo para la realización de la misma.

**Quinta**, se recopiló la información de geoposición sobre 10 ubicaciones diferentes, que coinciden con los fines comerciales del GrassBot, y se realizaron comparaciones entre la detección manual realizada, por el operario del sistema, y la detección obtenida por el sistema de visión artificial. Se aplicaron diferentes algoritmos de detección y se escogió el algoritmo del casco convexo para el análisis de resultados, el cual nos arrojó el menor porcentaje de error por emisión de detección dentro de las ubicaciones tomadas.

## **RECOMENDACIONES**

**Primera:** Contratar una API que brinden imágenes satelitales con una resolución mayor y de alta calidad, para poder realizar un mejor procesamiento de las imágenes y una mejor detección de las áreas deseadas. Del mismo modo, poder tener acceso a APIs que brinden imágenes satelitales actualizadas, o tener la capacidad de acceder a las imágenes satelitales sobre una fecha específica.

**Segunda:** Mejorar el sistema de visión artificial con algoritmos adaptativos, para que pueda identificar la convexidad de la superficie del área a detectar, con la finalidad de aplicar el algoritmo adecuado a la superficie, y obtener un mejor resultado de detección. También podemos llevar el sistema de visión artificial a un nivel más inteligente, para ello, se podrían revisar diversos algoritmos de inteligencia artificial, como por ejemplo la biblioteca UNet de Python, esta biblioteca nos brinda la capacidad realizar segmentación utilizando redes neuronales. Aplicar este tipo de algoritmos, con un buen entrenamiento, podrían garantizar una mayor precisión al momento de segmentar las áreas requeridas, y así lograr una mejor detección.

**Tercera:** Usar el sistema de visión artificial propuesto para exploración mediante drones, comparativa a través del tiempo de ciertas zonas o detección de otro tipo de áreas.

**Cuarta:** El uso de un sistema GPS de mayor calidad, o implementar un Sistema de Posicionamiento Local, para incrementar la exactitud de este sensor. En los modelos comerciales este error es de  $\pm 15$  metros típicamente.

**Quinta:** Es posible que la percepción de mejora en la interfaz se deba, no solo a la mejora iterativa de la interfaz basada en la retroalimentación dada por el equipo técnico, sino que haya sucedido la familiarización de la interfaz por el equipo técnico. Sería ideal hacer un estudio de comparación, con un grupo de control que vea la interfaz gráfica siempre por primera vez, para corroborar este hecho.

## ANEXO A

### REQUERIMIENTOS FUNCIONALES

RF-001	La interfaz gráfica debe permitir visualizar un listado de los robots disponibles.
Versión	1.0 - 15/10/2020
Dependencias	El sistema debe conectarse a la API de la empresa Opciones.
Descripción	<ul style="list-style-type: none"><li>• El sistema muestra los robots disponibles obtenidos en una lista.</li></ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 5: Requerimiento funcional RF-001.

Fuente: Autoría Propia.

RF-002	La interfaz gráfica debe mostrar una vista previa de la ubicación del robot seleccionado en un mapa.
Versión	1.0 - 15/10/2020
Dependencias	El sistema debe conectarse a la API de la empresa Opciones.
Descripción	<ul style="list-style-type: none"><li>• El sistema muestra la geoposición del robot a través de un mapa, obtenidos desde el sensor de GPS.</li></ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 6: Requerimiento funcional RF-002.

Fuente: Autoría Propia.

RF-003	La interfaz gráfica debe permitir visualizar la información de los diferentes sensores que el robot registra en la API de la nube.
Versión	1.0 - 15/10/2020
Dependencias	El sistema debe poder conectarse a la API de la empresa Opciones.
Descripción	<ul style="list-style-type: none"> <li>El sistema muestra la información de todos los sensores que el robot posee.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 7: Requerimiento funcional RF-003.

Fuente: Autoría Propia.

RF-004	La interfaz gráfica debe tener herramientas de dibujo sobre un mapa para indicar el área de podado.
Versión	1.0 - 15/10/2020
Dependencias	El usuario debe seleccionar un robot.
Descripción	<ul style="list-style-type: none"> <li>El sistema muestra las herramientas necesarias para determinar el área de podado, sobre un mapa.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 8: Requerimiento funcional RF-004.

Fuente: Autoría Propia.

RF-005	La interfaz gráfica debe tener una herramienta que aplica visión artificial para la detección de áreas de podado.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>El sistema debe realizar una autodetección sobre el mapa, para determinar si existe un área de podado y, la dibuja sobre dicho mapa.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 9: Requerimiento funcional RF-005.

Fuente: Autoría Propia.

RF-006	La interfaz gráfica debe permitir guardar la información del área de podado dibujada en el mapa.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>• El sistema debe realizar una autodetección sobre el mapa, para determinar si existe un área de podado y, la dibuja sobre dicho mapa.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 10: Requerimiento funcional RF-006.

Fuente: Autoría Propia.

## **ANEXO B**

### **REQUERIMIENTOS NO FUNCIONALES**

RNF-001	La interfaz gráfica será desarrollada en la arquitectura MVC.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"><li>• El sistema estará desarrollado bajo la arquitectura Modelo-Vista-Controlador.</li></ul>
Importancia	Media
Prioridad	Media
Estado	Finalizado
Comentarios	Ninguno

Tabla 11: Requerimiento no funcional RNF-001.

Fuente: Autoría Propia.

RNF-002	La interfaz gráfica debe tener UX simple.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"><li>• El Front-End del sistema tendrá características simples de UX.</li></ul>
Importancia	Media
Prioridad	Media
Estado	Finalizado
Comentarios	Ninguno

Tabla 12: Requerimiento no funcional RNF-002.

Fuente: Autoría Propia.

RNF-003	La interfaz gráfica debe tener las características de simplicidad, comunicación, estructura, navegabilidad y utilidad.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"><li>• El sistema de interfaz gráfica contará con las características de simplicidad, comunicación, estructura, navegabilidad y utilidad.</li></ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Características utilizadas para medir la satisfacción de interacción por parte del operario del sistema con la interfaz gráfica.

Tabla 13: Requerimiento no funcional RNF-003

Fuente: Autoría Propia.



RNF-004	La interfaz gráfica debe ser amigable, ligera, dinámica y multiplataforma.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>• El Front-End del sistema tendrá las propiedades: amigable, ligera, dinámica y multiplataforma.</li> </ul>
Importancia	Media
Prioridad	Media
Estado	Finalizado
Comentarios	Ninguno

Tabla 14: Requerimiento no funcional RNF-004.

Fuente: Autoría Propia.

RNF-005	La interfaz gráfica está implementada en HTML5, CSS3, JavaScript y Python.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>• El Front-End del sistema deberá estar programado en HTML5, CSS3 y JavaScript.</li> <li>• El Back-End del sistema deberá estar programado en Python.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 15: Requerimiento no funcional RNF-005.

Fuente: Autoría Propia.

RNF-006	La interfaz gráfica debe funcionar con CentOS 7.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>• El sistema deberá correr desde CentOS 7.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 16: Requerimiento funcional RNF-006.

Fuente: Autoría Propia.

RNF-007	El sistema de visión artificial debe estar implementado en Python.
Versión	1.0 - 15/10/2020
Dependencias	Ninguno
Descripción	<ul style="list-style-type: none"> <li>• El sistema de visión artificial, que se integrará al sistema, deberá estar implementado en Python.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 17: Requerimiento funcional RNF-007.

Fuente: Autoría Propia.

RNF-008	El sistema de visión artificial debe trabajar con una API gratuita que nos permita mapas (API Maps JavaScript de Google).
Versión	1.0 - 15/10/2020
Dependencias	El encargado del sistema debe registrar su cuenta en Google Cloud para obtener una llave de acceso a esta API.
Descripción	<ul style="list-style-type: none"> <li>• El sistema deberá integrar la API Maps JavaScript de Google, para poder visualizar mapas.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 18: Requerimiento funcional RNF-008.

Fuente: Autoría Propia.

RNF-009	El sistema de visión artificial debe trabajar con una API gratuita que nos permita obtener imágenes estáticas de los mapas (API Maps Static de Google).
Versión	1.0 - 15/10/2020
Dependencias	El encargado del sistema debe registrar su cuenta en Google Cloud para obtener una llave de acceso a esta API.
Descripción	<ul style="list-style-type: none"> <li>El sistema deberá integrar la API Maps Static de Google, para trabajar juntamente con el sistema de visión artificial.</li> </ul>
Importancia	Alta
Prioridad	Alta
Estado	Finalizado
Comentarios	Ninguno

Tabla 19: Requerimiento funcional RNF-009.

Fuente: Autoría Propia.

## ANEXO C

### ESPECIFICACIÓN DE CASOS DE USO

AT-001	Usuario
Versión	1.0 - 15/10/2020
Descripción	<ul style="list-style-type: none"><li>Este actor del sistema es el que representa el rol de operario del sistema, que interactuará directamente con la interfaz gráfica.</li></ul>
Comentarios	Ninguno.

Tabla 20: Actor AT-001.

Fuente: Autoría Propia.

ID001	Visualiza robots.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Acceder al sistema web.
Precondición	Ninguno
Flujo Principal	<ul style="list-style-type: none"><li>El usuario accede a la vista y el sistema muestra un listado de los robots activos.</li></ul>
Postcondición	El sistema debe estar en una red interna o externa.

Tabla 21: Caso de uso ID001.

Fuente: Autoría Propia.

ID002	Solicitud de nombres de todos los robots.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID001.
Precondición	El usuario debe acceder a la vista de listado de robots.
Flujo Principal	<ul style="list-style-type: none"><li>El sistema se conecta a una API para obtener los datos de los robots activos.</li></ul>
Postcondición	Ninguno.

Tabla 22: Caso de uso ID002.

Fuente: Autoría Propia.

ID003	Selecciona robot.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID001.

Precondición	Ninguno.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario selecciona un robot de la lista y el sistema muestra su geoposición en un mapa y los datos de sus diferentes sensores.</li> </ul>
Postcondición	El usuario puede monitorear al robot.

Tabla 23: Caso de uso ID003.

Fuente: Autoría Propia.

ID004	Solicitud de información del robot.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID003.
Precondición	El usuario debe seleccionar un robot de la lista de robots.
Flujo Principal	<ul style="list-style-type: none"> <li>El sistema se conecta a una API y solicita la información del robot seleccionado.</li> </ul>
Postcondición	Ninguno.

Tabla 24: Caso de uso ID004.

Fuente: Autoría Propia.

ID005	Ir a la vista de definición de área de podado.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID003.
Precondición	El usuario debe seleccionar un robot de la lista de robots.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario usa un botón para ir a la vista de la definición de área de podado y el sistema lo redirecciona.</li> </ul>
Postcondición	El usuario puede definir el área de podado para el robot seleccionado.

Tabla 25: Caso de uso ID005.

Fuente: Autoría Propia.

ID006	Nuevo Dibujo.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID005.
Precondición	Ninguno.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario selecciona el botón para realizar un nuevo dibujo y el sistema habilita al usuario poder dibujar vértices sobre el mapa.</li> </ul>
Postcondición	Ninguno.

Tabla 26: Caso de uso ID006.

Fuente: Autoría Propia.

ID007	Limpieza de Mapa
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID006.
Precondición	El usuario debe usar la herramienta "Borrar Vértices".
Flujo Principal	<ul style="list-style-type: none"> <li>• El sistema limpia el mapa para que el usuario pueda dibujar vértices.</li> </ul>
Postcondición	Ninguno.

Tabla 27: Caso de uso ID007.

Fuente: Autoría Propia.

ID008	Eliminar vértices.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID005.
Precondición	Ninguno.
Flujo Principal	<ul style="list-style-type: none"> <li>• El usuario selecciona el botón para eliminar vértices y el sistema habilita al usuario seleccionar vértices para su eliminación.</li> </ul>
Postcondición	Ninguno.

Tabla 28: Caso de uso ID008.

Fuente: Autoría Propia.

ID009	Seleccionar vértice.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID008.
Precondición	El usuario debe habilitar la opción para Eliminar vértices.
Flujo Principal	<ul style="list-style-type: none"> <li>• El usuario selecciona un vértice y el sistema muestra la opción para eliminar dicho vértice.</li> </ul>
Postcondición	El usuario podrá seleccionar la opción para eliminar el vértice seleccionado.

Tabla 29: Caso de uso ID009.

Fuente: Autoría Propia.

ID010	Eliminar vértice seleccionado del mapa.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID009.

Precondición	El usuario debe seleccionar sobre la opción desplegada para eliminar el vértice.
Flujo Principal	<ul style="list-style-type: none"> <li>El sistema elimina el vértice seleccionado.</li> </ul>
Postcondición	El usuario podrá seleccionar la opción para eliminar el vértice seleccionado.

Tabla 30: Caso de uso ID010

Fuente: Autoría Propia.

ID011	Guardar Cambios.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID005.
Precondición	El usuario debe dibujar un vértice sobre el mapa como mínimo.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario selecciona el botón guardar y el sistema guarda los datos de los vértices dibujados en el mapa.</li> </ul>
Postcondición	Ninguno.

Tabla 31: Caso de uso ID011

Fuente: Autoría Propia.

ID012	Guarda geoposición de los vértices.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID011.
Precondición	El usuario debe usar el botón guardar.
Flujo Principal	<ul style="list-style-type: none"> <li>El sistema se conecta a una API y envía la geoposición de cada vértice dibujado sobre el mapa para el guardado respectivo.</li> </ul>
Postcondición	El usuario podrá seleccionar la opción para eliminar el vértice seleccionado.

Tabla 32: Caso de uso ID012

Fuente: Autoría Propia.

ID013	Ir a la vista de listado de robots.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID005.
Precondición	Ninguno.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario usa un botón para ir a la vista de listado de robots y el sistema lo redirecciona.</li> </ul>
Postcondición	Ninguno.

Tabla 33: Caso de uso ID013

Fuente: Autoría Propia.

ID014	Detección por visión artificial.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID005.
Precondición	Ninguno.
Flujo Principal	<ul style="list-style-type: none"> <li>El usuario usa la herramienta para la detección por visión artificial y el sistema dibuja vértices sobre el mapa englobando un área de podado detectada.</li> </ul>
Postcondición	Ninguno.

Tabla 34: Caso de Uso ID014

Fuente: Autoría Propia.

ID015	Obtención de geoposición.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID014.
Precondición	El usuario debe usar la herramienta para detección por visión artificial.
Flujo Principal	<ul style="list-style-type: none"> <li>El sistema obtiene la geoposición del mapa.</li> </ul>
Postcondición	Ninguno.

Tabla 35: Caso de uso ID015.

Fuente: Autoría Propia.

ID016	Obtención de imagen de geoposición.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID015.
Precondición	El usuario debe usar la herramienta para detección por visión artificial.
Flujo Principal	<ul style="list-style-type: none"> <li>El sistema solicita a la API de Google, una imagen de la geoposición del mapa.</li> </ul>
Postcondición	Ninguno.

Tabla 36: Caso de uso ID016.

Fuente: Autoría Propia.

ID017	Procesamiento y análisis de imagen.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID016.
Precondición	El usuario debe usar la herramienta para detección por visión artificial.



Flujo Principal	<ul style="list-style-type: none"> <li>• El sistema aplica diversos algoritmos para detectar un área de podado y los vértices más relevantes del área.</li> </ul>
Postcondición	Ninguno.

Tabla 37: Caso de uso ID017.

Fuente: Autoría Propia.

ID018	Conversión de píxeles a coordenadas geodésicas.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID017.
Precondición	El usuario debe usar la herramienta para detección por visión artificial.
Flujo Principal	<ul style="list-style-type: none"> <li>• El sistema transforma los píxeles obtenidos a coordenadas geodésicas mediante las fórmulas requeridas por el sistema geodésico mundial EPSG:900913 Google Maps Global Mercator.</li> </ul>
Postcondición	Ninguno.

Tabla 38: Caso de uso ID018.

Fuente: Autoría Propia.

ID019	Seleccionar vértice.
Versión	1.0 - 15/10/2020
Actores	Usuario
Dependencias	Caso de uso ID018.
Precondición	El usuario debe usar la herramienta para detección por visión artificial.
Flujo Principal	<ul style="list-style-type: none"> <li>• El sistema ubica y dibuja las coordenadas obtenidas sobre el mapa de la vista para la definición de área de podado.</li> </ul>
Postcondición	Ninguno.

Tabla 39: Caso de uso ID019.

Fuente: Autoría Propia.

## ANEXO D

### CÓDIGO FUENTE

#### Conversión para sistemas geodésicos mundiales

```
1 def _lat_lng_to_meters(self, lat, lng):
2     "Converts given lat/lng in WGS84 Datum to XY in Spherical
3     Mercator EPSG:900913"
4
5     meter_x = lng * self.originShift / 180.0
6     meter_y = math.log( math.tan((90 + lat) * math.pi / 360.0 )) /
7     (math.pi / 180.0)
8     meter_y = meter_y * self.originShift / 180.0
9     return round(meter_x, 8), round(meter_y, 8)
10
11 def _meters_to_pixels(self, meter_x, meter_y, zoom):
12     "Converts EPSG:900913 to pyramid pixel coordinates in given zoom
13     level"
14
15     resolution = self.initialResolution / math.pow(2, zoom)
16     pixel_x = (meter_x + self.originShift) / resolution
17     pixel_y = (meter_y + self.originShift) / resolution
18     return pixel_x, pixel_y
19
20 def _pixels_to_meters(self, pixel_x, pixel_y, zoom):
21     "Converts pixel coordinates in given zoom level of pyramid to
22     EPSG:900913"
23
24     resolution = self.initialResolution / math.pow(2, zoom)
25     meter_x = pixel_x * resolution - self.originShift
26     meter_y = pixel_y * resolution - self.originShift
27     return round(meter_x, 8), round(meter_y, 8)
28
29 def _meters_to_lat_lng(self, meter_x, meter_y):
30     "Converts XY point from Spherical Mercator EPSG:900913 to lat/lng
31     in WGS84 Datum"
32
33     lng = (meter_x / self.originShift) * 180.0
34     lat = (meter_y / self.originShift) * 180.0
35
36     lat = 180 / math.pi * (2 * math.atan(math.exp(lat * math.pi /
37     180.0)) - math.pi / 2.0)
38     return round(lat, 8), round(lng, 8)
39
40 def _pruning_area_detector_coords(self, lat, lng, hull, centroid,
41 zoom):
42     meter_x, meter_y = self._lat_lng_to_meters(lat, lng)
43
44     pixel_x, pixel_y = self._meters_to_pixels(meter_x, meter_y, zoom)
45
```

```

46 pixel_x_cent = pixel_x + centroid[1] - 319
47 pixel_y_cent = pixel_y - (centroid[0] - 319)
48
49 lat_lng_hull = []
50 for i in hull:
51     p_x, p_y = pixel_x_cent, pixel_y_cent
52
53     p_x += i[0, 0] - centroid[1]
54     p_y -= i[0, 1] - centroid[0]
55
56     m_x, m_y = self._pixels_to_meters(p_x, p_y, zoom)
57     lat, lng = self._meters_to_lat_lng(m_x, m_y)
58
59     lat_lng_hull.append({
60         'lat': lat,
61         'lng': lng,
62     })
63 return lat_lng_hull

```

## Sistema de visión artificial

```

1 def _filter_green_color(self, image_rgb):
2     image_hsv = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2HSV)
3
4     mask = cv2.inRange(image_hsv, (36, 25, 25), (86, 255, 255)) #
5 Verde en el Espectro
6
7     imask = mask > 0
8     filtered_image = np.zeros_like(image_rgb, np.uint8)
9     filtered_image[imask] = image_rgb[imask]
10    filtered_image = cv2.cvtColor(filtered_image, cv2.COLOR_HSV2RGB)
11    filtered_image = cv2.cvtColor(filtered_image, cv2.COLOR_RGB2GRAY)
12    return filtered_image
13
14 def _pruning_area_detector_pixels(self, image_name, size=640,
15 scale=1):
16     # Read Image
17     image_bgr = cv2.imread(self.IMAGE_DIRECTORY + image_name)
18     image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)
19
20     # Filtering Green Color HSV and Return in Gray
21     filtered_image = self._filter_green_color(image_rgb)
22
23     # Applying THRESHHOLD to Binarize
24     th, im_th = cv2.threshold(filtered_image, 0, 255,
25 cv2.THRESH_BINARY)
26
27     # Applying DILATE 'cause some satellite images not the same
28     kernel = np.ones((3, 3), np.uint8)
29     # im_th = cv2.dilate(im_th, kernel, 1)
30

```

```

31 # Get Fill and Apply fill inv in mask
32 im_floodfill = im_th.copy()
33 h, w = im_th.shape[:2]
34 mask = np.zeros((h+2, w+2), np.uint8)
35
36 # Setting a square with 0
37 im_floodfill[0:10, :] = 0
38 im_floodfill[size-10:size, :] = 0
39 im_floodfill[:, 0:10] = 0
40 im_floodfill[:, size-10:size] = 0
41
42 cv2.floodFill(im_floodfill, mask, (0, 0), 255)
43 im_floodfill_inv = cv2.bitwise_not(im_floodfill)
44 im_out = im_th | im_floodfill_inv
45
46 # Deleting Principal Noise
47 test = cv2.erode(im_out, kernel, 9)
48 test = cv2.dilate(test, kernel, 9)
49
50 # Filling Principal Area
51 test = cv2.morphologyEx(test, cv2.MORPH_GRADIENT, kernel)
52 coord = int(size * scale / 2 - 1)
53 cv2.floodFill(test, None, (coord, coord), (255, 255, 255))
54
55 # Deleting Minimal Noise
56 kernel = np.ones((5, 5), np.uint8)
57 test = cv2.erode(test, kernel, 3)
58
59 # Get Hull Coords
60 contours, hierarchy = cv2.findContours(test, cv2.RETR_TREE,
61 cv2.CHAIN_APPROX_SIMPLE)
62 hull = []
63 for i in range(len(contours)):
64     hull.append(cv2.convexHull(contours[i], False))
65
66 if len(hull[0]) > 4:
67     # Get Object Centroid
68     label_img = label(test, connectivity=test.ndim)
69     props = regionprops(label_img)
70     pixel_x, pixel_y = int(props[0].centroid[0]),
71 int(props[0].centroid[1])
72     # print(hull[0], [pixel_x, pixel_y])
73     return hull[0], [pixel_x, pixel_y]
74 return [], []

```

**ANEXO E**  
**DOCUMENTOS**

**Carta de compromiso para entidad involucrada en Tesis**

Lima, 27 de Octubre de 2020.

La empresa OPCIONES, INGENIERÍA Y TECNOLOGÍA AMBIENTAL S.R.L. con Ruc. Nro. 20455796891, conforme lo establecido en el artículo 5.1 del Reglamento de Grado Académico de Bachiller y Título Profesional de la Universidad Tecnológica del Perú (la "UTP") y dentro del marco de los intereses de la UTP de favorecer acciones de responsabilidad social universitaria con diversas instituciones de la sociedad peruana, se dirige a la universidad para solicitar su contribución en la búsqueda de una solución al siguiente problema:

Desarrollar una interfaz gráfica que permita definir el área de corte de césped  
de una robot autónomo.

(el "Problema").

El Problema constituye un tema pertinente y actual en nuestra institución que aún no ha sido resuelto y no forma parte de ningún proyecto en vías de implementación. Siendo de nuestro interés incluir el Problema en el plan de trabajo para la titulación de la tesis denominado:

Asistencia de Deep Learning para definir el área de podado de césped en  
una interfaz gráfica.

Cuyo(s) autor(es) es(son):

Nombres y Apellidos	Carrera
Leandro Gustavo Ortega Palacios	Ingeniería de Sistemas e Informática.

Agradeciendo de antemano la contribución de la UTP en la solución del Problema, nos comprometemos a brindar la información de nuestra empresa que se requiera para el desarrollo de este trabajo, la misma que sólo podrá ser utilizada para fines estrictamente académicos vinculados al trabajo. Declaramos conocer que por disposiciones legales, la tesis será de público conocimiento luego de dos años de su sustentación.

Cordialmente,

Nombres y apellidos del representante de la institución:

Leather Edson Delgado Paredes

Cargo que ocupa: Gerente General  
D.N.I. 41298552

Firma y sello:   
Leather E. Delgado Paredes  
GERENTE GENERAL

## ENTREVISTA

La presente entrevista tiene como objetivo analizar los diferentes indicadores de la variable de investigación "Interfaz Gráfica", a través de preguntas que se han elaborado en base al sistema desarrollado que tiene como objetivo automatizar el proceso y facilitar las operaciones del mantenimiento del césped para el sistema informático del proyecto GrassBot.

Este documento va dirigido al ingeniero Hugo Barreda, quien es el supervisor y validador previo del trabajo realizado frente a la empresa OPCIONES, mediante convenio con la Universidad Tecnológica del Perú.

A continuación, se presenta un conjunto de preguntas para ser valoradas de acuerdo con la propia experiencia y teniendo en cuenta la siguiente escala:

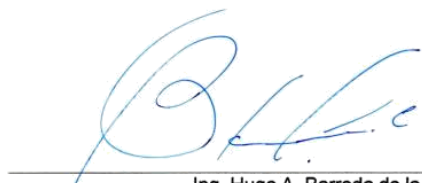
<b>Muy bajo</b>	<b>Bajo</b>	<b>Intermedio</b>	<b>Alto</b>	<b>Muy alto</b>
1	2	3	4	5

**Lea atentamente cada ítem y responda con sinceridad, recuerde que esto nos ayudará a mejorar la interfaz gráfica desarrollada.**

N°	ITEM	VALORACIÓN				
		1	2	3	4	5
1	¿Puede orientarse fácilmente dentro de la interfaz?					X
2	¿Puede ubicar la información requerida fácilmente?					X
3	¿La interfaz detalla toda la información necesaria para el cumplimiento de sus operaciones?					X
4	¿Se le informa correctamente sobre cada acción a realizar?				X	
5	¿El diseño de la interfaz es atractivo?					X
6	¿Le resulta fácil navegar por la interfaz?					X
7	¿Encuentra usted dinámica la interfaz para desplazarse entre las páginas?					X
8	¿El diseño de la interfaz está distribuido correctamente?				X	
9	¿Considera entendibles los cuadros de diálogo desplegados por la interfaz?					X
10	¿La interfaz ofrece las funcionalidades necesarias para cubrir las operaciones a realizar?				X	
11	¿Considera que la interfaz es intuitiva?					X
12	¿Considera que la información mostrada es veráz y actual?					X

BC

13	¿Considera que la herramienta de visión artificial brindada automatiza la operación de definición de área de podado?				X	
14	¿Considera que son útiles las diferentes herramientas que brinda la interfaz para el desarrollo de sus operaciones?					X




---

Ing. Hugo A. Barreda de la Cruz  
 Coordinador Académico  
 Convenio de Asociación para ejecución del Proyecto "Mejora e ingeniería optimizada para desarrollo de un prototipo autónomo y auto sostenible de diagnóstico y mantenimiento de áreas verdes en la región Arequipa. (Convenio N° 320-INNOVATEPERU-PIEC1-2019)

## **BIBLIOGRAFÍA**

- [1] F. R. C. Aguayo and R. M. B. Macias, "Estado del arte de robótica en la agricultura" 2018.
- [2] E. I. al día S.A.C. Opciones ingeniería y tecnología ambiental presenta el prototipo de su robot podador. [Online]. Available: <https://www.industriaaldia.com/noticias/146-opciones-ingenieria-y-tecnologia-ambiental-presenta-el-prototipo-de-su-robot-podador>.
- [3] Garcia. Automatización de procesos: Qué es y por qué deberías pensar en hacerlo. [Online]. Available: <https://trends.inycom.es/automatizacion-de-procesos-que-es-y-por-que-deberias-pensar-en-hacerlo/>
- [4] J. N. G. de Brito, "Manipulador robótico para poda automática (proyecto romovi)," 2018.
- [5] M. A. Muñoz Chacón, "Cortadora de césped automatizada," 2019.
- [6] E. B. . M. School. Geoposicionamiento: que es, principales tecnologías y formas de uso. [Online]. Available: <https://www.esic.edu/rethink/marketing-y-comunicacion/geoposicionamiento-que-es-principales-tecnologias-y-formas-de-uso>
- [7] N. G.-I. Agency. Office of geomatics. [Online]. Available: <https://earth-info.nga.mil/index.php?action=home>
- [8] G. Developers. Maps javascript api, google developers. [Online]. Available: <https://developers.google.com/maps/documentation/javascript/overview?hl=es-419>
- [9] Maps static api, google developers. [Online]. Available: <https://developers.google.com/maps/documentation/maps-static/overview?hl=es-419>.
- [10] S. L. E. Rafael C. Gonzalez and R. E. Woods, "Digital image processing using matlab," 2003.
- [11] N. A. Dobernack, "Implementación de un sistema de detección de señales de tráfico mediante visión artificial basado en fpga," Universidad de Sevilla, Sevilla, España. [Online] Available: [http://bibing.us.es/proyectos/abreproy/12112/fichero/Documentos\\_por\\_capitulos](http://bibing.us.es/proyectos/abreproy/12112/fichero/Documentos_por_capitulos), vol. 2, 2013.
- [12] M. Echenique. El color. [Online]. Available: <https://es.calameo.com/books/00204736273f7a25a93af>
- [13] J. A. Vega Uribe and M. A. Reyes Figueroa, "Transformaciones lineales y no lineales para espacios de color en procesamiento digital de imágenes," Revista internacional de métodos numéricos, 2006.
- [14] V. Popov, M. Ostarek, and C. Tenison, "Practices and pitfalls in inferring neural representations," NeuroImage, vol. 174, 03 2018.
- [15] A.d. I. Escalera Hueso, "Visión por computador: Fundamentos y métodos," 2001.



- [16] B.Jähne, "Digital image processing 5th revised and extended edition," Berlin: Springer-Verlag, doi, vol. 10, pp. 3–540, 2002.
- [17] S. L. E. Rafael C. González, Richard E. Woods, Digital Image Processing, 3rd edition. Pearson, 2004.
- [18] A.CARCEDO Y Franco, "Programa de segmentación de regiones en imágenes médicas en matlab," Ingeniería en Electrónica y Comunicaciones, UPLA-P, 2004.
- [19] E. Alegre, G. Pajares, and A. De la Escalera, "Conceptos y métodos en visión por computador," España: Grupo de Visión del Comité Español de Automática (CEA), 2016.
- [20] M. A. Nematollahi, S. Shahbazi, and N. Nabian, Computer Vision and Audition in Urban Analysis Using the Remorph Framework. Springer, 2019.
- [21] R. Szeliski, Computer vision: algorithms and applications. Springer Science & Business Media, 2010.
- [22] D. Mery. Domingo mery. [Online]. Available: <https://domingomery.ing.puc.cl/>
- [23] I.Garcia. Phyton. [Online]. Available: <https://www.python.org/>
- [24] A.Ronacher. Flask. [Online]. Available:<https://flask.palletsprojects.com/en/2.0.x/>
- [25] I.Corporation. Opencv. [Online]. Available: <https://opencv.org/>
- [26] S. van der Walt. Scikit-image. [Online]. Available: <https://scikit-image.org/>
- [27] T. Oliphant. Numpy. [Online]. Available: <https://numpy.org/>
- [28] S. Raheel, "Improving the user experience using an intelligent adaptive user interface in mobile applications," in 2016 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET). IEEE, 2016, pp. 64–68.
- [29] P. Wright, M. Blythe, and J. McCarthy, "User experience and the idea of design in hci," in International Workshop on Design, Specification, and Verification of Interactive Systems. Springer, 2005, pp. 1–14.
- [30] M. Osore. Guía esencial: La experiencia, elemento imprescindible para las empresas. [Online]. Available: <https://searchdatacenter.techtarget.com/es/guia/Guia-Esencial-La-experiencia-elemento-imprescindible-para-las-empresas>
- [31] F. D. G. Solfa, G. Amendolaggine, and T. A. A. Wall, "Nuevos paradigmas para el diseño de productos. design thinking, service design y experiencia de usuario," Arte e Investigación, no. 14, pp. e012–e012, 2018.
- [32] D. Knemeyer, "Design thinking and ux: Two sides of the same coin," Interactions, vol. 22, no. 5, pp. 66–68, 2015.
- [33] V. Balasubramoniam and N. Tungatkar, "Study of user experience (ux) and ux evaluation methods," International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 2, no. 3, pp. 1214–1219, 2013.
- [34] J. Nielsen, "Iterative user-interface design," Computer, vol. 26, no. 11, pp. 32–41, 1993.
- [35] A.Dillon, "User interface design," Encyclopedia of cognitive science, 2006.
- [36] S. Sridevi, "User interface design," International Journal of Computer Science and Information Technology Research, vol. 2, no. 2, pp. 415–426, 2014.
- [37] M. Hashemi and J. Herbert, "Uixsim: A user interface experience analysis framework," in 2014 5th International Conference on Intelligent Systems, Modelling and Simulation. IEEE, 2014, pp. 29–34.
- [38] C. A. Córdoba Cely, "La experiencia de usuario extendida (uxe): un modelo teórico sobre la aceptación tecnológica y un estudio de caso en entornos virtuales de aprendizaje." 2013.
- [39] M. Hassenzahl and N. Tractinsky, "User experience-a research agenda," Behaviour & information technology, vol. 25, no. 2, pp. 91–97, 2006.
- [40] I.Hussein, M. Mahmud, and A. O. M. Tap, "Hci knowledge for ux practices in the web development process," in International Conference of Design, User Experience, and Usability. Springer, 2014, pp. 116–126.

- [41] S. Adikari, C. McDonald, and J. Campbell, "Reframed contexts: design thinking for agile user experience design," in *International Conference of Design, User Experience, and Usability*. Springer, 2013, pp. 3–12.
- [42] J. Deng, Y. Huang, B. Chen, C. Tong, P. Liu, H. Wang, and Y. Hong, "A methodology to monitor urban expansion and green space change using a time series of multi-sensor spot and sentinel-2a images," *Remote Sensing*, vol. 11, no. 10, p. 1230, 2019.
- [43] H. Shahabi, H. Zabihian, and A. Shikhi, "Application of satellite images and gis in evaluation of green space destruction in urban area (case study: Boukan city)," *International Journal of Engineering*, vol. 1, no. 7, 2012.
- [44] C. Ludwig, R. Hecht, S. Lautenbach, M. Schorcht, and A. Zipf, "Mapping public urban green spaces based on openstreetmap and sentinel-2 imagery using belief functions," *ISPRS International Journal of Geo-Information*, vol. 10, no. 4, p. 251, 2021.
- [45] M. I. Riomoros Callejo, "Segmentación automática de texturas en imágenes agrícolas," 2016.
- [46] S. Sastoque, C. Narváez, and G. Garnica, "Metodología para la construcción de interfaces gráficas centradas en el usuario," *Nuevas Ideas en Informática Educativa*, vol. 12, pp. 314–324, 2016.
- [47] R. Hernández, C. B. P. FERNÁNDEZ, and P. Baptista, *Metodología de la Investigación*, México DF, México, Editorial Mc Graw Hill. Interamericana Ed. SA, 2003.
- [48] F. N. Kerlinger, H. B. Lee, L. E. Pineda, I. Mora Magaña et al., *Investigación del comportamiento*, 2002.
- [49] J. Bucanek, "Model-view-controller pattern," *Learn Objective-C for Java Developers*, pp. 353–402, 2009.
- [50] MapTiler. Azulejos a la google maps. [Online]. Available: <https://maptiler.es/google-maps-coordinates-tile-bounds-projection/>