Universidad Autónoma de Baja California
Instituto de Ingeniería

Metodología y algoritmo óptimo para el análisis de vitalidad foliar a través de la fusión inteligente de datos de Visión Estereoscópica y Barrido Láser

Tesis

que para obtener el título de

Doctor en Ciencias

presenta

Rubén Alaniz Plata

Director de tesis: Dr. Oleg Sergiyenko
Codirector de tesis: Dra. Wendy Flores Fuentes

Mexicali, B.C. Agosto de 2024

Resumen

La fusión de información es una técnica que integra múltiples capas de datos, permitiendo la toma de decisiones automatizadas e informadas. En este proyecto, se combina la tecnología de dos sistemas de visión artificial: un escáner láser y un sistema de visión estereoscópica. Cada uno de estos sistemas tiene la capacidad de capturar diversas variables del entorno, proporcionando una visión más completa y precisa para la automatización de decisiones. En el caso de la visión estereoscópica, es posible realizar la clasificación y segmentación de objetos, así como la estimación de sus coordenadas tridimensionales con un error absoluto medio de $1.563\,cm$. El escáner láser también permite obtener coordenadas tridimensionales con un error absoluto medio de $0.640\,cm$ y determinar propiedades cualitativas de las superficies escaneadas, como el color. Teniendo en cuenta esto, se presenta una metodología y algoritmos para la combinación y compensación de los datos tridimensionales adquiridos por ambos sistemas. Además, se introduce una metodología para la identificación y localización de hojas de plantas, así como la clasificación del verdor de las hojas, donde se alcanza una precisión de $91.7\,\%$. Estas capas de información obtenidas son útiles para la localización espacial de las plantas y determinar su estado nutricional.

Abstract

Information fusion is a technique to integrate multiple data layers, allowing for informed and automated decision-making. In this project, two artificial vision systems are combined: a laser scanner and a stereo vision system. Each system has the capability to capture multiple types of data, providing a wider and more precise vision for automated decision-making. With stereo vision, it is possible to classify and segment objects, as well as estimate their coordinates with a mean absolute error of $1.563\,cm$. The laser scanner also allows for the acquisition of three-dimensional data with a mean absolute error of $0.640\,cm$ and the determination of qualitative surface properties, such as color. Under these considerations, a methodology and algorithms for the combination and compensation of three-dimensional data acquired from both systems are presented. Additionally, a methodology for the identification and localization of plant leaves, as well as the classification of their greenness with a precision of $91.7\,\%$, is introduced. These information layers are useful for the spatial location of plants and the determination of their nutritional status.

Dedicatoria

Para Ale, mi persona favorita.

Agradecimientos

Mi más grande agradecimiento es para mi director de tesis, el Dr. Oleg Sergiyenko, por su sincera amistad y confianza. Es admirable su pasión por desarrollar personas independientes, creativas y comprometidas con la investigación.

A mi co-directora de tesis, Dra. Wendy Flores Fuentes, que sin ella no hubiera tenido la fortuna de iniciar mi proyecto de doctorado con personas tan profesionales. Siempre agradeceré su valiosa disposición a brindar conocimiento y ayuda.

Al Dr. Julio César Rodríguez Quiñonez, por estar constantemente impulsando nuestro desarrollo y buscando áreas de mejora. Le estaré siempre agradecido por su invaluable atención a mi desarrollo profesional.

A mis compañeros de posgrado, César, José y Humberto, gracias por su amistad y ayuda para que mi proyecto saliera adelante.

Agradezco al Instituto de Ingeniería, a sus investigadores y personal administrativo por hacer de esta universidad una segunda casa para todas las personas que aspiramos a aportar nuestro granito de arena al mundo de la ciencia y la tecnología.

Al CONAHCYT y al pueblo mexicano le agradezco el apoyo brindado a través del apoyo económico, indispensable para alcanzar mis estudios de doctorado.

Índice general

Índice de figuras

Índice de tablas

Introducción

La evaluación precisa del estado de salud de las plantas es fundamental en la agricultura de precisión, permitiendo una cuantificación detallada de su condición actual. Este análisis crítico facilita la toma de decisiones informadas sobre las intervenciones necesarias para mantener o mejorar la salud de las plantas de manera automatizada. Por ejemplo, si una planta experimenta estrés por falta de agua, la determinación oportuna de este estado permite implementar medidas de riego específicas. Del mismo modo, si se identifica una deficiencia de nutrientes, se puede proceder a una fertilización adecuada.

Para realizar esta cuantificación se han utilizado diversos índices medibles por dispositivos remotos automatizados que indican desde el verdor en las plantas, como el Índice de Vegetación de Diferencia Normalizada (NDVI) hasta los que estiman el nivel de estrés hídrico como el Índice Diferencial de Agua Normalizado (NDWI). Es a través de estos valores que se suelen tomar medidas de riego y fertilización.

La medición de índices como el NDVI y el NDWI se suele realizar de manera manual, utilizando dispositivos ópticos como el GreenSeeker [1] o de forma semi-automatizada, mediante la captura de imágenes hiperespectrales desde satélites o drones sobrevolando los cultivos. En ambos casos, la interpretación y validación de los datos dependen del análisis humano, quien es responsable de procesar estas lecturas.

En este proyecto de tesis se propone un sistema multi-sensorial capaz de detectar las hojas de las plantas, localizarlas espacialmente y analizarlas para determinar su nivel de verdor haciendo uso de novedosos algoritmos de procesamiento de datos. Esto con el fin de cuantificar su estado de salud foliar e indicar si es necesario tomar una medida como el riego o la fertilización.

De manera general, el sistema debe cumplir con tres componentes críticos para su funcionamiento. Primero, la clasificación de hojas mediante técnicas avanzadas de procesamiento de imágenes con el uso de Redes Neuronales. Segundo, la determinación de la ubicación espacial de las hojas, utilizando triangulación. Tercero, la clasificación de tonalidad foliar a través del procesamiento de señales.

Los tres puntos clave antes mencionados serán cubiertos haciendo uso de la información obtenida por un sistema de visión estereoscópica y un sistema de barrido láser. Ambos sistemas deberán funcionar de manera colaborativa y complementaria, por lo que también se propone una nueva metodología de fusión de información.

En el caso del sistema de barrido láser, se propone el uso de un Laser Scanner (LS), el cual es un sistema que proyecta un haz de luz hacia un objeto reflectante y captura, con una precisión de $\pm 0.640, cm$, las coordenadas espaciales de un punto. La luz reflejada es detectada por un foto-detector, cuya señal se envía a un sistema de adquisición de datos para calcular el ángulo de incidencia. Este ángulo permite determinar la posición del punto de incidencia del láser, proporcionando sus coordenadas rectangulares en el espacio.

Por otro lado, la visión estereoscópica es un sistema que facilita la estimación de la posición y distancia de los elementos dentro de una escena. Se utiliza un par de cámaras, cuyas especificaciones y posiciones espaciales se definen mediante un proceso de calibración. Gracias a estos parámetros, tras identificar y clasificar los objetos en las imágenes de ambas cámaras, se realiza una triangulación para cada objeto o píxel en la imagen, permitiendo determinar sus coordenadas espaciales. La precisión en la identificación y clasificación de los objetos incide directamente en la exactitud de las mediciones. En el sistema empleado en este proyecto, se logró una precisión de $\pm 1.563\, cm$.

A lo largo de esta tesis, se realizará una revisión de los principios de funcionamiento de los sistemas de visión estereoscópica y de barrido láser, incluyendo sus especificaciones técnicas y algoritmos de procesamiento de datos. Además, se presentarán pruebas experimentales que demuestran sus alcances y limitaciones para la fusión de información y cómo el procesamiento de imágenes y señales obtenidas con estos sistemas facilita el análisis de la salud foliar de las plantas.

1.1. Planteamiento del problema

Obtener indicadores cuantitativos que definan el estado de salud de las plantas de un cultivo mediante dispositivos optoelectrónicos es una tarea compleja, debido a la diversidad de factores que pueden influir en este estado. Estos factores están estrechamente relacionados con las condiciones climáticas, el riego y los nutrientes del suelo.

Actualmente, para determinar los índices que reflejan el estado de salud de las plantas, se recurre al procesamiento de imágenes hiper-espectrales obtenidas mediante satélites y drones. Estas imágenes permiten obtener un gran volumen de datos en cada captura de manera muy rápida. No obstante, variables naturales como la hora del día, la nubosidad y la estación del año pueden alterar la forma en que la luz se refleja en las plantas, afectando así la calidad de las imágenes capturadas.

Como alternativa para adquirir datos de forma remota, se hace uso de cámaras, que estiman el verdor de las hojas con un valor que va de 0 a 1. No obstante, la obtención e interpretación de estos datos dependen de la intervención humana, lo cual convierte su uso en el campo en una actividad exigente.

Se requiere un dispositivo que posibilite la recolección automatizada y fiable de información acerca del estado de salud de las plantas, minimizando considerablemente la dependencia de la intervención humana en el proceso. Este avance tecnológico deberá estar diseñado para operar de manera eficiente en diversos entornos agrícolas, asegurando precisión y reduciendo el margen de error en la evaluación de la vitalidad vegetal.

1.2. Justificación

Para realizar la detección básica de hojas, es esencial disponer de cámaras capaces de capturar y procesar imágenes para diferenciar las hojas del entorno. Estas cámaras también facilitan una estimación preliminar de la posición espacial de las hojas. Dicha estimación inicial permite orientar el sistema de barrido láser hacia las áreas de interés, empleando un algoritmo de fusión de datos que integra los parámetros geométricos y

operativos de ambos sistemas. El sistema de barrido láser, además, ayuda a complementar o corregir las estimaciones de datos tridimensionales y utiliza la misma señal para evaluar el nivel de verdor de las hojas, un indicador clave de la salud foliar. Este nivel de verdor se puede cotejar con mediciones de clorofila como variable de validación.

Gracias a estas capacidades, el uso de un sistema multi-sensorial puede determinar de forma autónoma la posición de las plantas y definir puntos de análisis para la recolección de datos, eliminando la necesidad de intervención humana tanto en la adquisición como en el procesamiento de la información. Tomando en consideración que los datos y funcionamiento de los sistemas a combinar deben ser compatibles y complementarios.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar formalismo teórico y algoritmos para la optimización de funcionamiento colaborativo de los sistemas técnicos de estereovisión y de barrido óptico con el propósito de sincronizar ROI y FOV para el aumento de la calidad y cantidad de información obtenida por el sistema de visión técnico, bajo la condición de minimización de tiempo y decrecimiento acelerado de entropía.

1.3.2. Objetivos específicos

- Definir la configuración óptima de la integración de dos sistemas para el aumento de la calidad de la información adquirida.
- Optimizar parámetros de resolución de las cámaras con el objetivo de facilitar interacción de Visión Estereoscópica con LS.
- Establecer el procedimiento de calibración del sistema de Visión Estereoscópica y LS para la mejora en la fiabilidad de las mediciones.
- Desarrollar metodología y formalismos matemáticos para relacionar las coordenadas rectangulares del sistema de Visión Estereoscopica con el posicionamiento

del láser del LS.

- Diseño y desarrollo de panel de control con interfaz de usuario para la adquisición, visualización y procesamiento de datos común para ambos sistemas.
- Desarrollar e integrar algoritmo para lograr la interfaz de comunicación entre el sistema de Visión Estereoscópica y el LS.
- Seleccionar parámetros de la señal del LS que permiten relacionar el nivel de verdor con la salud foliar de las plantas.
- Evaluar, a través de herramientas metrológicas confiables, la integración del sistema de Visión Estereoscópica y del LS.

1.4. Hipótesis

La implementación de un sistema multi-sensorial, integrando tecnologías de visión estereoscópica y barrido láser, facilitará la automatización del proceso de detección de hojas, identificación de las áreas a analizar y adquisición de datos fiables sobre la vitalidad foliar. La combinación de datos de estos dos sistemas proporciona la información necesaria para completar las tareas de escaneo foliar de manera eficiente, eliminando la necesidad de utilizar sensores o dispositivos adicionales.

1.5. Antecedentes

1.5.1. Vitalidad foliar

El distintivo color verde de las hojas de las plantas se debe a la clorofila. Este pigmento no solo es responsable de su apariencia, sino que también actúa como un indicador esencial de la salud vegetal. Las variaciones en el nivel de clorofila pueden indicar cambios importantes en el estado de salud de las plantas, reflejando el impacto de condiciones ambientales adversas como sequías, heladas, altas temperaturas y ataques de plagas. Así, monitorear la cantidad de clorofila proporciona una perspectiva valiosa

sobre el estado fisiológico de las plantas, facilitando la detección y el manejo de posibles estrés o desequilibrios ambientales [2]. La relevancia de este descubrimiento ha permitido relacionar el contenido de clorofila en las hojas con las necesidades de nitrógeno de la planta, como sugiere [3], y continúa siendo pertinente actualmente con el uso de otros indicadores como Índice de Color Verde Oscuro (DGCI), auxiliar en la determinación de la salud y el vigor de las plantas.

El DGCI presenta como principal ventaja la fácil implementación en dispositivos móviles con cámaras, permitiendo un análisis en tiempo real del verdor de las hojas [4]. Sin embargo, la precisión de estos sistemas depende de varios factores, incluyendo la resolución y nitidez de la imagen, las condiciones lumínicas, la presencia de oclusiones ambientales (como sombras) y fenómenos ópticos en los objetos dentro del campo de visión (como reflexiones) [5].

Además del DGCI, para evaluar la salud de las plantas existen múltiples aproximaciones e índices, principalmente asociados al escaneo remoto, procesamiento de imágenes y la implementación de Redes Neuronales. Estas incluyen el NDVI (Índice de Vegetación de Diferencia Normalizada) que se hace un análisis de forma global sobre el verdor de una o múltiples plantas, SAVI (Índice de Área de Vegetación Ajustado al Suelo), que ajusta el NDVI por la influencia del suelo, especialmente útil en áreas con baja cobertura vegetal [6]; el EVI (Índice de Vegetación Mejorado), que mejora la sensibilidad en regiones de alta biomasa y minimiza la influencia de la atmósfera [7]; y el NDWI (Índice de Agua de Diferencia Normalizada), enfocado en el contenido de humedad en la vegetación [8].

Una de las técnicas más empleadas para determinar distintos índices de salud vegetal es el procesamiento de imágenes espectrales e hiper-espectrales, que proporcionan información detallada sobre la reflexión de la luz en la vegetación a través de múltiples y bien definidas longitudes de onda [9]. La reflectividad de la luz en cada longitud de onda puede revelar parámetros y condiciones específicas que indican la salud de la vegetación, como el NDVI, el CWSI y la detección de plagas y enfermedades específicas [10, 11]. Estas técnicas se destacan por la gran cantidad de datos que pueden guardar en una sola captura. Sin embargo, la complejidad para procesar estos datos puede dificultar

la toma de decisiones automatizada, sumado a las significativas variaciones que pueden presentarse en las imágenes debido a que las condiciones atmosféricas afectan la reflectividad de la luz en las plantas [12].

Cuando se trata de adquisición de datos fiables y precisos, se recurre a sistemas opto-electrónicos como el LiDAR, útil en la clasificación de tipos de vegetación [13] y en la estimación de la humedad foliar [14]. Otro instrumento en esta categoría es el GreenSeeker, un dispositivo diseñado para evaluar el vigor de las plantas midiendo la reflectividad de la luz roja (570 a 680 nm) y de la luz infrarroja (725 a 1020 nm), generando un índice que varía de -1 a 1 [15]. Aunque las lecturas de estos sistemas pueden ser confiables, su operación requiere verificación y manejo manual en el campo. Además, dado que estos dispositivos se centran exclusivamente en capturar información de la vegetación, integrar sus datos con los de otros sensores para su uso en sistemas con localización automática presenta desafíos.

1.5.2. Identificación de hojas

Automatizar el análisis de salud foliar requiere que el sistema sea capaz de identificar a las hojas de las plantas como objeto de interés, localizarlas en el espacio y escanearlas de manera remota, asegurando la obtención de datos confiables. De esta forma, identificación precisa de las hojas en la vegetación es primordial para la evaluación foliar autónoma. El sistema debe ser capaz de distinguir las hojas de otros elementos en la escena. Por lo tanto, es fundamental emplear un algoritmo de segmentación de imágenes especializado en diferenciar la vegetación de elementos como el suelo, frutos o cualquier otro componente no foliar.

Actualmente, existen diversos métodos para segmentar hojas, donde la mayoría opta por técnicas de procesamiento de imágenes avanzadas, dejando de lado los métodos de procesamiento clásicos en favor de las Redes Neuronales. Un ejemplo notable es el trabajo de [16], que, mediante la implementación de LeafMask y utilizando el Leaf Segmentation Challenge Dataset, logra la mayor precisión reportada de 90.09 %. Esto es seguido por BlendMask [17], que alcanza un 87.9 % de precisión, y en tercer lugar, uno de los pioneros en segmentación con máscaras, Mask R-CNN [18], con un 86.9 %.

Para este proyecto se hará uso de un modelo de segmentación de código abierto, otorgado con licencia del MIT para uso libre [19]. Esta Red Neuronal está basada en el modelo U-NET, ampliamente utilizado para aplicaciones de Visión Artificial [20]. Se seleccionó este modelo debido a su fácil implementación, su libertad de uso y su velocidad de procesamiento de imágenes de 0.02235 segundos con imágenes de 579x588 píxeles con un procesador de 3.4 GHz.

1.5.3. Posicionamiento con imágenes

Al terminar correctamente el proceso de segmentación, el siguiente paso es determinar la ubicación espacial de las hojas identificadas. Para ello, es necesario emplear un sistema capaz de manejar estas imágenes de manera secuencial o en paralelo para calcular las coordenadas tridimensionales de los objetos en la escena. Aquí es donde se aplican diversos métodos de estimación de posición mediante el procesamiento de imágenes, incluyendo técnicas como la Visión Monocular. Esta última se emplea en aplicaciones donde una precisión baja en la estimación es aceptable, tales como la Realidad Aumentada y la Realidad Virtual [21].

Para mejorar la precisión, se recurre a la Visión Estereoscópica, que, a diferencia de la Visión Monocular que usa una sola cámara, utiliza dos cámaras con una posición relativa conocida entre ellas. Esto proporciona una referencia fija y confiable que facilita mediciones más precisas y exactas. En general, con esta técnica se busca identificar todos los objetos de la escena utilizando las dos cámaras y triangular sus posiciones espaciales utilizando la referencia fija. La exactitud depende de cuán bien se haya medido la referencia y de cuánto se conozcan las deformaciones en los lentes de las cámaras [22]. Asimismo, la confiabilidad de las mediciones está condicionada por la eficacia del algoritmo de procesamiento de imágenes en identificar el mismo objeto en las imágenes capturadas por ambas cámaras [23].

Cuando es necesario incrementar la cantidad de información espacial, se recurre a configuraciones avanzadas que involucran más de dos cámaras, englobadas en la técnica conocida como Visión Multi-Cámara. Estas configuraciones permiten obtener tres o más ángulos de un objeto o escena en una sola captura. No obstante, es evidente que esto

demanda una capacidad de procesamiento significativa y el uso de algoritmos avanzados para asegurar que la información obtenida por las múltiples cámaras se integre de forma óptima [24].

El sistema propuesto en este proyecto es el de Visión Estereoscópica, elegido por su capacidad de equilibrar la obtención de información confiable, exacta y precisa sin necesidad de hardware especializado ni algoritmos complejos. Adicionalmente, disponer de parámetros de referencia medibles permite realizar la segmentación y la estimación de la profundidad de las hojas en el espacio de manera paralela. Finalmente, los datos de profundidad son fundamentales para posicionar un sistema de barrido con láser, también encargado de obtener datos precisos de posicionamiento y de evaluar la salud de la vegetación.

1.5.4. Sistemas de barrido con láser

Los cambios ambientales pueden generar problemas ópticos, tales como oclusiones en la captura de imágenes o errores que dificultan el correcto procesamiento de las mismas [25]. Estos fenómenos son comunes al operar dispositivos en condiciones reales. Para enfrentar estos retos, los desarrolladores deben prever estas situaciones con el fin de corregir o excluir los datos afectados, asegurando así la fiabilidad del sistema.

Corregir todos los posibles errores ocasionados por fenómenos ópticos puede ser una tarea ardua que requiere implementar múltiples capas de procesamiento de imágenes [26]. Por esta razón, se recurre a sistemas de barrido con láser, que son eficaces en situaciones difíciles para las cámaras. Los métodos principales para la estimación de profundidad con láser incluyen el Tiempo de Vuelo (ToF) y la triangulación. El dispositivo más representativo del método ToF es el LiDAR [27], el cual consta de un emisor que envía un pulso de luz y un receptor que capta la señal de retorno de dicho pulso. Midiendo el tiempo entre la emisión y la recepción del pulso, y conociendo la velocidad de la luz, es posible estimar la distancia a la superficie reflejante. Dado que el LiDAR, en su funcionamiento básico, estima únicamente una distancia lineal, a menudo se combina con cámaras o sensores inerciales para determinar posiciones tridimensionales con coordenadas (x, y, z). Sin embargo, esto introduce una incertidumbre en la estimación

de la posición, ya que es importante una sincronización precisa entre el LiDAR y los sensores complementarios para realizar estimaciones acertadas.

En lo que respecta a la triangulación, esta también requiere el uso de un emisor y un receptor, así como la medición de los ángulos de emisión y recepción. Utilizando lo dos ángulos de emisión y recepción y la distancia entre ellos, se puede aplicar la ley de senos para determinar la posición espacial de la superficie sobre la que incide el haz. En esta técnica, se encuentran dispositivos que integran láseres con cámaras [28] para facilitar la triangulación, así como sistemas independientes, como el LS [29]. Este último ofrece la ventaja de obtener coordenadas tridimensionales con mayor precisión, gracias a su menor dependencia de dispositivos externos. A modo de referencia, la incertidumbre actual en la estimación de la posición de un LiDAR es de $\sim 6cm$ y en el caso del LS es de $\sim 0.5cm$.

Por otra parte, aunque existen metodologías que posibilitan determinar la salud de la vegetación mediante el procesamiento de imágenes [30, 31], la incertidumbre asociada a los resultados obtenidos suele ser considerable, debido principalmente a su fuerte dependencia de las condiciones climáticas y de iluminación del entorno analizado. Específicamente, la forma en que la luz incide y se refleja en la vegetación puede alterar la percepción del color en las imágenes y causar oclusión por reflejos [32]. Por esta razón, la evaluación de la salud vegetal frecuentemente se realiza utilizando múltiples sensores que proporcionan datos redundantes o complementarios, lo cual permite llegar a conclusiones más precisas y fundamentadas sobre el estado de la planta.

Los sistemas de medición con láser son poco empleados en el análisis de vitalidad vegetal [33], normalmente están más asociados con la estimación de densidad de vegetación [34]. Sin embargo, el LS se distingue por su capacidad para obtener las coordenadas del punto de incidencia del láser y, simultáneamente, generar una señal única para cada superficie en la que es incidido el láser. Esta señal puede utilizarse para clasificar el nivel de verdor en las hojas de las plantas y, por ende, estimar la cantidad de clorofila relacionada con el contenido de nitrógeno, principal indicador de la necesidad de fertilización. Así, el sistema realiza dos funciones con la misma señal: la estimación de posición y el análisis de la salud foliar.

En el Instituto de Ingeniería de la Universidad Autónoma de Baja California se han desarrollado tres prototipos y registrado dos patentes relacionadas con el LS [35, 36]. Este dispositivo es capaz de obtener coordenadas tridimensionales de un objeto y, al mismo tiempo, determinar el color de una superficie mediante el procesamiento de señales [37]. Por tanto, la integración del LS con un sistema de Visión Estereoscópica facilitará la identificación y análisis de hojas de plantas de manera completamente automatizada.

1.5.5. Fusión de información

Cuando se trata de automatizar procesos, se busca que un solo sensor o dispositivo pueda proporcionar toda la información necesaria para tomar decisiones. Este dispositivo debe tomar lecturas con un alto nivel de confianza para asegurar que las decisiones tomadas sean correctas. Sin embargo, para que un solo dispositivo se encargue de la toma de decisiones, es crucial conocer todas las condiciones ambientales que puedan afectar la captura de información, y su controlador debe estar preparado para filtrar cualquier información irrelevante. Así, la forma principal de asegurar el funcionamiento correcto de un sistema basado en la lectura de un solo dispositivo es en ambientes controlados.

Al involucrar variables incontrolables en el proceso, la confianza en las lecturas de este sensor disminuye drásticamente, lo que lleva a la necesidad de desarrollar algoritmos más complejos y avanzados para procesar esta información [38]. Aquí es donde la solución viable es agregar más sensores con principios de funcionamiento diferentes, capaces de complementar la información del primero [39–41]. Esta técnica, conocida como fusión de información, implica no solo la combinación de sensores, sino también la creación de metodologías y algoritmos que permitan integrar las lecturas de manera eficiente, evitando que los datos añadan incertidumbre a la toma de decisiones.

En el ámbito del posicionamiento, existen múltiples combinaciones de sistemas y técnicas que se han integrado para optimizar la ubicación espacial de robots destinados a tareas de navegación autónoma, monitoreo, digitalización y fotogrametría. Desde esta perspectiva, la propuesta de Dong y Xu consiste en fusionar un RADAR con cámaras

para potenciar la navegación autónoma de vehículos, empleando un algoritmo que logra una confiabilidad del 92.2 % [42]. No obstante, la densidad de datos proporcionada por el LiDAR en cortos periodos de tiempo resulta más atractiva para su combinación con cámaras, según indican [43, 44], a pesar de la considerable capacidad de procesamiento que esto requiere.

En este trabajo de tesis se propone un modelo capaz de fusionar la información tridimensional obtenida tanto por los sistemas de Visión Estereoscópica como por el LS, mediante el análisis de precisión en la estimación de profundidades y la ubicación espacial de los componentes del sistema combinado. Esto resulta en la asignación de un peso específico a cada sistema, que facilita la sincronización de las nubes de puntos medidos por ambos, con el objetivo de generar una nube de puntos unificada. Dicha nube presenta mediciones cuya incertidumbre se ve reducida, aprovechando los puntos fuertes de ambos sistemas.

Marco teórico

2.1. Sistema de barrido con láser

2.1.1. Laser Scanner

El LS es un sistema de visión artificial diseñado para determinar la posición espacial de superficies accesibles mediante un haz de láser. Teóricamente, cualquier superficie reflectante dentro del campo de visión del LS puede ser escaneada para crear una nube de puntos, representando digitalmente al objeto en tres dimensiones.

La principal ventaja de los sistemas de visión láser, como el LS, sobre aquellos basados en cámaras radica en la certeza de que cada punto detectado corresponde a un objeto físico que reflejó la luz, ofreciendo un método determinista para la estimación de coordenadas espaciales.

2.1.2. Triangulación Dinámica

El LS es un sistema de barrido con láser basado en el principio de triangulación dinámica, el cuál consiste en la formación de un triángulo cuyos vértices son el emisor, la superficie donde se refleja la luz emitida y el receptor (Fig. 2.1). La triangulación dinámica presenta la ventaja de ser un principio de alta precisión en la estimación de la posición pero con fuerte dependencia en el conocimiento de al menos un lado (a) y dos ángulos internos del triángulo (α y β).

Al formarse el triángulo y conociendo a α, β y a se puede determinar las coordenadas

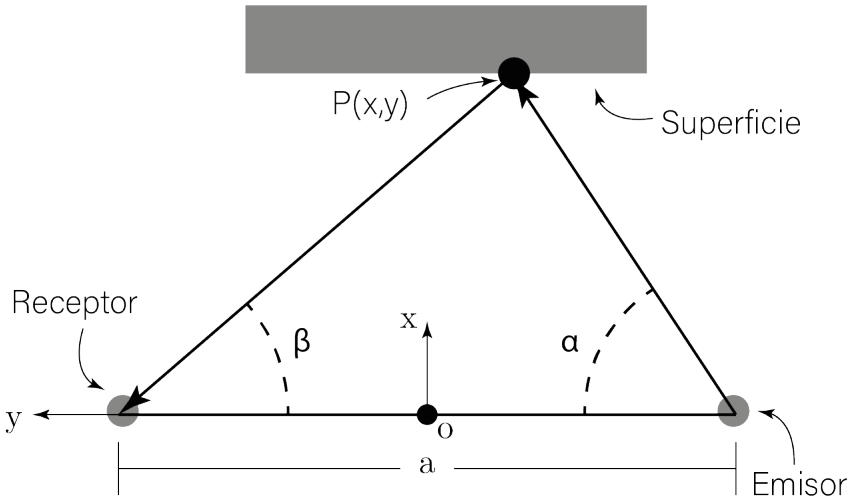


Figura 2.1: La triangulación y sus parámetros desde el plano xy.

en el espacio del punto de incidencia $P(x, y)$ con referencia al origen o, como se muestra en las ecuaciones 2.1 y 2.2.

$$x = a\left(\frac{sin(\alpha)sin(\beta)}{sin(\alpha+\beta)}\right) \tag{2.1}$$

$$y = a\left(\frac{1}{2} - \frac{cos(\alpha)sin(\beta)}{sin(\alpha+\beta)}\right) \tag{2.2}$$

Buscando agregar la coordenada z que otorga la tridimensionalidad a la localización espacial, es necesario ver al sistema y a la superficie de perfil para visualizar un segundo triángulo como el que se muestra en la Figura 2.2.

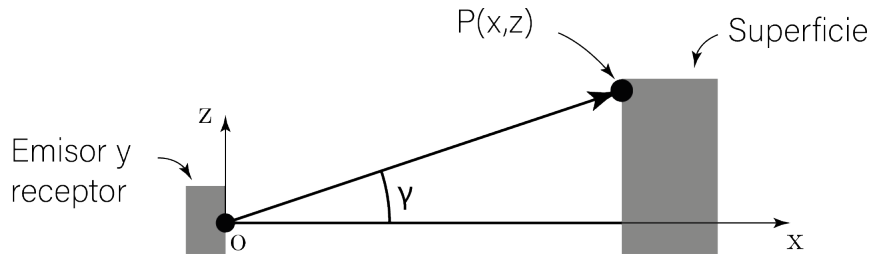


Figura 2.2: La triangulación y sus parámetros desde el plano xz.

Donde γ es el ángulo de posicionamiento vertical del sistema y permite obtener la coordenada z del sistema con la ecuación 2.3.

$$z = a\left(\frac{sin(\alpha)sin(\beta)tan(\gamma)}{sin(\alpha+\beta)}\right) \tag{2.3}$$

En la práctica, este sistema es como el de la figura 2.3, donde se pueden visualizar todos los componentes y parámetros antes mencionados desde una perspectiva tridimensional.

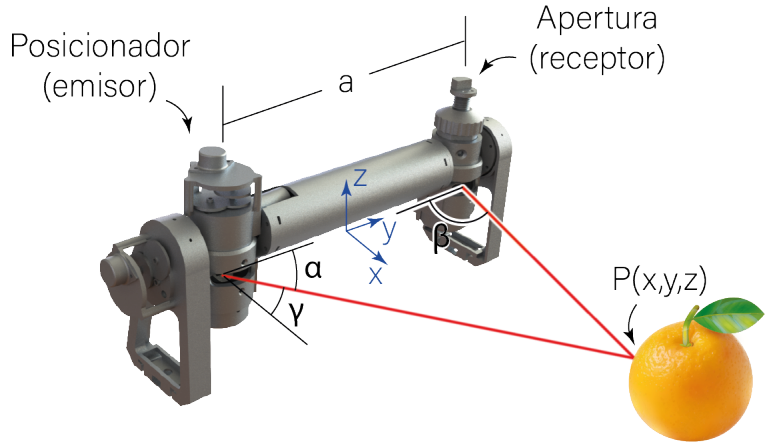


Figura 2.3: Representación del LS y los parámetros para la triangulación.

En el diseño del LS destacan el posicionador (emisor) y la apertura (receptor) como los elementos encargados de llevar a la práctica la triangulación dinámica y evaluar la posición espacial de cualquier superficie reflectante.

2.1.3. Posicionador láser

El posicionador láser como emisor del LS, está compuesto por láser que se refleja en dos espejos a 45 grados, estando el primer espejo fijo y, el segundo, sujeto a un motor a pasos que permite cambiar la dirección de la luz de manera horizontal (Figura 2.4).

El motor a pasos, mostrado en la Figura 2.4, rota el espejo a 45 grados para variar la posición del láser en cualquier proyección del plano perpendicular al espejo. Este proceso se considera un posicionamiento horizontal discreto, dependiente de la resolución del motor.

El motor puede rotar 360 grados en una cantidad fija de pasos discretos. Por ejemplo, los motores del prototipo giran completamente con 20 pasos, lo que significa que cada paso ajusta el movimiento horizontal en 18 grados. Así, posicionar el haz en un punto

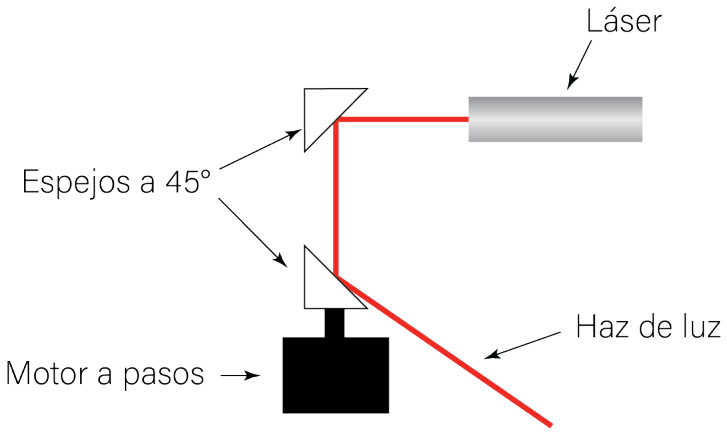


Figura 2.4: Diagrama de los componentes internos del posicionador láser

intermedio no es posible, generando áreas inaccesibles para adquirir datos. La cantidad de pasos por vuelta define la resolución de barrido alcanzable por el sistema.

Para aumentar la resolución de barrido, se requiere implementar sistemas de reducción que disminuyan el ángulo por paso. No obstante, aumentar la resolución conlleva una reducción de la velocidad angular, ralentizando el desplazamiento del motor a lo largo del campo de visión. Es crucial considerar, según la aplicación específica, si es más importante la alta resolución de barrido o la velocidad de posicionamiento.

Sin embargo, siguiendo la relación de transmisión como el principio fundamental de las transmisiones mecánicas, es notable que, al aumentar la resolución de barrido, se pierde velocidad angular, por lo que el movimiento de los motores a pasos a lo largo del campo de visión se vuelve lento. Bajo esta consideración, es necesario evaluar si para la aplicación específica es prioritario o no tener una alta velocidad de posicionamiento.

El control de los motores se realiza mediante una interfaz conectada a un circuito que envía una secuencia de pulsos separados por 11 ms. La interfaz, desarrollada en Python, ordena a un microcontrolador Teensy 4.1 para que envíe la secuencia de pulsos correcta a los motores, dirigiéndolos a una posición precalculada. Esta posición se registra como el ángulo α necesario para realizar la triangulación dinámica.

2.1.4. Apertura

Cuando el láser se refleja en una superficie, se espera que genere una reflexión especular, parte de la cual se dirige hacia el receptor del LS. Si la luz reflejada es suficientemente intensa como para ser detectada por un foto-sensor, entonces la apertura del LS actúa como el dispositivo encargado de estimar el ángulo de incidencia de esta luz reflejada.

Para realizar la estimación del ángulo de incidencia, la apertura dispone de seis elementos esenciales: un opto-acoplador, un foto-sensor, un espejo cortado a 45 grados, un motor de corriente continua (DC), un disco con una muesca y un lente biconvexo. El espejo y el disco se montan sobre el eje del motor DC, rotando a una velocidad angular constante. El emisor y receptor del opto-acoplador se sitúan alrededor del disco con la muesca, enviando un pulso cada vez que se completa una rotación. El lente biconvexo redirige la luz reflejada desde el espejo hacia el foto-sensor de forma paralela. El foto-sensor emite constantemente una señal basada en la intensidad de la luz reflejada. Estos componentes están gráficamente representados en la Figura 2.5.

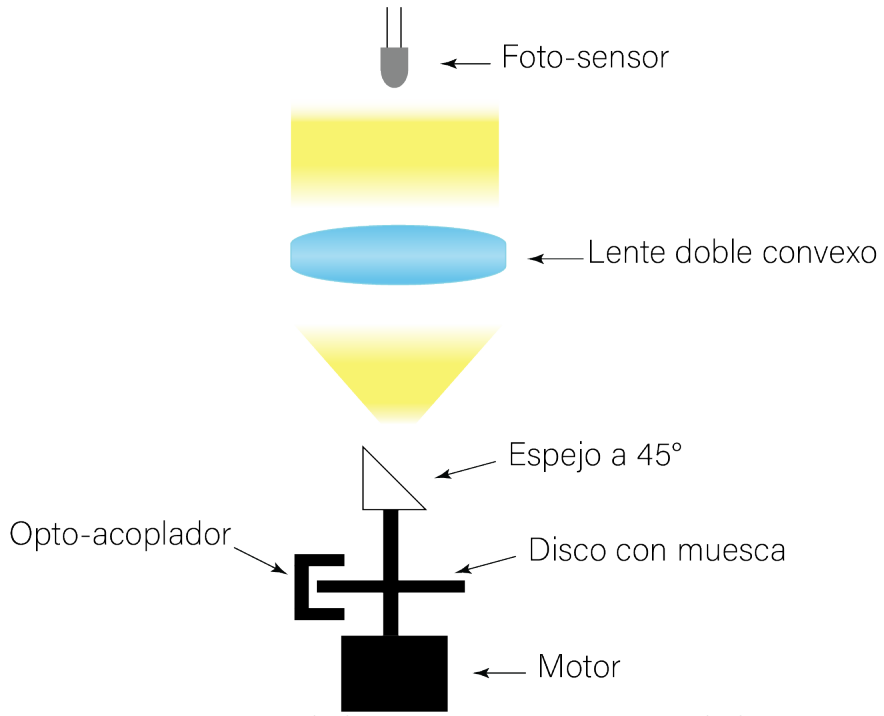


Figura 2.5: Diagrama de los componentes internos de la apertura

Para determinar el ángulo incidencia, se considera que la posición el opto-acoplador

es el ángulo cero o punto de partir. A partir de ahí se comienzan a comparar los valores de voltaje recibidos en el foto-sensor, representado con la señal de color azul en la Figura 2.6.

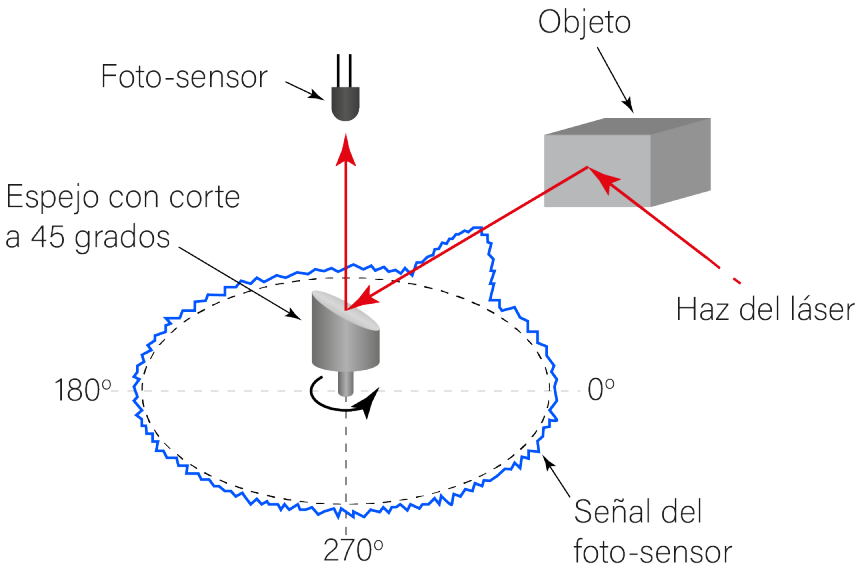


Figura 2.6: Diagrama de los componentes internos de la apertura

El valor más alto de voltaje en la señal del foto-sensor, que corresponde al ángulo de incidencia del láser, es identificado por el microprocesador para determinar el valor de β necesario para la triangulación dinámica.

2.1.5. Procesamiento de señales

Los motores utilizados para el posicionamiento láser y los sensores de la apertura están conectados a un micro-controlador Teensy 4.1 que se encarga de enviar los pulsos de movimiento a los motores del posicionador y de adquirir y procesar las señales del opto-acoplador y el foto-sensor de la apertura. El procesamiento de señales se hace de manera continua y es interrumpido cuando se manda la orden de mover cualquierda de los dos motores, de tal forma que la adquisición y procesamiento no pueden hacerse a la par del control de los motores. Esquemáticamente, esto se podría representar como en la Figura 2.7.

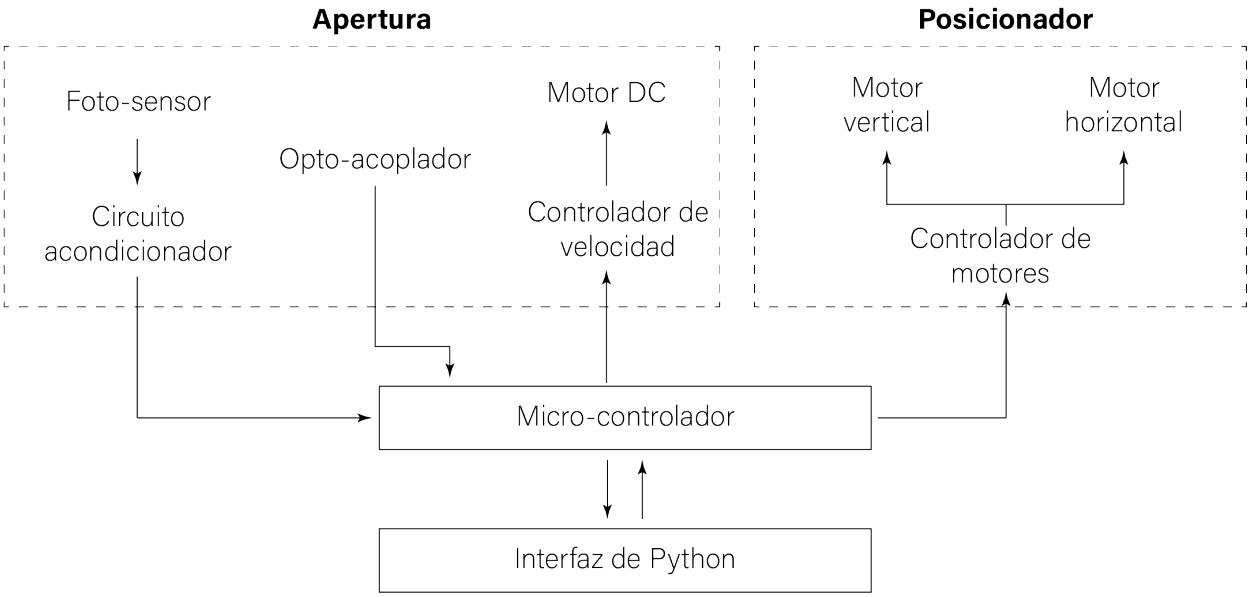


Figura 2.7: Esquema de sensores y actuadores del LS y la dirección de la transferencia de información.

En la Figura 2.7 se visualiza como el foto-sensor y el opto-acoplador están encargados de enviar señales continuas al microcontrolador. Estas señales se pueden visualizar como en la Figura 2.8, donde se aprecia como llegan de manera continua los pulsos del opto-acoplador (línea azul punteada) y el voltaje debido a la intensidad de luz recibida en el foto-sensor (línea roja).

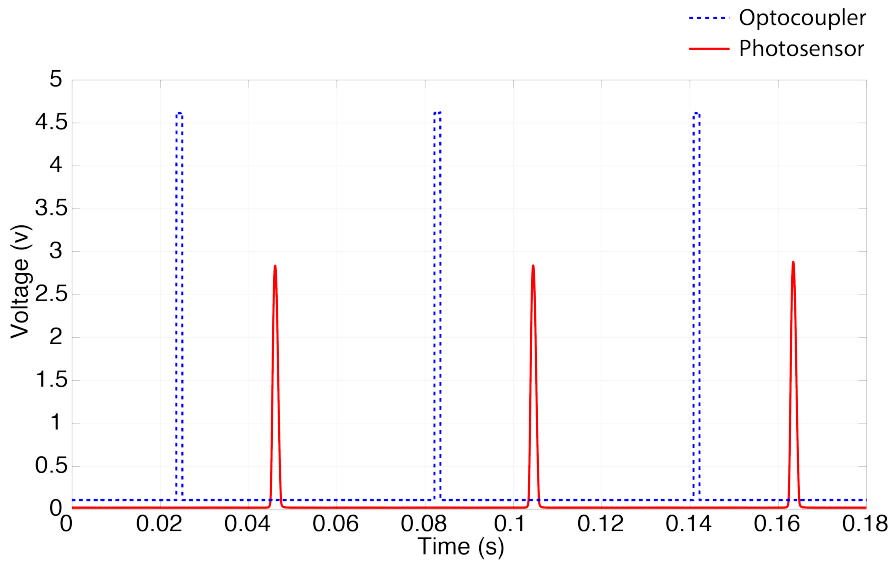


Figura 2.8: Par de señales obtenidas para 2 revoluciones del espejo de 45 grados de la apertura.

Esta señal es procesada entre pares de pulsos del opto-acoplador, representando la reflexión obtenida en cada giro del espejo de 45 grados de la apertura (Figura 2.9). En esta figura se puede notar que están indicadas dos variables, t_1 y t_2, que indican el tiempo desde que inicia la revolución hasta el valor pico obtenido por el foto-sensor y hasta el pulso que indica el fin de la revolución actual e inicio de la otra, sucesivamente.

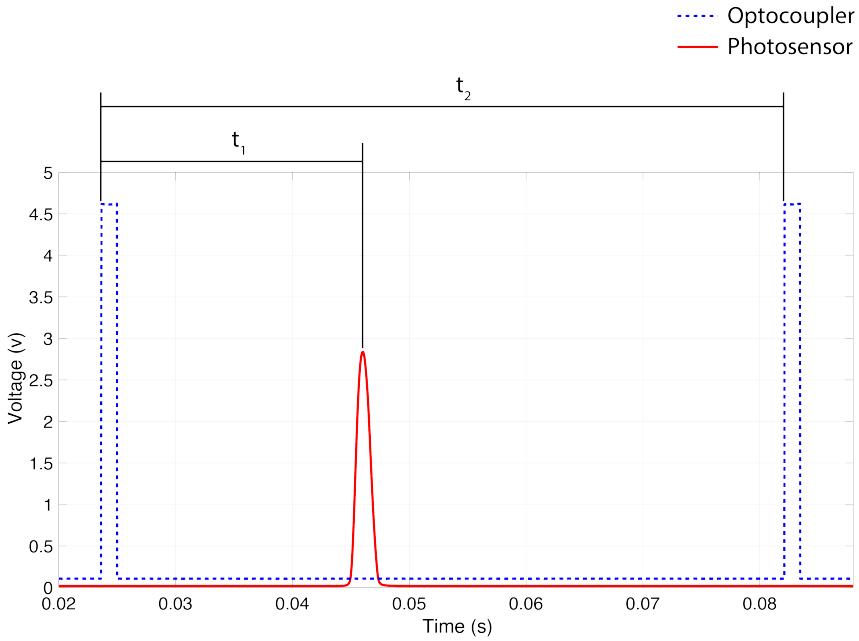


Figura 2.9: Par de señales obtenidas para cada revolución del espejo de 45 grados de la apertura.

Para obtener t_1 y t_2, en cada ciclo de trabajo, el micro-controlador toma una lectura de cada sensor, los guarda en variables independientes y realiza comparaciones para cada valor.

En el caso particular del opto-acoplador, el procesamiento es poco intensivo, ya que solamente compara el valor actual v_{opto} con un valor de referencia predefinido v_{ref}. Cuando $v_{opto} > v_{ref}$ y el valor de referencia $dire_{act}$ que indica si la señal está subiendo o bajando es igual a 1, una bandera c_{proc} se iguala a 1 para indicar que ya se completó el giro. Es aquí donde se guarda en el contador t_1 la cantidad de lecturas tomadas en toda la revolución

La señal del foto-sensor es cuyo procesamiento es más intensivo, ya que hace uso de v_{ref}, $dire_{act}$ y compara de manera constante el valor actual v_{foto} con el valor anterior $v_{fotoAnt}$. Cada que $v_{foto} > v_{fotoAnt}$, se guarda el número de lectura actual en la variable

t_2 y se reemplaza $v_{fotoAnt}$ por v_{foto}.

También, al terminar el ciclo, el microcontrolador envía el resultado a la interfaz de Python por protocolo serial a $9600bps$. El código utilizado en el microcontrolador se puede visualizar en el Apéndice B.

Entonces, la frecuencia con la que los datos son enviados a la interfaz dependen totalmente de la velocidad angular del motor en la apertura, la precisión de la estabilidad con la que el eje del motor rota y la exactitud en la estimación del ángulo de incidencia de la cantidad de muestras adquiridas en cada revolución. Entonces, es deseable que el eje del motor gire a altas velocidades, que su rotación sea uniforme y que la frecuencia de muestreo y procesamiento del micro-controlador sea alta.

2.1.6. Ángulo de incidencia en la apertura

Cada que el opto-acoplador manda un pulso indicando que se completó una revolución, se envía a la interfaz de Python una cadena de texto con el formato $Ot_2\ Ft_1$, que es procesada por un script encargado de separar la cadena en dos variables determinar el ángulo de incidencia utilizando la relación de la Ecuación 2.4.

$$\beta = \frac{360 * t_1}{t_2} \tag{2.4}$$

Nótese que en el algoritmo utilizado en este proyecto, se toma el valor pico de la curva del foto-sensor debido a la rapidez con la que es procesada. Sin embargo, se ha buscado aumentar la precisión en la estimación del ángulo de incidencia con métodos como el presentado por [45], donde explora la estimación de β con el centro energético de la señal.

2.2. Visión Estereoscópica

El funcionamiento de la Visión Estereoscópica, como técnica de estereoscopía, se inspira en la visión humana y utiliza dos cámaras con propiedades y posiciones definidas. La separación entre las cámaras produce paralaje, es decir, cada cámara capta

una perspectiva distinta de la escena. Este fenómeno se aprovecha para realizar una triangulación con las imágenes obtenidas por ambas cámaras, facilitando la estimación de la posición de los elementos dentro del campo de visión de las cámaras.

Esta técnica ofrece como ventaja principal la capacidad de adquirir un gran volumen de datos, permitiendo estimar la profundidad de todos los píxeles que coinciden en las capturas de ambas cámaras. Sin embargo, el proceso de clasificar y segmentar las imágenes para tomar decisiones incrementa significativamente los requerimientos computacionales.

El proceso general para la estimación de coordenadas rectangulares utilizando a la Visión Estereoscópica implica la captura de imágenes, filtrado, calibración, matching y triangulación. Como se puede visualizar en la Figura 2.10, el proceso de calibración solo se realiza una vez y la estimación de coordenadas se realiza de forma iterativa.

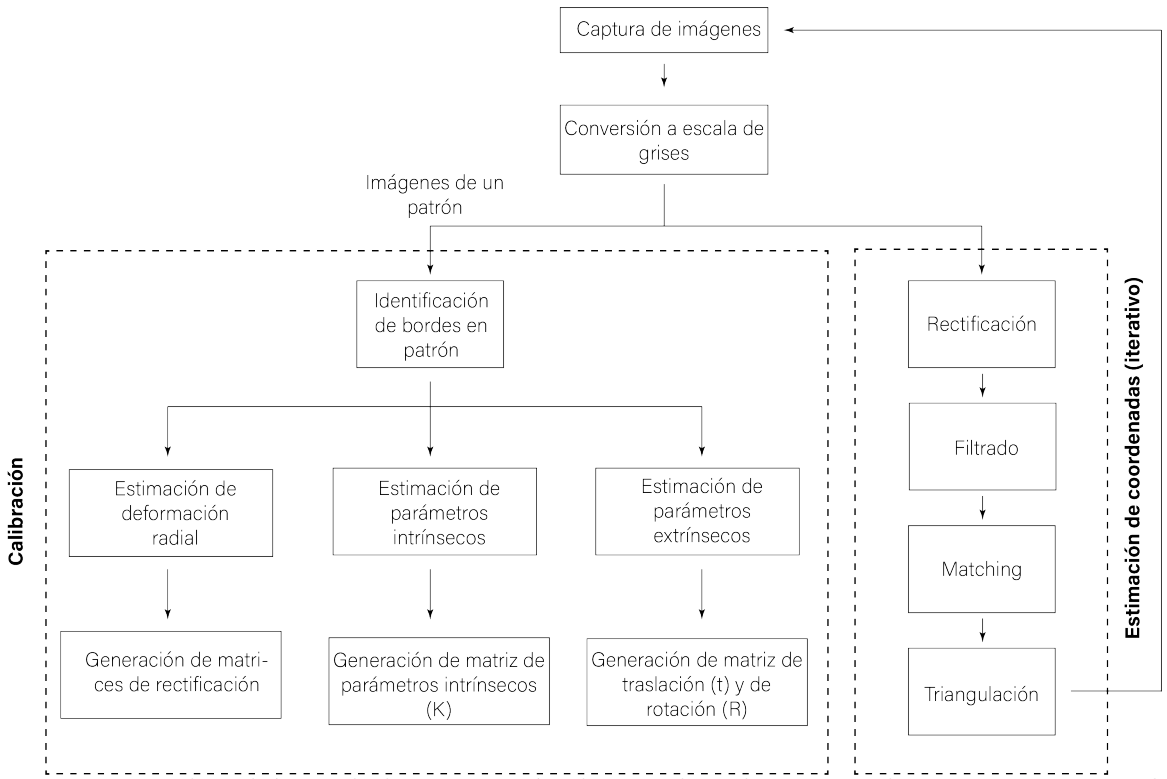


Figura 2.10: Flujo básico de la estimación de coordenadas espaciales con un sistema de Visión Estereoscópica.

En este capítulo se darán detalles del proceso necesario para la estimación de coordenadas rectangulares utilizando la Visión Estereoscópica, desde la calibración hasta la

triangulación.

2.2.1. Arreglo del sistema

Un sistema de Visión Estereoscópica emplea un par de cámaras con parámetros intrínsecos conocidos, como la distancia focal, el alto y ancho de la matriz del sensor y el sesgo en los píxeles del sensor. Cuando los objetos que se encuentran en la escena dentro del campo de visión de ambas cámaras, se puede trazar una linea imaginaria para cada píxel que forma a este objeto en la imagen. Esta linea imaginaria atraviesa el centro del lente de la cámara y llega al sensor, como en la Figura 2.11. Si se conoce la distancia entre los centros de las cámaras, entonces es notable que se puede efectuar una triangulación, como en el caso del LS.

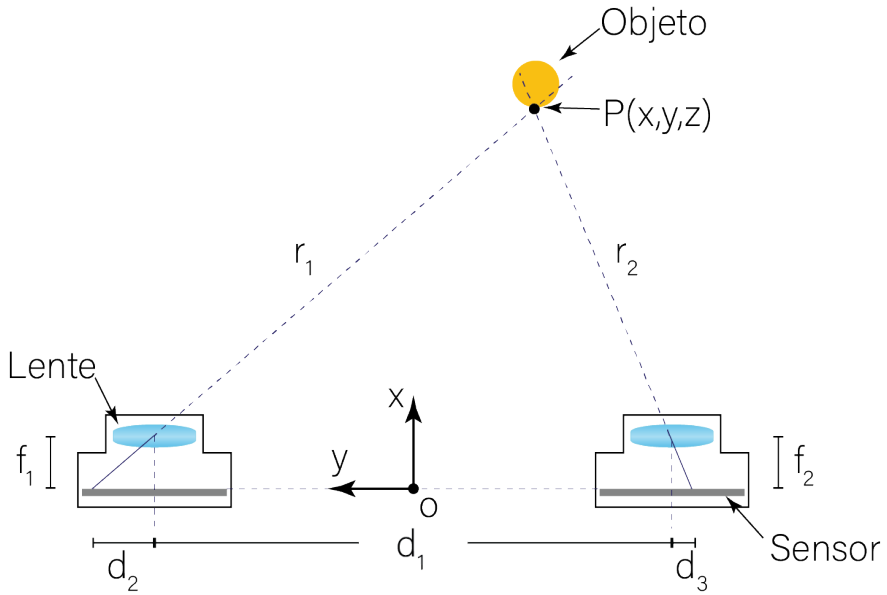


Figura 2.11: Arreglo y parámetros básicos de un sistema de visión estereoscópica.

En la Figura 2.11, la distancia d_1 es la distancia conocida entre los centros de las cámaras, d_2 y d_3 son las distancias del centro del sensor a cada píxel donde haya objetos coincidentes y f_1 y f_2 son las distancias focales (en píxeles) de las cámaras. Basta con una relación geométrica para determinar la profundidad (coordenada x) de $P(x, yz)$, como en la Ecuación 2.5.

$$x = \frac{f d_1}{d_2 - d_3} \tag{2.5}$$

Y a partir de x, se estiman las coordenadas z e y. Donde también se debe considerar la distancia vertical entre el centro del sensor y el píxel donde coinciden los puntos d_4.

$$y = \frac{xd_2}{f_1} = \frac{xd_3}{f_2} \tag{2.6}$$

$$z = \frac{xd_4}{f_1} = \frac{xd_4}{f_2} \tag{2.7}$$

En este caso, la distancia d_4 para las Ecuaciones 2.6 y 2.7 se toma igual para ambas coordenadas debido a que se considera que los centros de las dos cámaras coinciden en la misma linea horizontal.

2.2.2. Calibración de un sistema de Visión Estereoscópica

Para realizar estimaciones precisas de coordenadas rectangulares con un sistema de Visión Estereoscópica, es necesario que las imágenes capturadas por las cámaras estén libres de deformaciones, que los centros de las imágenes estén alineados verticalmente y que los parámetros extrínsecos e intrínsecos de las cámaras se conozcan con exactitud. Esto se logra mediante un proceso de calibración que incluye la corrección de las imágenes y la estimación de la posición relativa de las cámaras. Cada paso en la calibración es crucial para la precisión en la estimación de coordenadas. Al finalizar, se obtienen una matriz de transformación, una matriz de parámetros intrínsecos y dos matrices de corrección de deformación, una por cada cámara.

2.2.2.1. Corrección de deformación

Los objetivos de las cámaras contienen lentes esféricos que redirigen la luz reflejada por los objetos que se encuentran dentro de su campo de visión hacia el sensor de la cámara. Esto quiere decir que es necesario proyectar la luz que se refracta a través de una esfera sobre un plano.

Para lograr esta proyección, en lugar de un solo lente, se utiliza un complejo arreglo de lentes que buscan ajustar el tamaño de la imagen reflejada sobre el plano fijo del sensor. Sin embargo, como cualquier proyección esférica sobre un plano, se generan in-

evitables deformaciones sobre la imagen que alteran la forma de los elementos captados por las cámaras. Es decir, las lineas que forman a los elementos se comienzan a deformar dependiendo de qué lentes se utilizaron y cómo están acomodados. Estas deformaciones pueden ser poco o muy notorias de acuerdo con la distancia focal del objetivo utilizado. Normalmente, las distancias focales menores generan mayor deformación, ya que para aumentar el campo de visión, el radio de los lentes en el objetivo también debe aumentar.

Actualmente, existe una clasificación de diversos tipos de deformaciones en imágenes, como la deformación de barril, de almohada y de bigote que se muestran en la Figura 2.12.

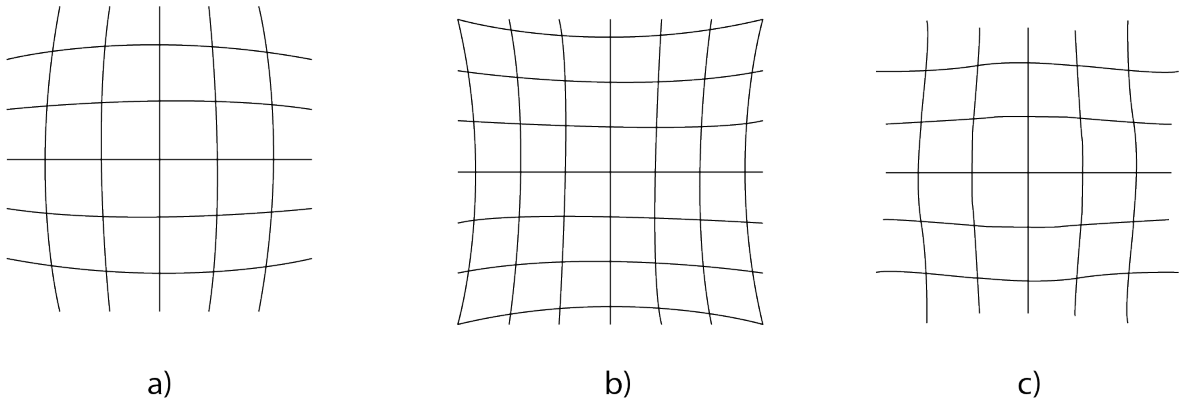


Figura 2.12: Deformaciones en imágenes a) de barril, b) almohada y c) bigote.

Realizar cálculos de posiciones tridimensionales con cámaras sin corregir deformaciones implica que las estimaciones no serán precisas. Las deformaciones en la imagen distorsionan la representación real de la escena, afectando la exactitud de las estimaciones tridimensionales.

Para corregir la deformación, se utilizan patrones con características y dimensiones conocidas. Estas características usualmente son formas conocidas que pueden ser fácilmente identificables por algoritmos de procesamiento de imágenes. El caso más usado es el de un plano con un patrón de tablero de ajedrez impreso, ya que los bordes de los recuadros son fáciles de ubicar espacialmente, considerando que sus dimensiones son conocidas, como se muestra en la Figura 2.13.

Teniendo en cuenta que los puntos detectados sobre el patrón de ajedrez deberían formar una línea recta, pero debido a las deformaciones en las imágenes esto no ocurre,

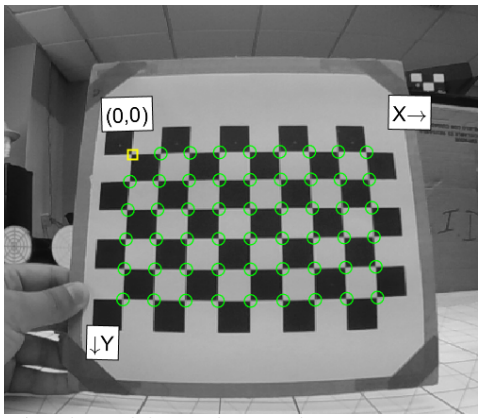
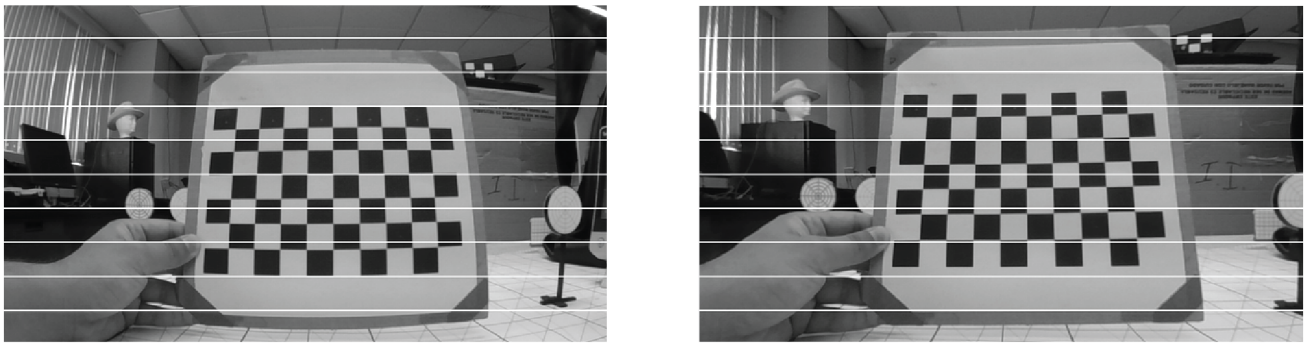


Figura 2.13: Hoja con patrón de tablero de ajedrez con puntos de intersección identificados (en verde).

se estima el desplazamiento necesario de los puntos para generar una matriz. Esta matriz indica cuánto se deben desplazar todos los píxeles para obtener una imagen rectificada. En la Figura 2.14-a se logra visualizar la deformación generada por un lente con un campo de visión de 125^o, las lineas blancas sirven como referencia para notar la curvatura en los recuadros del patrón. En la Figura 2.14-b se se aprecia el resultado de la rectificación de una imagen. En este caso particular se utilizó la librería OpenCV en Python para obtener las matrices de rectificación de cada imagen y corregir las deformaciones.



a) b)

Figura 2.14: Comparación de una imagen antes y después del proceso de rectificación a) imagen sin rectificar b) imagen rectificada.

Una vez que la imagen está rectificada, se tiene una representación más fiel de la escena real.

2.2.2.2. Estimación de parámetros intrínsecos

Los parámetros intrínsecos de una cámara son parte importante para el dimensionamiento de objetos reales. Estos parámetros, únicos para cada cámara, dependen de la calidad de los elementos ópticos del objetivo y la precisión en la construcción del sensor de la cámara.

La matriz K de parámetros intrínsecos (Ecuación 2.8) de dimensiones 3×3 incluye f_x y f_y, que representan la distancia focal en las direcciones horizontal y vertical, respectivamente. El parámetro de sesgo (s) indica la deformación en los píxeles del sensor; si $s = 0$, se asume que los píxeles son cuadrados. Los últimos parámetros, R_x y R_y, representan las coordenadas del centro óptico del sensor en las direcciones horizontal y vertical, respectivamente.

$$K = \begin{bmatrix} f_x & s & R_x \\ 0 & f_y & R_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

Para estimar los parámetros intrínsecos de una cámara, se emplean imágenes capturadas de un tablero con un patrón de ajedrez, similar al proceso utilizado para la corrección de deformaciones ópticas (Figura 2.13). Este método se basa en la relación geométrica conocida entre los puntos del patrón de ajedrez en el mundo real y su representación en la imagen capturada.

La matriz de parámetros intrínsecos K, se determina a través de un proceso iterativo. En cada iteración, se proponen valores para los parámetros intrínsecos de la cámara y se calcula una imagen reproyectada utilizando estos valores propuestos. Luego, se estima el error entre las coordenadas de los puntos reproyectados y las coordenadas reales de los puntos en las imágenes capturadas, basándose en la distancia conocida entre los bordes de los cuadros del patrón de ajedrez.

Para minimizar el error de reproyección y afinar la estimación de los parámetros intrínsecos, se pueden utilizar técnicas de optimización como el método de mínimos cuadrados. Este enfoque ajusta los parámetros de la matriz K para minimizar la suma

del cuadrado de las diferencias (errores) entre las posiciones de los puntos observados en las imágenes y las posiciones calculadas a partir del modelo matemático de la cámara con los parámetros intrínsecos propuestos.

2.2.2.3. Estimación de parámetros extrínsecos

Para realizar la triangulación utilizando dos cámaras, es necesario conocer tanto los parámetros intrínsecos como los extrínsecos. Los parámetros extrínsecos definen la posición y orientación relativas entre las cámaras, considerando una de ellas como el punto de referencia.

Los parámetros extrínsecos se pueden representar por separado con la matriz de traslación (t) y la matriz de rotación (R). La matriz de traslación indica el desplazamiento del origen de la cámara derecha con respecto a la izquierda, esta matriz de 3×1 contiene el desplazamiento en cada eje, como se muestra en la Ecuación 2.9.

$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.9}$$

En cambio, la matriz de rotación representa la diferencia de orientación de las cámaras, medida con los ángulos ψ, θ y ϕ, como se expresa en la Ecuación 2.10.

$$R = R_z(\psi)R_y(\theta)R_x(\phi) = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{2.10}$$

Ambas matrices se obtienen en el mismo proceso utilizado para la matriz de parámetros intrínsecos. Es decir, también se proponen valores para las matrices de traslación y rotación y se optimizan los valores hasta que se obtiene el mínimo error de reproyección.

Dependerá del usuario verificar que con estas matrices la estimación de coordenadas rectangulares de los elementos en una escena son correctos o si es necesario repetir el proceso de calibración.

Una vez obtenidas las matrices de corrección, así como las matrices de parámetros intrínsecos y extrínsecos, se puede considerar que la calibración ha concluido. Si no ocurre ningún cambio en la posición relativa entre las cámaras, todos los valores obtenidos durante la calibración se mantienen constantes. Sin embargo, asegurar que ambas cámaras conserven permanentemente la misma perspectiva y posición relativa es prácticamente inviable; por lo tanto, es necesario realizar la calibración nuevamente cada vez que el error en la estimación de coordenadas rectangulares se incremente hasta el punto de afectar el desempeño del sistema que emplea la Visión Estereoscópica.

Actualmente, hay varios algoritmos diseñados para calibrar sistemas de Visión Estereoscópica de manera autónoma, los cuales emplean patrones estacionarios colocados frente al sistema como referencias para facilitar la autocalibración [46, 47].

2.2.3. Filtrado

Para facilitar el procesamiento de las imágenes en la etapa de emparejamiento, como se muestra en la Figura 2.10, se realiza un filtrado previo. Este proceso implica reducir el ruido presente en las imágenes y resaltar las texturas, con el objetivo de mejorar la detección de objetos en ambas cámaras.

A continuación se describirán algunos de los filtros más utilizados en Visión Estereoscópica y las ventajas de su aplicación en esta técnica.

2.2.3.1. Filtro de la mediana

El filtro de la mediana es uno de los más utilizados para la reducción de ruido en imágenes. Es especialmente útil en imágenes de visión estereoscópica porque ayuda a preservar los bordes y detalles de la imagen mientras reduce el ruido aleatorio, lo cual es crucial para mantener la precisión en la detección de características y en la estimación de profundidad que se realiza a partir de la comparación de las imágenes capturadas por ambas cámaras. Este filtro funciona reemplazando cada píxel de la imagen por la mediana de los valores de intensidad en su vecindad. Este método es efectivo para reducir el ruido aleatorio, especialmente el ruido de tipo *salt and pepper*, sin desdibujar los bordes de los objetos.

2.2.3.2. Filtro Gaussiano

El filtro gaussiano suaviza las imágenes reduciendo el detalle y el ruido, aplicando una convolución con una función Gaussiana. Este enfoque difumina la imagen de manera que los píxeles cercanos tienen mayor influencia sobre el resultado que los más distantes, logrando un efecto de suavizado proporcional a la distancia.

En Visión Estereoscópica, el filtro Gaussiano es útil para mejorar el matching entre imágenes al minimizar el ruido y resaltar las estructuras importantes. Al suavizar las imágenes antes de calcular la disparidad, facilita la identificación precisa de puntos correspondientes entre las vistas de las cámaras, relevante para la estimación de posiciones rectangulares.

2.2.3.3. Filtro Guiado Adaptativo

El filtro guiado adaptativo es una técnica avanzada de procesamiento de imágenes que ajusta dinámicamente sus parámetros en función del contenido local de la imagen. A diferencia de los filtros tradicionales como el de la mediana y el Gaussiano, que aplican un mismo nivel de suavizado en toda la imagen, el filtro guiado adaptativo varía la intensidad del suavizado para preservar los bordes y detalles finos mientras reduce eficazmente el ruido en áreas con poca variación de texturas. Esto se realiza analizando una imagen guía, que puede ser la misma imagen u otra relacionada, para dirigir el proceso de filtrado.

En Visión Estereoscópica, el filtro guiado adaptativo es muy utilizado con el fin de mejorar el matching. Al aplicar este filtro, se pueden atenuar las discrepancias debidas al ruido y a las variaciones de textura, facilitando la identificación precisa de los puntos correspondientes en las imágenes estéreo.

2.2.3.4. Transformación Census

La transformación Census es un método utilizado para la comparación de píxeles en el procesamiento de imágenes y visión por computadora, especialmente útil en la estimación de disparidad para visión estereoscópica. Este algoritmo convierte cada píxel

de la imagen en un código binario basado en la comparación de la intensidad del píxel central con las de sus vecinos dentro de una ventana específica. La comparación resulta en un 1 si la intensidad del píxel vecino es menor que la del píxel central, y en un 0 en caso contrario. Este proceso de binarización captura la estructura local de la textura alrededor de cada píxel, haciéndolo robusto frente a variaciones de iluminación y ruido.

En la visión estereoscópica, la transformación Census se aplica para mejorar la correspondencia entre las imágenes de las dos cámaras, facilitando la identificación de los píxeles correspondientes en ambas imágenes. Al utilizar los códigos binarios generados por la transformación Census, es posible realizar comparaciones robustas entre píxeles que consideran la textura local en lugar de solo la intensidad, lo que mejora la precisión en la estimación de la disparidad. Esto es especialmente valioso en escenas con patrones repetitivos o en condiciones de iluminación desafiantes, donde los métodos basados en intensidad directa pueden fallar.

2.2.4. Matching

El reto que genera mayor error en las mediciones de la visión estereoscópica es la correspondencia de puntos, o *matching*. Esto es, identificar que punto de luz incidido en un sensor es el correspondiente en el sensor de la otra cámara.

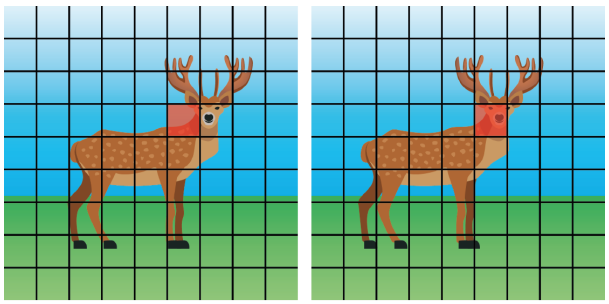


Figura 2.15: Ejemplo de error de correspondencia o matching en un sector de dos imágenes tomadas por un sistema de Visión Estereoscópica. a) imagen izquierda b) imagen derecha

Este error se presenta debido a que una cámara está desplazada con respecto a la otra y por lo tanto el punto de vista de la escena de cada cámara es distinto. Por lo que, si se desearan comparar dos datos por su ubicación en la matriz, se estaría analizando

elementos o profundidades distintos y, por lo tanto, el cálculo sería erróneo.

En la Figura 2.15 se muestran dos imágenes que representan lo que capturaría cada cámara de un sistema de Visión Estereoscópica, donde cada imagen aparece ligeramente desplazada respecto a la otra. Si se quisiera analizar el elemento dentro del recuadro resaltado en rojo en la imagen de la izquierda y se comparase con el píxel en la misma posición en la imagen de la derecha, en realidad se estaría comparando la ubicación de un elemento de la escena con otro diferente. Esto produce un error en el cálculo de la profundidad, ya que no se está comparando el mismo punto de la escena en ambas imágenes.

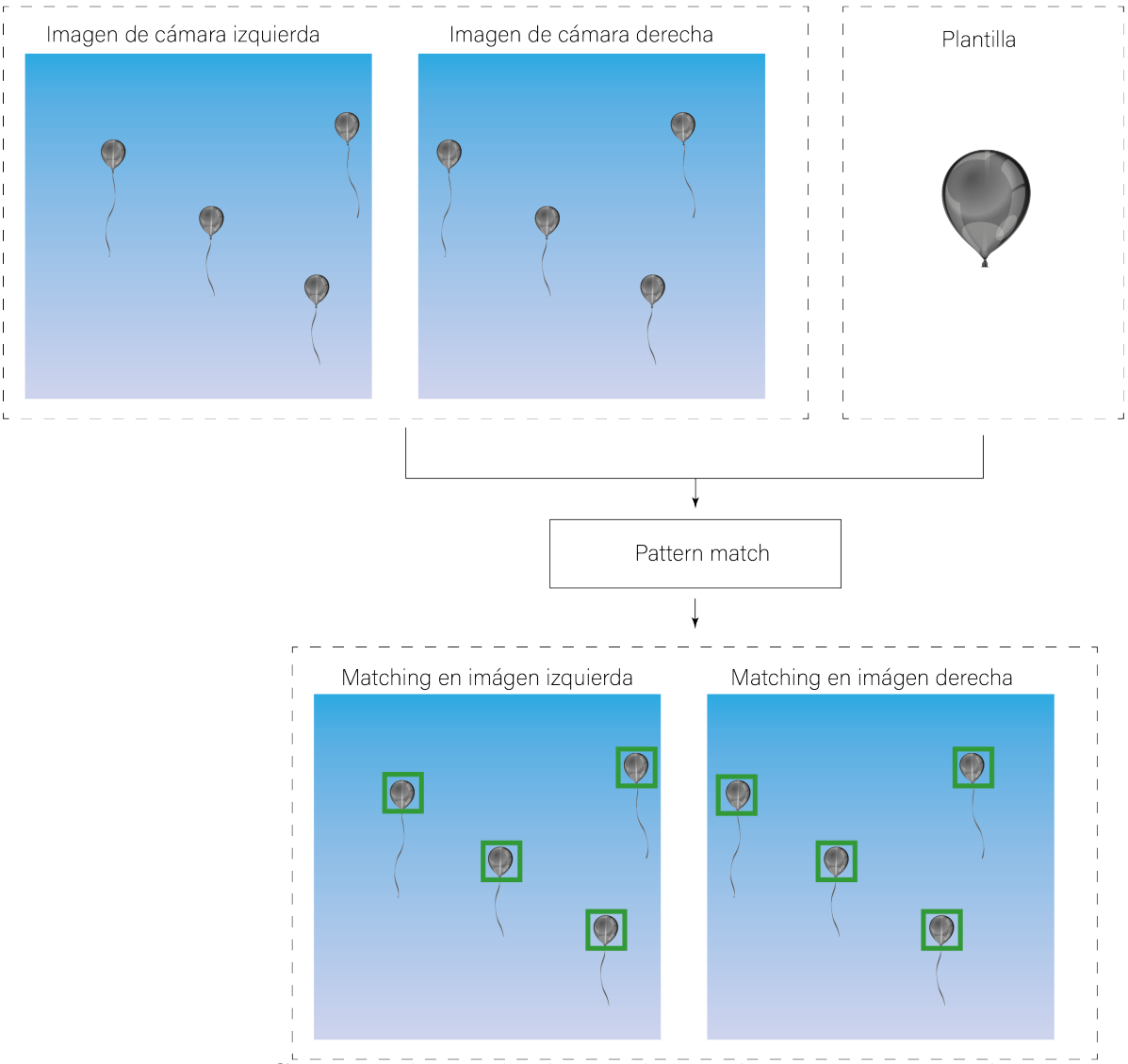


Figura 2.16: Correspondencia de objetos de interés con pattern match.

El objetivo del matching es identificar y localizar los mismos objetos en las matrices de imágenes capturadas por diferentes cámaras. Para lograr esta tarea, se han desarrollado diversos métodos. Uno de estos métodos es el *pattern matching*, que se emplea cuando el usuario está interesado en encontrar objetos que cumplan con características específicas, mientras ignora el resto de la información presente en la imagen. Este tipo de correspondencia se centra en la comparación de regiones específicas de la imagen con un patrón o modelo previamente definido. Esto permite identificar la presencia y la posición de objetos o características que coinciden con el patrón dentro de la imagen, como se muestra en la Figura 2.16.

El pattern matching logra de manera eficaz y precisa la localización de objetos específicos de interés, pero limita su alcance al posicionamiento de unos pocos elementos seleccionados, ignorando la presencia de otros objetos en la escena. Si lo que se busca es estimar la posición de todos los objetos dentro del campo de visión de un sistema de Visión Estereoscópica, la técnica adecuada es el block matching. Esta técnica implica la división de las imágenes en una serie de bloques o regiones pequeñas y la comparación de cada uno de estos bloques entre las diferentes imágenes para encontrar correspondencias (Figura 2.17).

Cada vez que se encuentra una correspondencia entre el bloque de una imagen y una sección correspondiente en la otra, se registra la distancia entre la posición del centro del bloque en una imagen y su ubicación correspondiente en la imagen de la otra cámara, antes de seleccionar un nuevo bloque en la primera imagen. Esta distancia es la diferencia entre las distancias d_2 y d_3 que se muestra en la Figura 2.11 y se le conoce como *disparidad*. Con el block matching se busca obtener la disparidad entre cada pixel coincidente de ambas cámaras, formando el *mapa de disparidades*. Un ejemplo de un mapa de disparidades se muestra en la parte inferior de la Figura 2.17.

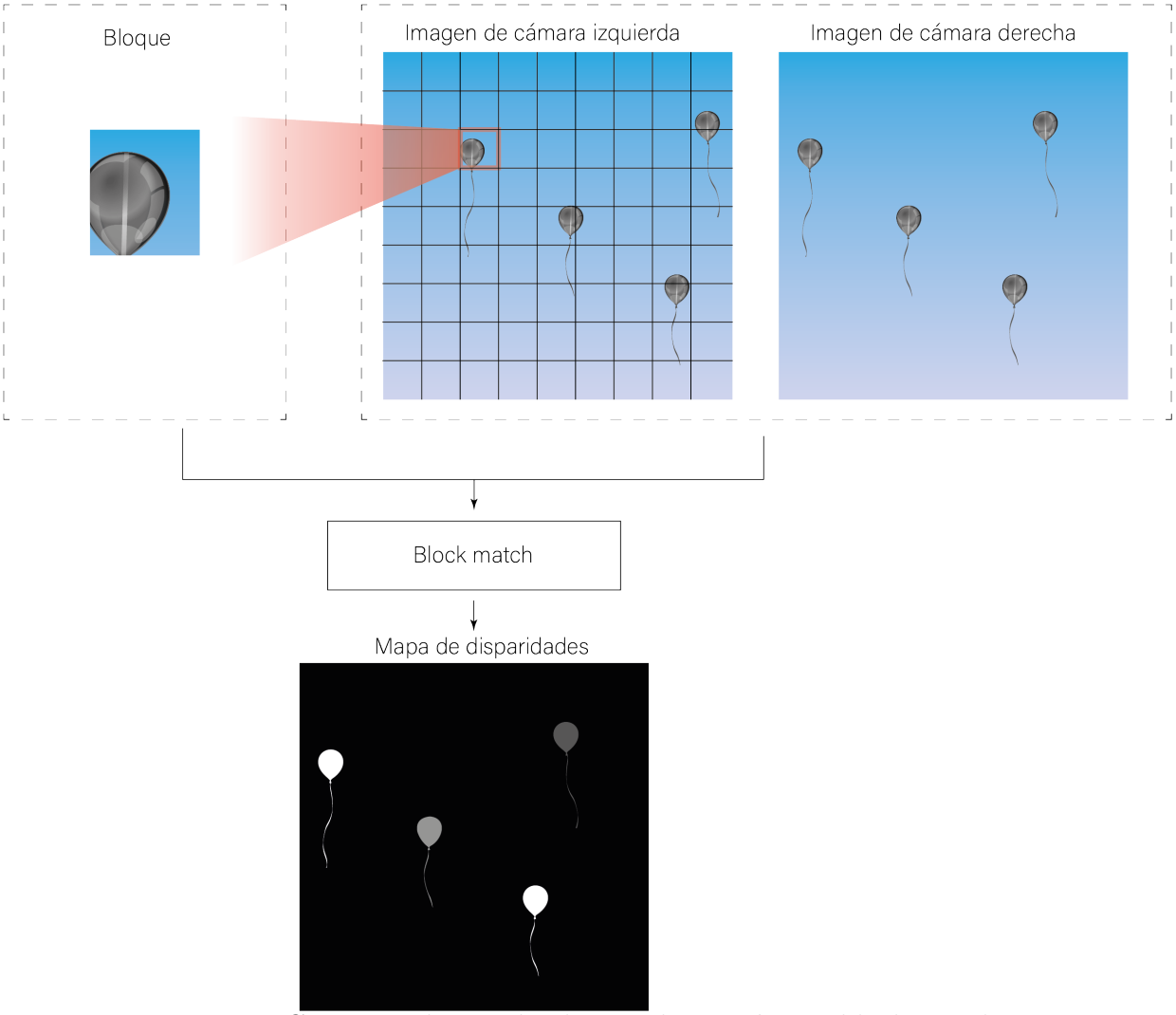


Figura 2.17: Correspondencia de objetos de interés con block match.

Tomando como ejemplo un caso real, en la Figura 2.18-c es notable como hay puntos con tonalidades grises que indican las zonas donde no se pudo llevar a cabo el matching con éxito, generando regiones con falta de información. Es evidente que en estas regiones no sería posible hacer una estimación de coordenadas rectangulares. Por lo que cualquier objeto que sea representado por estos píxeles, no podrá ser ubicado espacialmente.

Existen diversas condiciones que pueden llevar al error en el matching de regiones específicas. Algunos de estos casos son la falta de nitidez de las imágenes, oclusión o efectos ópticos que se perciben de distinta forma para cada cámara.

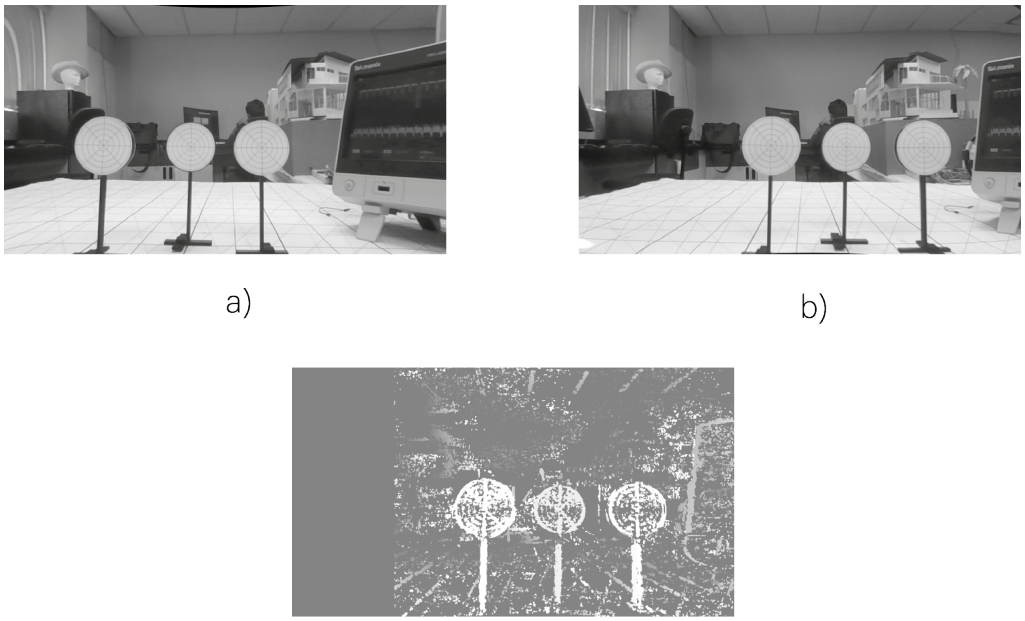


Figura 2.18: Ejemplo de mapa de disparidades. a) Imagen izquierda b) Imagen derecha c) Mapa de disparidades.

Buscando mejorar estos resultados, existen tres parámetros básicos que se ajustan de acuerdo a la escena que se desea analizar. El primer parámetro es el *window size* (W_s) que indica el tamaño de las muestras que se tomarán de la primera imagen para ser buscado en la segunda. Este valor siempre debe de ser un número non, con el fin de que siempre exista un pixel central en el recuadro o *window*. Las variaciones en W_s afectan a la suavidad de los objetos en el mapa de disparidades, entre mayor sea el valor de W_s, se van perdiendo detalles de la escena, pero se pueden llenar espacios que con un bajo valor de W_s pueden quedar vacios. En la Figura 2.19 se muestran diferentes valores de W_s y como afectan al mapa de disparidades.

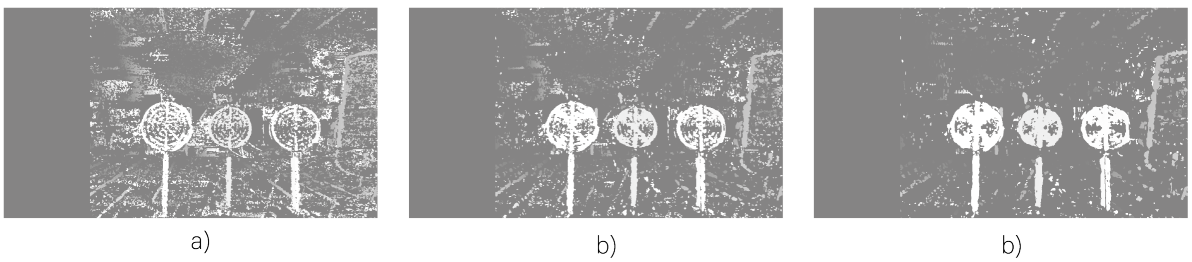


Figura 2.19: Efecto de la variación de W_s en el mapa de disparidades. a) $W_s = 5$ b) $W_s = 9$ c) $W_s = 13$.

El segundo parámetro es la disparidad máxima (d_{max}), que indica el valor máximo de disparidad permisible en el mapa de disparidad. Cualquier valor por encima de d_{max} se convierte en cero, con el fin de indicar que no se desea tener ese valor en el mapa de disparidades. En la estimación de coordenadas rectangulares, d_{max} indica la distancia mínima desde la que se desea medir, tomando como referencia al origen del sistema de Visión Estereoscópica. En la Figura 2.20 se puede apreciar como los objetos más cercanos aparece conforme aumenta el valor de d_{max}.

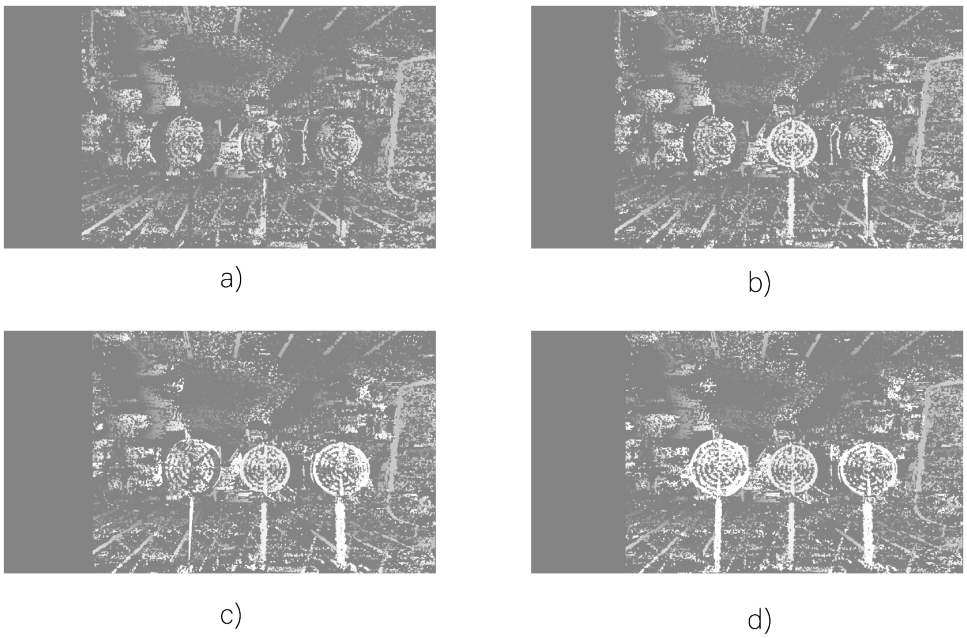


Figura 2.20: Efecto de la variación de d_{max} en el mapa de disparidades. a) $d_{max} = 132$ b) $d_{max} = 148$ c) $d_{max} = 164$ d) $d_{max} = 180$.

El tercer parámetro es la disparidad mínima (d_{min}) que indica el mínimo valor de disparidad permitido. La condición para d_{min} es opuesta a d_{max}, cualquier valor por debajo de d_{min} se convierte en cero. El parámetro d_{min} indica la distancia máxima hasta la que se estimará la profundidad. Entre más pequeño sea el valor de d_{min}, se tomarán las distancias de objetos más lejanos, como se muestra en la Figura 2.21, donde se visualiza como se comienzan a mostrar los objetos del fondo conforme disminuye el valor de d_{min}.

Se puede notar que d_{min} y d_{max} son el rango de visión del sistema de Visión Estereoscópica cuyos valores dependen totalmente de la aplicación en la que será utilizada este sistema y de las preferencias del usuario.

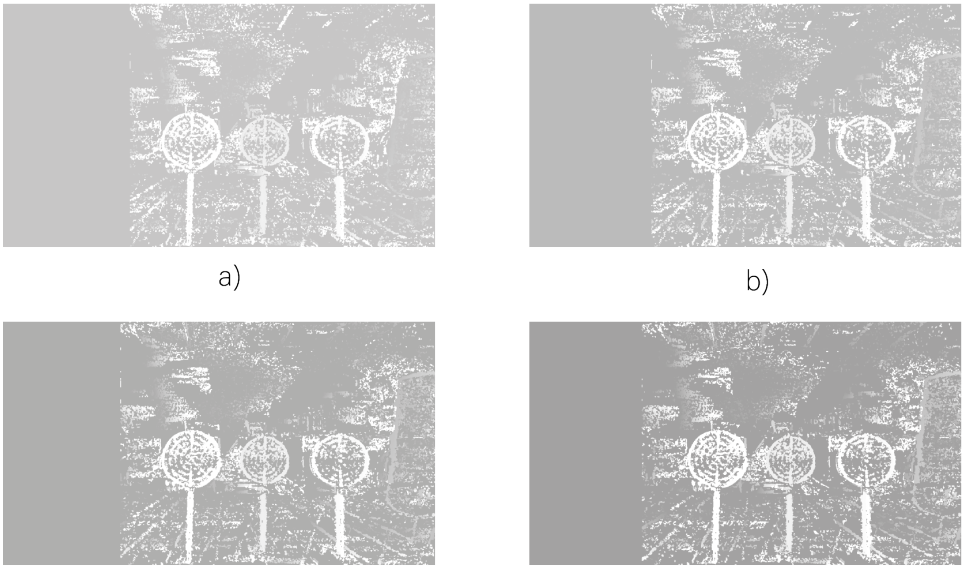


Figura 2.21: Efecto de la variación de d_{min} en el mapa de disparidades. a) $d_{min} = 180$ b) $d_{min} = 165$ c) $d_{min} = 150$ d) $d_{min} = 135$.

Además de W_s,d_{min} y d_{max}, existen otros parámetros que afectan al mapa disparidades resultante del matching, sin embargo, estos parámetros pueden ser o no utilizados dependiendo del algoritmo seleccionado para el matching.

A pesar de que el block matching tiene una metodología básica, existen diversos algoritmos que buscan optimizar la búsqueda y comparación de los bloques en ambas imágenes. Actualmente existen los algorítmos clásicos de matching, como la Suma de Diferencias Absolutas (SAD), Suma de Diferencias al Cuadrado (SSD), la Correlación Cruzada Normalizada (NCC) y el Semi-Global Block Matching (SGBM).

Todos estos algoritmos están enfocados en buscar de la forma más rápida y robusta posible a los bloques correspondientes. Como unidad de medida de la similitud entre bloques se introduce el término *costo* C. Si $C = 0$ entonces ambos bloques se consideran idénticos y entre más alejado sea C de cero, habrá más diferencias entre los bloques. Sin embargo aunque C tenga un valor alto, se toma el bloque cuyo costo sea el menor encontrado. Cada algoritmo tiene función de costo que permite determinar la disparidad entre bloques de dos imágenes. A continuación se describirán cada uno de estos algoritmos y las ventajas de su implementación en el matching.

2.2.4.1. Suma de Diferencias Absolutas

La Suma de Diferencias Absolutas implica comparar bloques o muestras de una imagen con otra, realizando una simple resta entre bloques de datos y sumando las diferencias obtenidas. El bloque de datos con el resultado más bajo se considera el más similar al bloque de muestra, lo que indica una correspondencia entre ambos bloques. Matemáticamente la función de costo del SAD se expresa como en la Ecuación 2.11, donde W_s sigue siendo el *window size* ya mencionado, I_L e I_R son las imágenes, d es la disparidad y los índices i y j indican la posición discreta de los bloques.

$$SAD(i,j,d) = \sum_{i,j \epsilon W_s} |I_L(i,j) - I_R(i,j-d)| \tag{2.11}$$

Este algoritmo destaca por su rapidez en la ejecución de la operación. Sin embargo, dado que la operación se efectúa directamente sobre las intensidades de luz capturadas por los sensores de las cámaras, cualquier variación en la iluminación entre las imágenes captadas puede causar un error de correspondencia. Esto podría llevar a correlacionar dos objetos que en realidad no se corresponden.

2.2.4.2. Suma de Diferencias Cuadráticas

El algoritmo de Suma de Diferencias Cuadráticas SSD es una variación del SAD. El SSD supone elevar al cuadrado los términos dentro de la sumatoria de diferencias en la función de costo (Ecuación 2.12) con el fin de disminuir el efecto de los cambios de iluminación.

$$SSD(i,j,d) = \sum_{i,j \epsilon W_s} (I_L(i,j) - I_R(i,j-d))^2 \tag{2.12}$$

Y aunque este cambio en apariencia sea simple, implica agregar una operación adicional al proceso la obtención de diferencias, por lo que es evidente que se obtiene un aumento en tiempo de procesamiento.

2.2.4.3. Correlación Cruzada Normalizada

Buscando la mayor robustés ante cambios lumínicos que puedan afectar a la correspondencia se llega a la Correlación Cruzada Normalizada (NCC). Este algoritmo de alta demanda computacional es útil cuando es requerido un resultado robusto y no es crítico el tiempo de procesamiento. El costo entre bloques se obtiene como se presenta en la Ecuación 2.13.

$$NCC(i,j,d) = \frac{\sum_{i,j\epsilon W_s} I_L(i,j) \cdot I_R(i,j-d)}{\sqrt{\sum_{i,j\epsilon W_s} I_L(i,j)^2 \cdot I_R(i,j-d)^2}} \tag{2.13}$$

2.2.4.4. Semi-Global Block Matching

Entre los métodos clásicos, el Semi-Global Block Matching (SGBM) es el más utilizado gracias al equilibrio entre robustez y velocidad de procesamiento que se ha conseguido con librerías como OpenCV. Los métodos mencionados anteriormente buscan el bloque de datos correspondiente en la misma línea horizontal de la otra imagen. Esto se debe a que el proceso de rectificación busca que ambas cámaras tengan sus centros alineados, por lo tanto, es lógico que el bloque buscado deba estar alineado con el bloque de muestra. Sin embargo, existen diversas condiciones que pueden generar errores de correspondencia en algoritmos como el SAD, SSD y NCC, tales como los cambios abruptos de profundidad y las oclusiones.

Bajo estas consideraciones, el SGBM es una aproximación que propone buscar el bloque muestra en la otra imagen en más de una dirección. Esto permite que los mapas de disparidades sean más densos y por lo tanto, se tiene más información de la escena. En general la función de costo del SGBM se puede expresar como en la Ecuación 2.14.

$$E(d) = \sum_{p}(C(p,d_p) + \sum_{q\epsilon N_p} PT[d_p - dq| \geq 1]) \tag{2.14}$$

Donde el término $C(p,d_p)$ define el costo y $PT[d_p - dq| \geq 1]$ es una penalidad para todos los pixeles (p) cuyos vecinos (q) tengan una disparidad distinta.

2.2.4.5. Filtrado de Mapas de Disparidades

Un paso opcional posterior al matching es la aplicación de filtros sobre el mapa de disparidades [48]. Los filtros que se pueden emplear son los clásicos ya mencionados como el Filtro de la Mediana, Filtro Gaussiano o el Filtro Guiado Adaptativo. Al implementar estos filtros se busca reducir el ruido, suavizar bordes y rellenar espacios vacíos de objetos en los que falló la correspondencia. En la Figura 2.22-b se muestra como la aplicación de un filtro de la mediana con un tamaño de muestra de 7×7 reduce el ruido en el mapa de disparidades.

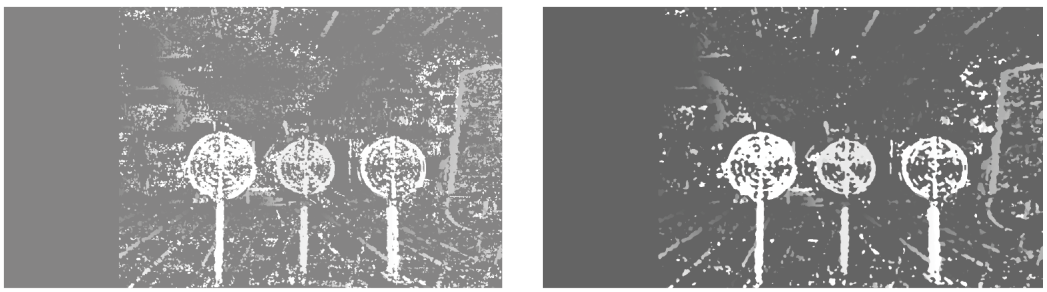


Figura 2.22: Reducción de ruido de un mapa de disparidades con el filtro de la mediana. a) Imagen sin filtrar b) Imagen filtrada

2.2.4.6. Matching con Redes Neuronales

Los métodos clásicos y el filtrado de mapas de disparidades siguen siendo valiosas herramientas para obtener mapas de disparidades de forma rápida. Sin embargo, existen diversos fenómenos ópticos, como la reflexión en superficies no Lambertianas y las oclusiones, que generan regiones con falta de información debido al ruido óptico. Esto conlleva a la implementación de algoritmos de post-procesamiento de mapas de disparidades cuya tasa de éxito no es elevada, ya que al haber falta de información solo se rellenan espacios con interpolaciones y estimaciones que desconocen la geometría del objeto donde se está generando este relleno [49, 50].

Hoy en día, gracias al exponencial incremento en la capacidad de procesamiento de las tarjetas gráficas, se ha ido dejando de lado la optimización de los algoritmos utilizados en el matching y se opta por entrenar grandes modelos de redes neurona-

les. Estos modelos buscan el mayor refinamiento posible en los mapas de disparidades sin preocuparse por lo complejo o costoso que puede ser el hardware requerido para procesarlos.

La Universidad de Middlebury tiene a disposición de los investigadores una de las bases de datos más populares para el entrenamiento y evaluación de algoritmos de visión estereoscópica, el *Middlebury Stereo Evaluation - Version 3* [51], que enlista a los algoritmos con menor error en la generación de mapas de disparidad [52]. En esta lista, se pueden visualizar los algoritmos para el matching con redes neuronales más avanzados aplicados a escenas mayormente de espacios interiores. En la Figura 2.23-b, se muestra el mapa disparidades del algoritmo con promedio de error más bajo (2.51 %), actualmente, según la lista de Middlebury.

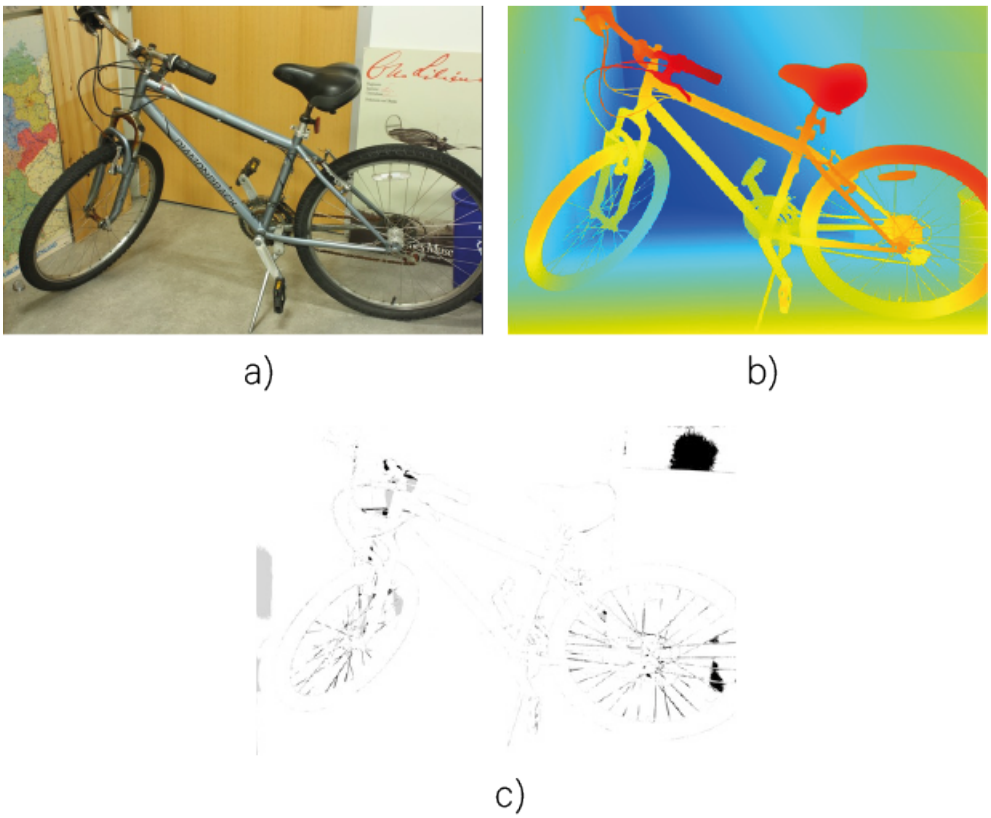


Figura 2.23: Disparidad de una imagen de prueba de Middlebury obtenida con Selective-IGEV. a) Imagen original b) Mapa de disparidades c) Errores marcados en negro.

Otra importante base de datos para el entrenamiento y evaluación es la *KITTI Vision Benchmark Suite*, proyecto del Instituto Tecnológico de Karlsruhe y el Instituto Tecnológico de Toyota en Chicago [53]. Esta base de datos se diferencia de la presentada

en Middlebury por el uso de imágenes de escenas reales y con condiciones de mayor complejidad de resolución para los algoritmos de visión estereoscópica [54]. También tiene un mayor enfoque en la visión artificial aplicada a la navegación autónoma. En la Figura 2.24-b, se muestra el mapa disparidades del algoritmo con promedio de error más bajo (1.28 %), actualmente, según la lista de KITTI.

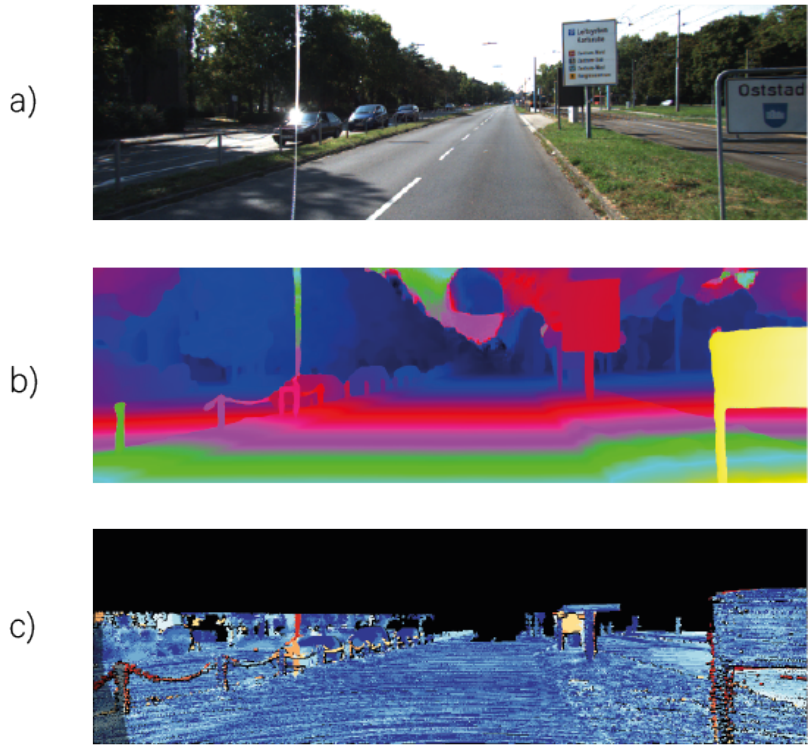


Figura 2.24: Disparidad de una imagen de prueba de KITTI obtenida con OpenStereo. a) Imagen original b) Mapa de disparidades c) Errores marcados en negro.

Entonces, si lo que se busca es obtener mapas de disparidades con el mínimo de ruido y falta de información, entonces se puede recurrir a la implementación de matching con redes neuronales. En este método, los algoritmos clásicos son solo una etapa en toda una metodología que busca obtener un mapa de disparidades preciso.

El uso de redes neuronales para el matching resulta atractivo; sin embargo, siempre es importante analizar su uso en la práctica, ya que el hardware requerido para su funcionamiento es complejo y costoso, lo que para aplicaciones portátiles no siempre resulta conveniente.

2.3. Segmentación de imágenes

Cuando se calculan las coordenadas espaciales de todos los elementos en una escena utilizando a la Visión Estereoscópica, el sistema genera una matriz de datos tridimensionales (nube de puntos) que no distingue a unos objetos de otros. La tarea de identificación y clasificación de objetos en las imágenes corresponde a la segmentación, que en general, es utilizada para clasificar datos que cumplen con cualidades específicas predefinidas por el usuario.

Entonces, será necesario un algoritmo de segmentación capaz de distinguir a las hojas de las plantas del resto de los elementos, con el fin de utilizar esta identificación en una imagen para extraer sus coordenadas espaciales correspondientes. En la Figura 2.25 se muestra como el proceso de segmentación para la extracción de coordenadas rectangulares.

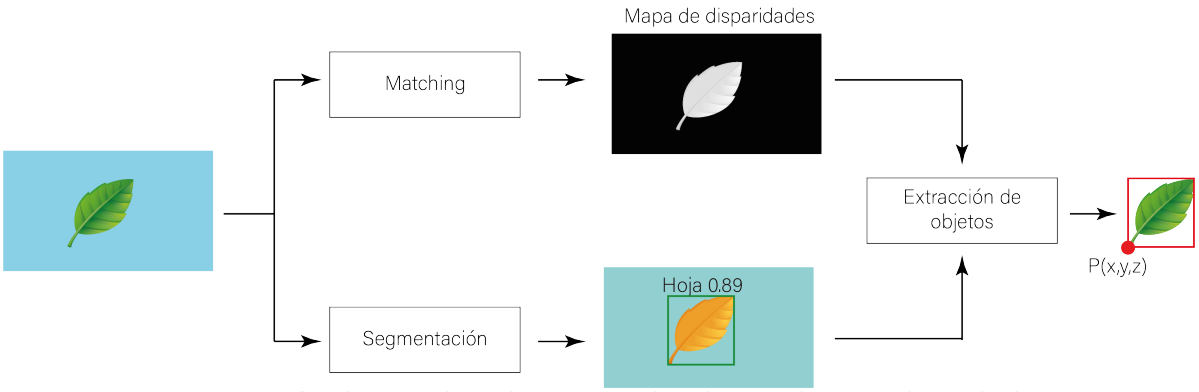


Figura 2.25: Extracción de coordenadas espaciales de una hoja utilizando la segmentación en imágenes.

Para lograr la segmentación de imágenes, es posible optar tanto por métodos clásicos como por el empleo de Redes Neuronales. Los métodos clásicos se basan en la identificación de características específicas definidas por el usuario, las cuales son utilizadas para extraer Regiones de Interés (ROI) de una imagen. Estas características pueden incluir propiedades como la intensidad del color, textura, bordes, entre otros. El proceso consiste en analizar la imagen en busca de patrones o criterios que coincidan con estas características para segmentar o dividir la imagen en partes o regiones que sean de especial interés para cualquier análisis posterior. Normalmente, se ajustan los criterios

del algoritmo de segmentación para el objeto o área particular que se desea segmentar, sin embargo, si ésta área contiene cualidades que se salen de los parámetros, entonces la segmentación fallará.

En cambio, las Redes Neuronales se entrenan para establecer de manera automática los límites y características que deben ser segmentados de la imagen, esto se logra utilizando técnicas de extracción y clasificación de características con el objetivo de minimizar la intervención humana en la segmentación.

A continuación, se describirán algunos de los algoritmos de segmentación más reconocidos y utilizados en el campo del procesamiento de imágenes.

2.3.1. Thresholding

Cuando se capturan imágenes en ambientes con iluminación controlada, las técnicas clásicas de segmentación siguen siendo herramientas muy confiables. El Thresholding (umbral) consiste en establecer rangos de valores para los diversos canales de una imagen, con el fin de generar una nueva matriz binaria, donde los valores que quedan dentro del umbral se representan con un 1 y lo que quedan fuera con un 0.

El Thresholding resulta muy útil cuando el área de interés resalta del resto de los elementos en la escena. En la Fig. 2.26 se muestra un ejemplo de una imagen que es binarizada utilizando un umbral entre 10 y 156 para una escala de grises de 8 bits (que va de 0 a 255). La imagen binaria puede ser utilizada para procesar la imagen original, tomando como referencia o máscara los píxeles con valores binarios.

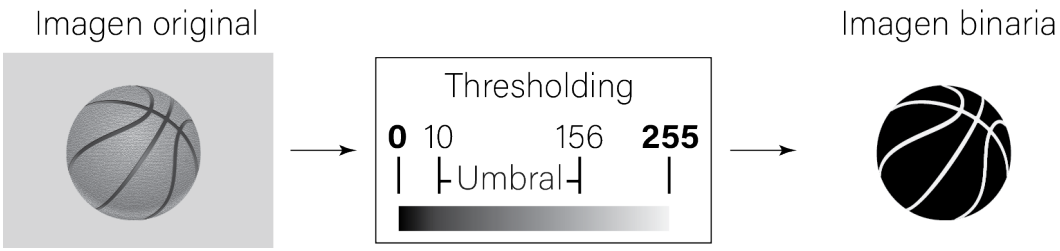


Figura 2.26: Binarización de una imagen con Thresholding.

La binarización también se puede aplicar a cada canal de una imagen RGB para seleccionar regiones que contengan rangos de valores específicos para cada color y hacer extracciones o modificaciones sobre esa región en particular.

2.3.2. Método de Otsu

Enfocado en dividir los píxeles del fondo de los elementos del plano frontal, el Método de Otsu consiste en obtener un valor que divida al histograma intensidades de píxeles en escala de grises de una imagen. De esta forma, se obtiene una imagen binaria como en el Thresholding. Sin embargo, en este caso no se utiliza un umbral establecido de forma manual, sino que el valor que divide al histograma se obtiene de forma automatizada.

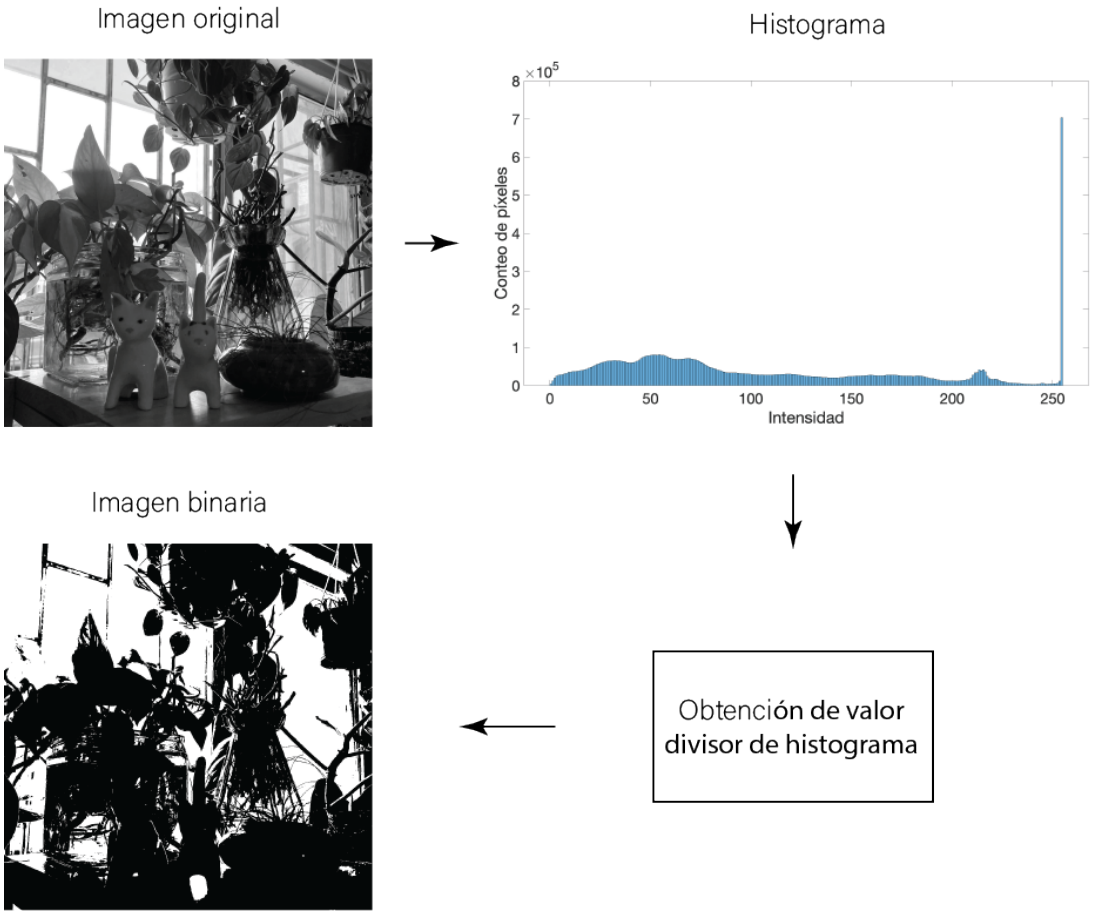


Figura 2.27: Binarización de una imagen con el Método Otsu.

El valor divisor del histograma es el valor máximo de la varianza entre clases (σ_B^2) para cada uno de los valores de intensidad (k) que se tienen en el histograma, como se expresa en la Ecuación 2.15.

$$\sigma_B^2(k) = P_1(k) \cdot P_2(k) \cdot [\mu_1(k) - \mu_2(k)]^2 \tag{2.15}$$

Donde $P1$ y $P2$ son las probabilidades de los dos grupos para cada valor de k que

va entre 0 y 255. En cada iteración, se van comparando el resultado de σ_B^2 con una variable que guarda el máximo obtenido y otra donde se guarda el k correspondiente a ese máximo. Cuando se terminan las iteraciones, el valor divisor corresponde al k donde se obtuvo el σ_B^2 máximo, es decir, donde la varianza entre clases es mayor.

2.3.3. Detector de Canny

El detector de Canny o Canny Edge es un método de segmentación basada en bordes que busca cambios abruptos de intensidad en los píxeles. Este método binariza la imagen, remarcando los bordes con un 1 y al resto de la imagen con un 0. Para lograrlo, el método se divide en cinco pasos:

1. Se convoluciona a la imagen con un filtro Gaussiano, siendo el tamaño del kernel y la desviación estándar del Gaussiano parámetros importantes que afectan a la binarización.

2. Se buscan gradientes de intensidad en la imagen, diferenciando a las tonalidades que corresponden a los bordes y a las que no.

3. Adelgaza los bordes detectados, comparando las intensidades de los píxeles.

4. Se aplica un thresholding con valores empíricos para discriminar a los píxeles que no corresponden a bordes.

5. Hace una comparación final de intensidades, donde los píxeles con intensidades bajas son descartados sin no están conectados a píxeles con intensidades altas.

Finalmente, la imagen binarizada resultante es como la que se muestra en la Figura 2.28.

2.3.4. Redes Neuronales Convolucionales

Los métodos clásicos de segmentación de imágenes son particularmente útiles en ambientes controlados donde no es necesario variar constantemente los parámetros de

Figura 2.28: Binarización de una imagen con el Método Canny Edge. a) Imagen original b) Binarizada

funcionamiento. Además, destacan por sus bajos requerimientos de procesamiento. Sin embargo, cuando se requiere segmentar imágenes con variaciones en las condiciones de iluminación, es preferible optar por algoritmos que se adapten a estos cambios y a las variaciones que pueden sufrir los objetos de interés. En estos casos, se suelen emplear algoritmos de segmentación más complejos basados en redes neuronales. Específicamente, para la segmentación se utilizan frecuentemente las Redes Neuronales Convolucionales (RNC), que buscan y detectan características específicas para las cuales han sido entrenadas, indicando su presencia en la imagen y la región donde se encuentran.

En la actualidad, las RNC a menudo incorporan el uso de conexiones residuales [55], que consisten en sumar características previamente obtenidas a los mapas de características que se generan mediante la convolución. Esto previene la degradación de los mapas de características en redes profundas. El uso de conexiones residuales permite una segmentación más precisa en comparación con métodos que no las utilizan.

Otro elemento que mejoró notablemente el rendimiento de las redes neuronales es la implementación de módulos de atención [56], priorizando la búsqueda de elementos especializados. Estos módulos son utilizados en redes neuronales que cumplen con funciones muy específicas, como la segmentación de imágenes médicas.

Algunos de los modelos de segmentación y detección de objetos que actualmente

son más populares incluyen YOLO (You Only Look Once), SegNet, U-Net, Mask R-CNN y DeepLab. Cada uno de estos modelos puede ser entrenado específicamente para la detección de objetos particulares, y cada uno posee características únicas que los hacen adecuados para diversas aplicaciones. A continuación se presenta la lista de estos modelos populares junto con sus principales cualidades.

- YOLO: Destaca por su alta velocidad y eficiencia en detección y segmentación de objetos. Suele ser implementado en aplicaciones en tiempo real.
- SegNet: También es un modelo que puede ser utilizado en tiempo real, sin embargo, sacrifica más precisión que YOLO para obtener una alta velocidad de procesamiento.
- U-Net: Es un modelo con mayor precisión que YOLO y SegNet. Es ampliamente utilizado para segmentación de imágenes médicas pero su uso se ha ido ampliando en otras aplicaciones.
- Mask R-CNN: Es un modelo de alta precisión en la segmentación, siendo incluso capaz de segmentar instancias de los objetos ya identificados.
- DeepLab: Es muy útil en la segmentación de elementos con dimensiones muy variadas. Normalmente es utilizado en condiciones urbanas y en reconocimiento de escenas.

La selección de la red neuronal adecuada usualmente recae en la necesidad de alta precisión o alta velocidad de procesamiento. Esto lo definirá la aplicación en la que serán utilizadas.

Metodología

Este proyecto está enfocado en el desarrollo de algoritmos y ecuaciones que permitan el funcionamiento colaborativo de los sistemas de Visión Estereoscópica y del LS. Esta colaboración brinda información rectificada y sincronizada para ejecutar acciones con incertidumbre minimizada. Particularmente, al fusionar dos nubes de puntos con estos sistemas se obtiene una matriz con mayor información espacial de los objetos dentro del campo de visión. Además, cada sistema es capaz de monitorear otros aspectos de la escena además de obtener información tridimensional, por ejemplo, las cámaras pueden clasificar objetos y segmentarlos. El LS puede clasificar superficies por su color. Cada una de estas capacidades genera una nueva capa de información que puede ser relevante en la toma de decisiones basado en la fusión de estas capas de información.

La metodología para la combinación de información del LS y la visión estereoscópica puede utilizarse en múltiples aplicaciones, como la Localización y Mapeo Simultáneo (SLAM), la fotogrametría y la identificación de colores. En esta tesis se explorará la clasificación del verdor de las hojas de las plantas como una prueba práctica de la compatibilidad entre los dos sistemas, los cuales aportan múltiples capas de información necesarias para el análisis automatizado del verdor de las hojas.

Esta metodología se divide en seis pasos, que se describirán en detalle a continuación. El orden de la metodología es una secuencia organizada que va desde la identificación de la hoja hasta el análisis del verdor.

3.1. Identificación de hojas

Con el fin de evaluar la salud de las plantas a partir del verdor de sus hojas, la primera tarea para lograr la automatización implica identificar y segmentar las hojas del resto de los elementos de la escena. Para esto, es necesario utilizar imágenes captadas con cámaras, ya que este sistema de visión permite recopilar información de todo lo que se encuentra dentro de su campo de visión en cada captura.

En este primer paso, se propone el uso de cámaras del sistema de Visión Estereoscópica y la implementación del modelo Leaf Segmentation U-Net. Como su nombre lo indica, este modelo está basado en la arquitectura U-Net y ha sido entrenado con el Plant Pathology 2021 Challenge y el DenseLeaves Dataset [57, 58], dos bases de datos ampliamente utilizadas en el entrenamiento de redes neuronales para la identificación y segmentación de hojas.

Como resultado de la segmentación, se generan matrices binarias que indican en qué regiones de la imagen capturada se encuentran las hojas de las plantas. Esta información será fundamental para combinarla con la estimación de profundidades proporcionada por el sistema de visión estereoscópica, con el fin de determinar la ubicación espacial de los objetos de interés. Como muestra del funcionamiento de Leaf Segmentation U-Net, se procesó una imagen que contiene una planta Dracaena Sanderiana Gold, en la Fig. 3.1-a se muestra la imagen original y en la Fig. 3.1-b el resultado de la segmentación.

Es importante mencionar que la imagen procesada en la Fig. 3.1-a no fue capturada con el sistema de Visión Estereoscópica, solo se utilizó una imagen de la planta para probar el funcionamiento de la red neuronal.

3.2. Localización espacial de las hojas

De forma paralela a la segmentación, se hace uso de las imágenes capturadas por el sistema de Visión Estereoscópica con el fin de determinar la localización espacial de las hojas. Para determinar las disparidades, se propone el uso de el método clásico Semi-Global Block Matching (SGBM), ya que al tener el enfoque únicamente sobre

Figura 3.1: Segmentación con Leaf Segmentation U-Net. a) imagen original b) imagen segmentada.

la localización de las hojas, no es necesario obtener una versión refinada del mapa de disparidades para conocer a detalle todo el entorno.

Con el SGBM se puede hacer una primera estimación de las coordenadas espaciales de los elementos del entorno y con los datos de segmentación se seleccionan solo las profundidades que corresponden a las hojas.

La implementación del matching se hará a través de la función **StereoSGBM** de la librería OpenCV en el lenguaje de programación Python. Esta función requiere de el par de imágenes del sistema de Visión Estereoscópica, el tamaño del bloque, las disparidades mínimas y máximas y variables que permiten refinar el mapa de disparidades entre las que se encuentran el PreFilterCap, UniquenessRatio, speckleRange, speckleWindowSize. Estas ultimas variables pueden ser o no utilizadas de acuerdo al criterio del programador. Para este proyecto, se programó una interfaz gráfica que muestra cómo se ve afectado el mapa de disparidades con los cambios las variables, este programa es llamado Stereo Tunner y su interfaz se muestra en la Fig. 3.2 y el código en el Apéndice C.

Como resultado, el Stereo Tunner permite ajustar todos los parámetros del matching mostrados en la Fig. 3.2, guardarlos en un archivo para su posterior uso y utilizarlos cada que se requiera procesar una imagen.

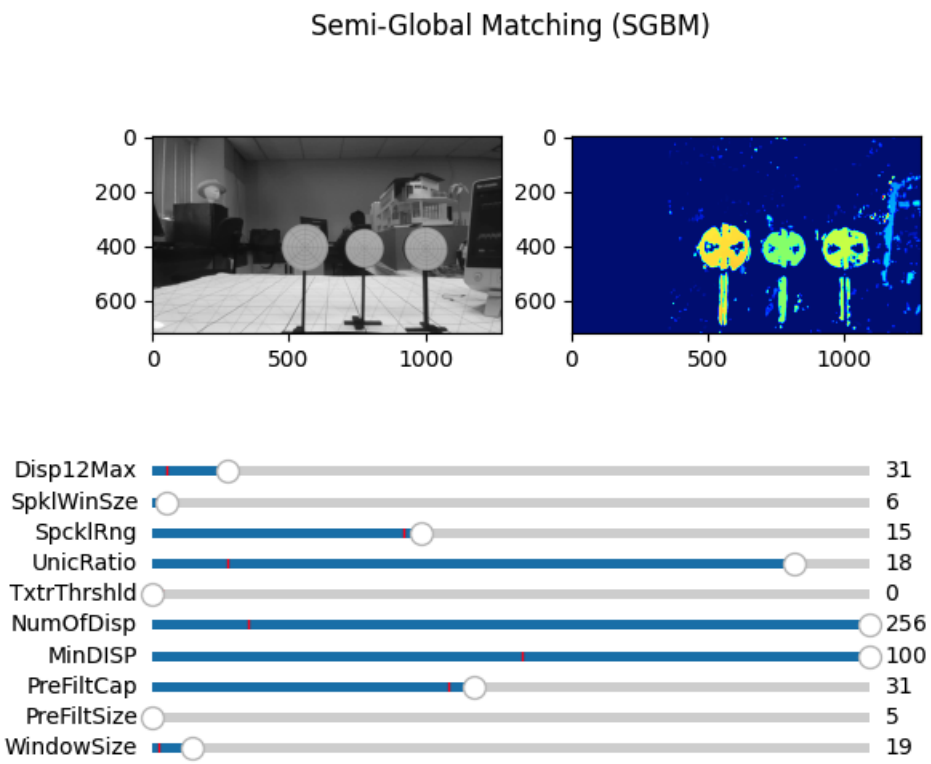


Figura 3.2: Stereo Tunner: Interfaz para refinar parámetros del matching con SGBM

Cuando obtienen mapas de disparidad que logran identificar a los elementos deseados, se realiza la triangulación para todo el mapa, utilizando las ecuaciones que se presentaron en la sección 2.2.1. Esto genera una nube de puntos como la de la Fig. 3.3.

3.3. Selección de puntos de barrido

Desde la imagen segmentada, es necesario determinar cuáles son los puntos que serán escaneados. En este paso, se realizará una aleatorización de los posibles puntos de escaneo. Para aumentar la probabilidad de tomar lecturas sobre las hojas, se implementará un post-procesamiento de la imagen. Este consistirá en seleccionar como posibles zonas de escaneo únicamente aquellos píxeles cuyo verdor se encuentre dentro de un umbral preestablecido sobre el canal verde de la imagen. En la Figura 3.4, se muestra cómo el post-procesamiento aumenta las probabilidades de seleccionar una hoja de la planta para el análisis posterior.

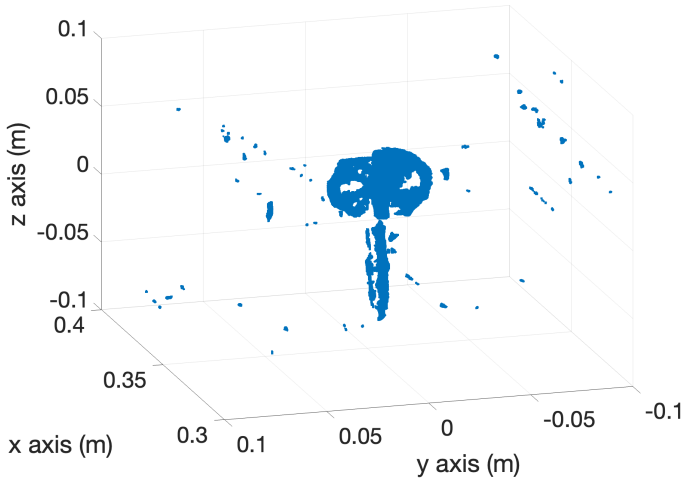


Figura 3.3: Nube de puntos de un objeto de pruebas, obtenida con el sistema de Visión Estereoscópica.

Figura 3.4: Post-procesamiento de imagen segmentada con dos distintos valores de umbral sobre el canal verde. a) Segmentación inicial b) Umbral de 180 c) Umbral de 200

En la matriz binaria de la imagen segmentada se realiza una aleatorización de sus índices. Se establece un condicional en el que, si el valor correspondiente a los índices seleccionados es diferente de cero (es decir, no es fondo), se guardan los índices en una nueva matriz. Este proceso se repite en función de la cantidad de puntos que se consideran representativos del estado de salud de la planta.

Para la aletorización, sea Ω el conjunto de todos los píxeles de la imagen segmentada I de dimensiones $m \times n$, donde cada píxel $\omega \in \Omega$ tiene una posición (i, j) y un valor asociado $I(i, j)$, se define el espacio de eventos E como el conjunto de todos los píxeles

en Ω cuyos valores son distintos a 0, es decir, no pertenecen al fondo de la imagen.

$$E = \{\omega \in \Omega | I(\omega) > 0\} \tag{3.1}$$

La selección de un punto en E se describe como una elección aleatoria de un píxel que no pertenece al fondo. Si ω es un píxel elegido aleatoriamente de Ω, la probabilidad de seleccionar un píxel que no sea fondo se expresa en la Eq. 3.2.

$$P(\omega \in E) = \frac{|E|}{|\Omega|} \tag{3.2}$$

Donde $|E|$ es el numero de píxeles que no son fondo y $|\Omega|$ es el numero total de píxeles en la imagen.

Para seleccionar los índices de un píxel, se deben de enumerar todos los píxeles ω en Ω que satisfacen la condición $I(\omega) > 0$. Esto genera una lista filtrada solo con los píxeles que no son fondo y se seleccionan índices (i, j) de forma aleatoria.

Entonces, se considera a X como una variable aleatoria que representa el índice de un píxel de la lista filtrada. La distribución de probabilidad de X denotada como $P(X = r)$, donde r es el índice de un píxel en E, se define uniformemente para todos los píxeles en E. Entonces, para $r \in E$ se tiene la Eq. 3.3.

$$P(X = r) = \frac{1}{|E|} \tag{3.3}$$

En la Fig. 3.5 se puede ver una representación de puntos seleccionados de forma aleatoria para el posterior posicionamiento del láser del LS sobre cada uno.

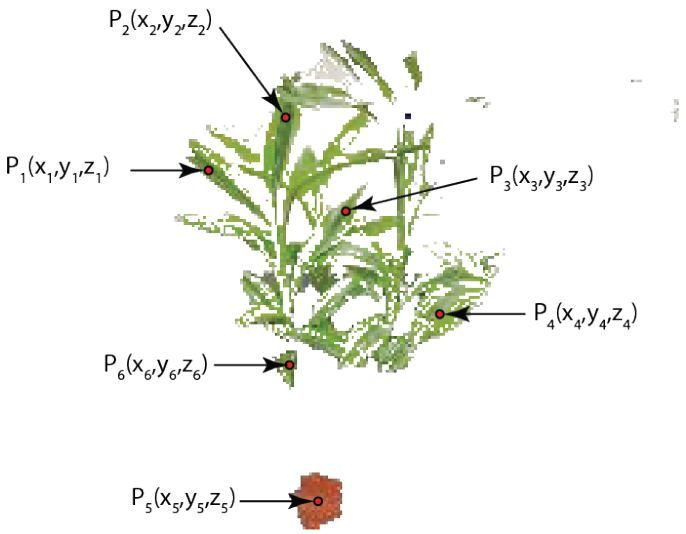


Figura 3.5: Puntos seleccionados de forma aleatoria.

3.4. Posicionamiento láser sobre puntos seleccionados

Los puntos seleccionados indican las posiciones angulares que debe alcanzar el rayo del láser. Es decir, las coordenadas espaciales obtenidas con el sistema de Visión Estereoscópica deben convertirse en posicionamiento angular para el posicionador del LS. Para lograrlo, tanto el LS como el sistema de Visión Estereoscópica deben de compartir el mismo origen, de tal forma que la información adquirida por uno de los dos sistemas debe de ser transformado y rotado al marco de referencia del otro sistema, utilizando las ecuaciones 2.9 y 2.10. Las matrices están formadas por el vector de traslación t entre los origines y por los parámetros de rotación que hay entre ellos. Gráficamente se puede visualizar como en la Fig. 3.6.

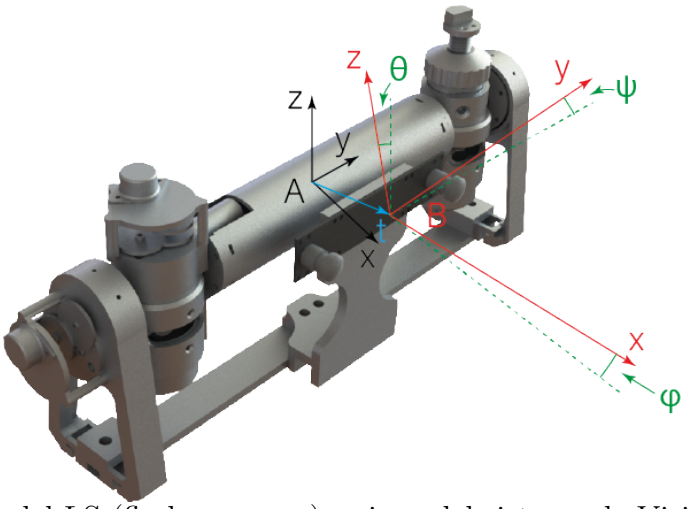
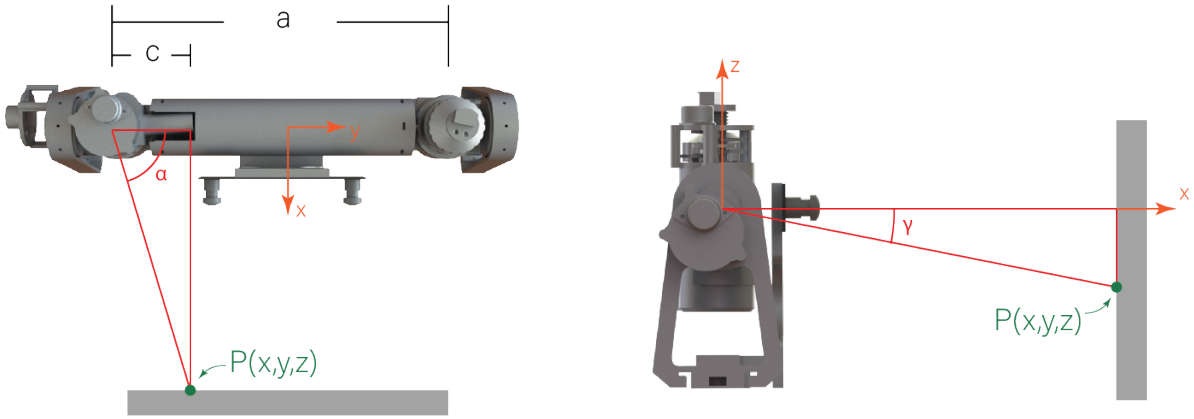


Figura 3.6: Origen del LS (flechas negras), origen del sistema de Visión Estereoscópica (flechas rojas), vector de traslación (flecha azul), parámetros de rotación (verde).

En el caso de la fusión de estos dos sistemas, se considera la traslación del origen del sistema de Visión Estereoscópica hacia el origen del LS, ya que a pesar de los movimientos del posicionador láser, el origen del LS permanece en el mismo lugar. Sin embargo, en el caso de la Visión Estereoscópica, cualquier movimiento de las cámaras implica un cambio de posición del origen, haciendo que la ubicación relativa entre los sistemas sea variable, lo cual no es deseable. Los parámetros de rotación y traslación en este proyecto se considerarán de acuerdo a las posiciones físicas de los sistemas en el prototipo.

Entonces, para obtener los ángulos horizontal (α) y vertical (γ) para cada posición tridimensional (i, j) deseada, se considera que el origen de las mediciones del sistema de Visión Estereoscópica coincide con las del LS. De esta forma, las componentes (x, y, z) de cada punto medido y la distancia entre el origen del LS y el centro del posicionador, se convierten en los catetos de los triángulos rectángulos que se forman entre estas tres partes, como se muestra en la Fig. 3.7.



a) b)

Figura 3.7: Componentes de la conversión de coordenadas rectangulares de un punto ángulos de posicionamiento del láser. a) Vista superior b) Vista lateral

Entonces, α y γ se pueden estimar con las ecuaciones 3.4 y 3.5.

$$\alpha = 90 - arctan\left(\frac{c}{x}\right) \tag{3.4}$$

$$\gamma = arctan\left(\frac{z}{x}\right) \tag{3.5}$$

Donde c es la diferencia sobre el eje y de la posición del posicionador láser y la coordenada en y del punto de interés (Eq. 3.6).

$$c = \frac{a}{2} + y \tag{3.6}$$

3.5. Información complementaria

Si se asume que las estimaciones de profundidad del sistema de Visión Estereoscópica son perfectas, entonces sería adecuado proceder directamente al último paso de esta metodología. Sin embargo, existen fenómenos ópticos que generan errores en el matching y la segmentación, los cuales se reflejan como mediciones incorrectas o falta de mediciones en distintas regiones. Por ejemplo, en el caso del matching, estos errores pueden manifestarse como ruido, como se observa en la Fig. 2.18. En la segmentación, la falta de precisión es evidente en la Fig. 3.4. Algunos fenómenos ópticos que generan

estos errores incluyen las reflexiones no Lambertianas, las superficies de color uniforme, los objetos sin texturas y la mala calidad en la fabricación de los componentes ópticos.

Es por esto que se vuelve relevante compensar y complementar las mediciones obtenidas con el sistema de Visión Estereoscópica mediante el uso de otro sistema. En este paso de la metodología, se propone la toma de mediciones adicionales con el LS, con el objetivo de obtener medidas redundantes que sirvan para ajustar el posicionamiento del láser sobre los puntos aleatorios seleccionados de la hoja y también para rellenar áreas donde falta información, lo cual es importante para corroborar la existencia de un objeto o elemento omitido durante la segmentación y el matching.

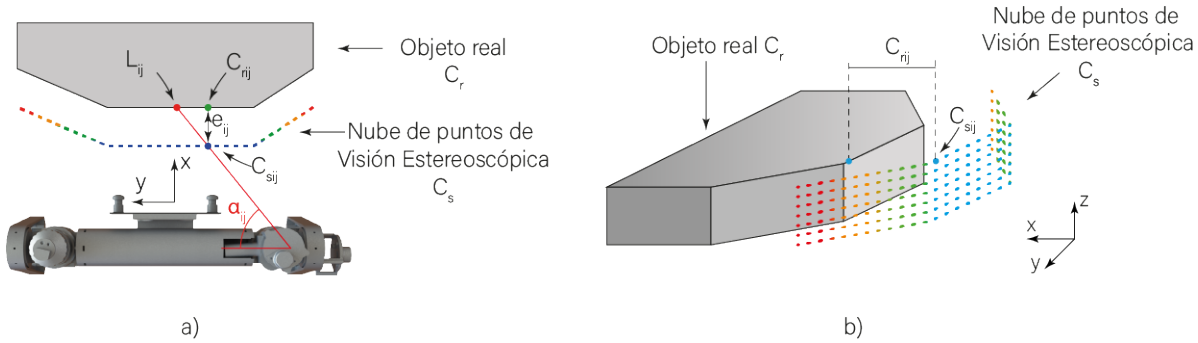


Figura 3.8: Error en posicionamiento láser con retroalimentación del sistema de Visión Estereoscópica. a) vista superior b) vista isométrica.

Comenzando por la falta de precisión en el posicionamiento, cuando se suma el error del matching con la falta de precisión en la estimación de las matrices extrínsecas e intrínsecas de las cámaras, es inevitable que la nube de puntos que se genera al rededor de los objeto no esté desplazada de la posición real del objeto. Entonces se tiene una nube de puntos que representa a la posición ideal del objeto (C_i) y la nube de puntos estimada por el sistema de Visión Estereoscópica (C_s). Ambas nubes tienen un desplazamiento variable para todos los objetos dentro del campo de visión del sistema de Visión Estereoscopica, por lo que, al efectuar el procedimiento de posicionamiento sobre un punto deseado como se indicó en la sección 3.4, la ubicación espacial de la posición del láser no coincidiría con el punto seleccionado. Esto se puede visualizar en la Fig. 3.8, donde se aprecia el punto real ($C_{r_{ij}}$) donde se deseaba colocar el punto del láser (L_{ij}) y el punto correspondiente estimado con Visión Estereoscópica ($C_{s_{ij}}$). La diferencia entre cada par de puntos $C_{r_{ij}}$ y $C_{s_{ij}}$ está asociado al error en la medición e_{ij}.

Debido a que la deformación en la estimación de profundidades en el campo de visión del sistema de Visión Estereoscópica no es uniforme, el error asociado a cada par de mediciones e_{ij} se vuelve variable. Por lo que un solo valor de compensación no es suficiente para corregir este desplazamiento. Es en este caso donde es útil tomar medidas con el LS para corregir valores locales.

Para el proceso de corrección, lo primero que se busca es que el LS mueva el láser de forma autónoma hacia el punto que inicialmente se consideraba deseado $C_{r_{ij}}$. Bajo esta idea, se estima la distancia entre las coordenadas y y z entre la posición medida con el LS L_{ij} y las del sistema de Visión Estereoscópica $C_{s_{ij}}$, definidos como Δx y Δz. Sabiendo que los pasos discretos del motor a pasos y los definidos ángulos de visión de los píxeles de las cámaras difícilmente van a coincidir, se establece una región de tolerancia (p_t) donde se considera que L_{ij} está lo suficientemente cerca de $C_{s_{ij}}$. Entonces, se busca posicionar el laser del LS sobre estas coordenadas con un acercamiento por pasos S_y y S_z, propuestos como la relación entre Δx y Δz sobre p_t, siendo p_t un valor definido por el usuario. Se debe considera que p_t debe de ser mayor a cero y que, entre mayor sea su valor, los pasos de acercamiento serán menores, representando un acercamiento lento a $C_{s_{ij}}$.

$$S_y = \frac{\Delta y}{p_t} \tag{3.7}$$

$$S_z = \frac{\Delta z}{p_t} \tag{3.8}$$

Los parámetros antes mencionados se muestran en la Fig. 3.9, donde se visualizan la región de tolerancia (en verde) dependiente de p_t y los avances S_y y S_z correspondientes.

Cuando L_{ij} entra en la zona de tolerancia, el LS obtiene una ultima medida que difícilmente coincidirá con $C_{s_{ij}}$, por lo que se puede establecer una búsqueda del punto más cercano y generar un vector entre L_{ij} y $C_{s_{ij}}$ para efectuar una traslación directa de la nube de puntos del sistema de Visión Estereoscópica hacia el punto medido por el LS. En teoría, esta solución parece simple y directa, sin embargo, en la práctica habrá ocasiones en las que el punto más cercano no sea el representativo del área en la

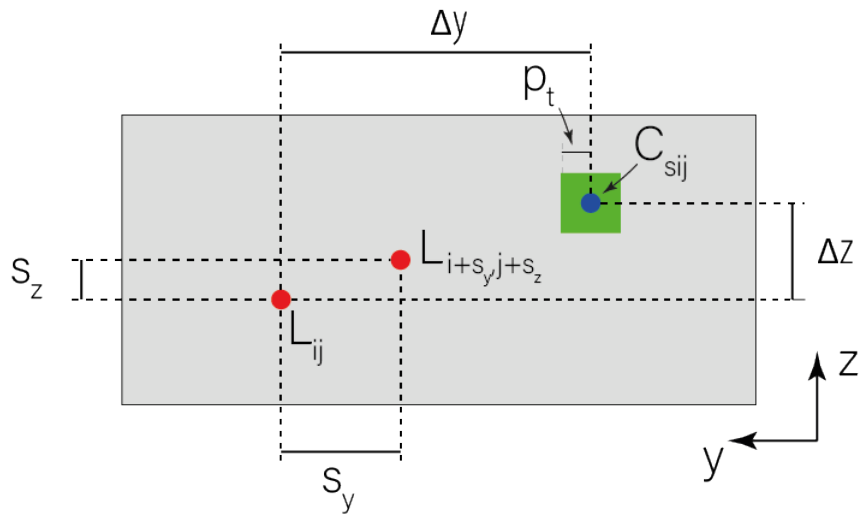


Figura 3.9: Parámetros del reposicionamiento con datos propios del LS.

que está escaneando el LS, debido a que la nube de puntos obtenida con el sistema de Visión Estereoscópica usualmente presenta zonas sin información, o con datos dispersos, provocando un posible desplazamiento de la nube de puntos a otra zona no apegada a la realidad. Es bajo esta situación que se propone que en lugar de solo tomar un punto con el LS, se realice todo un barrido sobre la región de interés, estableciendo una variable que define la dimensión del área por barrer a_t y la densidad lineal de puntos a_d que establece la cantidad de lecturas que se van a tomar. Si se considera al ultimo punto medido L_{ij} como el centro de a_t, entonces se pueden definir las coordenadas angulares de inicio y fin de barrido (α_i, γ_i) y (α_f, γ_f), respectivamente. De esta forma, el área por barrer corresponde a la de un rectángulo esférico, cuya magnitud está dada por la Eq. 3.9.

$$a_t = x^2 \cdot (\alpha_f - \alpha_i) \cdot sen(\gamma_f - \gamma_i) \tag{3.9}$$

Si lo que se busca es obtener los ángulos que definen al área correspondiente, entonces se puede partir de la suposición de que lo que se formará es un cuadrado cuya distancia de centro a borde corresponde a la Eq. 3.10.

$$d_b = \frac{\sqrt{a_t}}{2} \tag{3.10}$$

Así, las coordenadas rectangulares de dos esquinas opuestas del cuadrado están dadas por la suma de las coordenadas (y, z) del punto central con d_b y la resta para el segundo punto y a partir de estas coordenadas. Posteriormente, se hace la conversión a posiciones angulares como se expresó en las Eq. 3.4 y 3.5.

La densidad lineal de puntos a_d es una relación entre cantidad de puntos y una unidad de longitud lineal (puntos/metro) y ayuda a calcular la cantidad de puntos (η) entre pares de ángulos verticales y horizontales (Eq. 3.11).

$$\eta = \frac{a_d}{\sqrt{a_t}} \tag{3.11}$$

De esta forma, el desplazamiento angular entre puntos está dado por las Eq. 3.12 y 3.13.

$$\Delta\alpha = \frac{\alpha_f - \alpha_i}{\eta} \tag{3.12}$$

$$\Delta\gamma = \frac{\gamma - \gamma}{\eta} \tag{3.13}$$

Bajo la consideración de utilizar un área totalmente cuadrada se tiene que $\Delta\alpha = \Delta\gamma$.

Todos estos parámetros ayudan a formar la secuencia de barrido, donde el LS escanea punto a punto hasta obtener una nube de puntos. Este patrón y los parámetros mencionados se pueden visualizar en la Fig. 3.10.

Una vez que el LS toma todas las lecturas y se forma la nube de puntos, se deberá fusionar con la nube de puntos obtenida con el sistema de Visión Estereoscopica. Para esto, se podria darle prioridad a cualquiera de los dos sistemas y arrastrar una nube de puntos hacia la otra. Sin embargo, ambas nubes de puntos presentan dispersión en sus datos, por lo que en esta consideración se propone un desplazamiento proporcional para cada nube de puntos, donde la magnitud del vector de traslación dependa de un coeficiente de traslación (ε) que funciona como peso. Así que, podemos hacer esto calculando los centroides de ambas matrices, C_s y C_t. El centroide de C_s está denotado por P, mientras que el centroide de C_t está denotado por O.

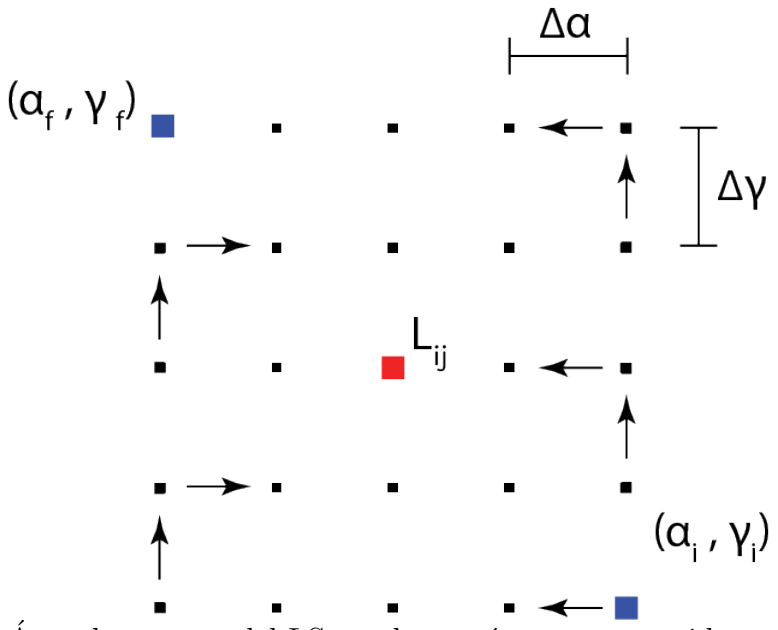


Figura 3.10: Área de escaneo del LS con los parámetros requeridos para definirla.

$$P = \begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix}^T \tag{3.14}$$

$$O = \begin{bmatrix} o_1 & o_2 & o_3 \end{bmatrix}^T$$

Donde $P \epsilon \mathbb{R}^3$ y $O \epsilon \mathbb{R}^3$ y el vector distancia, tomando al centroide como referencia, es la diferencia entre P y O.

$$\vec{OP} = P - O = \begin{bmatrix} p_1 - o_1 \\ p_2 - o_2 \\ p_3 - o_3 \end{bmatrix} \tag{3.15}$$

$$\vec{PO} = O - P = \begin{bmatrix} o_1 - p_1 \\ o_2 - p_3 \\ o_3 - p_3 \end{bmatrix}$$

Y su módulo está representado como se muestra en la Eq. 3.16.

$$|\vec{OP}| = \sqrt{\sum_{i=0}^{3} (p_i - o_i)^2} \tag{3.16}$$

$$|\vec{PO}| = \sqrt{\sum_{i=0}^{3} (o_i - p_i)^2}$$

Para obtener el vector de traslación s necesario conocer cada vector unitario de $\vec{OP}$ y $\vec{PO}$.

$$\vec{u}_{\vec{OP}} = \frac{\vec{OP}}{|\vec{OP}|} \tag{3.17}$$

$$\vec{u}_{\vec{PO}} = \frac{\vec{PO}}{|\vec{PO}|}$$

En este punto se tienen que hacer dos suposiciones. La primera consiste en que los dos vectores unitarios deben de compartir la misma linea de acción.

$$\vec{u}_{\vec{OP}} = -\vec{u}_{\vec{PO}} \tag{3.18}$$

Y el coeficiente de traslación (ε) se agrega para obtener el desplazamiento proporcional.

La segunda suposición consiste en que el coeficiente complementario está dado por la diferencia entre la unidad y ε, como se muestra en la Eq. 3.19.

$$\varepsilon' = 1 - \varepsilon \tag{3.19}$$

Considerando que $\varepsilon \, \epsilon \, [0, 1]$ y que está definido por la media del error absoluto en las mediciones del LS y (ϵ_{LS}) y del sistema de Visión Estereoscópica (ϵ_{stereo}). En este caso, se necesita la distribución de error de cada uno de los sistemas para obtener este parámetro y utilizarlo en la Eq. 3.20.

$$\varepsilon = 1 - \frac{\epsilon_{LS}}{\epsilon_{LS} + \epsilon_{stereo}} \tag{3.20}$$

Entonces, se forman dos vectores $\vec{v_\varepsilon}$ y $\vec{v_{\varepsilon'}}$ que definen la traslación.

$$\vec{v_\varepsilon} = \varepsilon \cdot \vec{OP} = \begin{bmatrix} v_{\epsilon_1} & v_{\epsilon_2} & v_{\epsilon_3} \end{bmatrix}^T \tag{3.21}$$

$$\vec{v_{\varepsilon'}} = \varepsilon' \cdot \vec{PO} = \begin{bmatrix} v_{\epsilon'_1} & v_{\epsilon'_2} & v_{\epsilon'_3} \end{bmatrix}^T$$

Entendiendo que la transformación es un caso particular de la transformación, se utilizan las coordenadas homogéneas para representar al vector de traslación como matriz y expresarlo como una transformación lineal sobre un espacio de orden superior (T_{v_ε} para C_s y $T_{v'_\varepsilon}$ para C_t).

$$T_{v_\varepsilon} = \begin{bmatrix} 1 & 0 & 0 & v_{\varepsilon_1} \\ 0 & 1 & 0 & v_{\varepsilon_2} \\ 0 & 0 & 1 & v_{\varepsilon_3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

$$T_{v_{\varepsilon'}} = \begin{bmatrix} 1 & 0 & 0 & v_{\varepsilon'_1} \\ 0 & 1 & 0 & v_{\varepsilon'_2} \\ 0 & 0 & 1 & v_{\varepsilon'_3} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Consecutivamente, para homogeneizar las matrices caracterizadas por P y O, se utiliza una adicional.

$$P = \begin{bmatrix} p_1 & p_2 & p_3 & 1 \end{bmatrix}^T \tag{3.23}$$

$$O = \begin{bmatrix} o_1 & o_2 & o_3 & 1 \end{bmatrix}$$

La representación de la localización de los objetos utilizando coordenadas homogeneas en un un espacio n-dimensional se realiza con las coordenadas de un espacio $n-1$ dimensional, como muestra la modificacion de P y O. En consecuencia, se considera $n \to 4$ debido a la tridimensionalidad de nuestra tarea.

Finalmente, P y O se transforman con T_{v_ε} y $T_{v_{\varepsilon'}}$.

$$P' = T_{v_\varepsilon} P = \begin{bmatrix} 1 & 0 & 0 & v_{\varepsilon_1} \\ 0 & 1 & 0 & v_{\varepsilon_2} \\ 0 & 0 & 1 & v_{\varepsilon_3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix} = \begin{bmatrix} p_1 + v_{\varepsilon_1} \\ p_2 + v_{\varepsilon_2} \\ p_3 + v_{\varepsilon_3} \\ 1 \end{bmatrix} \tag{3.24}$$

$$O' = T_{v'_\varepsilon} O = \begin{bmatrix} 1 & 0 & 0 & v_{\varepsilon'_1} \\ 0 & 1 & 0 & v_{\varepsilon'_2} \\ 0 & 0 & 1 & v_{\varepsilon'_3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} o_1 \\ o_2 \\ o_3 \\ 1 \end{bmatrix} = \begin{bmatrix} o_1 + v_{\varepsilon'_1} \\ o_2 + v_{\varepsilon'_2} \\ o_3 + v_{\varepsilon'_3} \\ 1 \end{bmatrix}$$

Los vectores resultantes, P' y O', también podrían ser representados como el desplazamiento de P y O.

$$P' = \begin{bmatrix} p'_1 & p'_2 & p'_3 & 1 \end{bmatrix}^T \tag{3.25}$$

$$O' = \begin{bmatrix} o'_1 & o'_2 & o'_3 & 1 \end{bmatrix}^T$$

Programaticamente, las matrices de transformación T_{v_ε} y $T_{v'_\varepsilon}$ se aplican a cada punto de C_s y C_t, respectivamente, y las dos matrices se unen en una sola matriz resultante.

Este paso de la metodología también puede ser utilizado para rellenar espacio vacíos o con falta de información. Sin embargo, la definición de estos espacios implica la implementación de un método o algoritmo automatizado capaz de distinguir qué espacios son relevantes para ser rellenados y cuáles podrían omitirse.

3.6. Indicador de verdor

El último paso consiste en hacer un escaneo sobre los puntos seleccionados en la sección 3.3, buscando hacer una correlación de la cantidad de clorofila en las hojas con parámetros en la señal adquirida por el fotodiodo que se encuentra en la apertura del TVS. Para lograr la correlación, se obtienen parámetros de la señal cuyos valores sean

únicos cuando se toman mediciones en distintas regiones de la planta. Estos parámetros deben de ser capaces de distinguir señales que corresponden a diversos niveles de verdor en las plantas.

Previo a la obtención de los parámetros, se extrae la porción más informativa (S_{seg}) de la señal del fotodiodo (S_{raw}), es decir, la curva que representa la excitación del sensor debido a la luz reflejada sobre una superficie. Esta porción es extraída utilizando un voltaje de referencia (V_{ref}) que divide a la señal de interés del ruido, como se muestra en la Fig. 3.11.

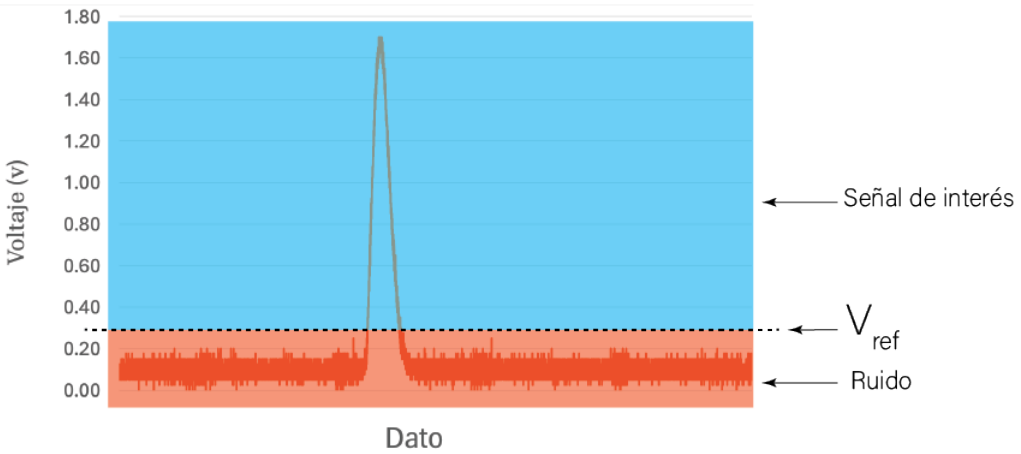


Figura 3.11: División de una curva obtenida en una sola revolución del espejo de la apertura.

Entonces, si s_v es el conjunto de todos los datos de S_{raw} con longitud m, donde cada valor de voltaje v tiene una posición i y un valor asociado $S_{raw}(i)$, se define a S_{seg} como el conjunto de todos los voltajes en s_v mayores que V_{ref} (Eq. 3.26).

$$S_{seg} = \{v \varepsilon s_v | S_{raw}(i) > V_{ref}\} \tag{3.26}$$

Al segmentar la señal, lo que se obtiene es solamente la curva de excitación del fotodiodo, cuya forma es única para cada tipo de superficie donde se refleja el rayo del láser (Fig. 3.12).

Esta porción, posteriormente se normaliza dividiendo a todo el vector de datos sobre el valor pico del mismo vector, como se expresa en la Eq. 3.27.

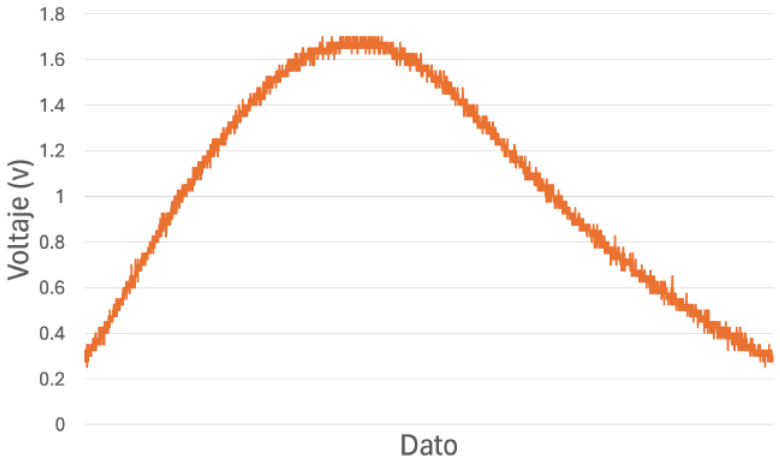


Figura 3.12: Porción informativa segmentada de una señal.

$$S_n = S_{seg}/argmax(S_{seg}) \tag{3.27}$$

Con el fin de distinguir entre señales recibidas al incidir el láser sobre distintas superficies, se obtienen parámetros que permitirán distinguir a una curva de la otra. Estos parámetros tanto estadísticos como típicos en procesamiento de señales, cambian para cada forma recibida y se describen en la Tabla 3.1.

Posteriormente, se realiza un Análisis de Componentes Principales (PCA), que permite identificar y definir cuáles componentes están más relacionados con la mayor variabilidad en el sistema. Este proceso permite reducir la dimensionalidad de los datos al transformar las variables originales en un nuevo conjunto de variables ortogonales llamadas componentes principales. Estas componentes principales capturan la mayor parte de la varianza presente en los datos originales, facilitando la interpretación y mejorando el rendimiento de los modelos de análisis subsiguientes. A partir de la medición de la variabilidad se pueden seleccionar los parámetros estadísticos que pueden generar un mejor entrenamiento para la segmentación.

Una vez establecidos los parámetros, se toman lecturas de las diferentes clases por segmentar. Se deben de establecer los nombres de las clases y un valor medible de referencia que permita distinguir entre cada clase de forma notoria, tomando como referencia a la Tabla 3.2.

Tabla 3.1: Parámetros para la segmentación.

Parámetro	**Simbologia**	**Descripción**
Relación señal/ruido	S_{tnr}	Establece cuál es la amplitud mínima necesaria para que la curva tenga una forma válida.
Entropía	E_I	Es una medida del desorden en un conjunto de datos. Cuando el valor de entropía supera un umbral, el ruido hace que la señal pierda su forma.
Sesgo	S_k	Cuando una señal externa o el ruido afecta a la forma de la señal, el sesgo es notablemente afectado. Este es un indicador de que la captura de la señal no es correcta.
Media	$\bar{S_{seg}}$	Indica la amplitud promedio de la señal normalizada.
Mediana	$\tilde{S_{seg}}$	Es el valor medio del conjunto de datos ordenado.
Media cuadrática	$S_{seg_{rms}}$	Indica el valor promedio de los datos de la señal elevados al cuadrado y posteriormente se obtiene la raiz.
Kurtosis	S_{seg_k}	Indica el peso de las colas en la señal y la pronunciación del pico.
Potencia	$S_{seg_{pow}}$	La potencia la suma de los módulos de las componentes en el dominio de la frecuencia. Es necesario hacer una transformada de Fourier para obtener este parámetro.
Varianza	σ^2	Es el indicador de la variabilidad de los datos de la señal.
Desviación estándar	σ	Indica la variabilidad de los datos de la señal en volts.
Valor eficaz (RMS)	V_{rms}	Es el cuadratico medio de la señal.
Raíz de la suma de los cuadrados	V_{rss}	Es la suma de los cuadrados de los voltajes de la señal.
Distancia valle-cresta	V_A	Es la diferencia entre el volataje mínimo y el máximo de cada señal.

Tabla 3.2: Formato base de clases e índices de clorofila correspondientes.

Clase	**Verdor**	ChI_a	ChI_b
1	Verdor 1	ChI_{a1}	ChI_{b1}
2	Verdor 2	ChI_{a2}	ChI_{b2}
3	Verdor 3	ChI_{a3}	ChI_{b3}
⋮	⋮	⋮	⋮
n	Verdor n	ChI_{an}	ChI_{bn}

En este proyecto se utilizará un DUALEX Scientific DX15530, cuya estimación de la clorofila se basa en la medición de la luz roja e infrarroja cercana refractada en las hojas de las plantas. El método de medición de este dispositivo consiste en colocar un clip atrapando a una hoja. De las dos superficies que entran en contacto con la hoja, una trabaja como emisor y la otra como receptor de luz. Al pulsar un botón de medición, el emisor enciende una fuente de luz roja ($\lambda = 730nm$) y otra de luz infrarroja ($\lambda = 850nm$) y el receptor mide la intensidad de la luz refractada en ambas longitudes de onda (ρ_{730} y ρ_{850}, respectivamente). Así, el Índice de Clorofila (ChI) se calcula utilizando la Eq. 3.28.

$$ChI = \frac{\rho_{850}}{\rho_{730}} - 1 \tag{3.28}$$

Para la segmentación de clases, se utilizará el método de agrupamiento k-medias, que implica la obtención de un centroide para cada una de las clases. Este proceso se realiza de manera iterativa, calculando las medias de los parámetros relevantes y ajustando la posición del centroide en cada iteración. Una vez obtenidos los centroides, la evaluación de nuevos datos depende de la distancia Euclidiana entre el nuevo dato y los centroides. El nuevo dato se asigna a la clase con la menor distancia.

Para entrenar el modelo de k-medias, será necesario que el conjunto de datos tenga un formato como el de la Tabla 3.3.

Tabla 3.3: Formato de matriz de datos a procesar con k-medias.

No.	$Clase$	Par_1	Par_2	Par_3	Par_4	Par_5	Par_6	$\cdots$	Par_m
1	$Clase_1$	Par_{1_1}	Par_{2_1}	Par_{3_1}	Par_{4_1}	Par_{5_1}	Par_{6_1}	$\cdots$	Par_{m_1}
2	$Clase_2$	Par_{1_2}	Par_{2_2}	Par_{3_2}	Par_{4_2}	Par_{5_2}	Par_{6_2}	$\cdots$	Par_{m_2}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	$Clase_n$	Par_{1_n}	Par_{2_n}	Par_{3_n}	Par_{4_n}	Par_{5_n}	Par_{6_n}	$\cdots$	Par_{m_n}

Donde el color de cada clase C_{clase_i} deberá tener asignado el número de color correspondiente descritos en la primera columna de la Tabla 3.2.

Entonces, para cada punto escaneado se asignará una etiqueta que indique la clasificación del verdor. Este vector será útil para determinar el estado de salud de la planta, comparándolo con valores de referencia para cada planta que se esté analizando.

Experimentación y resultados

4.1. Arreglo experimental e interfaz de control

Para poner a prueba la metodología se combinaron dos diferentes juegos de cámaras y un LS. En el caso del LS, su linea base es de $a = 20\,cm$ desde el centro del posicionador hasta el centro de la apertura. El posicionador cuenta con un láser de infrarrojo cercano con una longitud de onda de $\lambda = 808\,nm$ (medido con sensor de potencia THORLABS S121C) y una potencia de emisión de $10\,mW$. El posicionador también cuenta con dos motores a pasos, uno de 20 pasos por giro para el posicionamiento horizontal con una transmisión de engranes rectos que tiene una relación de 1:256 (0.070312^o por paso) y otro de 200 pasos por giro para el posicionamiento vertical con una transmisión planetaria con una relación de 1:25 (0.071^o por paso). En la apertura está colocado un lente biconvexo con distancia focal de $4\,cm$, un opto-acoplador colocado con un desfase de 95^o con respecto al eje de rotación del LS y un foto-diodo con sensibilidad entre $804\,nm$ y $810\,nm$.

La adquisición y procesamiento de las señales del opto-acoplador y del foto-diodo se hacen con un Teensy 4.1 en su configuración estándar, trabajando a una frecuencia de $600\,MHz$, conectado con comunicación serial ($9600\,bps$) a una interfaz programada en Python. En el Apéndice B se muestra todo el código utilizado en el Teensy para adquisición de datos, procesamiento de señales y posicionamiento de los motores a pasos.

En el caso del primer juego de cámaras, son dos cámaras GL68 con resolución

estándar (SD), separadas a $10.4\,cm$ con un sensor de $3.6mm\,x\,2.4mm$, resolución de $640px\,x\,480px$ conectadas de forma asíncrona por protocolo USB. Las lentes de ambas cámaras cuentan con una distancia focal de $f = 820px$, lo que porporciona un $FoV = 38.0227^o$. La combinación de estas cámaras con el LS se ven como se presenta en la Fig. 4.1.

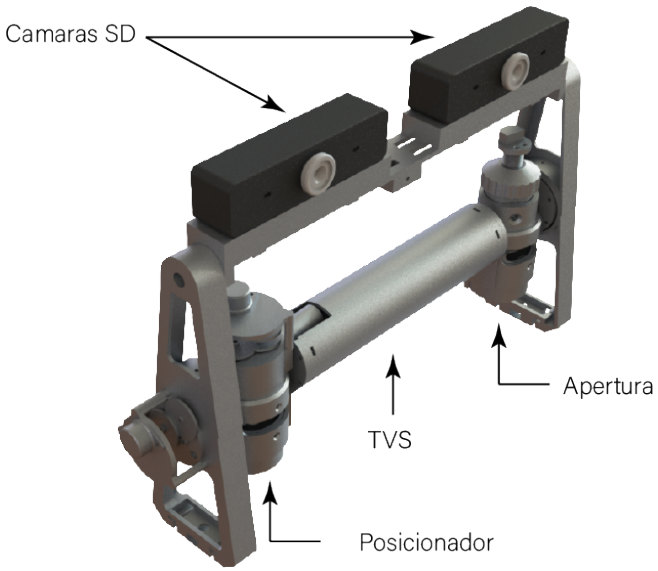


Figura 4.1: Diseño de la configuración LS + Cámaras SD

El segundo juego de cámaras corresponde a un modulo bicámaras GXIVision modelo LSM22100, con módulo de sincronización de captura conectado por protocolo USB, separadas a $8.5\,cm$, tiene un sensor de $5.534mm\,x\,3.113mm$ con resolución de $1280px\,x\,720px$, distancia focal de $980.18px$, que le da un $FoV = 115^o$. Este módulo combinado con el LS se ve como se presenta en la Fig. 4.2.

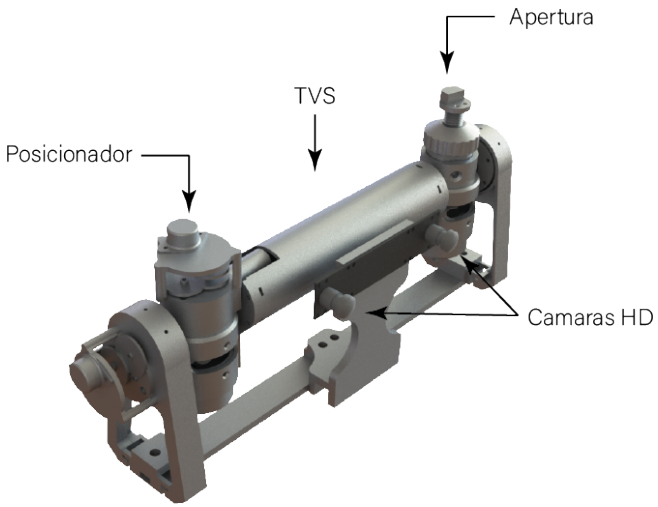


Figura 4.2: Diseño de la configuración LS + Cámaras HD

Ambas combinaciones de LS + Cámaras fueron impresas en 3D utilizando PLA como material base y los prototipos se ven como en la Fig. 4.3.

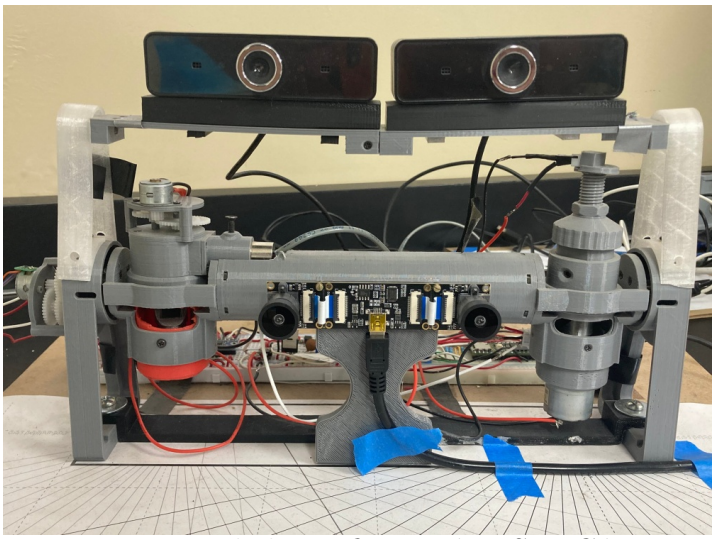


Figura 4.3: Diseño de la configuración LS + Cámaras HD

La interfaz gráfica, denominada "Panel de Control", está programada en Python de forma modular. De tal manera que cumple con múltiples funciones para adquisición y procesamiento de datos e imágenes, calibración de parámetros de funcionamiento del LS, calibración de las cámaras de Visión Estereoscópica y visualización de datos. En la primera pestaña se pueden ver todos los parámetros controlables para el escaneo con el LS, desde ángulos de posicionamiento hasta modos de operación (Fig. 4.4).

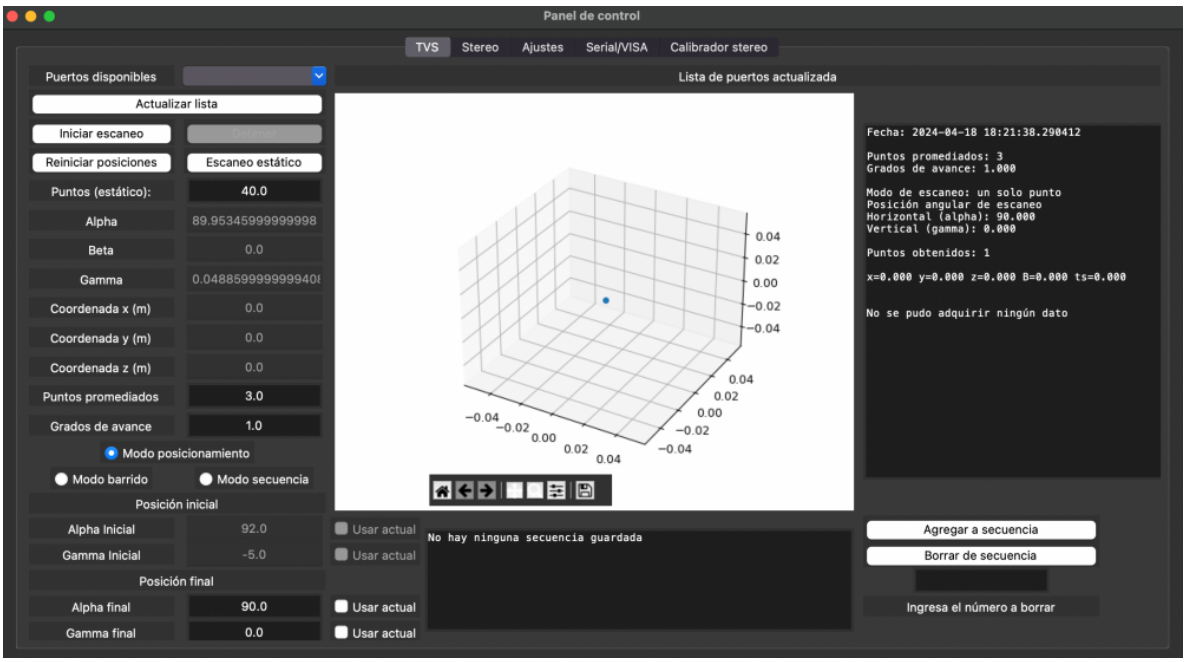


Figura 4.4: Panel de Control. Pestaña de adquisición y procesamiento de datos del LS.

En la segunda pestaña se encuentran todas las opciones que permiten adquirir y procesar imágenes para estimación de coordenadas tridimensionales con el sistema de Visión Estereoscópica. En esta sección se puede seleccionar el algoritmo para matching, modificar sus parámetros, estimar las coordenadas de los elementos en el campo de visión de las cámaras, seleccionar puntos de interés y posicionar el LS sobre ellos y segmentar las imágenes (Fig. 4.5).

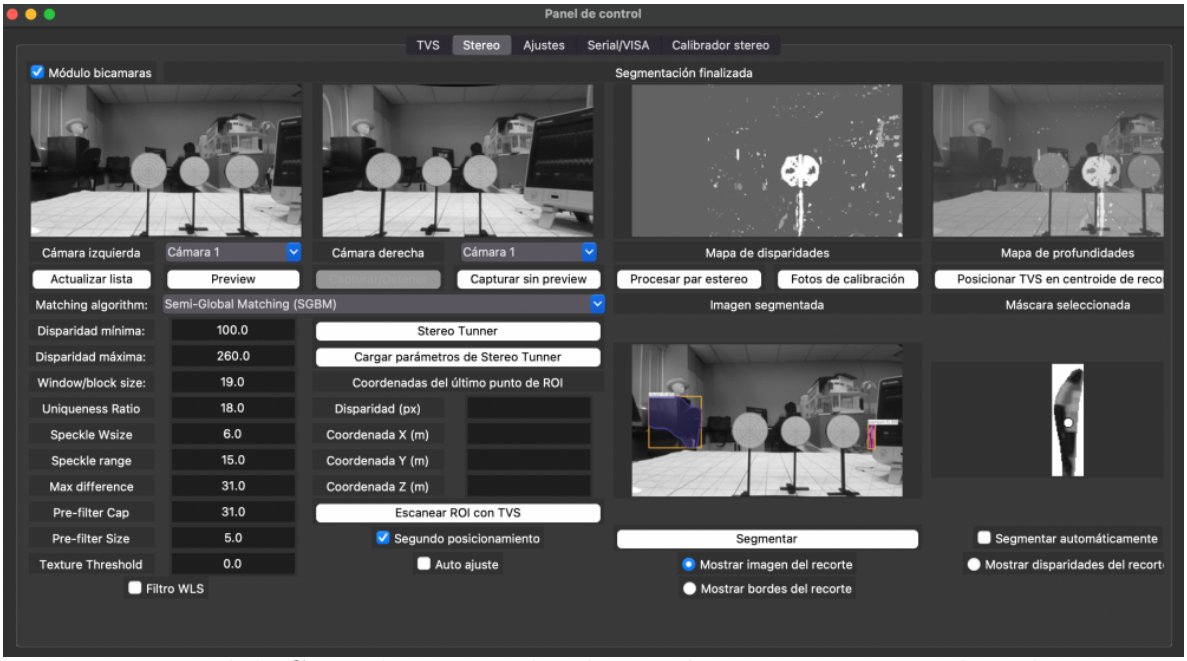


Figura 4.5: Panel de Control. Pestaña de adquisición y procesamiento de imágenes para Visión Estereoscópica.

En la tercera pestaña se encuentran todos los parámetros de ajuste del LS, así como las matrices intrínsecas y extrínsecas de las cámaras (Fig. 4.6). En esta sección, es importante introducir los parámetros con la mayor exactitud posible, ya que de ellos dependerán tanto la precisión como la exactitud de las mediciones de profundidad. Además, una correcta configuración de estos parámetros asegura la coherencia entre las diferentes capas de información proporcionadas por el sistema, lo que mejora la calidad de los datos y, en última instancia, los resultados finales del análisis.

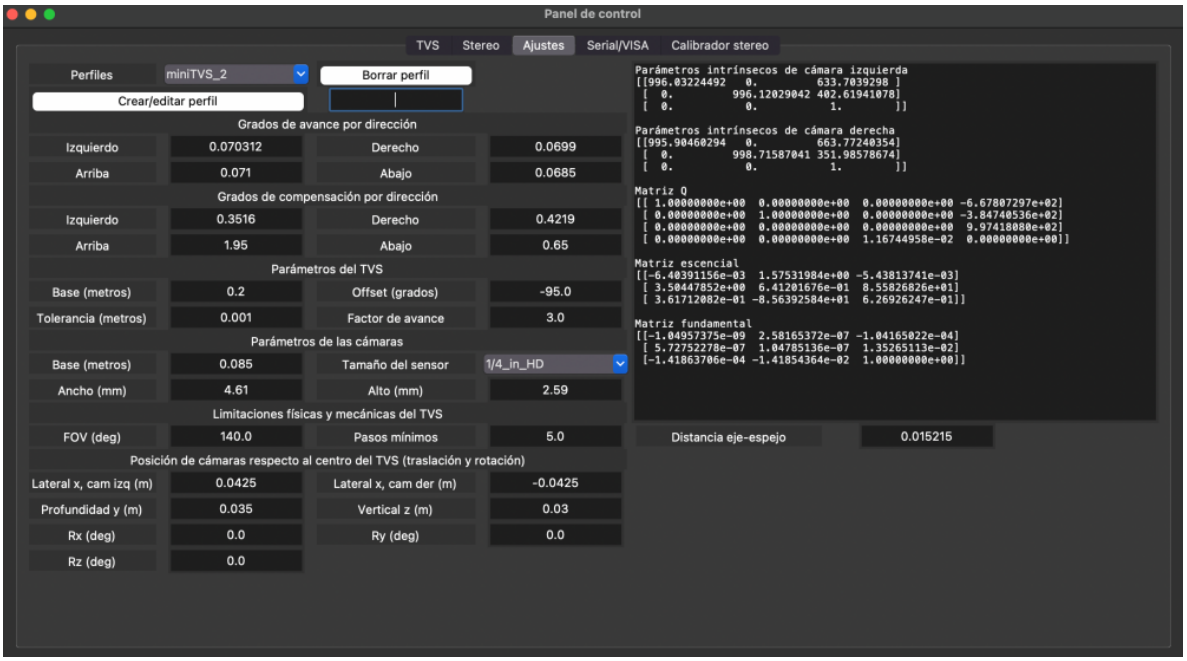


Figura 4.6: Panel de Control. Pestaña de ajustes de parámetros de funcionamiento del LS y las cámaras.

La cuarta pestaña cuenta con utilidades para corroborar el correcto funcionamiento en la adquisición y envío de comandos por comunicación serial. Así como la revisión de la veracidad en la estimación del ángulo de recepción β en el LS, utilizando como referencia un osciloscopio Tektronix TBS2104, por protocolo VISA (Fig. 4.7).

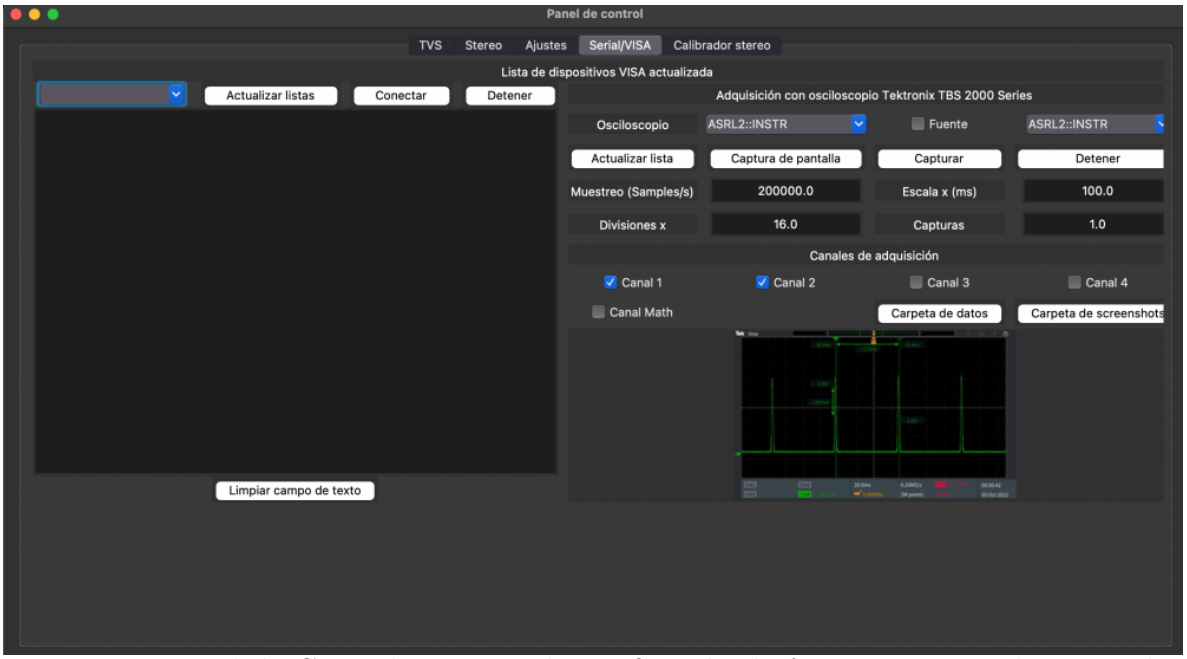


Figura 4.7: Panel de Control. Pestaña de verificación de funcionamiento de comandos y revisión de datos con instrumentación adicional.

Finalmente, la última pestaña es una sección dedicada a la calibración de las cámaras. Aquí se pueden visualizar los pares de imágenes capturadas, establecer las características del patrón de referencia y pre-visualizar los resultados de la calibración del sistema de Visión Estereoscópica (Fig. 4.8).

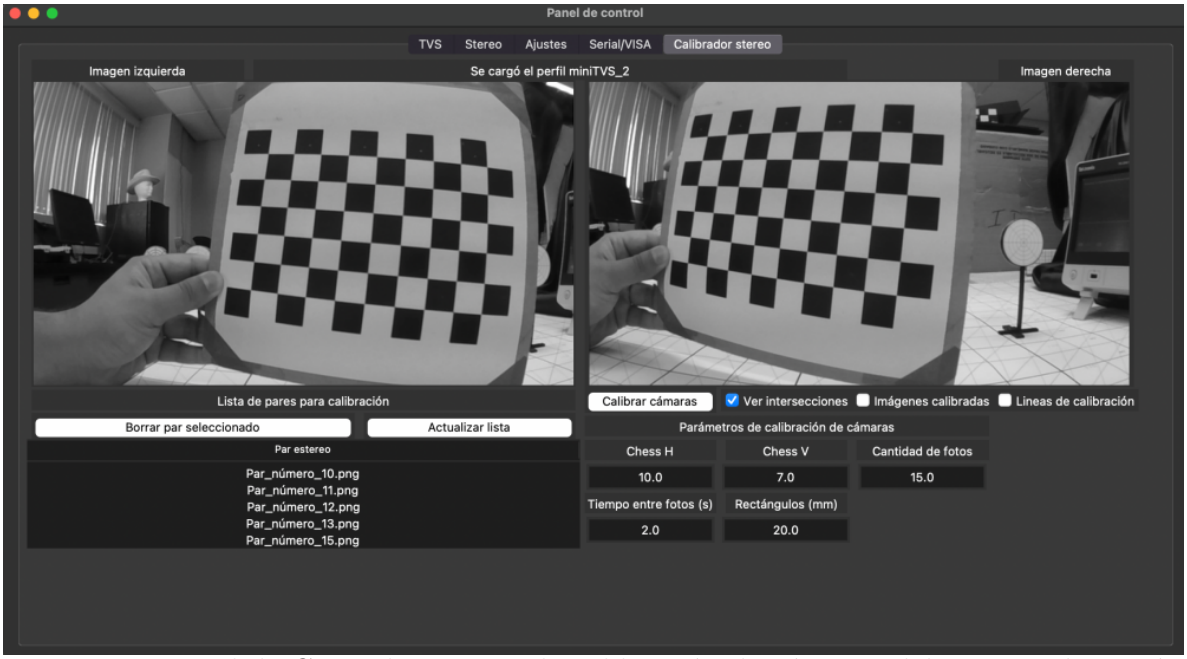


Figura 4.8: Panel de Control. Pestaña de calibración de cámaras del sistema de Visión Estereoscopica.

4.2. Error en estimación de profundidades

Para la caracterización del error en la estimación de coordenadas tridimensionales de cada sistema, se posicionaron cinco diferentes objetos de prueba sobre una mesa con una hoja de calibración. Cada objeto fue posicionado en coordenadas rectangulares distintas, como se muestra en la Fig. 4.9

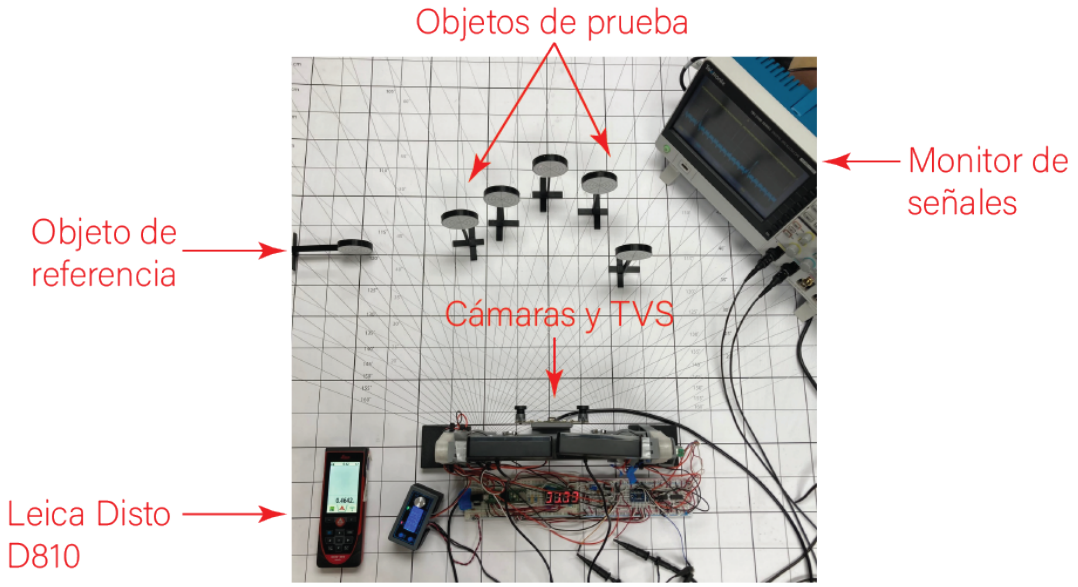


Figura 4.9: Arreglo experimental para las pruebas de posicionamiento láser con retroalimentación del sistema de Visión Estereoscópica.

Las posiciones de los objetos de prueba, están indicadas en la Tabla 4.1. Donde también se aprecia la posición real medida con el Leica Disto D810 y las diferencias entre los valores reales y los medidos por eje y en términos absolutos.

Tabla 4.1: Posiciones rectangulares de objetos de prueba y las mediciones tomadas por el LS y el sistema de Visión Estereoscópica con cámaras SD y HD. Las mediciones indicadas son la media de 300 mediciones independientes.

Sistema	**Objeto**	**Coord. real (cm)**			**Coord. medida (cm)**			**Diferencia (cm)**			**Dif. abs. (cm)**
		x	**y**	**z**	**x**	**y**	**z**	**x**	**y**	**z**	
Cámaras SD	1	48	-3	0.6	50.173	-0.083	2.056	2.173	2.916	1.456	3.918
	2	40	2	0.6	42.350	5.381	1.850	2.350	3.381	1.255	4.305
	3	42	0	0.6	45.577	0.093	2.025	3.570	0.093	1.425	3.852
	4	35	-4	0.6	34.910	-3.189	2.102	0.090	0.810	1.502	1.709
	5	28	2	0.6	32.988	0.226	0.730	4.988	1.773	0.130	5.296
Cámaras HD	1	48	-3	0.6	47.94	-3.850	0.121	0.050	0.85	0.478	0.977
	2	40	2	0.6	41.175	0.572	0.569	1.175	1.427	0.030	1.849
	3	42	0	0.6	42.430	-0.704	0.452	0.430	0.704	0.147	0.838
	4	35	-4	0.6	36.273	-4.185	0.862	1.273	0.185	0.262	1.313
	5	28	2	0.6	30.808	2.377	0.410	2.808	0.377	0.189	2.839
LS	1	48	-3	0.6	47.451	-3.545	0.370	0.548	0.545	0.229	0.806
	2	40	2	0.6	40.210	2.230	0.538	0.210	0.230	0.061	0.318
	3	42	0	0.6	42.930	-0.064	0.685	0.930	0.064	0.085	0.937
	4	35	-4	0.6	35.046	-4.153	0.372	0.046	0.153	0.227	0.278
	5	28	2	0.6	28.008	1.311	0.084	0.008	0.688	0.515	0.859

A partir de estos datos, se estimó el error promedio que presenta cada sistema en la estimación de profundidades, presentado en la Tabla 4.1 como la diferencia absoluta. El sistema de Visión Estereoscópica con cámaras tiene un error absoluto de $\bar{\epsilon_{SD}} = 3.816cm$ para las cámaras SD y $\bar{\epsilon_{HD}} = 1.563cm$ para las cámaras HD. El LS presentó un error de

$\bar{\epsilon_{LS}} = 0.640cm$. A partir de estas medidas en el error se puede generar la distribución de cada sistema, quedando como las que se muestran en la Fig. 4.10, donde es visible que el LS presenta menor error que las cámaras pero mayor dispersión en los datos.

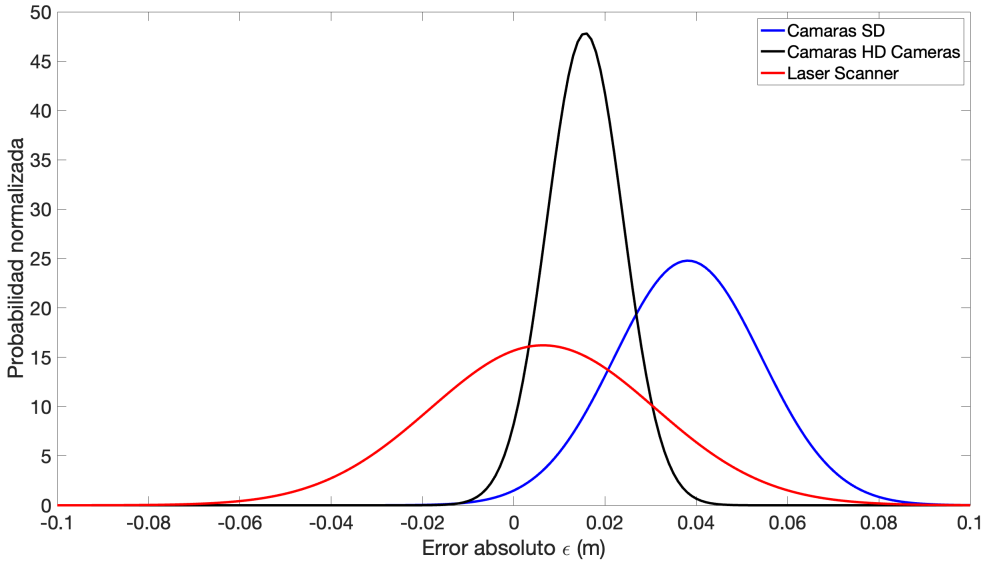


Figura 4.10: Distribución normalizada del error.

Nótese que las tres medias quedan desplazadas hacia el lado positivo del error. Esto se debe a que la medición del error en un espacio tridimensional la distancia absoluta entre la posición ideal de un punto y la real medida por cualquiera de los sistemas.

4.3. Posicionamiento del LS con retroalimentación de Estereo Visión

Una vez que se calcularon los errores en la estimación de coordenadas rectangulares para los tres sistemas, se pueden hacer pruebas sobre el posicionamiento con retroalimentación del sistema de Visión Estereoscópica, donde los errores en la estimación servirán como factores de compensación en el posicionamiento del láser.

Considerando que el origen del LS se utilizará como origen del todo el sistema combinado, para el arreglo con cámaras SD se tiene el vector de traslación $t_{SD}(0m, 0m, 0.125m)$ y los parámetros de rotación $R_{SD}(-7^o, 0^o, 0^o)$. Con las cámaras HD se tiene un vector de traslación $t_{HD}(0.035m, 0m, 0.02m)$ y parámetros de rotación $R_{SD}(0^o, 0^o, 0^o)$. Estos parámetros fueron obtenidos siguiendo el diseño CAD del prototipo.

Entonces, utilizando los mismos objetos de prueba presentados en la medición del error y posicionados en las coordenadas expresadas en la Tabla 4.1, se hizo la estimación de profundidad con cada sistema de Visión Estereoscópica, se seleccionó manualmente el punto representativo del centro del círculo de los objetos, se calcularon las coordenadas angulares correspondientes con las Eq. 3.4 y 3.5, se movió el láser a la posición estimada y se midió la diferencia entre la distancia entre el centro real del objeto y la posición del láser para cada sistema, como se muestra en la segunda columna de la Tabla 4.2. En la segunda columna de la misma tabla, se muestra el error cuando el LS toma medición de la posición actual e intenta reposicionarse siguiendo la metodología presentada en la sección 3.5.

Tabla 4.2: Error de posicionamiento del láser con retroalimentación de las cámaras y autoajuste del LS. El error medido es la media de 300 mediciones independientes.

Sistema	**Error con retroalimentación estéreo (cm)**	**Error con retroalimentación estéreo y autoajuste del LS (cm)**	**Diferencia (cm)**
LS + Cámaras SD	3.051	0.970	2.081
	4.600	1.722	2.877
	4.716	1.002	3.714
	2.365	1.989	0.375
	4.727	0.904	3.824
LS + Cámaras HD	2.357	1.191	0.691
	2.007	1.050	0.550
	2.198	1.895	1.395
	2.834	1.680	1.180
	1.985	1.945	1.445

A pesar de que previamente se calculó el error medio en la estimación de coordenadas rectangulares, se sigue obteniendo un error medio en el posicionamiento con cámaras SD de $\epsilon_{SDP} = 3.892$ cm. Para las cámaras HD, se tiene que $\epsilon_{HDP} = 2.574$ cm. Cuando se utilizan los datos del LS para el reposicionamiento, se logra disminuir el error de forma poco significativa para ambas combinaciones. En la combinación LS + Cámaras SD, el error se disminuye a $\epsilon_{SDTVS} = 1.317$ cm, y para LS + Cámaras HD, a $\epsilon_{HDTVS} = 1.053$ cm. En la Fig. 4.11, se muestra una comparación del error con las combinaciones de los sistemas para los cinco diferentes objetos de prueba.

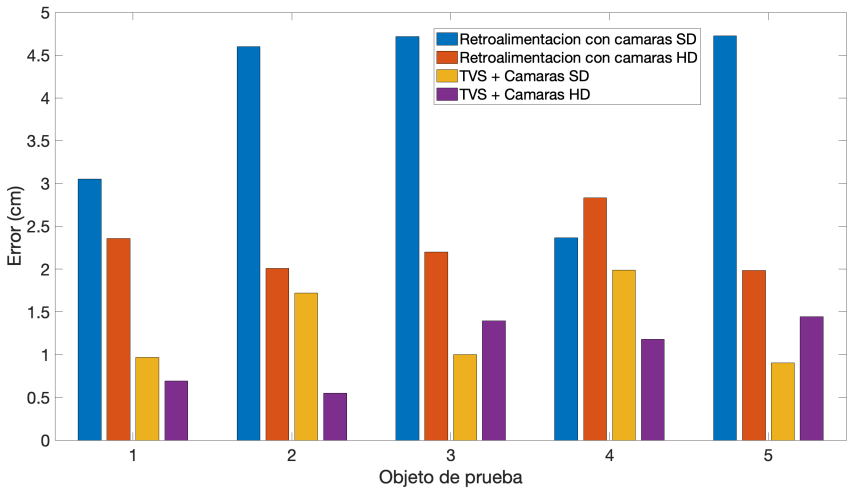


Figura 4.11: Error en el posicionamiento utilizando retroalimentación solo de las cámaras y con autoajuste del LS.

Esto se debe a que el error en la estimación de profundidades no es uniforme en el campo de visión de los tres sistemas, por lo que utilizar un solo valor como factor de ajuste solo puede ser válido para las regiones donde se muestrearon los datos.

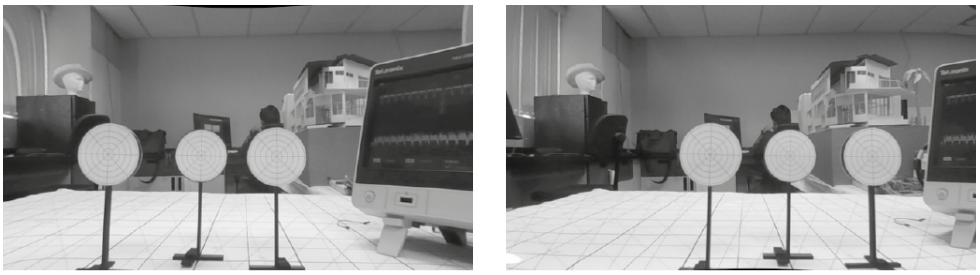
4.4. Fusión de datos tridimensionales

A partir de las estimaciones de los errores para los tres sistemas, se puede determinar que al combinar el LS con cámaras HD se obtiene un error medio de $\varepsilon = 0.7094$, según la Eq. 3.20. Al combinar el LS con cámaras SD, el error medio es de $\varepsilon = 0.8563$.

De tal manera que si se hace un barrido sobre cada uno de los objetos de prueba utilizando el sistema de Visión Estereoscópica y el LS, se puede hacer un ajuste final sobre las matrices resultantes utilizando la metodología presentada en la sección 3.5, utilizando los centroides de ambas nubes de puntos y trasladándolos en proporción a ε.

A partir de esta etapa, se hizo uso solamente de las cámaras HD, ya que no es necesario hacer comparaciones entre juegos de cámaras, si no que se va a evaluar la metodología presentada.

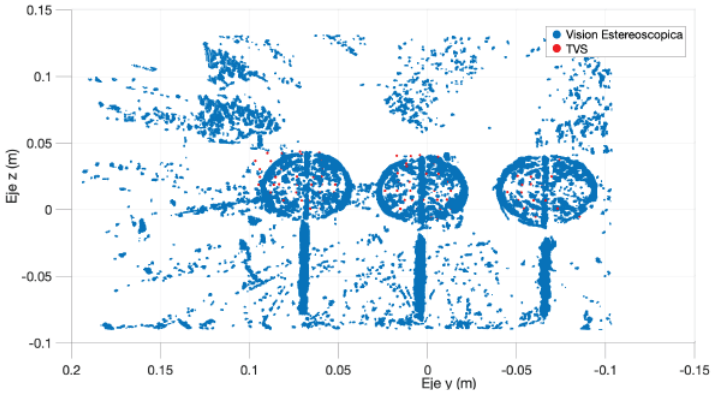
Se colocaron tres objetos de prueba frente a las cámaras cuyos centros se encontraban en las coordenadas Tar_1 =(35 cm, 7.5 cm, 0.6 cm), Tar_2 =(40 cm, 0 cm, 0.6 cm) y Tar_3 =(37.5 cm, -7.5 cm, 0.6 cm). Los tres objetos se encontraban en el campo de visión del LS y de las cámaras, como se muestra en la Fig. 4.12.



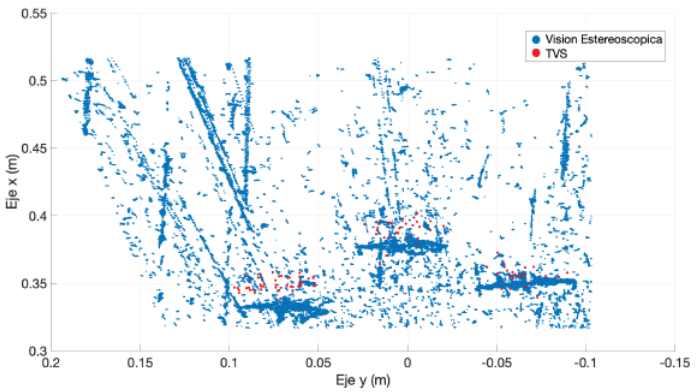
a) b)

Figura 4.12: Objetos de prueba capturados por el sistema de Visión Estereoscópica. a) Imagen izquierda b) Imagen derecha.

La nube de puntos de ambos sistemas se pueden visualizar en la Fig. 4.13, donde se aprecian los puntos obtenidos por el sistema de Visión Estereoscópica en azul y en rojo para el LS. En la Fig. 4.13-b es apreciable la diferencia en la estimación de la profundidad, por lo que es necesaria una corrección.



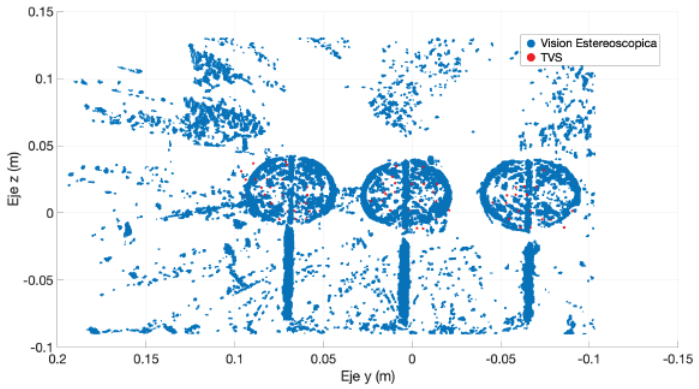
a)



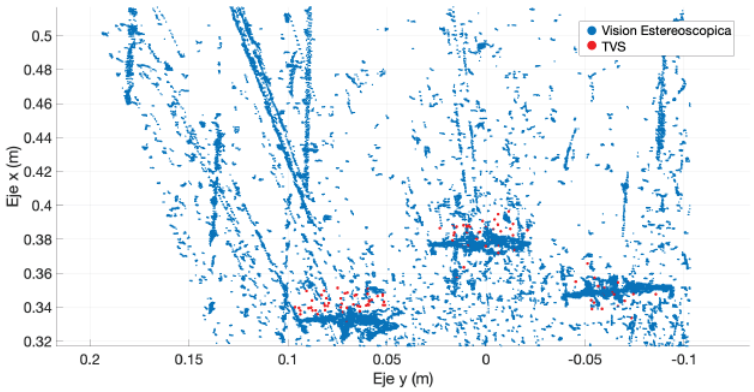
b)

Figura 4.13: Nubes de punto sin corregir de objetos de prueba escaneados con el sistema de Visión Estereoscópica y el LS. a) Vista frontal b) Vista superior.

Considerando el valor de $\varepsilon = 0.7094$, previamente obtenido con la combinación del LS y cámaras HD, se obtuvieron dos vectores de traslación, uno para la nube de puntos del sistema de Visión Estereoscópica $t_{stereo}(17.02mm, 1.844mm, 11.72mm)$ y, para la nube de puntos del LS, $t_{LSTVS}(6.974mm, 0.755m, 4.804mm)$. Resultando en un desplazamiento de ambas nubes de puntos, como se muestra en la Fig. 4.14. Y ya que el desplazamiento entre ambas superficies asegura la unión entre ambas, el error medido es la diferencia entre la posición real del objeto y la posición de la nube de puntos unificada. Dando como resultado un error medio por componente de $\epsilon_x = 0.02259m$, $\epsilon_y = 0.02510251m$ y $\epsilon_z = 0.00823073m$.



a)



b)

Figura 4.14: Nubes de punto corregida de objetos de prueba escaneados con el sistema de Visión Estereoscópica y el LS. a) Vista frontal b) Vista superior.

Estos resultados demuestran la compatibilidad entre la información tridimensional proporcionada por ambos sistemas y como son útiles en la corrección del posicionamiento espacial.

4.5. Clasificación de verdor

4.5.1. Índice de Clorofila como referencia

Después de comprobar el funcionamiento colaborativo del LS con el sistema de Visión Estereoscópica para la estimación de posiciones tridimensionales y el escaneo de objetos, se realizó una clasificación del verdor en dos especímenes vegetales: Dracaena Sanderiana Gold y Maranta Arandinacea. El objetivo de esta clasificación fue demostrar la versatilidad de ambos sistemas y la información adicional que pueden proporcionar, así como su utilidad en la identificación del verdor de las plantas.

Se establecieron cinco distintos verdores como las clases a segmentar, cada clase fue establecida utilizando como referencia el Índice de Clorofila (ChI) de las hojas de las plantas seleccionadas, con el medidor de estado de las plantas DUALEX Scientific DX15530, colocando las hojas de las especies seleccionadas entre el emisor y el receptor de luz del dispositivo, como se muestra en la Fig. 4.15.

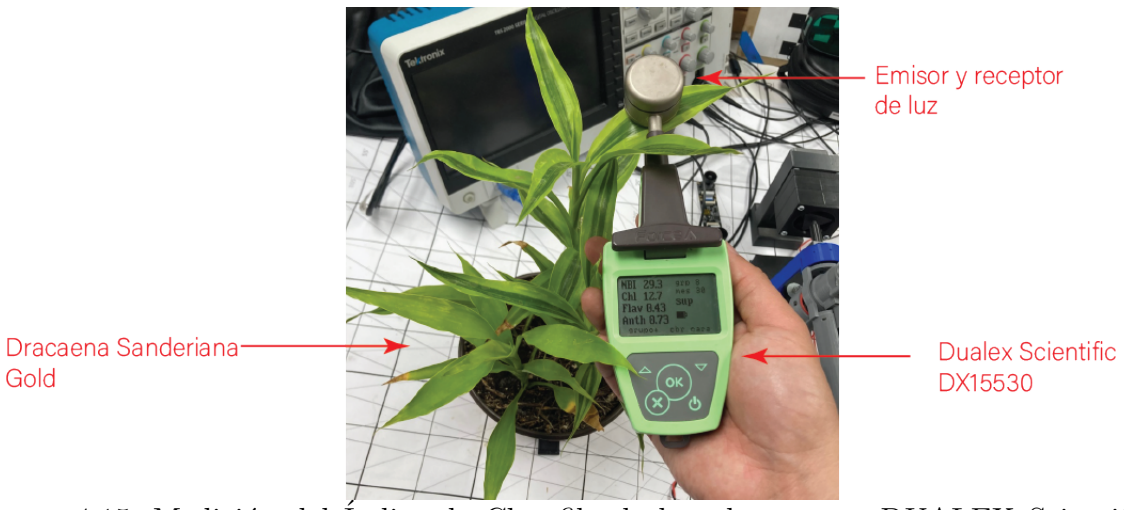
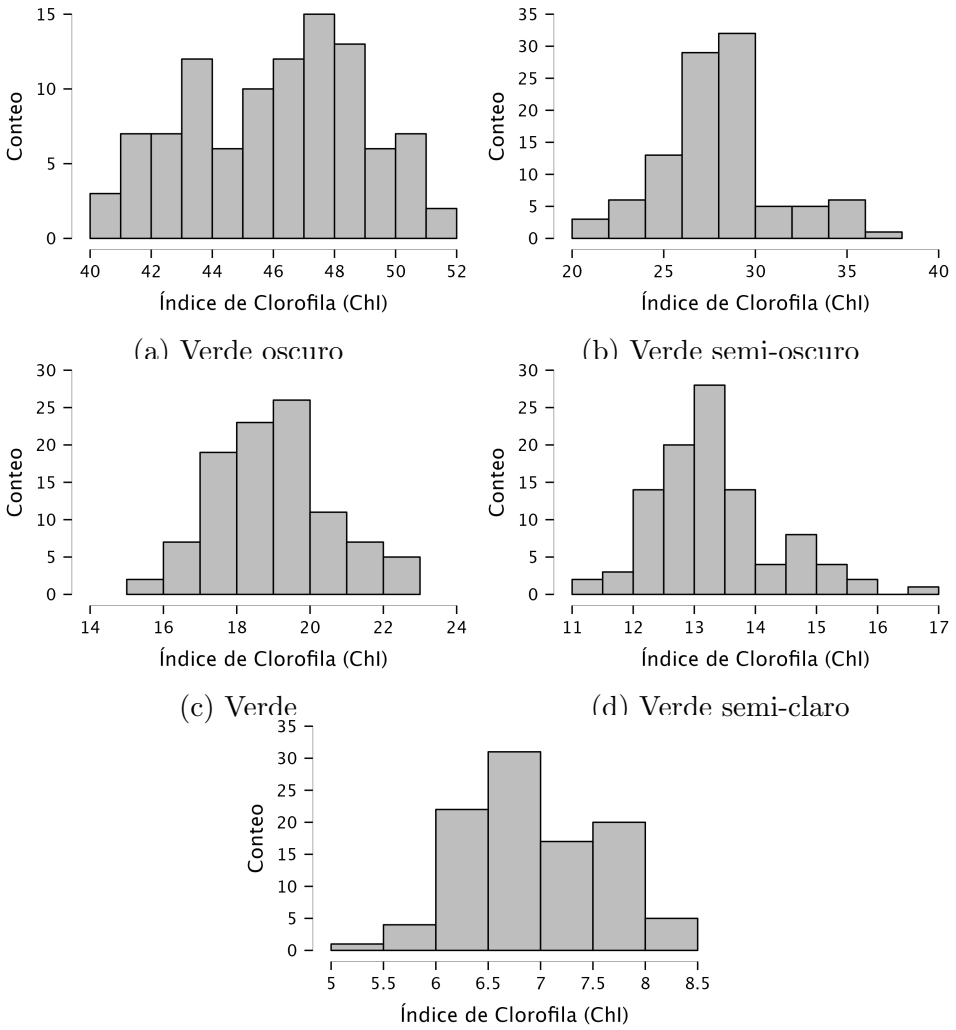


Figura 4.15: Medición del Índice de Clorofila de las plantas con DUALEX Scientific DX15530.

Se hicieron 100 mediciones para cada una de las clases de cada planta, con un total de 1000 mediciones y se obtuvo la distribución de datos para cada clase. Para la Dracaena Sanderiana Gold se obtuvieron las distribuciones de catidades de clorofila de cada clase mostradas en la Fig. 4.16.



(a) Verde oscuro

(b) Verde semi-oscuro

(c) Verde

(d) Verde semi-claro

(e) Verde claro

Figura 4.16: Distribución de Índice de Clorofila para cada clase de Dracaena Sanderiana Gold.

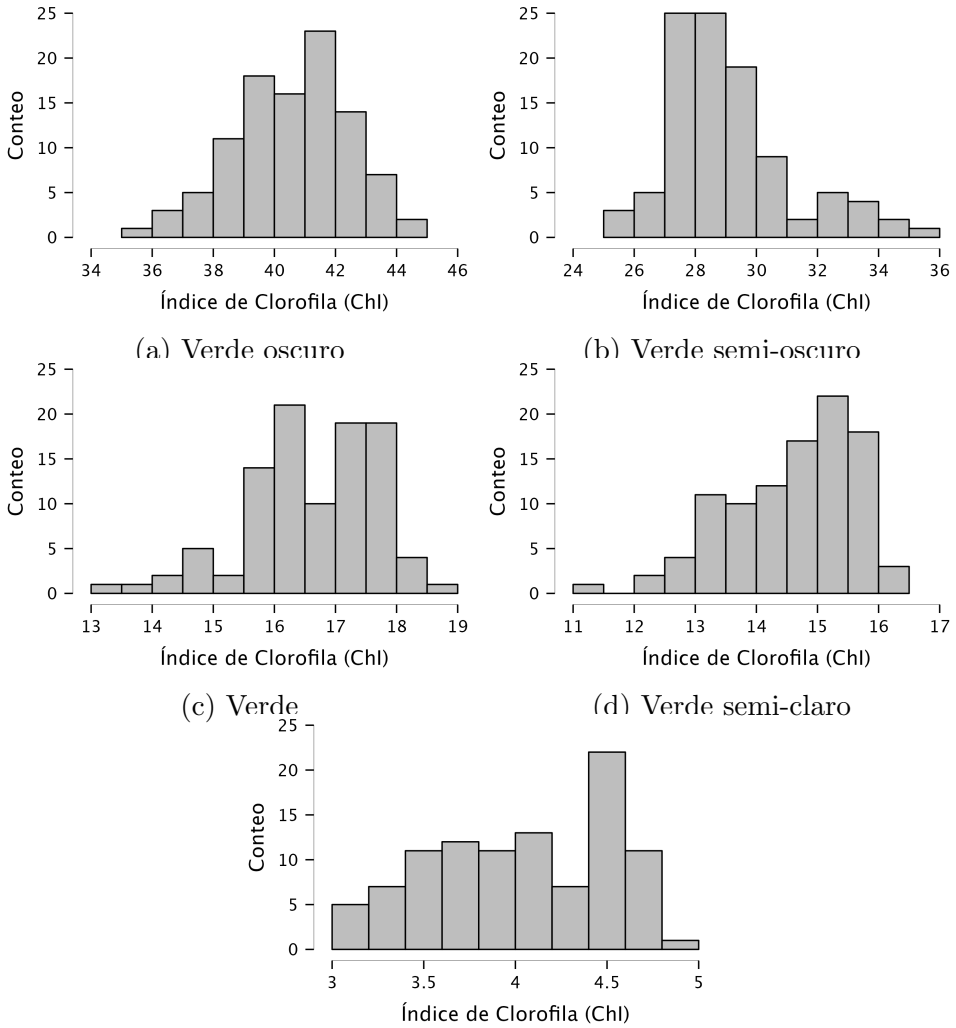
Correspondientemente, los parámetros estadísticos de las distribuciones se muestran en la Tabla 4.3.

Tabla 4.3: Parámetros estadísticos descriptivos de las mediciones de verdor en Dracaena Sanderiana Gold.

Clase	Media	Desv. Estándar	Sesgo	Kurtosis
Verde Oscuro	46.112	2.828	−0.185	−0.958
Verde Semi-Oscuro	27.994	3.153	0.515	0.679
Verde	19.047	1.575	0.194	0.094
Verde Semi-Claro	13.314	0.980	0.720	0.754
Verde Claro	6.947	0.653	0.052	−0.409

En el caso de la Maranta Arandinacea, se obtuvieron las distribuciones de cantidades

de clorofila de cada clase mostradas en la Fig. 4.17.



(a) Verde oscuro

(b) Verde semi-oscuro

(c) Verde

(d) Verde semi-claro

(e) Verde claro

Figura 4.17: Distribución de Índice de Clorofila para cada clase de Maranta Arandinacea.

En este caso, los parámetros estadísticos de las distribuciones se muestran en la Tabla 4.4.

Tabla 4.4: Parámetros estadísticos descriptivos de las mediciones de verdor en Maranta Arandinacea.

Clase	Media	Desv. Estándar	Sesgo	Kurtosis
Verde Oscuro	40.559	1.863	−0.192	−0.387
Verde Semi-Oscuro	29.090	1.973	1.080	1.089
Verde	16.612	1.052	−0.680	0.375
Verde Semi-Claro	14.603	1.017	−0.697	0.375
Verde Claro	4.039	0.487	−0.284	−1.175

A partir de los valores de desviación estándar, sesgo y curtosis, se puede interpretar que los valores medidos tienen una distribución cercana a la normal y que no hay una dispersión significativa respecto a la media. Estos datos son útiles como referencia de la consistencia en la cantidad de clorofila en cada clase para cada planta.

4.5.2. Clasificación del verdor con el LS

Con el fin de analizar el efecto de la amplitud de la señal sobre la clasificación del verdor, se colocó un objeto de prueba en diferentes puntos de una mesa óptica THORLABS MB6090/M. Se direccionó el láser del LS hacia el centro del objeto, se obtuvieron 10 señales de la apertura en cada posición y se calculó la media de la relación señal/ruido en cada posición. Con la relación señal/ruido se hizo un mapa del campo de visión del prototipo del LS utilizado en esta experimentación y la intensidad con la que se recibió la señal. Este mapa se muestra en la Fig. 4.18, con la escala de la relación señal/ruido en decibelios. Fuera del área de color no se recibió suficiente luz reflejada para que fuera visible en la señal del foto-diodo.

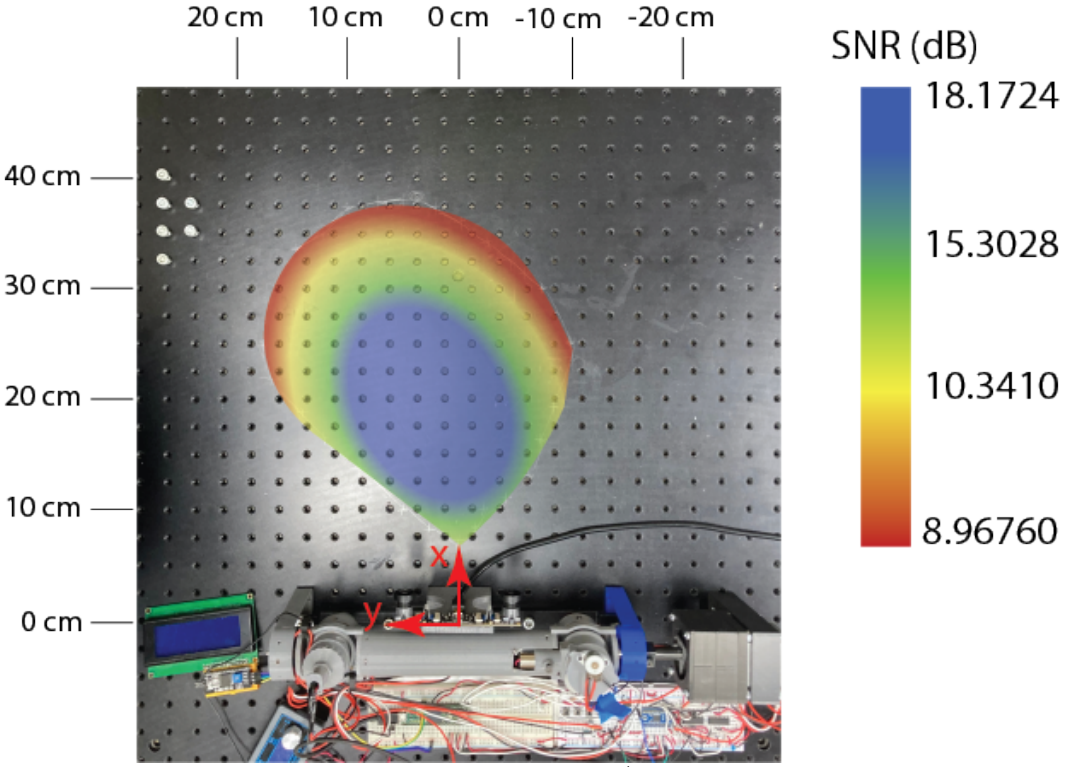


Figura 4.18: Campo de visión con la relación Señal/Ruido correspondiente en decibelios.

Dentro del campo de visión, se establecieron tres posiciones donde se esperan variaciones en la relación señal/ruido, con el fin de determinar el valor límite a partir del cual la señal se deforma lo suficiente como para no ser válida en la clasificación del verdor. Las variaciones en las posiciones se realizaron en el plano xy, aprovechando la precisión de la mesa óptica para lograr posicionamientos exactos. Las coordenadas de posicionamiento están expresadas en la Tabla 4.5.

Tabla 4.5: Posiciones de escaneo de los diferentes niveles de verdor de ambas plantas.

Posición	x (cm)	y (cm)
1	10	3.75
2	15	6.25
3	20	8.75

En cada posición se localizó el láser sobre cada uno de los verdores definidos por las regiones donde se tomaron las mediciones del Índice de Clorofila. Se obtuvieron 500 curvas para cada clase. Generando una base de datos de 15,000 curvas que serán utilizadas para la clasificación.

Se evaluaron los parámetros utilizando el Analisis de Componentes Principales (PCA), comenzando con una prueba χ^2 para corroborar la adecuación del modelo. Se obtuvo un valor de 358931.034 para 53 grados de libertad, lo que resultó en $p < 0.001$. Esto indica que los datos capturados son significativos, permitiendo la realización del PCA.

Tabla 4.6: Cargas de los componentes

	RC1	RC2	Unicidad
Potencia	1.100		0.149
Varianza	1.017		0.066
Mediana	−0.986		0.096
Desviación Estándar	0.919		0.036
Desviación media absoluta	0.915		0.041
Raíz de Suma de Cuadrados	0.904		0.042
Media	−0.895		0.079
Distancia cresta-valle	0.856		0.018
Relación señal/ruido	0.849		0.249
Voltaje RMS	−0.795		0.168
Entropía	0.711		0.104
Sesgo		1.027	0.046
Kurtosis		0.918	0.141

Con el PCA se obtienen las cargas de los componentes, que indican la correlación entre las variables originales y los componentes principales. Aquí se presentan las cargas para los dos primeros componentes principales ($RC1$ y $RC2$) y la unicidad. En la Tabla 4.6 se indican los resultados para todas las variables.

Con el fin de obtener un modelo robusto, se seleccionaron los parámetros con una carga positiva superior a 0.95 en ambos componentes principales para su uso en el entrenamiento del modelo y se definió al verdor como la clase por clasificar para un primer entrenamiento del modelo con k-medias. En configuración básica del modelo k-medias, se seleccionaron pesos rectangulares con distancias euclidianas, utilizando el 60 % de las lecturas como datos de entrenamiento, 20 % de pruebas y 20 % de validación. Como resultado, en la curva de exactitud en la clasificación (Fig. 4.19) se aprecia como la exactitud del modelo con los datos de entrenamiento comienza en 1 y disminuye hasta 0.94 conforme aumentan la cantidad de iteraciones (k), siendo un claro indicador de sobre-ajuste en el modelo.

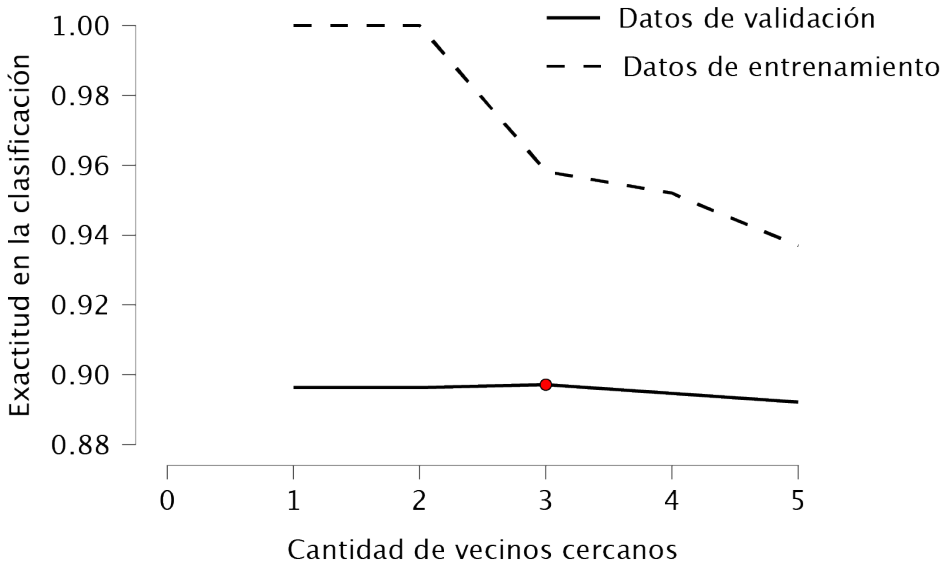
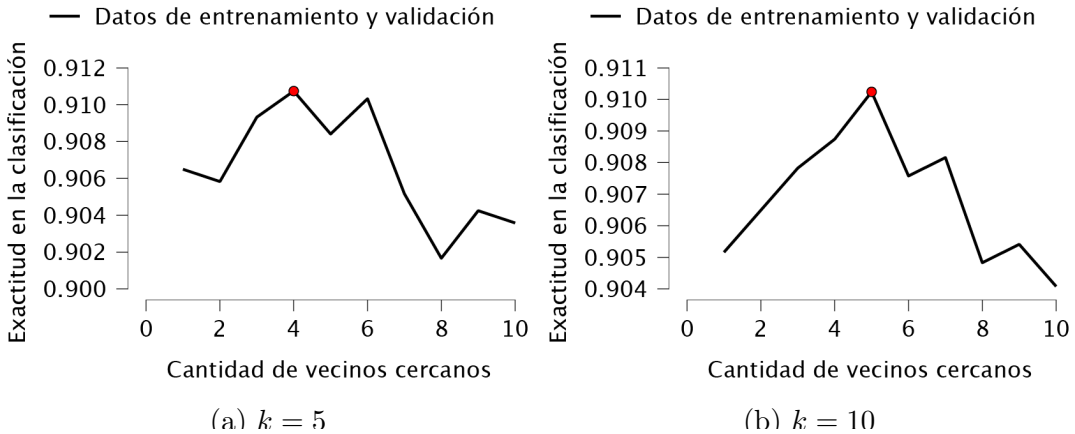


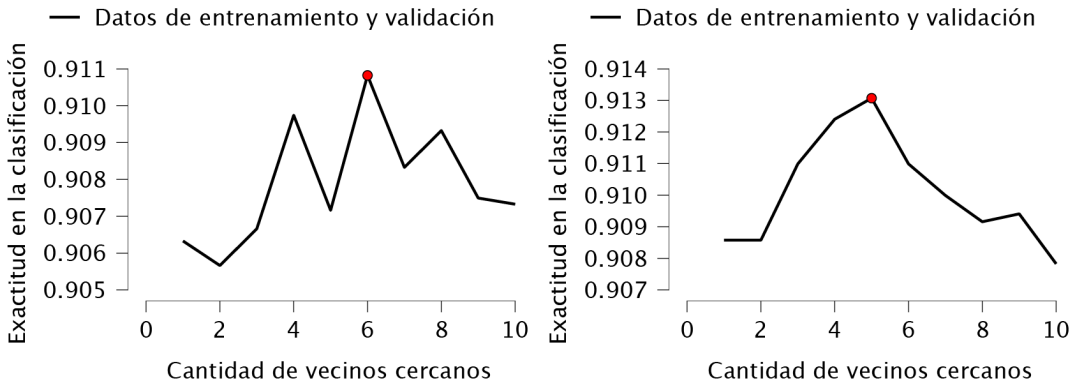
Figura 4.19: Exactitud del modelo inicial.

Ya que siempre se busca evitar el sobre-ajuste para obtener un modelo robusto que generalice datos nuevos, se cambio la configuración de entrenamiento agregando validación cruzada (k-folds) con la técnica Leave-One-Out y con 5, 10 y 20 entrenamientos. También, se cambiaron los pesos rectangulares por triangulares y se mantuvo la distan-

cia euclidiana. Al realizar la validación cruzada se evita el sobre-ajuste por completo para todos los casos, como se muestra en la Fig. 4.20.



(a) $k = 5$ (b) $k = 10$



(c) $k = 20$ (d) Leave-One-Out

Figura 4.20: Variación en la exactitud con múltiples ciclos de entrenamiento para validación cruzada.

Las exactitudes en la clasificación están expresadas en la Tabla 4.7.

Tabla 4.7: Exactitudes para los entrenamientos con validación cruzada.

k-fold	Exactitud de validación (%)	Exactitud de prueba (%)	Precision promedio
$k = 5$	91.7	92.8	91.7
$k = 10$	91.9	92.2	92.3
$k = 20$	91.5	90.5	90.6
Leave-One-Out	90.9	91.3	91.6

Para este proyecto se utilizó $k = 10$ debido a que presenta mayor precisión media (92.3 %) y un F1 Score de 91.7 %. Esto indica que el modelo es realmente confiable y que hace una correcta clasificación de las cinco clases. Los resultados completos de evaluación del modelo se muestran en la Tabla 4.8.

Tabla 4.8: Parámetros de rendimiento del modelo.

Parámetro / Clase	Verde Oscuro	Verde Semi-Oscuro	Verde	Verde Semi-Claro	Verde Claro	Media / Total
Soporte	560	610	597	616	619	3002
Exactitud	0.955	0.989	0.970	0.964	0.956	0.967
Precisión	0.882	0.966	0.921	0.915	0.900	0.917
Verdaderos positivos	0.877	0.982	0.931	0.909	0.885	0.917
Falsos positivos	0.027	0.009	0.020	0.022	0.026	0.021
Tasa de descubrimiento falso	0.118	0.034	0.079	0.085	0.100	0.083
F1 Score	0.879	0.974	0.926	0.912	0.893	0.917
Área bajo la curva	0.982	0.998	0.991	0.985	0.984	0.988
Verdaderos negativos	0.972	0.995	0.983	0.977	0.970	0.979
Tasa de verdaderos negativos	0.973	0.991	0.980	0.978	0.974	0.979
Tasa de falsos negativos	0.123	0.018	0.069	0.091	0.115	0.083
Tasa de omisiones falsas	0.028	0.005	0.017	0.023	0.030	0.021
Puntuación de riesgo	2.443	11.302	4.058	3.500	2.839	4.828
Paridad estadística	0.186	0.207	0.201	0.204	0.203	1.000

A partir de estos resultados, se puede establecer que el uso de k-medias es más que suficiente para realizar una estimación exacta y precisa del verdor utilizando la señal capturada por la apertura del LS.

Conclusiones

Las técnicas de Visión Artificial permiten obtener diversos tipos de datos útiles para resolver tareas específicas, como la localización, segmentación y clasificación de objetos y escenas. Sin embargo, la abundancia de información no garantiza su precisión y exactitud, ya que es poco probable que un solo sistema pueda abordar eficazmente todas las condiciones que se presentan en una aplicación real. Al final, la calidad de la información adquirida por estos sistemas afecta directamente a la toma de decisiones automatizada. Si no se cuenta con información veraz, las decisiones basadas en estos datos pueden ser erróneas y llevar a resultados alejados de lo óptimo o incluso a fallos en el sistema. Es bajo esta idea que la fusión de información toma relevancia, ya que al tener sistemas que agregan capas de información disponible, se pueden tomar decisiones más informadas del estado del entorno.

En este proyecto se presentó un sistema multi-visión compuesto por LS impreso en 3D con una línea base de $20\,cm$ y un sistema de visión estereoscópica configurado de dos maneras distintas: con un par de cámaras HD y otro par de cámaras SD. Cada sistema fue evaluado de manera independiente para determinar su precisión en la estimación de coordenadas tridimensionales. Los resultados mostraron que el sistema LS puede obtener coordenadas tridimensionales con un error absoluto medio de $0.640\,cm$. En cuanto al sistema de visión estereoscópica, las cámaras HD presentaron un error absoluto medio de $1.563, cm$, mientras que las cámaras SD tuvieron un error absoluto medio de $3.816, cm$ en la estimación de coordenadas tridimensionales.

Al emplear la metodología propuesta de fusión de datos tridimensionales, se obtuvo que la combinación de las nubes de puntos pueden ser fusionadas con un error de $\epsilon_x =$

$0.02259m$, $\epsilon_y = 0.02510251m$ y $\epsilon_z = 0.00823073m$, demostrando la alta compatibilidad práctica entre la información tridimensional proporcionada por ambos sistemas. Estos operan de manera sincronizada a través de un panel de control desarrollado en Python, que permite controlar todos los parámetros de funcionamiento de ambos sistemas y visualizar los datos capturados.

Tomando como ejemplo la clasificación automatizada del verdor de las hojas, es notable la relevancia que presenta cada sistema en distintos procesos clave en la automatización de este proceso. En el caso de las cámaras, es posible realizar la segmentación de hojas utilizando redes neuronales y la estimación de coordenadas tridimensionales. Por su parte, el LS también puede obtener coordenadas tridimensionales y clasificar el verdor de las hojas con una precisión del 91.7 %. Esta precisión es comparable a la de los sistemas de estimación de verdor con cámaras, que alcanzan un 91 %, pero con la ventaja de que el LS no se ve afectado por las condiciones ambientales y puede operar incluso en oscuridad total.

Con ambos sistemas se obtienen capas de información que permiten segmentar los objetos de interés, localizarlos en el espacio con información redundante capaz de complementarse y corregirse, y caracterizar los objetos para tomar decisiones informadas de manera robusta.

Bibliografía

[1] Ke Zhang, Xiaokang Ge, Xia Liu, Zeyu Zhang, Yan Liang, Yongchao Tian, Qiang Cao, Weixing Cao, Yan Zhu, and Xiaojun Liu. Evaluation of the chlorophyll meter and greenseeker for the assessment of rice nitrogen status. *Advances in Animal Biosciences*, 8:359–363, 1 2017.

[2] Lei Zhang and Li Wang. A novel pmma composite containing multi-walled carbon nanotubes/copper phthalocyanine hybrid and its optical limiting effect. *Polymer-Plastics Technology and Engineering*, 51:6–11, 1 2012.

[3] I. Filella, L. Serrano, J. Serra, and J. Peñuelas. Evaluating wheat nitrogen status with canopy reflectance indices and discriminant analysis. *Crop Science*, 35:1400–1405, 9 1995.

[4] Yuan Yuan, Lei Chen, Miao Li, Na Wu, Li Wan, and Shimei Wang. Diagnosis of nitrogen nutrition of rice based on image processing of visible light. *Proceedings - 2016 IEEE International Conference on Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications, FSPMA 2016*, pages 228–232, 1 2017.

[5] Christelle Gée, Emmanuel Denimal, Maël de Yparraguirre, Laurence Dujourdy, and Anne Sophie Voisin. Assessment of nitrogen nutrition index of winter wheat canopy from visible images for a dynamic monitoring of n requirements. *Remote Sensing 2023, Vol. 15, Page 2510*, 15:2510, 5 2023.

[6] Hongrui Ren, Guangsheng Zhou, and Feng Zhang. Using negative soil adjustment factor in soil-adjusted vegetation index (savi) for aboveground living biomass estimation in arid grasslands. *Remote Sensing of Environment*, 209:439–445, 5 2018.

[7] Gang Yang, Ke Huang, Weiwei Sun, Xiangchao Meng, Dehua Mao, and Yong Ge. Enhanced mangrove vegetation index based on hyperspectral images for mapping mangrove. *ISPRS Journal of Photogrammetry and Remote Sensing*, 189:236–254, 7 2022.

[8] K. Chandrasekar, P. Srikanth, Abhishek Chakraborty, Karunkumar Choudhary, and K. V. Ramana. Response of crop water indices to soil wetness and vegetation water content. *Advances in Space Research*, 73:1316–1330, 1 2024.

[9] Haiyong Weng, Jingwen Lv, Haiyan Cen, Mubin He, Yibing Zeng, Shijia Hua, Hongye Li, Youqing Meng, Hui Fang, and Yong He. Hyperspectral reflectance imaging combined with carbohydrate metabolism analysis for diagnosis of citrus huanglongbing in different seasons and cultivars. *Sensors and Actuators B: Chemical*, 275:50–60, 12 2018.

[10] Max Gerhards, Gilles Rock, Martin Schlerf, Thomas Udelhoven, and Willy Werner. Water stress detection using hyperspectral thermal infrared remote sensing. *Workshop on Hyperspectral Image and Signal Processing, Evolution in Remote Sensing*, 2015-June, 7 2015.

[11] Yue Shi, Wenjiang Huang, Juhua Luo, Linsheng Huang, and Xianfeng Zhou. Detection and discrimination of pests and diseases in winter wheat based on spectral indices and kernel discriminant analysis. *Computers and Electronics in Agriculture*, 141:171–180, 9 2017.

[12] Yelu Zeng, Dalei Hao, Alfredo Huete, Benjamin Dechant, Joe Berry, Jing M. Chen, Joanna Joiner, Christian Frankenberg, Ben Bond-Lamberty, Youngryel Ryu, Jingfeng Xiao, Ghassem R. Asrar, and Min Chen. Optical vegetation indices for monitoring terrestrial ecosystems globally. *Nature Reviews Earth & Environment 2022 3:7*, 3:477–493, 5 2022.

[13] Suzanne Mariëlle Marselis, Hao Tang, John David Armston, Kim Calders, Nicolas Labrière, and Ralph Dubayah. Distinguishing vegetation types with airborne waveform lidar data in a tropical forest-savanna mosaic: A case study in lopé national park, gabon. *Remote Sensing of Environment*, 216:626–634, 10 2018.

[14] David Chaparro, Thomas Jagdhuber, María Piles, François Jonard, Anke Fluhrer, Mercè Vall-llossera, Adriano Camps, Carlos López-Martínez, Roberto Fernández-Morán, Martin Baur, Andrew F. Feldman, Anita Fink, and Dara Entekhabi. Vegetation moisture estimation in the western united states using radiometer-radar-lidar synergy. *Remote Sensing of Environment*, 303:113993, 3 2024.

[15] Hiroshi Nakano, Ryo Tanaka, Senlin Guan, and Hideki Ohdan. Predicting rice grain yield using normalized difference vegetation index from uav and greenseeker. *Crop and Environment*, 2:59–65, 6 2023.

[16] Ruohao Guo, Liao Qu, Dantong Niu, Zhenbo Li, and Jun Yue. LeafMask: Towards Greater Accuracy on Leaf Segmentation. *Proceedings of the IEEE International Conference on Computer Vision*, 2021-October:1249–1258, 2021.

[17] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. BlendMask: Top-Down Meets Bottom-Up for Instance Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 8570–8578, jan 2020.

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):386–397, mar 2017.

[19] Generative and Visual Intelligence Learning Research Group. Leaf-Segmentation-Unet: a unet model trained for the semantic segmentation of leaf images, 2021.

[20] Shanwen Zhang and Chuanlei Zhang. Modified U-Net for plant diseased leaf image segmentation. *Computers and Electronics in Agriculture*, 204:107511, jan 2023.

[21] Wang Li, Junfeng Wang, Maoding Liu, and Shiwen Zhao. Real-time occlusion handling for augmented reality assistance assembly systems with monocular images. *Journal of Manufacturing Systems*, 62:561–574, 1 2022.

[22] S. Kumar, C. Micheloni, C. Piciarelli, and G. L. Foresti. Stereo rectification of uncalibrated and heterogeneous images. *Pattern Recognition Letters*, 31:1445–1452, 8 2010.

[23] Wenjie Luo, Alexander G. Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:5695–5703, 2016.

[24] Akos Zarandy, Zoltan Nagy, Balint Vanek, Tamas Zsedrovits, Andras Kiss, and Mate Nemeth. A five-camera vision system for uav visual attitude calculation and collision warning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7963 LNCS:11–20, 2013.

[25] Shihui Liu, Meiping Song, Hui Li, Tingting Yang, Bolun Cui, Xin Li, Jiakang Li, and Dayong Xu. Hyperspectral real-time online processing local anomaly detection via multiline multiband progressing. *IEEE Transactions on Geoscience and Remote Sensing*, 61, 2023.

[26] Hitika Tiwari, Vinod K. Kurmi, Venkatesh K. Subramanian, and Yong Sheng Chen. Distilling knowledge for occlusion robust monocular 3d face reconstruction. *Image and Vision Computing*, 137:104763, 9 2023.

[27] Catalin Golban, Petrut Cobarzan, and Sergiu Nedevschi. A comparison study on replacing stereo disparity with lidar in visual odometry methods. *Proceedings - 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing, ICCP 2020*, pages 277–282, 9 2020.

[28] Guoqiang Fu, Arianna Menciassi, and Paolo Dario. Development of a low-cost active 3d triangulation laser scanner for indoor navigation of miniature mobile robots. *Robotics and Autonomous Systems*, 60:1317–1326, 10 2012.

[29] Luis C. Básaca-Preciado, Oleg Yu. Sergiyenko, Julio C. Rodríguez-Quinonez, Xochitl García, Vera V. Tyrsa, Moises Rivas-Lopez, Daniel Hernandez-Balbuena, Paolo Mercorelli, Mikhail Podrygalo, Alexander Gurko, Irina Tabakova, and Oleg Starostenko. Optical 3d laser measurement system for navigation of autonomous mobile robot. *Optics and Lasers in Engineering*, 54:159–169, 3 2014.

[30] Samera Samsuddin Sah, Khairul Nizam Abdul Maulud, Suraya Sharil, Othman A. Karim, and Biswajeet Pradhan. Monitoring of three stages of paddy growth using multispectral vegetation index derived from uav images. *The Egyptian Journal of Remote Sensing and Space Sciences*, 26:989–998, 12 2023.

[31] Juan Villacrés and Fernando A. Auat Cheein. Construction of 3d maps of vegetation indices retrieved from uav multispectral imagery in forested areas. *Biosystems Engineering*, 213:76–88, 1 2022.

[32] Zhe Wang, Wei Chen, Jianghe Xing, Xuepeng Zhang, Haijing Tian, Hongzhao Tang, Pengshuai Bi, Guangchao Li, and Fengjiao Zhang. Extracting vegetation information from high dynamic range images with shadows: A comparison between deep learning and threshold methods. *Computers and Electronics in Agriculture*, 208:107805, 5 2023.

[33] Geoffrey H. Donovan, Demetrios Gatziolis, Kristen Jakstis, and Saskia Comess. The natural environment and birth outcomes: Comparting 3d exposure metrics derived from lidar to 2d metrics based on the normalized difference vegetation index. *Health & Place*, 57:305–312, 5 2019.

[34] Qinan Lin, Huaguo Huang, Jingxu Wang, Ling Chen, Huaqiang Du, and Guomo Zhou. Early detection of pine shoot beetle attack using vertical profile of plant traits through uav-based hyperspectral, thermal, and lidar data fusion. *Interna-*

tional Journal of Applied Earth Observation and Geoinformation, 125:103549, 12 2023.

[35] Oleg Sergiyenko, Vira Tyrsa, Moisés Rivas López, Daniel Hernández Balbuena, Carlos Básaca Preciado, Rodríguez Quiñonez Julio Cesar, and Wendy Flores Fuentes. Sistema óptico de triangulación dinámica para la medición de ángulos y coordenadas en un espacio tridimensional, 2015.

[36] Oleg Sergiyenko, Vira Tyrsa, Moisés Rivas López, Daniel Hernández Balbuena, Carlos Básaca Preciado, Rodríguez Quiñonez Julio Cesar, and Wendy Flores Fuentes. Sistema técnico de visión por triangulación dinámica generando un campo de visión continuo, 2017.

[37] Wendy Flores-Fuentes, Eduardo Arellano-Vega, Oleg Sergiyenko, Iván Y. Alba-Corpus, Julio C. Rodríguez-Quiñonez, Moises J. Castro-Toscano, Félix F. González-Navarro, S. Vasavi, Jesús E. Miranda-Vega, Daniel Hernández-Balbuena, Fabián N. Murrieta-Rico, and Moisés Rivas-López. Surface color estimation in 3d spatial coordinate remote sensing by a technical vision system. *Optical and Quantum Electronics*, 56:1–30, 3 2024.

[38] Yanda Shao, Ling Li, Jun Li, Qilin Li, Senjian An, and Hong Hao. Monocular vision based 3d vibration displacement measurement for civil engineering structures. *Engineering Structures*, 293:116661, 10 2023.

[39] Shreyas S. Shivakumar, Kartik Mohta, Bernd Pfrommer, Vijay Kumar, and Camillo J. Taylor. Real time dense depth estimation by fusing stereo with sparse depth measurements. *Proceedings - IEEE International Conference on Robotics and Automation*, 2019-May:6482–6488, 5 2019.

[40] Ting Lu, Kexin Ding, Wei Fu, Shutao Li, and Anjing Guo. Coupled adversarial learning for fusion classification of hyperspectral and lidar data. *Information Fusion*, 93:118–131, 2023.

[41] Zhehui Huang and Dong Li. A 3d reconstruction method based on one-dimensional galvanometer laser scanning system. *Optics and Lasers in Engineering*, 170:107787, 11 2023.

[42] Xu Dong, Binnan Zhuang, Yunxiang Mao, and Langechuan Liu. Radar camera fusion via representation learning in autonomous driving. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1672–1681, 2021.

[43] Xiangmo Zhao, Pengpeng Sun, Zhigang Xu, Haigen Min, and Hongkai Yu. Fusion of 3d lidar and camera data for object detection in autonomous vehicle applications. *IEEE Sensors Journal*, 20:4901–4913, 5 2020.

[44] Miguel Oliveira, Vitor Santos, and Angel D. Sappa. Multimodal inverse perspective mapping. *Information Fusion*, 24:108–121, 2015.

[45] Wendy Flores-Fuentes, Moises Rivas-Lopez, Oleg Sergiyenko, Julio C. Rodríguez-Quiñonez, Daniel Hernández-Balbuena, and Javier Rivera-Castillo. Energy center detection in light scanning sensors for structural health monitoring accuracy enhancement. *IEEE Sensors Journal*, 14:2355–2361, 2014.

[46] Rou Su, Jingliang Zhong, Qiaoliang Li, Suwen Qi, Huisheng Zhang, and Tianfu Wang. An automatic calibration system for binocular stereo imaging. *Proceedings of 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference, IMCEC 2016*, pages 896–900, 2 2017.

[47] Piotr Perek, Dariusz Makowski, Aleksander Mielczarek, Andrzej Napieralski, and Przemysław Sztoch. Towards automatic calibration of stereoscopic video systems. *Proceedings of the 22nd International Conference Mixed Design of Integrated Circuits and Systems, MIXDES 2015*, pages 134–137, 8 2015.

[48] Mustafa Ghanim, Ozgur Tasdizen, H. Fatih Ugurdag, and Ilker Hamzaoglu. An efficient algorithm for disparity map compression based on spatial correlations and its low-cost hardware architecture. *Integration*, 93:102069, 11 2023.

[49] Subhayan Mukherjee and Ram Mohana Reddy Guddeti. A hybrid algorithm for disparity calculation from sparse disparity estimates based on stereo vision. *2014 International Conference on Signal Processing and Communications, SPCOM 2014*, 12 2014.

[50] Bin Chen and Xin Cheng Tan. Linear weighted median filtering for stereo disparity refinement. *Proceedings of the 32nd Chinese Control and Decision Conference, CCDC 2020*, pages 469–475, 8 2020.

[51] Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

[52] Middlebury. Middlebury stereo evaluation - version 3.

[53] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:60–76, 6 2018.

[54] Karlsruhe Technology Institute and Toyota Technology Institute. The kitti vision benchmark suite.

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:770–778, 12 2015.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017-December:5999–6009, 6 2017.

[57] Ranjita Thapa, Kai Zhang, Noah Snavely, Serge Belongie, and Awais Khan. The plant pathology challenge 2020 data set to classify foliar disease of apples. *Applications in Plant Sciences*, 8, 9 2020.

[58] Michigan State University. Dense-leaves.

Apéndice A: Publicaciones

Artículo de revista

- Multi-view 3D data fusion and patching to reduce Shannon entropy in Robotic Vision. **Optics and Lasers in Engineering**

Capítulo de libro

- ROS and Stereovision Collaborative System. *Optoelectronic Devices in Robotic Systems*. **Springer**

Artículos de conferencia

- Obstacle Coordinates Transformation from TVS Body-Frame to AGV Navigation-Frame. **IEEE International Symposium on Industrial Electronics**.
- Detection of Atypical Data in Point Cloud of Technical Vision System using Digital Filtering. **2022 IEEE Mexican International Conference on Computer Science**.
- Analysis of the construction of an autonomous robot to improve its energy efficiency when traveling through irregular terrain. **IECON 2022**.
- Laser Scanning Point Cloud Improvement by Implementation of RANSAC for Pipeline Inspection Application. **IECON 2023**
- Advances in Laser Positioning of Machine Vision System and Their Impact on 3D Coordinates Measurement. **IECON 2023**

Apéndice B: Código de control y adquisición del LS

```
#include <SevSeg.h>
#include <Stepper.h>

SevSeg sevseg;

//Motor 1
#define m1_1 25
#define m1_2 26
#define m1_3 27
#define m1_4 28

//Motor 2
#define m2_1 29
#define m2_2 30
#define m2_3 31
#define m2_4 32

#define pinLaser 23
#define stepsrev 20
#define velrev 20
```

```
Stepper stepperH(stepsrev, m1_1, m1_2, m1_3, m1_4);
Stepper stepperV(stepsrev, m2_1, m2_2, m2_3, m2_4);

#define left 1
#define right 2
#define up 3
#define down 4


//Variables para el control de motores
int dir = 0, lastdirH = 0, lastdirV = 0, steps = 0;
int binH = 1, binV = 1;
bool enc = false;
float vMotor = 0;

//Variable controladora del laser
int laser = 0;

//Variables para la adquisicion

//Voltajes
float vmaxO = 100,vminO = 999,vrefO = 0;

//Banderas
int capture = 0, c = 0, pos = 0, pos_ant = 0, change = 0;
bool sendData = false;

//Datos individuales
float opto = 0,foto = 0;
```

```
//Posiciones de picos y valores
int picoO[2] = {0}, picoF = 0, maxF = -9999, cpicosO = 0;
int diffO = 0,diffF = 0;

//Tolerancia al ruido
int tolR = 95;

//Variables del tacometro
int est = 0, est_ant = 0, cp = 5, l = 0;
int t1 = 0, t2 = 0;
float rps = 0, dif = 0;

//Verificador de pasos
int pasosContados = 0;

void setup() {
  byte numDigits = 4;
  byte digitPins[] = {11,10,9,8};
  byte segmentPins[] = {7,6,5,4,3,2,1,0};
  bool resistorsOnSegments = 0;

  sevseg.begin(COMMON_ANODE, numDigits, digitPins,...
  segmentPins, resistorsOnSegments);
  sevseg.setBrightness(90);

  sevseg.setChars("inic");

  stepperH.setSpeed(velrev);
  stepperV.setSpeed(velrev);
```

```
  analogReadAveraging(2);
  pinMode(A0,INPUT);
  pinMode(A2,INPUT);

  pinMode(pinLaser,OUTPUT);

  pinMode(m1_1,OUTPUT);
  pinMode(m1_2,OUTPUT);
  pinMode(m1_3,OUTPUT);
  pinMode(m1_4,OUTPUT);
  pinMode(m2_1,OUTPUT);
  pinMode(m2_2,OUTPUT);
  pinMode(m2_3,OUTPUT);
  pinMode(m2_4,OUTPUT);

  digitalWrite(m1_1,0);
  digitalWrite(m1_2,0);
  digitalWrite(m1_3,0);
  digitalWrite(m1_4,0);
  digitalWrite(m2_1,0);
  digitalWrite(m2_2,0);
  digitalWrite(m2_3,0);
  digitalWrite(m2_4,0);


  Serial.begin(9600);
}

void loop() {
```

```
// Posicionador
if (Serial.available() > 0){
  String dataSt = Serial.readString();
  int str_len = dataSt.length() + 1;
  char data[str_len];
  dataSt.toCharArray(data, str_len);
  //sscanf(data,"D\%d S\%d L\%d",\&dir,\&steps,\&laser);
  sscanf(data,"D\%d-S\%d",\&dir,\&steps);

  if (dir!=0){
  switch(dir){
    case 1:
      dir = left;
      break;
    case 2:
      dir = right;
      break;
    case 3:
      dir = up;
      break;
    case 4:
      dir = down;
      break;
    }

  if (lastdirH == 0 \&\& dir < 3){
    lastdirH = dir;
    }

  if (lastdirV == 0 \&\& dir > 2){
```

```
    lastdirV = dir;
    }

    if (dir == 1 || dir == 2){
      if (lastdirH != dir){
        if (binH>1){binH = -(binH-6);}
              lastdirH = dir;
        }
      }

    if (dir == 3 || dir == 4){
      if (lastdirV != dir){
        if (binV>1){binV = -(binV-6);}
              lastdirV = dir;
        }
      }

int i = 0;

    for (i = 1;i <= steps+1;i++){
      delay(10);

      if (i > 1){
        if(dir == 2 || dir == 4){
          if (dir == 2){digitalWrite(m1_1+binH-1,0);}
          if (dir == 4){digitalWrite(m2_1+binV-1,0);}
        }

        if(dir == 1 || dir == 3){
          if (dir == 1){digitalWrite(m1_4-binH+1,0);}
```

```
      if (dir == 3){digitalWrite(m2_4-binV+1,0);}
    }

  if (dir == 1 || dir == 2){
    if (binH == 4){binH = 0;}
    binH++;
  }else{
    if (binV == 4){binV = 0;}
    binV++;}

    }

  if (dir == 1 || dir == 2){
    if (dir == 2){digitalWrite(m1_1+binH-1,1);}
    if (dir == 1){digitalWrite(m1_4-binH+1,1);}

  }

  if (dir == 3 || dir == 4){
    if (dir == 4){digitalWrite(m2_1+binV-1,1);}
    if (dir == 3){digitalWrite(m2_4-binV+1,1);}
  }
}

digitalWrite(m1_1,0);
digitalWrite(m1_2,0);
digitalWrite(m1_3,0);
digitalWrite(m1_4,0);
digitalWrite(m2_1,0);
digitalWrite(m2_2,0);
```

```
      digitalWrite(m2_3,0);
      digitalWrite(m2_4,0);

      Serial.println((String) 5);
    }
    }

if (c>50000){
  vmaxO = vminO * 2;
  }

opto = analogRead(A0);
foto = analogRead(A2);

if (foto <= tolR){foto = 0;}

opto = - (opto-1023);

//Obtiene el valor de vref del opto
if (opto < vminO){
     vminO = opto;
  }
if (opto > vmaxO || vrefO > vmaxO){
     vmaxO = opto;
  }

vrefO = (vminO + vmaxO) / 3;

//Determina cuando se comienzan a analizan datos
if (opto<=vrefO){
```

```
    capture=1;
  }

//Analiza datos
if (capture==1){
  //Obtiene el maximo del fotosensor
  if (foto > maxF){
    maxF = foto;
    picoF = c;
    }

  c++;

  if (opto > vrefO * 1.1){
      pos = 1;
      if (pos_ant == 0){

        t1 = t2;
        t2 = millis();

        dif = abs(t2-t1);
        rps = 1000/dif;

        l++;

          change = 1;
          if (cpicosO == 1){
            picoO[0] = c;
            }
          else{
```

```
            picoO[1] = c;
            }
        }
      pos_ant = 1;
    }
  else if (opto < vrefO * .9){
      pos = 0;
      if (pos_ant == 1){
          change = 2;
          if (cpicosO == 0){
            cpicosO++;
            }
          else{
            cpicosO=2;
            }
        }
      pos_ant = 0;
    }

  if (cpicosO == 2 \&\& change  == 1){
    diffO=picoO[1]-picoO[0];
    diffF=picoF-picoO[0];

    Serial.println((String) "O" + diffO + "-F" + diffF);

    maxF = -9999;
    c = 0;
    cpicosO = 1;
    picoO[0] = 0;
    picoO[1] = 0;
```

```
    }
  }

  if (l == cp){
    sevseg.setNumber(rps*100, 2);
    l = 0;
  }

  sevseg.refreshDisplay();
}
```

Apéndice C: Stereo Tunner

```
import cv2
from os.path import exists
from matplotlib import pyplot as plt
from matplotlib.widgets import Slider
import numpy as np
import json
import actualPath
import readCSV
import os

def tunning(alg):
    dirImg = actualPath.dirImg()
    dirData = actualPath.dirData()
    imgL = cv2.imread(dirImg+"imI.png")
    imgR = cv2.imread(dirImg+"imD.png")

    imgL = cv2.cvtColor(imgL, cv2.COLOR_BGR2GRAY)
    imgR = cv2.cvtColor(imgR, cv2.COLOR_BGR2GRAY)
    rectified_pair = (imgL, imgR)

    #Valores por default
    SWS = 7
```

```
PFS = 7
PFC = 29
MDS = -3
NOD = 48
TTH = 13
uniqr = 3
speckleR = 14
SPWS = 2
MDF = 10
loading_settings = 0

algName = alg.get()
alg = alg.current()

axcolor = 'lightgoldenrodyellow'
fig = plt.subplots(1,2)
plt.suptitle(algName)
plt.subplots_adjust(left=0.15, bottom=0.5)
plt.subplot(1,2,1)

dmObject = plt.imshow(rectified_pair[0], 'gray')

disparity = stereo_depth_map(rectified_pair,SWS,PFS,PFC,MDS,NOD,...
TTH,uniqr,speckleR,SPWS,MDF,alg)

plt.subplot(1,2,2)
dmObject = plt.imshow(disparity, aspect='equal', cmap='jet')

SPWSaxe = plt.axes([0.15, 0.33, 0.7, 0.025],facecolor=axcolor)
SRaxe = plt.axes([0.15, 0.29, 0.7, 0.025],facecolor=axcolor)
```

```
URaxe = plt.axes([0.15, 0.25, 0.7, 0.025],facecolor=axcolor)
TTHaxe = plt.axes([0.15, 0.21, 0.7, 0.025],facecolor=axcolor)
NODaxe = plt.axes([0.15, 0.17, 0.7, 0.025],facecolor=axcolor)
MDSaxe = plt.axes([0.15, 0.13, 0.7, 0.025],facecolor=axcolor)
PFCaxe = plt.axes([0.15, 0.09, 0.7, 0.025],facecolor=axcolor)
PFSaxe = plt.axes([0.15, 0.05, 0.7, 0.025],facecolor=axcolor)
SWSaxe = plt.axes([0.15, 0.01, 0.7, 0.025],facecolor=axcolor)
MDFaxe = plt.axes([0.15, 0.37, 0.7, 0.025],facecolor=axcolor)

sSPWS = Slider(SPWSaxe, 'SpklWinSze', 0.0, 300.0, valinit=SPWS)
sSR = Slider(SRaxe, 'SpcklRng', 0.0, 40.0, valinit=speckleR)
sUR = Slider(URaxe, 'UnicRatio', 1.0, 20.0, valinit=uniqr)
sTTH = Slider(TTHaxe, 'TxtrThrshld', 0.0, 1000.0, valinit=TTH)
sNOD = Slider(NODaxe, 'NumOfDisp', 16.0, 256.0, valinit=NOD)
sMDS = Slider(MDSaxe, 'MinDISP', -100.0, 100.0, valinit=-MDS)
sPFC = Slider(PFCaxe, 'PreFiltCap', 5.0, 63.0, valinit=PFC)
sPFS = Slider(PFSaxe, 'PreFiltSize', 5.0, 255.0, valinit=PFS)
sSWS = Slider(SWSaxe, 'WindowSize', 5.0, 255.0, valinit=SWS)
sMDF = Slider(MDFaxe, 'Disp12Max', 5.0, 255.0, valinit=MDF)

sSPWS.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sSR.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sUR.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sTTH.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
```

```
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sNOD.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sMDS.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sPFC.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sPFS.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sSWS.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))
sMDF.on_changed(lambda val:update(val,sSWS,sPFS,sPFC,sMDS,...
sNOD,sTTH,sUR,sSR,sSPWS,sMDF,loading_settings,rectified_pair,...
dmObject,alg))

if alg == 0:
    sPFS.set_active(False)
    sTTH.set_active(False)

load_map_settings(0,sSWS,sPFS,sPFC,sMDS,sNOD,sTTH,sUR,sSR,...
sSPWS,sMDF,rectified_pair,dmObject,alg)

plt.show()
```

```
def stereo_depth_map(rectified_pair ,SWS,PFS,PFC,MDS,NOD,TTH, uniqr ,...
speckleR ,SPWS,MDF, alg ):
    dirData = actualPath.dirData()

    c, r = rectified_pair[0].shape
    disparity = np.zeros((c, r), np.uint8)

    if alg == 1:
        m = cv2.StereoBM_create()
        m.setPreFilterType(0)
        m.setPreFilterSize(PFS)
        m.setTextureThreshold(TTH)
    else:
        m = cv2.StereoSGBM_create()

    m.setPreFilterCap(PFC)
    m.setMinDisparity(MDS)
    m.setNumDisparities(NOD)
    m.setUniquenessRatio(uniqr)
    m.setSpeckleRange(speckleR)
    m.setSpeckleWindowSize(SPWS)
    m.setBlockSize(SWS)
    m.setDisp12MaxDiff(MDF)

    dmLeft = rectified_pair[0]
    dmRight = rectified_pair[1]

    filt = False
    if exists(dirData + "wlsfilter.csv") == True:
```

```
        data = readCSV.csvread(dirData+"wlsfilter.csv")
        if int(data) == 1:
            filt = True

    if filt == True:
        disparity_visual = filtering(dmLeft,dmRight,m)
    else:
        disparity =m.compute(dmLeft, dmRight)
        disparity = disparity / 16
        local_max = disparity.max()
        local_min = disparity.min()
        disparity_visual = (disparity-local_min)*(1.0/(local_max-local_mi
        local_max = disparity_visual.max()
        local_min = disparity_visual.min()

    '''dirData = actualPath.dirData()
    writeCSV.csvwrite(dirData+"dispTunner.csv", disparity)'''


    return disparity_visual

def filtering(imI,imD,lm):
    rm = cv2.ximgproc.createRightMatcher(lm)

    # FILTER Parameters
    lmbda = 80000
    sigma = 1.3
    visual_multiplier = 7

    wls_filter = cv2.ximgproc.createDisparityWLSFilter(matcher_left=lm)
```

```
    wls_filter.setLambda(lmbda)

    wls_filter.setSigmaColor(sigma)
    displ = lm.compute(imI, imD)
    dispr = rm.compute(imD, imI)
    displ = np.int16(displ)
    dispr = np.int16(dispr)
    filteredImg = wls_filter.filter(displ, imI, None, dispr)

    filteredImg = cv2.normalize(src=filteredImg,...
    dst=filteredImg, beta=0, alpha=255, norm_type=cv2.NORM_MINMAX);
    filteredImg = np.uint8(filteredImg)

    return filteredImg

def save_map_settings(alg):
    global SWS, PFS, PFC, MDS, NOD, TTH, uniqr, speckleR, SPWS, MDF

    dirData = actualPath.dirData()
    result = json.dumps({'WindowSize':SWS,
              'preFilterSize':PFS,
              'preFilterCap':PFC,
              'minDisparity':MDS,
              'numberOfDisparities':NOD,
              'textureThreshold':TTH,
              'uniquenessRatio':uniqr,
              'speckleRange':speckleR,
              'speckleWindowSize':SPWS,'Disp12Max':MDF},
              sort_keys=True, indent=4,
              separators=(',',':'))
```

```
    if alg == 0:
        fName = 'stereoTunnerSGBM.txt'
    else:
        fName = 'stereoTunnerBM.txt'
    f = open(dirData+str(fName), 'w')
    f.write(result)
    f.close()

def load_map_settings( event,sSWS,sPFS,sPFC,sMDS,sNOD,sTTH,...
sUR,sSR,sSPWS,sMDF,rectified_pair,dmObject,alg):
    dirData = actualPath.dirData()
    loading_settings = 1

    if alg == 0:
        fName = 'stereoTunnerSGBM.txt'
    else:
        fName = 'stereoTunnerBM.txt'

    if exists(dirData+fName):
        f=open(dirData+fName, 'r')
        data = json.load(f)
        if alg == 1:
            sPFS.set_val(data['preFilterSize'])
            sTTH.set_val(data['textureThreshold'])
        else:
            sPFS.set_val(5)
            sTTH.set_val(0)

        sPFC.set_val(data['preFilterCap'])
```

```
        sSWS.set_val(data['WindowSize'])
        sMDS.set_val(data['minDisparity'])
        sNOD.set_val(data['numberOfDisparities'])
        sUR.set_val(data['uniquenessRatio'])
        sSR.set_val(data['speckleRange'])
        sSPWS.set_val(data['speckleWindowSize'])
        sMDF.set_val(data['Disp12Max'])
        f.close()

        loading_settings = 0
        update(0,sSWS,sPFS,sPFC,sMDS,sNOD,sTTH,sUR,...
        sSR,sSPWS,sMDF,loading_settings,rectified_pair,dmObject,alg)

def update(val,sSWS,sPFS,sPFC,sMDS,sNOD,sTTH,sUR,sSR,sSPWS,...
sMDF,loading_settings,rectified_pair,dmObject,alg):
    global SWS, PFS, PFC, MDS, NOD, TTH, uniqr, speckleR, SPWS,MDF

    PFS = int(sPFS.val/2)*2+1
    TTH = int(sTTH.val)
    SWS = int(sSWS.val/2)*2+1
    PFC = int(sPFC.val/2)*2+1
    MDS = int(sMDS.val)
    NOD = int(sNOD.val/16)*16
    uniqr = int(sUR.val)
    speckleR = int(sSR.val)
    SPWS= int(sSPWS.val)
    MDF = int(sMDF.val)

    if ( loading_settings == 0 ):
        disparity = stereo_depth_map(rectified_pair,SWS, PFS,...
```

```
PFC, MDS, NOD, TTH, uniqr, speckleR, SPWS,MDF, alg)
dmObject.set_data(disparity)
plt.draw()
save_map_settings(alg)
```