
UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

TESIS DOCTORAL

---

# Detección Espacial de Objetos Utilizando un Sistema de Visión Estereoscópica para Navegación Autónoma

---

*Autor:*
Oscar Real Moreno

*Director de Tesis:*
Julio C. Rodríguez Quiñonez

*Co-Director de Tesis:*
Oleg Sergiyenko

*Tesis presentada para cumplir con los requisitos para el grado de Doctor en Ciencias*

*en el campo del conocimiento de Instrumentación y mediciones automáticas*

Facultad de Ingeniería
Campus Mexicali

1 de agosto de 2024

# Declaración de Autoría

Yo, Oscar Real Moreno, declaro que esta tesis titulada, «Detección Espacial de Objetos Utilizando un Sistema de Visión Estereoscópica para Navegación Autónoma», y el trabajo presentado en ella son de mi autoría. Confirmo que:

- Este trabajo fue realizado en su totalidad o principalmente mientras estaba matriculado en un programa de investigación en esta Universidad.
- Donde alguna parte de esta tesis fue presentada previamente para un grado o cualquier otra calificación en esta Universidad u otra institución, esto se ha indicado claramente.
- Cuando he consultado el trabajo publicado de otros, siempre se ha atribuido claramente.
- Cuando he citado el trabajo de otros, siempre se ha indicado la fuente. Con la excepción de tales citas, esta tesis es enteramente de mi propia autoría.
- He reconocido todas las fuentes principales de ayuda.
- Cuando la tesis se basa en trabajo realizado por mí en colaboración con otros, he dejado claro exactamente lo que hicieron otros y lo que contribuí yo mismo.

Firmado:

---

Fecha:

---

«*Making mistakes is a lot better than not doing anything*.»

Billie Joe Armstrong

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

# *Resumen*

Facultad de Ingeniería
Campus Mexicali

Doctor en Ciencias

**Detección Espacial de Objetos Utilizando un Sistema de Visión Estereoscópica para Navegación Autónoma**

por Oscar Real Moreno

En este trabajo se aborda la detección espacial de objetos para la navegación autónoma mediante la combinación de la visión estereoscópica y la detección de objetos. Tradicionalmente, la visión estereoscópica utiliza mapas de disparidad, pero estos métodos son computacionalmente intensivos. Se propone un nuevo algoritmo de coincidencia de plantillas, SoRA, diseñado para aplicaciones en tiempo real, con tiempos de ejecución más rápidos y menor costo computacional. Los experimentos muestran que SoRA supera a métodos existentes, manteniendo la precisión con una reducción significativa en el costo computacional, lo que lo hace ideal para la detección espacial de objetos en la navegación autónoma. Además, se presenta un método innovador de calibración de cámaras para mejorar la precisión en la estimación de la profundidad en sistemas de visión estereoscópica. Este método aborda la distorsión de la lente y la orientación relativa de las cámaras, siendo robusto a variaciones de iluminación. Los experimentos muestran mejoras sustanciales en la precisión de la estimación de profundidad en comparación con métodos convencionales. Los experimentos de la combinación del sistema de visión estereoscópica y el algoritmo de detección de objetos demuestran la capacidad del sistema para ser aplicado en el contexto de la navegación autónoma. En conjunto, esta investigación aporta eficientes algoritmos para la detección de objetos y la calibración de sistemas de visión estereoscópica, mejorando las capacidades de la navegación autónoma.

# *Reconocimientos*

Al Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), por brindar el apoyo económico para financiar la investigación con becas para realizar los estudios de posgrado.

A la Universidad Autónoma de Baja California (UABC), por la oportunidad para estudiar un posgrado y obtener el grado de Doctor en Ingeniería. A la Facultad de Ingeniería de la UABC, por las aulas, laboratorios y profesores para la formación académica de estudios de posgrado.

Al Dr. Julio Cesar Rodríguez Quiñonez, mi director de tesis, por darme la oportunidad de realizar este proyecto de investigación, por su apoyo constante a lo largo de mis estudios de posgrado y por compartir su conocimiento y experiencia.

Al Dr. Oleg Sergiyenko, por trabajar con el Dr. Julio Cesar Rodríguez Quiñonez para realizar este proyecto de investigación, por su tiempo y consejos al proyecto.

A mi comité de tesis, conformado por el Dr. Daniel Hernández Balbuena, Dr. Jesús Elías Miranda Vega y Dra Wendy Flores Fuentes, por su apoyo y retroalimentación de cada semestre en las presentaciones de avance de tesis.

# Índice general

# Índice de figuras

# Índice de Tablas

# Capítulo 1

# Introducción

La navegación autónoma es un tema de investigación activo que implica la resolución de varios problemas. Uno de estos problemas es el enfoque de investigación de esta tesis, que se centra en el sistema de visión artificial, su precisión y su capacidad para reconocer objetos en el entorno circundante. Los sistemas de visión artificial para navegación autónoma tienen varias tareas que abordar, como la detección de objetos, el reconocimiento de señales de tráfico y la predicción de carriles como se muestran en la figura 1.1.

La visión artificial es la capacidad de las máquinas para interpretar y comprender el mundo visual utilizando sensores y algoritmos. Juega un papel importante al habilitar a las máquinas para navegar, percibir su entorno y llevar a cabo tareas que involucran la parte de visión. Ejemplos de cómo la visión artificial es utilizada incluyen la detección de objetos (Zou et al., 2023), donde los sistemas pueden ser entrenados para reconocer e identificar objetos específicos; control de calidad (Moru y Borro, 2020), donde los sistemas pueden ser utilizados para inspeccionar productos y encontrar defectos; robótica (Pérez et al., 2016), donde los sistemas de visión artificial pueden guiar robots en tareas de navegación, agarre y manipulación de objetos; imágenes médicas (Yi, Walia y Babyn, 2019), donde se han aplicado redes generativas adversarias para la generación eficiente de datos sin modelado explícito de la densidad de probabilidad; realidad aumentada (Kapp et al., 2021), donde la visión artificial puede rastrear los movimientos de la cabeza y las manos del usuario y superponer objetos virtuales en el mundo real; vehículos autónomos (Badue et al., 2021), donde la visión artificial es un componente crítico para permitir que los vehículos autónomos perciban y comprendan su entorno, incluido el reconocimiento de otros vehículos, peatones y señales de tráfico.

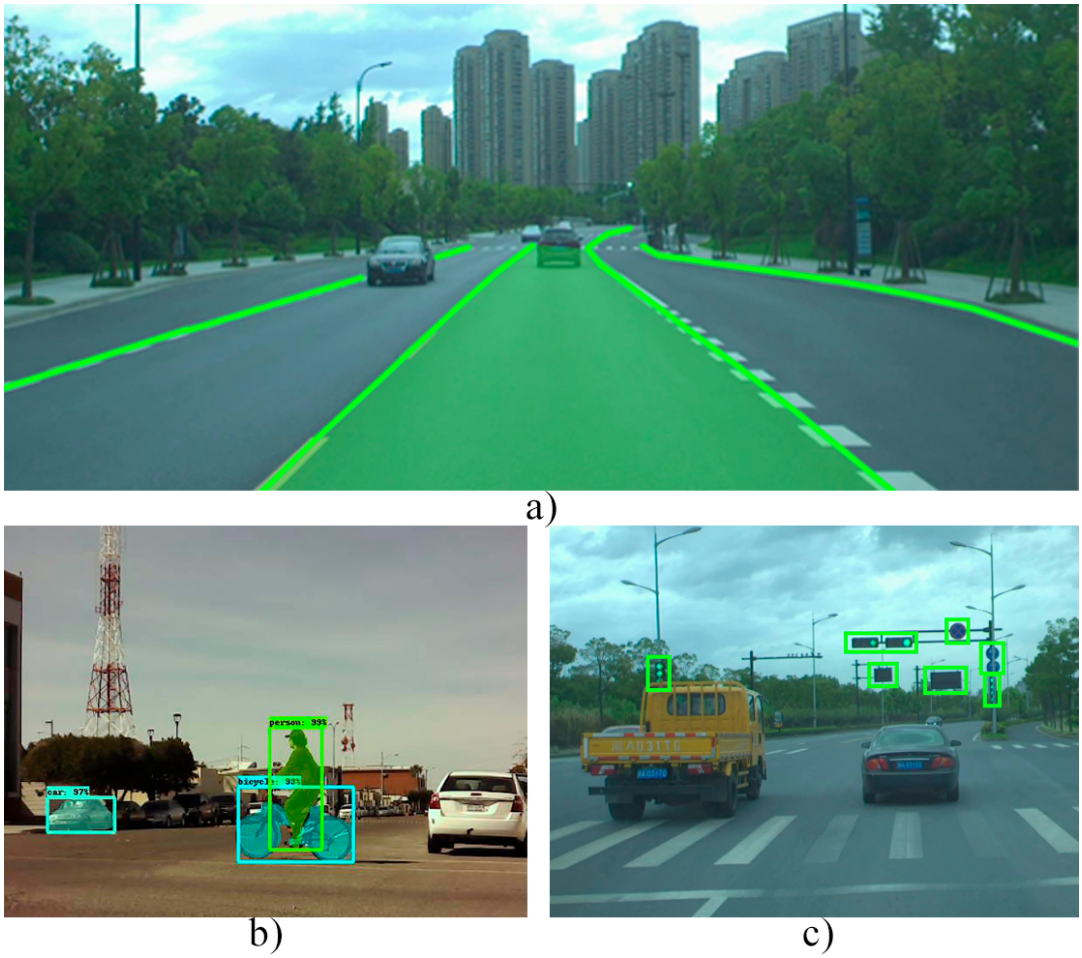



FIGURA 1.1: Tareas de los sistemas de visión artificial para navegación autónoma. a) Predicción de carriles b) Detección de objetos c) Reconocimiento de señales de trafico.

## 1.1. Antecedentes

La tarea de visión artificial en navegación autónoma se ha intentado resolver mediante diferentes métodos. Por ejemplo, en el trabajo de Yang et al., 2020, se utilizan sistemas con múltiples cámaras que incluyen localización y mapeo simultáneo en entornos exteriores, los cuales suelen ser complicados debido a la luz directa del sol, la oclusión ambiental, caminos complejos, entre otros factores. Otro método empleado para abordar esta tarea es el uso de sistemas de visión estereoscópica, como se presenta en el trabajo de Konolige et al., 2008, donde se describe un sistema de visión estereoscópica utilizado para la navegación y el mapeo para el control de un robot pequeño. También existen métodos que prescinden del uso de cámaras, como el sistema de triangulación dinámica propuesto por Básaca-Preciado et al., 2014, que opera mediante un punto láser y un fotoreceptor. Este sistema proporciona

datos tridimensionales precisos del entorno y permite la evasión de objetos y la planificación de trayectorias, ofreciendo mediciones confiables y funcionando eficientemente en entornos con poca iluminación. Por otro lado, hay sistemas que combinan el uso de láseres y cámaras, como se describe en el trabajo de Bjerkeng et al., 2021, donde se emplea la triangulación láser como solución para la navegación autónoma en el interior de una jaula para peces. Este sistema utiliza dos líneas láser y una cámara para estimar la posición de un vehículo autónomo submarino en relación con la red de la jaula. Otro método muy utilizado dentro de los sistemas de visión por láser es el LiDAR (de sus siglas en inglés, Light Detection and Ranging), como se muestra en el trabajo de Li et al., 2021, donde los autores presentan un método de navegación autónoma para un robot de cuatro ruedas en ambientes exteriores complejos. Los autores utilizan un LiDAR 3D y una cámara CCD para percepción basada en visión. El método propuesto combina información de la red de carreteras de OpenStreetMap con percepción local obtenida mediante la multifusión de sensores, con el objetivo de mejorar la precisión de navegación. Por otro lado, los sistemas de escaneo de ondas milimétricas, comúnmente conocidos como radar, se pueden utilizar para mitigar las desventajas del GPS (de sus siglas en inglés, Global Positioning System). En Rouveure et al., 2019, los autores introducen una solución para la localización y navegación en robots móviles utilizando un sistema de radar de microondas terrestre llamado PELICAN. En este trabajo se destacan las limitantes de la localización basada en GPS y proponen el uso del radar como un sistema de posicionamiento complementario. Los sensores ultrasónicos también pueden ser utilizados para el reconocimiento de obstáculos, como se muestra en el trabajo de Meliones, Filios y Llorente, 2022, donde estos sensores son utilizados con un algoritmo de reconocimiento de obstáculos para la navegación en exteriores a ciegas. El algoritmo analiza datos de un dispositivo sonar externo y proporciona advertencias audibles a individuos con discapacidad visual sobre obstáculos cercanos. El dispositivo recopila información utilizando un sensor ultrasónico y un módulo GPS para detectar obstáculos en el camino del usuario y proporcionar instrucciones sobre su distancia, tamaño, movimiento y cómo esquivarlos.

## 1.2. Planteamiento del problema

En el contexto de la navegación autónoma, uno de los desafíos más significativos es la capacidad de un sistema robótico para percibir y comprender

su entorno de manera precisa y confiable. Esto es esencial para que los robots autónomos operen de manera segura y eficiente en entornos diversos y complejos, como carreteras, entornos urbanos y rurales, almacenes industriales y más. La percepción del entorno es fundamental para que los robots tomen decisiones informadas, eviten obstáculos, naveguen de manera autónoma y, en última instancia, cumplan con sus tareas y objetivos asignados. En este contexto, la visión artificial ha surgido como una herramienta esencial para la percepción del entorno en la navegación autónoma, desempeñando un papel crucial en la detección de objetos y la toma de decisiones.

Si bien los métodos mencionados han demostrado ser útiles en diversos contextos de navegación autónoma, también presentan desventajas particulares que limitan su aplicabilidad en ciertos escenarios. Por ejemplo, los sistemas basados en LiDAR, si bien proporcionan mediciones tridimensionales precisas, pueden ser costosos y su rendimiento puede verse afectado por la presencia de obstáculos opacos o condiciones meteorológicas adversas, como lluvia intensa o niebla. En cuanto a los sistemas de radar, aunque ofrecen una alternativa al GPS en entornos con limitaciones de señal, también pueden tener limitaciones en la resolución espacial y pueden requerir una infraestructura de estaciones base para una precisión adecuada. Por otro lado, los sensores ultrasónicos, son efectivos para el reconocimiento de obstáculos cercanos, pero pueden ser limitados en su alcance y capacidad para detectar objetos con precisión en distancias más largas. Por lo tanto, a pesar de sus ventajas, cada uno de estos métodos plantea desafíos específicos que necesitan ser abordados para lograr una navegación autónoma más confiable y versátil. Esta tesis se centra en la visión estereoscópica y la detección de objetos como un enfoque prometedor para superar estas limitaciones y avanzar en la navegación autónoma.

La visión artificial permite que las máquinas interpreten y procesen información visual, capturando datos sobre el mundo que les rodea. En particular, los sistemas de visión estereoscópica han demostrado ser prometedores para mejorar la percepción tridimensional del entorno, lo que es esencial para la navegación autónoma precisa y segura. Sin embargo, a pesar de los avances en este campo, todavía existen desafíos significativos que deben abordarse para aprovechar el potencial de la visión estereoscópica en la detección espacial de objetos y la navegación autónoma. Esta tesis se enfoca en la resolución de estos desafíos, en el desarrollo de un sistema de visión estereoscópica avanzado para la detección y localización de objetos en navegación autónoma. Para comprender plenamente la magnitud de estos desafíos, primero

exploramos los antecedentes y enfoques previos en la navegación autónoma y la visión artificial, identificando las limitaciones existentes y las oportunidades de mejora.

La navegación autónoma es un tema de investigación activo, el cual enfrenta muchos desafíos por resolver. Uno de ellos es el sistema de visión, que debe abordar diversas tareas como la detección de objetos, el reconocimiento de señales de tráfico y la predicción de carriles. El enfoque de esta tesis se centra en la detección espacial de objetos, tarea para la cual es necesario combinar un sistema de visión estereoscópica con un algoritmo de detección de objetos. Con esta combinación, se puede obtener una descripción del objeto detectado (automóvil, persona, bicicleta, perro, etc.), incluyendo su altura, ancho y la distancia del objeto al vehículo, como se muestra en la figura 1.2.


FIGURA 1.2: Detección espacial de objetos utilizando visión estereoscópica.

Varios trabajos se han publicado sobre la detección de objetos aplicada en la navegación autónoma para la percepción del ambiente, localización, planificación de ruta y toma de decisiones, como se muestra en la revisión de Dai et al., 2021. Por otro lado, la combinación entre la detección de objetos y los sistemas de visión estereoscópica ha sido propuesta en trabajos previos, como en Chen et al., 2017, donde los autores presentan un enfoque para la detección tridimensional de objetos para navegación autónoma utilizando imágenes estéreo para generar un conjunto de propuestas de objetos 3D con una red neuronal convolucional (CNN, por sus siglas en inglés, Convolutional Neural Network) construida a partir de Fast R-CNN (Girshick, 2015). En Tao et al., 2021, se propone un método de detección de automóviles basado en R-CNN estéreo a nivel de puntos para la conducción autónoma. Los autores aplican una región de interés en las imágenes izquierda y derecha para minimizar el error fotométrico y obtener la posición precisa; el tamaño del

marco detectado y el cubo delimitador 3D de los vehículos se utilizan para la segmentación de la nube de puntos. En Coenen y Rottensteiner, 2021, los autores utilizan imágenes estéreo para la estimación de la posición y la reconstrucción tridimensional de los vehículos, proponiendo una forma previa consciente de la subcategoría con una predicción basada en una CNN del tipo de vehículo. La combinación entre detección de objetos y sistemas de visión estereoscópica no está limitada únicamente a la navegación autónoma; existen otras aplicaciones, como en el trabajo de Chang, Wang y Chen, 2021, donde los autores desarrollaron un sistema automatizado de irrigación de orquídeas para reducir la posibilidad de enfermedades en las plantas Phalaenopsis. Utilizaron el algoritmo de detección de objetos YOLOv3 (Redmon y Farhadi, 2018) para detectar y encuadrar los centros de las plántulas de orquídeas y luego calcular las coordenadas en 3D de la unión raíz-tallo que debía ser regada utilizando la información de YOLOv3 y el sistema de visión estéreo.

En las siguientes secciones se muestra la justificación y uso de resultados, donde se presenta con mayor precisión el alcance de este problema y se presentan los objetivos generales y específicos de investigación, así como las contribuciones que esta tesis busca aportar al campo de la navegación autónoma y la visión artificial.

## 1.3. Justificación y uso de los resultados

Para lograr una combinación precisa entre la detección de objetos y el sistema de visión estereoscópica, con la velocidad de procesamiento suficiente para aplicarla a la navegación autónoma, hay varios detalles que se deben tener en cuenta. El primero es buscar la correspondencia del objeto en las dos imágenes estéreo en tiempo real, y el segundo es la calibración del sistema, que debe permitir corregir los errores de medición generados por parámetros intrínsecos y extrínsecos sin afectar de manera sustancial la velocidad de procesamiento de las mediciones.

La tarea de correspondencia de objeto, también conocida como coincidencia de plantillas, debe encontrar el objeto detectado en una imagen en la otra imagen para poder calcular la información 3D de los objetos en la escena. En trabajos recientes, uno de los métodos más comunes para realizar esta tarea es utilizar imágenes de disparidad o mapas de profundidad (ver figura 1.3), creadas a partir de imágenes estéreo utilizando enfoques globales, como en

el trabajo de Königshof, Salscheider y Stiller, 2019, donde se propone un método de detección de objetos en 3D y estimación de poses para navegación autónoma utilizando imágenes estéreo. En este trabajo, el método se implementó a través de un GPU para asegurar el procesamiento en tiempo real, logrando tiempos de ejecución de 79 ms. También, en Wang, Lin y Chang, 2021, los autores presentan un enfoque de detección de objetos y estimación de profundidad basado en técnicas profundas de aprendizaje, donde introducen visión binocular a la red de estimación de disparidad basada en visión monocular para estimación de profundidad. En Zhou et al., 2021, presentan un nuevo método para detección simultánea de vehículos y estimación de disparidad desde una imagen combinada de múltiples vistas, donde esta imagen contiene implícitamente la información de apariencia y disparidad que se utiliza para la detección de objetos y estimación de disparidad. En el trabajo de Tankovich et al., 2021, se presenta una arquitectura de una red neuronal para llevar a cabo la coincidencia de plantillas en tiempo real, esta arquitectura se basa en un paso de inicialización multi-resolución rápido, mecanismos de deformación y propagación geométrica 2D diferenciables para inferir la hipótesis de disparidad, logrando tiempos de ejecución más rápidos que 100 ms en los conjuntos de datos KITTI (Geiger et al., 2013). Otro trabajo que logra tiempos de ejecución rápidos es StereoNet (Khamis et al., 2018), el cual es una arquitectura para la tarea de coincidencia de plantillas en tiempo real que se ejecuta a 60 cuadros por segundo en una GPU de alta gama Nvidia Titan X.


FIGURA 1.3: Mapa de profundidad en una imagen del dataset Driving Stereo Yang et al., 2019.

Los enfoques globales han demostrado ser más rápidos que los enfoques locales para la disparidad en las imágenes, pero con un costo computacional alto, ya que la mayoría requiere una GPU de alta gama. Por otro lado, los enfoques locales no requieren una GPU de alta gama, por lo tanto, tienen un costo computacional más bajo, y el tiempo de ejecución depende del algoritmo utilizado. Los enfoques locales trabajan centrando una ventana de

plantilla en la referencia del píxel de una imagen y deslizan esta ventana de plantilla en la otra imagen hasta que se seleccione el punto con mayor correlación como coincidencia válida (Ouyang et al., 2012). El tiempo de ejecución de una coincidencia de un solo píxel en un enfoque local puede variar según el algoritmo, y algunos de ellos pueden lograr tiempos más rápidos que los enfoques globales para un solo píxel. Dado que la fusión entre la detección de objetos y los sistemas de visión estereoscópica solo requiere una coincidencia de un solo píxel para cada objeto, un enfoque local puede tener un costo computacional menor y consumir menos tiempo para realizar la tarea de encontrar el objeto detectado en el par de imágenes estéreo. Algunos métodos con enfoques locales incluyen suma de diferencias absolutas Kuhn, 2013 (SAD, por sus siglas en inglés, Subtraction of Absolute Differences), suma de diferencias de cuadrados Po y Guo, 2007 (SSD, por sus siglas en inglés, Sum of Squared Differences), correlación cruzada normalizada Yoo y Han, 2009 (NCC, por sus siglas en inglés, Normalized Cross-Correlation), SIFT Lowe, 1999 (por sus siglas en inglés, Scale Invariant Feature Transform) y SURF Bay, Tuytelaars y Van Gool, 2006 (por sus siglas en inglés, Speeded-Up Robust Features).

Otros enfoques para la tarea de coincidencia estéreo incluyen el trabajo de Chen, Chen y Chen, 2003, el cual es un algoritmo rápido utilizado para acelerar el proceso de coincidencia estéreo con estimadores "M" para eliminar valores atípicos. En Pele y Werman, 2008 se presenta un método robusto en tiempo real que determina si dos imágenes son similares con respecto a un miembro de la familia de distancia Hamming de imágenes. Otro algoritmo rápido para esta tarea es el muestreo de baja discrepancia, el cual es un algoritmo de coincidencia de patrones basado en el secuenciado de baja discrepancia seleccionando puntos aleatorios de acuerdo a qué tan preciso representa al vecindario (Nair y Wenzel, 1999). En Korman et al., 2013, los autores presentan un método para la comparación de plantillas bajo transformaciones afines 2D minimizando el error de SAD. Un enfoque similar es el de Oron et al., 2017, donde presentan un método para la comparación de plantillas en entornos sin restricciones; este método es llamado Best-Buddies Similarity (BBS), que es una medida de similitud robusta y libre de parámetros entre dos conjuntos de puntos.

Los algoritmos de coincidencia de plantillas mencionados son rápidos, precisos y tienen un menor costo computacional que los enfoques globales para esta tarea, pero no son lo suficientemente rápidos como para ser una mejor alternativa para la fusión entre la detección de objetos y los sistemas de

visión estéreo, como se muestra en los resultados de nuestros experimentos. Los algoritmos SAD y LDS toman tiempos de 3.12 y 38.5 ms en sus mejores resultados, lo que puede parecer un tiempo suficientemente rápido para ser utilizado en la aplicación prevista, pero estos tiempos se sumarán por cada objeto detectado y si un algoritmo de detección de objetos como YOLOv4, que funciona en tiempos de alrededor de 20 ms, solo requeriría 4 objetos en la escena para duplicar el tiempo de ejecución con el algoritmo SAD. Es por eso que en esta tesis se propone un algoritmo de coincidencia de plantillas con un enfoque local para obtener tiempos de coincidencia más rápidos con un menor costo computacional para aplicaciones en tiempo real.

Por otro lado, la calibración de los sistemas de visión estéreo juega un papel importante para lograr reconstrucciones 3D precisas a partir de imágenes 2D capturadas por el par de cámaras. Este proceso conlleva determinar los parámetros intrínsecos y extrínsecos del sistema, donde los parámetros intrínsecos corresponden a la distorsión del lente y los parámetros extrínsecos corresponden a la posición y orientación de las cámaras. Una vez completada la calibración, el sistema puede utilizar esta información para triangular de forma precisa los objetos en la escena, lo que resulta en la generación de nubes de puntos 3D. Sin embargo, la capacidad del sistema para producir reconstrucciones 3D precisas se ve comprometida sin una calibración precisa, lo que lleva a resultados distorsionados. De este modo, la calibración de la cámara juega un papel importante en el proceso de visión estéreo y debe ejecutarse con alta precisión.

Varias técnicas se han desarrollado a lo largo de los años para lograr una calibración de cámara precisa. Estas técnicas incluyen métodos de calibración tradicionales como la calibración por medio de un tablero de ajedrez, así como también técnicas más recientes, como los métodos de auto-calibración. Uno de los métodos más comunes es el método de calibración por medio de un tablero de ajedrez, en el cual el patrón del tablero con dimensiones conocidas se utiliza para determinar los parámetros intrínsecos y extrínsecos de las cámaras. Zhang, 2000 propuso una técnica que calibra la cámara mediante un patrón plano mostrado en diferentes orientaciones para modelar la distorsión radial del lente. El procedimiento consiste en una solución de forma cerrada, seguida por un refinamiento no lineal basado en el criterio de máxima verosimilitud. Por otro lado, Cronk, Fraser y Hanley, 2006 presenta un método para la calibración automática de cámaras digitales de consumo. Este método utiliza los atributos de color RGB en el diseño de objetivos codificados detectados e identificados automáticamente. También incluye un

enfoque para determinar el valor inicial de la orientación relativa junto con aspectos de precisión y el impacto de la aberración cromática.

Otro método estándar es el método fiduciario controlado, en donde un patrón fiduciario controlado con dimensiones conocidas se utiliza para determinar los parámetros de la cámara (Garrido-Jurado et al., 2014). Otro método común es el de optimización por mínimos cuadrados, como el propuesto por Zheng y Peng, 2013. En este método, se utiliza una condición mínima de calibración que consiste en dos puntos de fuga y una línea de fuga. Emplea la optimización de mínimos cuadrados en lugar de cálculos de forma cerrada, donde se aprovechan múltiples observaciones de los puntos de fuga y distancias conocidas en el camino o alturas conocidas por encima del camino para la estimación del vector de traslación de la cámara.

Además, existen métodos basados en algoritmos de aprendizaje de máquinas para la calibración de cámaras, así como métodos de aprendizaje profundo. Bartl y Herout, 2019 propone un enfoque de calibración automática para cámaras de vigilancia que utiliza objetos rígidos en la escena, como vehículos, para estimar puntos de referencia automáticamente y determinar posiciones 3D utilizando una red neuronal convolucional. Esto mejora la precisión de la calibración y reduce las restricciones en los datos de entrada. Por otro lado, Huo et al., 2022 propone un método de calibración de cámara que puede utilizarse en áreas fuera de enfoque utilizando un patrón de calibración circular basado en codificación de fase. También emplea una red de recuperación de fase basada en aprendizaje profundo (Phase-Net) para superar la dificultad de calibrar en zonas peligrosas o de difícil acceso y corregir la excentricidad de la elipse.

A pesar de la eficacia de varias técnicas de calibración para solucionar el problema de distorsión del lente, cada método tiene sus propias limitaciones. Los métodos tradicionales requieren múltiples imágenes del patrón de calibración para modelar de manera precisa la distorsión del lente, haciéndolos lentos y laboriosos. Otros métodos, como el de distorsión radial, fueron diseñados para aplicaciones específicas, como cámaras de vigilancia, y pueden no ser adecuados para otros tipos de cámaras o sistemas de imágenes. Por otro lado, las técnicas basadas en aprendizaje profundo, como las redes neuronales convolucionales y redes totalmente convolucionales, ofrecen una alternativa prometedora para corregir la distorsión del lente. Sin embargo, estos métodos necesitan la generación de un dataset amplio, lo cual puede ser una tarea significativa. Además, estos métodos suelen incurrir en un coste computacional más alto, volviéndolos menos prácticos para aplicaciones

en tiempo real o para las aplicaciones con recursos computacionales limitados. Es por eso que en esta tesis se propone un método de calibración de cámaras que incluye calibración de parámetros intrínsecos y extrínsecos mediante una sola imagen por cámara para modelar la distorsión del lente y la corrección de la posición relativa de las cámaras calculando un ángulo de compensación.

Esta tesis presenta un enfoque integral para implementar la detección espacial de objetos utilizando un sistemas de visión estereoscópica en el contexto de la navegación autónoma. Las principales contribuciones de esta investigación incluyen un sistema de detección de objetos que combina eficazmente la información de imágenes estéreo con un nuevo algoritmo de coincidencia de plantillas con la capacidad de trabajar en tiempo real y alta precisión en la búsqueda de patrones. Además, se propone un método de calibración de cámaras que permite obtener mediciones de alta precisión al corregir los errores de distorsión del lente y la posición relativa de las cámaras, todo ello utilizando una sola imagen por cámara. Estas innovaciones permiten la generación de nubes de puntos 3D precisas y en tiempo real, lo que facilita la navegación autónoma en diversos entornos y aplicaciones.

## 1.4. Objetivos de la investigación

### 1.4.1. Objetivo general

El objetivo general de esta tesis es desarrollar un sistema para la detección espacial de objetos en tiempo real mediante la combinación de un sistema de visión estereoscópica y un algoritmo de detección de objetos. Este sistema combina eficazmente la información de imágenes estéreo, mediante el desarrollo de un algoritmo de coincidencia de plantillas con precision y velocidad superior a los algoritmos actuales, ademas de realizar una calibración precisa de las cámaras para obtener mediciones 3D de alta precisión mediante un nuevo metodo de calibracion de camaras. La finalidad de esta investigación es facilitar la navegación autónoma en diversas aplicaciones, mejorando la percepción y comprensión del entorno por parte de los sistemas autónomos, lo que contribuirá a su seguridad y eficacia en diferentes escenarios dentro de la navegacion autonoma.

### 1.4.2. Objetivos específicos

- Construir un prototipo del sistema de visión estereoscópica con una geometría coplanar y una separación entre ellas que permita realizar mediciones a corta distancia, en un rango de unos pocos metros, así como a largas distancias, entre 15 y 20 metros. La elección de la geometría coplanar se basa en la simplificación de la calibración y el alineamiento de las cámaras, lo que reducirá la complejidad del sistema y asegurará mediciones consistentes y confiables en tiempo real.

- Desarrollar un algoritmo para la coincidencia de patrones que optimice la velocidad de procesamiento sin comprometer la precisión, superando así el rendimiento de los algoritmos actuales. Esta mejora en la velocidad se busca para garantizar una respuesta más rápida del sistema, sin sacrificar la exactitud en la detección de objetos.

- Realizar una serie de experimentos para evaluar el rendimiento del algoritmo de coincidencia de patrones propuesto en comparación con otros algoritmos disponibles en la literatura, centrándose en el contexto de la navegación autónoma. Estos experimentos se diseñarán específicamente utilizando imágenes relevantes para la navegación autónoma. Los resultados obtenidos se analizarán meticulosamente para determinar las fortalezas y limitaciones del algoritmo propuesto en comparación con sus contrapartes, proporcionando así una evaluación objetiva de su rendimiento y su potencial dentro del ámbito de la navegación autónoma.

- Desarrollar un método de calibración de cámaras que permita ajustar con precisión los parámetros intrínsecos y extrínsecos del sistema de visión estereoscópica. Este método se diseñará para garantizar una calibración robusta y confiable que optimice la precisión de las mediciones 3D obtenidas por el sistema. Se empleará una modelación de la distorsión del lente de las cámaras mediante regresión cuadrática multivariable, con el fin de lograr una estimación precisa de los parámetros intrínsecos de las cámaras. Además, se desarrollarán procedimientos para determinar con precisión los parámetros extrínsecos, tales como la posición y orientación relativa de las cámaras, asegurando mediciones 3D consistentes y confiables en diversas condiciones y escenarios.

- Realizar experimentos para comparar el rendimiento de las mediciones obtenidas por el sistema de visión estereoscópica utilizando el método

de calibración propuesto en comparación con otro método de calibración ampliamente utilizado en la literatura. Estos experimentos se diseñarán meticulosamente para abarcar el campo de visión del sistema. Los resultados obtenidos se analizarán en profundidad para evaluar la eficacia relativa de los dos métodos de calibración, proporcionando así una evaluación objetiva de su desempeño y su idoneidad para su implementación en aplicaciones de visión estereoscópica.

- Integrar el sistema de visión estereoscópica con un algoritmo de detección de objetos con el fin de realizar la detección espacial de objetos. Este proceso de integración se llevará a cabo mediante la implementación de un flujo de trabajo que permita la captura, procesamiento y análisis de imágenes estereoscópicas para identificar y localizar objetos en el espacio tridimensional.

- Conducir una serie de mediciones utilizando objetos posicionados en diferentes zonas frente al sistema de visión estereoscópica, estas mediciones abarcaran diversas distancias y ángulos de visión con el fin de evaluar y validar su funcionamiento. Se recopilarán datos detallados sobre la capacidad del sistema para detectar, localizar y estimar la posición tridimensional de los objetos en cada zona evaluada. Los resultados obtenidos se utilizarán para identificar posibles áreas de mejora y optimización, con el objetivo de garantizar un rendimiento consistente y confiable del sistema en una amplia gama de situaciones de uso.

## 1.5. Metodología

La metodología utilizada en esta investigación está diseñada para lograr la detección espacial de objetos utilizando un sistema de visión estereoscópica combinado con un algoritmo de detección de objetos con el propósito de aplicarlos en el contexto de la navegación autónoma. El enfoque metodológico se basa en varios pasos interrelacionados que abordan el desarrollo de un algoritmo para encontrar la coincidencia de plantillas (En este caso se utilizara para realizar la coincidencia de objetos), el desarrollo de un método de calibración para el sistema de visión estereoscópica, la detección de objetos y la generación de información espacial 3D.

Para llevar a cabo esta investigación se realizó un estudio del estado del arte de los sistemas de visión estereoscópica y los algoritmos de detección de objetos. Posteriormente, se construyó un prototipo del sistema de visión

estereoscópica utilizando una geometría coplanar, con un par de cámaras digitales de lente intercambiable separadas por 100 mm. Las cámaras fueron montadas utilizando piezas fabricadas con impresión 3D en material PETG.

En el estudio del estado del arte se detectó que uno de los problemas en los sistemas de visión estereoscópica para trabajar en tiempo real es el tiempo de ejecución de los algoritmos de coincidencia de plantillas. Por lo cual se desarrolló un nuevo algoritmo que mantuviera la precisión de los algoritmos ya utilizados pero con un tiempo de ejecución superior. Para comprobar el funcionamiento del método se realizaron experimentos donde se compara el método propuesto con otros métodos que realizan la misma tarea.

Otro punto importante que se detectó en el estado del arte de los sistemas de visión estereoscópica recae en el método de calibración de parámetros intrínsecos y extrínsecos. Para obtener mediciones confiables, es importante contar con un método de calibración preciso y en el caso de la aplicación de este sistema, también es importante ajustar parámetros tales como la distorsión generada por el lente y la orientación relativa de las cámaras sin afectar la velocidad de ejecución del sistema para lograr mediciones en tiempo real. Por lo cual se desarrolló un método de calibración que ajusta los parámetros intrínsecos y extrínsecos del sistema que cumple con estas características. Para comprobar el desempeño del sistema se realizaron experimentos donde se llevarón acabo mediciones sobre un patrón con medidas conocidas y se compararon con las mediciones realizadas mediante uno de los métodos más utilizados para calibrar sistemas de visión estereoscópica.

Se combinó el sistema de visión estereoscópica con el algoritmo de detección de objetos conocido como YOLOR, el cual tiene la capacidad de procesar entre 12 y 16 cuadros por segundo al detectar objetos. Este sistema se ejecuta en una laptop con un procesador Intel Core i7 de decimosegunda generación y una GPU Nvidia RTX 3060. Para comprobar el funcionamiento del sistema se realizaron experimentos posicionando objetos frente al sistema en diferentes zonas para comparar la medición obtenida con el valor real de la posición del objeto.

Esta metodología proporciona una base sólida para abordar los objetivos de investigación y permite una comprensión general de cómo se llevó a cabo la investigación. Los detalles específicos de cada paso se describirán con mayor profundidad en la sección 3 de "Procedimiento de Investigación".

# Capítulo 2

# Marco Teórico

En este capítulo, se aborda la teoria detras de la visión estereoscópica y la detección de objetos en sistemas de visión por computadora, con un enfoque específico en su aplicación para la navegación autónoma. Se presenta un análisis detallado de las técnicas clave en cada uno de estos campos, así como su relevancia y aplicaciones en el contexto de la navegación autónoma.

## 2.1. Visión estereoscópica

La visión estereoscópica es una técnica que permite a las computadoras percibir el mundo en tres dimensiones, así como lo hacen los humanos (Real et al., 2019). Los sistemas de visión estereoscópica utilizan dos cámaras, o una sola cámara con dos puntos de vista diferentes (Lin, Tsai et al., 2021), para capturar imágenes de la escena. Después, el sistema procesa estas imágenes para determinar las coordenadas 3D reales de objetos y/o puntos en la escena. Esto se logra mediante un proceso llamado triangulación, en el cual el sistema crea un triángulo entre el objeto y sus proyecciones 2D en las cámaras. Después, al analizar la forma y posición del triángulo, el sistema puede calcular las coordenadas 3D del objeto con un alto grado de precisión. Esto es un componente esencial en muchas aplicaciones de visión por computadoras, ya que esto permite al sistema interactuar con el mundo físico de manera más natural e intuitiva.

El sistema de visión estereoscópica presentado en esta tesis trabaja conforme al diagrama de bloques de la figura 2.1. El sistema inicia revisando si es necesario realizar la calibración de los parámetros intrínsecos y extrínsecos del sistema. Si es necesario, en este paso se implementa el método de calibración propuesto en esta tesis. Una vez que el sistema está calibrado, se obtiene el par de imágenes estéreo y se selecciona un punto en la imagen izquierda que es buscado en la imagen derecha utilizando un algoritmo de coincidencia de plantillas. En este paso se utiliza el algoritmo de coincidencia de plantillas

propuesto en esta tesis. Después, el sistema calibra las coordenadas rectangulares de los puntos de coincidencia en ambas imágenes para compensar la distorsión del lente de cada cámara. Por último, las coordenadas rectangulares calibradas se convierten en ángulos de detección para poder calcular las coordenadas del mundo real X, Y y Z de los puntos seleccionados utilizando triangulación como se puede observar en la sección 2.1.5.

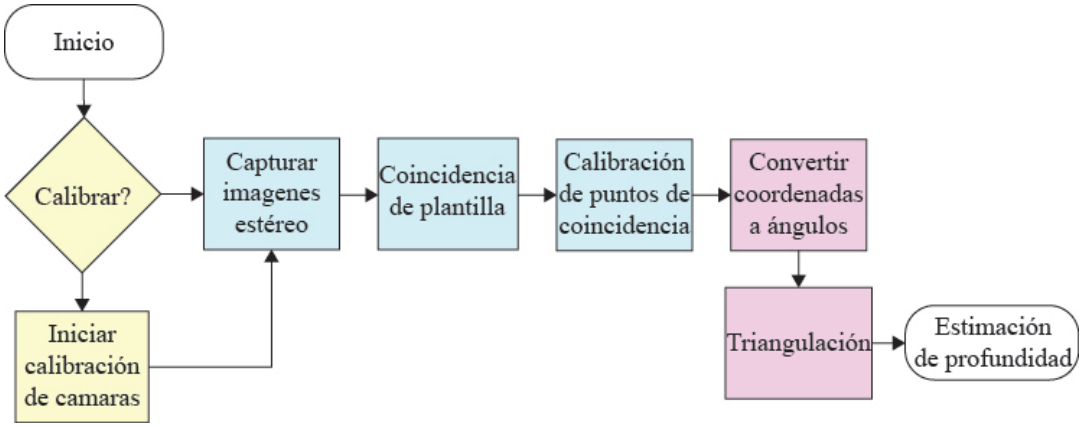



FIGURA 2.1: Diagrama de bloques del sistema de visión estereoscópica.

### 2.1.1. Calibración de parámetros extrínsecos e intrínsecos

La distorsión del lente es una problemática común en las cámaras, manifestándose como una deformación de la imagen que origina curvaturas debido al diseño del lente de la cámara. Esta distorsión puede manifestarse de tres maneras principales: la distorsión de barril, la distorsión de cojín y la distorsión de mostacho. En la figura 2.2 se ilustran estos distintos tipos de distorsión que pueden surgir en las cámaras. La distorsión de barril se evidencia en lentes de gran angular, donde el campo de visión es más amplio que el tamaño del sensor, generando líneas curvas hacia el interior de la imagen. Por otro lado, la distorsión de cojín se observa en lentes telefoto, donde el campo de visión es más estrecho que el sensor de la cámara, resultando en líneas curvas hacia afuera. Por último, la distorsión de mostacho se presenta en lentes con distancia focal variable, donde las líneas rectas se curvan hacia adentro en el centro de la imagen y hacia afuera en los bordes. La distorsión del lente ocasiona imprecisiones en las mediciones realizadas por los sistemas de visión estereoscópica, especialmente en los bordes de la imagen, donde la distorsión tiene un mayor impacto Ramírez-Hernández et al., 2020.

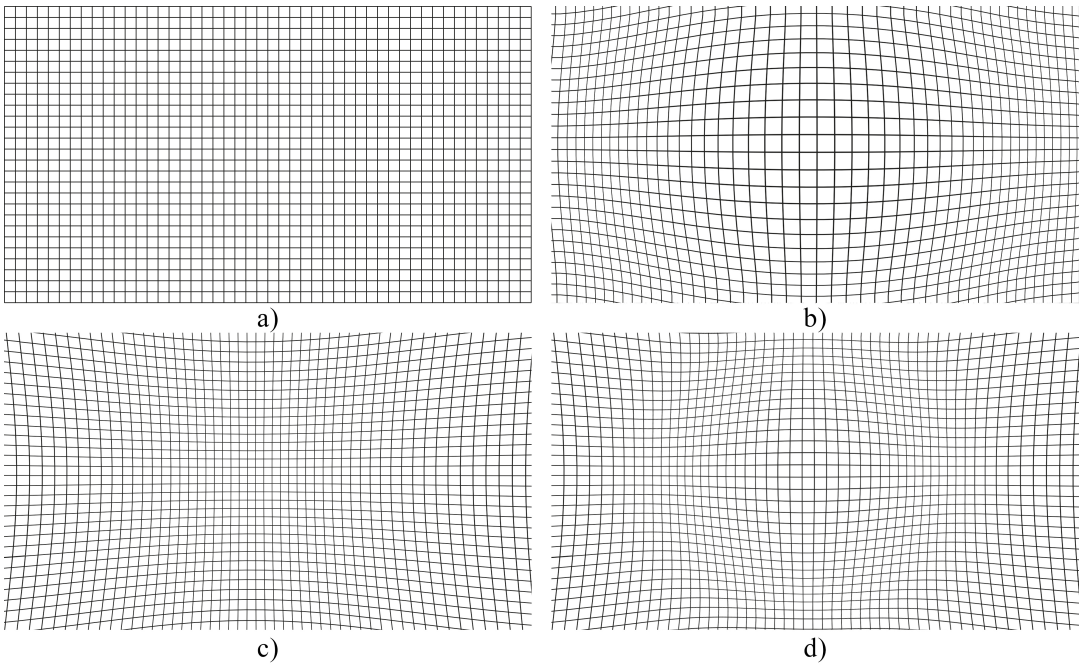



FIGURA 2.2: Tipos de distorsión de lente. a) Sin distorsión b) Distorsión de barril c) Distorsión de cojín d) Distorsión de mostacho.

En los sistemas de visión estereoscópica, es fundamental comprender y corregir los efectos de la distorsión del lente para lograr mediciones precisas de la profundidad y la geometría de la escena. Esto se logra mediante la calibración de la cámara, que implica determinar y ajustar los parámetros intrínsecos y extrínsecos de las cámaras estéreo.

Los parámetros intrínsecos se refieren a las características individuales de cada cámara, como la longitud focal, el centro óptico y los coeficientes de distorsión. Estos parámetros describen cómo la cámara captura una imagen y cómo la luz incide en el sensor. La corrección de la distorsión y la estimación precisa de las coordenadas de los puntos en la imagen dependen de estos parámetros intrínsecos.

Por otro lado, los parámetros extrínsecos se relacionan con la posición relativa y la orientación de las cámaras en el sistema estéreo. Estos parámetros describen cómo las cámaras están dispuestas en el espacio y cómo se alinean con respecto a la escena que se está observando. La corrección de la distorsión y la triangulación precisa también dependen de la calibración y conocimiento de los parámetros extrínsecos.

En conjunto, la calibración intrínseca y extrínseca de las cámaras estéreo es esencial para garantizar mediciones precisas y la correcta reconstrucción

tridimensional de la escena a partir de las imágenes estereoscópicas. La corrección de la distorsión y la alineación adecuada de las cámaras permiten que los sistemas de visión estereoscópica logren una representación fiel de la realidad en tres dimensiones.

### 2.1.2. Par de imágenes estéreo

El primer paso en un sistema de visión estereoscópica es obtener el par de imágenes estéreo, lo cual se lleva a cabo al capturar dos imágenes de la escena desde dos diferentes y definidos puntos espaciales, en los cuales es posible detectar técnicamente la dirección geométrica al punto-objetivo. Estas imágenes son capturadas por un par de cámaras posicionadas en una geometría coplanar, como se muestra en la figura 2.3. El par de imágenes estéreo se utiliza para calcular la información de profundidad de los objetos en la escena. Esto se realiza al analizar las posiciones relativas de los puntos en las superficies en cada plano de imagen, tomando en cuenta distintos factores, como el ángulo de visión de las cámaras y la línea de separación estéreo, es decir, la distancia entre el centro de las cámaras.

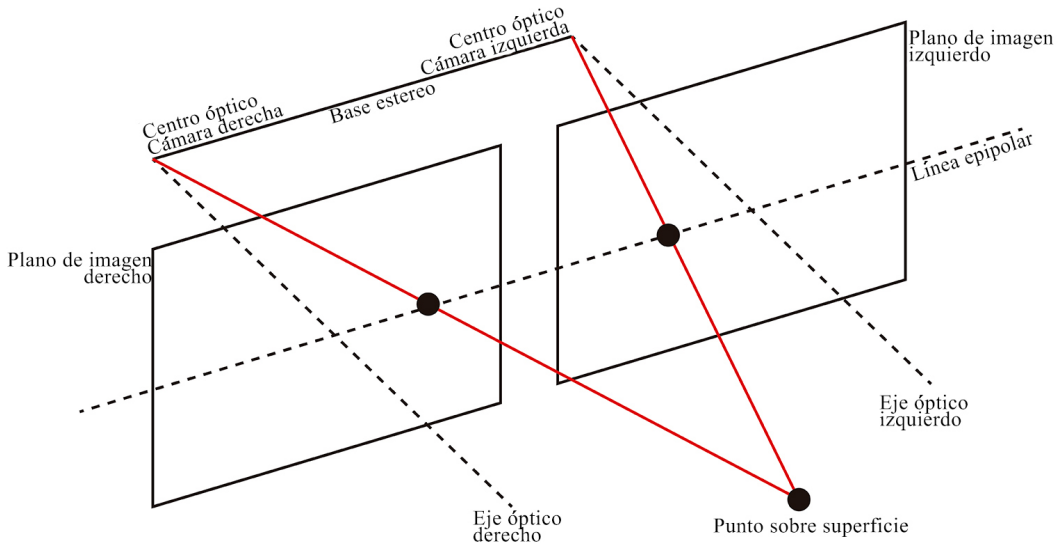



FIGURA 2.3: Principio de triangulación en sistema de visión estereoscópica.

### 2.1.3. Coincidencia de plantilla

Una vez que el par de imágenes estéreo es adquirido, el punto de interés sobre la superficie debe ser identificado en una de las imágenes. Esto se puede realizar mediante una variedad de técnicas, como un algoritmo de detección de objetos Liu et al., 2020, un algoritmo de detección de puntos de referencia Wu y Ji, 2019 o mediante selección manual. Una vez que el punto de interés es seleccionado en una de las imágenes, la misma proyección de este punto debe ser localizada en la otra imagen. A este proceso se le conoce como coincidencia de plantilla, en la literatura existen muchos métodos para llevar a cabo esta tarea Ouyang et al., 2011. Algunas de las técnicas más populares para la coincidencia de plantillas incluyen métodos basados en características, como SIFT Lowe, 1999 y SURF Bay, Tuytelaars y Van Gool, 2006; también métodos basados en correlación como la correlación cruzada normalizada Sarvaiya, Patnaik y Bombaywala, 2009. La figura 2.4 muestra un ejemplo de como se busca la coincidencia de plantilla mediante ventanas deslizantes, esto puede realizarse con distintos métodos como SAD, SSD, NCC, etc. Otro método muy popular es el uso de enfoques basados en técnicas de aprendizaje profundo, como redes neuronales convolucionales (CNNs, por sus siglas en inglés, convolutional neural networks) Chen, Sahdev y Tsotsos, 2017. Estos métodos pueden utilizarse para extraer características de las imágenes y después localizar esas características entre las dos imágenes para encontrar el punto correspondiente en cada plano.


FIGURA 2.4: Coincidencia de plantillas mediante ventanas deslizantes.

### SAD y SDD

La suma de diferencias absolutas (SAD, por sus siglas en inglés Sum of Absolute Differences) es una medida fundamental de similitud ampliamente utilizada en la correlación de imágenes. Esta medida consiste en calcular la diferencia absoluta entre cada píxel de la imagen original y su correspondiente en la imagen de referencia, utilizando una ventana de búsqueda para la comparación Dall'Asta y Roncella, 2014. Por otro lado, la suma de diferencias al cuadrado (SSD, por sus siglas en inglés Sum of Squared Differences) implica elevar al cuadrado las diferencias entre los píxeles correspondientes antes de sumarlas. Luego, estas diferencias se suman y se optimizan mediante la estrategia conocida como "el ganador se lo lleva todo" (WTA, por sus siglas en inglés winner-take-all) Kanade et al., 1995. Las fórmulas para SAD y SSD se expresan en las ecuaciones 2.1 y 2.2, respectivamente:

$$SAD = \sum_{i=1} \sum_{j=1} |f(i,j) - g(i,j)| \tag{2.1}$$

$$SSD = \sum_{i=1} \sum_{j=1} (f(i,j) - g(i,j))^2 \tag{2.2}$$

### NCC

La correlación cruzada normalizada (NCC por sus siglas en inglés Normalized Cross Correlation) es más sofisticada en comparación con SAD y SSD (suma de diferencias al cuadrado), pero posee invariancia ante cambios lineales en la amplitud de la imagen. Al normalizar los vectores de características a unidades de longitud, las medidas de similitud entre las características se vuelven independientes de las variaciones radiométricas (Yoo y Han, 2009). El NCC encuentra coincidencias de una plantilla de referencia $f(i,j)$ de tamaño $m \times n$ en una imagen de escena $g(x,y)$ de tamaño $M \times N$ y se define en la ecuación 2.3.

$$NCC = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} [f(i,j) \cdot g(x+i, y+j)]}{\sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} [f(i,j)^2]} \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} [g(x+i, y+j)^2]}} \tag{2.3}$$

### LDS

El muestreo de baja discrepancia (LDS por sus siglas en inglés Low Discrepancy Sampling) ha demostrado ser un enfoque prometedor en el campo

del emparejamiento de patrones debido a su capacidad para generar secuencias de puntos pseudoaleatorios con una distribución más uniforme en el espacio. Al aplicar LDS al emparejamiento de patrones, se puede lograr una selección más uniforme y efectiva de las muestras de patrones de referencia para su comparación con patrones de consulta en grandes conjuntos de datos. Este enfoque ayuda a mejorar la precisión y eficiencia de los algoritmos de emparejamiento al garantizar una cobertura más equitativa del espacio de búsqueda, lo que a su vez reduce el riesgo de pasar por alto posibles coincidencias. La distribución más uniforme de las muestras de patrones también puede disminuir la probabilidad de errores y falsos positivos, lo que resulta en una mejora general en la precisión y confiabilidad del proceso de emparejamiento (Nair y Wenzel, 1999).

#### BBS

El método Best-Buddies Similarity (BBS) se utiliza para establecer conexiones entre los píxeles de dos imágenes, identificando pares de píxeles que muestran una relación de "mejores amigos" en función de su similitud. BBS se basa en la premisa de que los píxeles similares en diferentes imágenes deben compartir una relación de correspondencia significativa, lo que es fundamental para identificar objetos y patrones comunes en las imágenes. El proceso de emparejamiento en BBS implica la comparación de características visuales de los píxeles en ambas imágenes. Las métricas de similitud pueden variar y pueden incluir diferencias de intensidad, color, textura u otras características visuales relevantes. Una de las ventajas clave de BBS es su capacidad para encontrar correspondencias robustas incluso en presencia de ruido, deformaciones locales o variaciones de iluminación en las imágenes. El cálculo de la similitud en BBS se realiza típicamente mediante diversas medidas de distancia o similitud, como la distancia euclidiana, la correlación cruzada, o medidas de similitud basadas en características específicas. Una vez que se establecen los pares de "mejores amigos", estos puntos pueden utilizarse para realizar tareas de alineación, registro de imágenes, superposición de imágenes, y otras aplicaciones de procesamiento de imágenes que requieran la correspondencia precisa entre los contenidos visuales de las imágenes. BBS ha demostrado ser una herramienta valiosa en una variedad de contextos de procesamiento de imágenes donde la precisión y la fiabilidad en el emparejamiento de patrones son cruciales para el análisis de datos visuales. El BBS entre dos sets de puntos $P$ y $Q$ esta dado por la ecuación 2.4.

$$BBS(P,Q) = \frac{1}{N}\sum_{i=1}\sum_{j=1} bb(p_i, q_j, P, Q) \tag{2.4}$$

Donde *bb* identifica la similaridad, $p_i$ es el vecino mas cercano de $q_j$ en el set Q y viceversa Oron et al., 2017.

**Fast-Match**

FAsT-Match (Fast Affine Template Matching) es un algoritmo utilizado en el campo del procesamiento de imágenes y coincidencia de plantillas. Este método se enfoca en encontrar correspondencias precisas entre una plantilla de referencia y regiones de interés en una imagen, incluso cuando hay presentes transformaciones afines, como rotaciones, escalado y sesgos. Una de las principales fortalezas de FAsT-Match es su capacidad para lograr una correspondencia rápida y precisa, lo que lo convierte en una herramienta valiosa para una variedad de aplicaciones de visión por computadora. La técnica de muestreo de baja discrepancia utilizada por FAsT-Match permite la selección eficiente de puntos de interés en la imagen y la plantilla. Esto facilita una estimación precisa de la transformación afín entre la plantilla y las regiones correspondientes en la imagen. A través de este proceso, se logra una alineación robusta y precisa que permite un emparejamiento efectivo entre la plantilla y las áreas de interés. La versatilidad de FAsT-Match ha llevado a su aplicación en numerosos campos, incluido el seguimiento de objetos en movimiento, el registro de imágenes y la detección de objetos en entornos de imágenes complejas. La combinación de eficiencia y precisión lo convierte en una opción atractiva para aquellos que buscan una solución robusta y confiable para el emparejamiento de patrones en imágenes complejas y en tiempo real (Korman et al., 2013).

### 2.1.4. Calibración de puntos de coincidencia

Una parte importante del proceso de triangulación para realizar estimaciones de profundidad precisas son las coordenadas de los puntos de las proyecciones sobre los planos de las imágenes. Estas coordenadas están dadas por el algoritmo de coincidencia de plantillas, el cual obtiene las coordenadas del par de imágenes estéreo. Sin embargo, el problema con las coordenadas obtenidas a partir de estas imágenes es que tienen un error de distorsión producido al momento de la captura de la imagen (ver figura 2.5). Esta distorsión puede ser producida por múltiples factores, tales como la distorsión

del lente, la transparencia del medio, el ángulo de visión, el gradiente térmico y la inestabilidad dinámica de los objetos en la escena, entre otros. Esta distorsión provoca imprecisiones en el cálculo de profundidad, por lo cual es necesario corregirla. Esto se realiza mediante la calibración de la cámara, que consiste en el proceso de determinar los coeficientes de distorsión. Esto permite la corrección de la distorsión del lente en las coordenadas de los puntos de las proyecciones, lo que resulta en estimaciones de profundidad más precisas. En esta tesis se presenta un método para corregir la distorsión del lente, el cual se explica con más detalle en el capítulo 3, Procedimiento de Investigación. El método de calibración propuesto utiliza un conjunto de puntos conocidos en el mundo real y sus coordenadas correspondientes en las imágenes estéreo. Al comparar estas coordenadas, los coeficientes de calibración pueden ser determinados y utilizados los valores correctos para las coordenadas de las proyecciones en el plano de la imagen. Mediante la implementación de este método, la precisión de la estimación de profundidad se ve mejorada significativamente.

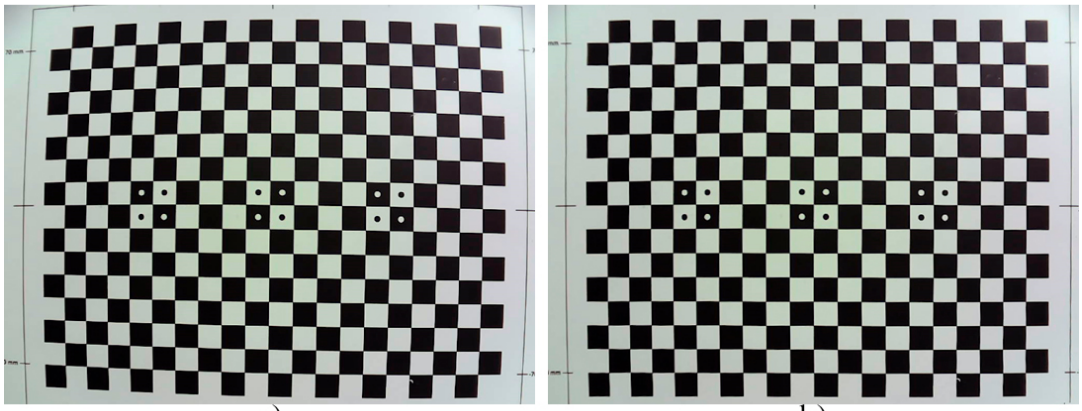



FIGURA 2.5: Comparación de a) imagen de patrón con distorsión y b) imagen de patrón con distorsión corregida.

### 2.1.5. Conversión de coordenadas rectangulares a angulares y triangulación

Una vez que el punto de la proyección sobre los planos de las imágenes es calibrado, el proceso de triangulación se puede llevar a cabo. Este proceso consiste en utilizar las ecuaciones 2.5, 2.6 y 2.7 para calcular las coordenadas $X$, $Y$ y $Z$ del mundo real. Estas ecuaciones son iguales a las ecuaciones utilizadas en el sistema de visión técnica de Rodríguez-Quiñonez et al., 2014.

$$X = b\left(\frac{sinB \cdot sinC}{sin(B+C)}\right) \tag{2.5}$$

$$Y = b\left(\frac{cosB \cdot sinC}{sin(B+C)} - \frac{1}{2}\right) \tag{2.6}$$

$$Z = b\left(\frac{sinB \cdot sinC \cdot tan\beta}{sin(B+C)}\right) \tag{2.7}$$

Donde $b$ es la distancia entre el centro de las cámaras, también conocido como base estéreo. Las variables $B$, $C$, y $\beta$ se obtienen a partir de los puntos de coincidencia. La figura 2.6 muestra una vista superior del sistema de visión estereoscópica donde se puede apreciar la triangulación entre las cámaras y el punto a escanear; también se pueden notar los ángulos $B$ y $C$. Estos ángulos se calculan en relación al ángulo de visión de la cámara y el punto de proyección en la imagen, como se muestra en la figura 2.6. $B$ y $C$ se calculan utilizando las ecuaciones 2.8 y 2.9.

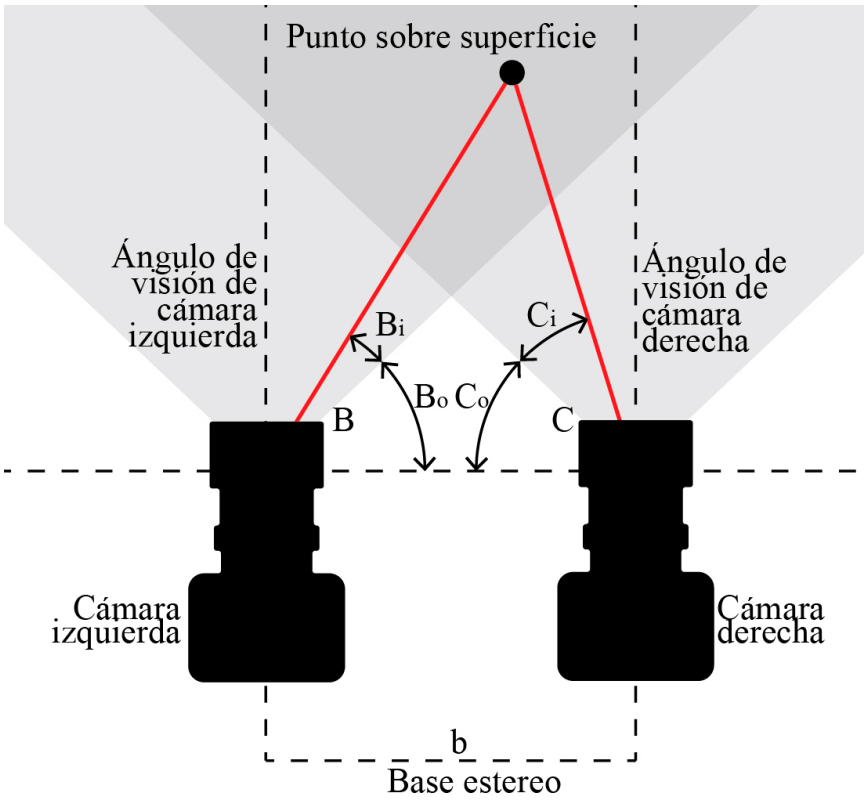



FIGURA 2.6: Ángulos de visión horizontales en sistema de visión estereoscópica (vista superior).

$$B = B_i + B_o \tag{2.8}$$

$$C = C_i + C_o \tag{2.9}$$

Donde $B_i$ y $C_i$ son las partes de los ángulos $B$ y $C$ que se encuentran dentro del ángulo de visión de su respectiva cámara. Estos ángulos se calculan mediante las ecuaciones 2.10 y 2.11. Además, $B_o$ y $C_o$ son los ángulos que van desde la base estéreo $b$ hasta el inicio de su respectivo ángulo de visión, que es la parte del ángulo que falta para completar los ángulos de detección $B$ y $C$. Los ángulos $B_o$ y $C_o$ se calculan utilizando las ecuaciones 2.12 y 2.13. La orientación relativa de las cámaras es un parámetro extrínseco que necesita ser calibrado para obtener mediciones precisas. Esto es parte del método de calibración propuesto, el cual se presenta con más detalle en el capítulo 3 "Procedimiento de Investigación".

$$B_i = H_{av}\frac{W_{is} - SP_{xl}}{W_{is}} \tag{2.10}$$

$$C_i = H_{av}\frac{SP_{xr}}{W_{is}} \tag{2.11}$$

$$B_o = 90^\circ - \frac{H_{av}}{2} \tag{2.12}$$

$$C_o = 90^\circ - \frac{H_{av}}{2} \tag{2.13}$$

Donde $H_{av}$ es el ángulo horizontal de visión de la cámara correspondiente, $W_{is}$ es el ancho de la imagen correspondiente en píxeles, $SP_{xl}$ es el valor $x$ en píxeles de las coordenadas del punto en la superficie en la imagen izquierda y $SP_{xr}$ es el valor $x$ de las coordenadas del punto en la superficie en la imagen derecha. Por otro lado, el ángulo $\beta$ se calcula utilizando la ecuación 2.14.

$$\beta = (V_{av})\left(\frac{\frac{V_{is}}{2} - SP_{ylr}}{V_{is}}\right) \tag{2.14}$$

Donde $V_{av}$ es el ángulo vertical de visión de la cámara, $V_{is}$ es la altura de la imagen en píxeles, $SP_{ylr}$ es el valor $y$ de las coordenadas del punto en la superficie. La Figura 2.7 muestra la vista lateral del sistema de visión estéreo donde se pueden apreciar el ángulo vertical de visión y el ángulo $\beta$.

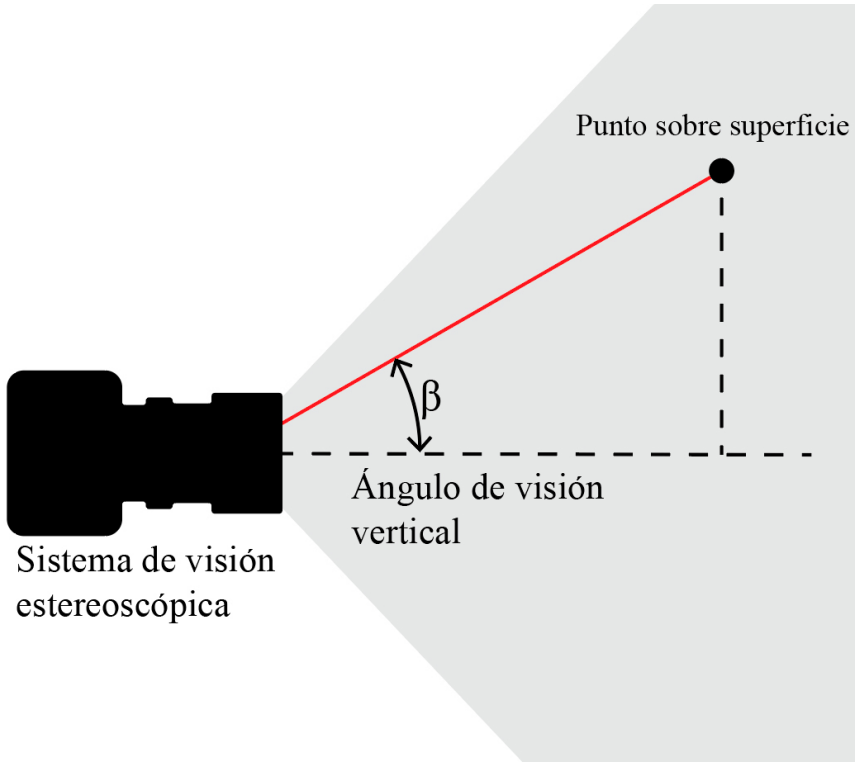



FIGURA 2.7: Ángulo de visión vertical en sistema de visión estereoscópica (vista lateral).

## 2.2. Detección de objetos

La detección de objetos, en el ámbito de la visión por computadora, desempeña una función importante al identificar ejemplares de objetos visuales pertenecientes a diversas clases (como seres humanos, animales, automóviles, entre otros) en imágenes digitales. Su principal propósito consiste en elaborar modelos informáticos y metodologías que suministren una de las nociones fundamentales requeridas en las aplicaciones de la visión por computadora: ¿cuáles objetos están presentes en la escena? Dentro de este contexto, las dos medidas más relevantes para evaluar el rendimiento en la detección de objetos son la precisión (que involucra tanto la clasificación como la localización de los objetos) y la velocidad Zou et al., 2023.

La detección de objetos constituye el fundamento sobre el cual se sustentan numerosas otras tareas en el campo de la visión por computadora, abarcando desde la segmentación de objetos (Hariharan et al., 2014, Hariharan et al., 2015, Dai, He y Sun, 2016 y He et al., 2017), hasta la generación de descripciones textuales de imágenes (Karpathy y Fei-Fei, 2015, Xu et al., 2015 y Wu et al., 2017), pasando por el seguimiento de objetos (Kang et al., 2017), entre otras aplicaciones. La detección de objetos es un pilar esencial

en una amplia gama de aplicaciones del mundo real, incluyendo la navegación autónoma, la visión de robots, la videovigilancia, entre otras. La figura 2.8 muestra un ejemplo de detección de objetos utilizando el algoritmo Mask R-CNN de He et al., 2017.


FIGURA 2.8: Tres objetos de interés (Automóvil, bicicleta y persona) detectados utilizando Mask R-CNN He et al., 2017.

Aunque las distintas tareas de detección conllevan objetivos y limitaciones particulares, se enfrentan a una serie de dificultades que varían de una tarea a otra. Además de los desafíos comunes que se encuentran en otras áreas de la visión por computadora, como la detección de objetos desde diferentes ángulos, variaciones en la iluminación y diferencias intraclase, los obstáculos en la detección de objetos abarcan aspectos tales como la rotación de los objetos, variaciones en la escala, precisión en la detección, localización de objetos densamente agrupados y objetos ocluidos, así como la velocidad de detección, entre otros.

En 2012, se experimentó un resurgimiento de las CNNs Krizhevsky, Sutskever e Hinton, 2012. Dado que estas redes profundas tienen la capacidad de aprender representaciones sólidas y de alto nivel en imágenes, su implementación en algoritmos de detección de objetos se convirtió en una progresión

natural. Girshick et al., 2015 introdujo la idea de Regiones con Redes Neuronales Convolucionales (RCNN en inglés). A partir de ese momento, la detección de objetos comenzó a evolucionar a un ritmo más acelerado. En la era del aprendizaje profundo, existen dos grupos de detectores: Los "detectores de dos etapas" y los "detectores de una etapa". El primer enfoque considera la detección como un proceso que va desde lo general a lo específico, mientras que el segundo aborda la detección como un proceso que se realiza de manera integral en un solo paso.

### 2.2.1. Detectores de dos etapas

#### RCNN

La idea detrás de RCNN es sencilla. Se inicia con la obtención de un conjunto de sugerencias de objetos (cajas que contienen candidatos a objetos) a través de una búsqueda selectiva Uijlings et al., 2013. Después, cada sugerencia se ajusta a un tamaño de imagen fijo y se introduce en una CNN previamente entrenada en ImageNet para extraer características. Finalmente, se emplea un clasificador lineal de máquina de soporte vectorial para determinar si un objeto está presente en la región y para identificar la categoría del objeto. En la figura 2.9 se puede observar el procedimiento de entrenamiento. RCNN tiene un mAP (mean average precisión por sus siglas en inglés) de 58.5 % A pesar de los notables avances logrados por RCNN, aún presenta algunas desventajas, como la redundancia en los cálculos de características en múltiples sugerencias superpuestas, lo que resulta en una detección excepcionalmente lenta (alrededor de 14 segundos por imagen cuando se trabaja en una GPU).

#### SPPNet

Luego de RCNN, en el mismo año, se introdujo SPPNet como una solución a este problema. He et al., 2015 presentó SPPNet, cuyas siglas significan Spatial Pyramid Pooling Networks en inglés. Los modelos previos de CNN requerían un tamaño de imagen constante. La principal innovación de SPPNet radica en la incorporación de una capa de SPP (Spatial Pyramid Pooling en inglés), que permite que una CNN genere representaciones de longitud fija sin importar el tamaño de la imagen o la región de interés, sin necesidad de redimensionarla. Al emplear SPPNet para la detección de objetos, los mapas de características pueden calcularse de una sola vez en toda la imagen, y posteriormente se pueden generar representaciones de longitud fija

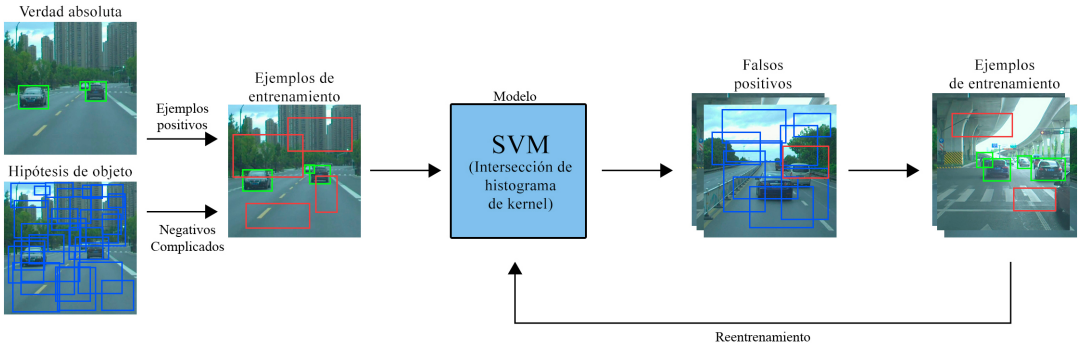



FIGURA 2.9: Procedimiento de entrenamiento para el reconocimiento de objetos de RCNN. (Las siglas SVM del inglés Support Vector Machine)

para regiones arbitrarias con el propósito de entrenar detectores, lo que evita cálculos repetitivos de características convolucionales. SPPNet es aproximadamente 20 veces más rápido que RCNN sin comprometer la precisión en la detección que llega hasta el 59.2 % mAP. A pesar de que SPPNet mejoró la velocidad de detección, aún presenta algunas desventajas, como la necesidad de un entrenamiento en múltiples etapas y el ajuste fino solo se aplica a sus capas completamente conectadas, mientras que simplemente se omiten las capas anteriores.

### Fast RCNN

Como respuesta a la problemática de SPPNet, Girshick, 2015 propuso Fast R-CNN, que es una mejora de R-CNN y SPPNet. Fast R-CNN permite entrenar simultáneamente un detector y un regresor de cuadro delimitador bajo la misma configuración de la red. La arquitectura de Fast RCNN se puede observar en la figura 2.10. Esta mejora incrementó el mAP hasta el 70.0 %, con una velocidad de detección hasta 200 veces más rápida que R-CNN. Fast R-CNN integra con éxito las ventajas de R-CNN y SPPNet, sin embargo, su velocidad de detección sigue estando limitada por el proceso de generación de propuestas.

### Faster RCNN

Más tarde, en 2015, Ren et al., 2015 propuso Faster R-CNN, el primer detector de objetos basado en aprendizaje profundo que logró funcionar casi

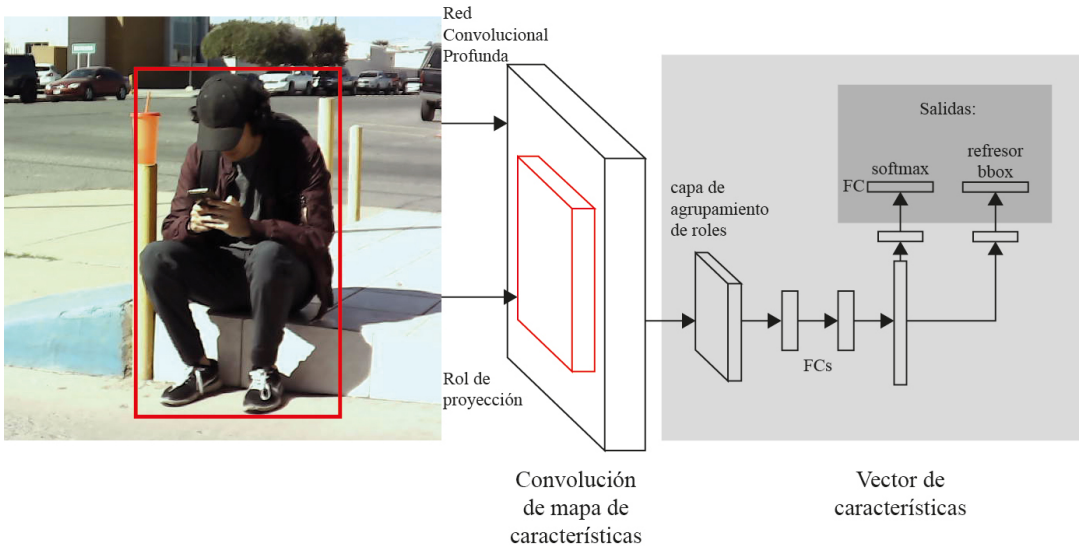



FIGURA 2.10: Arquitectura de Fast RCNN donde una imagen de entrada y múltiples regiones de interés se introducen en una red totalmente convolucional

en tiempo real, alcanzando velocidades de hasta 17 fps y un mAP del 73.2 % usando ZF-Net Zeiler y Fergus, 2014. La contribución fundamental de Faster R-CNN radica en la introducción de la RPN (Red de Propuesta de Región en inglés), que permite generar propuestas de regiones prácticamente sin costo adicional. En el camino de R-CNN a Faster R-CNN, la mayoría de los componentes individuales de un sistema de detección de objetos, como la generación de propuestas, la extracción de características, la regresión del cuadro delimitador, entre otros, se han integrado de manera gradual en un sistema unificado de aprendizaje de extremo a extremo. Aunque Faster R-CNN superó la limitación de velocidad de Fast R-CNN, todavía existen cálculos redundantes en la etapa de detección posterior. Mas adelante, se propusieron diversas mejoras, incluyendo RFCN Dai et al., 2016 y Light-Head R-CNN Li et al., 2017.

### Feature Pyramid Networks (FPN)

Las Feature Pyramid Networks (FPN), también conocidas como redes piramidales de características, fueron propuestas por Lin et al., 2017a. Antes de la introducción de FPN, la mayoría de los detectores basados en aprendizaje profundo se centraban únicamente en los mapas de características de la capa superior de las redes. Si bien las características de las capas más profundas en una CNN son beneficiosas para el reconocimiento de categorías, no son

tan efectivas para la localización precisa de objetos. Por lo tanto, FPN cuenta con una arquitectura que funciona de arriba hacia abajo y utiliza conexiones laterales para generar información semántica de alto nivel en todas las escalas. En la figura 2.11 se puede observar la arquitectura de FPN. Dado que una CNN naturalmente forma una pirámide de características a medida que se propaga hacia adelante, FPN ha demostrado ser una solución efectiva para la detección de objetos a diversas escalas. Cuando se incorpora FPN en un sistema Faster R-CNN estándar, logra resultados de detección de vanguardia con un solo modelo. Hoy en día, FPN se ha convertido en una característica fundamental en muchos de los detectores más avanzados.

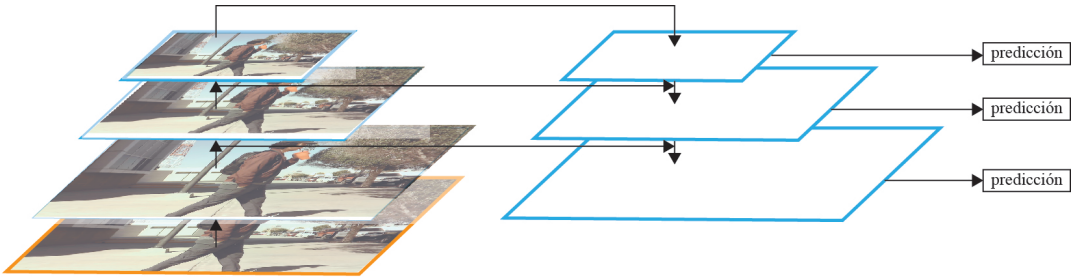



FIGURA 2.11: Arquitectura de la red piramidal de características con predicciones independientes en cada nivel.

### 2.2.2. Detectores de una etapa

La mayoría de los detectores de dos etapas siguen un enfoque que va desde un procesamiento de alto nivel hacia uno de nivel más bajo. El alto nivel se concentra en mejorar la capacidad de retención de información, mientras que el nivel más bajo refina la ubicación basándose en la detección de alto nivel y se centra en la capacidad de discriminación. Aunque pueden lograr una alta precisión, rara vez se implementan en ingeniería debido a su baja velocidad de procesamiento y gran complejidad. Por otro lado, los detectores de una sola etapa pueden recuperar todos los objetos en una sola etapa de inferencia. Son muy apreciados en dispositivos móviles con aplicaciones en tiempo real y son fáciles de implementar, pero su rendimiento se ve notoriamente afectado al detectar objetos densos y pequeños.

#### You Only Look Once (YOLO)

El algoritmo "Sólo se mira una vez" ampliamente conocido como YOLO (acrónimo de "You Only Look Once" en inglés), fue presentado por Redmon

et al., 2016. Se trata del primer detector de una sola etapa en la era del aprendizaje profundo. YOLO destaca por su excepcional velocidad; una versión rápida de YOLO logra operar a 155 fps con un mAP del 52.7 %, mientras que una versión mejorada alcanza 45 fps con un mAP del 63.4 %. YOLO sigue un enfoque completamente diferente en comparación con los detectores de dos etapas. Aplica una única red neuronal a la imagen completa, la cual segmenta la imagen en regiones y predice simultáneamente los cuadros delimitadores y las probabilidades para cada región. En la figura 2.12 se puede apreciar el funcionamiento del sistema YOLO. A pesar de su considerable mejora en el tiempo de detección, YOLO experimenta una reducción en la precisión de la localización en comparación con los detectores de dos etapas, particularmente al identificar objetos pequeños. Versiones posteriores de YOLO, como las mencionadas en Redmon y Farhadi, 2018, Bochkovskiy, Wang y Liao, 2020, Redmon y Farhadi, 2017, junto con SSD (Liu et al., 2016), se enfocaron en abordar este problema. Actualmente existen diversas versiones de YOLO que intentan resolver esta misma problemática, como YOLOv7 desarrollado por Wang, Bochkovskiy y Liao, 2023 y YOLOv8 Terven y Cordova-Esparza, 2023, las cuales superan a la mayoría de los detectores de objetos en términos de velocidad y precisión al introducir estructuras optimizadas, como la asignación dinámica de etiquetas y la reparametrización de la estructura del modelo.

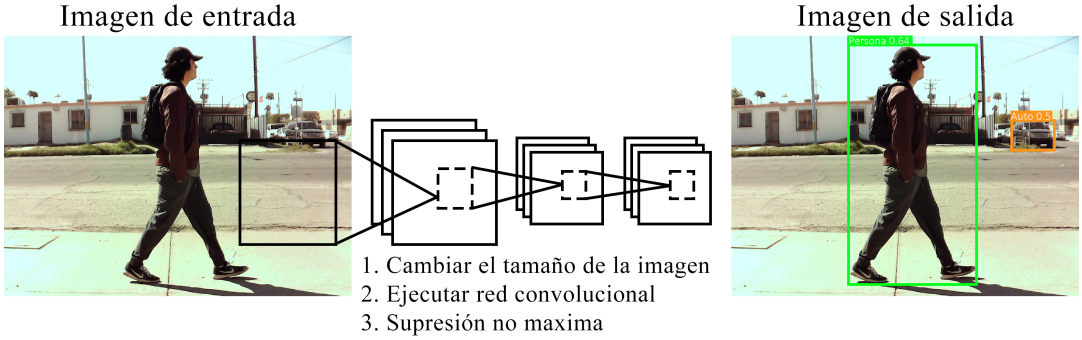


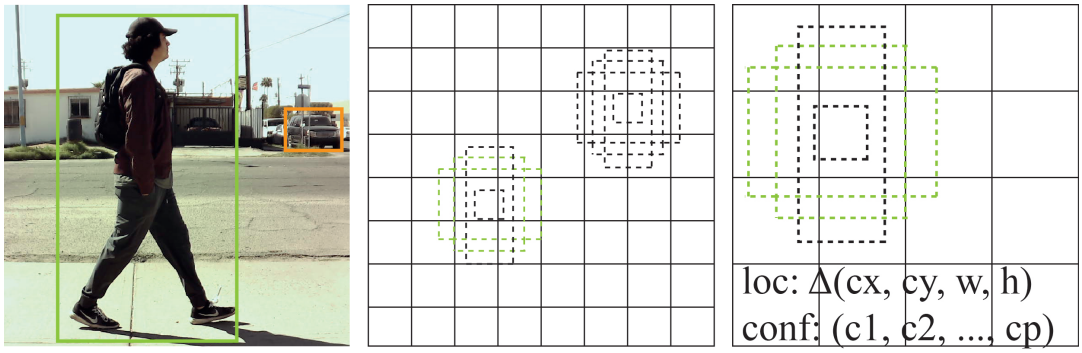

FIGURA 2.12: El sistema de detección YOLO. (1) el sistema ajusta el tamaño de la imagen a 448x448, (2) ejecuta solo una red neuronal convolucional en la imagen y (3) establece umbrales a las detecciones resultantes por la confianza del modelo.

### Single Shot Multibox Detector (SSD)

El detector multicaja de disparo único o SSD (acrónimo de Single Shot Multibox Detector en inglés) fue presentado por Liu et al., 2016. Su contribución principal radica en la implementación de técnicas de referencia múltiple y multirresolución, lo cual resulta en una mejora sustancial de la precisión de detección, especialmente en el caso de objetos pequeños. En la figura 2.13 se puede observar un ejemplo del marco de SSD. (a) SSD solo necesita una imagen de entrada y cajas de verdad absoluta para cada objeto durante el entrenamiento. De forma convolucional, se evalúa un pequeño conjunto (por ejemplo, 4) de cuadros predeterminados de diferentes relaciones de aspecto en cada ubicación en varios mapas de características con diferentes escalas (por ejemplo, $8 \times 8$ y $4 \times 4$ en (b) y (c)). Para cada cuadro predeterminado, se predice tanto los desplazamientos de forma como las confianzas para todas las categorías de objetos (c1, c2, $\cdots$, cp). En el momento del entrenamiento, primero se hace coincidir estos cuadros predeterminados con los cuadros de verdad absoluta. Por ejemplo, se relacionan dos casillas predeterminadas con la persona y una con el automóvil, que se tratan como positivas y el resto como negativas. La pérdida del modelo es una suma ponderada entre la pérdida de localización (por ejemplo, Smooth L1 Girshick, 2015) y la pérdida de confianza (por ejemplo, Softmax). SSD destaca por su capacidad para lograr un equilibrio entre precisión y velocidad, obteniendo un mAP del 46.6 % en su versión rápida, la cual opera a 59 fps. La diferencia fundamental entre SSD y sus predecesores radica en su capacidad para detectar objetos de diversas escalas en múltiples capas de la red, en contraposición a las versiones anteriores que solo realizaban detección en las capas superiores.

### RetinaNet

A pesar de su alta velocidad y simplicidad, los detectores de una etapa han estado rezagados en términos de precisión en comparación con los detectores de dos etapas durante muchos años. Lin et al., 2017b han presentado RetinaNet, abordando un problema fundamental que radica en el desequilibrio extremo de clases entre objetos en primer plano y el fondo que se encuentra comúnmente durante el entrenamiento de detectores de objetos densos. Para solucionar esto, RetinaNet introdujo una nueva función de pérdida denominada "pérdida focal", que remodela la pérdida de entropía cruzada estándar, de manera que el detector se enfoque más en ejemplos difíciles y mal clasificados durante el proceso de entrenamiento. Gracias a la pérdida




a) Imagen con cajas de verdad absoluta

b) Mapa de características de 8x8

c) Mapa de características de 4x4

FIGURA 2.13: Marco SSD. a) imagen de ejemplo con cajas de verdad absoluta. b) representación de mapa de características de la imagen en un tamaño de 8x8. c) representación de mapa de características de la imagen en un tamaño de 4x4.

focal, los detectores de una etapa pueden alcanzar una precisión comparable a la de los detectores de dos etapas, manteniendo una velocidad de detección muy alta y obteniendo un mAP del 59.1 %.

**CornerNet**

Los enfoques previos se basaban principalmente en el uso de cuadros de anclaje para proporcionar puntos de referencia en la clasificación y regresión de objetos. Dado que los objetos a menudo presentan variaciones en cuanto al número, posición, tamaño, relación, entre otros aspectos, se requería la configuración de numerosos cuadros de referencia para que se ajustaran de manera óptima a las verdades de terreno, con el fin de lograr un alto rendimiento. Sin embargo, este enfoque conllevaba desafíos, como un desequilibrio pronunciado en las categorías, la necesidad de ajustar numerosos hiperparámetros de forma manual y un largo tiempo de convergencia. Para abordar estos problemas, Law y Deng, 2018 adoptaron un enfoque diferente en el que la detección se concibe como un problema de predicción de puntos clave (las esquinas de un cuadro delimitador). Una vez que se obtienen estos puntos clave, se procede a desvincularlos y reagruparlos utilizando información de incrustación adicional para formar los cuadros delimitadores (ver 2.14). Se detecta un objeto como un par de esquinas de un cuadro delimitador agrupadas. Una red convolucional genera un mapa de calor para todas las esquinas superiores izquierdas, un mapa de calor para todas las esquinas inferiores derechas y un vector de incrustación para cada esquina

detectada. La red está entrenada para predecir incrustaciones similares para esquinas que pertenecen al mismo objeto. CornerNet superó a la mayoría de los detectores de una etapa en ese momento, logrando un mAP del 57.8 %.

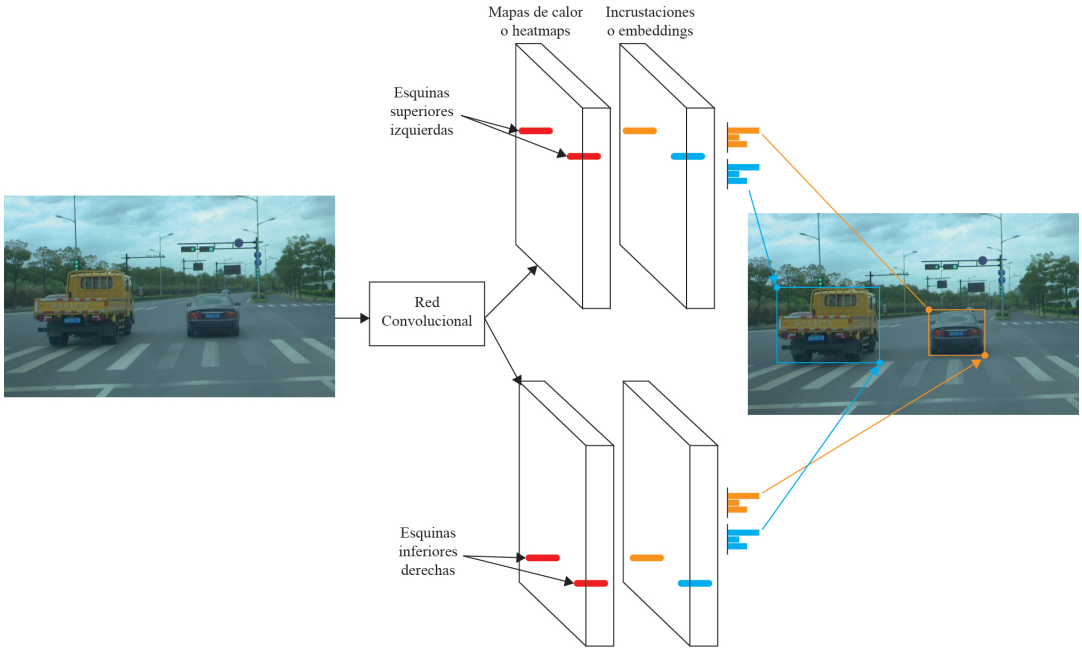



FIGURA 2.14: Marco de CornerNet con red convolucional, mapas de calor e incrustaciones.

### CenterNet

Zhou, Wang y Krähenbühl, 2019 introdujeron CenterNet, el cual sigue un enfoque de detección basado en puntos clave, pero simplifica los procesos posteriores costosos, como la asignación de puntos clave basada en grupos (como se hacía en CornerNet Law y Deng, 2018, ExtremeNet Zhou, Zhuo y Krahenbuhl, 2019, entre otros). Esto da como resultado una red de detección que opera de manera totalmente integral de extremo a extremo. La perspectiva de CenterNet considera que un objeto se representa como un solo punto, que es el centro del objeto, y realiza una regresión de todos sus atributos (como tamaño, orientación, ubicación, postura, entre otros) con respecto al punto central de referencia como se muestra en la figura 2.15. El modelo es notablemente sencillo y elegante, pudiendo abordar diversas tareas en un solo marco, como la detección de objetos tridimensionales, la estimación de la postura humana, el aprendizaje de flujo óptico, la estimación de profundidad, entre otras. A pesar de esta conceptualización de la detección, CenterNet también logra resultados de detección comparables con un mAP del 61.1 %.

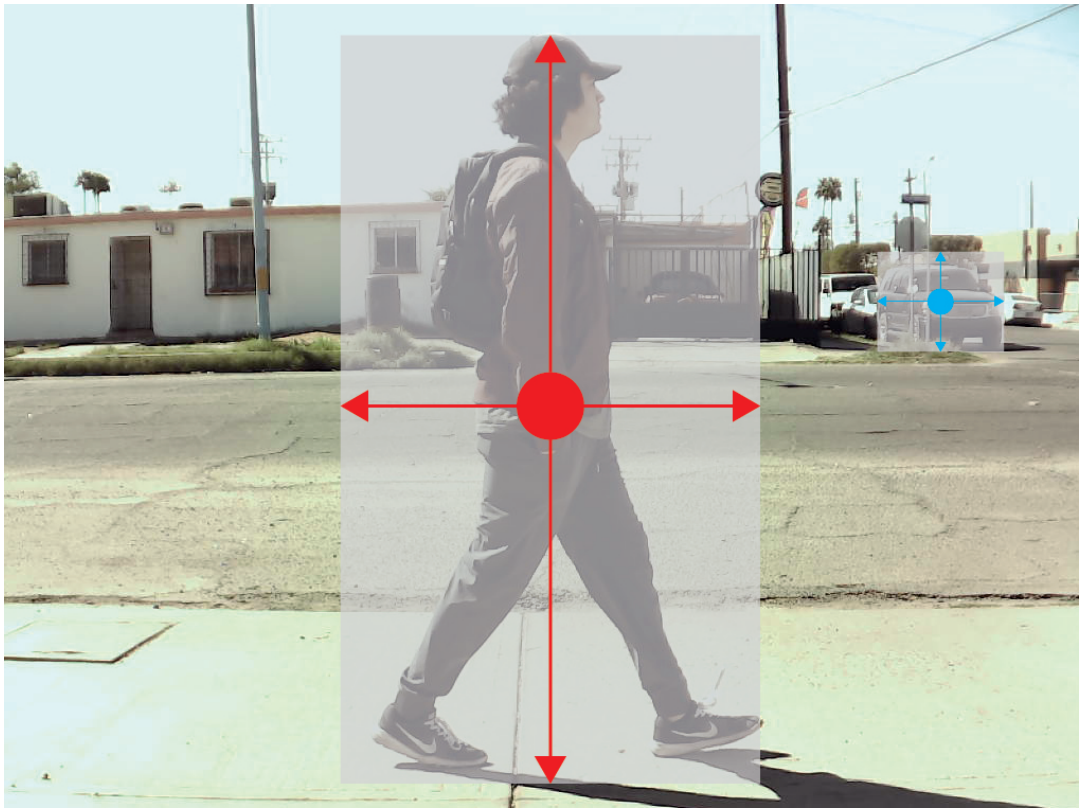


FIGURA 2.15: Se modela un objeto como el punto central de su cuadro delimitador. El tamaño del cuadro delimitador y otras propiedades del objeto se infieren a partir de la característica del punto clave en el centro.

**DETR**

En los últimos años, los Transformadores han tenido un impacto significativo en el campo del aprendizaje profundo, particularmente en la visión por computadora. Estos modelos han reemplazado al operador de convolución tradicional con el cálculo de atención exclusiva, superando así las limitaciones de las CNN y permitiendo un campo receptivo a escala global. En su trabajo, Carion et al., 2020 presentaron DETR, donde concibieron la detección de objetos como un problema de predicción establecido y propusieron una red de detección integral con Transformers. Este enfoque ha marcado una nueva era en la detección de objetos, en la que los objetos pueden detectarse sin la necesidad de utilizar cajas o puntos de anclaje. Posteriormente, Zhu et al., 2020 propusieron DETR deformable para abordar los problemas de tiempo de convergencia prolongado de DETR y su rendimiento limitado en la detección de objetos pequeños. En DETR deformable, se utilizan módulos de atención deformables (multiescala) para almacenar y colocar los módulos de

atención del transformador que procesan mapas de características, como se muestra en la figura 2.16. Esta mejora logra un rendimiento líder en el campo con un mAP de 71.9 %.

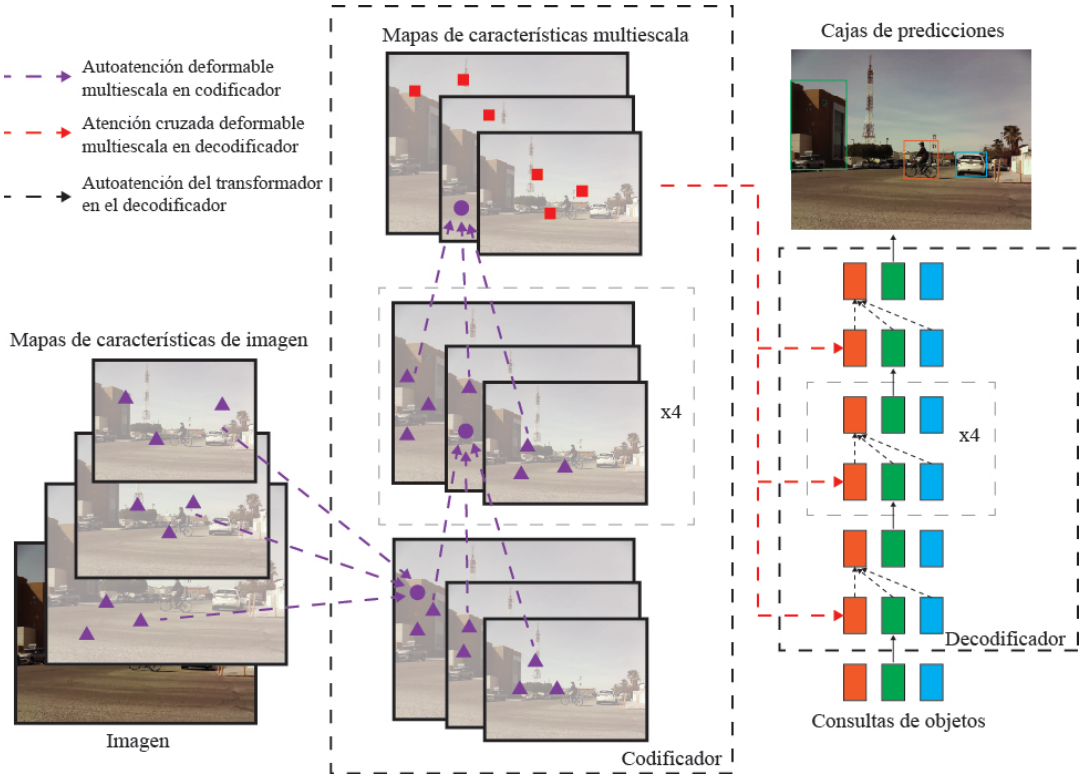



FIGURA 2.16: Ilustración del detector de objetos DETR deformable propuesto por Zhu et al., 2020

Por ende, la combinación de la visión estereoscópica con la detección de objetos representa un avance significativo en el campo de la navegación autónoma. Este capítulo ha proporcionado un análisis exhaustivo de las técnicas clave en ambos campos, destacando su importancia y aplicaciones en la navegación autónoma. Al comprender y aprovechar las capacidades de la visión estereoscópica para la percepción tridimensional del entorno y la detección precisa de objetos mediante algoritmos avanzados, se sientan las bases para el desarrollo de sistemas de navegación autónoma más seguros, eficientes y adaptables.

# Capítulo 3

# Procedimiento de Investigación

La visión estereoscópica, permite a las máquinas percibir la profundidad de los objetos en un entorno, ha desempeñado un papel fundamental en el avance de la visión artificial y la navegación autónoma. La posibilidad de capturar información tridimensional ha permitido una comprensión más precisa y detallada de los entornos, facilitando así la toma de decisiones más inteligentes y adaptativas. Además, la detección de objetos ha demostrado ser un componente esencial en el campo de la visión artificial, ya que permite identificar y localizar objetos específicos en imágenes. Esta capacidad es importante para una variedad de aplicaciones, desde la conducción autónoma hasta la robótica industrial, proporcionando a las máquinas la capacidad de interactuar de manera segura y efectiva con su entorno. Este capítulo se enfoca en el procedimiento de investigación detrás de tres contribuciones fundamentales presentadas en esta tesis. En primer lugar, se presenta un nuevo método para la coincidencia de plantillas que mejora significativamente la velocidad de la detección de objetos. Posteriormente, se aborda la cuestión de la calibración precisa en cuanto a los parámetros intrínsecos y extrínsecos de los sistemas de visión estereoscópica, presentando un enfoque novedoso que permite una mayor fiabilidad y exactitud en la percepción del entorno. Por último, se examina la integración de un algoritmo de detección de objetos con el sistema de visión estereoscópica, lo que brinda una solución más completa y robusta para la navegación autónoma en diversos escenarios. A lo largo de este capítulo, se detallará el procedimiento de investigación seguido para desarrollar y validar estas contribuciones.

## 3.1. SoRA, un nuevo método de coincidencia de plantillas

Entre los métodos de coincidencia de plantilla mencionados previamente en el marco teórico, SAD es de los mas mencionados en la literatura, particularmente en sistemas de visión estéreo, donde las imágenes izquierda y derecha muestran similitudes notables. Sin embargo, su principal inconveniente radica en su requerimiento de tiempo y su creciente tiempo de cálculo conforme aumenta el tamaño de la plantilla y la imagen objetivo. El tiempo de ejecución de una coincidencia de píxeles individual en un enfoque local puede variar dependiendo del algoritmo, y algunos de ellos pueden lograr tiempos más rápidos que los enfoques globales para un solo píxel. Dado que la fusión entre la detección de objetos y los sistemas de visión estéreo solo requiere una coincidencia de píxeles para cada objeto, un enfoque local puede tener un costo computacional menor y llevar menos tiempo para realizar la tarea de encontrar el objeto detectado en el par de imágenes estéreo. Por esta razón, esta tesis propone un nuevo algoritmo de coincidencia de plantillas optimizado para sistemas de visión estéreo y más rápido que los enfoques locales clásicos. El algoritmo propuesto tiene una complejidad algorítmica lineal $O(n)$, lo que se traduce en una reducción del tiempo de ejecución de coincidencia de patrones en comparación con otros algoritmos como SAD, que tiene una complejidad algorítmica cuadrática $O(n^2)$, lo que produce tiempos de ejecución más largos a medida que aumenta el tamaño del patrón. El algoritmo propuesto está diseñado para ser utilizado en conjunto con un algoritmo de detección de objetos en un sistema de visión estéreo para obtener información espacial sobre los objetos en el campo de visión. La siguiente lista contiene las contribuciones del algoritmo propuesto:

- Coincidencia de plantillas precisa. El algoritmo de coincidencia de plantillas propuesto tiene un error de coincidencia bajo que se puede comparar con algoritmos de coincidencia de plantillas robustos.
- Tiempos de ejecución rápidos. A diferencia de los algoritmos robustos con bajo error de coincidencia, nuestra propuesta funciona varias veces más rápido que otros métodos.
- Complejidad algorítmica lineal. El método propuesto mantiene tiempos de ejecución bajos incluso con plantillas más grandes.

- Bajo coste computacional. En comparación con otros algoritmos de coincidencia de plantillas, nuestra propuesta tiene un número reducido de operaciones matemáticas en comparación con otros algoritmos de coincidencia de plantillas y mantiene la misma precisión.

En esta tesis, proponemos SoRA, un algoritmo de coincidencia de imágenes binoculares que logra una alta precisión con un bajo costo computacional. Para evaluar el rendimiento de SoRA, lo comparamos con cuatro algoritmos de coincidencia de plantillas ampliamente utilizados: SAD, BBS, LDS y Fast-Match. A partir de un análisis de la función de cada algoritmo, se pueden derivar sus respectivas complejidades algorítmicas. SAD tiene una complejidad cuadrática de $O(N^2)$, mientras que BBS tiene una complejidad de $O(N^3)$. LDS tiene una complejidad de $O(N^2 \log(N))$. Fast-Match tiene una complejidad de $O(N^3 \log(N))$. En contraste, SoRA tiene una complejidad lineal de $O(N)$. Por lo tanto, SoRA ofrece ventajas significativas en cuanto a eficiencia computacional sobre otros algoritmos. Además, la simplicidad y el bajo costo computacional de SoRA lo convierten en una opción atractiva para aplicaciones en tiempo real y sistemas integrados con capacidad de procesamiento limitada. El rendimiento de estos algoritmos en términos de precisión y velocidad de ejecución se verifica aún más en los experimentos presentados en el capítulo cuatro.

El método propuesto en esta tesis se basa parcialmente en el algoritmo SAD y emplea una medida de disimilitud que busca reducir la carga computacional mediante la creación de un arreglo de relaciones para calcular este valor en tiempos de ejecución más eficientes. Al algoritmo presentado se le dio el nombre de SoRA, que proviene de la abreviatura en inglés de Subtraction of Relationship Array (Resta de Arreglo de Relaciones). Para el resto de este documento, consideramos que el par de imágenes estéreo posee una geometría epipolar en la que los planos de imagen son paralelos y están alineados en el eje horizontal. El diseño del algoritmo propuesto (SoRA) tiene como objetivo buscar la correspondencia directamente en la fila esperada de la imagen objetivo, que sería la misma fila de donde se extrajo la plantilla. En caso de necesitar realizar una búsqueda completa de la imagen con el algoritmo, se podrían realizar algunos ajustes menores, los cuales afectarían el tiempo de ejecución.

Como se mencionó anteriormente, la tarea del algoritmo SoRA es calcular la medida de disimilitud para todos los candidatos con la menor cantidad de operaciones, y consta de: Preparación de la plantilla, preparación de los candidatos, cálculo de la medida de disimilitud y selección de la coincidencia.

Para llevar a cabo estas tareas, el algoritmo se divide en ocho pasos, como se indica a continuación:

**Preparación de plantilla**

1. Media de columnas de plantilla.
2. Arreglo de relaciones de plantilla.

**Preparación de candidatos**

3. Media de las filas del candidato.
4. Extraer candidato.
5. Arreglo de relaciones de candidato.

**Cálculo de la medida de disimilitud**

6. Cálculo de la medida de disimilitud.
7. Repetir para el siguiente candidato.

**Selección de coincidencia**

8. Selección de coincidencia.

### 3.1.1. Búsqueda de plantilla fila esperada

La preparación de la plantilla lleva a cabo los pasos uno y dos, la preparación de los candidatos implica los pasos tres, cuatro y cinco, el cálculo de la medida de disimilitud comprende los pasos seis y siete, y la selección de coincidencia realiza el paso 8. La figura 3.1 muestra una representación gráfica de la plantilla, que es una matriz cuadrada de tamaño $n \times n$ con filas representadas por $i$ y columnas por $j$.

**Media de columnas de plantilla**

El primer paso consiste en calcular la media de las columnas ($\mu_T$) de la plantilla dada utilizando la ecuación 3.1. La Figura 3.2 muestra una representación gráfica de $\mu_T$, donde cada elemento representa la media de cada columna de $I_T$. Donde $I_T$ es la imagen de la plantilla y $n$ es el tamaño de la plantilla.

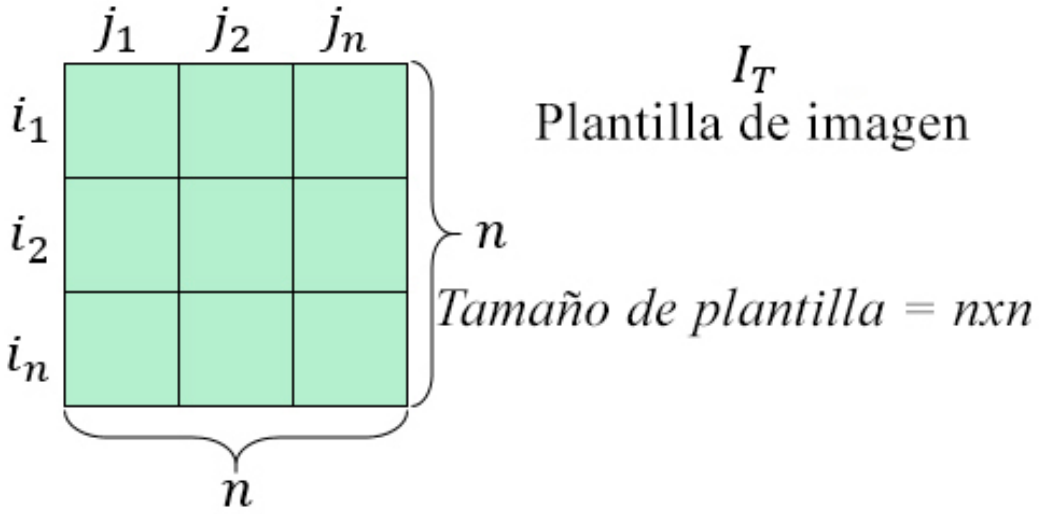



FIGURA 3.1: Representación matricial de la plantilla de la imagen.

$$\mu_{T,j} = \frac{\sum_{i=1}^{n} I_{T,(i,j)}}{n}; j = 1, 2, 3, ..., n \tag{3.1}$$


FIGURA 3.2: Arreglo de la media de columnas de la plantilla.

**Arreglo de relaciones de plantilla**

Una vez que se calcula $\mu_T$, se puede calcular el arreglo de relaciones de la plantilla ($T_{RA}$) utilizando las ecuaciones 3.2 y 3.3. $T_{RA}$ es un arreglo que establece la relación entre el primer elemento del arreglo y el resto de los elementos, luego almacena el valor del primer elemento de $\mu_T$ en el primer elemento de $T_{RA}$ para evitar errores de coincidencia cuando los candidatos tienen una relación similar pero con un valor de píxel diferente. Si este elemento no está incluido en el arreglo, el algoritmo tiende a hacer coincidir el objetivo con lugares de la imagen que generaron la misma relación entre el

primer elemento y el resto de los valores de píxeles. La representación gráfica de $T_{RA}$ se muestra en la Figura 3.3.

$$T_{RA,1} = \mu_{T,1} \tag{3.2}$$

$$T_{RA,j>1} = |\mu_{T,1} - \mu_T, j|; j = 2, 3, ..., n \tag{3.3}$$

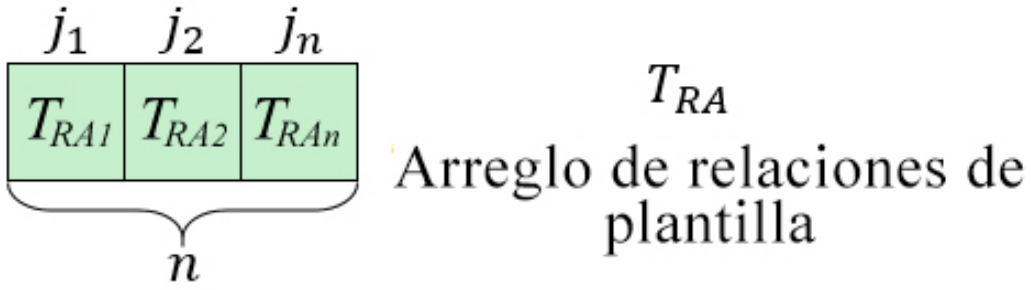



FIGURA 3.3: Arreglo de relaciones de plantilla.

**Media de las filas del candidato**

Una vez que se calcula $T_{RA}$, se finaliza la preparación de la plantilla. El siguiente paso es extraer las filas de candidatos ($I_{CR}$) de la imagen objetivo. $I_{CR}$ es una matriz que toma todas las columnas de la imagen objetivo y el mismo número de filas que $I_T$, dejando una matriz de tamaño $n \times w$, como se muestra en la Figura 3.4, donde $w$ es el ancho de la imagen.

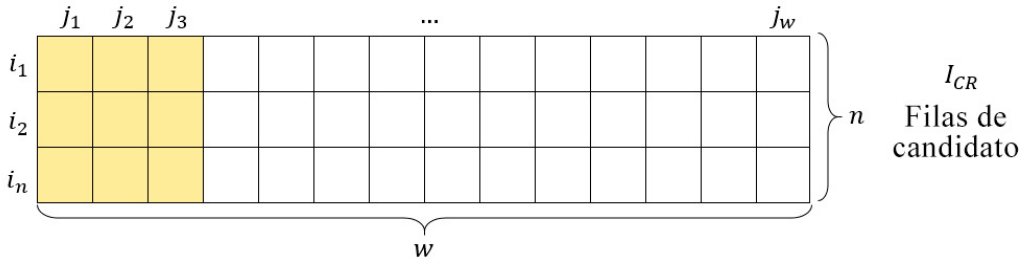



FIGURA 3.4: Filas candidato de la imagen objetivo.

Después de extraer $I_{CR}$, se debe calcular la media de las filas de candidatos ($\mu_{CR}$) utilizando la ecuación 3.4. La Figura 3.5 muestra una representación gráfica de $\mu_{CR}$ donde cada elemento representa la media de cada columna de $I_{CR}$. Donde $I_{CR}$ son las filas de candidatos y $w$ es el ancho de la imagen en píxeles.

$$\mu_{CR,j} = \frac{\sum_{i=1}^{n} I_{CR,(i,j)}}{n}; j = 1, 2, 3, ..., w \tag{3.4}$$

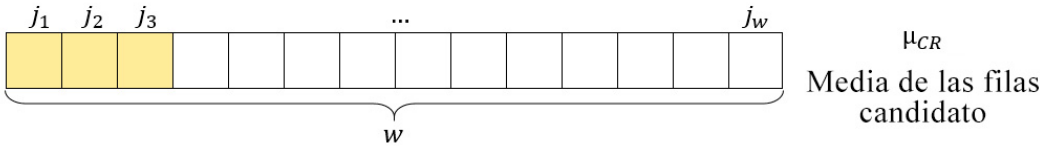



FIGURA 3.5: Filas candidato de la imagen objetivo.

Ahora, a partir de $\mu_{CR}$ se puede extraer un candidato de longitud $n$ ($\mu_C$) utilizando la ecuación 3.5. Este candidato se comparará con la plantilla después de cierta preparación que consiste en preparar el arreglo de relaciones de candidato para realizar menos operaciones. La Figura 3.6 muestra una representación gráfica de $\mu_C$ con un cuadrado amarillo que son los mismos cuadrados amarillos de la figura 3.5. Donde $m$ es el número de candidato que comienza en uno, este valor se incrementa en cada iteración del algoritmo hasta que se utilicen todos los candidatos para calcular la medida de disimilitud. El valor máximo de $m$ es $w - (n - 1)$.

$$\mu_C = \{\mu_{CR,m}, \mu_{CR,m+1}, ..., \mu_{CR,m+n}\} \tag{3.5}$$

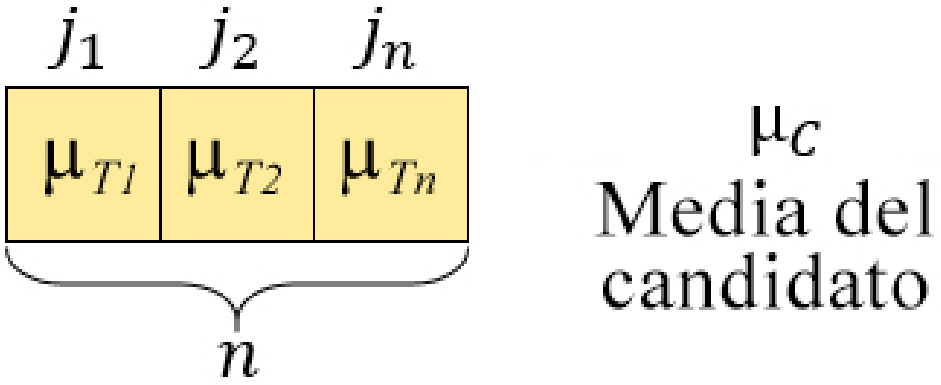



FIGURA 3.6: Arreglo de medias del candidato.

**Arreglo de relaciones de candidato**

Después, al igual que en la preparación de la plantilla, se debe calcular el arreglo de relaciones para $\mu_C$ utilizando las ecuaciones 3.6 y 3.7. La Figura

3.7 muestra una representación gráfica de $\mu_{CR}$.

$$C_{RA,1} = \mu_{C,1} \tag{3.6}$$

$$C_{RA,j>1} = |\mu_{C,1} - \mu_T, j|; j = 2, 3, ..., n \tag{3.7}$$

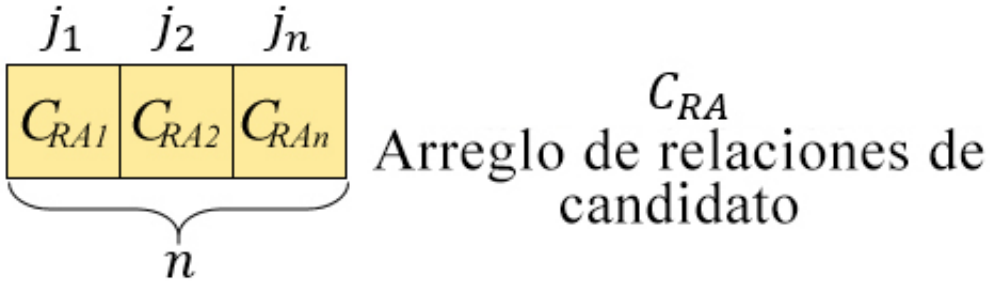



FIGURA 3.7: Arreglo de relaciones del candidato.

**Cálculo de la medida de disimilitud**

Una vez que se calculan $T_{RA}$ y $C_{RA}$, pueden compararse para obtener la medida de disimilitud ($dis_m$) utilizando la ecuación 3.8.

$$dis_m = \sum |T_{RA} - C_{RA}| \tag{3.8}$$

**Repetir para el siguiente candidato**

Una vez que se ha calculado $dis_m$, el proceso se repite a partir del paso cuatro (Extraer candidato) con $m = m + 1$ hasta que se calculen todas las medidas de disimilitud para los posibles candidatos en $\mu CR$, comenzando con el primer candidato en la posición uno y terminando con el último candidato en la posición $w - (n - 1)$.

**Selección de coincidencia**

Después se selecciona la coincidencia con la menor medida de disimilitud como se muestra en la ecuación 3.9. Las coordenadas de la coincidencia serán el número de fila de donde se extrajo la plantilla de la imagen fuente y, para el valor de columna, se tomará el valor calculado de *match*. La Figura 3.8 muestra una representación gráfica de *dis*, que contiene todas las medidas de

disimilitud calculadas, y el cuadrado verde representa el valor más bajo en $dis_m$, y la posición de este elemento se utiliza para calcular el *match*.

$$match = \text{mín}\,(dis_m) + \frac{n}{2} \tag{3.9}$$

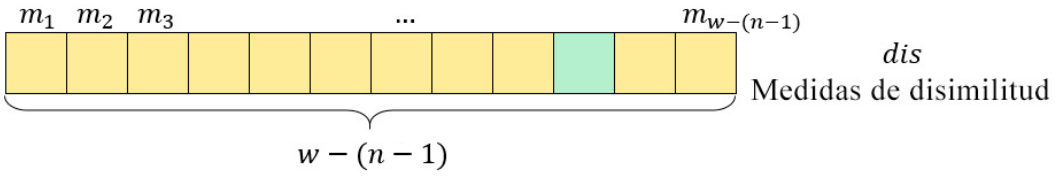



FIGURA 3.8: Representación gráfica del arreglo de medidas de disimilitud.

Se debe tener en cuenta que el cálculo de la medida de disimilitud es similar al algoritmo SAD, la principal diferencia radica en que el algoritmo SAD resta dos matrices y el algoritmo SoRA resta los arreglos de relaciones, que tienen menos elementos para calcular, lo que resulta en un cálculo más rápido de la medida de disimilitud. Estos tiempos de ejecución más rápidos se pueden observar en los resultados de los experimentos.

### 3.1.2. Búsqueda de plantilla en imagen completa

El algoritmo SoRA está diseñado para trabajar con imágenes estéreo, por lo tanto, las filas candidatas seleccionadas deben extraerse en la misma posición de fila que la imagen de la plantilla extraída de la imagen izquierda. Si se desea ejecutar una búsqueda de imagen completa con este algoritmo, se puede realizar repitiendo el algoritmo a partir del paso 3 y seleccionando filas diferentes cada vez hasta que se utilicen todas las filas para calcular la medida de disimilitud para cada candidato posible en la imagen objetivo, como se muestra en el diagrama de bloques de la figura 3.9.

La preparación de la plantilla incluye el cálculo de $\mu_T$ y $T_{RA}$, extraer filas candidatas se refiere a las primeras $n$ filas como $I_{CR}$, después se calcula $\mu_{CR}$ y se extrae el primer candidato $\mu_C$, que es igual a los primeros $n$ elementos de la media de las filas candidatas. Posteriormente, se prepara el candidato calculando $C_{RA}$ y luego $dis_m$. Si hay más candidatos en $\mu_{CR}$, se extrae el siguiente candidato; si no hay más candidatos, el algoritmo selecciona el siguiente $I_{CR}$ moviéndose un píxel en el eje vertical de la imagen objetivo y seleccionando las siguientes $n$ filas. Cuando se calcula $dis_m$ para todas las filas candidatas, se selecciona la coincidencia y el algoritmo se detiene. Un

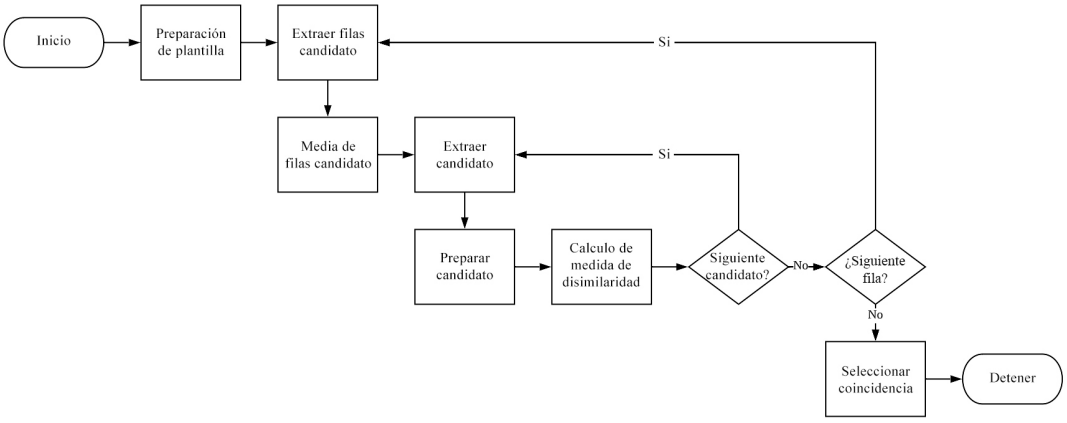



FIGURA 3.9: Diagrama de bloques para búsqueda de plantilla en imagen completa.

ejemplo de selección de coincidencias se muestra en la figura 3.10 con un mapa de disimilitud. La plantilla se extrae de la imagen izquierda y la coincidencia se encuentra en la imagen derecha (Objetivo). El mapa de disimilitud muestra una representación gráfica de $dis_m$, donde los tonos azul oscuro son los valores con mayor disimilitud y los tonos amarillos claros son los valores con menor disimilitud. Finalmente, el mapa de disimilitud ampliado muestra el valor con la disimilitud más baja que se selecciona como la coincidencia para la plantilla dada.

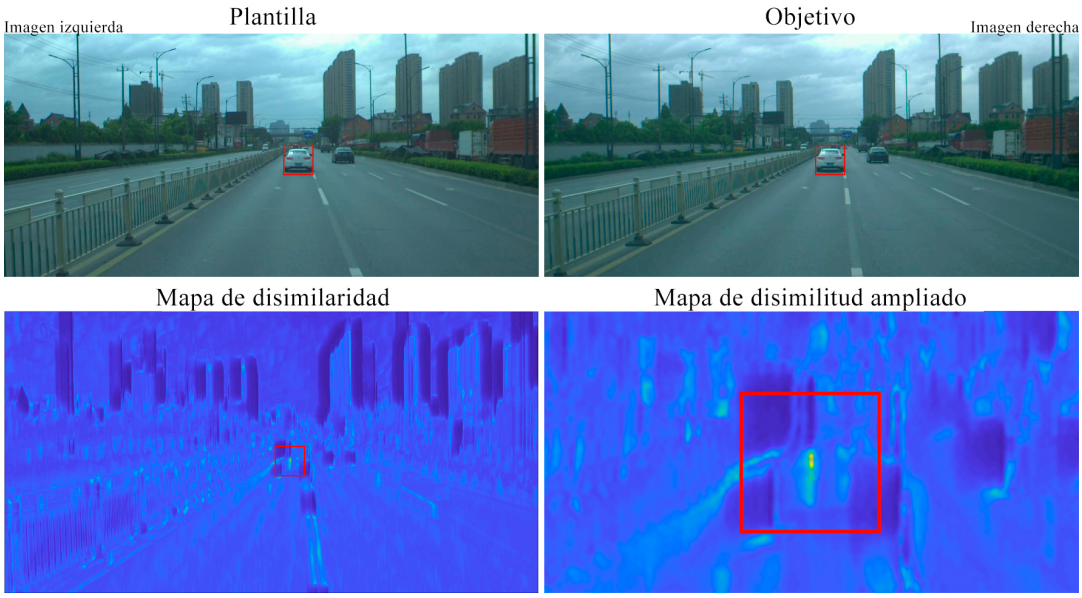



FIGURA 3.10: Mapa de disimilitud generado al buscar el objeto de la imagen izquierda (cuadro rojo) en la imagen derecha.

### 3.1.3. Implementación de SoRA en LabVIEW

Para comprobar el funcionamiento de SoRA y compararlo con otros algoritmos se llevó a cabo la implementación de estos en LabVIEW. En la figura 3.11 se puede observar la interfaz de usuario de este programa. Este cuenta con dos recuadros para mostrar la imagen izquierda y la imagen derecha. La imagen izquierda sería la imagen fuente, de donde se extrae la plantilla (en la figura se puede observar en un cuadro rojo la plantilla seleccionada) y la imagen derecha es la imagen en donde se busca esta plantilla, donde se puede observar en un cuadro rojo la plantilla seleccionada como la de mayor similitud. Del lado izquierdo se encuentran botones e indicadores para manipular el programa y observar resultados. El botón “IMPORT” permite seleccionar las imágenes que se utilizarán para aplicar el algoritmo, el botón “READY” inicia la ejecución del algoritmo para coincidencia de plantilla seleccionado, el control “Box size” permite modificar el tamaño de la plantilla, en este caso tiene un tamaño de $20 \times 20$, el control “Algorithm” permite elegir el algoritmo que se utilizará para la búsqueda de plantilla (SoRA, SAD y LDS), el control “ROI or Coords” permite cambiar entre las formas de seleccionar la plantilla origen, si el control está en “Off” la selección de la plantilla se hace seleccionando un punto sobre la imagen con el mouse y si el control está en “On” la selección de la plantilla se realiza introduciendo las coordenadas del punto en los controles “X” y “Y” que están abajo de este control. Después, abajo de estos controles se encuentra el indicador "Match"que muestra las coordenadas del punto de coincidencia en la imagen derecha, el tiempo de ejecución del algoritmo se muestra en la parte inferior en el indicador "Match Time (ms)". Por último, se muestra el botón "STOP", que se utiliza para detener el programa.

Este programa fue desarrollado utilizando una arquitectura de programación de máquina de estados basada en eventos como la que se muestra en la figura 3.12. El primer estado, "Initialize", se encarga de inicializar las variables del programa para pasar al estado "Wait for event", que simplemente espera una acción del usuario para ir a otro evento. El estado "Import" se activa cuando el usuario presiona el botón "Import", y es el que importa las imágenes que se utilizarán para aplicar el algoritmo seleccionado. El estado "ROI" se activa al presionar el botón "Ready", y su función es preparar la plantilla de la imagen izquierda para luego pasar al estado "Match", donde se aplica el algoritmo de coincidencia de plantillas seleccionado para buscar su correspondencia en la imagen derecha. Finalmente, el estado "Stop"se activa al presionar el botón "STOP" y su función es detener el programa.


FIGURA 3.11: Interfaz de usuario en LabVIEW para implementar SoRA, SAD y LDS.

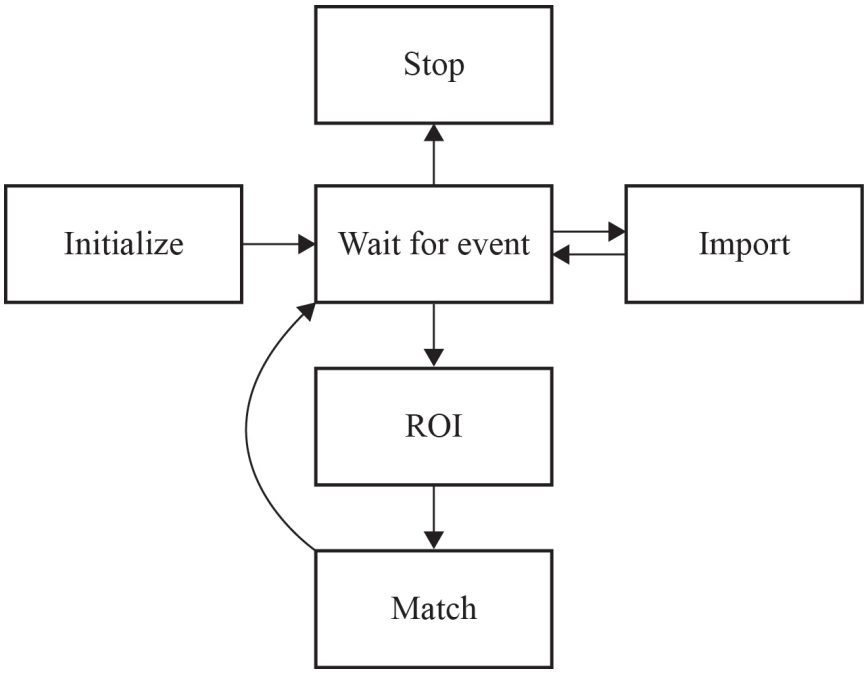



FIGURA 3.12: Diagrama de la máquina de estados basada en eventos para el programa desarrollado.

### Initialize

En la figura 3.13 se puede observar el diagrama de bloques para el estado initialize, el cual tiene la función de reiniciar los controles e indicadores a sus valores predeterminados y mandar el programa al estado Wait for event.

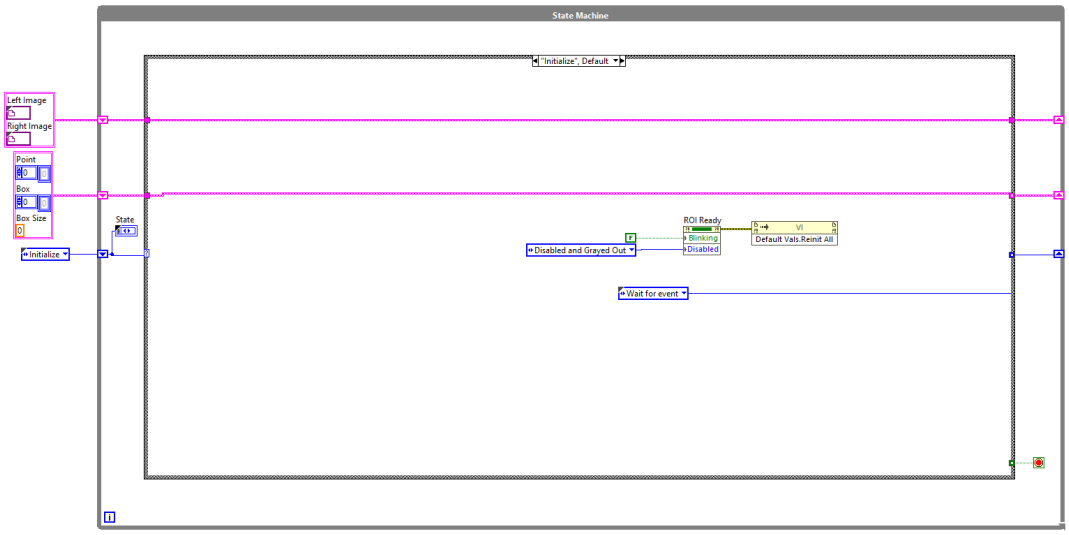



FIGURA 3.13: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Initialize

### Wait for event

En la figura 3.14 se puede observar el diagrama de bloques para el estado Wait for event, este tiene la función de esperar a que el usuario realice alguna acción para cambiarse al estado correspondiente, el cual puede ser Stop, Import o ROI.

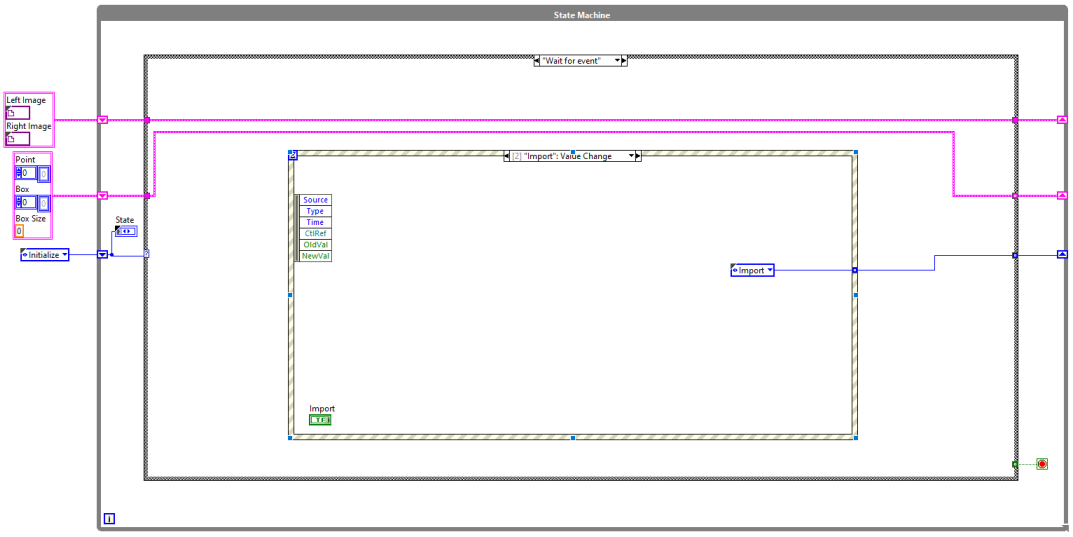



FIGURA 3.14: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Wait for event

### Stop

En la figura 3.15 se puede observar el diagrama de bloques para el estado Stop, el cual tiene la función de detener el programa.

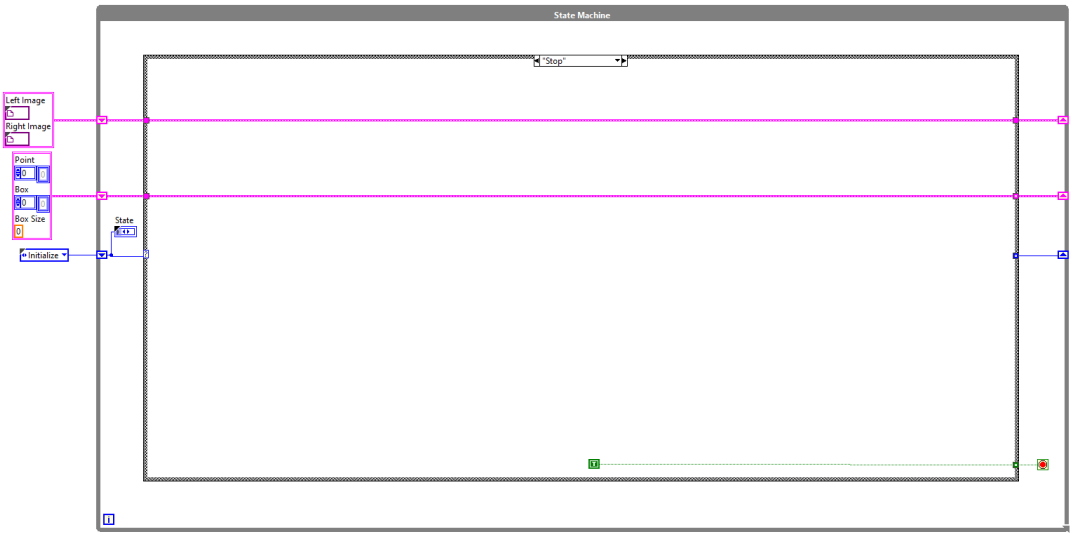



FIGURA 3.15: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Stop

### Import

En la figura 3.16 se puede observar el diagrama de bloques para el estado Import, el cual tiene la función de importar las imágenes que se utilizaran para implementar el método de coincidencia de plantillas seleccionado. Al finalizar este estado, regresa al estado Wait for event para esperar alguna acción por parte del usuario.

### ROI

En la figura 3.17 se puede observar el diagrama de bloques para el estado ROI, el cual tiene la función de extraer la plantilla de la imagen izquierda que se va a utilizar de acuerdo al tamaño y las coordenadas seleccionadas. Si el control ROI or Coords está en posición True las coordenadas a utilizar serán las de los controles X y Y, si el control está en False, las coordenadas a utilizar serán las coordenadas del punto seleccionado manualmente sobre la imagen. La selección manual se realiza dando clic en el punto de interés en la imagen izquierda y las coordenadas X y Y de este punto serán utilizadas. Una vez que se deciden las coordenadas a utilizar, se dibuja un cuadro rojo sobre la

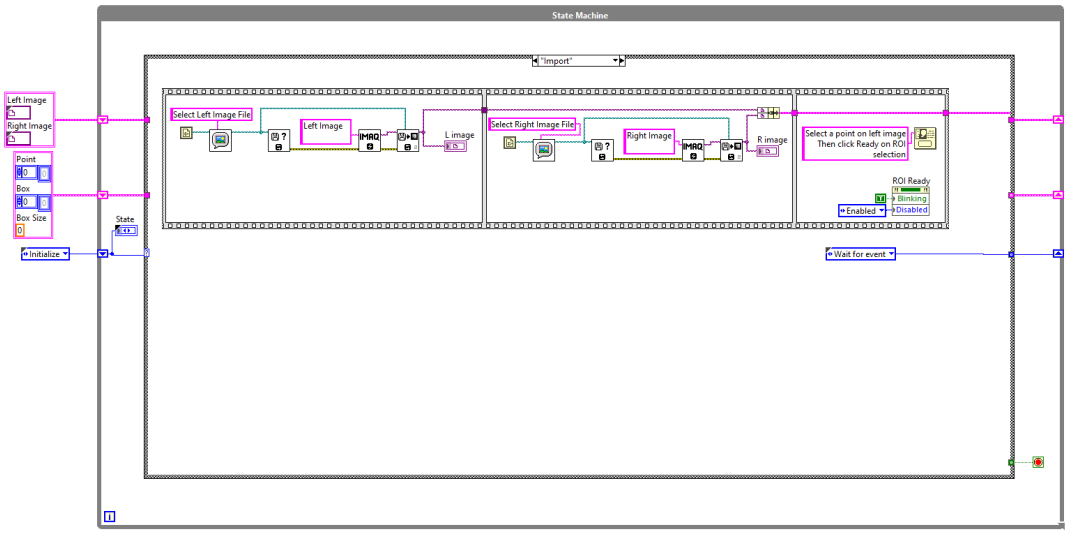



FIGURA 3.16: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Import

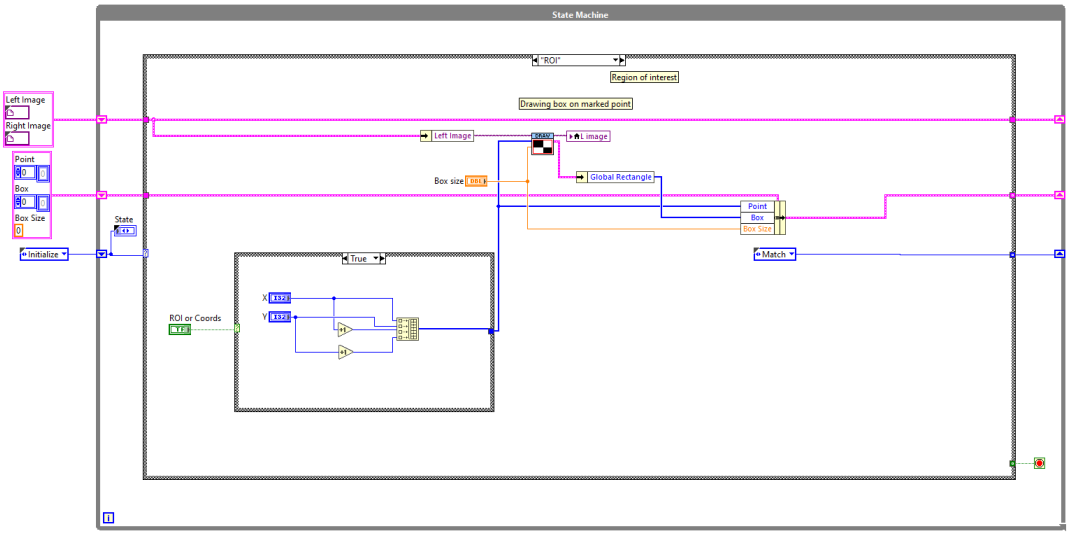



FIGURA 3.17: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado ROI

imagen para enmarcar la plantilla seleccionada, se guarda la información de la plantilla en un cluster y el programa avanza al estado Match.

### Match

En la figura 3.18 se puede observar el diagrama de bloques para el estado Match con el algoritmo SAD seleccionado. La función de este estado es aplicar el algoritmo para coincidencia de plantillas seleccionado, tomar el tiempo

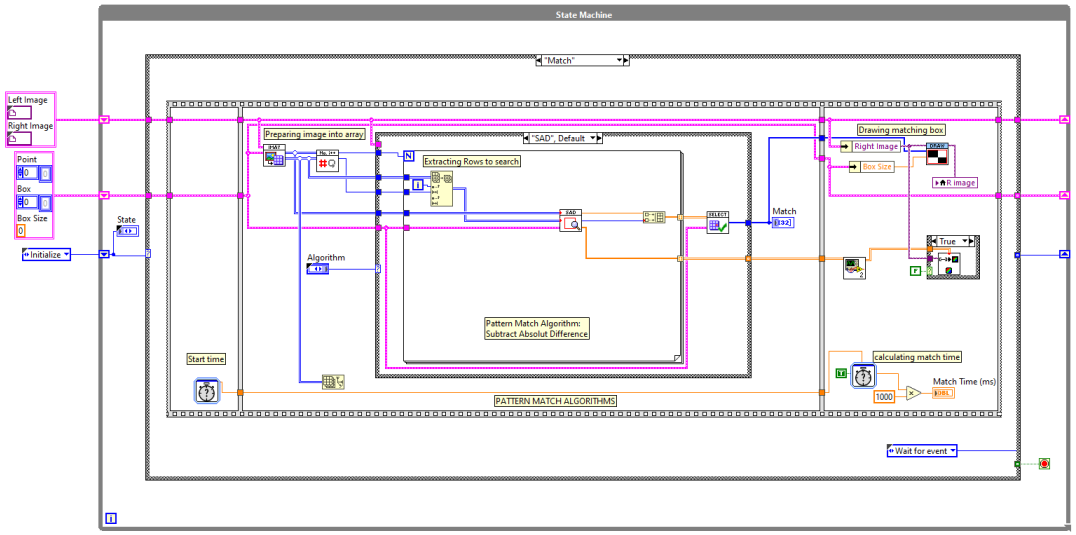



FIGURA 3.18: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Match con el algoritmo SAD

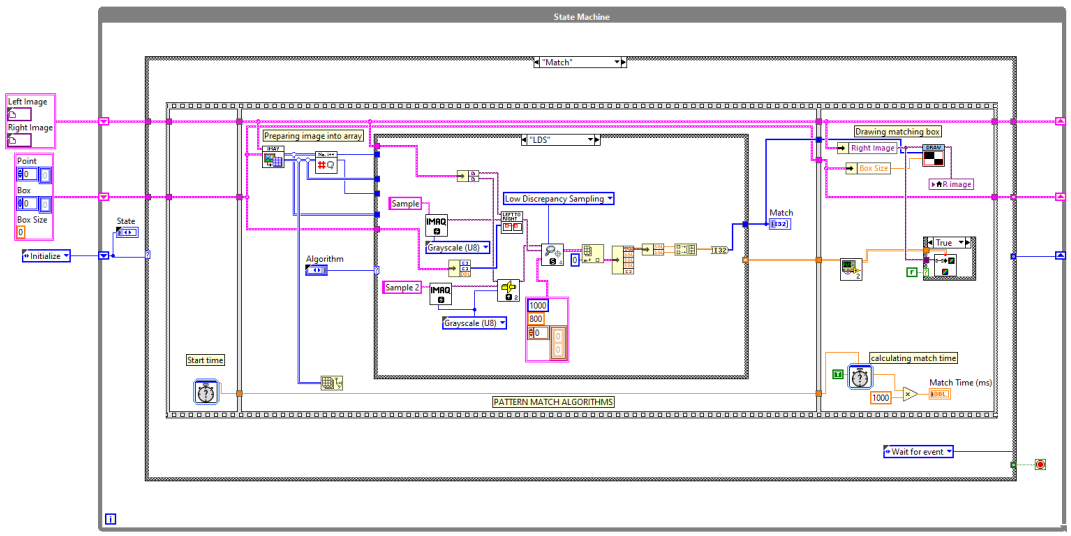



FIGURA 3.19: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Match con el algoritmo LDS

de ejecución y dibujar un cuadrado rojo en la imagen derecha mostrando el resultado de la coincidencia de plantillas.

En la figura 3.19 se puede observar el diagrama de bloques para el estado Match, pero ahora con el algoritmo LDS seleccionado. Para este algoritmo primero se deben convertir las imágenes a escala de grises antes de poder aplicarlo.

En la figura 3.20 se puede observar el diagrama de bloques para el estado Match, pero ahora con el algoritmo SoRA seleccionado.

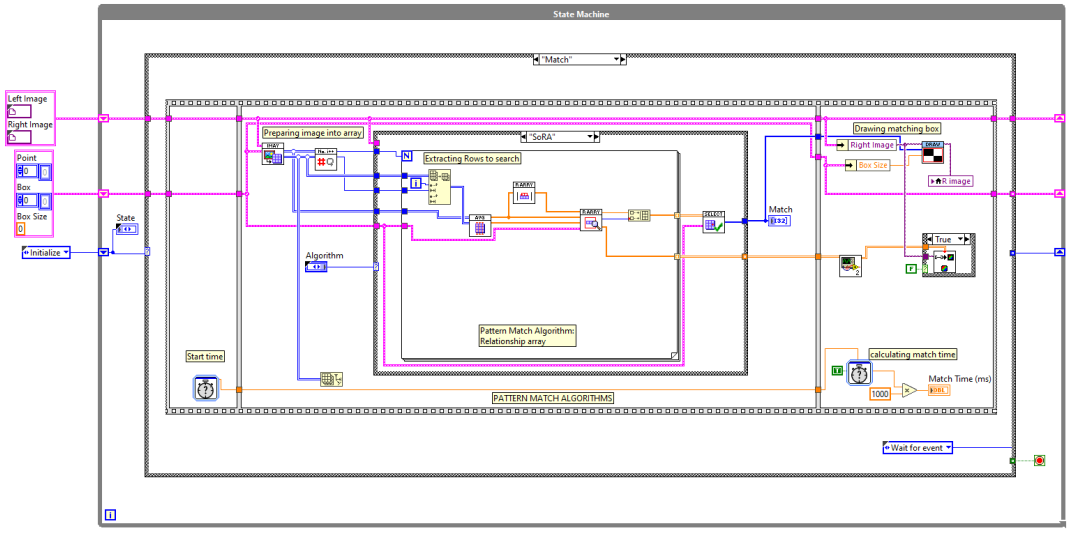



FIGURA 3.20: Diagrama de bloques del programa para coincidencia de plantillas mostrando el estado Match con el algoritmo SoRA

El algoritmo SoRA está dividido en 4 subVIs, el código del primer subVI se puede observar en la figura 3.21, la tarea de este subVI es calcular la media de columnas para la plantilla y la imagen objetivo como se mostró en las figuras 3.2 y 3.5. En el segundo SubVI se puede observar en la figura 3.22 y este calcula el arreglo de relaciones de plantilla como se mostró en la figura 3.3.

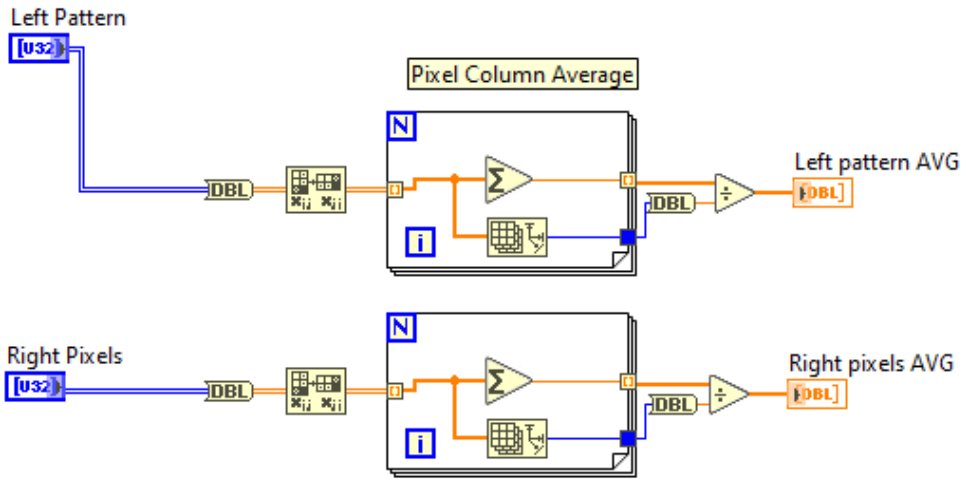



FIGURA 3.21: Diagrama de bloques del primer SubVI para el algoritmo SoRA.

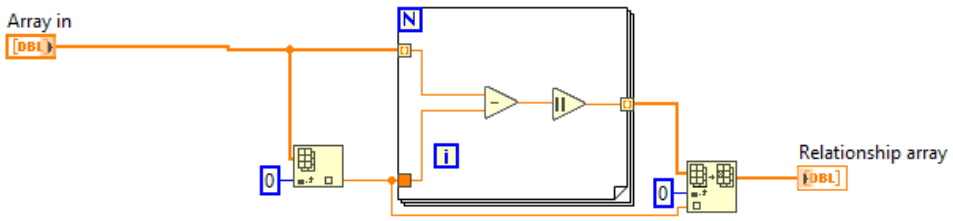



FIGURA 3.22: Diagrama de bloques del segundo SubVI para el algoritmo SoRA.

El tercer subVI se puede observar en la figura 3.23 y este calcula una calificación de coincidencia para cada uno de los candidatos a comparar con la plantilla original como se mostró en la ecuación 3.8.

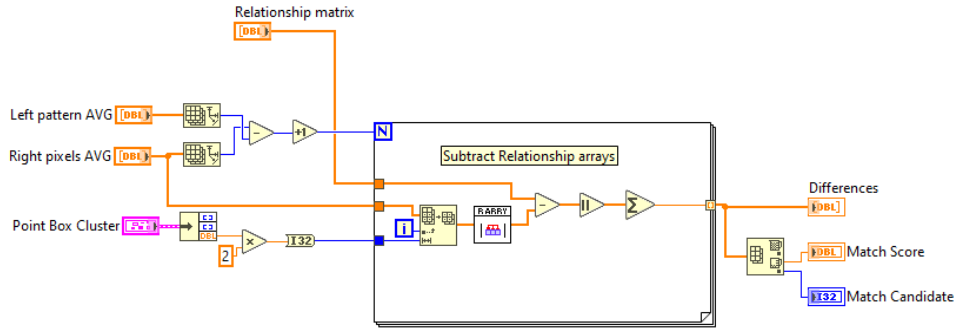



FIGURA 3.23: Diagrama de bloques del tercer SubVI para el algoritmo SoRA.

Por último, el cuarto subVI se puede observar en la figura 3.24 y este tiene la función de seleccionar la posición considerada como la coincidencia de la plantilla.

## 3.2. Método de calibración de cámaras mediante regresión cuadrática multivariable

A pesar de la eficacia de varios métodos de calibración para abordar la distorsión de lentes, cada enfoque presenta sus propias limitaciones. Las técnicas convencionales requieren la captura de múltiples imágenes de un patrón de calibración para lograr un modelado preciso de la distorsión de la lente, lo que los convierte en procedimientos lentos y laboriosos. Otros enfoques, como el modelo de distorsión radial, están específicamente diseñados

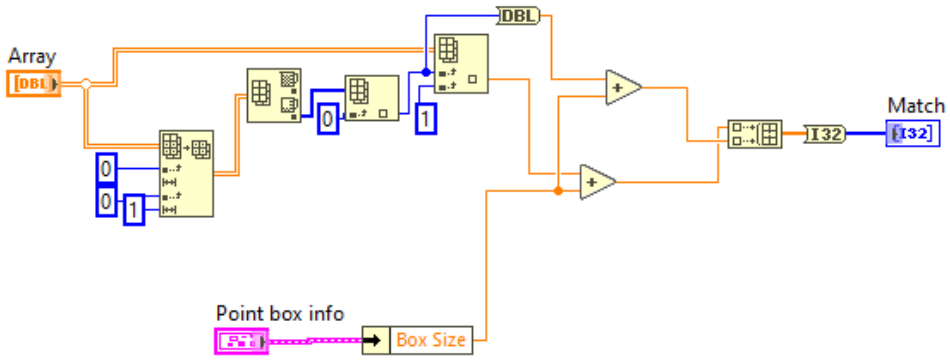



FIGURA 3.24: Diagrama de bloques del cuarto SubVI para el algoritmo SoRA.

para aplicaciones particulares, como cámaras de vigilancia o cámaras de tráfico, y podrían no resultar adecuados para otros tipos de cámaras o sistemas de imagen. Por otro lado, las estrategias basadas en aprendizaje profundo, como las redes neuronales convolucionales y completamente convolucionales, plantean una alternativa prometedora para corregir la distorsión de lentes. No obstante, estos enfoques demandan la generación de un conjunto de datos extenso, lo que puede constituir una tarea considerable. Además, suelen implicar un costo computacional más elevado, lo que limita su viabilidad para aplicaciones en tiempo real o aquellas con recursos computacionales limitados. De este modo, las principales contribuciones del método propuesto para la calibración de cámaras incluyen un enfoque de calibración mediante una sola imagen que simplifica el proceso a través de un patrón de calibración, un modelado minucioso de la distorsión de lentes para una calibración precisa y una fórmula de corrección de distorsión correspondiente que posibilita una corrección en tiempo real y mejora la utilidad práctica.

El método propuesto de calibración de cámara emplea un modelo de regresión cuadrática multivariable para estimar con precisión y modelar la distorsión de la lente. Además, corrige la orientación relativa de las cámaras y calcula un ángulo de compensación. Esta estrategia es altamente eficiente, ya que solo necesita una sola imagen del patrón de calibración en lugar de múltiples imágenes, como suele ser necesario en otros casos. Asimismo, la eficacia de la ecuación matemática utilizada permite un procesamiento de imágenes rápido, lo que lo hace idóneo para aplicaciones en tiempo real, especialmente en sistemas de visión estéreo, utilizados, por ejemplo, en la navegación autónoma. Estos sistemas necesitan una calibración rápida y precisa para funcionar de manera óptima. Además, la técnica propuesta es robusta

a las condiciones de iluminación, lo que implica que no se ve afectada por los cambios en el entorno lumínico. Esta es una ventaja significativa, ya que algunos métodos de calibración de cámaras pueden ser considerablemente afectados por las variaciones en la iluminación, lo que puede conducir a resultados imprecisos. Con este método, una vez que se ha modelado la distorsión de la lente, se puede corregir la posición de cualquier píxel utilizando la ecuación derivada, independientemente de las condiciones de iluminación. En resumen, la técnica propuesta de calibración de cámara es un método altamente eficiente y resistente para estimar y modelar la distorsión de la lente. Su capacidad para ser implementado en tiempo real, junto con su habilidad para mantenerse constante en distintas condiciones de iluminación, sumado a su necesidad de una sola imagen para la calibración y corrección de la distorsión de la lente, lo hace un instrumento valioso para sistemas de visión estéreo y otras aplicaciones que requieren una calibración precisa y confiable de la cámara.

En esta tesis se introduce un enfoque novedoso de calibración de cámara destinado a mejorar la exactitud en la estimación de la profundidad en un sistema de visión estéreo. El método se basa en el uso de un patrón específico de calibración para calcular los parámetros de calibración, que posteriormente se emplean en un conjunto de ecuaciones para corregir la distorsión de la imagen causada por la lente. Al introducir las coordenadas de un píxel, el método determina la posición calibrada correspondiente. Este procedimiento contribuye a una estimación más precisa de la profundidad.

El método de calibración de cámara propuesto esta basado en nuestro método propuesto anteriormente en Real-Moreno et al., 2022, opera directamente con las coordenadas de píxeles de los puntos de coincidencia. Su propósito es ajustar las coordenadas afectadas por la distorsión de la lente, particularmente en los bordes de la imagen, donde la distorsión es más prominente y provoca errores considerables en las mediciones de las coordenadas 3D. En consecuencia, el método propuesto segmenta la imagen en cuadrantes (con el píxel central de la imagen como origen de los cuadrantes) y aplica la ecuación 3.10 para corregir el eje $X$ y la ecuación 3.11 para corregir el eje $Y$.

$$X = X_{uc} + C_v X_{uc} Y_{uc} \tag{3.10}$$

$$Y = Y_{uc} + C_v X_{uc} Y_{uc} \tag{3.11}$$

Donde $X$ y $Y$ representan las coordenadas ajustadas, $X_{uc}$ y $Y_{uc}$ representan las coordenadas no ajustadas, y $C_v$ denota el valor de ajuste, el cual constituye una mejora con respecto a nuestra investigación previa. En el método anterior, una matriz de valores constantes compensa el error según el cuadrante de las coordenadas del píxel a ajustar; estos valores constantes corrigen el error originado por la distorsión de la lente. No obstante, a partir de este método, se evidencia que el ajuste podría mejorarse si se considera el comportamiento cuadrático de la distorsión de la lente. En este nuevo método, el valor de $C_v$ se adapta de acuerdo con el cuadrante y los valores de coordenadas del píxel, lo que resulta en una mejor compensación de los valores de las coordenadas. El valor de $C_v$ se calcula mediante la Ecuación 3.12.

$$C_v = a_{(q,0)} + a_{(q,1)}X_{uc} + a_{(q,2)}Y_{uc} + a_{(q,3)}X_{uc}^2 + a_{(q,4)}Y_{uc}^2 \tag{3.12}$$

Donde $q$ representa la sección del plano cartesiano en la que se encuentran las coordenadas y los valores de $a$ son los coeficientes utilizados para la calibración, los cuales se generan a partir de un análisis de regresión cuadrática multivariable. El procedimiento para obtener estos coeficientes se describe detalladamente en el diagrama de bloques ilustrado en la Figura 3.25.

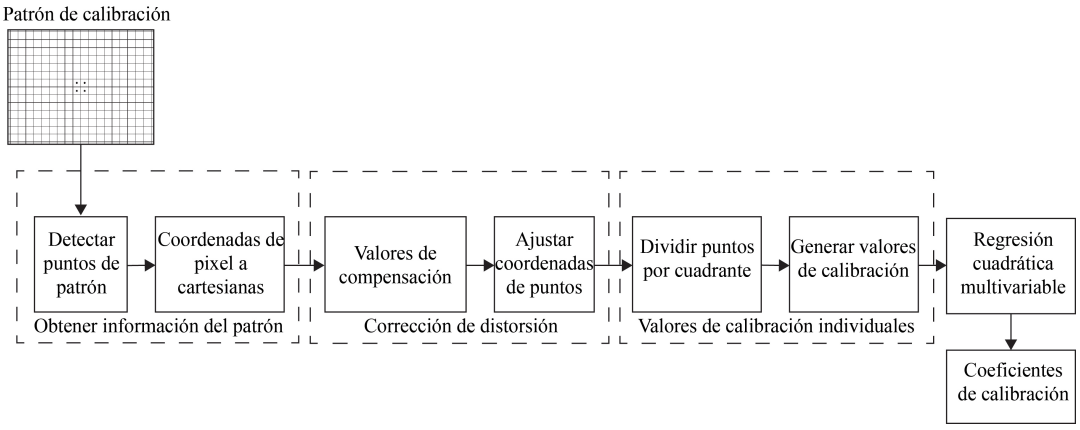



FIGURA 3.25: Diagrama de bloques del método de calibración propuesto.

**Detectar puntos de patrón**

En la primera etapa se emplea el patrón de calibración ilustrado en la Figura 3.26. Este patrón consta de 19 líneas verticales y 15 líneas horizontales. La función de este paso es identificar la posición en píxeles donde las líneas verticales y horizontales se cruzan.

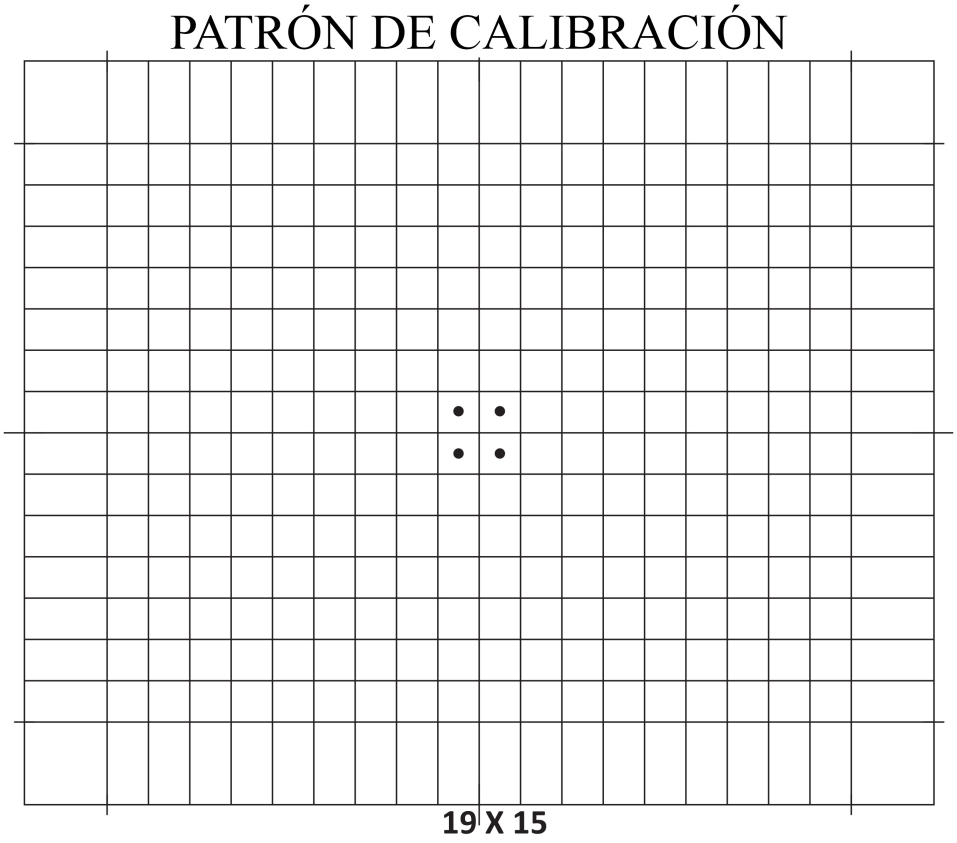



FIGURA 3.26: Patrón de calibración desarrollado con 19 líneas verticales y 15 líneas horizontales. Las intersecciones de las líneas se utilizan como puntos del patrón.

**Coordenadas de pixel a cartesianas**

Considerando que las coordenadas de las imágenes se inician en (0,0) en el píxel superior izquierdo, las posiciones de píxeles de los puntos del patrón de calibración se convierten en coordenadas cartesianas para establecer el origen del sistema de coordenadas (0,0) en el centro de la imagen.

**Valores de compensación**

En esta etapa se eligen los puntos del patrón de calibración en el eje-x y en el eje-y como valores de compensación. En estos puntos, la distorsión de las lentes es insignificante.

**Ajustar coordenadas de puntos**

Los valores de compensación se emplean para corregir los otros puntos de calibración. Esto se logra utilizando los valores de calibración del eje-x y

del eje-y como punto de referencia para ajustar los puntos de calibración.

**Dividir puntos por cuadrante**

En esta etapa, los puntos corregidos y los puntos sin calibrar se organizan por cuadrante, filtrando los puntos del eje-x y del eje-y y generando cuatro matrices para cada cuadrante.

**Generar valores de calibración**

Una vez que los puntos se dividen por cuadrante, se calcula un valor de calibración para cada punto utilizando la ecuación 3.13.

$$C_{xy,(i,j)} = \frac{P_{adj,(i,j)} - P_{raw,(i,j)}}{P_{adj,(i,j)}} \tag{3.13}$$

Donde $P_{raw}$ es una matriz con los puntos no calibrados, $P_{adj}$ es una matriz con los puntos ajustados, y $C_{xy,(i,j)}$ es una matriz que contiene los valores de calibración para las coordenadas x en la primera columna y los valores de calibración para las coordenadas y en la segunda columna.

**Regresión cuadrática multivariable**

Finalmente, se utiliza la regresión cuadrática multivariable (MQR, por sus siglas en inglés) para obtener los coeficientes de calibración y ajustar la coordenada x, y para obtener los coeficientes de calibración y ajustar la coordenada y. La entrada del MQR es la matriz $P_{raw}$ y la salida es $C_{xy,(i,0)}$ para los coeficientes de coordenadas x y $C_{xy,(i,1)}$ para los coeficientes de coordenadas y. La Ecuación 3.14 muestra la MQR utilizada para obtener los coeficientes de calibración.

$$\begin{bmatrix} n & \sum X_{uc} & \sum Y_{uc} & \sum X_{uc}^2 & \sum Y_{uc}^2 \\ \sum X_{uc} & \sum X_{uc}^2 & \sum X_{uc}Y_{uc} & \sum X_{uc}^3 & \sum X_{uc}^2 Y_{uc} \\ \sum Y_{uc} & \sum X_{uc}Y_{uc} & \sum Y_{uc}^2 & \sum X_{uc}^2 Y_{uc} & \sum Y_{uc}^3 \\ \sum X_{uc}^2 & \sum X_{uc}^3 & \sum X_{uc}^2 Y_{uc} & \sum X_{uc}^4 & \sum X_{uc}^2 Y_{uc}^2 \\ \sum Y_{uc}^2 & \sum X_{uc}Y_{uc}^2 & \sum Y_{uc}^3 & \sum X_{uc}^2 Y_{uc}^2 & \sum Y_{uc}^4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} \sum C_{xy} \\ \sum C_{xy}X_{uc} \\ \sum C_{xy}Y_{uc} \\ \sum C_{xy}X_{uc}^2 \\ \sum C_{xy}Y_{uc}^2 \end{bmatrix} \tag{3.14}$$

Donde $a_0$, $a_1$, $a_2$, $a_3$ y $a_4$ son los coeficientes de calibración para el cuadrante y el eje actuales. La MQR debe realizarse dos veces por cuadrante,

una para los coeficientes que calibran la coordenada X y otra para los coeficientes de la coordenada Y. Estos coeficientes se almacenan en la matriz $a$ que se muestra en la ecuación 3.15.

$$a = \begin{bmatrix} a_{(0,0)} & \cdots & a_{(0,m)} \\ \vdots & \ddots & \vdots \\ a_{(q,0)} & \cdots & a_{(q,m)} \end{bmatrix} \tag{3.15}$$

Donde cada fila contiene los coeficientes de calibración para los valores de $X$ e $Y$ de cada cuadrante, y las primeras cinco columnas contienen los coeficientes de calibración para los valores de $X$ y las últimas cinco columnas contienen los coeficientes de calibración para los valores de $Y$.

Una vez que se calculan los coeficientes de calibración, el siguiente paso es calcular los parámetros extrínsecos para el sistema de visión estéreo, específicamente, la orientación relativa de las cámaras. Esto consiste en calcular un ángulo de compensación para una de las cámaras, que se realiza teniendo en cuenta la distancia del patrón de calibración a las cámaras y la distancia medida como se muestra en la ecuación 3.16.

$$C_{comp} = \cot^{-1}\left(\frac{b}{X_r} - \cot B\right) - C \tag{3.16}$$

Donde $b$ es la línea base estéreo, $X_r$ es el valor de distancia real, el cual es obtenido mediante un medidor láser $B$ y $C$ son los ángulos de la Figura 2.6. Este ángulo de compensación se utiliza para calcular un nuevo valor para $C$ como $C' = C + C_{comp}$. Donde $C'$ se usará en lugar de $C$ al tomar medidas con el sistema calibrado.

### 3.2.1. Diseño de prototipo

Para comprobar el funcionamiento del sistema de calibración propuesto se desarrollo un prototipo para el sistema de visión estereoscópica como el que se muestra en la Figura 3.27. El prototipo incluye dos cámaras USB equipadas con lentes varifocales que van desde 5 mm hasta 50 mm y un sensor Sony IMX179. Estas cámaras están acomodadas en un sistema de geometría coplanar con una separación de 100 mm entre ellas. Esta disposición es común en sistemas de visión estéreo, ya que permite la generación de mapas de profundidad al analizar la diferencia de perspectiva entre ambas cámaras. Las cámaras están conectadas a una computadora portátil Asus TUF que ejecuta el sistema operativo Windows 11 y cuenta con un procesador Ryzen 5 4600H a una velocidad de reloj de 3.00 GHz y una GPU NVIDIA RTX 3050.

Esta configuración de hardware garantiza que el sistema disponga de la potencia de procesamiento y las capacidades informáticas necesarias para llevar a cabo los algoritmos de visión estéreo.

FIGURA 3.27: Prototipo de sistema de visión estereoscópica.

El software utilizado para el sistema de visión estéreo se muestra en la Figura 3.28, fue creado utilizando LabVIEW y el módulo de desarrollo de visión. El lado izquierdo de la interfaz de usuario contiene los controles del sistema utilizados para importar los coeficientes de calibración desde un archivo .csv, iniciar/detener la adquisición de imágenes y calcular las coordenadas 3D de cada intersección cuadrada en el tablero de ajedrez utilizando los botones "MATCH" y "TRIANGULATE"; esta sección también incluye los parámetros del tamaño de la imagen del sistema de visión estéreo, el ángulo de compensación para el alineamiento estéreo en la orientación relativa de las cámaras y a la derecha los ángulos de visión de cada cámara. La parte inferior derecha de la interfaz de usuario contiene los coeficientes de calibración de cada cámara. La mitad de la interfaz de usuario muestra la imagen adquirida por ambas cámaras. Finalmente, la parte superior izquierda del área de la cámara tiene una pestaña para cambiar las cámaras y mostrar gráficos que presentan las representaciones tridimensionales de las mediciones realizadas, un gráfico contiene las mediciones no calibradas y el otro gráfico

contiene las mediciones calibradas, estos gráficos 3D se detallan en el capítulo cuatro. Este software detecta todas las intersecciones entre los cuadrados blancos y negros y realiza la triangulación con los puntos en la imagen derecha para obtener las coordenadas tridimensionales de cada intersección en el espacio real, es decir, en el mundo físico en el que se encuentra el objeto o la escena que está siendo observada por las cámaras.

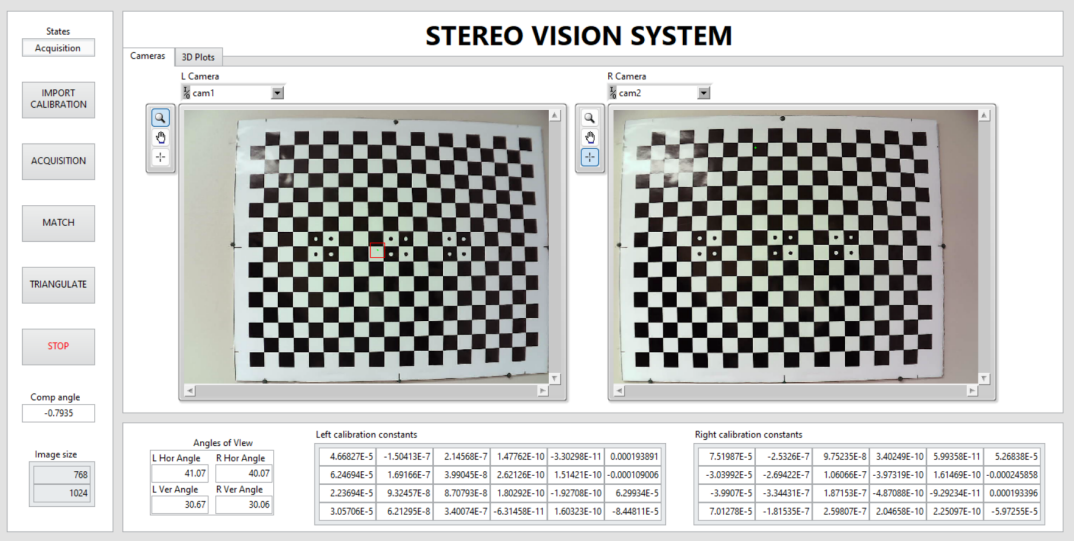



FIGURA 3.28: Software del sistema de visión estéreo utilizado para obtener las coordenadas tridimensionales del mundo real del patrón.

El archivo .csv con las constantes de calibración utilizadas en el software del sistema de visión estéreo se genera utilizando otro software, que también fue creado utilizando LabVIEW. La interfaz de usuario del software de calibración de la cámara se muestra en la Figura 3.29. Los controles en el lado izquierdo de la interfaz de usuario se utilizan para importar la imagen de la cámara para calibrar, para emparejar las intersecciones de línea del patrón de calibración, para calcular los coeficientes de calibración según el método propuesto, y para exportar los coeficientes de calibración como un archivo .csv y detener el programa. El lado derecho de la interfaz de usuario muestra la imagen importada y dibuja cajas rojas sobre la intersección cuadrada. Finalmente, la parte inferior de la interfaz de usuario contiene una pantalla que muestra los coeficientes de calibración calculados.

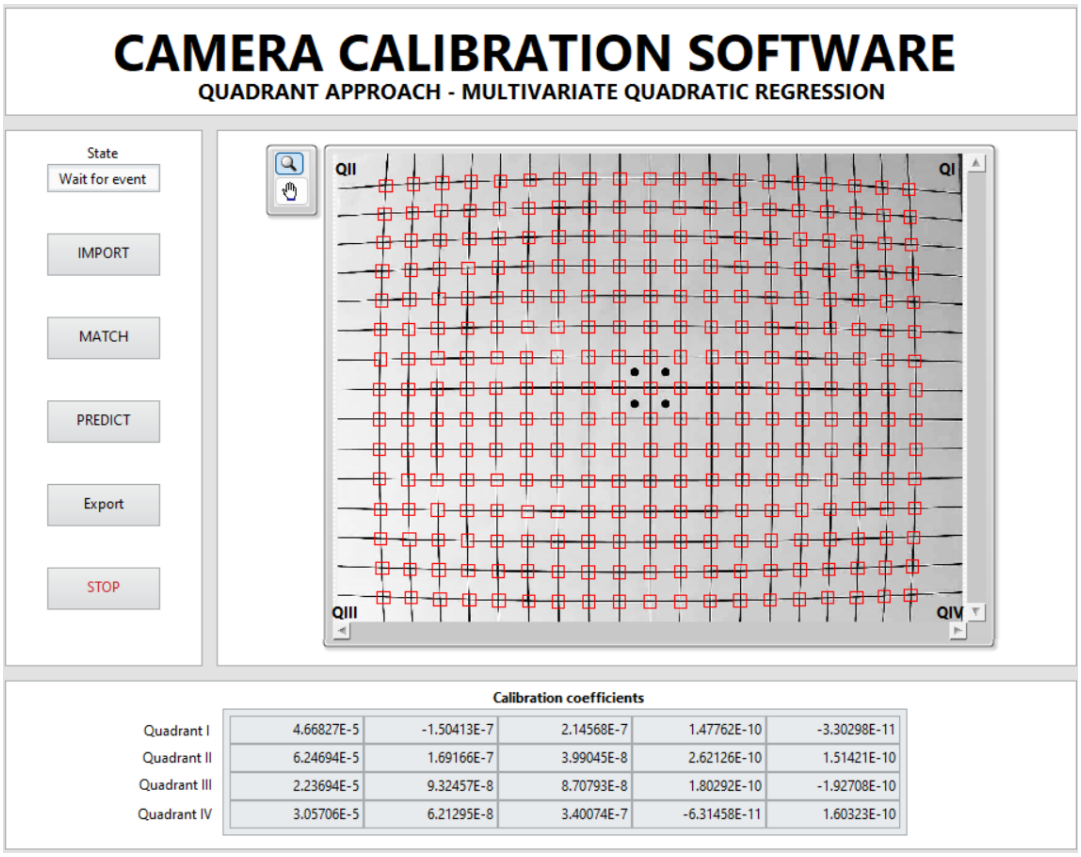



FIGURA 3.29: Software para calibrar el sistema mediante regresión cuadrática multivariable.

## 3.3. Combinación de sistema de visión estereoscópica y algoritmo de detección de objetos

Una vez que se han abordado con éxito los desafíos asociados con la coincidencia de plantillas y la calibración precisa de los parámetros intrínsecos y extrínsecos, se abre la puerta para integrar el sistema de visión estereoscópica con un algoritmo de detección de objetos. En este contexto, se ha optado por la implementación del algoritmo YOLOR (You Only Learn One Representation), desarrollado por Wang, Yeh y Liao, 2021. YOLOR es una variante del conocido algoritmo YOLO (You Only Look Once), conocido por su eficiencia y precisión en la detección de objetos en imágenes y videos.

El algoritmo YOLOR se destaca por su capacidad para realizar detecciones en un solo paso a través de la red neuronal, lo que significa que, a diferencia de enfoques tradicionales, evalúa la imagen completa en una sola inferencia. Este enfoque resulta en una eficiencia significativa, ya que evita repeticiones innecesarias al analizar cada región por separado. Además,

YOLOR se beneficia de una arquitectura de red profunda, que le permite aprender representaciones jerárquicas complejas de las características de la imagen, contribuyendo así a su robustez en la identificación de objetos en diversas condiciones. En la figura 3.30 se puede apreciar un ejemplo del algoritmo YOLOR haciendo inferencia sobre una imagen con diversos objetos.

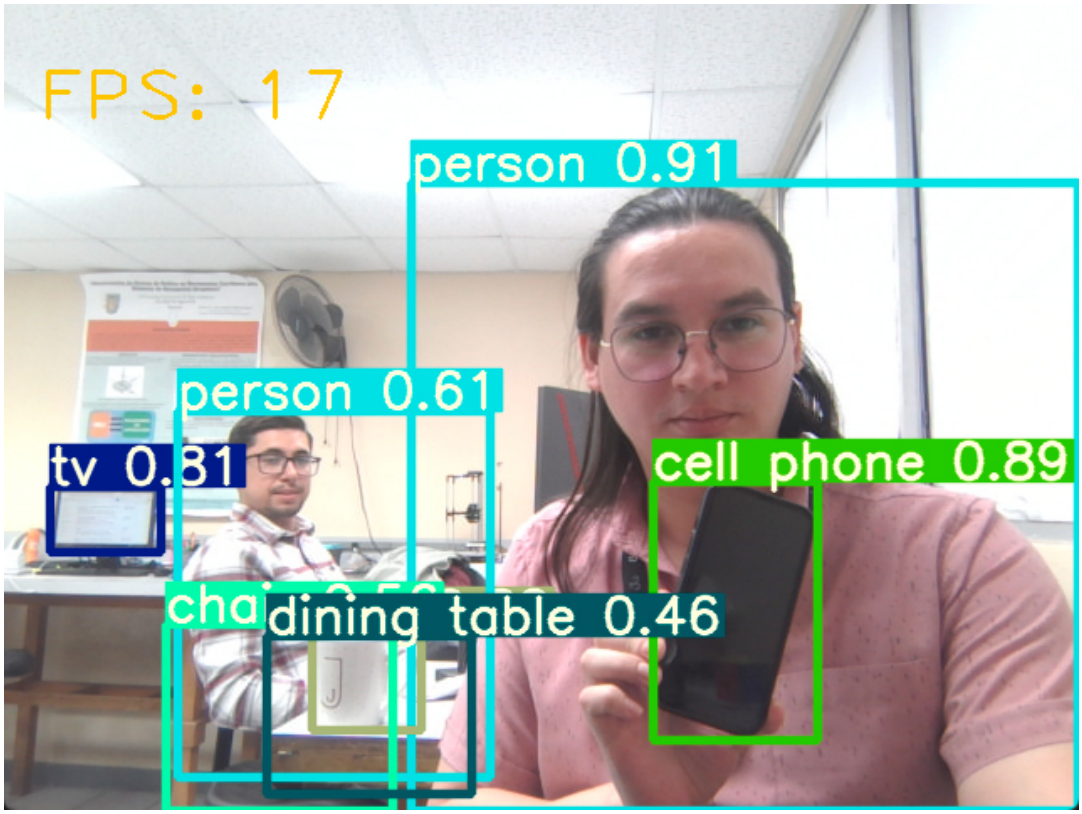



FIGURA 3.30: Ejecución del algoritmo YOLOR detectando una variedad de objetos.

El modelo YOLOR ha demostrado ser especialmente efectivo en entornos donde la velocidad de procesamiento es crucial, como en sistemas de tiempo real o aplicaciones de navegación autónoma con una precisión promedio $AP_{50}$ de 75 % a 30 cuadros por segundo. Al integrar este algoritmo con el sistema de visión estereoscópica, se busca aprovechar la rapidez y precisión de YOLOR para detectar objetos en la imagen izquierda y, mediante el uso de la información estereoscópica, realizar una triangulación precisa para determinar la posición tridimensional de dichos objetos en el espacio. Este enfoque busca mejorar la capacidad del sistema para abordar desafíos específicos asociados con la navegación autónoma, proporcionando así una base sólida para investigaciones y aplicaciones futuras en este campo.

El algoritmo de detección de objetos, implementado en Python, es un componente esencial dentro del marco de trabajo propuesto. Esta elección se basa en las ventajas inherentes de Python en términos de flexibilidad, eficiencia y la disponibilidad de bibliotecas especializadas para visión por computadora. Esta transición de LabVIEW a Python no solo permite la ejecución eficiente del algoritmo de detección, sino que también facilita la integración con el sistema de visión estereoscópica.

La migración a Python ha demostrado ser beneficiosa, especialmente en lo que respecta a la transferencia de información entre las etapas del proceso. Al realizar la detección de objetos en la imagen izquierda, la flexibilidad de Python ha permitido establecer un flujo de datos eficiente y coherente que simplifica la comunicación entre el algoritmo y la etapa de triangulación. Esto se traduce en una mejora significativa en la eficiencia del sistema, ya que la información relevante sobre los objetos detectados se transfiere de manera fluida a la etapa de correspondencia en la imagen derecha.

Además, la elección de Python facilita la implementación de las funciones necesarias para la triangulación precisa de los objetos detectados. Las bibliotecas de visión por computadora en Python, como OpenCV, ofrecen herramientas avanzadas para manipular imágenes. Esta transición también ha permitido aprovechar las numerosas capacidades que Python ofrece, mejorando así la robustez y la precisión de todo el sistema.

En resumen, la adopción de Python como el entorno principal para el algoritmo de detección de objetos no solo ha simplificado la implementación y ejecución del algoritmo, sino que también ha facilitado una integración más eficiente con el sistema de visión estereoscópica. Esta elección unificada de Python no solo mejora la coherencia en el flujo de trabajo, sino que también sienta las bases para futuras expansiones y mejoras en la investigación y desarrollo de sistemas de navegación autónoma basados en visión por computadora. Para proporcionar una visión más clara de la implementación, a continuación se presenta una versión simplificada del pseudocódigo del programa, destacando las funciones esenciales del proceso.

En la Figura 3.31, se presenta un ejemplo del sistema de visión estereoscópica combinado con el algoritmo de detección de objetos. En la figura 3.31A, se muestra la imagen capturada por la cámara izquierda. El algoritmo de detección de objetos realiza inferencia en esta imagen, y en los recuadros de color rosa se destacan dos personas detectadas con una confianza del 0.94. Además, en la esquina superior izquierda se indica la cantidad de fotogramas procesados por segundo. La figura 3.31B presenta la imagen capturada

**Algoritmo 1** Pseudocódigo del Proceso de Detección de Objetos y Triangulación

```
Importar librerías y módulos
Inicializar variables y configuraciones
Iniciar cámaras
Cargar modelo de detección en modo de ejecución para GPU
while Stop == false do                                   ▷ Ciclo principal
    Capturar imagen izquierda y derecha
    Realizar inferencia de detección en la imagen izquierda
    Procesar las detecciones
    for cada detección do
        Realizar la coincidencia de objetos en la imagen derecha
        if hay coincidencia en imagen derecha then
            Calibrar puntos de coincidencia
            Triangulación de puntos
            Guardar datos en una lista
            Generar visualización de resultados
        end if
    end for
    Mostrar imágenes y resultados
    if se presiona 'q' then
        Guardar datos en un archivo CSV
        Finalizar el programa
    end if
end while
```

por la cámara derecha. El algoritmo SoRA lleva a cabo la coincidencia de objetos, mostrando en recuadros de color azul las correspondencias encontradas para cada objeto. Finalmente, el texto en negrita sobre los recuadros azules revela las coordenadas $x$, $y$, y $z$ de cada objeto.

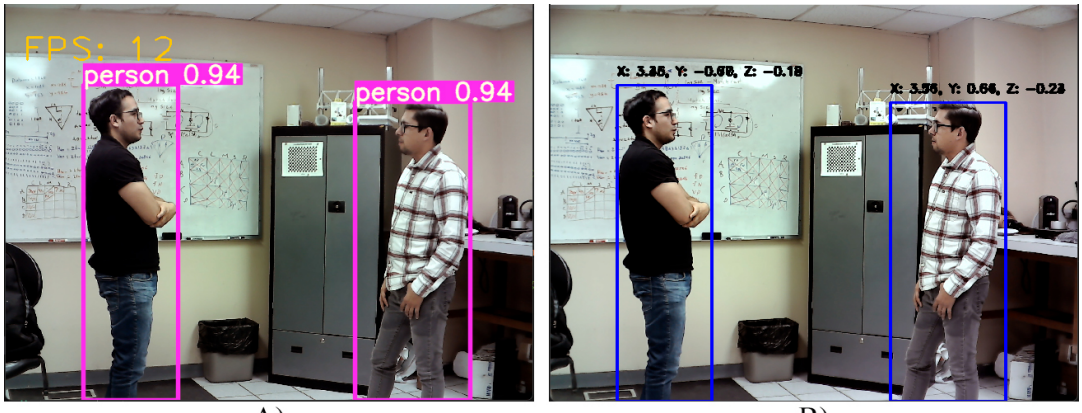



FIGURA 3.31: Combinación entre detección de objetos y visión estereoscópica. A) Imagen izquierda B) Imagen derecha.

# Capítulo 4

# Experimentos y Análisis de Resultados

## 4.1. Experimento y resultados de SoRA

Para probar el tiempo de ejecución y la precisión de nuestro algoritmo, se realizaron varios experimentos utilizando el conjunto de datos DrivingStereo Yang et al., 2019, que es un conjunto de datos estéreo a gran escala que contiene más de 180 000 imágenes con una amplia variedad de escenarios de conducción, lo que hace que este conjunto de datos sea ideal para la aplicación prevista.

También se llevaron a cabo experimentos con otros métodos de coincidencia de plantillas y detección de objetos, como SAD, LDS, BBS y Fast-Match, con el fin de comparar los tiempos de ejecución y la precisión. Se realizó el mismo número de experimentos bajo condiciones idénticas a las de nuestro algoritmo con estos métodos. Se llevó a cabo un experimento de correspondencia de objetos para evaluar la eficacia del algoritmo propuesto en la tarea de correspondencia de objetos en imágenes estéreo. Específicamente, el experimento tenía como objetivo determinar si el algoritmo propuesto puede identificar con precisión los objetos en la imagen derecha que originalmente fueron detectados en la imagen izquierda por un algoritmo de detección de objetos. Se comparó esto con el rendimiento de un algoritmo de detección de objetos que opera en ambas imágenes, izquierda y derecha. El algoritmo YOLOR Wang, Yeh y Liao, 2021 se utilizó como el algoritmo de detección de objetos en este experimento.

Los experimentos se dividen en tres secciones, que comprenden la búsqueda de la imagen completa, la búsqueda en fila esperada y la correspondencia de objetos. La búsqueda de la imagen completa implica la búsqueda tradicional de plantillas en la que se busca la coincidencia en toda la imagen. Mientras que la búsqueda de fila esperada implica buscar la coincidencia en

la misma coordenada de fila que la plantilla (por ejemplo, si las coordenadas de la plantilla son 432, 155, la fila esperada para encontrar la coincidencia será la fila 432). Esto se debe a la geometría epipolar de los planos de imagen paralelos entre sí y alineados en el eje horizontal. El experimento de correspondencia de objetos implica ejecutar el algoritmo de detección de objetos en la imagen izquierda, luego encontrar cada objeto en la imagen derecha utilizando el algoritmo SoRA y el algoritmo de detección de objetos para comparar sus resultados en términos de velocidad y precisión.

Cada etapa del experimento se realiza con diferentes tamaños de plantillas que van desde 20x20, 30x30, 40x40, 50x50 y 100x100 píxeles en imágenes de tamaño 881x400 píxeles, utilizando una computadora con un procesador Intel Core i7-6500U a 2.5 GHz. El entorno de ejecución para el algoritmo propuesto (SoRA), SAD y LDS es LabVIEW 2021, mientras que el entorno de ejecución para BBS y Fast-Match es Matlab. Para evaluar la precisión del algoritmo, tanto la plantilla como la coincidencia se seleccionaron manualmente. Las plantillas seleccionadas incluyen vehículos, camiones, motocicletas, personas y señales de tráfico, que son los elementos previstos para ser identificados en la aplicación. La Figura 4.1 ilustra un ejemplo de A) búsqueda completa de la imagen y B) búsqueda de fila esperada utilizando una plantilla de 50x50 píxeles, donde la imagen izquierda representa la extracción de la plantilla, marcada con un recuadro rojo, y la imagen derecha es el objetivo de búsqueda, señalada con un recuadro rojo indicando la coincidencia de la plantilla.


FIGURA 4.1: Ejemplo de A)Búsqueda en imagen completa B)Búsqueda en fila esperada

La figura 4.2 muestra ejemplos del experimento realizado en varias imágenes diferentes. Las coincidencias se han ampliado para facilitar la apreciación. Cada par de imágenes muestra la plantilla extraída de la imagen de origen y las coincidencias obtenidas por los distintos algoritmos en la imagen

objetivo. Cada resultado de los algoritmos se representa con un recuadro de un color diferente. Es importante tener en cuenta que en casos donde algunos recuadros de color no aparecen, es debido a que se superponen con otro recuadro de color de otro algoritmo que dio el mismo resultado. En estos ejemplos, todos los algoritmos produjeron resultados similares dentro de un rango de ±2 píxeles, con la excepción de la plantilla del semáforo, donde la mayoría de los algoritmos se confundieron con la segunda luz. Únicamente nuestro algoritmo y LDS lograron la coincidencia exacta, mientras que Fast-Match obtuvo un resultado mejor que BBS y SAD.

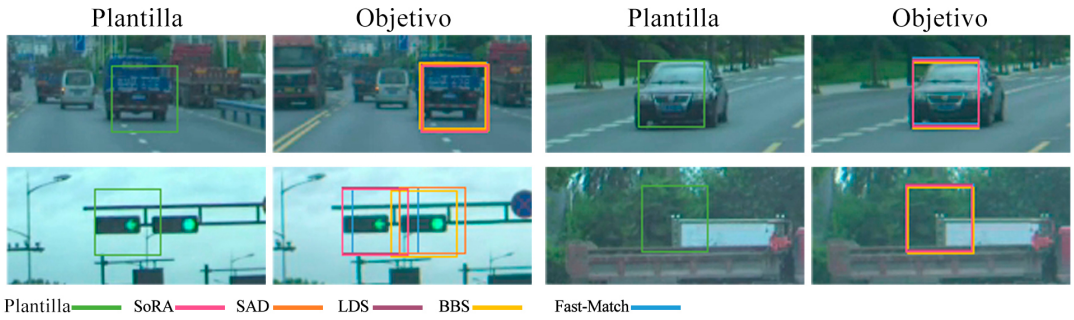



FIGURA 4.2: Resultados de coincidencia de plantillas utilizando diferentes algoritmos

Otra forma de comprender mejor cómo se comportan los algoritmos dada la misma plantilla es comparar los mapas de disimilitud. La Figura 4.3 muestra los mapas de disimilitud para los algoritmos SoRA, SAD y BBS en cuatro imágenes. SoRA y SAD muestran los puntos con una disimilitud más alta en tonos azul oscuro y los puntos con una disimilitud más baja (que son los puntos más cercanos a ser una coincidencia) con tonos amarillos claros. Mientras tanto, los mapas de disimilitud para BBS muestran los puntos con una disimilitud más alta en tonos azul oscuro, los puntos medios con tonos amarillos y los puntos con una disimilitud más baja con tonos rojos. Idealmente, estos mapas de disimilitud deberían ser de un tono azul oscuro con solo la coincidencia exacta en un color amarillo claro y se puede observar que el algoritmo SoRA tiene más tonos azules oscuros que SAD y BBS, lo que significa que es menos probable que no coincida el objetivo. La única excepción se encuentra en la imagen de los semáforos donde aparecen más tonos amarillos, pero SoRA aún logró la coincidencia exacta donde la mayoría de los algoritmos no coincidieron.

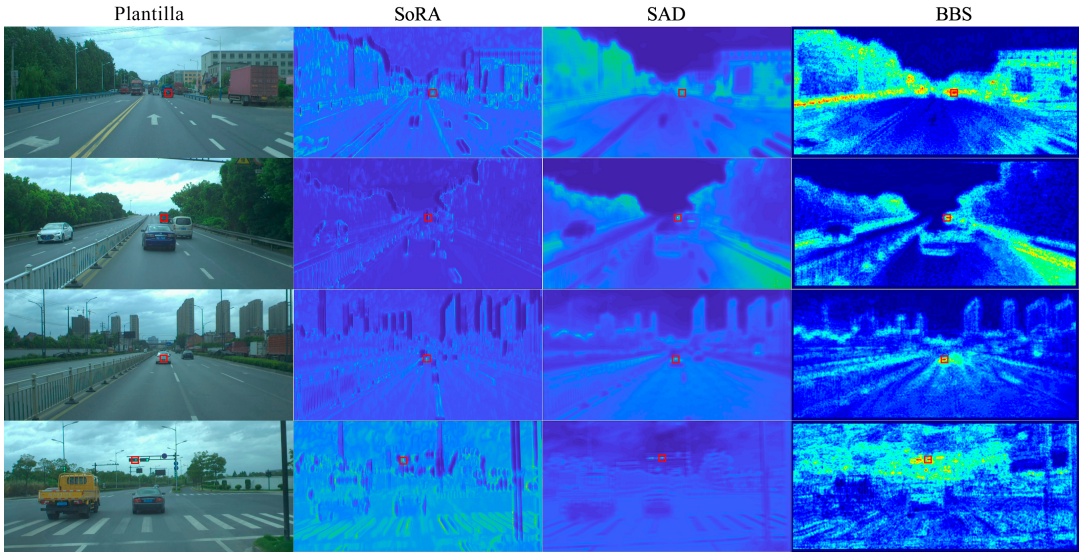



FIGURA 4.3: Mapas de disimilitud en diferentes algoritmos

### 4.1.1. Resultados de búsqueda en la imagen completa

Los resultados de los experimentos de búsqueda de imagen completa realizados con SoRA, SAD, LDS, BBS y Fast-Match se pueden ver en la tabla 4.1. Esta tabla muestra el tiempo de ejecución promedio (AVG Time) en milisegundos, el error cuadrático medio (RMSE) y las coincidencias fallidas para tamaños de plantilla de 20x20, 30x30, 40x40, 50x50 y 100x100 píxeles. Una coincidencia se considera fallida cuando el algoritmo no coincide con un error de 15 píxeles o más. Hay que tener en cuenta que el algoritmo Fast-Match tiene la leyenda N/A (No aplicable) en sus resultados para los tamaños de plantilla de 20x20 y 30x30, esto se debe a que ese algoritmo no funciona con esos tamaños de plantilla. Además, observe que los mejores resultados en cada columna están marcados en negrita.

TABLA 4.1: Resultados de búsqueda de imagen completa con diferentes tamaños de plantilla

| | Plantilla de 20x20 píxeles | | | Plantilla de 30x30 píxeles | | | Plantilla de 40x40 píxeles | | | Plantilla de 50x50 píxeles | | | Plantilla de 100x100 píxeles | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match |
| SoRA | 160.95 | 1.8165 | **0** | 208.07 | 2.3664 | **0** | 228.96 | 3.3916 | 4 | 238.30 | 4.2088 | 3 | 332.38 | 7.5166 | 4 |
| SAD | 496.84 | **1.2909** | 1 | 1000.20 | **0.9654** | 2 | 1822.06 | **0.8165** | 1 | 2256.98 | **0.8165** | 1 | 8030.284 | 2.4494 | **0** |
| LDS | **33.004** | 1.3416 | **0** | **24.93** | 1.4832 | **0** | **28.40** | 1.4142 | 1 | **46.56** | 2.2852 | 1 | **59.64** | **1.8516** | 3 |
| BBS | 2800.01 | 3.0937 | 3 | 8482.87 | 2.6267 | **0** | 16341.14 | 2.0275 | 1 | 38184.80 | 2.6832 | **0** | 692418.411 | 4.0987 | **0** |
| Fast-Match | N/A | N/A | N/A | N/A | N/A | N/A | 178778.12 | 3.8078 | **0** | 89056.71 | 3.9874 | **0** | 28628.90 | 9.9211 | 3 |

La figura 4.4 muestra las gráficas de los resultados de la tabla 4.1 para el tiempo promedio. El gráfico del tiempo promedio de coincidencia de la imagen completa muestra los resultados de los tres algoritmos más rápidos para una mejor comparación. Estos algoritmos son SoRA, SAD y LDS, y de

este gráfico se puede concluir que el algoritmo más rápido es el LDS con un promedio de 38,5 ms. También hay que tener en cuenta que a medida que aumenta el tamaño de la plantilla, el tiempo de ejecución del algoritmo SAD también aumenta, esto también es cierto para el algoritmo SoRA, pero con incrementos más bajos debido a la complejidad algorítmica lineal. En los resultados del RMSE se puede observar que en el tamaño de plantilla más pequeño, los algoritmos SoRA, SAD y LDS dieron los mejores resultados, pero a medida que aumenta el tamaño de la plantilla, también lo hace el RMSE para el algoritmo SoRA, mientras que SAD y LDS se mantienen casi en los mismos valores. El aumento en el RMSE solo ocurre en plantillas fijas, pero se mantiene bajo cuando las plantillas son proporcionadas por el algoritmo de detección de objetos y la correspondencia de objetos se realiza mediante el algoritmo SoRA, que es la aplicación prevista de nuestra propuesta. Esta afirmación se demuestra en los experimentos de la correspondencia de objetos.

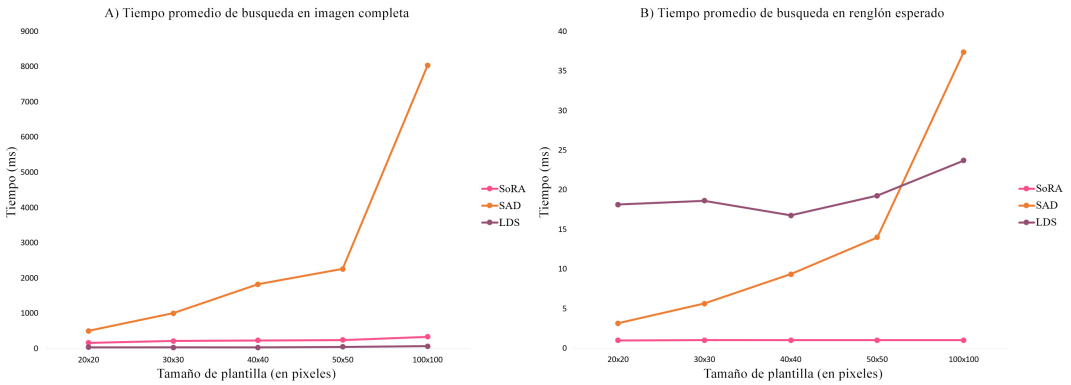



FIGURA 4.4: Tiempo promedio de búsqueda con distintos algoritmos en A) Imagen completa B) Renglón esperado

### 4.1.2. Resultados de búsqueda en renglón esperado

A partir de los resultados de la búsqueda en imagen completa, se puede observar que LDS es la mejor opción para la tarea de detección espacial de objetos, pero con un tiempo de 38,5 ms no es lo suficientemente rápido como para ser una mejor alternativa que los enfoques globales, ya que con solo tres objetos detectados, el algoritmo LDS tardará más tiempo de ejecución que los enfoques globales. Por esta razón, es necesario realizar experimentos de búsqueda de imágenes de fila esperada con los mismos algoritmos. La tabla 4.2 muestra los resultados de estos experimentos. Los resultados muestran que

TABLA 4.2: Resultados de búsqueda en renglón esperado con diferentes tamaños de plantilla

| | Plantilla de 20x20 píxeles | | | Plantilla de 30x30 píxeles | | | Plantilla de 40x40 píxeles | | | Plantilla de 50x50 píxeles | | | Plantilla de 100x100 píxeles | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match | AVG Time (ms) | RMSE | Failed match |
| SoRA | **0.9972** | **0.6324** | **0** | **0.9989** | 0.8819 | 1 | **0.9973** | 1.1547 | 1 | **0.9983** | 1.7320 | **0** | **0.9984** | 4.7140 | 1 |
| SAD | 3.1217 | **0.6324** | **0** | 5.6077 | **0.4472** | **0** | 9.3209 | **0.3162** | **0** | 13.98 | **0.4472** | **0** | 37.36 | 2.3874 | **0** |
| LDS | 18.11 | 0.7071 | 6 | 18.6026 | 0.6546 | 3 | 16.74 | 0.8165 | 4 | 19.23 | 0.7071 | 4 | 23.68 | **1.9493** | 5 |
| BBS | 85.43 | 5.9160 | **0** | 102.92 | 1.4142 | **0** | 166.46 | 3.1464 | **0** | 336.07 | 1.7888 | **0** | 6769.5 | 2.7748 | **0** |
| Fast-Match | 17108.65 | 1.1832 | **0** | 15593.47 | 1.2247 | **0** | 13440.23 | 2.1908 | **0** | 12836.42 | 5.1575 | **0** | 9175.2896 | 2.2803 | 5 |

el tiempo de ejecución en todos los algoritmos se redujo significativamente con la búsqueda de fila esperada, lo cual es lógico porque requiere menos cálculos. Además, el RMSE se redujo en la mayoría de los casos, pero en el caso de LDS, el algoritmo no logró realizar coincidencias más que en la búsqueda en imagen completa, lo que lo hace no adecuado para la búsqueda de imágenes de fila esperada.

Según los resultados mostrados en la Figura 4.4B, es evidente que el algoritmo propuesto, SoRA, es la opción más rápida para aplicaciones que requieren tiempos de ejecución altos, como el sistema de visión estéreo para la navegación autónoma. Tanto el gráfico del tiempo de coincidencia promedio para la búsqueda de plantillas en la fila esperada como el gráfico de búsqueda en imagen completa indican que SoRA, SAD y LDS son los tres algoritmos más rápidos. Sin embargo, SoRA supera a los otros dos algoritmos con tiempos constantes inferiores a un milisegundo, lo que lo convierte en la mejor opción para la tarea de correspondencia de objetos en términos de velocidad. Además, si bien los resultados de RMSE para la búsqueda de imágenes de fila esperada muestran algunas diferencias entre SoRA y SAD, los dos algoritmos tienen el mismo valor para la plantilla de 20x20, lo que indica que SoRA puede lograr una precisión comparable con SAD en algunos casos y superar a LDS, BBS y Fast-Match. Por lo tanto, para aplicaciones donde la velocidad es una tarea de gran importancia, SoRA es una mejor opción que el algoritmo SAD, especialmente teniendo en cuenta su tiempo de procesamiento reducido para plantillas más grandes.

Estos experimentos demostraron la diferencia de rendimiento entre nuestro algoritmo y otros enfoques que utilizan tamaños de plantilla fijos. Sin embargo, en aplicaciones prácticas de correspondencia de objetos en un sistema de visión estéreo para navegación autónoma, SoRA recibe plantillas de objetos del algoritmo de detección de objetos con tamaños variables. Esto se puede observar en el experimento de correspondencia de objetos.

### 4.1.3. Resultados de correspondencia de objetos

La aplicación del algoritmo SoRA consiste en utilizarlo en un sistema de visión estéreo combinado con un algoritmo de detección de objetos (OD) para obtener información espacial sobre los objetos en el campo de visión. El experimento de correspondencia de objetos consiste en utilizar el algoritmo de detección de objetos (YOLOR) para encontrar objetos en la imagen izquierda, y luego encontrar el mismo objeto en la imagen derecha (Correspondencia de objetos). Esto se puede realizar repitiendo el uso de YOLOR en la imagen derecha o con un algoritmo de coincidencia de plantillas (TM) utilizando el objeto detectado en la imagen izquierda. Para aplicaciones de navegación autónoma, la correspondencia de objetos debe ser precisa y con tiempos de ejecución bajos. Como se muestra en la Sección 4.1.1, en la tabla 4.1 y en la tabla 4.2, SoRA puede cumplir con la tarea de correspondencia de objetos con un tiempo de ejecución más bajo que los otros métodos, lo cual es importante para la aplicación prevista. Para demostrar la precisión y velocidad del método propuesto, se comparan dos enfoques para realizar la tarea de correspondencia de objetos (OD-OD y OD-TM), ver la tabla 4.3, el segundo enfoque (OD-TM) también se realizó con el algoritmo SAD ya que fue el algoritmo que dio los mejores resultados de los métodos utilizados para comparar con SoRA.

TABLA 4.3: Enfoques comparados para encontrar la correspondencia de objetos en un sistema de visión estéreo

| Enfoque | Búsqueda en imagen izquierda | Búsqueda en imagen derecha |
|---|---|---|
| OD-OD | YOLOR | YOLOR |
| OD-TM | YOLOR | SoRA |
| OD-TM | YOLOR | SAD |

Dado que la búsqueda en la imagen izquierda es la misma para ambos enfoques, la comparación de los tiempos de ejecución se realizó únicamente para la búsqueda en la imagen derecha, donde los enfoques para localizar el objeto son diferentes. La tabla 4.4 muestra los resultados de los experimentos realizados con 83 objetos a encontrar en diversas escenas del conjunto de datos DrivingStereo. El tamaño de la plantilla utilizado para los algoritmos SoRA y SAD fue el mismo que el tamaño del cuadro delimitador del objeto detectado. Ambos algoritmos se ejecutaron en la misma computadora con un procesador Intel Core i7-6500U a 2.5 GHz, el entorno de ejecución para el algoritmo YOLOR es Python y el entorno de ejecución para el algoritmo propuesto es LabVIEW.

TABLA 4.4: Correspondencia de objetos en la imagen derecha realizada con YOLOR y SoRA.

| | RMSE | Tiempo de ejecución promedio | Coincidencia fallida |
|---|---|---|---|
| YOLOR | 2.3713 | 3386.35ms | 6 |
| SoRA | 1.3268 | 6.8691ms | 1 |
| SAD | 1.3353 | 60.3184ms | 1 |

Los resultados de la Tabla 4.4 muestran que el algoritmo SoRA es 8.78 veces más rápido que el algoritmo SAD con un RMSE similar (incluso ligeramente más bajo en el caso de SoRA). Los resultados del algoritmo SoRA mejoraron significativamente en comparación con los experimentos anteriores, esto se debe a que el algoritmo funciona mejor con una plantilla bien definida, como en el caso de la plantilla proporcionada por el algoritmo de detección de objetos. Además, se observa una reducción en las coincidencias fallidas en el algoritmo SoRA, igualando al algoritmo SAD en términos de precisión. Los resultados obtenidos demuestran que SoRA es la mejor opción para la tarea de correspondencia de objetos. Por otro lado, SoRA es 492.98 veces más rápido que el algoritmo YOLOR en la CPU, y tiene un RMSE un 55.92 % más bajo. Esta diferencia en el RMSE se debe a que, aunque el algoritmo YOLOR encontró el objeto en ambas imágenes, el cuadro delimitador puede variar de una imagen a otra, como se muestra en la Figura 4.5, donde se detecta un camión en las imágenes izquierda y derecha, pero el cuadro delimitador izquierdo es más grande que el derecho, ya que ocupa más espacio a la derecha. Este tipo de error de coincidencia genera errores considerables al calcular la distancia del objeto en sistemas de visión estéreo. Por esta razón, se requiere una coincidencia precisa entre los objetos para obtener la información espacial sobre el objeto en el campo de visión. Otra ventaja del enfoque propuesto es que el uso de una plantilla del tamaño del objeto detectado (como se presenta en la navegación autónoma) mejora los resultados del algoritmo SoRA, ofreciendo resultados equivalentes de coincidencias fallidas y RMSE al algoritmo SAD pero con una mejora significativa en la ejecución, debido al contraste generado entre el objeto y su entorno al momento de ejecutar la búsqueda de coincidencia de plantillas.

En resumen, los experimentos realizados demostraron que el algoritmo SoRA tiene una alta precisión y funciona más rápido que los algoritmos comparados, lo que lo hace adecuado para la aplicación prevista de detección espacial de objetos, ya que se ejecuta en menos de un milisegundo. Cabe destacar que el uso de plantillas proporcionadas por el algoritmo de detección de objetos mejoró significativamente los resultados de RMSE del algoritmo


FIGURA 4.5: Algoritmo YOLOR detectando un objeto en la imagen izquierda y derecha donde se puede apreciar la diferencia en las cajas de deteccion (especificamente al lado derecho de cada imagen)

propuesto. Esta mejora puede atribuirse al contraste generado por el objeto y su entorno, lo que proporciona una distinción más clara para el algoritmo SoRA durante el proceso de coincidencia. Por lo tanto, se puede concluir que entre los algoritmos de coincidencia de plantillas mencionados, SoRA es la mejor opción para la tarea de correspondencia de objetos, especialmente al utilizar plantillas proporcionadas por el algoritmo de detección de objetos. Aunque el algoritmo SAD puede parecer una alternativa viable, sus requisitos de tiempo de ejecución lo hacen inadecuado para la aplicación dada.

## 4.2. Experimento y resultados del método de calibración propuesto

El sistema de visión estéreo se probó en una configuración experimental como se ilustra en la figura 4.6. La figura muestra el sistema de visión estéreo prototipo, que consta de dos cámaras y una lente varifocal, conectadas a una computadora portátil que ejecuta el software de visión estéreo desarrollado. El software se utiliza para analizar las imágenes capturadas por las cámaras y generar mapas de profundidad del tablero de ajedrez. El objetivo del sistema es un patrón de tablero de ajedrez colocado frente al sistema. El tablero de ajedrez permite una comparación precisa entre los valores medidos por el sistema y los valores reales. Al medir la profundidad en cada intersección, se

puede validar el sistema y determinar su precisión. El patrón de tablero de ajedrez es un objetivo comúnmente utilizado en sistemas de visión estéreo, ya que proporciona un patrón conocido que puede identificarse fácilmente en las imágenes capturadas por las cámaras. La configuración experimental permite probar y evaluar el rendimiento del sistema de visión estéreo en condiciones controladas.

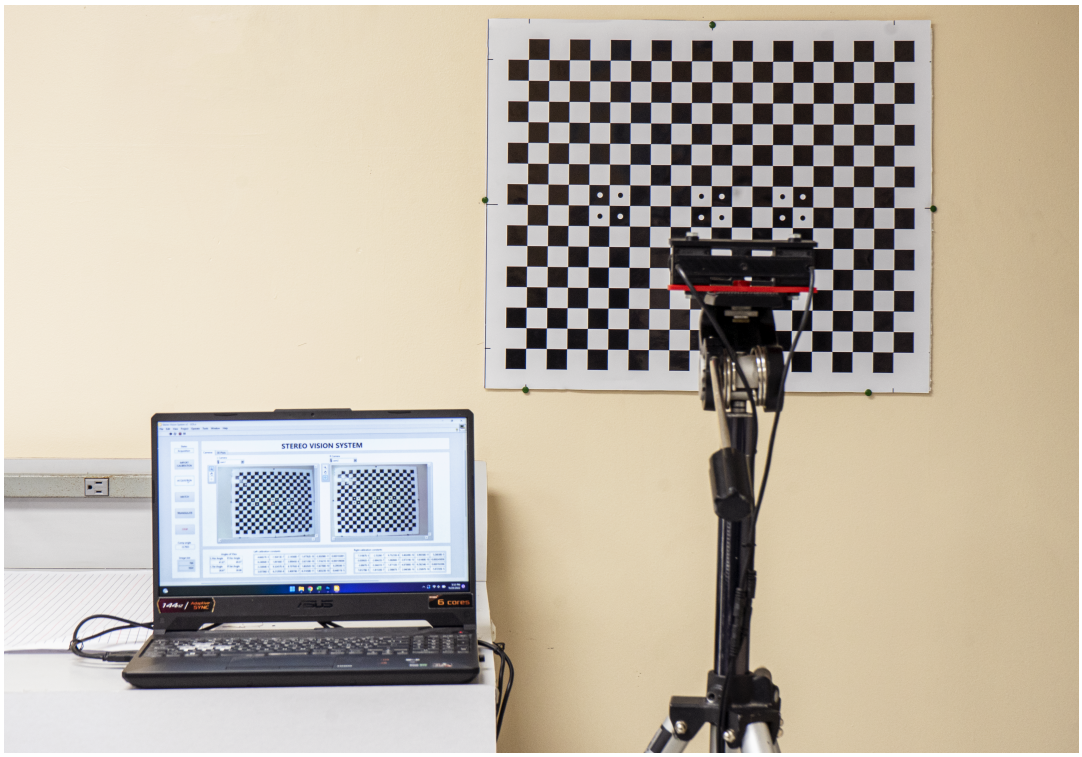



FIGURA 4.6: Configuración del experimento. Software del sistema de visión estéreo, sistema de visión estéreo y patrón.

Los puntos de superficie medidos para probar el sistema se ubicaron según se muestra en el diagrama de la figura 4.6. Se posicionó un patrón de tablero de ajedrez a 1000 mm en el eje $x$ con diferentes distancias en el eje $y$, manteniendo los puntos dentro del campo de visión de ambas cámaras. Este conjunto de puntos se repite en diferentes alturas (eje $z$) para abarcar todos los cuadrantes donde hay diferentes niveles de distorsión de la lente.

La figura 4.7 ilustra la vista superior de la configuración experimental. Los puntos se posicionaron para permanecer dentro del campo de visión de ambas cámaras. El conjunto de puntos utilizado para las pruebas incluye mediciones a diferentes alturas en el eje $z$ para capturar puntos en diferentes cuadrantes con diferentes niveles de distorsión. Esto permite evaluar la capacidad del sistema para medir la profundidad y el rendimiento en todos los

cuadrantes. Además, esta configuración experimental permite probar el sistema de visión estéreo en condiciones controladas y validar su rendimiento. Los puntos del patrón de tablero de ajedrez a diferentes alturas proporcionan un conjunto completo de mediciones que se pueden utilizar para evaluar la exactitud y precisión del sistema. La combinación de diferentes alturas y cuadrantes del patrón de tablero de ajedrez permite comprender mejor el rendimiento del sistema y los efectos de la distorsión de la lente en la precisión del sistema. El experimento se realizó de tres maneras diferentes: una sin calibración, otra utilizando el método propuesto y otra empleando el método OpenCV. Los tres experimentos se realizaron a la misma distancia, de manera consecutiva, sin modificar la configuración, como se muestra en la figura 4.7.

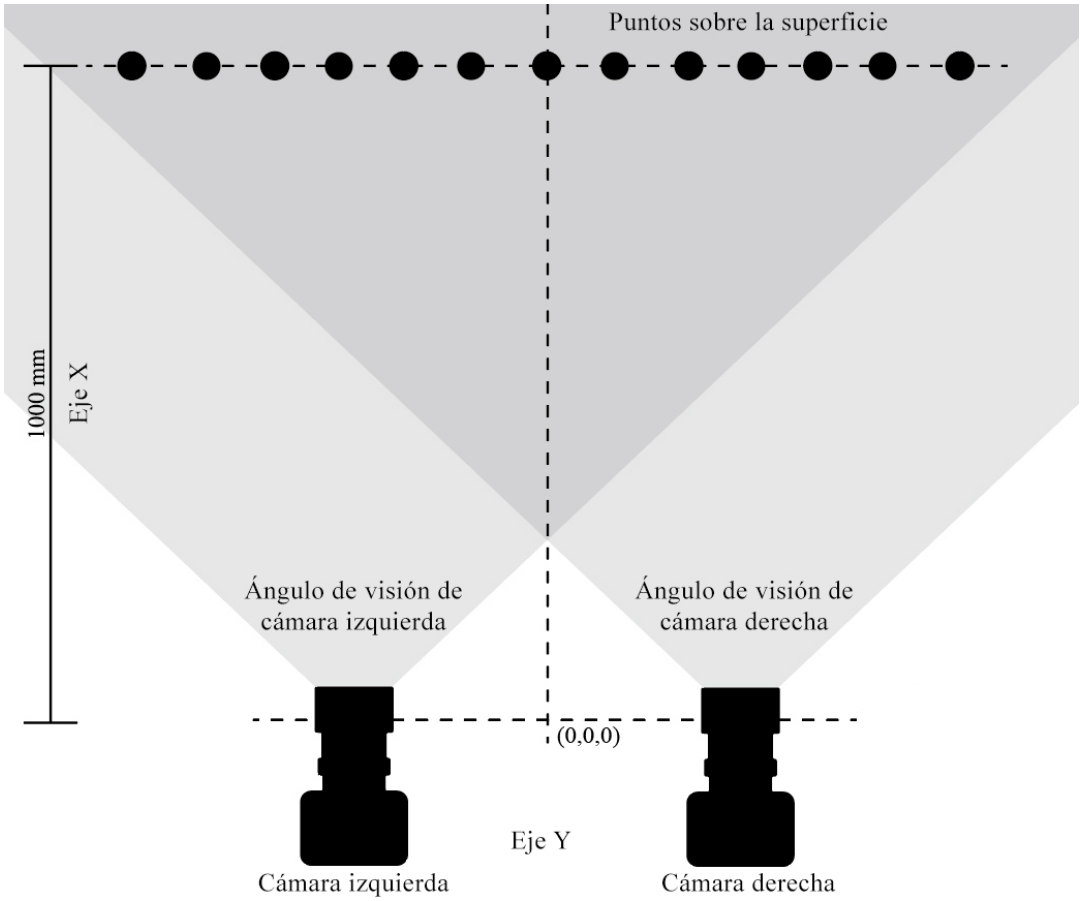



FIGURA 4.7: Diagrama de los puntos de superficie a medir (vista superior)

Las coordenadas 3D $X$, $Y$ y $Z$ de los puntos en la escena utilizando el método OpenCV se calcularon con los siguientes pasos: Calibración de las

Cámaras, donde ambas cámaras izquierda y derecha se calibraron para determinar sus parámetros intrínsecos y extrínsecos. Captura de Imágenes Estéreo, donde se capturaron imágenes estéreo simultáneamente utilizando las cámaras calibradas. Correspondencia de Puntos, donde se utilizó OpenCV para encontrar puntos correspondientes en las imágenes izquierda y derecha, obteniendo la disparidad $d$ entre los puntos. Cálculo de la Profundidad, donde la profundidad $X$ se calculó para cada punto utilizando la ecuación 4.1, con $X$ que representa la profundidad en centímetros (cm), $f$ denotando la longitud focal en píxeles (convertida de milímetros a píxeles), y $b$ representando la línea base estéreo. Finalmente, el Cálculo de Coordenadas calcula $Z$ y $Y$ utilizando las Ecuaciones 4.2 y 4.3, respectivamente, con left_point[0] representando las coordenadas $x$ y $y$ del punto en la imagen izquierda.

$$X = \frac{f \cdot b}{d} \tag{4.1}$$

$$Z = \frac{\text{left\_point}[0] \cdot X}{f} \tag{4.2}$$

$$Y = \frac{\text{left\_point}[0] \cdot X}{f} \tag{4.3}$$

Se realizó un análisis comparativo del rendimiento del sistema utilizando el método de calibración propuesto al obtener mediciones de los puntos de superficie especificados con y sin la técnica de calibración. Además, se realizaron las mismas mediciones utilizando la metodología de calibración empleada por la biblioteca OpenCV para sistemas de visión estéreo, con funciones como cv2.StereoCalibrate, cv.stereoRectify, cv.initUndistortRectifyMap y cv2.remap. En el proceso de calibración, se implementó el método de OpenCV siguiendo procedimientos estándar. Se llevaron a cabo múltiples sesiones de calibración utilizando distintas cantidades de imágenes, que iban desde 5 hasta 16 imágenes. Para el análisis comparativo con el método de calibración propuesto, se seleccionó el resultado de calibración obtenido con 12 imágenes, ya que demostró el mejor rendimiento. Las imágenes utilizadas en el proceso de calibración fueron seleccionadas cuidadosamente para abarcar diversos puntos de vista y ángulos, asegurando la robustez de la calibración. Este método de calibración de OpenCV es un método ampliamente utilizado para sistemas de visión estéreo en la literatura reciente, como se muestra en Zhong y Dong, 2015, Pomaska, 2019, Zoetgnandé et al., 2019 y Li, Papachristou y Weyer, 2018. Estas mediciones se realizaron a la misma distancia y posición simultáneamente. La Tabla 4.5 muestra los resultados de las cinco

TABLA 4.5: Las cinco medidas con menos error (valores en mm).

| Sin calibración | | | | OpenCV | | | | Método propuesto | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| X | Y | Z | X ERROR | X | Y | Z | X ERROR | X | Y | Z | X ERROR |
| 1140.38 | -46.81 | 15.90 | 140.38 | 958.65 | 95.18 | 216.50 | 41.35 | 1000.31 | 159.55 | -133.25 | 0.31 |
| 1140.82 | 21.17 | -17.49 | 140.82 | 957.83 | 185.41 | 188.00 | 42.17 | 999.67 | 185.12 | 13.94 | 0.33 |
| 1140.82 | 21.17 | -50.13 | 140.82 | 957.70 | 154.88 | 216.68 | 42.30 | 1000.55 | 218.10 | -163.22 | 0.55 |
| 1140.82 | 21.17 | -83.64 | 140.82 | 957.69 | 124.41 | 216.52 | 42.31 | 999.41 | 8.83 | 188.17 | 0.59 |
| 1141.07 | 20.38 | 16.70 | 141.07 | 957.44 | 94.60 | 186.43 | 42.56 | 999.41 | 8.83 | 158.74 | 0.59 |

TABLA 4.6: Las cinco medidas con menos precisión (valores en mm).

| Sin calibración | | | | OpenCV | | | | Método propuesto | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| X | Y | Z | X ERROR | X | Y | Z | X ERROR | X | Y | Z | X ERROR |
| 1229.44 | -331.91 | -274.35 | 229.44 | 904.23 | -271.03 | -184.63 | 95.77 | 978.59 | 71.87 | -216.23 | 21.41 |
| 1217.85 | -331.05 | -235.45 | 217.85 | 905.95 | -270.35 | -126.06 | 94.05 | 1017.92 | -288.66 | -170.42 | 17.92 |
| 1213.67 | -292.74 | -270.83 | 213.67 | 906.09 | -270.88 | -213.63 | 93.91 | 982.52 | 42.43 | -188.53 | 17.49 |
| 1213.35 | 295.23 | 219.70 | 213.35 | 906.48 | -270.55 | -155.07 | 93.52 | 982.74 | 41.76 | -130.21 | 17.26 |
| 1207.70 | -255.70 | -269.50 | 207.70 | 906.88 | -241.51 | -212.98 | 93.12 | 982.74 | 41.76 | -158.91 | 17.26 |

mejores mediciones sin calibración, con OpenCV y con el método propuesto. Además, la tabla 4.6 muestra los resultados de las cinco mediciones con menor precisión para cada punto de comparación.

A partir de estas tablas, se puede observar que el método propuesto resuelve el problema de calibración. La Tabla 4.5 muestra los resultados de las cinco mejores mediciones, donde se puede observar que el método propuesto reduce significativamente el error en comparación con el método de calibración de OpenCV. Lo mismo ocurre con los resultados en la Tabla 4.6, que muestra los resultados de las cinco mejores mediciones con mayor precisión, donde el método propuesto también reduce significativamente el error en las mediciones en comparación con el método de OpenCV. Por otro lado, los resultados de estas mediciones se presentan en varias formas gráficas en las Figuras 4.8, 4.9 y 4.10, donde cada color representa una altura diferente. Cada figura contiene tres representaciones gráficas; cada una muestra los resultados desde la misma perspectiva, una sin calibración, una con OpenCV y otra con el método propuesto. El gráfico de vista superior (Figura 4.8) muestra la profundidad desde la cámara hasta el patrón objetivo a lo largo del eje X y la medición horizontal a lo largo del eje Y, visto desde arriba. El gráfico de vista lateral (Figura 4.9) representa la medición vertical a lo largo de los ejes X y Z, visto desde una perspectiva lateral. Por último, el gráfico de vista isométrica en la Figura 4.10 presenta mediciones en los tres ejes, XYZ.

A partir de estos gráficos, se puede observar que el error se reduce significativamente al utilizar el método de calibración propuesto. Por otro lado, los gráficos de las mediciones sin calibración muestran una curvatura pronunciada que aumenta el error en la medición en términos de la profundidad, anchura y altura del tablero de ajedrez medido. Esto se debe principalmente a la distorsión generada por el arreglo óptico de las cámaras, que es más

pronunciada en los bordes de las imágenes. Por otro lado, las mediciones realizadas con el método de calibración de OpenCV mejoran significativamente el error y la curvatura. Además, el método propuesto supera a otros métodos en esta tarea al demostrar una corrección de curvatura significativa, lo que resulta en mediciones más cercanas al valor real de 1000 mm.

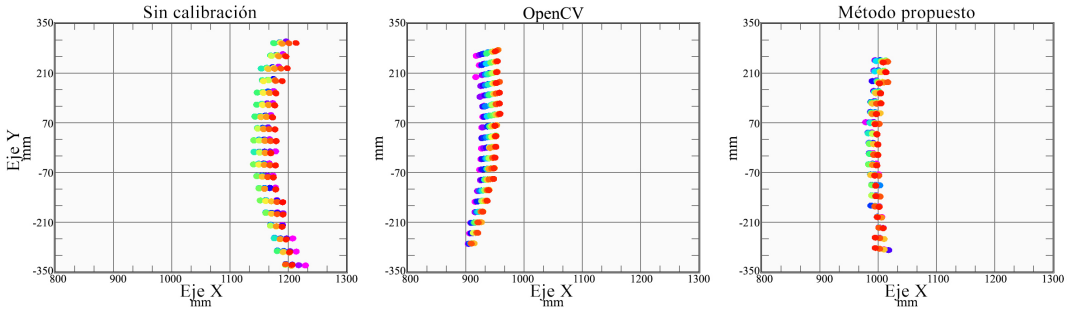



FIGURA 4.8: Resultados del sistema de visión estéreo, vista superior

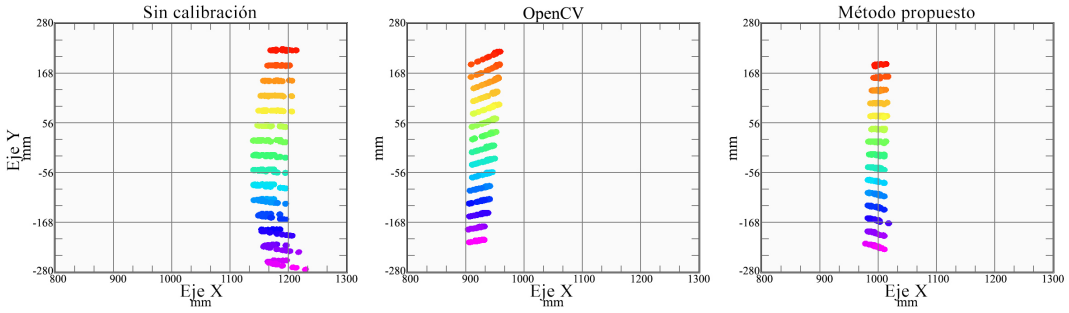



FIGURA 4.9: Resultados del sistema de visión estéreo, vista lateral

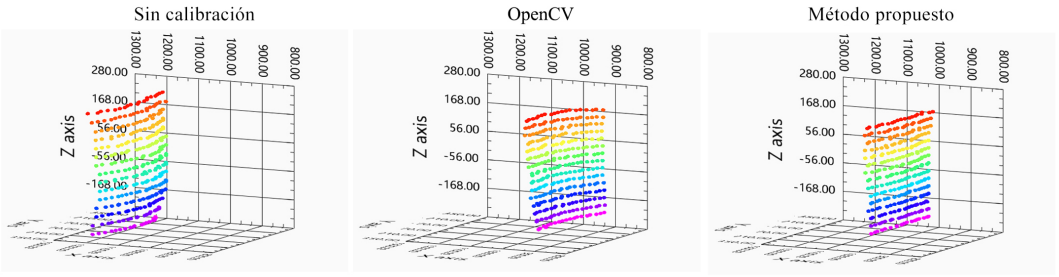



FIGURA 4.10: Resultados del sistema de visión estéreo, vista isométrica

Otro aspecto importante es examinar en qué medida los métodos propuestos preservan las proporciones del patrón del tablero de ajedrez visto

desde la perspectiva frontal. El patrón del tablero de ajedrez utilizado tiene dimensiones de 529 mm de ancho y 412 mm de alto. Una comparación de las mediciones de los cuatro lados (top, bottom, left y right) del tablero de ajedrez con respecto a sus proporciones se presenta en la tabla 4.7 en tres tamaños diferentes como se muestra en la figura 4.11. Los resultados demuestran que el método propuesto generalmente logra una mejor preservación de las proporciones del tablero de ajedrez.

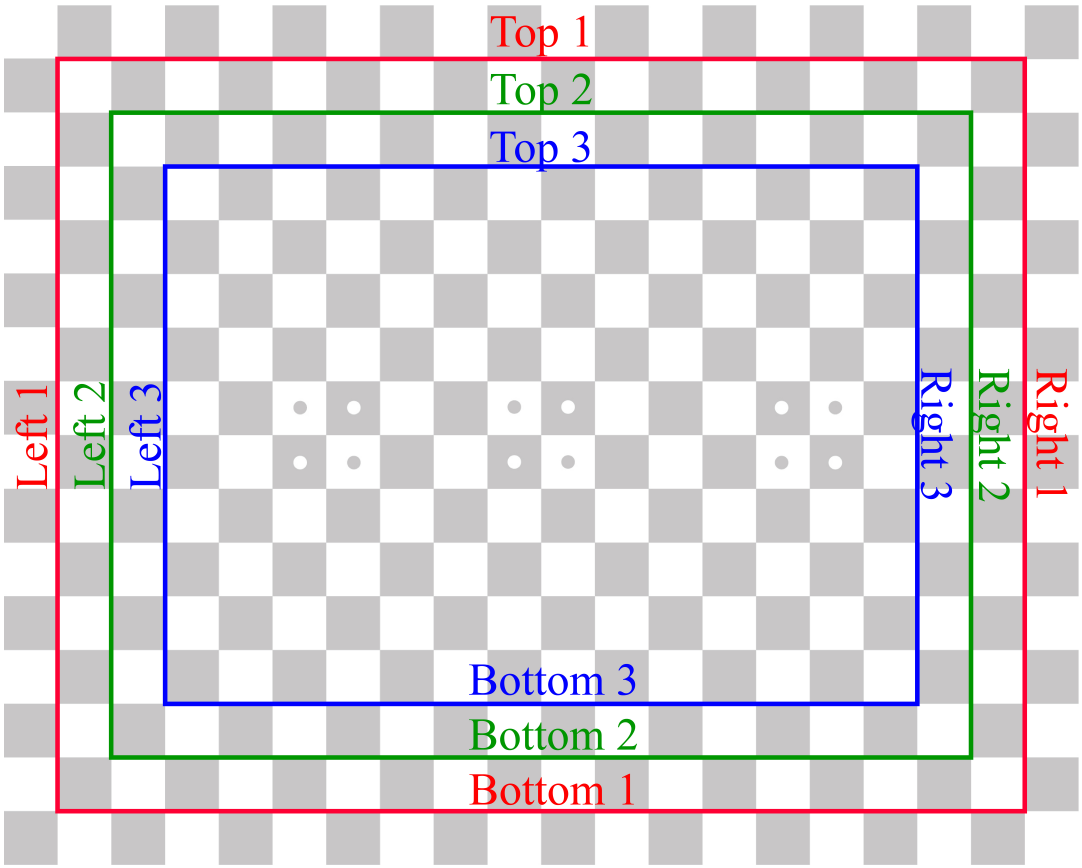



FIGURA 4.11: Representación de valores reales del patrón

La representación gráfica en la figura 4.12 muestra los resultados de las mediciones obtenidas desde la vista frontal, donde se utiliza una variación en el color de los puntos de datos para indicar el nivel de desviación. Este gráfico utiliza un código de color para representar el nivel de desviación, que va desde el verde para errores mínimos hasta el amarillo y naranja para aquellos cada vez más significativos. Además, el gráfico muestra cómo cada método mantiene las proporciones del patrón de tablero de ajedrez y el error de profundidad en cada punto.

Las figuras presentadas ayudan a identificar regiones caracterizadas por

TABLA 4.7: Resultados de las mediciones de proporciones del patrón (valores en mm).

| Lado del patrón | Valor real | Sin calibración | OpenCV | Método propuesto |
|---|---|---|---|---|
| Top 1 | 529 | 625.427 | 541.4218 | **522.697** |
| Bottom 1 | 529 | 630.64 | **529.9564** | 532.716 |
| Left 1 | 412 | 493.669 | 401.5648 | **412.194** |
| Right 1 | 412 | 473.692 | 423.3886 | **410.443** |
| Top 2 | 466 | 553.362 | 483.4338 | **467.988** |
| Bottom 2 | 466 | 553.203 | 477.3608 | **471.863** |
| Left 2 | 355 | 419.205 | 347.2456 | **353.161** |
| Right 2 | 355 | 402.582 | 363.1172 | **351.355** |
| Top 3 | 409 | 479.525 | 422.8615 | **409.14** |
| Bottom 3 | 409 | 483.221 | 418.188 | **412.168** |
| Left 3 | 291 | 348.253 | **293.1584** | 293.191 |
| Right 3 | 291 | 333.852 | 302.0254 | **291.629** |

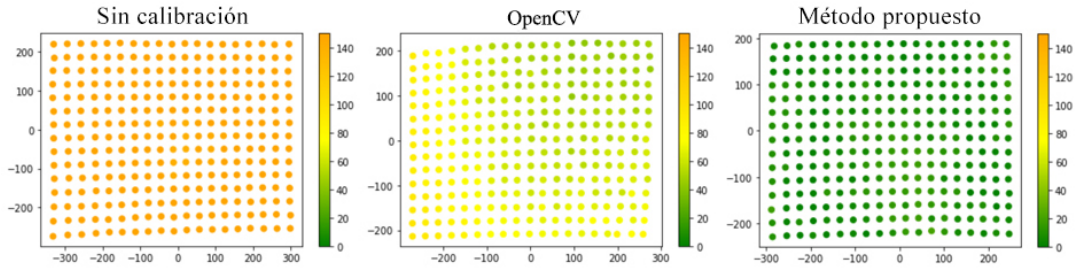



FIGURA 4.12: Gráfico de dispersión de mapa de calor de error, vista frontal.

un mayor grado de error, especialmente en el caso de mediciones no calibradas. Estas áreas se destacan específicamente en las esquinas de la imagen, donde la distorsión es más frecuente. Además, la representación gráfica de las mediciones calibradas con el método propuesto ilustra la efectividad de los algoritmos de corrección utilizados, ya que es evidente que el color verde predomina en la mayor parte de la imagen. Esto indica que las áreas de distorsión más significativa han sido corregidas de manera efectiva, lo que resulta en mediciones más precisas con el método propuesto.

La Tabla 4.8 muestra el error absoluto medio (MAE), el error cuadrático medio (RMSE) y la desviación estándar (STD) de las mediciones tomadas a una distancia de 1000 mm en el eje $X$. La tabla compara el MSE y el STD entre el sistema que utiliza el método propuesto de calibración de la cámara, el método OpenCV y sin ningún método de calibración. Los resultados muestran una mejora del 61.74 % en el MAE, 61.18 % en el RMSE y 24.1 % en el STD. Por otro lado, el método propuesto muestra una mejora del 95.9 % en el

TABLA 4.8: Comparación de los métodos de calibración de cámaras de visión estéreo: Evaluación del rendimiento con sistema sin calibración, OpenCV y método propuesto.

| Tipo de medición | Sin calibración | OpenCV | Método propuesto |
|---|---|---|---|
| MAE | 170.53 | 65.24 | **6.99** |
| RMSE | 171.40 | 66.53 | **8.45** |
| STD | 17.23 | 13.08 | **4.74** |

MAE, 95.07 % en el RMSE y 72.49 % en el STD.

## 4.3. Experimento y resultados de la combinación del sistema de visión estereoscópica y algoritmo de detección de objetos

Una vez que el sistema realiza la coincidencia de objetos de manera rápida y precisa, y que los parámetros de calibración intrínsecos y extrínsecos fueron calibrados correctamente, el sistema está listo para llevar a cabo experimentos que combinan el sistema de visión estereoscópica con el algoritmo de detección de objetos YOLOR. En la Figura 4.13, se puede apreciar la configuración del experimento, que incluye el software del sistema de visión estereoscópica con el algoritmo de detección de objetos ejecutándose con Python 3.10 en una computadora Lenovo Legión S7. Esta computadora cuenta con un procesador Intel Core i7 de 12va generación, una tarjeta gráfica Nvidia RTX 3070 y 16 GB de memoria RAM. Además, la Figura también presenta el prototipo del sistema de visión estereoscópica mostrado en secciones anteriores y el sujeto de prueba, que se posicionará a diferentes distancias con respecto al sistema de visión.

Una de las complicaciones al intentar comparar los resultados obtenidos con los valores reales de la posición del objeto $x$, $y$ y $z$ es que al intentar alinear el prototipo del sistema de visión con un plano $XY$, cualquier error mínimo de alineación representará una gran discrepancia conforme el objeto a detectar se aleje del sistema. Para poder realizar una comparación adecuada entre los valores medidos del sistema y el valor real, se optó por posicionar al sujeto a ciertas distancias fijas, como se observa en la figura 4.14. Se tomaron mediciones a distancias de 5, 10, 15 y 20 metros con respecto al sistema de visión. Para comparar los valores reales con los medidos, se calculó la distancia euclidiana con los valores $x$ y $y$ de las mediciones dadas por el sistema de visión.


FIGURA 4.13: Configuración experimental que incluye el software del sistema de estéreo visión y detección de objetos, el prototipo del sistema de visión estéreo y la presencia del sujeto a ser detectado.

La gráfica de la Figura 4.15 muestra un resumen de los resultados obtenidos de las mediciones a diferentes distancias en una gráfica de velas japonesas. Nótese que para cada distancia aparecen tres columnas; una representa el objeto a la distancia indicada, pero posicionado al lado izquierdo del sistema; otra representa el objeto posicionado al centro del sistema y la última representa el objeto posicionado a la derecha del sistema. Las barras de color azul indican el valor real del objeto en milímetros. El punto amarillo representa el promedio de las mediciones, el punto negro representa el valor máximo de las mediciones y el rombo negro representa el valor mínimo de las mediciones efectuadas.

La Tabla 4.9 presenta los resultados generales obtenidos a distancias de 5, 10, 15 y 20 metros, expresados en milímetros. Se compararon los valores

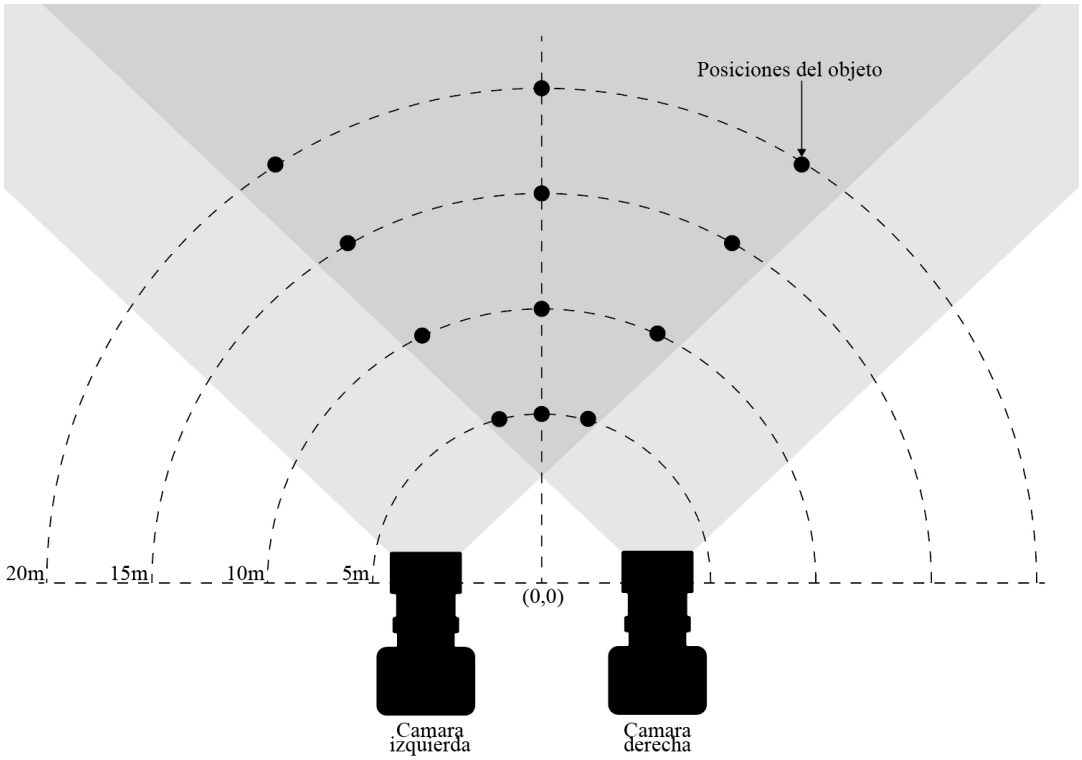



FIGURA 4.14: Diagrama ilustrativo de las mediciones efectuadas a distancias de 5, 10, 15 y 20 metros.

reales con los valores promedio medidos por el sistema de visión, revelando ciertas discrepancias significativas. A 5 metros, el sistema exhibió un pequeño error promedio de 117.28 milímetros con una desviación estándar de 170.56 milímetros. Sin embargo, a medida que la distancia aumentaba, la discrepancia se ampliaba. A 10 metros, el error promedio se elevó a 795.56 milímetros con una desviación estándar de 620.75 milímetros. A 15 y 20 metros, los errores promedio fueron de 144.73 y 2277.08 milímetros, respectivamente, acompañados de desviaciones estándar (1457.20 y 1382.88 milímetros). Se observa un aumento en el error promedio y la desviación estándar a medida que la distancia aumenta en la mayoría de los casos. Esto se debe a que cualquier variación en un píxel representa un cambio en los ángulos para la triangulación, afectando especialmente el cálculo de las coordenadas y la profundidad. En la Tabla 4.9, la única excepción parece ser el error promedio a 15 metros, que es similar incluso al de 5 metros. Esto podría deberse a ocasiones en las que un píxel coincide con el cálculo para esa distancia, a pesar de la falta de resolución para esa distancia.

Estos resultados destacan la influencia significativa de la distancia en la

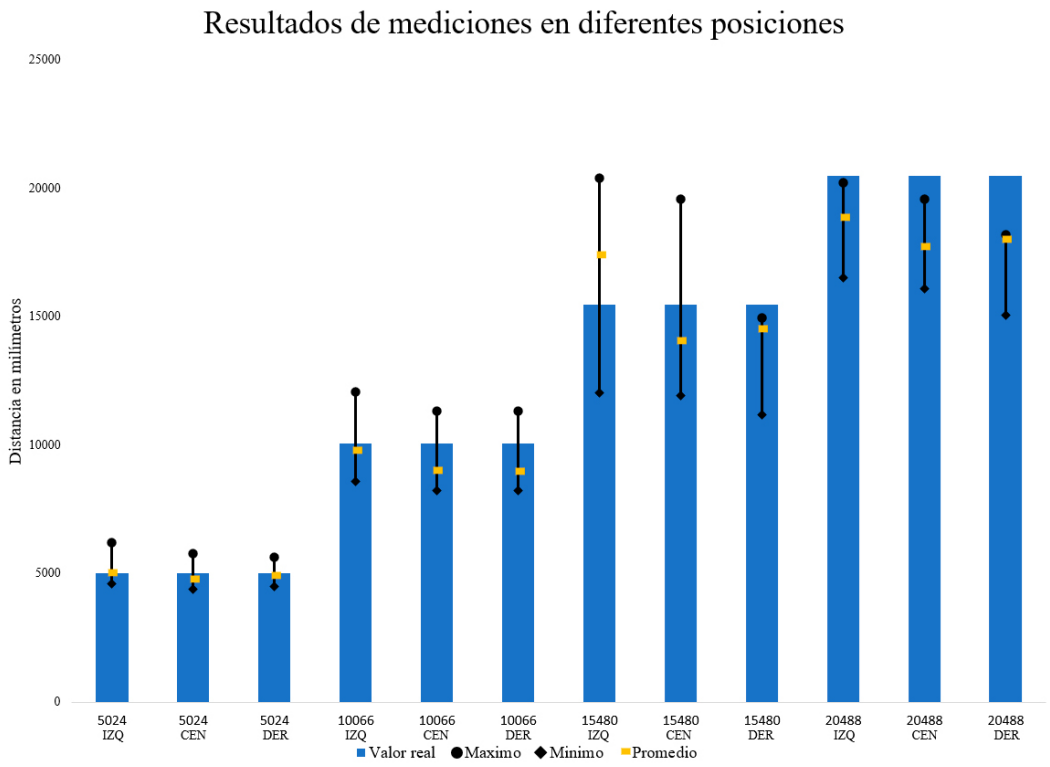



FIGURA 4.15: Resultados en diferentes posiciones a 5, 10, 15 y 20 metros con el objeto a la izquierda, centro y derecha.

TABLA 4.9: Resultados generales a 5, 10, 15 y 20 metros. Unidades en milímetros.

| Valor real | Promedio | Error promedio | Desviación estandar |
|---|---|---|---|
| 5024 | 4906.72 | 117.28 | 170.56 |
| 10066 | 9270.44 | 795.56 | 620.75 |
| 15480 | 15335.27 | 144.73 | 1457.20 |
| 20488 | 18210.92 | 2277.08 | 1382.88 |

precisión del sistema, lo cual tiene sentido, ya que a medida que un objeto se aleja del sistema, cualquier variación en un pixel, se traduce en una variación de un ángulo para la triangulación resultando en variaciones más grandes en cuanto a la distancia. En situaciones donde la precisión extrema no es indispensable, como la detección de objetos lejanos, el sistema puede proporcionar datos útiles para la toma de decisiones, sin comprometer significativamente la eficacia global del vehículo autónomo. Esta capacidad de distinguir la utilidad de las mediciones en función de la distancia demuestra la adaptabilidad del sistema a diversas condiciones, garantizando una toma de decisiones eficiente en entornos dinámicos y desafiantes. Además, para

objetos más cercanos, donde el sistema presenta una mayor precisión, la información se vuelve aún más valiosa para un vehículo autónomo. Con esta precisión, el sistema puede tomar decisiones importantes, como detenerse, avanzar o evadir el objeto en cuestión, lo que añade una capa adicional de seguridad y eficiencia en la conducción autónoma en entornos cercanos. Esta combinación de capacidades, que se adapta tanto a distancias cortas como largas, refuerza la versatilidad y utilidad del sistema en diversas situaciones de navegación.

# Capítulo 5

# Conclusiones

La investigación abordó los desafíos en la navegación autónoma, centrándose en la combinación de un sistema de visión estereoscópica y un algoritmo de detección de objetos para mejorar la detección espacial y la navegación de vehículos autónomos. Los antecedentes revisados destacaron la diversidad de enfoques previos, desde sistemas de múltiples cámaras hasta LiDAR y radar, subrayando la importancia de encontrar soluciones eficientes y adaptables.

## 5.1. Conclusiones generales

El planteamiento del problema resaltó la importancia de la visión artificial, especialmente la visión estereoscópica, para superar las limitaciones de métodos existentes como LiDAR y sensores ultrasónicos. La propuesta de utilizar visión estereoscópica y detección de objetos se justificó como un enfoque prometedor, respaldado por trabajos previos en la detección de objetos mediante visión estereoscópica en contextos de navegación autónoma y aplicaciones diversas.

La justificación y uso de los resultados subrayaron la importancia de lograr una combinación precisa para aplicaciones en tiempo real de navegación autónoma. El algoritmo propuesto de coincidencia de plantillas (SoRA) y el método de calibración innovador demostraron mejoras sustanciales en la generación de nubes de puntos 3D, fundamentales para la percepción del entorno.

## 5.2. Cumplimiento de objetivos

La presente investigación abordó el desarrollo de un sistema para la detección espacial de objetos en tiempo real mediante la combinación de un

sistema de visión estereoscópica y un algoritmo de detección de objetos. Cada objetivo específico se abordó meticulosamente, y los resultados obtenidos demuestran el éxito en el cumplimiento de dichos objetivos.

El objetivo general de desarrollar un sistema para la detección espacial de objetos se logró mediante una metodología que incluyó el desarrollo y evaluación del algoritmo SoRA, el método de calibración, y la integración con el algoritmo de detección de objetos YOLOR. Los experimentos destacaron la eficacia de la combinación del sistema de visión estereoscópica y YOLOR, especialmente en distancias cortas (5-15m), proporcionando información valiosa para la toma de decisiones en vehículos autónomos.

Se logró construir un prototipo del sistema de visión estereoscópica con una geometría coplanar y una separación de 100 mm entre las cámaras. Este diseño proporcionó la base necesaria para la realización de experimentos y pruebas que validaran la eficacia del sistema.

El algoritmo propuesto, SoRA, sobresale al ofrecer una velocidad superior y una precisión competitiva en comparación con algoritmos existentes. En la búsqueda de imagen completa, SoRA superó a SAD, BBS y Fast-Match, con tiempos de ejecución notoriamente más bajos. Por ejemplo, para una plantilla de 50x50 píxeles, SoRA tuvo un tiempo promedio de 238.30 ms, mientras que SAD requirió 2256.98 ms y Fast-Match 89056.71 ms. Asimismo, en la búsqueda de fila esperada, SoRA mantuvo su superioridad, con un tiempo menor a 1 ms, el cual es significativamente inferior a SAD, LDS, BBS y Fast-Match. En la correspondencia de objetos, SoRA destacó por su rapidez y precisión, superando a SAD y YOLOR en tiempos de ejecución y error cuadrático medio. SoRA tuvo un tiempo promedio de ejecución de 6.8691 ms, mientras que SAD requirió 60.3184 ms y YOLOR 3386.35 ms. Estos resultados respaldan la eficiencia SoRA, especialmente en aplicaciones de visión estéreo para la navegación autónoma, donde la velocidad y precisión son de suma importancia. La implementación de plantillas proporcionadas por el algoritmo de detección de objetos mejoró aún más el rendimiento de SoRA, consolidándolo como la elección óptima para esta tarea.

La investigación resultó en un método innovador de calibración de cámaras que optimiza parámetros intrínsecos y extrínsecos del sistema. Este enfoque se destacó por su eficiencia, permitiendo un procesamiento de imágenes rápido y mostrando robustez ante variaciones de iluminación. Los experimentos, realizados en una configuración experimental específica, consistieron en pruebas del sistema de visión estéreo mediante la colocación de un patrón de tablero de ajedrez a diversas alturas. Estos experimentos revelaron

mejoras sustanciales en la precisión en comparación con métodos convencionales. El análisis comparativo de rendimiento, evaluando mediciones de puntos de superficie en diferentes configuraciones, demostró que el método propuesto supera a la biblioteca OpenCV. El método propuesto muestra una mejora del 95.9 % en el MAE, 95.07 % en el RMSE y 72.49 % en el STD con respecto al sistema sin calibrar. Estas mejoras cuantitativas respaldan la eficacia del método propuesto, proporcionando mediciones más precisas y consistentes en diversas condiciones.

La integración del sistema de visión estereoscópica con el algoritmo de detección de objetos YOLOR fue exitosa. Este paso fue esencial para la realización de la tarea de detección espacial de objetos, aprovechando la capacidad de YOLOR para realizar detecciones eficientes en un solo paso a través de la red neuronal.

En el análisis de resultados, se observó que, a pesar de las discrepancias entre los valores reales y medidos, el sistema demostró adaptabilidad a diversas condiciones. La capacidad de discernir la utilidad de las mediciones en función de la distancia refuerza la versatilidad del sistema, destacando su eficacia tanto en entornos cercanos como lejanos.

Se llevaron a cabo experimentos detallados para comparar el desempeño del sistema en diferentes condiciones y distancias. Aunque se observaron discrepancias entre los valores reales y medidos, especialmente a mayores distancias, esto no afecta la aplicación en navegación autónoma ya que la precisión obtenida a grandes distancias es suficiente para la toma de decisiones. Por otro lado, a cortas distancias se tiene una mayor precisión, lo cual es indispensable en estos casos para poder, evitar alguna colisión, detenerse, avanzar, etc. La capacidad de discernir la utilidad de las mediciones en función de la distancia resaltó la versatilidad del sistema en entornos dinámicos y desafiantes.

## 5.3. Consideraciones finales

Esta investigación no solo cumplió con sus objetivos específicos sino que también ofreció contribuciones significativas al campo de la navegación autónoma. La combinación de un sistema de visión estereoscópica eficiente con un algoritmo de detección de objetos rápido y preciso proporciona una herramienta valiosa para mejorar la percepción y comprensión del entorno por parte de los sistemas autónomos.

En resumen, la investigación logró desarrollar un sistema efectivo para la detección espacial de objetos en tiempo real, combinando la visión estereoscópica con un algoritmo de detección de objetos. Aunque se identificaron desafíos y discrepancias en los resultados, la adaptabilidad del sistema y su capacidad para proporcionar información valiosa en diversas situaciones demuestran su potencial para mejorar la seguridad y eficiencia de la navegación autónoma. Es importante destacar que una de las limitaciones identificadas es que la resolución de las mediciones se ve afectada por la resolución de las imágenes, lo que sugiere una oportunidad para futuras investigaciones en este campo con el fin de mejorar la precisión y versatilidad del sistema.

# Bibliografía

Badue, Claudine et al. (2021). «Self-driving cars: A survey». En: *Expert Systems with Applications* 165, pág. 113816.

Bartl, Vojtěch y Adam Herout (2019). «Optinopt: Dual optimization for automatic camera calibration by multi-target observations». En: *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, págs. 1-8.

Básaca-Preciado, Luis C et al. (2014). «Optical 3D laser measurement system for navigation of autonomous mobile robot». En: *Optics and Lasers in Engineering* 54, págs. 159-169.

Bay, Herbert, Tinne Tuytelaars y Luc Van Gool (2006). «Surf: Speeded up robust features». En: *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer, págs. 404-417.

Bjerkeng, Magnus et al. (2021). «ROV navigation in a fish cage with laser-camera triangulation». En: *Journal of Marine Science and Engineering* 9.1, pág. 79.

Bochkovskiy, Alexey, Chien-Yao Wang y Hong-Yuan Mark Liao (2020). «Yolov4: Optimal speed and accuracy of object detection». En: *arXiv preprint arXiv:2004.10934*.

Carion, Nicolas et al. (2020). «End-to-end object detection with transformers». En: *European conference on computer vision*. Springer, págs. 213-229.

Chang, Ting-Wei, Wei-Cheng Wang y Rongshun Chen (2021). «Intelligent Control System to Irrigate Orchids Based on Visual Recognition and 3D Positioning». En: *Applied Sciences* 11.10, pág. 4531.

Chen, Bao Xin, Raghavender Sahdev y John K Tsotsos (2017). «Integrating stereo vision with a CNN tracker for a person-following robot». En: *Computer Vision Systems: 11th International Conference, ICVS 2017, Shenzhen, China, July 10-13, 2017, Revised Selected Papers 11*. Springer, págs. 300-313.

Chen, Jiun-Hung, Chu-Song Chen y Yong-Sheng Chen (2003). «Fast algorithm for robust template matching with M-estimators». En: *IEEE Transactions on signal processing* 51.1, págs. 230-243.

Chen, Xiaozhi et al. (2017). «3d object proposals using stereo imagery for accurate object class detection». En: *IEEE transactions on pattern analysis and machine intelligence* 40.5, págs. 1259-1272.

Coenen, Max y Franz Rottensteiner (2021). «Pose estimation and 3D reconstruction of vehicles from stereo-images using a subcategory-aware shape prior». En: *ISPRS Journal of Photogrammetry and Remote Sensing* 181, págs. 27-47.

Cronk, Simon, Clive Fraser y Harry Hanley (2006). «Automated metric calibration of colour digital cameras». En: *The photogrammetric record* 21.116, págs. 355-372.

Dai, Deyun et al. (2021). «A review of 3D object detection for autonomous driving of electric vehicles». En: *World Electric Vehicle Journal* 12.3, pág. 139.

Dai, Jifeng, Kaiming He y Jian Sun (2016). «Instance-aware semantic segmentation via multi-task network cascades». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 3150-3158.

Dai, Jifeng et al. (2016). «R-fcn: Object detection via region-based fully convolutional networks». En: *Advances in neural information processing systems* 29.

Dall'Asta, Elisa y Riccardo Roncella (2014). «A comparison of semiglobal and local dense matching algorithms for surface reconstruction». En: *The international archives of the photogrammetry, remote sensing and spatial information sciences* 40, págs. 187-194.

Garrido-Jurado, Sergio et al. (2014). «Automatic generation and detection of highly reliable fiducial markers under occlusion». En: *Pattern Recognition* 47.6, págs. 2280-2292.

Geiger, Andreas et al. (2013). «Vision meets robotics: The kitti dataset». En: *The International Journal of Robotics Research* 32.11, págs. 1231-1237.

Girshick, Ross (2015). «Fast r-cnn». En: *Proceedings of the IEEE international conference on computer vision*, págs. 1440-1448.

Girshick, Ross et al. (2015). «Region-based convolutional networks for accurate object detection and segmentation». En: *IEEE transactions on pattern analysis and machine intelligence* 38.1, págs. 142-158.

Hariharan, Bharath et al. (2014). «Simultaneous detection and segmentation». En: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*. Springer, págs. 297-312.

— (2015). «Hypercolumns for object segmentation and fine-grained localization». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 447-456.

He, Kaiming et al. (2015). «Spatial pyramid pooling in deep convolutional networks for visual recognition». En: *IEEE transactions on pattern analysis and machine intelligence* 37.9, págs. 1904-1916.

He, Kaiming et al. (2017). «Mask r-cnn». En: *Proceedings of the IEEE international conference on computer vision*, págs. 2961-2969.

Huo, Junzhou et al. (2022). «Feature points extraction of defocused images using deep learning for camera calibration». En: *Measurement* 188, pág. 110563.

Kanade, Takeo et al. (1995). «Development of a video-rate stereo machine». En: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 3. IEEE, págs. 95-100.

Kang, Kai et al. (2017). «T-cnn: Tubelets with convolutional neural networks for object detection from videos». En: *IEEE Transactions on Circuits and Systems for Video Technology* 28.10, págs. 2896-2907.

Kapp, Sebastian et al. (2021). «Arett: Augmented reality eye tracking toolkit for head mounted displays». En: *Sensors* 21.6, pág. 2234.

Karpathy, Andrej y Li Fei-Fei (2015). «Deep visual-semantic alignments for generating image descriptions». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 3128-3137.

Khamis, Sameh et al. (2018). «StereoNet: Guided Hierarchical Refinement for Real-Time Edge-Aware Depth Prediction». En: *CoRR* abs/1807.08865. arXiv: 1807.08865. URL: http://arxiv.org/abs/1807.08865.

Königshof, Hendrik, Niels Ole Salscheider y Christoph Stiller (2019). «Realtime 3d object detection for automated driving using stereo vision and semantic information». En: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, págs. 1405-1410.

Konolige, Kurt et al. (2008). «Outdoor mapping and navigation using stereo vision». En: *Experimental Robotics: The 10th International Symposium on Experimental Robotics*. Springer, págs. 179-190.

Korman, Simon et al. (2013). «Fast-match: Fast affine template matching». En: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, págs. 2331-2338.

Krizhevsky, Alex, Ilya Sutskever y Geoffrey E Hinton (2012). «Imagenet classification with deep convolutional neural networks». En: *Advances in neural information processing systems* 25.

Kuhn, Peter M (2013). *Algorithms, complexity analysis and VLSI architectures for MPEG-4 motion estimation*. Springer Science & Business Media.

Law, Hei y Jia Deng (2018). «Cornernet: Detecting objects as paired keypoints». En: *Proceedings of the European conference on computer vision (ECCV)*, págs. 734-750.

Li, Jing et al. (2021). «Openstreetmap-based autonomous navigation for the four wheel-legged robot via 3d-lidar and ccd camera». En: *IEEE Transactions on Industrial Electronics* 69.3, págs. 2708-2717.

Li, Yaqi, Christos Papachristou y Daniel Weyer (2018). «Road pothole detection system based on stereo vision». En: *NAECON 2018-IEEE National Aerospace and Electronics Conference*. IEEE, págs. 292-297.

Li, Zeming et al. (2017). «Light-head r-cnn: In defense of two-stage object detector». En: *arXiv preprint arXiv:1711.07264*.

Lin, Huei-Yung, Chun-Lung Tsai et al. (2021). «Depth measurement based on stereo vision with integrated camera rotation». En: *IEEE Transactions on Instrumentation and Measurement* 70, págs. 1-10.

Lin, Tsung-Yi et al. (2017a). «Feature pyramid networks for object detection». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 2117-2125.

Lin, Tsung-Yi et al. (2017b). «Focal loss for dense object detection». En: *Proceedings of the IEEE international conference on computer vision*, págs. 2980-2988.

Liu, Li et al. (2020). «Deep learning for generic object detection: A survey». En: *International journal of computer vision* 128, págs. 261-318.

Liu, Wei et al. (2016). «Ssd: Single shot multibox detector». En: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, págs. 21-37.

Lowe, David G (1999). «Object recognition from local scale-invariant features». En: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee, págs. 1150-1157.

Meliones, Apostolos, Costas Filios y Jairo Llorente (2022). «Reliable ultrasonic obstacle recognition for outdoor blind navigation». En: *Technologies* 10.3, pág. 54.

Moru, Desmond K y Diego Borro (2020). «A machine vision algorithm for quality control inspection of gears». En: *The International Journal of Advanced Manufacturing Technology* 106, págs. 105-123.

Nair, Dinesh y Lothar Wenzel (1999). «Image processing and low-discrepancy sequences». En: *Advanced Signal Processing Algorithms, Architectures, and Implementations IX*. Vol. 3807. SPIE, págs. 102-111.

Oron, Shaul et al. (2017). «Best-buddies similarity—Robust template matching using mutual nearest neighbors». En: *IEEE transactions on pattern analysis and machine intelligence* 40.8, págs. 1799-1813.

Ouyang, Wanli et al. (2011). «Performance evaluation of full search equivalent pattern matching algorithms». En: *IEEE transactions on pattern analysis and machine intelligence* 34.1, págs. 127-143.

Ouyang, Wanli et al. (2012). «Performance Evaluation of Full Search Equivalent Pattern Matching Algorithms». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.1, págs. 127-143. DOI: 10.1109/TPAMI.2011.106.

Pele, Ofir y Michael Werman (2008). «Robust real-time pattern matching using bayesian sequential hypothesis testing». En: *IEEE transactions on pattern analysis and machine intelligence* 30.8, págs. 1427-1443.

Pérez, Luis et al. (2016). «Robot guidance using machine vision techniques in industrial environments: A comparative review». En: *Sensors* 16.3, pág. 335.

Po, Lai-Man y Kai Guo (2007). «Transform-Domain Fast Sum of the Squared Difference Computation for H.264/AVC Rate-Distortion Optimization». En: *IEEE Transactions on Circuits and Systems for Video Technology* 17.6, págs. 765-773. DOI: 10.1109/TCSVT.2007.896663.

Pomaska, G (2019). «Stereo vision applying opencv and raspberry pi». En: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 42, págs. 265-269.

Ramírez-Hernández, Luis R et al. (2020). «Improve three-dimensional point localization accuracy in stereo vision systems using a novel camera calibration method». En: *International Journal of Advanced Robotic Systems* 17.1, pág. 1729881419896717.

Real, Oscar Real et al. (2019). «Surface measurement techniques in machine vision: Operation, applications, and trends». En: *Optoelectronics in machine vision-based theories and applications*. IGI global, págs. 79-104.

Real-Moreno, Oscar et al. (2022). «A Quadrant Approach of Camera Calibration Method for Depth Estimation Using a Stereo Vision System». En: *IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, págs. 1-6.

Redmon, Joseph y Ali Farhadi (2017). «YOLO9000: better, faster, stronger». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 7263-7271.

— (2018). «Yolov3: An incremental improvement». En: *arXiv preprint arXiv:1804.02767*.

Redmon, Joseph et al. (2016). «You only look once: Unified, real-time object detection». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 779-788.

Ren, Shaoqing et al. (2015). «Faster r-cnn: Towards real-time object detection with region proposal networks». En: *Advances in neural information processing systems* 28.

Rodríguez-Quiñonez, JC et al. (2014). «Improve 3D laser scanner measurements accuracy using a FFBP neural network with Widrow-Hoff weight/bias learning function». En: *Opto-Electronics Review* 22, págs. 224-235.

Rouveure, Raphaël et al. (2019). «Robot localization and navigation with a ground-based microwave radar». En: *2019 International Radar Conference (RADAR)*. IEEE, págs. 1-4.

Sarvaiya, Jignesh N, Suprava Patnaik y Salman Bombaywala (2009). «Image registration by template matching using normalized cross-correlation». En: *2009 international conference on advances in computing, control, and telecommunication technologies*. IEEE, págs. 819-822.

Tankovich, Vladimir et al. (2021). «Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching». En: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, págs. 14362-14372.

Tao, Chongben et al. (2021). «Stereo priori RCNN based car detection on point level for autonomous driving». En: *Knowledge-Based Systems* 229, pág. 107346.

Terven, Juan y Diana Cordova-Esparza (2023). «A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond». En: *arXiv preprint arXiv:2304.00501*.

Uijlings, Jasper RR et al. (2013). «Selective search for object recognition». En: *International journal of computer vision* 104, págs. 154-171.

Wang, Chien-Yao, Alexey Bochkovskiy y Hong-Yuan Mark Liao (2023). «YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors». En: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, págs. 7464-7475.

Wang, Chien-Yao, I-Hau Yeh y Hong-Yuan Mark Liao (2021). «You only learn one representation: Unified network for multiple tasks». En: *arXiv preprint arXiv:2105.04206*.

Wang, Huai-Mu, Huei-Yung Lin y Chin-Chen Chang (2021). «Object detection and depth estimation approach based on deep convolutional neural networks». En: *Sensors* 21.14, pág. 4755.

Wu, Qi et al. (2017). «Image captioning and visual question answering based on attributes and external knowledge». En: *IEEE transactions on pattern analysis and machine intelligence* 40.6, págs. 1367-1381.

Wu, Yue y Qiang Ji (2019). «Facial landmark detection: A literature survey». En: *International Journal of Computer Vision* 127, págs. 115-142.

Xu, Kelvin et al. (2015). «Show, attend and tell: Neural image caption generation with visual attention». En: *International conference on machine learning*. PMLR, págs. 2048-2057.

Yang, Guorun et al. (2019). «Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios». En: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, págs. 899-908.

Yang, Yi et al. (2020). «Multi-camera visual SLAM for off-road navigation». En: *Robotics and Autonomous Systems* 128, pág. 103505.

Yi, Xin, Ekta Walia y Paul Babyn (2019). «Generative adversarial network in medical imaging: A review». En: *Medical image analysis* 58, pág. 101552.

Yoo, Jae-Chern y Tae Hee Han (2009). «Fast normalized cross-correlation». En: *Circuits, systems and signal processing* 28, págs. 819-843.

Zeiler, Matthew D y Rob Fergus (2014). «Visualizing and understanding convolutional networks». En: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*. Springer, págs. 818-833.

Zhang, Zhengyou (2000). «A flexible new technique for camera calibration». En: *IEEE Transactions on pattern analysis and machine intelligence* 22.11, págs. 1330-1334.

Zheng, Yuan y Silong Peng (2013). «A practical roadside camera calibration method based on least squares optimization». En: *IEEE Transactions on Intelligent Transportation Systems* 15.2, págs. 831-843.

Zhong, Wanzhen y Xiaona Dong (2015). «Camera calibration method of binocular stereo vision based on OpenCV». En: *AOPC 2015: Image Processing and Analysis*. Vol. 9675. SPIE, págs. 571-576.

Zhou, Changxin et al. (2021). «Vehicle detection and disparity estimation using blended stereo images». En: *IEEE Transactions on Intelligent Vehicles* 6.4, págs. 690-698.

Zhou, Xingyi, Dequan Wang y Philipp Krähenbühl (2019). «Objects as points». En: *arXiv preprint arXiv:1904.07850*.

Zhou, Xingyi, Jiacheng Zhuo y Philipp Krahenbuhl (2019). «Bottom-up object detection by grouping extreme and center points». En: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, págs. 850-859.

Zhu, Xizhou et al. (2020). «Deformable detr: Deformable transformers for end-to-end object detection». En: *arXiv preprint arXiv:2010.04159*.

Zoetgnandé, Yannick Wend Kuni et al. (2019). «Robust low resolution thermal stereo camera calibration». En: *Eleventh International Conference on Machine Vision (ICMV 2018)*. Vol. 11041. SPIE, págs. 353-360.

Zou, Zhengxia et al. (2023). «Object detection in 20 years: A survey». En: *Proceedings of the IEEE*.