Arroyo, Sebastián Ismael

Sistema de georreferenciamiento vehicular por visión monocular fisheye y calibración bayesiana

Sebastián Ismael Arroyo, Repositorio Institucional Digital de Acceso Abierto,
Abril de 2022, pp. 137,
http://ridaa.unq.edu.ar,
Universidad Nacional de Quilmes, Secretaría de Posgrado,
Doctorado en Ciencia y Tecnología

Sistema de georreferenciamiento vehicular por visión monocular fisheye y calibración bayesiana

TESIS DOCTORAL

Sebastián Ismael Arroyo

seba.arroyo@protonmail.com

Resumen

Los sistemas de transporte juegan un rol clave en el desarrollo económico y cultural, distribuyendo bienes y personas a través de vías progresivamente más complejas, interconectadas y multimodales. Los avances en tecnologías de información y comunicación llevaron al surgimiento de los Sistemas Inteligentes de Transporte. Una de las fuentes de datos de los Sistemas de Transporte son las cámaras de monitoreo, que permiten medir datos relevantes del estado del tránsito aplicando algoritmos de visión artificial. Hay dos tipos principales de aplicaciones para estos datos: mejorar la seguridad y mejorar la movilidad.

En particular, algunas aplicaciones requieren conocer las posiciones de los vehículos con un alto grado de precisión. Estas posiciones pueden calcularse a partir de imágenes tomadas por las cámaras de monitoreo usando técnicas de fotogrametría. En particular, la retro-proyección es una función que toma las coordenadas imagen de un objeto detectado y calcula sus coordenadas en el marco referencia del mundo. Para ello, es necesario calibrar la cámara y luego, suponiendo que las posiciones de los vehículos están en el plano del suelo, calcular la retro-proyección desde la imagen a coordenadas georreferenciadas. Esta hipótesis de que los objetos de interés están en el plano del suelo compensa la falta de información tridimensional y hace posible la reconstrucción de las posiciones físicas de los objetos a partir de la imagen 2D.

Las cámaras utilizadas tradicionalmente en video vigilancia son cámaras fijas que poseen poca distorsión óptica. Tienen la desventaja de poder observar una región de 60◦ × 60◦. Mientras que las cámaras omnidireccionales (como las cámaras ojo de pez o *fisheye*) tienen un campo de visión de aproximadamente 360◦ × 180◦, es decir que observan el hemisferio visual frente a ellas por completo. Pueden ver `de horizonte a horizonte', son ideales para

monitorear zonas muy amplias sin perder de vista ninguna región, permitiendo observar trayectorias completas de vehículos.

En esta tesis se resuelve cómo retro-proyectar coordenadas desde la imagen de una cámara fisheye a coordenadas georreferenciadas, considerando las distorsiones ópticas de la cámara descriptas por el modelo estereográfico. Se exploran algunas aplicaciones: retro-proyectar la posición de un objeto de interés en una cámara fisheye para apuntar hacia el mismo una cámara Pan-Tilt-Zoom y captar una imagen con mayor detalle del mismo; y retro-proyectar las posiciones de vehículos detecta-dos en una cámara fisheye para estimar la probabilidad de que los vehículos estén superando la velocidad máxima permitida.

No hay un método de calibración y retro-proyección que calcule las incertezas de todos los parámetros ni de todas las variables involucradas. El tratamiento de incertezas es fundamental para la toma de decisión de si hubo o no infracción. Entonces se formuló *ab initio* con un enfoque bayesiano un esquema de calibración y retro-proyección basado en la propagación lineal de incerteza. Una idea clave de la formulación es que la incerteza de los parámetros y de la posición en la imagen deben propagarse a través de la retro-proyección hacia las coordenadas georreferenciadas. Se hace la propagación linealizando la función de retro-proyección y restringiendo la cuantificación de incerteza sólo a matrices de covarianza. El esquema de calibración tiene dos etapas. La primera asociada a los parámetros de distorsión óptica, y la segunda a los parámetros de posición y orientación de la cámara. El resultado de la calibración es la media y la varianza de la distribución de probabilidad *a posteriori* de los parámetros. La retro-proyección toma estas distribuciones de probabilidad de los parámetros calibrados y estima la distribución de probabilidad de las coordenadas retro-proyectadas.

Se evaluó el método con datos reales y datos simulados. Aplicando el método en datos simulados se comprobó que la aproximación lineal necesaria para hacer los cálculos es válida. Se comprobó también que la calibración y retro-proyección estiman correctamente los parámetros de calibración y las coordenadas georreferenciadas con su incerteza. Aplicando el método en datos reales se comprobó el comportamiento esperado de la retro-proyección: hay una magnificación de la incerteza si el punto retro-proyectado está lejos de la cámara y tiene un factor de vista pequeño, y también debido a la distorsión óptica.

El método bayesiano de calibración en dos etapas y de retro-proyección se resolvió para el modelo de distorsión estereográfico pero puede extenderse fácilmente a otros modelos de distorsión óptica. Con éste método una cámara de monitoreo se convierte en un sensor que provee la posición georreferenciada de vehículos con una estimación correcta de la incerteza.

Universidad Nacional de Quilmes

Departamento de Ciencia y Tecnología

Sistema de georreferenciamiento vehicular por visión monocular fisheye y calibración bayesiana

por Sebastián I. Arroyo

Tesis presentada ante la Secretaría de Posgrado de la Universidad Nacional de Quilmes como requisito para la obtención del grado académico de

Doctor en Ciencia y Tecnología

por la Universidad Nacional de Quilmes

Director

Dr. Damián Oliva

Codirector

Mgtr. Félix Safar

Tutor

Dr. Manuel Eguía

Lugar de trabajo: Departamento de Ciencia y Tecnología, Universidad Nacional de Quilmes.

Bernal, Buenos Aires, 28 de abril de 2022.

Agradecimientos

A mi familia, Sabina, Joaquín, Verónica y Alejandro.

A los amigos y amigas, a los que tuve cerca y los que tuve lejos. A Bárbara por todo su apoyo y empuje.

A mis directores. A mis compañeros y amigos de la UNQ. A Félix que no llegó a ver finalizada esta tesis.

Índice general

Capítulo 1

Motivación y objetivos

Los sistemas de transporte juegan un rol estratégico en la economía mundial distribuyendo personas y bienes a través de vías progresivamente más complejas, interconectadas y multimodales [1].

> Los **Sistemas Inteligentes de Transporte** (ITS) unen tecnologías variadas con el objetivo de proveer mejoras en la movilidad y seguridad. Utilizan tecnologías de datos, comunicación y computación para proveer servicios y aplicaciones resolviendo un ancho abanico de problemas de transporte. [2, 1]

Los avances en la comunicación y computación trajeron avances en las tecnologías de recolección de datos de los ITS. Los municipios instalan una cantidad creciente de cámaras con la finalidad de detectar infracciones de tránsito y asistir a la seguridad ciudadana. Es así que las cámaras de monitoreo urbano son una de las muchas fuentes de datos de ITS [1]. En particular las infracciones de velocidad requieren resolver el mapeo geométrico entre posiciones de píxel en imagen a coordenadas del mundo real con un alto grado de precisión [3].

> La **Fotogrametría** puede definirse como "la ciencia de medir fotografías". Para determinar características geométricas (distancias, áreas, etc.) de un objeto o terreno se deben conseguir coordenadas del mismo en la fotografía a partir de las cuales calcular la geometría o crear mapas. [4]

Esta tesis doctoral se planteó con el objetivo de mejorar dos aspectos de la visión artificial: usar cámaras tipo ojo de pez para abarcar una región mayor a las cámaras tradicionales y desarrollar técnicas para la estimación de posición de vehículos con incerteza.

En general no es posible reconstruir información 3D a partir de una imagen 2D, salvo que se tengan múltiples cámaras (visión binocular), se complemente con algún otro tipo de sensor (lidar, radar, etc) o se compense la falta de información de distancia con un modelo del ambiente. Pero en la mayoría de sistemas de monitoreo urbano no se instalan sensores complementarios a la cámara por el costo adicional que implica. También es normal que se instalen varias cámaras para vigilar una escena pero en general se las dispone tal que los campos visuales tengan poca superposición para maximizar la zona vigilada. Entonces no es esperable poder ver un objeto en más de una cámara a la vez. Por estas condiciones habituales de operación de los ITS, esta tesis es aplicable a **sistemas de visión monoculares**, una sola cámara y sin sensores adicionales. A su vez, las escenas urbanas traen una ventaja: los eventos de interés asociados al comportamiento de vehículos o peatones se producen *sobre la superficie terrestre*. Esta hipótesis de que los objetos de interés están en el plano del suelo compensa la falta de información tridimensional y hace posible la reconstrucción de las posiciones físicas de los objetos a partir de la imagen 2D.

Actualmente las cámaras más usadas para video vigilancia se describen por el modelo de proyección central (cámaras 'planas'), es decir que las líneas rectas en el mundo físico se proyectan en la imagen también como líneas rectas. Esto es deseable para que la imagen obtenida se vea natural y sin deformaciones. Pero estas cámaras tienen típicamente un campo de visión de $60° \times 60°$ y si se quiere monitorear una región amplia es necesario usar múltiples cámaras, lo que aumenta el costo de infraestructura, instalación, sincronización, mantenimiento y operación. Alternativamente se puede usar una sola cámara pero con la capacidad de cambiar su orientación (movimientos de pan-tilt) pero requieren la intervención de un operario que comande el movimiento y no se puede observar la región completa en simultáneo. Éstas limitaciones se pueden subsanar usando cámaras omnidireccionales.

> Las **cámaras omnidireccionales** tienen un campo de visión de 360° × 180° es decir que observan completo el hemisferio visual frente a ellas, sin necesidad de movimiento mecánico ni de múltiples cámaras. La desventaja es que la imagen que capturan está severamente curvada. Las cámaras ojo de pez o *fisheye* son un tipo de cámara omnidireccional, en lugar de espejos o una composición de múltiples cámaras en un arreglo radial usan lentes. [5]

La primera mitad de la tesis tiene como objetivo *hacer fotogrametría con una cámara fisheye*. Para ello primero se trata la teoría de los modelos de formación de imagen y distorsión óptica y se muestran pruebas experimentales relacionadas. Luego, para hacer fotogrametría con una cámara *fisheye* se resolvió la *retro-proyección* (revirtiendo la distorsión óptica) desde coordenadas en la imagen hacia el mundo físico usando la hipótesis de que los objetos están sobre el suelo.

Para los ITS es fundamental llegar a una toma de decisión, es decir responder una pregunta concreta sobre el vehículo. Por ejemplo, ¿está superando el límite de velocidad? o ¿está invadiendo una zona prohibida? Pero no alcanza con un cálculo puntual de posición o velocidad. Es necesario cuantificar un intervalo de confianza o incerteza de esa posición o velocidad retro-proyectados. La segunda mitad de la tesis tiene como objetivo *cuantificar la incerteza de la retro-proyección*.

> La **inferencia bayesiana** esta basada en principios probabilísticos y dado un modelo probabilístico prescribe una única solución posible. En términos prácticos, el enfoque bayesiano se ha hecho más atractivo con los avances en métodos computacionales que permiten implementar con poco esfuerzo modelos complicados. [6] En los últimos cincuenta años se han dado duros debates entre dos escuelas estadísticas: frecuentistas y bayesianos. Y aunque no se puede declarar "ganadores" claramente el enfoque bayesiano es el mejor adaptado a las necesidades de las ciencias actuales. [7]

Teniendo resuelta la fotogrametría, se tomó un enfoque bayesiano y se formuló desde primeros principios un modelo probabilístico para cuantificar la incerteza. Para

finalizar, se hace una predicción de la posición física de un vehículo acompañada de una cuantificación de la incerteza en la posición.

1.1. Contenidos y contribuciones de esta tesis

En el capítulo 2 se presentan las bases de modelos de formación de imagen, distorsión óptica, métodos de calibración e inferencia bayesiana. El método de calibración es fundamental porque permite estimar los parámetros de distorsión óptica y de posición de la cámara, sin ellos es imposible hacer fotogrametría. A continuación en el capítulo 3 se resuelve la retro-proyección, es decir se usan las coordenadas de un objeto detectado en imagen para estimar su posición en el plano horizontal del suelo. Además se resuelve un caso de control motor que consiste en usar la detección de un objeto en la cámara ojo de pez para apuntar una cámara móvil hacia el mismo. Se muestra un primer acercamiento al problema de la cuantificación de incertezas para el cálculo de infracción de velocidad. Este punto disparó la realización de la formulación bayesiana que se presenta en el capítulo 5. La validación de la teoría se presenta en el capítulo 6 donde se evalúan datos reales y simulados. El último capítulo, 7, presenta la discusión y las conclusiones.

Las contribuciones principales del trabajo son dos: 1) Resolver y poner a prueba un algoritmo de retro-proyección desde la imagen a coordenadas georreferenciadas para cámaras ojo de pez; y 2) formular el proceso de calibración de cámara en términos bayesianos.

Otras contribuciones son: resolver la retro-proyección para otros modelos de distorsión óptica permitiendo generalizar el resultado a otras cámaras (dióptricas o catadióptricas); apuntar una cámara móvil hacia un objeto detectado en una cámara ojo de pez; comprobar que una propagación lineal de incertezas es válida, a pesar de las fuertes no linealidades de la retro-proyección; y mostrar la predicción bayesiana de la retro-proyección con su incerteza.

Los aportes de esta tesis se publicaron en los siguientes trabajos:

[8] Damián Oliva, Agustín Yabo, Lilián García, Sebastián I Arroyo, and Félix G Safar. Implementación de un sistema para la medición del flujo de tránsito

y detección de embotellamientos en autopistas. In *Argentine Symposium on Artificial Intelligence (ASAI 2015)-JAIIO 44 (Rosario, 2015)*, 2015

[9] Sebastian I. Arroyo, Felix Safar, and Damian Oliva. Georeferenced feature tracking in wide field images. In *2015 16th Workshop on Information Processing and Control, RPIC 2015*, pages 1–6, Cordoba, 2016. IEEE. ISBN 9781467384667. doi: 10.1109/RPIC.2015.7497125

[10] Damián Ezequiel Stanganelli. Implementación de un sistema de seguimiento de objetos múltiples. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Director: Dr. Damián Oliva, Co-Director: Mg. Sebastián I. Arroyo, Universidad Nacional de Quilmes, 2016

[11] Agustín Yabo, Sebastián I Arroyo, Félix G Safar, and Damián Oliva. Vehicle classification and speed estimation using computer vision techniques. In *XXV Congreso Argentino de Control Automático (AADECA 2016)(Buenos Aires, 2016)*, 2016

[12] Sebastián I. Arroyo, Félix Safar, and Damián Oliva. Probabilidad de infracción de velocidad de vehículos utilizando visión artificial en cámaras de campo amplio. In *ARGENCON 2016, 3rd Biennial Congress of IEEE Argentina*, pages 1–6, Buenos Aires, 2016

[13] Lilián García Rojas. Sistema de videovigilancia integrado por una cámara fisheye y una cámara pan-tilt-zoom. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Director: Mg. Sebastián I. Arroyo, Co-Director: Dr. Damián Oliva, Universidad Nacional de Quilmes, October 2018

[14] Sebastián I. Arroyo, Ulises Bussi, Félix Safar, and Damián Oliva. A monocular wide-field vision system for geolocation with uncertainties in urban scenes. *Engineering Research Express*, 2(2):025041, 06 2020. doi: 10.1088/2631-8695/ab9b36

[15] Juan Manuel Castellano. Incerteza de localización de vehículos con redes neuronales en videos de tránsito. Trabajo Final de la Carrera Ingeniería en

Automatización y Control Industrial, Director: Mg. Sebastián I. Arroyo, Co-Director: Dr. Damián Oliva, Universidad Nacional de Quilmes, March 2019

[16] Sebastián Arroyo, Lilian Garcia, Felix Safar, and Damian Oliva. Sistema de video-detección basado en cámaras fisheye y ptz. *IEEE Latin America Transactions*, 100(1e), February 2021. (Early Access)

1.2. Fuentes de financiamiento

[17] Consejo Nacional de Investigaciones Científicas y Técnicas. Becas internas de postgrado tipo I, September 2013

[18] Consejo Nacional de Investigaciones Científicas y Técnicas. Beca interna doctoral para temas estratégicos. Resolución D Nº 3423, September 2014

[19] Secretaría de Políticas Universitarias. Sistemas de Video Detección Vehicular con Visión de Campo Amplio, y su aplicación a los Sistemas Inteligentes de Transporte (SIT). 1ra. Convocatoria de Proyectos de Investigación Básica y Aplicada "Universidad y Transporte", 2014. Director: Mg. Félix Safar, Co-Director: Dr. Damián Oliva

[20] Secretaria de Investigación Universidad Nacional de Quilmes. Programa: Estrategias de ingeniería en automatización, computación y procesos industriales apicados a la resolución de problemas tecnológicos; proyecto: Desarrollo de sistemas autónomos basados en visión. Archivo Público de Actos Resolutivos RR-1076-15 (#11084), May 2015. URL `https://apar.unq.edu.ar/archivo/detalle.php?idArchivo=11084`. Director de Programa: Safar, Félix Gustavo. Director de Proyecto: Oliva, Damián

[21] Secretaria de Investigación Universidad Nacional de Quilmes. Programa: Estrategias de ingeniería en automatización, computación y procesos industriales apicados a la resolución de problemas tecnológicos; proyecto: Desarrollo de sistemas autónomos basados en visión. EXPTE 1406/15 (02/05/2017 - 30/04/2019), 2017. URL `http://secretariadeinvestigacion.web.unq.edu.ar/programas-proyectos/`. Director de Programa: Safar, Félix Gustavo. Director de Proyecto: Oliva, Damián

Capítulo 2

Introducción

Visión artificial en el contexto de Sistemas de Transporte, modelos de formación de imagen, calibración de sus parámetros e inferencia bayesiana.

En este capítulo se explica el contexto del problema de monitoreo de tránsito y se presentan los conceptos elementales de fotogrametría, calibración de cámara e inferencia bayesiana en que se basa el resto del trabajo.

Primero se mencionan las ventajas de aplicar visión artificial para resolver problemas de tránsito, se explica por qué se usa una cámara ojo de pez a pesar de su severa distorsión óptica y por qué el problema puede resolverse con una sola cámara.

Luego se presenta el modelo *pinhole* (estenopéico), que es el modelo de formación de imagen para el caso mas sencillo de una cámara fija sin distorsión óptica (luego se agrega la distorsión óptica). Este modelo relaciona la posición de un objeto en el mundo físico 3D con la posición que tiene la proyección de dicho objeto en la imagen (que se mide en píxeles para una cámara digital).

Un tipo muy popular de cámara de vigilancia es la *Pan-Tilt-Zoom* que cuenta con dos motores que rotan la cámara (movimientos de pan y tilt), así el operario puede apuntarla en la dirección que desee. Por ello se extiende el modelo de formación de imagen agregando la posibilidad de orientar la cámara.

Si se quieren obtener medidas de distancias o posiciones de objetos en el espacio físico se necesita estimar los parámetros de proyección. La sección de calibración describe los métodos estándar para estimar los parámetros intrínsecos y los extrínsecos.

Finalmente, se presenta la teoría básica de inferencia bayesiana en contexto del problema de calibración y predicción, que se retoma en el capítulo 5 para desarrollar un esquema de calibración y predicción adecuado.

2.1. Visión artificial en el contexto de sistemas inteligentes de transporte

Actualmente e históricamente los sistemas de transporte han jugado un rol clave en el desarrollo económico y cultural humanos al conectar ciudades, regiones y naciones, y al distribuir bienes y personas a través de vías progresivamente más complejas, interconectadas y multimodales [1]. Los avances en tecnologías de información y comunicación permiten la implementación de soluciones de transporte que llevaron al surgimiento de los Sistemas Inteligentes de Transporte (ITS). Los ITS ofrecen servicios innovadores, como información de asistencia a conductores, mejoras en el flujo de tránsito en una ciudad y mejoras en la seguridad [2]. Los ITS son parte de *Internet de las Cosas* y de las ciudades inteligentes, con tecnologías de sensores, de control y de cómputo en la nube.

Las aplicaciones de ITS tienen dos categorías principales: la movilidad y la seguridad. Las aplicaciones en movilidad proveen servicios para hacer el tránsito más eficiente, como cálculos de ruta entre origen y destino optimizando distancia, tiempo o consumo de combustible, también considerando el estado del tránsito en base a la información colectada por las fuentes de datos de los ITS. Las aplicaciones en seguridad incluyen, por ejemplo, alertas al conductor de una colisión inminente, informes del estado de la ruta, si hay mal clima o si hubo un accidente. Hay cuatro grupos de fuentes de datos: datos provistos por viajeros (redes sociales o aplicaciones como *Waze* [22]), datos de áreas extensas (imágenes aéreas y satelitales), datos de base vehicular (sensores montados en los vehículos como GPS, dispositivos Bluetooth o

WiFi) y datos tomados en la vía.

Los datos en vía son tomados por sensores ubicados en posiciones fijas a lo largo de las calles, rutas y autopistas. Algunas de estos sensores se usan desde hace décadas y pueden ser pasivos, es decir que no interrumpen el flujo de tránsito. Los más usados son el de lazo inductivo para conteo de vehículos y medición de velocidad [23] y el radar de microondas que detecta flujo, presencia y velocidad de vehículos. Los sensores infrarrojos pueden usarse para inferir el tipo de vehículo y los sensores ultrasónicos para detectar presencia de vehículos y ocupación de carril. Finalmente, otra tecnología de colección de datos en vía son las cámaras de monitoreo de tránsito que permiten medir datos relevantes del tránsito aplicando algoritmos de visión artificial.

La visión artificial se puede describir como la tarea de interpretar y dar sentido al mundo a partir de imágenes y videos [24, 25, 26]. Es una rama de la inteligencia artificial cuyo objetivo es emular la inteligencia humana. Se puede pensar a los procesos de la visión artificial en tres niveles, el mas bajo esta asociado a reducir el ruido, aumentar el contraste, realzar bordes. En el nivel medio hay procesos que segmentan la imagen en regiones asociadas a objetos, describen cuantitativamente algunas características de las regiones en base a su forma y color. El tercer nivel de procesos, el mas alto, se superpone explícitamente con la inteligencia artificial ya que intenta "dar sentido" a los objetos en la imagen y emular las funciones cognitivas asociadas a la percepción visual humana. La visión artificial es difícil en parte porque se trata de resolver un problema inverso: recuperar información del mundo físico (tridimensional) a partir de información insuficiente (bidimensional). Aún así tiene una variedad muy amplia de aplicaciones, entre ellas: navegación de robots móviles, control de calidad industrial, análisis de imágenes médicas, seguridad, vigilancia, reconstrucción 3D.

En el contexto de los Sistemas Inteligentes de Transporte, la visión artificial permite la interpretación del entorno con el propósito de mejorar la seguridad, la aplicación de la ley o la eficiencia del tránsito (reducir errores, costos, esfuerzo y tiempo) [27, 28]. Algunos problemas específicos donde se aplica la visión artificial son [1, 29, 30]:

- cruces de semáforos en rojo,
- avances en contramano,
- giros no permitidos,
- reconocimiento de patentes,
- cuantificación del flujo de tránsito,
- detección de situaciones anómalas.

Actualmente, existe una infraestructura de cámaras ya instaladas en la vía pública que puede ser utilizada para este tipo de análisis. En comparación con las otras fuentes de datos en vía de los ITS, la visión artificial aplicada a cámaras de monitoreo permite una cobertura más amplia que radares y lazos inductivos, porque no se ve afectada por la densidad de tránsito y porque la recolección de datos es continua. Una desventaja es que el mal clima puede obstaculizar la visión de las cámaras.

En particular, el servicio ofrecido por *Traffic Vision* [29] utiliza infraestructura ya existente de decenas de cámaras de monitoreo donde normalmente la vigilancia sería manual. Este servicio provee una visualización inmediata de los vehículos detenidos como resultado de una colisión, detecta vehículos en contramano o detenidos, detecta cambios inesperados en el volumen de tránsito, genera automáticamente alertas de incidentes en autopistas para alivianar el trabajo de operarios humanos y reducir el tiempo de respuesta, provee funcionalidades analíticas específicas de transporte (adaptadas a cambios de luz y movimientos de cámara) y colecta en tiempo real y en continuado estadísticas de tránsito, clasificando los vehículos según tamaño.

Otros problemas que se busca resolver usando visión artificial [31, 1] son:

- conteo de vehículos en estacionamientos,
- detección de colisiones,
- sobrepasos,
- no respetar distancia de frenado,
- bloqueos de box o de zona de intersección,
- conducción peligrosa,
- infracciones de velocidad (requiere alta precisión de detección).

Las cámaras utilizadas tradicionalmente en video vigilancia son las cámaras fijas (figura 2.1a) o *Pan-Tilt-Zoom* (PTZ, figura 2.1b y 2.2b). Las imágenes capturadas por los dos tipos de cámaras son similares, pero las PTZ pueden cambiar su dirección de apunte y nivel de zoom.

(a) Dos cámaras fijas.

(b) Dos cámaras PTZ.

Figura 2.1: Ejemplos de cámaras instaladas en la vía publica.

Las cámaras PTZ tienen como ventajas (figura 2.3a): no presentar distorsiones y tener capacidad de cambiar su campo de visión (zoom). Tiene como desventaja un campo de visión máximo de $60° \times 60°$ que solo permite ver una parte reducida de la escena. Por lo tanto, si se quiere monitorear una región amplia es necesario usar múltiples cámaras. También requieren de un operador para controlar la dirección de observación y el nivel de zoom.

(a) Cámara *fisheye* VIVOTEK modelo FE8172.

(b) Cámara PTZ Hecker modelo HE-Z9116.

Figura 2.2: Las dos cámaras usadas en esta tesis.

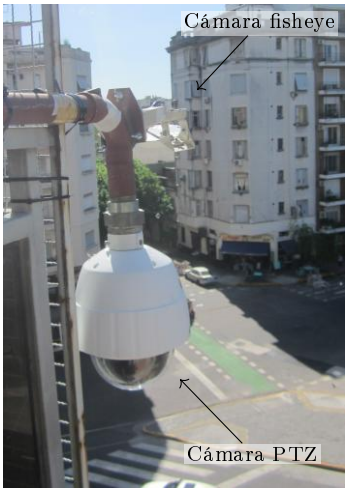
Las cámaras omnidireccionales (por ejemplo las cámaras ojo de pez o *fisheye*, figura 2.2a) tienen un campo de visión de aproximadamente $360° \times 180°$, es decir que observan por completo el hemisferio visual frente a ellas, minimizando los puntos ciegos (figura 2.3b).

(a) Captura de PTZ.

+ No hay distorsión óptica.
+ Zoom hasta ×30.
- Zona de visión limitada a 60° × 60°.
- Reorientaciones necesitan intervención humana o pre-programación.

(b) Captura de *fisheye*.

+ Campo visual 360° × 183°.
+ Observación simultánea de toda la escena.
+ No hay desgaste mecánico.
- Severa distorsión óptica.



(c) PTZ y *fisheye*.

Figura 2.3: (a)-(b) Ejemplos de una captura y ventajas y desventajas de las cámaras PTZ y *fisheye* respectivamente. (c) Las dos cámaras montadas en un mismo soporte desde donde se tomaron las capturas.

En las figuras 2.2a y 2.3b se muestra una cámara *fisheye* y una PTZ instaladas en una misma locación y se puede apreciar la diferencia en el tamaño de los campos visuales de ambas cámaras. Este experimento se realizó en una intersección compleja (Parque Lezama, CABA, Argentina). La desventaja principal de las cámaras omnidireccionales es la marcada deformación que sufre la imagen que dificulta los cálculos de fotogrametría.

En particular el servicio ofrecido por *Gridsmart* [30] provee una solución para la recolección de datos de tránsito, permitiendo la actuación en intersecciones de calles y proveyendo conocimiento situacional usando una única cámara *fisheye*. Al estar instalada especialmente sobre la intersección, provee visión completa de la zona, en especial el centro donde se cruzan vehículos, peatones y ciclistas, lo cual resulta ideal para la gestión de incidentes. Gridsmart explota las ventajas de la cámara *fisheye* (figura 2.3a):

- adquiere simultáneamente toda la escena en una única captura;
- no es necesario que un operario la apunte, cambie el zoom;
- y requiere poco mantenimiento ya que no tiene partes móviles que se pueden

deteriorar por rozamiento.

Gridsmart utiliza algoritmos de visión artificial que reconstruyen un modelo 3D de vehículos, peatones y otros objetos que se acercan a la intersección. Realiza el seguimiento de vehículos a través de zonas predefinidas por el usuario, retornando estadísticas de [30]: paso de vehículos, maniobras de giro, paso de autos en verde, duración del ciclo de semáforo y una clasificación de los vehículos basada en la longitud.

Por los motivos anteriores la *fisheye* es particularmente superior a la PTZ para el seguimiento simultáneo de múltiples vehículos en intersecciones complejas. También se evitan los problemas de retardos que surgen cuando se utilizan distintas cámaras cuyas capturas deben ser sincronizadas. Se reduce la cantidad de puntos ciegos simplificándose los procesos de calibración y procesamiento. Y al ser una sola cámara necesita menos infraestructura y la instalación es mas rápida y económica.

2.2. Proyección simple: modelo pinhole

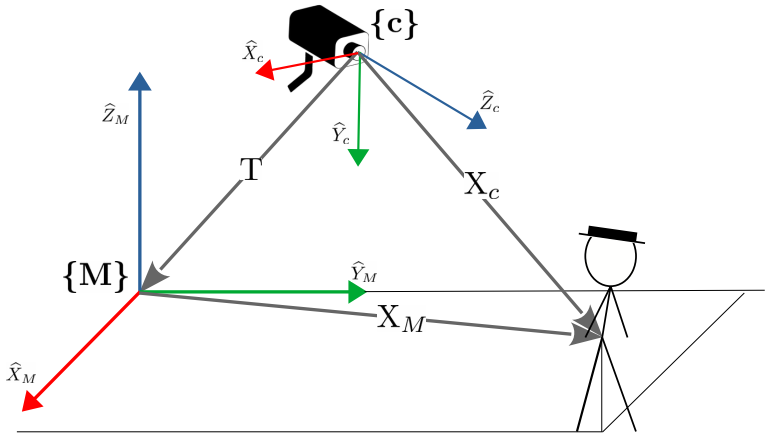


Figura 2.4: Cambiar la base {M} de las coordenadas X_{M} donde se ubica el objeto a la base {C} de la cámara.

El modelo *pinhole* es la descripción más sencilla de cómo se forma la imagen [32], sus componentes elementales son: una roto traslación que calcula las coordenadas del objeto en el marco de referencia de la cámara, la pérdida de información de distancia a la cámara y la conversión a escala (resolución) del sensor donde se captura la imagen.

Como ilustra en la figura 2.4 las coordenadas de un objeto $X_{\mathrm{M}} = [x_{\mathrm{M}}, y_{\mathrm{M}}, z_{\mathrm{M}}]^{\top}$ son roto trasladadas para obtener las coordenadas X_{C} en el marco de referencia 'cámara' según

$$X_{\mathrm{C}} \equiv \begin{bmatrix} x_{\mathrm{C}} \\ y_{\mathrm{C}} \\ z_{\mathrm{C}} \end{bmatrix} = \mathbf{R}(r_1, r_2, r_3) \begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ z_{\mathrm{M}} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \mathbf{R}\ X_{\mathrm{M}} + \mathrm{T}. \tag{2.1}$$

La matriz de rotación $\mathbf{R}$ se construye a partir de tres valores r_1, r_2, r_3 que según la convención elegida pueden ser tres ángulos de rotación *yaw*, *pitch* y *roll* (giros alrededor de los ejes fijos X-Y-Z), los ángulos de Euler (giros alrededor de los ejes móviles X-Y-Z) o las tres componentes del vector de Rodrigues [33]. La matriz de rotación cambia la base tal que X_{M} se exprese en coordenadas del marco de referencia de la cámara. El vector de traslación T es la posición del origen de coordenadas del mundo en el marco de referencia cámara. Se denota el vector de parámetros extrínsecos de seis componentes como

$$\Theta = \{r_1, r_2, r_3, t_x, t_y, t_z\}.$$

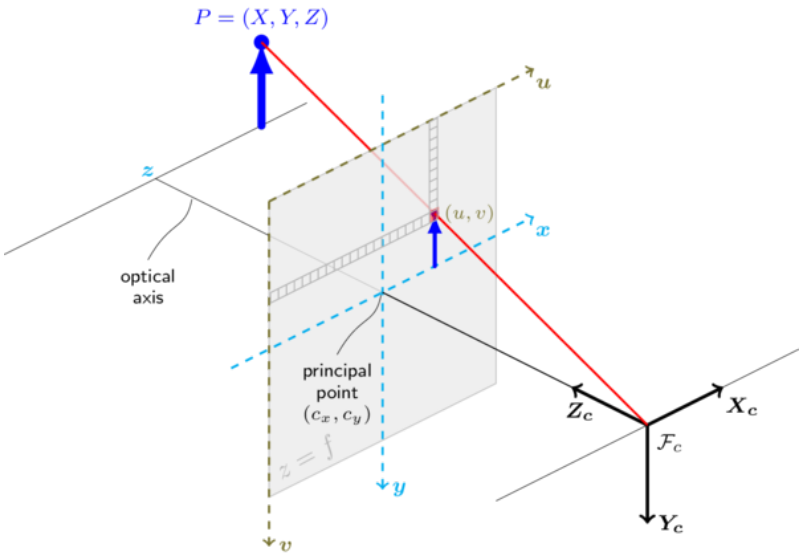


Figura 2.5: Esquema de proyección *pinhole* de Bradski [34].

Tomando prestado de la física el concepto de observador: el cambio de coordenadas en las que se vé un objeto estático dependen del cambio de pose de observación. En este caso pasar de observar desde el marco de referencia del mundo las coordenadas X_{M} a observar desde el marco de referencia de la cámara las coordenadas X_{C} requiere saber cómo se movió el observador. Esa información está condensada

en los seis elementos de Θ o equivalentemente en el par $(\mathbf{R}, \mathrm{T})$ que representan la posición y orientación de la cámara respecto al mundo. Se dice que ésta es la *pose* de la cámara.

Continuando con el modelo *pinhole*, se interseca la recta definida por el vector X_C con el plano imagen, ubicado a distancia unitaria de la cámara en $z_\mathrm{C} = 1$, paralelo al plano x_C-y_C como se ilustra en la figura 2.5. Esto se denomina modelo colineal [35] donde el centro de proyección de la cámara, el punto en la imagen y el objeto se hallan sobre la misma línea recta. En símbolos, al dividir por z_C,

$$X_\mathrm{h} \equiv \begin{bmatrix} x_\mathrm{h} \\ y_\mathrm{h} \end{bmatrix} = \begin{bmatrix} x_\mathrm{C} \\ y_\mathrm{C} \end{bmatrix} / z_\mathrm{C}, \tag{2.2}$$

se obtiene una representación en coordenadas homogéneas de la *dirección de llegada* del haz de luz.

Finalmente la posición en píxeles del sensor digital que se activa con ese haz de luz se calcula

$$X_\mathrm{I} \equiv \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x x_\mathrm{h} + c_x \\ f_y y_\mathrm{h} + c_y \end{bmatrix} \tag{2.3}$$

introduciendo los parámetros f_x, f_y (comúnmente llamados *distancia focal*) que convierten la escala a píxeles y c_x, c_y (el *centro óptico*) que desplazan el origen de coordenadas a la esquina superior izquierda de la imagen. Se denota el vector de parámetros intrínsecos para el modelo *pinhole* como

$$\Gamma = \{f_x, f_y, c_x, c_y\}.$$

El modelo *pinhole* puede escribirse como el mapeo $[x_\mathrm{M}, y_\mathrm{M}, z_\mathrm{M}]^\top \to [u, v]^\top$ que en su forma compacta es

$$z_\mathrm{C} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{c|c} \mathbf{R} & \mathrm{T} \end{array}\right] \begin{bmatrix} x_\mathrm{M} \\ y_\mathrm{M} \\ z_\mathrm{M} \\ 1 \end{bmatrix}. \tag{2.4}$$

El lado derecho de la ecuación es una operación lineal. La distancia entre el objeto y la cámara, z_C, no es parte de las variables de salida del mapeo (recordar la ecuación (2.2)). El vector resultante en el lado izquierdo $[uz_\mathrm{C}, vz_\mathrm{C}, z_\mathrm{C}]$ tiene z_C en la tercera componente. Se normaliza el vector por su tercera componente y se obtienen los valores de salida del mapeo $[u, v, 1]$. Se interpreta z_C como una variable de cálculo

intermedio, es información de distancia tridimensional que se perdió en el resultado final. La normalización por z_C hace que el mapeo sea no lineal.

Es una transformación que conserva la colinealidad porque las líneas rectas en el dominio siguen siendo rectas en el codominio. Muchas cámaras digitales conservan la colinealidad para que la perspectiva de la escena se perciba 'natural' en la imagen, por lo tanto el modelo *pinhole* es aceptable para describir muchas de las cámaras convencionales.

En resumen, un *modelo de formación de imagen* describe el camino que recorre un haz de luz desde el objeto en marco de referencia 'mundo' hasta llegar al sensor CCD o CMOS. Se lo puede descomponer en bloques como muestra la figura 2.6. Una primera parte de roto traslación lleva al objeto desde el marco de referencia del mundo al de la cámara (ver también figura 2.4), y una segunda parte de proyección al chip CCD a través de la óptica de la cámara. Cada parte tiene asociados parámetros. Primero, los parámetros *extrínsecos* que representan la posición y orientación de la cámara en el espacio, son tres para la posición y tres para la orientación. Segundo, los *intrínsecos* que cuantifican la distorsión óptica de la cámara, la resolución del sensor, etc. Dependen sólo de la cámara y no importa donde se la instale, son independientes de la pose.

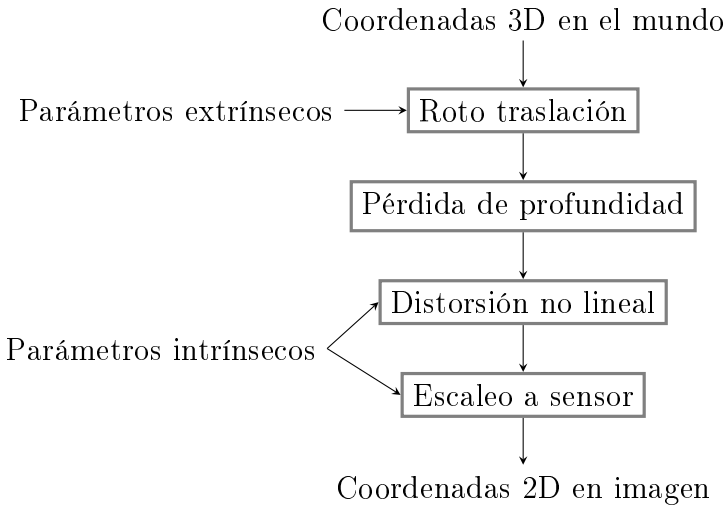


Figura 2.6: Diagrama de flujo del modelo de formación de imagen.

2.3. Distorsión radial

Si se pretende describir con precisión la formación de imagen el modelo *pinhole* no alcanza, ya que es imposible que la luz llegue al sensor digital manteniendo una trayectoria perfectamente recta respecto a su dirección de llegada al aparato (dada por X_{h}). Esta deformación puede ser accidental (por ejemplo error constructivo de las lentes) o intencional (por ejemplo para aumentar el campo visual de la cámara). En todo caso hay que expandir el modelo introduciendo una función no lineal para modelar la distorsión óptica.

Figura 2.7: Efecto de barril [24].

La más notable forma de distorsión y que siempre es la primera en usarse para extender el modelo es la distorsión *radial* [24]. Se manifiesta en el efecto 'barril' de la figura 2.7, las líneas rectas de la escena ya no se ven rectas en la imagen. Formalmente consiste en cambiar el ángulo que forman los rayos de luz con respecto al eje óptico antes de proyectarlos sobre el sensor digital como se muestra en la figura 2.8. Tiene simetría cilíndrica respecto al eje óptico y consiste en agregar un paso de cálculo intermedio entre la pérdida de perspectiva luego de la roto traslación y la proyección al sensor digital.

El algoritmo 1 condensa en una función los pasos del diagrama de flujo de la figura 2.6. Difiere del modelo *pinhole* en las líneas 7 y 8 donde se transforma el módulo del vector de coordenadas homogéneas $[x_{\mathrm{h}}, y_{\mathrm{h}}]^\top$ según una función f y sus parámetros $(k_1, \dots)$ que solo altera la distancia del vector respecto al centro óptico entregando como resultado las coordenadas homogéneas distorsionadas $[x_{\mathrm{d}}, y_{\mathrm{d}}]^\top$.

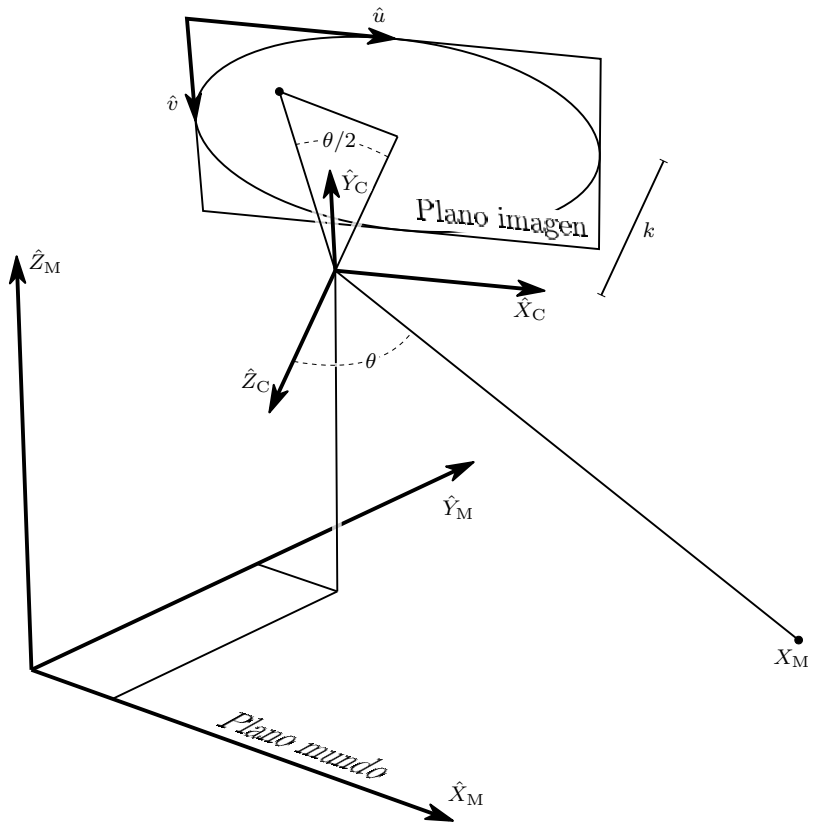


Figura 2.8: El punto X_M es el origen del haz de luz, que forma el ángulo θ respecto al eje óptico de la cámara $\hat{Z}_C$. En el modelo estereográfico el haz se quiebra al pasar por el centro de la cámara y sale hacia atrás formando el ángulo $\theta/2$. Finalmente incide en el sensor donde se forma la imagen, que está a distancia k del centro de la cámara.

2.3.1. Modelo polinómico y cociente de polinomios

La manera estándar [36, 37] de modelar la distorsión radial es aproximar la función de distorsión como una serie de Taylor, es decir, aproximar f como un polinomio en r_h, en símbolos

$$r_d = r_h \left(1 + k_1 r_h^2 + k_2 r_h^4 + k_3 r_h^6\right).$$

Es una aproximación útil cuando la cámara tiene una distorsión leve, es decir que no difiere mucho de una cámara plana excepto por defectos indeseados en la óptica.

En cámaras construidas intencionalmente con un campo de visión más amplio la aproximación polinómica falla cuando la luz llega formando un ángulo grande respecto al eje óptico de la cámara (ver figura 2.8). Se expande el modelo haciendo

Algoritmo 1 Esquema de proyección mundo→imagen con distorsión radial. La función escalar no lineal que distorsiona el radio es f.

1: **function** $\mathcal{F}(X_{\mathrm{M}}, \Theta, \Gamma)$
Variables de entrada: $X_{\mathrm{M}}, \Theta, \Gamma$
Roto traslación y pérdida de profundidad
2: $[r_1, r_2, r_3, t_x, t_y, t_z] \leftarrow \Theta$ ▷ Vector de Rodrigues y de traslación
3: $\mathbf{R} \leftarrow \texttt{Rodrigues}(r_1, r_2, r_3)$ ▷ Matriz de rotación según Rodrigues
4: $$\begin{bmatrix} x_{\mathrm{C}} \\ y_{\mathrm{C}} \\ z_{\mathrm{C}} \end{bmatrix} \leftarrow \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \end{bmatrix} \begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ z_{\mathrm{M}} \\ 1 \end{bmatrix}$$
5: $$\begin{bmatrix} x_{\mathrm{h}} \\ y_{\mathrm{h}} \end{bmatrix} \leftarrow \frac{1}{z_{\mathrm{C}}} \begin{bmatrix} x_{\mathrm{C}} \\ y_{\mathrm{C}} \end{bmatrix}$$
Distorsión no lineal
6: $[f_x, f_y, c_x, c_y, k_1, k_2, \dots] \leftarrow \Gamma$ ▷ Parámetros intrínsecos
7: $r_{\mathrm{h}} \leftarrow \sqrt{x_{\mathrm{h}}^2 + y_{\mathrm{h}}^2}$ ▷ Radio en coordenadas homogéneas
8: $r_{\mathrm{d}} \leftarrow f(r_{\mathrm{h}}, k_1, k_2, \dots)$ ▷ Distorsión radial no lineal
9: $$\begin{bmatrix} x_{\mathrm{d}} \\ y_{\mathrm{d}} \end{bmatrix} \leftarrow \frac{r_{\mathrm{d}}}{r_{\mathrm{h}}} \begin{bmatrix} x_{\mathrm{h}} \\ y_{\mathrm{h}} \end{bmatrix}$$
Proyección al sensor digital
10: $$\begin{bmatrix} u \\ v \end{bmatrix} \leftarrow \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \begin{bmatrix} x_{\mathrm{d}} \\ y_{\mathrm{d}} \\ 1 \end{bmatrix}$$
11: $$X_{\mathrm{I}} \leftarrow \begin{bmatrix} u \\ v \end{bmatrix}$$
Variable de Salida: X_{I}
12: **end function**

el cociente de polinomios [38]

$$r_{\mathrm{d}} = r_{\mathrm{h}} \left(\frac{1 + k_1 r_{\mathrm{h}}^2 + k_2 r_{\mathrm{h}}^4 + k_3 r_{\mathrm{h}}^6}{1 + k_4 r_{\mathrm{h}}^2 + k_5 r_{\mathrm{h}}^4 + k_6 r_{\mathrm{h}}^6} \right). \tag{2.5}$$

Este modelo tiene un vector de parámetros intrínsecos de diez componentes,

$$\Gamma = \{f_x, f_y, c_x, c_y, k_1, k_2, k_3, k_4, k_5, k_6\}.$$

En caso que los parámetros k_4, k_5, k_6 sean cero, el modelo se reduce al caso polinómico.

2.3.2. Modelo ojo de pez según OpenCV

La librería OpenCV [34] define este modelo, conceptualmente combina las propiedades de la función tangente del estereográfico con la adaptabilidad del modelo polinómico. Si el campo de visión se acerca a 180° un polinomio en el radio de las coordenadas será inadecuado porque en los bordes de la imagen r_{h} tiende a infinito.

Se formula f como un polinomio en el ángulo de llegada de la luz (ver figura 2.8), magnitud que no tiene ese problema de divergencia. Se calcula el ángulo,

$$\theta = \arctan(r_{\mathrm{h}}), \tag{2.6}$$

y el radio en escala distorsionada es

$$r_{\mathrm{d}} = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8). \tag{2.7}$$

Este modelo tiene un vector de parámetros intrínsecos de ocho componentes,

$$\Gamma = \{f_x, f_y, c_x, c_y, k_1, k_2, k_3, k_4\}.$$

2.3.3. Modelo estereográfico

Típicamente se aproxima la distorsión radial usando alguno de los modelos anteriores. Los modelos basados en polinomios son prácticos porque computacionalmente son fáciles de tratar: existen rutinas eficientes que calculan sus raíces, ajustan sus coeficientes a un set de datos y calculan sus derivadas. Además pueden aproximar un gran abanico de curvas de distorsión radial. Tienen la desventaja de introducir múltiples parámetros que no son interpretables físicamente. Y como se verá más adelante, es fundamental *estimar* los valores de Γ; la estimación se vuelve más difícil mientras más dimensiones tenga la variable a estimar. En este trabajo se usará una cámara ojo de pez cuya distorsión ya se sabe puede describirse adecuadamente [39] con el modelo *estereográfico* [40] que tiene un solo parámetro. La distorsión óptica se define como

$$\theta = \arctan(r_{\mathrm{h}}), \tag{2.8}$$

$$r_{\mathrm{d}} = k\tan(\theta/2).$$

El parámetro k cumple la función de absorber el escaleo a píxeles haciendo innecesario definir distancias focales. Así la matriz de proyección al sensor digital tiene como valores constantes que $f_x = f_y = 1$. Además k tiene una interpretación física simple: considérese un haz de luz que llega formando un ángulo de $\theta = \pi/2$ respecto al eje óptico, es decir que llega desde un lateral de la cámara. Este rayo pasará por la lente *fisheye* y llegará al sensor digital CCD a k píxeles del centro óptico. Puede comprobarse en la ecuación 2.8 que reempalzando ese valor de θ resulta en $r_{\mathrm{d}} = k$.

Este modelo tiene un vector de parámetros intrínsecos de solo tres componentes,

$$\Gamma = \{k, c_x, c_y\},$$

uno para la distorsión no lineal más dos para definir el centro óptico. En contraste con los modelos polinómico, cociente de polinomios y ojo de pez que tienen siete, diez y ocho parámetros. La interpretabilidad del parámetro facilita la estimación ya que se puede proponer *ad hoc* un valor semilla de la estimación.

2.3.4. Comparación de modelos de distorsión radial

Se realizó un experimento controlado para comparar cualitativamente los modelos de distorsión radial. El objetivo es comprobar cómo ajustan el modelo *pinhole*, polinómico, el cociente de polinomios, el *fisheye* y el estereográfico. Se dispuso la cámara *fisheye* directamente enfrentada a un patrón de puntos equiespaciados formando una grilla [10, 39] como se muestra en la figura 2.9a. El ángulo θ que forman los puntos respecto al eje óptico de la cámara puede calcularse con la distancia conocida entre la cámara y el patrón y así calcular $r_\mathrm{h} = \tan(\theta)$. La distancia radial r de los puntos en la imagen al centro de la misma se extrae manualmente de la imagen capturada (figura 2.9b).

En la figura 2.9c se muestran los puntos experimentales de la relación entre r y r_h. Se hacen ajustes que minimizan el error cuadrático entre éstos y las funciones de distorsión radial:

$$\textit{Pinhole}\text{: } r = f_x r_\mathrm{h}.$$

$$\text{Polinómico: } r = f_x r_\mathrm{h}\left(1 + k_1 r_\mathrm{h}^2 + k_2 r_\mathrm{h}^4 + k_3 r_\mathrm{h}^6\right).$$

$$\text{Cociente de polinomios: } r = f_x r_\mathrm{h}\left(\frac{1 + k_1 r_\mathrm{h}^2 + k_2 r_\mathrm{h}^4 + k_3 r_\mathrm{h}^6}{1 + k_4 r_\mathrm{h}^2 + k_5 r_\mathrm{h}^4 + k_6 r_\mathrm{h}^6}\right).$$

$$\textit{fisheye}\text{: } \begin{cases} \theta = \arctan(r_\mathrm{h}), \\ r = f_x\theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8). \end{cases}$$

$$\text{Estereográfico: } \begin{cases} \theta = \arctan(r_\mathrm{h}), \\ r = k\tan(\theta/2). \end{cases}$$

Se multiplica por f_x porque el radio en la imagen r es igual al radio distorsionado r_d salvo el cambio de escalada dado por la distancia focal f_x, f_y (supuestas iguales para simplificar).

Observar el comportamiento cualitativo de los modelos ajustado en la figura 2.9c. Los modelos cociente de polinomios, *fisheye* y estereográfico describen adecua-

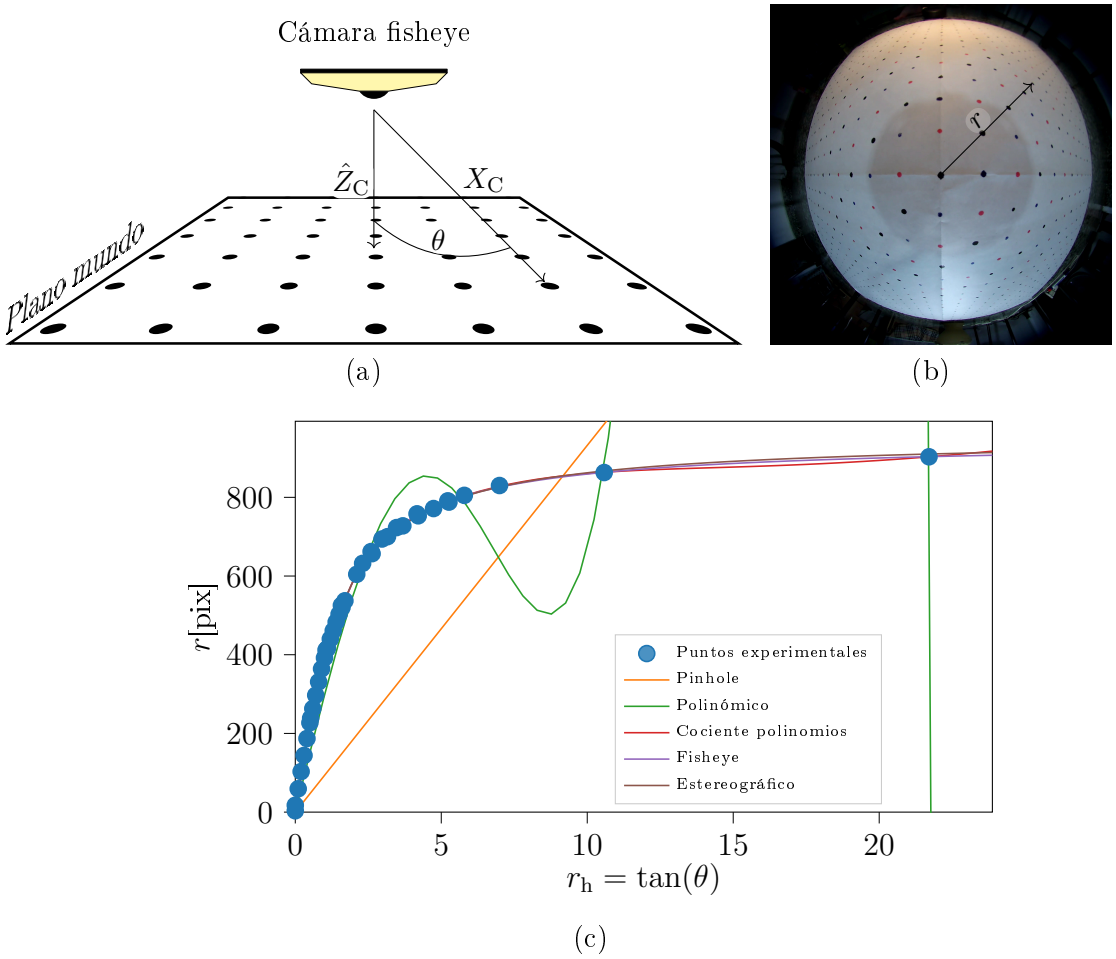


Figura 2.9: (a) se dispone la cámara frente a una grilla de puntos equiespaciados a una distancia conocida. Se calcula $r_h = \tan\theta$. (b) Imagen tomada en el experimento, sobre ella se mide directamente r. (c) Ajuste de los modelos de distorsión a los pares empíricos de (r_h, r).

damente la distorsión de la lente con muy pequeñas diferencias entre ellos, no así el *pinhole* y polinómico. El modelo polinómico para valores altos de r_h va a estar dominado por el coeficiente de orden superior del polinomio y tenderá a hacerse muy positivo o muy negativo dependiendo del signo, entonces no puede representar una forma funcional que tiende a una meseta. Notablemente el modelo estereográfico tiene un solo parámetro y cualitativamente funciona igual de bien que el cociente de polinomios y el *fisheye* que requieren muchos mas parámetros.

Una conclusión importante es que esto pone al estereográfico por encima de los demás, tener un solo parámetro facilita notablemente la manipulación del modelo. Es por eso que en esta tesis se considera que el estereográfico es el más adecuado. Como referencia el ajuste retornó $k = 952 \pm 5$ píxeles.

2.4. Cámara de orientación variable: Pan-Tilt

La PTZ es un modelo popular de cámara de vigilancia. En la figura 2.10 se indican los dos grados de libertad que le permiten apuntar en cualquier dirección. El pan es una rotación alrededor de un eje vertical y el tilt es una rotación alrededor del eje móvil horizontal. El ajuste de Zoom se puede formalizar como una variación de las distancias focales f_x, f_y pero es un grado de libertad que no va a ser estudiado en este trabajo. Salvo que se aclare lo contrario, se va a suponer que la cámara PTZ se deja configurada con valor de zoom fijo de modo que sus parámetros intrínsecos no cambian.

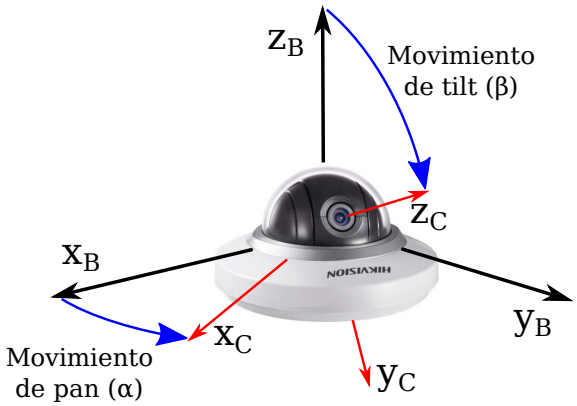


Figura 2.10: El movimiento de pan rota respecto al eje z de la base de la cámara y el movimiento de tilt rota respecto a eje móvil x de la cámara.

El chasis principal o 'la base' de la cámara PTZ se monta fijo en una pared, mástil, etc. y lo que se mueve es la cámara respecto al chasis por acción de motores paso a paso. Para describir la pose de la cámara es necesario diferenciar estos dos marcos de referencia, el de la base y el de la cámara propiamente dicha (o 'lente', para evitar confusión). La descripción del modelo de proyección de una cámara PTZ toma prestados algunos términos y conceptos del área de la robótica.

La roto traslación que convierte coordenadas del mundo a la cámara se compone ahora de dos partes, la primera lleva del marco de referencia del mundo al marco de referencia de la base o chasis y la segunda parte, lleva desde ahí al marco de referencia de la lente. En símbolos

$$X_{\mathrm{C}} = \mathbf{R}_{\mathrm{PT}} \, (\mathbf{R}_{\mathrm{b}} \, X_{\mathrm{M}} + \mathrm{T}_{\mathrm{b}}). \tag{2.9}$$

Donde $\mathbf{R}_{\mathrm{b}}$ y T_{b} cumplen el mismo papel que en la ecuación (2.1). Pero $\mathbf{R}_{\mathrm{PT}}$ está

definida por la actuación de los motores, que en principio están bajo el control del operario, de modo que no es necesario estimar esa matriz de rotación. Para hallarla se examina la matriz de rotación que se obtiene al rotar alrededor del eje x el ángulo β y alrededor del eje z el ángulo α,

$$\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix} =$$

$$= \begin{bmatrix} \cos\alpha & -\sin\alpha\cos\beta & -\sin\alpha\sin\beta \\ \sin\alpha & \cos\alpha\cos\beta & \cos\alpha\sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}.$$

Las columnas son las coordenadas de los versores de la cámara expresados en el marco de referencia de la base. Por ejemplo la primera columna, $[\cos\alpha, \sin\alpha, 0]^\top$, es claramente el versor x de la cámara pues su proyección en z de la base es siempre cero (está en el plano horizontal) y se descompone sobre el plano x-y como si se tratara de coordenadas polares. La tercera columna,

$$\begin{bmatrix} -\sin(\alpha)\sin(\beta) \\ \cos(\alpha)\sin(\beta) \\ \cos(\beta) \end{bmatrix}^\top ,$$

corresponde a la representación de un vector de módulo uno en coordenadas esféricas. Es decir que esa matriz rota un vector en marco de referencia cámara a marco de referencia base. $\mathbf{R}_{\mathrm{PT}}$ es entonces su traspuesta [32, 41]:

$$\mathbf{R}_{\mathrm{PT}}(\alpha,\beta) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha\cos\beta & \cos\alpha\cos\beta & -\sin\beta \\ -\sin\alpha\sin\beta & \cos\alpha\sin\beta & \cos\beta \end{bmatrix}.$$

2.5. Calibración

En el contexto de visión artificial *calibrar* significa estimar los parámetros del modelo de formación de imagen. La calibración intrínseca puede hacerse en condiciones controladas porque la óptica, la resolución del sensor y demás detalles constructivos son independientes de la localización de la cámara. Una vez instalada en su posición definitiva se puede hacer la calibración extrínseca para estimar el vector de traslación y los parámetros de orientación. En esta sección se describen los algoritmos

estándar de calibración.

Para realizar las estimaciones se requieren datos patrón, un modelo que describa la relación entre los datos y un criterio de optimalidad [42, 43]. El modelo es $\mathcal{F}$, descripto en el algoritmo 1. Los datos serán recabados fotografiando un objeto patrón y el criterio de optimalidad es minimizar el error cuadrático.

2.5.1. Calibración intrínseca con distorsión óptica

El algoritmo propuesto por Zhang [37] implementado en [36] se ha vuelto un estándar debido a su flexibilidad y facilidad de uso. Consiste en usar un patrón de calibración plano como se muestra la figura 2.11, ejemplo extraído de Bouguet [36], típicamente un tablero de cuadrados blancos y negros impreso en papel, ubicarlo frente a la cámara y tomar alrededor de una decena de fotografías cambiando la pose relativa entre el tablero y la cámara. Los puntos de calibración serán los vértices internos del tablero.

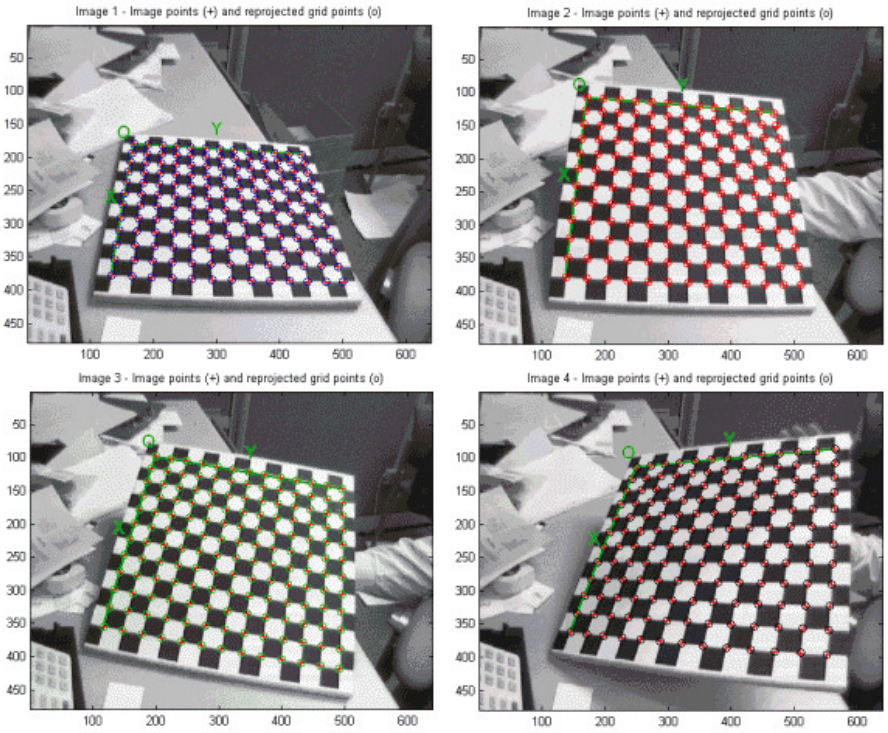
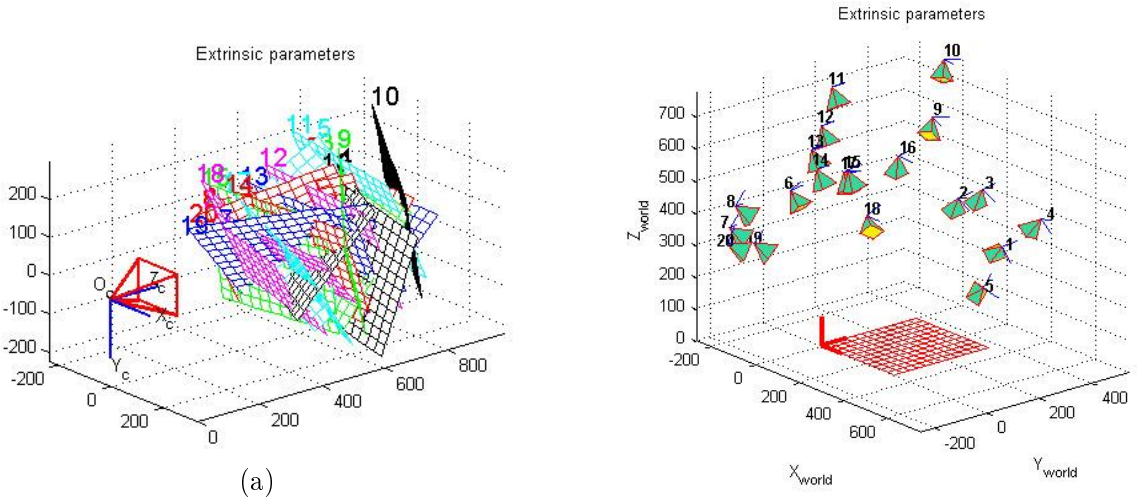


Figura 2.11: Cuatro imágenes del patrón del calibración y las esquinas internas detectadas [36]. Las esquinas internas del tablero se indican con un '+' y son las que se usan para estimar los parámetros de proyección. Los puntos de proyección se indican con '$\circ$' y coinciden con los vértices cuando la estimación es exitosa.



(a) (b)

Figura 2.12: Uso del toolbox de Bouget [36], en (a) se muestran las posiciones estimadas del tablero respecto a la cámara y en (b) su equivalente desde el marco de referencia del tablero.

El patrón de calibración es plano y si se pone el marco de referencia del mundo solidario al tablero los puntos de calibración tienen $z_{\mathrm{M}} = 0$ y sus coordenadas x-y están dadas por la construcción del patrón (en general una grilla de celdas cuadradas de lado unitario). Se usan algoritmos estándar para extraer las coordenadas en píxeles de los vértices del tablero, en la figura 2.11 se indican con '+'. En cada una de las m imágenes hay n puntos de calibración. Se tiene entonces una lista de pares de coordenadas en el mundo $\{X_{\mathrm{M}i,j}\}$ y en la imagen $\{X_{\mathrm{I}i,j}\}$ que se corresponden, con $j \in [1, \ldots, m]$ e $i \in [1, \ldots, n]$. La calibración intrínseca consiste en estimar los parámetros intrínsecos y extrínsecos tal que se minimice el error de proyección, en símbolos

$$\hat{\Gamma}, \{\hat{\Theta}_j\} = \underset{\Gamma, \{\Theta_j\}}{\arg\min} \sum_{j,i} \left| X_{\mathrm{I}i,j} - \mathcal{F}(X_{\mathrm{M}i,j}, \Gamma, \Theta_j) \right|^2 . \tag{2.10}$$

Además de los intrínsecos que son los que se quiere estimar, forzosamente hay que estimar los extrínsecos para cada imagen. En la figura 2.12 extraída de Bouguet [36] se muestran las posiciones estimadas de la cámara tomando como marco de referencia al mundo y viceversa. Para evaluar si la calibración fue exitosa se usan los parámetros estimados para proyectar las posiciones de los puntos de la grilla $\{X_{\mathrm{M}i,j}\}$ hacia la imagen. En la figura 2.11 se indican con '$\circ$' y puede verse que coinciden con los vértices del tablero.

2.5.2. Calibración extrínseca

El problema *Perspective-n-Point* (PnP) consiste en determinar la posición y orientación de una cámara a partir de la observación de puntos de calibración conocidos y los parámetros intrínsecos de la cámara [44, 45]. OpenCV (ver figura 2.13) implementa varias soluciones. En todas ellas se toman n puntos 3D con sus correspondientes proyecciones en la imagen y los parámetros intrínsecos $\hat{\Gamma}$ calibrados como en la sección anterior 2.5.1, y devuelven una estimación del vector de Rodrigues y el de traslación ($\hat{\Theta}$) minimizando el error de proyección. Es decir que optimizan el mismo funcional que en la calibración intrínseca (ecuación (2.10)),

$$\hat{\Theta} = \arg\min_{\Theta} \sum_{i} \left| X_{\mathrm{I}i} - \mathcal{F}(X_{\mathrm{M}i}, \hat{\Gamma}, \Theta) \right|^2 . \tag{2.11}$$

La lista de pares de coordenadas en el mundo $\{X_{\mathrm{M}i}\}$ y en la imagen $\{X_{\mathrm{I}i}\}$ se indexan por el punto de calibración con $i \in [1, \ldots, n]$. Los puntos 3D de calibración pueden ser coplanares esta vez, que es justamente el caso que se va a trabajar, tanto los puntos de calibración como los objetos de interés estarán en el plano horizontal del mundo.

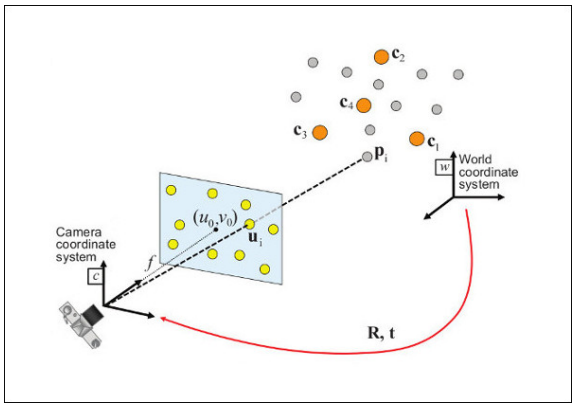
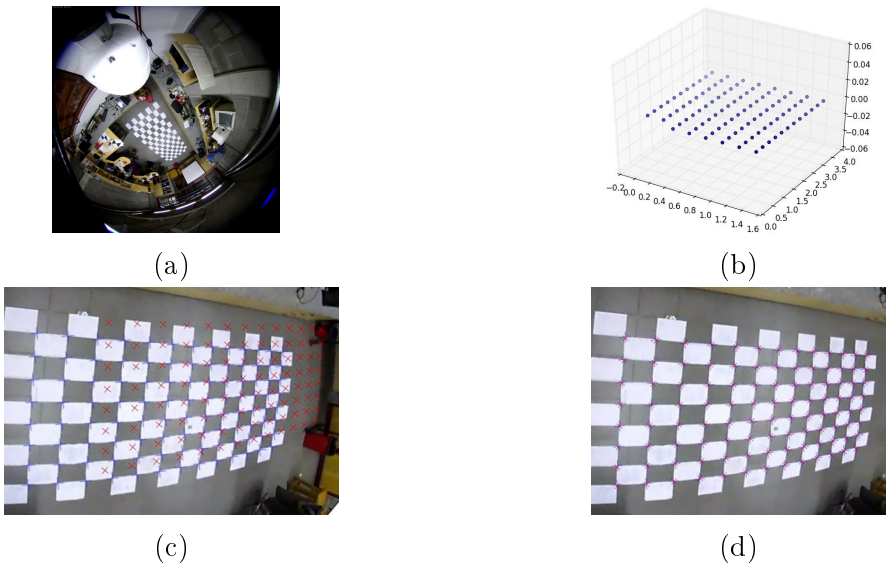


Figura 2.13: Documentación de la función `solvePnP` [34]. Las coordenadas de los puntos de calibración en el marco de referencia mundo y sus coordenadas marcadas en la imagen permiten estimar la pose de la cámara.

2.5.3. Ejemplo de calibración extrínseca

En la figura 2.14 se muestra el resultado de una calibración extrínseca. Se tiene una imagen capturada con la cámara *fisheye* de parámetros intrínsecos conocidos. Se colocó en el suelo un patrón tipo grilla. Las esquinas internas del mismo pueden detectarse en la figura usando funciones de OpenCV, estas son las coordenadas



(a) (b)

(c) (d)

Figura 2.14: (a) Imagen tomada por la cámara *fisheye* del patrón de calibración. (b) Posiciones 3D en el marco de referencia mundo de las esquinas internas. (c) Proyecciones de las esquinas internas usando esos valores semilla en rojo y en azul las esquinas internas detectadas usando OpenCV. (d) Proyecciones usando la pose optimizada.

'imagen' del patrón de calibración. Sus coordenadas 'mundo' se hallan físicamente en el plano del suelo formado una grilla que se muestra en la figura 2.14b.

La estimación de la pose se hace minimizando el error cuadrático de proyección en la imagen. Esa minimización necesita valores semilla que se proponen *ad hoc*. Se midió con cinta métrica la posición de la cámara en el marco de referencia mundo (que tiene origen de coordenadas en el una de las esquinas internas de la grilla en el suelo). El vector que une el punto del suelo que está en el centro de la imagen con la posición de la cámara definen la dirección del eje óptico. Con esas medidas se aproximó el vector de traslación y el vector de Rodrigues para iniciar la optimización:

$$\Theta_{inicial} = [0.980,\ 2.82,\ 0.505,\ 0,\ 0,\ 2.7].$$

La proyección de los puntos de calibración con estos valores de pose puede verse en la figura 2.14c, es aceptable como semilla de la optimización. En pruebas exploratorias el algoritmo demostró tener una cuenca de convergencia bastante grande, no siendo necesario proponer una pose inicial precisa.

Se optimiza Θ con respecto al error de proyección y se obtiene

$$\Theta_{optimizado} = [0.877,\ 2.81,\ 0.546,\ -0.500,\ -0.437,\ 2.56].$$

En la figura 2.14d se ve que se ha convergido a un valor de pose que permite una

proyección precisa.

2.6. Retro-proyección de pinhole mediante proyección de perspectiva

Aquí se presenta el concepto de *retro-proyección* para el modelo *pinhole*. La idea de retro-proyección está omnipresente en toda la tesis, significa calcular las coordenadas mundo de un objeto a partir de las coordenadas imagen. El modelo de formación de imagen es un mapeo mundo (3D) $\rightarrow$ imagen (2D) y en general no puede calcularse el mapeo inverso por la pérdida de información de distancia. Pero si se restringen los grados de libertad del objeto con alguna hipótesis extra, la retro-proyección sí puede calcularse.

Como ilustra la figura 2.15 [46] la luz que va del objeto a la cámara sigue una línea recta que interseca el plano imagen. Si las coordenadas de los objetos están todas contenidas en el mismo plano la fórmula del modelo *pinhole* se reduce a una homografía plano a plano. Sin perder generalidad se supone al plano donde está el objeto como el plano horizontal de $z_{\mathrm{M}} = 0$. Recordando la ecuación (2.4) del modelo *pinhole*

$$z_{\mathrm{C}} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\mathbf{R} \,\middle|\, \mathrm{T} \right] \begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ z_{\mathrm{M}} \\ 1 \end{bmatrix}.$$

Pero si $z_{\mathrm{M}} = 0$ la tercera columna de la matriz $[\mathbf{R} \mid \mathrm{T}]$ se omite y se llega a la expresión general de la transformación de perspectiva

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H \begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ 1 \end{bmatrix}.$$

Donde H es la matriz homogénea de 3×3 que representa la pose relativa entre los planos y se ha reemplazado z_{C} por s que absorbe la distancia del objeto a la cámara y también un factor de escala de la matriz H. Es decir que todas las matrices que sean múltiplo de H dan como resultado la misma proyección.

El mapeo al revés, imagen $\rightarrow$ mundo, se calcula con la inversa de H (los casos

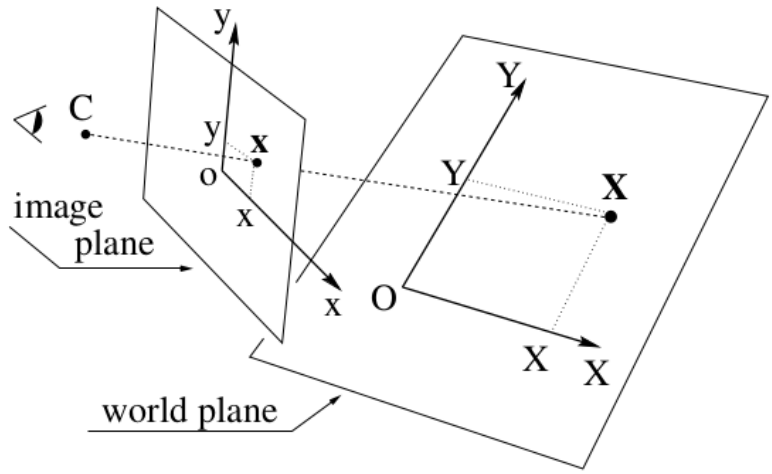


Figura 2.15: Homografía plano a plano (extracto de Criminisi et al. [46]). Para convertir a la notación del presente trabajo:
Coordenadas en el plano mundo: $\mathbf{X} \equiv [X, Y]^\top \rightarrow X_\mathrm{M} \equiv [x_\mathrm{M}, y_\mathrm{M}, 0]^\top$
Coordenadas en el plano imagen: $\mathbf{x} \equiv [x, y]^\top \rightarrow X_\mathrm{I} \equiv [u, v]^\top$

en que no sea invertible no son de interés) haciendo

$$\frac{1}{s}\begin{bmatrix} x_\mathrm{M} \\ y_\mathrm{M} \\ 1 \end{bmatrix} = H^{-1}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}.$$

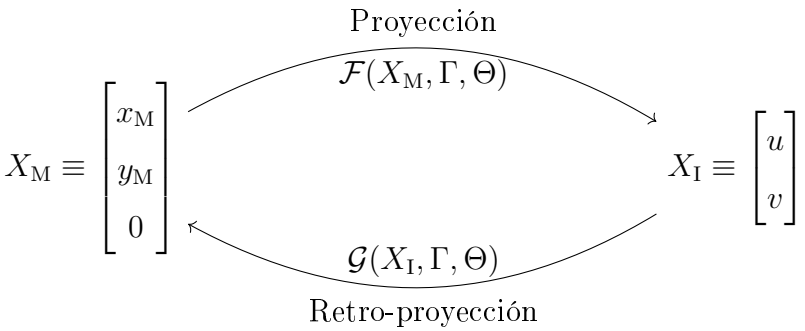
Que no es más que otra proyección de perspectiva.

2.7. Resumen de retro-proyección

Acotar el problema de fotogrametría a aplicaciones de monitoreo de vehículos en un ambiente urbano trae dos condicionantes. El primero es que no se tiene información de distancia, solo se tiene un sistema de visión monocular. Eso es porque las cámaras que se instalan en la vía pública no se acompañan de un sensor complementario que mida distancias debido a su alto costo (lidar, radar, etc). Las cámaras son ubicadas tal que sus campos de visión tengan poco solapamiento para aumentar el área cubierta con un mínimo de cámaras. Eso descarta la posibilidad de usar visión binocular. Pero el segundo condicionante es que los objetos que interesa detectar son peatones y vehículos, ambos se mueven sobre el suelo (asfalto y vereda), es decir que a efectos prácticos no tienen tres grados de libertad sino dos. Eso reduce la cantidad de incógnitas y hace que calcular la posición física del objeto sea posible teniendo una sola imagen.

Pero usar una proyección de perspectiva no es suficiente. Formalmente los objetos de interés tienen $z_{\mathrm{M}} = 0$. Si la cámara obedece el modelo *pinhole* entonces la retro-proyección puede resolverse como se explicó arriba usando una proyección de perspectiva. Pero las cámaras que interesa trabajar son ojo de pez y tienen una severa distorsión óptica, el modelo *pinhole* no sirve. Además se quiere respetar la parametrización estándar de OpenCV que describe la pose con Θ y la distorsión como Γ que son parámetros interpretables y se pueden calibrar por separado (no como la matriz H). Esto motiva a desarrollar una solución de retro-proyección nueva.

El modelo de formación de imagen con distorsión óptica se simboliza con $\mathcal{F}$ (detallado en secciones anteriores) y su inversa, que es posible suponiendo $z_{\mathrm{M}} = 0$, es $\mathcal{G}$ (se la resuelve en el capítulo 3). Ambas funciones necesitan los parámetros de pose y distorsión Γ, Θ como ilustra el siguiente diagrama:



El mapeo sobre el que se enfoca esta tesis es la retro-proyección, considerando distorsión radial no lineal y tomando como parámetros Γ, Θ. Así, a partir de detectar un objeto en la imagen en la posición X_{I} se puede calcular su posición X_{M} ubicada en el plano del suelo usando los parámetros intrínsecos Γ (estimados en la instancia de calibración intrínseca) y los extrínsecos Θ (estimados en la instancia de calibración extrínseca).

2.8. Estadística bayesiana

En las subsecciones anteriores se muestra el modelo físico que vincula 'rígidamente' las variables y los parámetros punto a punto, es decir que las variables $X_{\mathrm{M}}, X_{\mathrm{I}}$ y los parámetros Θ, Γ deben siempre cumplir la relación matemática del modelo de

proyección. Los métodos bayesianos [42, 43, 47] permiten naturalmente y fácilmente: (1) relacionar el modelo físico con datos experimentales, (2) incorporar información externa a los datos (información a priori) y (3) hacer predicciones cuantificando la incerteza. Se puede pensar los métodos bayesianos en tres pasos, primero se define el *modelo probabilístico* del sistema. Segundo, se usa la regla de Bayes para inferir información de los parámetros a partir de los datos e información a priori [42]. Y finalmente se predice un dato nuevo.

En esta sección se presentan las bases del proceso de inferencia y predicción bayesiana. Se discutirá la necesidad de una forma de regresión bayesiana que incorpore incerteza en la variable de entrada. En el capítulo 5 se deriva desde un planteo bayesiano general las ecuaciones que se usarán para inferencia y predicción.

2.8.1. Inferencia y predicción

Para repasar la teoría se usa una forma simplificada de la notación de las subsecciones anteriores. Considérese la variable X asociada a una magnitud física que se puede medir directamente y el parámetro Θ. Se denota un conjunto de valores independientes medidos experimentalmente de la variable como $\mathcal{D} = \{X'_1, \cdots, X'_n\}$. Se plantea primero un modelo de la probabilidad conjunta entre la variable y el parámetro

$$p(X, \Theta) = p(X|\Theta)p(\Theta).$$

Condicionando en los datos conocidos y aplicando la regla de Bayes se obtiene la probabilidad a posteriori del parámetro,

$$p(\Theta|\mathcal{D}) = \frac{p(\mathcal{D}|\Theta)p(\Theta)}{p(\mathcal{D})} \propto p(\mathcal{D}|\Theta)p(\Theta).$$

Donde $p(\mathcal{D}|\Theta) = \prod_i (X'_i|\Theta)$ porque los datos son independientes. Este término se denomina verosimilitud. Y $p(\Theta)$ es la probabilidad a priori del parámetro En la forma proporcional de la regla de Bayes el denominador se omite (es una constante de normalización que no depende de Θ), que en palabras es

$$\text{posterior} \propto \text{verosimilitud} \times \text{prior}$$

Si se quiere inferir información sobre un dato nuevo X, es decir hacer una predicción en base a los datos tomados anteriormente la probabilidad predictiva a posteriori

es

$$p(X|\mathcal{D}) = \int p(X|\Theta)p(\Theta|\mathcal{D})\mathrm{d}\Theta$$

El paso de inferir información sobre los parámetros en el contexto de visión artificial se llama calibración, y $\mathcal{D}$ son los datos de calibración.

Tanto $p(\Theta|\mathcal{D})$ como $p(X|\mathcal{D})$ son los enunciados probabilísticos que se obtienen como resultado de la inferencia. Pero presentar una distribución de probabilidad suele ser muy abstracto y comúnmente se presenta un único valor, que es mucho más fácil de comunicar. Es lo que se llama una estimación puntual. Por ejemplo reportar como resultado de la calibración de Θ el valor del parámetro que tenga máxima probabilidad a posteriori $\Theta_{\mathrm{Cal}} = \arg\max\{p(\Theta|\mathcal{D})\}$ o el que sea la esperanza de la probabilidad a posteriori $\Theta_{\mathrm{Cal}} = \mathbf{E}[p(\Theta|\mathcal{D})]$.

2.8.2. Cadenas de Markov y Metrópolis-Hastings

Simulación de Monte Carlo por Cadena de Markov [42] (MCMC por sus siglas en inglés) es un método general basado en extraer secuencialmente valores de Θ tal que la población de muestras obtenidas sirva como aproximación a la distribución de probabilidad objetivo $p(\Theta|\mathcal{D})$. Es útil cuando, por no conocerse una forma paramétrica de la distribución, no se puede tratar analíticamente. La cadena de Markov empieza en un valor semilla Θ^0 y se extraen los valores sucesivos usando una distribución de transición que depende de la muestra anterior.

Dentro de la familia de algoritmos de cadena de Markov está Metrópolis-Hastings. Los pasos del algoritmo son

1. Extraer una valor candidato a ser la próxima muestra Θ^* de la cadena usando una distribución de propuestas $J(\Theta^*|\Theta^{i-1})$.

2. Calcular el cociente

$$r = \frac{p(\Theta^*|\mathcal{D})/J(\Theta^*|\Theta^{i-1})}{p(\Theta^{i-1}|\mathcal{D})/J(\Theta^{i-1})|\Theta^*)}.$$

3. Establecer la muestra que sigue en la cadena como

$$\Theta^i = \begin{cases} \Theta^* & \text{con probabilidad mín}(r, 1) \\ \Theta^{i-1} & \text{en cualquier otro caso.} \end{cases}$$

El objetivo es iterar suficiente cantidad de veces para que la distribución de las muestras haya convergido a la distribución objetivo. Se simulan en paralelo varias cadenas de Markov, y se comprueba la convergencia comparando las poblaciones de las cadenas entre si. Estos algoritmos son estándares, están implementados en librerías como PyMC3 [48].

2.9. Resumen del capítulo y relación con los capítulos posteriores

En este capítulo se presentó la base de conocimientos en que se apoya el resto de la tesis. El modelo de formación de imagen describe la proyección de una posición 3D en el mundo sobre la imagen 2D. Comenzando por el modelo *pinhole*, luego incorporando la distorsión radial no lineal y el modelo para cámara PTZ.

En el capítulo 3 se muestra el mapeo inverso que permite retro-proyectar desde la imagen al mundo (suponiendo al objeto sobre un plano) y en el capítulo 4 se lo aplica a varios problemas prácticos. Para estimar los parámetros de distorsión de una cámara el método estándar es obtener varias capturas de un tablero cuadriculado plano y aplicar el método de Zhang [37]. Con los parámetros intrínsecos y la cámara montada en su lugar definitivo se puede hacer la estimación de la pose. Estos algoritmos son el punto de partida para desarrollar la retro-proyección del capítulo 3 y luego las aplicaciones que se muestran en el capítulo 4. Como se ve en esos capítulos el tratamiento estadístico de la calibración y la retro-proyección necesita un mayor desarrollo.

La teoría de inferencia bayesiana es la base para el capítulo 5 donde se parte de primeros principios y se llega al procedimiento de calibración y predicción que se pone a prueba en el capítulo 6.

Capítulo 3

Retro-proyección

La formación de una imagen implica la pérdida de información tridimensional, por eso para deducir la posición de objetos en el mundo no alcanza con la información de una sola imagen. Típicamente se usan dos o más imágenes usando el formalismo de visión binocular [25], o se agrega un sensor de profundidad como las cámaras Kinect [49, 50], o se usa un objeto de geometría conocida [51, 52]. En este problema no se puede suponer que se tiene más de una cámara, ni que hay sensores adicionales, pero está claro que los objetos de interés van a estar moviéndose sobre el suelo, es decir pueden restringirse las posiciones física 3D de lo que se detecte en la imagen a un plano horizontal. Con esa hipótesis, $z_{\mathrm{M}} = 0$, se suplanta la información de distancia a la cámara. Como ilustra la figura 3.1 se puede pensar que el modelo de proyección puede invertirse calculándose la retro-proyección. El mapeo $X_{\mathrm{M}} = \mathcal{G}(X_{\mathrm{I}}, \Gamma, \Theta)$ calcula la posición en el plano del suelo X_{M} dadas las posiciones en la imagen X_{I}, los parámetros intrínsecos Γ y extrínsecos Θ.

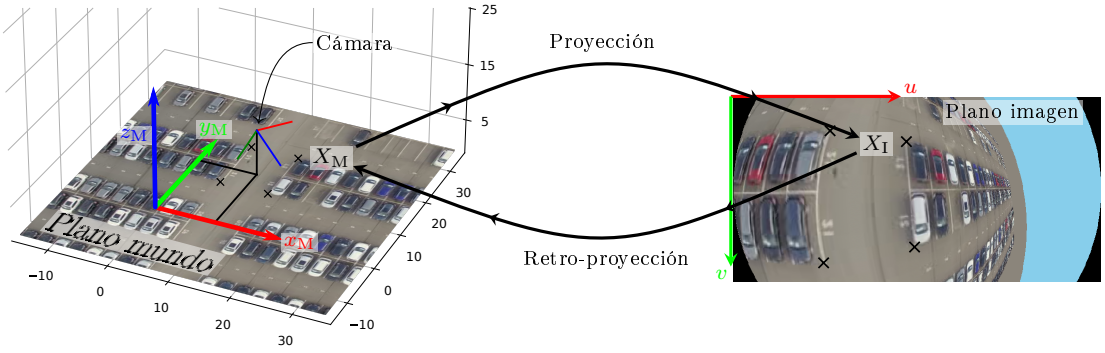


Figura 3.1: Esquema conceptual de la proyección de coordenadas 3D (supuestas en el plano del suelo) a la imagen 2D y la retro-proyección.

En este capítulo se resuelve la retro-proyección, es decir, calcular las posiciones físicas de los objetos detectados en la imagen suponiendo que se hallan en el suelo. Para el modelo *pinhole* se resuelve el equivalente a una proyección de perspectiva, luego se agrega el tratamiento de las funciones de distorsión radial.

Adicionalmente se trata un problema relacionado: apunte de la PTZ. La cámara PTZ puede apuntar moviéndose en Pan y Tilt, la pregunta es: conociendo la posición física de un objeto, ¿qué Pan y Tilt hacen que el objeto se vea centrado en la imagen?

3.1. Base general, retro-proyección de pinhole

Se tiene u y v (la posición en imagen de algún objeto de interés), los parámetros intrínsecos de la cámara (Γ) y su rototraslación respecto al mundo ($\mathbf{R}, \mathrm{T}$): hay que calcular X_M. Se repetirán los pasos del algoritmo 1 en orden inverso pero dejando para la sección que sigue el tratamiento de la distorsión radial. Se comienza revirtiendo la proyección al CCD (línea 10 del algoritmo 1):

$$\begin{bmatrix} x_\mathrm{d} \\ y_\mathrm{d} \end{bmatrix} = \begin{bmatrix} (u - c_x)/f_x \\ (v - c_y)/f_y \end{bmatrix}.$$

A continuación se pueden obviar las líneas 7-9 del algoritmo 1 porque no hay distorsión radial, entonces $[x_\mathrm{d}, y_\mathrm{d}]^\top = [x_\mathrm{h}, y_\mathrm{h}]^\top$.

Las líneas 4 y 5 del algoritmo 1, es decir, las referidas a la rototraslación (ver las

ecuaciones (2.1) y (2.3) que describen el modelo colineal [35]) se reescriben como

$$z_{\mathrm{C}} \begin{bmatrix} x_{\mathrm{d}} \\ y_{\mathrm{d}} \\ 1 \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ 0 \end{bmatrix} + \mathrm{T}.$$

Donde ya se ha reemplazado $z_{\mathrm{M}} = 0$. Las incógnitas son $x_{\mathrm{M}}, y_{\mathrm{M}}$ pero también se necesita resolver z_{C}. Es un sistema de tres ecuaciones con tres incógnitas. Despejando $x_{\mathrm{M}}, y_{\mathrm{M}}$ se simplifica el sistema

$$\begin{bmatrix} x_{\mathrm{M}} \\ y_{\mathrm{M}} \\ 0 \end{bmatrix} = z_{\mathrm{C}} \mathbf{R}^{\top} \begin{bmatrix} x_{\mathrm{d}} \\ y_{\mathrm{d}} \\ 1 \end{bmatrix} - \mathbf{R}^{\top}\mathrm{T} = z_{\mathrm{C}}\ X'_{\mathrm{d}} - \mathrm{T}'. \tag{3.1}$$

Donde los vectores auxiliares $X'_{\mathrm{d}} \equiv [X'_{\mathrm{d}1}, X'_{\mathrm{d}2}, X'_{\mathrm{d}3}]^{\top}$ y $\mathrm{T}' \equiv [\mathrm{T}'_1, \mathrm{T}'_2, \mathrm{T}'_3]^{\top}$ pueden calcularse porque $x_{\mathrm{d}}, y_{\mathrm{d}}, \mathbf{R}, \mathrm{T}$ son conocidos. Se resuelve fácilmente $z_{\mathrm{C}} = \mathrm{T}'_3 / X'_{\mathrm{d}3}$ con la tercera fila (ecuación) del sistema. Nótese que es justamente este paso de la resolución que se habilita por la hipótesis de que el objeto está en el suelo: $z_{\mathrm{M}} = 0$. Para terminar:

$$x_{\mathrm{M}} = z_{\mathrm{C}} X'_{\mathrm{d}1} - \mathrm{T}'_1,$$
$$y_{\mathrm{M}} = z_{\mathrm{C}} X'_{\mathrm{d}2} - \mathrm{T}'_2.$$

Es decir que conociendo la rototraslación de la cámara y sus parámetros intrínsecos se puede en pocos sencillos pasos calcular la retro-proyección de una coordenada en la imagen al plano del suelo. En el algoritmo 2 se resumen estos pasos contemplando, además, el caso donde hay distorsión radial.

3.2. Funciones inversas de distorsión radial no lineal

En el modelo de formación de imagen (capítulo 2, sección 2.3) se calcularon las coordenadas distorsionadas usando $r_{\mathrm{d}} = f(r_{\mathrm{h}}, \Gamma)$, ahora se necesita invertir esa función tal que $r_{\mathrm{h}} = g(r_{\mathrm{d}}, \Gamma) = f^{-1}(r_{\mathrm{d}}, \Gamma)$. Se va a despejar algebraicamente r_{h} a partir de r_{d}, Γ partiendo de las definiciones de f para cada modelo.

Algoritmo 2 Esquema genérico de proyección imagen→mundo con distorsión radial. La función de distorsión radial f se usó en algoritmo 1, aquí se usa su inversa $g = f^{-1}$.

function $\mathcal{G}(X_\mathrm{I}, \Theta, \Gamma)$

Proyección desde sensor digital a homogéneas

$[k_1, \ldots, k_6, f_x, f_y, c_x, c_y] \leftarrow \Gamma$ ▷ Parámetros intrínsecos

$$\begin{bmatrix} x_\mathrm{d} \\ y_\mathrm{d} \end{bmatrix} \leftarrow \begin{bmatrix} (u - c_x)/f_x \\ (v - c_y)/f_y \end{bmatrix}$$

Distorsión no lineal

$r_\mathrm{d} \leftarrow \sqrt{x_\mathrm{d}^2 + y_\mathrm{d}^2}$ ▷ Radio distorsionado

$r_\mathrm{h} \leftarrow g(r_\mathrm{d}, k_1, \ldots)$ ▷ Inversa de distorsión radial

$$\begin{bmatrix} x_\mathrm{h} \\ y_\mathrm{h} \end{bmatrix} \leftarrow \frac{r_\mathrm{h}}{r_\mathrm{d}} \begin{bmatrix} x_\mathrm{d} \\ y_\mathrm{d} \end{bmatrix}$$

Retro-proyección al plano horizontal

$[r_1, r_2, r_3, t_x, t_y, t_z] \leftarrow \Theta$ ▷ Vector de Rodrigues y de traslación

$\mathbf{R} \leftarrow \texttt{Rodrigues}(r_1, r_2, r_3)$ ▷ Matriz de rotación según Rodrigues

$\mathrm{T} \leftarrow [t_x, t_y, t_z]$ ▷ Vector de traslación

$$X'_\mathrm{d} \leftarrow \mathbf{R}^\top \begin{bmatrix} x_\mathrm{d} \\ y_\mathrm{d} \\ 1 \end{bmatrix}$$ ▷ Cálculos auxiliares

$\mathrm{T}' \leftarrow \mathbf{R}^\top \mathrm{T}$

$z_\mathrm{C} \leftarrow \mathrm{T}'_3 / X'_{\mathrm{d}3}$

$x_\mathrm{M} \leftarrow z_\mathrm{C} X'_{\mathrm{d}1} - \mathrm{T}'_1$ ▷ Coordenadas en el plano mundo

$y_\mathrm{M} \leftarrow z_\mathrm{C} X'_{\mathrm{d}2} - \mathrm{T}'_2$

end function

3.2.1. Inversa de modelos polinómico y cociente de polinomios

Reordenando la ecuación (2.5) se obtiene

$$0 = k_3\, r_\mathrm{h}^7 - r_\mathrm{d} k_6\, r_\mathrm{h}^6 + k_2\, r_\mathrm{h}^5 - r_\mathrm{d} k_5\, r_\mathrm{h}^4 + k_1\, r_\mathrm{h}^3 - r_\mathrm{d} k_4\, r_\mathrm{h}^2 + r_\mathrm{h} - r_\mathrm{d},$$

un polinomio en r_h cuyos coeficientes se calculan a partir de los parámetros k y r_d (conocidos). No existen soluciones de forma cerrada para las raíces de un polinomio de grado 7 pero la solución numérica esta disponible en muchas librerías de análisis numérico[53] (por ejemplo la función `roots` de la librería Numpy[54]). Se define la inversa de la función de distorsión como

$$r_\mathrm{h} = g(r_\mathrm{d}, k_1, \ldots) = \texttt{roots}([k_3, -r_\mathrm{d} k_6, k_2, -r_\mathrm{d} k_5, k_1, -r_\mathrm{d} k_4, 1, -r_\mathrm{d}]).$$

De las siete raíces que retorne se selecciona la menor entre las que sean reales y positivas.

3.2.2. Inversa de ojo de pez según OpenCV

Recordando de las ecuaciones (2.6) y (2.7) se necesita calcular primero θ a partir de r_d (resolver la raíz de un polinomio de grado nueve) y después r_h a partir de θ (inversa de arcotangente). En símbolos

$$\theta = \texttt{roots}([k_4, 0, k_3, 0, k_2, 0, k_1, 0, 1, -r_\mathrm{d}]),$$
$$r_\mathrm{h} = \tan(\theta).$$

3.2.3. Inversa de estereográfico

La ecuación (2.8) se invierte fácilmente

$$\theta = 2\arctan(r_\mathrm{d}/k),$$
$$r_\mathrm{h} = \tan(\theta).$$

Así, la formulación del modelo de retro-proyección de la imagen al plano horizontal del mundo queda completa. Consiste en conocer de antemano los parámetros de distorsión de la cámara $(k_1, k_2, \dots)$ y su pose, $\mathbf{R}, \mathrm{T}$; aplicar el algoritmo 2 reemplazando la función que invierte la distorsión radial en la línea 5 según el modelo de distorsión: polinómico, cociente de polinomios, ojo de pez o estereográfico.

3.3. Apunte de cámara pan-tilt

La versatilidad de la cámara Pan-Tilt es que puede apuntar en la dirección que se desee, y por ejemplo poner a un objeto de interés en el centro de la imagen. Si se conoce por algún sensor externo la posición en el marco de referencia mundo de un objeto de interés, ¿qué ángulos de pan y tilt ubican la cámara tal que el objeto se vea en el centro de la imagen? En la figura 3.2 se ilustra el problema. El punto de observación tiene coordenadas conocidas X_M y también se conoce la pose de la base de la cámara, es decir la rototraslación que lleva desde el marco de referencia del mundo a la base. El problema se plantea formalmente pidiendo que el versor $\hat{z_\mathrm{C}}$ de la lente (que es el eje óptico de la cámara) esté en la dirección del punto de observación. Entonces la posición del objeto en el marco de referencia de la cámara solo tiene componente z, en símbolos $X_\mathrm{C} = [0, 0, \|X_\mathrm{C}\|]^\top$.

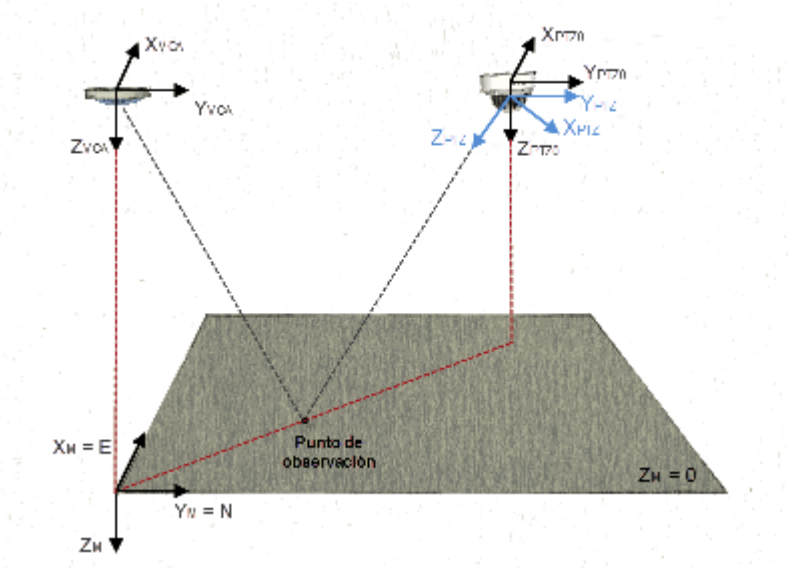


Figura 3.2: Medir en la *fisheye* y después apuntar con la PTZ.

Esta expresión debe reemplazarse en $X_{\mathrm{C}} = \mathbf{R}_{\mathrm{PT}}(\alpha, \beta)\ (\mathbf{R}_{\mathrm{b}}\ X_{\mathrm{M}} + \mathrm{T}_{\mathrm{b}})$, (ecuación (2.9)) para resolver α, β. Pero se supone conocida la posición del objeto en el mundo, X_{M}, y también la posición de la base de la cámara, $\mathbf{R}_{\mathrm{b}}, \mathrm{T}_{\mathrm{b}}$, entonces se puede directamente trabajar con

$$X_{\mathrm{b}} = \mathbf{R}_{\mathrm{b}}\ X_{\mathrm{M}} + \mathrm{T}_{\mathrm{b}}. \tag{3.2}$$

Y como la matriz $\mathbf{R}_{\mathrm{PT}}$ es de determinante 1 por ser una matriz de rotación entonces $\|X_{\mathrm{C}}\| = \|X_{\mathrm{b}}\|$. Con estos reemplazos el sistema a resolver queda

$$\begin{bmatrix} 0 \\ 0 \\ \|X_{\mathrm{b}}\| \end{bmatrix} = \mathbf{R}_{\mathrm{PT}}(\alpha, \beta) X_{\mathrm{b}},$$

y pasando $\mathbf{R}_{\mathrm{PT}}$ al lado izquierdo, al multiplicarse por un vector que tiene dos elementos cero se simplifica a

$$\begin{bmatrix} -\sin\alpha \sin\beta \\ \cos\alpha \sin\beta \\ \cos\beta \end{bmatrix} = X_{\mathrm{b}}/\|X_{\mathrm{b}}\|.$$

Como si se tratara de resolver una cinemática inversa en robótica, se puede despejar que

$$\alpha = \arctan2(x_{\mathrm{b}}, y_{\mathrm{b}}),$$

$$\text{y} \quad \beta = \arctan2(\sqrt{x_{\mathrm{b}}^2 + y_{\mathrm{b}}^2}, z_{\mathrm{b}}).$$

Entonces partiendo de un punto de observación X_{M} y que la base de la cámara tiene

pose conocida se calcula el punto de observación según la base (ecuación (3.2)) y con estas dos últimas expresiones se calcula el pan y tilt que apunta la cámara hacia ese punto.

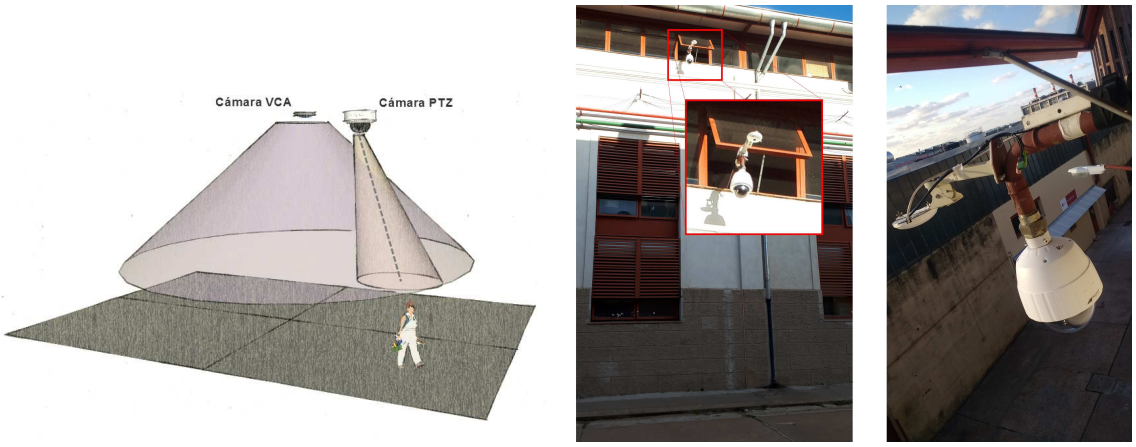
Capítulo 4

Aplicaciones de retro-proyección

En este capítulo se usa la retro-proyección para orientar o apuntar una cámara PTZ conformando un sistema que integra cámara *fisheye* con PTZ; se georreferencian vehículos obteniendo sus trazas en el plano mundo; y se calcula la probabilidad de que esté superando un umbral de velocidad. Estos casos de aplicación motivaron el trabajo de los capítulos subsiguientes sobre calibración con enfoque bayesiano.

4.1. Apunte de cámara PTZ

En [13] se combinaron dos cámaras: los objetos detectados en la imagen de una cámara *fisheye* se georreferencian mediante retro-proyección y se apunta una cámara PTZ hacia ellos para obtener una captura de mayor resolución. La *fisheye* tiene la ventaja de monitorear un área grande y por lo tanto puede ver en cualquier dirección, pero no con gran resolución; la PTZ puede hacer zoom en una zona más acotada y capturar imágenes de mayor resolución (ver figura 4.1a). El trabajo consistió primero en las calibraciones intrínsecas de las cámaras como se explicó en el capítulo 2. Para la *fisheye* se usó un modelo estereográfico para describir la severa distorsión y para la PTZ un modelo *pinhole* fue suficiente. Luego se las montó (figura 4.1b) en alguna locación desde donde se pudieran marcar puntos de calibración extrínseca en las imágenes tomadas por ambas cámaras (figuras 4.3a y 4.3b) para calibrar la pose de ambas respecto al marco de referencia mundo. Finalmente se eligió a mano en la imagen *fisheye* la posición (en píxeles) de un objeto de interés, se lo retro-proyectó



(a) (b)

Figura 4.1: El sistema *fisheye*+PTZ aprovecha que la *fisheye* observa permanentemente una región muy amplia y la PTZ puede apuntar y adquirir imágenes de gran resolución. Se montaron ambas cámaras en el mismo soporte a unos 10m de altura. (a) Esquema de funcionamiento. (b) Cámaras montadas.

al plano del mundo (usando intrínseca y extrínseca de la *fisheye*) y luego se apuntó la PTZ (usando su calibración extrínseca como se resolvió en la sección 3.3).

Se descubrió que la calibración extrínseca debe hacerse distinto según sea para después usar la proyección $\mathcal{F}$ o la retro-proyección $\mathcal{G}$. La calibración extrínseca de las cámaras se hizo primero tal como se explica en el capítulo 2 calculando la pose que minimiza el error de proyección (ecuación (2.11)). En la figura 4.2 se muestra un ensayo de laboratorio respecto a un tablero plano (porque es fácil detectar sus esquinas internas y usarlas como puntos de calibración). A partir de una imagen de donde se obtienen puntos de calibración (figura 4.2a) se calculan los parámetros de roto-traslación de la cámara *fisheye* con respecto al marco de referencia tablero. Con esos valores de roto-traslación se testea tanto la proyección de las coordenadas 3D del tablero hacia la imagen (figura 4.2b) como la retro-proyección de las coordenadas imagen de las esquinas internas hacia las coordenadas mundo en el plano $z_{\mathrm{M}} = 0$ (figura 4.2c). Aquí se descubrió algo importante: Que la proyección (figura 4.2b) es adecuada *pero la retro-proyección es deficiente* (figura 4.2c).

Para que esta retro-proyección funcione se agregó un segundo paso de optimización que minimiza el error cuadrático de retro-proyección,

$$\hat{\Theta} = \arg\min_{\Theta} \sum_i \left| \mathcal{G}(X_{\mathrm{I}i}, \hat{\Gamma}, \Theta) - X_{\mathrm{M}i} \right|^2 .$$

El $\hat{\Theta}$ así obtenido produce una retro-proyección adecuada, ver figura 4.2d.

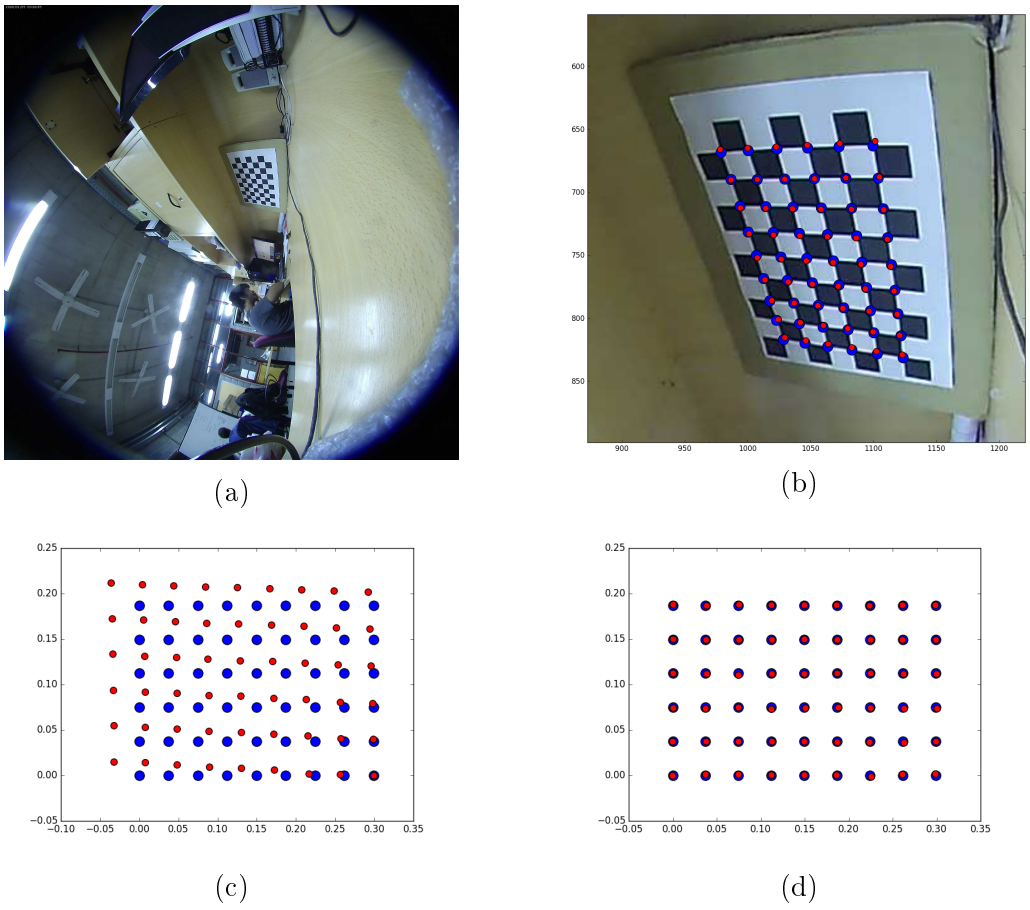
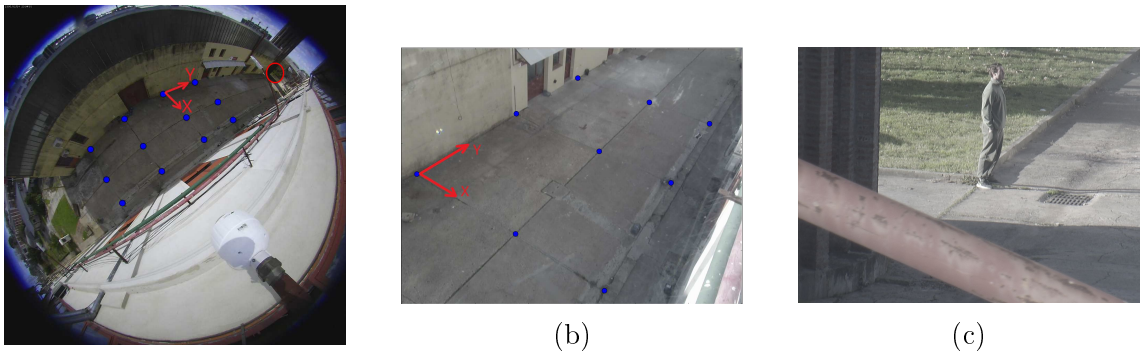


Figura 4.2: (a) Imagen para testear calibración extrínseca. (b) Proyección de puntos de calibración con parámetros que minimizan el error de $\mathcal{F}$. (c) Retro-Proyección deficiente. (d) Retro-Proyección usando la pose que minimiza el error de $\mathcal{G}$.

En las figuras 4.3a y 4.3b se muestra en azul los puntos de calibración usados para estimar la pose de las cámaras *fisheye* (optimizando la retro-proyección) y PTZ (la pose de la base) respectivamente. Para el experimento se escribió software que mostraba en pantalla las imágenes de ambas cámaras y permitía capturar clics sobre la imagen de *fisheye* y comandar los ángulos de pan y tilt de la PTZ. En la imagen *fisheye* se clicó sobre un objeto de interés, las coordenadas de ese clic se retro-proyectaron al plano mundo (como se explicó en la figura 3.2), se calcularon los ángulos de pan y tilt que apuntan la PTZ al objeto y el resultado del apunte se muestra en la figura 4.3c.

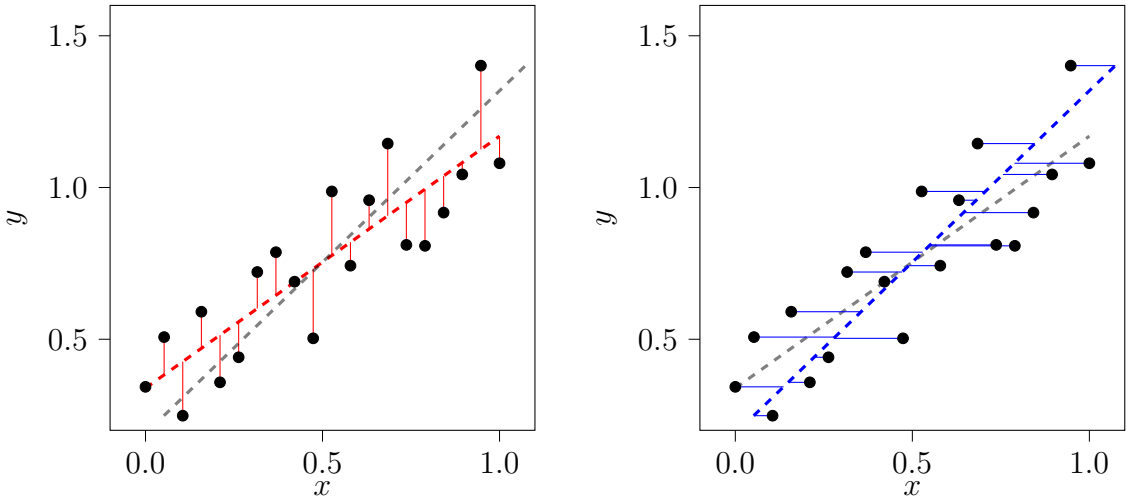
El sistema de apunte PTZ sirvió para validar el modelado geométrico del sistema de cámaras, los cambios de marcos de referencia y la retro-proyección. Se pudo hacer apuntar la PTZ exitosamente, demostrando que todas las componentes del sistema funcionan.



(a) (b) (c)

Figura 4.3: (a) Puntos de calibración extrínseca de *fisheye* y objeto de interés. (b) Puntos de calibración extrínseca de PTZ. (c) Apunte de PTZ al objeto de interés.

La calibración debe hacerse optimizando la proyección que quiera usarse después. Es un concepto general de ajuste por mínimos cuadrados. Para ilustrar el concepto, en la figura 4.4 se muestran pares ordenados de datos (x, y) que presumiblemente se modelan por $y(x) \equiv ax + b$. En la figura 4.4a se muestra el ajuste que minimiza el error cuadrático y en segmentos rojos los residuos del mismo. Alternativamente se ajusta los mismos datos pero suponiendo que el modelo subyacente es $x(y) \equiv ay + b$, y se muestra en azul en la figura 4.4b. Queda claro que según si se minimiza el error cuadrático en x ó en y se obtienen ajustes distintos.



(a) (b)

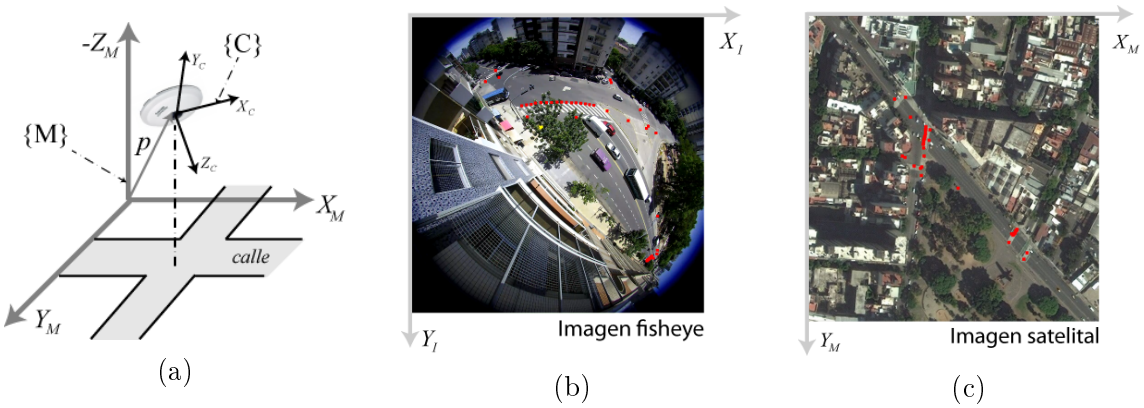
Figura 4.4: Los mismos datos (puntos negros) y dos regresiones lineales. (a) Ajuste y residuos de minimizar error cuadrático en y en rojo. (b) Idem minimizando error cuadrático en x en azul.

De la misma manera se puede hacer la calibración extrínseca optimizando el

error proyección (error cuadrático en la imagen) o el error de retro-proyección (error cuadrático sobre el plano mundo) obteniendo parámetros de pose diferentes. ¿Cuál de las dos usar? La que minimice el error del mapeo que se usará después: Si se quiere usar la calibración para hacer una retro-proyección desde la imagen al mundo entonces la calibración de pose debe hacerse minimizando el error en el mundo. Pero si se quiere hacer predicciones proyectando desde el mundo a la imagen entonces la calibración de pose debe hacerse minimizando el error en la imagen.

4.2. Georreferenciación de vehículos e infracción de velocidad

En [9, 12] se aplicó la retro-proyección para calcular la posición de vehículos y estimar su velocidad para determinar si estaban superando un límite de velocidad. Se instaló la cámara *fisheye* tal que observara un cruce de calles y se realizó la calibración extrínseca. Con métodos estándar de procesamiento de imágenes se segmentaron las regiones de la imagen asociadas a vehículos en movimiento y se localizaron puntos característicos para calcular el desplazamiento de los vehículos entre dos fotogramas contiguos del video. Se retro-proyectaron esos puntos para calcular la velocidad del vehículo y la probabilidad de que éste haya incurrido en una infracción de velocidad.



(a) (b) (c)

Figura 4.5: Experimento de monitoreo de tránsito con cámara *fisheye*, ubicada donde se ve una intersección, se identifican 42 puntos de calibración para estimar la pose. (a) Posicionamiento de cámara. (b) Imagen *fisheye* con puntos de calibración marcados. (c) Imagen satelital indicando los puntos de calibración en el mundo.

La figura 4.5a esquematiza la instalación de la cámara sobre el plano de la calle. Se identifican manualmente 42 puntos de calibración como se muestra en las figuras

4.5b y 4.5c. Las coordenadas mundo se definieron en unidades físicas (metros) pero se trabajó también en latitud-longitud (grados) por la necesidad de interactuar con un servicio de mapa; y también en imagen satelital (píxeles) para poder mostrar las coordenadas mundo en las figuras. La conversión entre estos sistemas se hizo simplemente con un reescaleo usando un factor *ad hoc*.

4.2.1. Detección de infracción de velocidad

Mediante visión no se miden de manera directa las velocidades de los objetos sino sus posiciones. A partir de ellas se puede usar cualquier método de derivada numérica para calcular la velocidad. En esta subsección se describe primero cómo se detectaron automáticamente las posiciones de los vehículos usando técnicas estándar de detección de movimiento y puntos de interés. Se retro-proyectaron esas posiciones a coordenadas mundo. Para calcular la derivada de las posiciones en función del tiempo se ajustaron polinomios a las trayectorias obtenidas y se calculó la derivada de los polinomios. Sobre las velocidades obtenidas se aplica un proceso de toma de decisión.

La sustracción de fondo es un método ampliamente utilizado para la detección de objetos en movimiento[55]. Este enfoque se basa en la diferencia entre el cuadro actual y una imagen de referencia (llamada 'imagen de fondo') que debe actualizarse a las condiciones de luminosidad variables en el tiempo. En este trabajo se utiliza una variante de métodos estándar de segmentación *frente-fondo*[39], que determina qué píxeles en el fotograma están asociados a los objetos en movimiento. Es importante destacar que una ventaja de este clasificador es la velocidad de procesamiento y el bajo requisito de memoria para cada píxel. El resultado son regiones asociados a objetos en movimiento, *blobs*. En la figura 4.6A se muestra un fotograma de ejemplo con distintos vehículos en movimiento. En las figuras 4.6Bi y 4.6Bii se amplían algunos de los blobs detectados. Si la cantidad de vehículos en la escena fuese baja se podría calcular la velocidad de los vehículos realizando un seguimiento de los centros de cada blob. Lamentablemente esta solución no es robusta, la visión de los vehículos suele ocluirse por otros vehículos, se ven como blobs 'pegados' (figura 4.6Biii), y la determinación del centro tiene discontinuidades importantes. Esto lleva

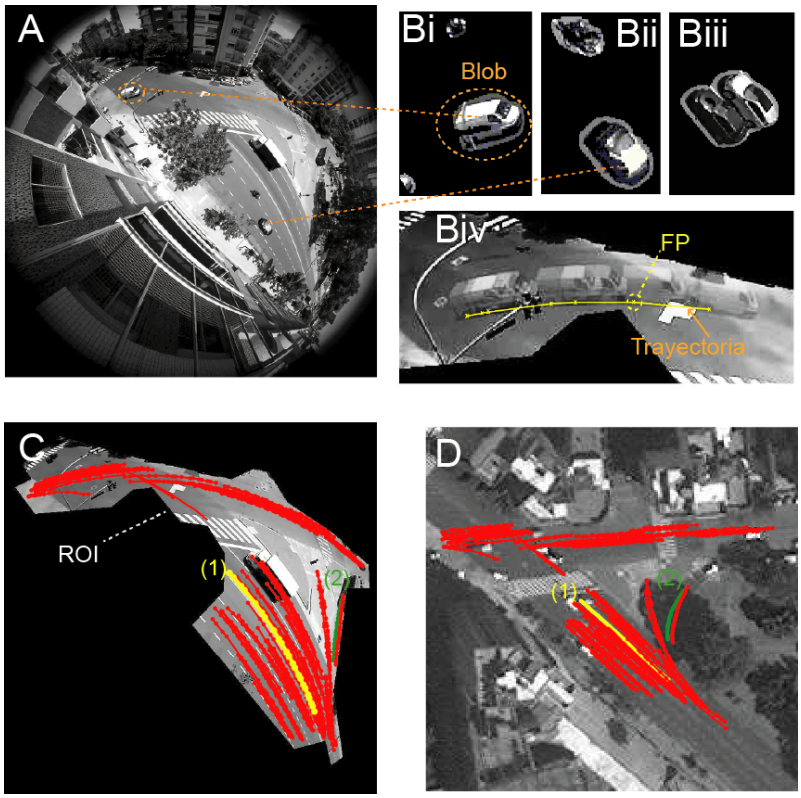


Figura 4.6: (A) Un fotograma completo. (Bi-iii) Acercamientos a los blobs detectados por segmentación frente-fondo. (Biv) Trayectoria de un punto de interés asociado a un vehículo en movimiento. (C) Trazas de puntos de interés dentro de la máscara de la región de interés. (D) Retro-proyecciones de las trazas sobre el plano mundo.

a proponer un método basado en el seguimiento de puntos de interés.

Una técnica muy útil en procesamiento de videos es la detección y seguimiento de puntos de interés. Se trata de coordenadas en la imagen cuyo entorno tiene una estructura visual única, fácil de seguir y comparar [34]. En general son malos puntos de interés las regiones de color muy homogéneo. Son buenos puntos de interés las esquinas afiladas o los puntos brillantes porque si se mueve un poco la cámara y se captura una imagen nueva probablemente se vean muy parecidos. El procedimiento general es usar un algoritmo que los detecte, es decir definir dónde están en la imagen. Luego un segundo algoritmo que calcule descriptores que cuantifican la estructura visual de la región circundante. Un tercer algoritmo usa los descriptores para comparar puntos de interés entre sí aplicando una métrica de similaridad. La comparación se resume en decir si son lo suficientemente parecidos como para considerar que son el mismo punto de interés pero movido.

En [9] se utilizan las técnicas SURF [56] y FLANN [57] implementadas en

OpenCV [34]. Se aplica SURF para detectar puntos de interés y calcular sus descriptores. Gracias a la robustez de los descriptores SURF ante cambios de escala, rotación e iluminación se puede encontrar esos mismos puntos en el frame siguiente. En [12] se realiza la detección de los puntos de interés, seleccionando solo los que se hallen dentro de blobs. Se calculan los descriptores del punto de interés y se resuelve la correspondencia entre conjuntos de puntos de frames sucesivos con FLANN. En la figura 4.6Biv se muestra un ejemplo del seguimiento de un punto de interés. A medida que se estiman las trayectorias en la imagen, se calcula la trayectoria en el mapa con la fórmula de retro-proyección. En la figura 4.6 paneles C y D se muestran ejemplos de trayectorias detectadas y su correspondiente georreferenciación.

Para estimar la velocidad y la probabilidad de infracción se propone que las trayectorias de los vehículos en el mapa sigan un modelo polinómico. A continuación se explica el ajuste del modelo y el cálculo de la incerteza para la componente x_{M} de movimiento (para y_{M} es idéntico). Se parte de la suposición de que la trayectoria en mapa $x_{\mathrm{M}}(t)$ de un punto de interés puede ajustarse por un polinomio de grado n_p:

$$x_{\mathrm{M}}(t) = \epsilon_t + \sum_{n=0}^{n_p} t^n b_n \tag{4.1}$$

donde ϵ_t corresponde a un ruido gaussiano de media nula. La ecuación (4.1) expresada para cada medición a tiempo t es un conjunto de n_t ecuaciones que pueden escribirse vectorialmente como $\overline{x} = A\,\overline{b} + \overline{\epsilon}$, donde $\overline{x}$ es el vector de los n_t datos de la trayectoria y $\overline{b}$ el vector de coeficientes del polinomio. Los valores de tiempo elevados a las n_p potencias componen la matriz de Vandermonde [58] $A^{n_t \times n_p}$.

Con estas definiciones, a partir de las trayectorias retro-proyectadas como las de la figura 4.6Biv, con métodos estándar de ajuste de polinomios [59], se estiman los coeficientes del polinomio y de sus covarianzas. El ajuste retorna la estimación de los coeficientes $\widehat{b}_n$, la matriz de covarianzas $\widehat{\mathbf{C}}_b$:

$$\mathrm{E}[\overline{b}] \;=\; \widehat{\overline{b}} = (A^T A)^{-1} A^T \overline{x}, \tag{4.2}$$

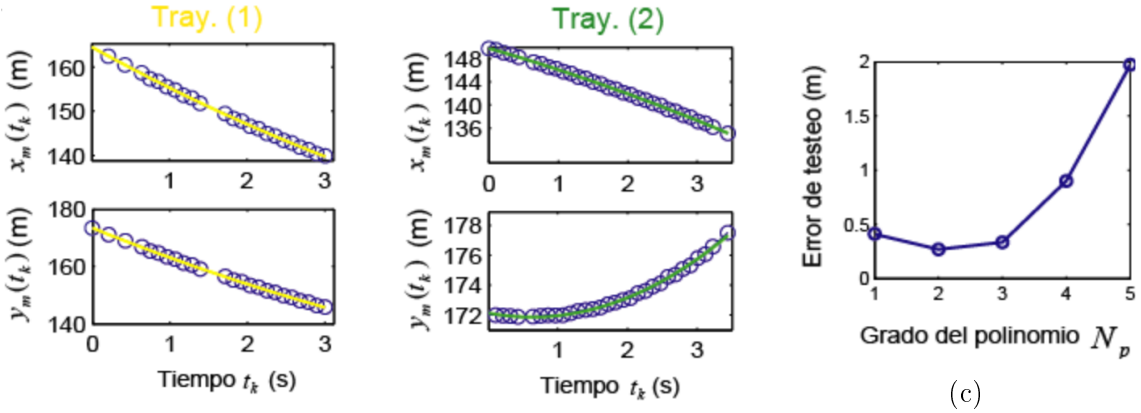
$$\text{y Cov}\left[\widehat{\overline{b}}\right] \;=\; \hat{\sigma}_\epsilon^2 (A^T A)^{-1} = \hat{\sigma}_\epsilon^2 \mathcal{P}. \tag{4.3}$$

La intensidad del ruido gaussiano se estima como la desviación estándar del error

en la varianza $\hat{\sigma}_\epsilon^2$:

$$\hat{\sigma}_\epsilon^2 = \frac{\left|\overline{x} - A\,\widehat{\overline{b}}\right|^2}{n_t - n_p}. \tag{4.4}$$

Esta estimación se denomina *offline* porque usa todos los datos de la trayectoria al finalizar la medición. En las figuras 4.7a y 4.7b se muestran dos ejemplos de ajustes (con $n_p = 2$) para las trayectorias indicadas con color amarillo y verde en la figura 4.6 paneles C y D.



(a) (b) (c)

Figura 4.7: En (a) y (b) dos trayectorias típicas ajustadas. (c) El error de testeo es mínimo cuando el polinomio es de grado dos.

¿Qué grado de polinomio usar? Se utilizó un esquema de validación cruzada (*Leave-One-Out Cross Validation*) [60] para 250 trayectorias y se encontró que el error de prueba es menor para el caso de polinomios de grado 2. Ver figura 4.7c.

Pensando en la aplicación en tiempo real, hay que estimar los coeficientes del polinomio en simultáneo al seguimiento de los puntos de interés, esta estimación *online* debe ser es causal y recursiva. Causal porque en cada medición no tiene sentido suponer que se tienen datos de tiempo posterior a la misma. Recursiva para que la estimación sea numéricamente eficiente, al no tener que re calcular usando todos los datos anteriores sino las estimaciones de tiempo anterior y el dato nuevo. Se aplica el algoritmo de cuadrado mínimos recursivos (RLS) [59].

Denominando $\overline{a}_t^T$ a las filas de la matriz de Vandermonde A se puede expresar la ecuación 4.1 como $x_{\mathrm{M}}(t) = \overline{a}_t^T \overline{b}_t + \epsilon_t$. Para la aplicación real, a tiempo t no se tiene la matriz A completa, y se recurre a un estimador recursivo de cuadrados mínimos. El estimador calcula loa parámetros $\widehat{\overline{b}}_t$ y su covarianza $\mathcal{P}_t$ en función del tiempo actual

(dado por $\overline{a}_t^T$), el valor actual de posición $x(t)$ y la estimación de tiempo anterior. El cálculo es

$$\mathcal{P}_t = \mathcal{P}_{t-1} - \frac{\mathcal{P}_{t-1}\ \overline{a}_t \overline{a}_t^T\ \mathcal{P}_{t-1}}{1 + \overline{a}_t^T\ \mathcal{P}_{t-1}\ \overline{a}_t}, \tag{4.5}$$

$$\widehat{\overline{b}}_t = \widehat{\overline{b}}_{t-1} + \mathcal{P}_t\ \overline{a}_t\ (x(t) - \overline{a}_t^T\ \widehat{\overline{b}}_{t-1}), \tag{4.6}$$

donde se cumple que al terminar de recorrer todos los puntos se obtiene lo mismo que en la estimación *offline*, $\widehat{\overline{b}}_{n_t} = \widehat{\overline{b}}$ y $\mathcal{P} = \mathcal{P}_{n_t}$. Para inicializar el algoritmo se necesita una propuesta inicial para $\widehat{\overline{b}}_1$ y $\mathcal{P}_1$ que fue elegida por un ajuste según el método *offline* aplicado a las primeras 4 mediciones, utilizando las ecuaciones (4.2) y (4.3).

La covarianza t-ésima es $\mathcal{P}_k$ multiplicada por una estimación de la desviación estándar del error. Simplemente se aplica la ecuación (4.4) en cada iteración con todos los datos hasta el momento t en lugar de usar todos y se calcula $\hat{\sigma}^2_{\epsilon,t}$.

Ambas estimaciones, online y offline, retornan la esperanza y varianza de la distribución de probabilidad de los parámetros del ajuste dados los datos de posición. Si se descarta que los momentos de orden superior sean significativos se puede decir que la distribución de probabilidad de los parámetros dados los datos es una distribución normal.

La componente x de la velocidad se calcula como la derivada del polinomio ajustado. El valor esperado de la velocidad a tiempo t es

$$\widehat{v}_x(t) = \sum_{n=0}^{N_p} n\ t^{n-1}\ \widehat{b}_n$$

La varianza de la velocidad se calcula usando la propiedad de combinación lineal de variables aleatorias con densidad de probabilidad gaussianas [61], en símbolos

$$\mathrm{Var}[v_x] = \hat{\sigma}^2_{vx} = [0, 1, 2t]\ \widehat{\mathbf{C}}_b \begin{bmatrix} 0 \\ 1 \\ 2t \end{bmatrix},$$

y se obtiene la esperanza y varianza de la componente x de la velocidad como función del tiempo

$$\widehat{v}_x(t) = \widehat{b}_1 + 2\ t\ \widehat{b}_2, \tag{4.7}$$

$$\hat{\sigma}^2_{vx}(t) = \widehat{\mathbf{C}}_{b22} + 4\ t\ \widehat{\mathbf{C}}_{b23} + 4\ t^2\ \widehat{\mathbf{C}}_{b33}. \tag{4.8}$$

Donde $\widehat{\mathbf{C}}_{bij}$ es el elemento ij de $\widehat{\mathbf{C}}_b$.

Es decir que con una secuencia de posiciones retro-proyectadas $(x_{\mathrm{M}}(1), ..., x_{\mathrm{M}}(t))$ se puede ajustar un polinomio y con sus parámetros se puede calcular la esperanza y varianza de la velocidad, $(\widehat{v}_x(t), \hat{\sigma}^2_{vx}(t))$. Es razonable suponer que la distribución de probabilidad es normal, en símbolos se denota

$$v_x(t) \mid x_{\mathrm{M}}(1), ..., x_{\mathrm{M}}(t) \ \sim \ \mathcal{N}(v_x \mid \widehat{v}_x(t), \hat{\sigma}^2_{vx}(t)).$$

Idem para la componente y. Y la distribución de probabilidad del vector velocidad del vehículo será el producto de las gaussianas para las componentes x e y.

En resumen, se presentan dos esquemas para la estimación de la velocidad del vehículo. El método *offline* estima los parámetros $\overline{b}$ usando la información de toda la trayectoria y utiliza las ecuaciones (4.2), (4.3). El método *online* actualiza en tiempo de ejecución la estimación para cada medición utilizando las ecuaciones (4.5), (4.6). Luego de esto, se estima la velocidad y su incerteza utilizando las ecuaciones (4.7), (4.8) obteniendo a tiempo t las estimaciones $(\widehat{v}_x, \widehat{v}_y, \hat{\sigma}^2_{v_x}, \hat{\sigma}^2_{v_y})$.

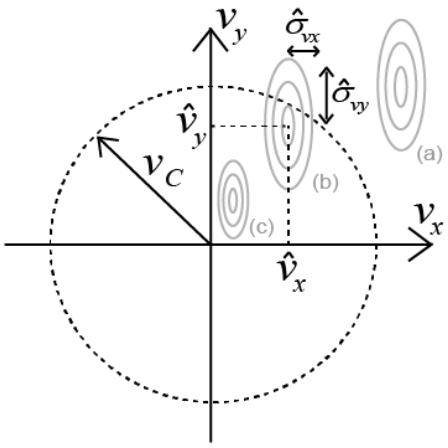
A continuación se calcula la probabilidad de que el módulo de la velocidad $v = \sqrt{(v_x^2 + v_y^2)}$ sea mayor a un valor crítico denominado v_c. La probabilidad de superar v_c, de cometer una infracción, es $\rho = \mathrm{P}(v > v_c \,|\, \widehat{v}_x, \widehat{v}_y, \hat{\sigma}_{v_x}, \hat{\sigma}_{v_y})$. El problema se esquematiza en la figura 4.8a, donde la distribución de probabilidad del vector velocidad tiene una densidad de probabilidad gaussiana bivaluada no correlacionada y centrada en $(\widehat{v}_x, \widehat{v}_y)$ con anchos característicos $\hat{\sigma}_{v_x}$ y $\hat{\sigma}_{v_y}$. La probabilidad ρ de que se supere la velocidad crítica es la integral de la gaussiana bivaluada fuera del círculo.

En la implementación numérica se calcula ρ como

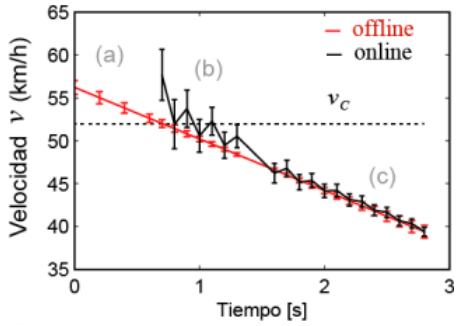
$$\rho = 1 - p(v \leq v_c \,|\, \widehat{v}_x, \widehat{v}_y, \hat{\sigma}_{v_x}, \hat{\sigma}_{v_y}).$$

La probabilidad de $v \leq v_c$ es la integral dentro del círculo. El cálculo numérico se plantea así porque es más sencillo integrar en un dominio finito. Dado que la distribución gaussiana bivaluada de la velocidad no tiene componentes cruzadas es fácil separar la densidad de probabilidad en una componente en v_x y otra en v_y que deben multiplicarse,

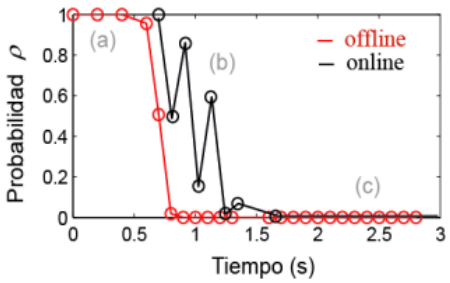
$$\rho = 1 - \int_{-v_c}^{v_c} \mathcal{N}(v_x \mid \widehat{v}_x, \hat{\sigma}^2_{vx}) \int_{-\sqrt{v_c^2 - v_x^2}}^{\sqrt{v_c^2 - v_x^2}} \mathcal{N}(v_y \mid \widehat{v}_y, \hat{\sigma}^2_{vy}).$$



(a)



(b)



(c)

Figura 4.8: (a) La distribución de probabilidad (indicada con curvas de nivel) de $(v_{xM}(t), v_{yM}(t))$ en tres situaciones características: (a) superando v_c, (b) cerca del límite de velocidad y (c) totalmente dentro del limite permitido de velocidad. (b) Módulo de la velocidad de un vehículo en función del tiempo, la línea punteada indica el valor critico de velocidad. (c) Probabilidad ρ de superar v_c. Es uno en caso (a), cero en (c) y (b) es una transición entre ellos.

La integral en una de las componentes puede expresarse en términos de la función error erf llegando a la expresión

$$\rho = 1 - \int_{-v_c}^{v_c} \frac{1}{2}\left[\text{erf}\left(\frac{\sqrt{v_c^2 - v_x^2} - \widehat{v}_y}{\hat{\sigma}_{v_y}\sqrt{2}}\right) - \text{erf}\left(\frac{-\sqrt{v_c^2 - v_x^2} - \widehat{v}_y}{\hat{\sigma}_{v_y}\sqrt{2}}\right)\right] \mathcal{N}(v_x \mid \widehat{v}_x, \hat{\sigma}_{vx}^2).$$

Esta integral en el disco de radio v_c se evaluó numéricamente la integral de Riemann usando 10^3 intervalos de integración.

En la figura 4.8b se muestra la estimación de la velocidad utilizando el método offline (trazo rojo) y el online (trazo negro); la línea punteada horizontal indica el valor elegido de v_c para ejemplificar un caso de infracción. La estimación offline es una recta por ser la derivada del polinomio de grado 2 ajustado con todos los datos, las barras de error son menores cerca del centro del rango donde se ajustó, dado que es donde hay menos margen de incertidumbre. Por otro lado se tiene el ajuste

online con RLS, que posee barras de error grandes al principio y tienden a decrecer a medida que el algoritmo ajusta más datos. Hacia al final del rango de ajuste, RLS converge a hacia el ajuste offline.

Las situaciones (a)-(c), corresponden a tiempos en los que el vehículo pasa de una velocidad mayor al valor crítico seleccionado, hasta un valor menor por debajo de v_c. La figura 4.8c muestra la probabilidad ρ en función del tiempo.

La toma de decisión es el ultimo paso. Una manera simple de hacerlo es poner un umbral a la probabilidad y decir que se cometió infracción si la probabilidad superó ese umbral. Por ejemplo, si para decidir que hubo infracción la probabilidad debe superar el 90 % entonces las primeros cuatro detecciones se consideran en infracción, ver figura 4.8c. Si el umbral es de 80 % entonces según el método offline (línea roja) los primeras cuatro detecciones están en infracción. Pero según el método online (línea negra) están en infracción las primeras cinco detecciones, la sexta no, y la séptima vuelve a estar en infracción. El problema de definir un umbral (o una estrategia general de toma de decisión a partir de la probabilidad) queda fuera de los alcances de esta tesis. La decisión de si un vehículo está o no cometiendo una infracción puede estar equivocada, al fin y al cabo todo sensor es imperfecto y hay una incerteza asociada al cálculo de velocidad. Si se detecta infracción pero el vehículo en realidad no está cometiendo una infracción se tiene un falso positivo. Si, al revés, un vehículo está superando la velocidad máxima pero la probabilidad detectada no supera el umbral establecido, no se lo detecta como infracción y se tiene un falso negativo. Los casos donde la decisión de si hubo infracción es correcta son los verdaderos positivos y verdaderos negativos. Para aplicar teoría de toma de decisión requiere que se defina un función de utilidad sobre verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos [62, 43]. Se debería definir el umbral usando el principio de maximizar la utilidad esperada, la clave es la función de utilidad. Pero es una tarea que excede a esta tesis porque no corresponde específicamente al uso de cámaras como sensor del estado del tránsito. Corresponde más a estudiar los riesgos y costos de los accidentes de tránsito en general.

(a) (b)

Figura 4.9: (a) Generación de *bounding boxes* de Yolo [63]. (b) Algunas imágenes de COCO donde hay vehículos.

4.3. Cuantificación de la incerteza de detección de objetos en la imagen

En las secciones anteriores se formuló un esquema para calcular la probabilidad de infracción a partir de la incerteza en las coordenadas georreferenciadas. Falta una componente clave, las incertezas de las coordenadas imagen, es decir la incerteza de la medición directa de X_{I}. En [15] se ejecuta una red neuronal pre-entrenada sobre una base de datos de imágenes de tránsito para detectar vehículos y se mide su error de detección. El objetivo de esta sección es evaluar la incerteza con que una red neuronal localiza vehículos en una imagen.

Se usó la red neuronal convolutiva Yolo versión 3 (Yolo-v3) [64] que se caracteriza por su bajo costo computacional, entrenada con el dataset COCO (Common Objects in Context) [65] que consiste en imágenes naturales de 91 clases de objetos comunes clasificados y localizados. Este dataset no es específico de monitoreo de tránsito si no que consiste en escenas cotidianas, por lo tanto el punto de vista es en general parecido al que tiene una persona, y pocas veces incluye imágenes con una perspectiva parecida a la de una cámara de monitoreo (ver figura 4.9b). Yolo-v3 recibe una imagen y retorna el *bounding box* (ver figura 4.9a) e indica a cuál de las 91 clases posibles pertenece el objeto. Cinco de ellas son vehículos, y como [15] es un trabajo orientado a tránsito las demás 86 clases son ignoradas. El *bounding box* es el menor rectángulo de lados verticales y horizontales que encierra un objeto. Se

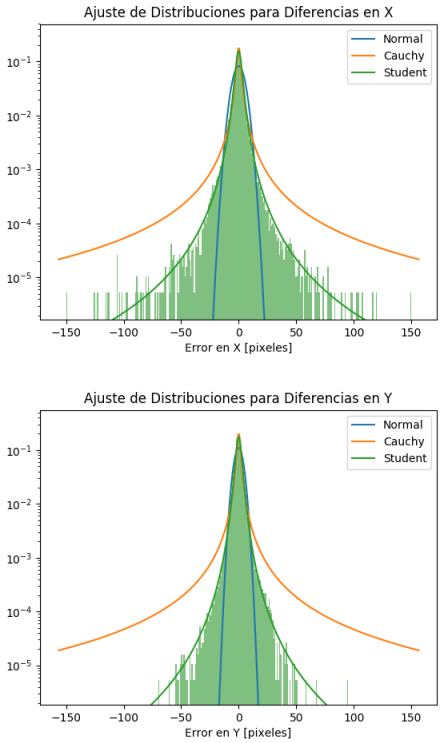
lo parametriza con cuatro valores, por ejemplo con las coordenadas 2D de uno de sus vertices y su ancho y alto (con las coordenadas 2D de dos vertices opuestos). Los creadores de COCO cuidaron que la clasificacion y localización de los objetos sea correcta, por eso la clase y *bounding box* provistas en COCO se consideran el *ground truth*. Es decir, son los valores reales que nuestro modelo debería tratar de predecir.

Se evaluó el desempeño de Yolo-v3 entrenada con COCO sobre un dataset específico a monitoreo de tránsito. MIO-TCD [66] consiste en 110.000 imágenes adquiridas en diferentes horas del día y periodos del año por cámaras de tráfico ubicadas en Estados Unidos y Canadá. Estas imágenes cubren una amplia gama de escenarios de tráfico vehicular, como también diferentes perspectivas y poses de las cámaras. En general se trata de cámaras con poca distorsión óptica. MIO-TCD tiene definidas 11 clases de objetos, todos vehículos, y provee también sus *bounding boxes*.

En esta tesis, localización se refiere a coordenadas puntuales, por eso de los *bounding boxes* se toma su centro geométrico como su localización. En la figura 4.10a se muestra un ejemplo de los *bounding boxes* del dataset (considerados como el *ground truth*) y los detectados por Yolo-v3. El error de detección de la red neuronal es la distancia entre los centros de los rectángulos *ground truth* y el predicho por la red. En la figura 4.10b se muestran los histogramas de las componentes u, v de los errores de detección de 188 346 vehículos. La media cuadrática es de 4.800px para u y 3.628px en v. Pero además puede verse que la distribución, si bien unimodal, está lejos de ser gaussiana, se describe más adecuadamente como T de Student.

La distribución T de Student se ajusta bien y se podría haber usado sus parámetros para cuantificar la incerteza de detección. Pero no está bien pensar que esa parametrización sea generalizable porque la forma de la distribución del error obedece a detalles específicos de este caso. La perspectiva desde donde se ven los vehículos en COCO es más a nivel del suelo mientras que MIO-TCD son mas bien desde altura. La red no fue entrenada para minimizar el error de localización. COCO y MIO-TCD no usaron las mismas definiciones de las clases de vehículos (por ejemplo MIO-TCD define *truck* incluyendo el acoplado de camión en el *bounding box*, pero COCO no incluye el acoplado). Por todos estos factores es que se considera que

(a)

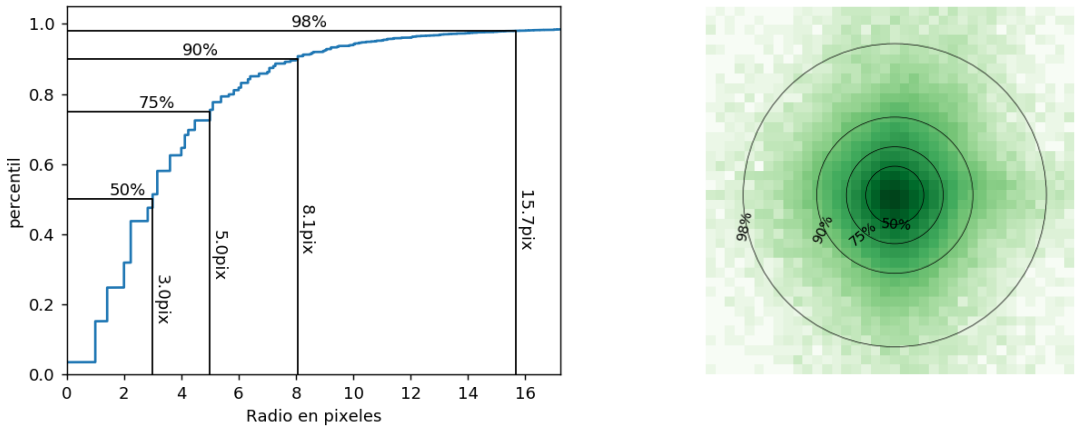


(b)

Figura 4.10: (a) Bounding boxes del dataset y detectados por Yolo-v3 en una imagen de MIO-TCD. (b) Histogramas de los errores de detección de Yolo-v3 respecto al *ground truth* de MIO-TCD.

la distribución de los errores de detección no se debe considerar en general como una T de Student. Si la red se entrenara para minimizar el error de localización entonces la distribución tendría seguramente forma gaussiana.

Una forma más general de cuantificar la dispersión del error es usando intervalos de confianza. En el espacio del error de detección se pueden construir regiones de confianza circulares centradas en el origen, ver la figura 4.11b donde se muestra el histograma 2D de los errores de detección. Dado un radio se calcula la fracción de casos que caen dentro del círculo. Al considerar radios mayores crece la fracción de casos dentro. En la figura 4.11a se muestra la curva del porcentaje de datos incluidos en el círculo con respecto a su radio. Un 98 % de los casos tienen un error de menos de 15.7 píxeles. La región de confianza del 90 % tiene un radio de 8.1 píxeles. Al construir estas regiones de confianza se puede retornar una cuantificación de la incerteza de detección que no necesita suponer una forma específica de la distribución.



(a) (b)

Figura 4.11: (a) Percentil de datos abarcado por la región de confianza en función del radio de la región. (b) Histograma bidimensional del error de detección y regiones de confianza.

Pero MIO-TCD es un dataset de imágenes que no tienen la distorsión de una cámara ojo de pez. Entonces se midió también el rendimiento de la red en un dataset propio de capturas de *fisheye*. Este dataset se construyó a partir de los videos de tránsito capturados en Parque Lezama, marcando las localizaciones y etiquetando manualmente 436 vehículos. En la figura 4.12 se muestra un ejemplo de las detecciones. Se cuantificó el error de localización de la misma manera que para MIO-TCD y se obtuvo que el 90 % de las localizaciones tienen un error menor a 13.0 píxeles. Un resultado adicional es que cerca de la mitad de los vehículos no fueron detectados.

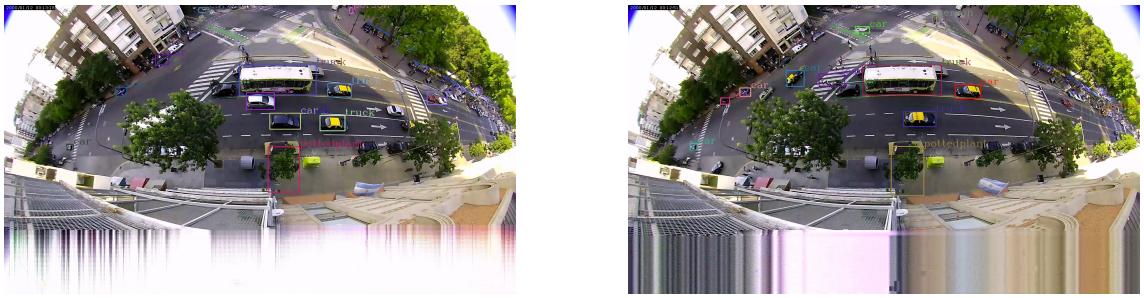


Figura 4.12: Detecciones de Yolo-v3 en imágenes *fisheye*.

Es esperable que si la red tuviera una arquitectura diseñada para localización y se la entrenara con un dataset específico de monitoreo de transporte, entonces el error de localización seguramente sería mucho menor.

4.4. Resumen de aplicaciones de retro-proyección

En el capítulo anterior se resolvió cómo hacer la retro-proyección y en este capítulo se mostraron aplicaciones de esa solución. Se planteó un sistema combinado de cámara ojo de pez y cámara PTZ que permitiría combinar las ventajas de ambas: una cámara observa toda la escena en simultáneo constantemente y la otra puede obtener capturas de alta resolución de una parte específica de la escena. Se combinaron las herramientas de retro-proyección, calibración y cinemática inversa del capítulo anterior para que el cálculo de posición de los motores que apuntan la PTZ se haga automáticamente. Para ambas cámaras se hace la calibración intrínseca con el método estándar. Luego se las instala *in situ* y se marcan puntos de calibración extrínseca en el suelo. La cámara ojo de pez debe calibrarse minimizando el error de retro-proyección. Este fue un descubrimiento importante: los parámetros de pose deben calibrarse según luego se quiera usar el sistema para proyectar o retro-proyectar. La calibración extrínseca de la PTZ se hace conociendo los ángulos de motor en que se obtiene la imagen de calibración y minimizando el error de proyección (error en la imagen). Con el sistema calibrado se marca en la imagen de la cámara omnidireccional algún punto de interés, y el sistema lo retro-proyecta al plano mundo para saber en qué ángulos de pan y tilt debe posicionar a la PTZ para ver al objeto en el centro de la imagen.

Hasta aquí se han validado los modelos geométricos de cambio de marcos de referencia, de proyección de perspectiva y de distorsión óptica necesarios para hacer la retro-proyección. Luego se encaró el problema de estimar la incerteza de las velocidades de los vehículos porque es necesario para los sistemas de toma de decisión. Para eso se consideró resuelto el problema de calibración y retro-proyección y se mostró que se puede hacer un seguimiento de puntos de interés de objetos en movimiento. Las trazas de los vehículos retro-proyectadas se pueden modelar como parábolas cuyos parámetros son estimados con su respectiva incerteza (matrices de covarianza). Las velocidades se calculan derivando las parábolas y la velocidad resultante tiene también una covarianza que se calcula propagando las covarianzas de los parámetros estimados de la parábola. Con una estimación de la velocidad y su incerteza se puede calcular *la probabilidad de que un vehículo supere el límite de*

velocidad, imprescindible para un sistema de toma de decisión de infracciones de tránsito.

Luego se midió que la red neuronal Yolo entrenada en el dataset COCO puede localizar con 90 % de confianza vehículos en la imagen (dataset de tránsito MIO-TCD) dentro de un radio de unos 8 píxeles. Yolo fue diseñada para retornar *bounding boxes* minimizando el error cuadrático de los cuatro parámetros necesarios para definirlo. No fue diseñada para retornar una localización puntual del objeto. Esto explica en primer lugar porqué el error de localización (tomando como localización al centro geométrico del *bounding box*) es tan grande. En comparación, un tracking de puntos de interés (SURF + FLANN) puede fácilmente tener precisión sub-píxel. También, la localización lograda por un operario humano al trackear un punto de referencia de un vehículo haciendo clic en cada fotograma del video es repetible con un error que ronda 1 píxel. Si la red neuronal fuera diseñada específicamente para localización, la incerteza de localización sería mucho menor. No interesa en esta tesis abrir la discusión de *cómo definir* la localización y mucho menos proponer mejoras a Yolo para que sirva para localización. El seguimiento de puntos de interés SURF y el error de localización por un operario llevan a pensar que seguramente una red neuronal diseñada y entrenada apropiadamente tendría una incerteza de localización del orden de 1 píxel. Finalmente, nótese que en el dataset MIO-TCD el error es de 8 píxeles y en el dataset de video *fisheye* es de 13 píxeles: el dataset debe consistir en imágenes que se asemejen a las imágenes que se deberá procesar en la aplicación final. Si se hubiera entrenado la red con imágenes *fisheye* el error sería menor.

4.5. Se necesita un andamiaje general de manejo de incertezas

Las soluciones estándar de calibración y proyección de perspectiva hacen un tratamiento incompleto de la incerteza. Se mostró que los métodos de estimación puntual funcionan, pero hay que extenderlos para estimar las incertezas. En esta etapa quedó claro que se necesita dar un orden y sistematizar el tratamiento de las incertezas de todas las variables involucradas; puntos de calibración y parámetros intrínsecos, extrínsecos. Y proponer una manera de usarlas para retro-proyectar ha-

cia el plano mundo la detección con incerteza de un vehículo en la imagen. Está claro que un error en la detección en la imagen se magnifica por efecto de la perspectiva y que la retro-proyección tiene de alguna manera que trasladar la incerteza desde la imagen al plano del mundo. La calibración debe optimizar un funcional que tome en cuenta la manera en que se va a hacer la retro-proyección después. Y si ésta se hace propagando incertezas entonces la calibración debe optimizar un funcional que incluya la retro-proyección de incertezas.

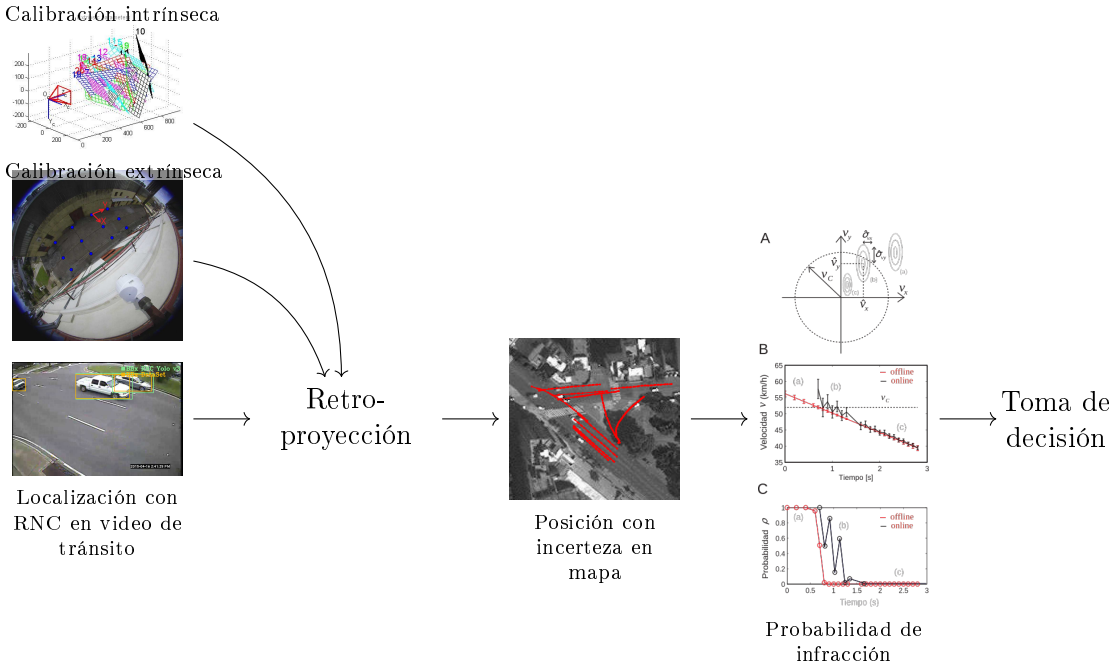


Figura 4.13: Conceptualmente, se combina la información de parámetros intrínsecos, extrínsecos y la localización en imagen (todos con su incerteza) a través de la función de retro-proyección que retorna la posición en el mapa (con su incerteza). Se calcula la probabilidad de infracción y con ella se toma la decisión de si hubo o no infracción.

La figura 4.13 conceptualiza la estructura que se entiende ahora que debe tener el sistema. La calibración intrínseca se hace en condiciones controladas y su resultado es una estimación de los parámetros que describen la distorsión de la óptica y la resolución del sensor CCD de la cámara, pero acompañados de alguna manera de una cuantificación de incerteza. Una vez instalada la cámara en el lugar que se quiere monitorear, se estiman los parámetros extrínsecos también cuantificando su incerteza. Usando algún método de detección y localización de vehículos en imágenes se obtienen coordenadas en píxeles, que ya se mostró tienen también asociada una incerteza (para el caso de Yolo-v3 de alrededor de 10 píxeles). La información de parámetros de calibración y posición en la imagen se combina a través del mo-

delo de retro-proyección y el resultado es información de la posición del vehículo en coordenadas mundo. Según qué tipo de infracción se quiera detectar, se puede calcular la probabilidad de que el vehículo esté incurriendo en una maniobra ilegal y finalmente se toma la decisión de si hubo o no hubo infracción.

Para acomodar todos estos requerimientos se incursionó en la inferencia bayesiana, que tiene la ventaja de formular todo en términos probabilísticos con suficiente generalidad. En el capitulo que sigue se muestra la solución encontrada que se formuló con un enfoque bayesiano desde primeros principios.

Capítulo 5

Formulación bayesiana

En este capítulo se formula la calibración y predicción desde un enfoque bayesiano. Se comienza con un repaso de inferencia bayesiana en general y se discute si sirve un esquema de regresión bayesiana estándar. En la sección 5.2 se parte de la expresión general de la distribución predictiva y se deduce la calibración en dos pasos: calibración intrínseca y extrínseca. El resultado de la calibración es una aproximación de la distribución a posteriori de los parámetros dados los datos. Con esos posteriors estimados, la distribución predictiva se aproxima con una propagación lineal de incertezas.

5.1. Inferencia bayesiana

Todo tratamiento bayesiano se hace a través de *enunciados probabilísticos* que se formalizan como alguna expresión matemática sobre $p(\cdot)$, la función densidad de probabilidad (PDF por sus siglas en inglés) de alguna variable aleatoria. La probabilidad conjunta de las variables y parámetros del modelo de formación de imagen es, por la definición de probabilidad condicional,

$$p(X_{\mathrm{M}}, X_{\mathrm{I}}, \Gamma, \Theta) = p(X_{\mathrm{M}}, X_{\mathrm{I}} \mid \Gamma, \Theta)p(\Gamma, \Theta)$$

Se toman mediciones de calibración de pares $X_{\mathrm{M}}, X_{\mathrm{I}}$ denotados por $\mathcal{D} = \{({X'_{\mathrm{M}}}^{(i)}, {X'_{\mathrm{I}}}^{(i)})\}_i$, lo que permite inferir un enunciado probabilístico de los parámetros a través de la regla de Bayes,

$$p(\Gamma, \Theta \mid \mathcal{D}) \propto p(\mathcal{D} \mid \Gamma, \Theta)p(\Gamma, \Theta).$$

La verosimilitud $p(\mathcal{D} \mid \Gamma, \Theta)$ es la probabilidad de los datos dados los parámetros y se calcula suponiendo que los pares de datos son independientes entre sí como una productoria de las probabilidades de cada uno:

$$p(\mathcal{D} \mid \Gamma, \Theta) = \prod_i p({X'_{\mathrm{M}}}^{(i)}, {X'_{\mathrm{I}}}^{(i)} \mid \Gamma, \Theta). \tag{5.1}$$

La definición de la verosimilitud $p({X'_{\mathrm{M}}}^{(i)}, {X'_{\mathrm{I}}}^{(i)} \mid \Gamma, \Theta)$ es el corazón del modelo probabilístico. No existe una teoría general de donde deducirla, debe definirse *ad hoc* tal que capture el comportamiento físico determinista del sistema (en este caso el mapeo $X_{\mathrm{M}} = \mathcal{G}(X_{\mathrm{I}}, \Gamma, \Theta)$) y la parte incierta o aleatoria, en general referida al proceso de medición.

La probabilidad a priori $p(\Gamma, \Theta)$ representa información externa a los datos. Puede usarse para insertar información que se tenga previa a la toma de datos o como regularizador de la inferencia.

Una vez inferida información sobre los parámetros dados los datos de calibración $\Gamma, \Theta \mid \mathcal{D}$ usando la ecuación (5.1) se hace la predicción de una nueva observación. Es decir, se calcula la probabilidad predictiva a posteriori de un dato nuevo $X'_{\mathrm{M}}, X'_{\mathrm{I}}$ dados los datos de calibración por el teorema de probabilidad total, haciendo la integral

$$p(X'_{\mathrm{M}}, X'_{\mathrm{I}} \mid \mathcal{D}) = \int p(X'_{\mathrm{M}}, X'_{\mathrm{I}} \mid \Gamma, \Theta)\, p(\Gamma, \Theta \mid \mathcal{D})\, \mathrm{d}\Gamma\, \mathrm{d}\Theta. \tag{5.2}$$

La integral es sobre el espacio de los parámetros Γ, Θ. Aparece de nuevo la verosimilitud $p(X'_{\mathrm{M}}, X'_{\mathrm{I}} \mid \Gamma, \Theta)$. Estos son los pasos para hacer inferencia y predicción bayesiana.

5.1.1. Regresión bayesiana

Notar que en la verosimilitud (ecuación (5.1)) X_{M} no está explícitamente condicionada en X_{I}. Además X_{I} está condicionada en los parámetros Γ, Θ a pesar de que la coordenada en la imagen donde se vea detectado un objeto depende del movimiento del objeto y no de los parámetros del modelo de formación de imagen. Es más adecuado pensar el caso como un modelo de regresión donde se estudia cómo depende X_{M} (la variable de salida o de respuesta) de X_{I} (la variable explicativa o de entrada) y la verosimilitud se escribe $p(X_{\mathrm{M}} \mid X_{\mathrm{I}}, \Gamma, \Theta)$.

Reformulando los pasos de inferencia y predicción en el esquema de regresión bayesiana [42] se toman los mismos datos de $\mathcal{D}$ pero agrupando por un lado las componentes de entrada $\mathcal{D}_{\mathrm{I}} = \{X_{\mathrm{I}}'^{(i)}\}_i$ y por otro las de salida $\mathcal{D}_{\mathrm{M}} = \{X_{\mathrm{M}}'^{(i)}\}_i$. La probabilidad de los parámetros dados los datos (otra vez regla de Bayes) queda

$$p(\Gamma, \Theta \mid \mathcal{D}_{\mathrm{I}}, \mathcal{D}_{\mathrm{M}}) \propto p(\mathcal{D}_{\mathrm{M}} \mid \mathcal{D}_{\mathrm{I}}, \Gamma, \Theta) p(\Gamma, \Theta).$$

De nuevo la verosimilitud puede calcularse como la productoria de las probabilidades de cada par de datos:

$$p(\mathcal{D}_{\mathrm{M}} \mid \mathcal{D}_{\mathrm{I}}, \Gamma, \Theta) = \prod_i p(X_{\mathrm{M}}'^{(i)} \mid X_{\mathrm{I}}'^{(i)} \Gamma, \Theta).$$

Ahora la verosimilitud tiene la forma $p(X_{\mathrm{M}}'^{(i)} \mid X_{\mathrm{I}}'^{(i)} \Gamma, \Theta)$, notablemente la variable $X_{\mathrm{I}}'^{(i)}$ pasó al lado derecho del condicional.

Luego se tiene una nueva observación X_{I}' en la variable de entrada y se quiere predecir la variable de salida X_{M}'. En símbolos

$$p(X_{\mathrm{M}}' \mid X_{\mathrm{I}}', \mathcal{D}_{\mathrm{I}}, \mathcal{D}_{\mathrm{M}}) = \int p(X_{\mathrm{M}}' \mid X_{\mathrm{I}}', \Gamma, \Theta)\ p(\Gamma, \Theta \mid \mathcal{D}_{\mathrm{I}}, \mathcal{D}_{\mathrm{M}})\ \mathrm{d}\Gamma\ \mathrm{d}\Theta.$$

Este proceso de trabajo se acerca mucho más a lo que se necesita, pero tiene un faltante: considerar que las mediciones X_{I}' pueden tener incerteza. En la sección 4.3 se mostró que las detecciones en la imagen pueden tener una incerteza significativa, es fundamental poder incluirla en el modelo probabilístico.

Para incorporar incerteza en X_{I}' al método de regresión bayesiana en la sección siguiente se deducen los pasos del método desde primeros principios. Se parte de la expresión integral de la probabilidad predictiva de la ecuación (5.2), se define la verosimilitud (inspirada en el concepto de regresión) y se resuelven de a uno los términos de la integral pudiendo incorporar incerteza en la medición de coordenadas imagen X_{I}' y tener dos pasos de calibración para parámetros intrínsecos y extrínsecos.

5.2. Método propuesto de calibración y predicción

En la figura 5.1 se demuestran gráficamente los pasos del método propuesto y la relación entre ellos. En la figura 5.1a se toman m imágenes del tablero y se detectan automáticamente las esquinas internas. Así se genera el dataset $\mathcal{D}_1$ que permite estimar la media y varianza $\tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma$ de la probabilidad a posteriori de los parámetros intrínsecos. En la figura 5.1b se muestran las imágenes de calibración extrínseca: el frame de la cámara ojo de pez y a la derecha una imagen satelital. En rojo los

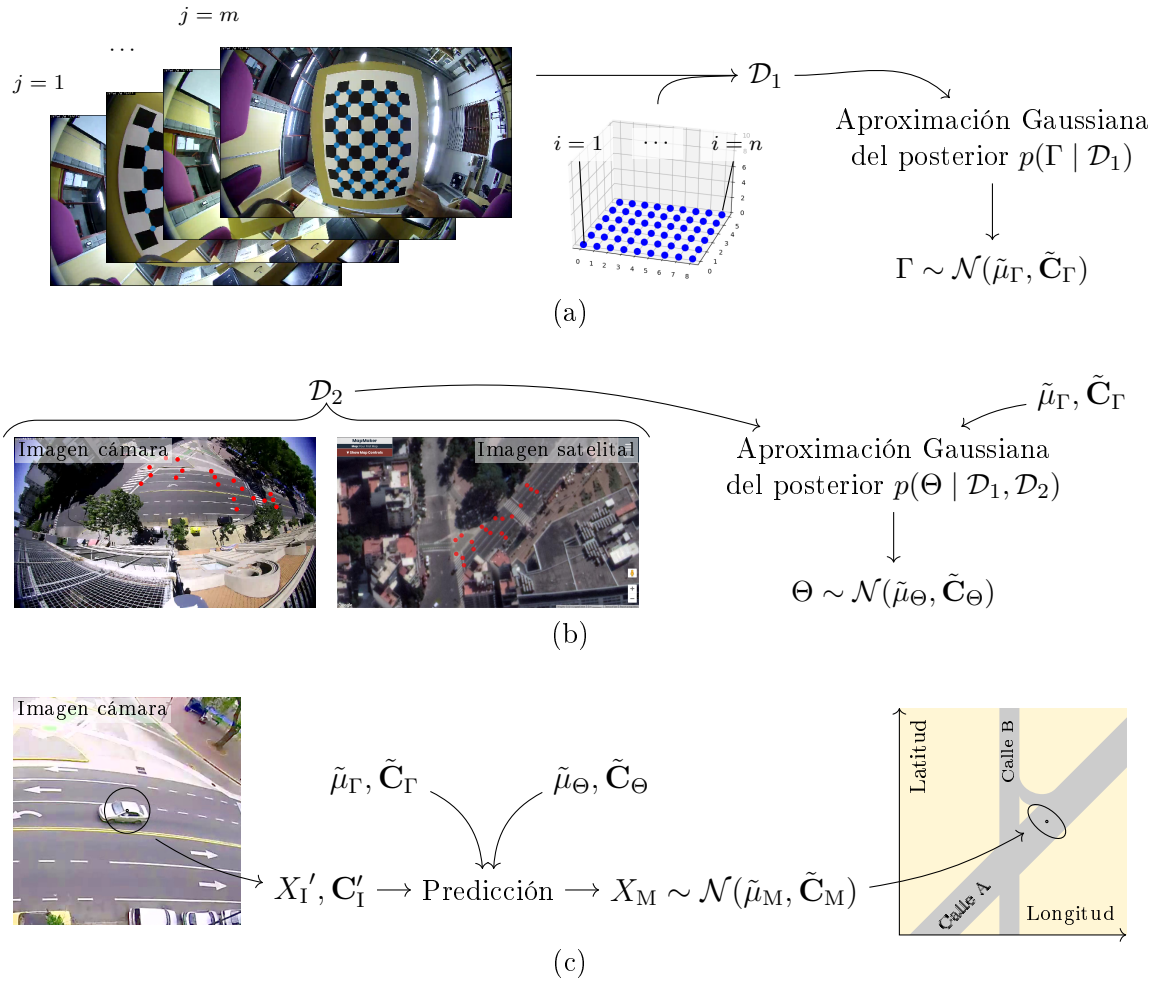


Figura 5.1: (a) Calibración intrínseca. (b) Calibración extrínseca. (c) Geolocalización de vehículos (predicción).

puntos de calibración. Este dataset, $\mathcal{D}_2$, y la información a priori de la pose de la cámara permiten estimar la media y varianza $\tilde{\mu}_\Theta, \tilde{\mathbf{C}}_\Theta$ de la probabilidad a posteriori de la pose de la cámara. Finalmente en la figura 5.1c se detecta el objeto de interés en la imagen(X_I', con incerteza $\mathbf{C}_\mathrm{I}'$) y se combina esta información con lo obtenido en las calibraciones para predecir la localización del objeto en el marco de referencia mundo y su incerteza, $\tilde{\mu}_\mathrm{M}, \tilde{\mathbf{C}}_\mathrm{M}$.

La distribución predictiva de la posición de un objeto en el mundo X_M está condicionada en la detección del objeto en la imagen y en los datos de calibración, $p(X_\mathrm{M} \mid X_\mathrm{I}', \mathbf{C}_\mathrm{I}', \mathcal{D}_1, \mathcal{D}_2)$ [67, 42]. En concordancia con los procedimientos estándar de calibración de cámara, los datos de calibración se separan en dos, datos de calibración intrínseca, $\mathcal{D}_1$, y datos de calibración extrínseca, $\mathcal{D}_2$. Una medición nueva, X_I', corresponde a la localización en la imagen, de un objeto nuevo. El método de

detección de objetos no es de interés en este trabajo pero se supone que retorna una cuantificación de la incerteza de detección. Se denotará como $\mathbf{C}_{\mathrm{I}}'$ la matriz de covarianza asociada a la incerteza de detectar objetos en la imagen.

Por la definición de probabilidad total la distribución de probabilidad predictiva puede expandirse como

$$p(X_{\mathrm{M}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}', \mathcal{D}_1, \mathcal{D}_2) = \int p(X_{\mathrm{M}} \mid X_{\mathrm{I}}, \Gamma, \Theta)\; p(X_{\mathrm{I}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}') \\ p(\Gamma \mid \mathcal{D}_1)\; p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1)\; \mathrm{d}X_{\mathrm{I}}\; \mathrm{d}\Gamma\; \mathrm{d}\Theta.$$

Dentro de la integral hay cuatro términos que se tratarán de la siguiente manera:

- El primer término es la delta de Dirac en la función de retro-proyección,
$$p(X_{\mathrm{M}} \mid X_{\mathrm{I}}, \Gamma, \Theta) = \delta[X_{\mathrm{M}} - \mathcal{G}(X_{\mathrm{I}}, \Gamma, \Theta)].$$
- El segundo término describe la PDF de una variable aleatoria de posición en la imagen, se la supondrá gaussiana dada una medición directa sobre la imagen. Los parámetros de la gaussiana son $X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}'$. En símbolos
$$p(X_{\mathrm{I}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}') = \mathcal{N}(X_{\mathrm{I}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}').$$
- El tercer término $p(\Gamma \mid \mathcal{D}_1)$ es la probabilidad a posteriori de los parámetros intrínsecos dados los datos de calibración intrínseca. Se tratará este término en la sección 5.2.1. El resultado de la estimación de la media y varianza de dicha PDF, supuesta gaussiana también, es
$$p(\Gamma \mid \mathcal{D}_1) = \mathcal{N}(\Gamma \mid \tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma).$$
- El cuarto término $p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1)$ es la PDF a posteriori de la pose de la cámara dados los datos de calibración extrínseca, pero también dados los datos de calibración intrínseca. Éstos últimos son necesarios para la calibración extrínseca en el sentido que requiere tener resuelta previamente la calibración intrínseca. Se expandirá en la sección 5.2.2. La distribución de probabilidad a posteriori, de nuevo, es una distribución gaussiana
$$p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1) = \mathcal{N}(\Theta \mid \tilde{\mu}_\Theta, \tilde{\mathbf{C}}_\Theta).$$

Que la distribución de los parámetros $p(\Gamma \mid \mathcal{D}_1)$ y $p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1)$ sean gaussianas es una decisión de modelo. Se eligió así porque en cálculos exploratorios la población

de parámetros era unimodal y acampanada; y además es lo que permite avanzar con el cálculo de la integral a continuación. Sustituyendo las PDF por las gaussianas que serán estimadas en las secciones subsiguientes resulta en

$$p(X_{\mathrm{M}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}', \mathcal{D}_1, \mathcal{D}_2) = \int \delta[X_{\mathrm{M}} - \mathcal{G}(X_{\mathrm{I}}, \Gamma, \Theta)]\, \mathcal{N}(X_{\mathrm{I}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}') \\ \mathcal{N}(\Gamma \mid \tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma)\, \mathcal{N}(\Theta \mid \tilde{\mu}_\Theta, \tilde{\mathbf{C}}_\Theta)\, \mathrm{d}X_{\mathrm{I}}\, \mathrm{d}\Gamma\, \mathrm{d}\Theta.$$

A pesar de que las PDFs de la integral son supuestas como gaussianas, la integral sigue siendo difícil de evaluar dada la no linealidad de $\mathcal{G}$. Puede resolverse usando estrategias computacionales costosas, pero con el objetivo de que el cálculo se pueda hacer online se aproxima $\mathcal{G}$ como una serie de Taylor a primer orden alrededor de $X_{\mathrm{I}}', \tilde{\mu}_\Gamma, \tilde{\mu}_\Theta$ lo que simplifica la integral al caso de una combinación lineal de variables aleatorias normales mutuamente independientes [61].

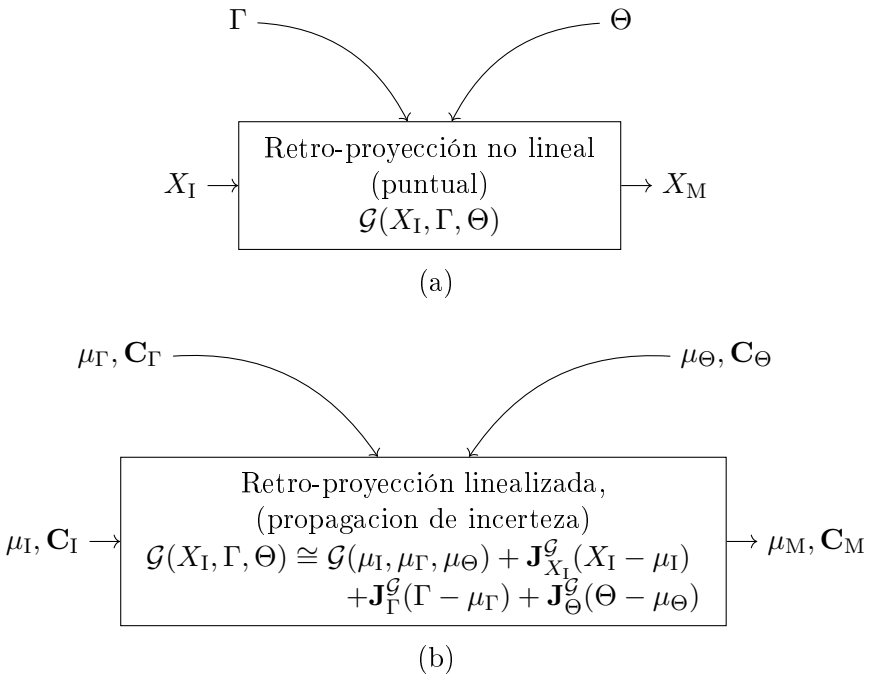


Figura 5.2: (a) La retro-proyección como función que mapea $X_{\mathrm{I}} \to X_{\mathrm{M}}$ y de parámetros Γ, Θ. (b) La propagación de incertezas además de retro-proyectar linealiza $\mathcal{G}$ para estimar la incerteza en X_{M} propagando las incertezas de $X_{\mathrm{I}}, \Gamma, \Theta$.

Se obtiene la fórmula para calcular la PDF predictiva

$$p(X_{\mathrm{M}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}', \mathcal{D}_1, \mathcal{D}_2) = \mathcal{N}(X_{\mathrm{M}} \mid \tilde{\mu}_{\mathrm{M}}, \tilde{\mathbf{C}}_{\mathrm{M}}) \tag{5.3}$$

$$\text{donde } \tilde{\mu}_{\mathrm{M}} = \mathcal{G}(X_{\mathrm{I}}', \tilde{\mu}_\Gamma, \tilde{\mu}_\Theta),$$

$$\text{y } \tilde{\mathbf{C}}_{\mathrm{M}} = \mathbf{J}^{\mathcal{G}}_{X_{\mathrm{I}}} \mathbf{C}_{\mathrm{I}}' {\mathbf{J}^{\mathcal{G}}_{X_{\mathrm{I}}}}^{\top} + \mathbf{J}^{\mathcal{G}}_{\Gamma} \tilde{\mathbf{C}}_\Gamma \mathbf{J}^{\mathcal{G}\top}_{\Gamma} + \mathbf{J}^{\mathcal{G}}_{\Theta} \tilde{\mathbf{C}}_\Theta \mathbf{J}^{\mathcal{G}\top}_{\Theta}.$$

En palabras, la PDF de la posición predicha en el mundo, que es aproximadamente

gaussiana, tiene una media $\tilde{\mu}_{\mathrm{M}}$ que es la evaluación directa de la retro-proyección en las medias de la detección del objeto en la imagen y en las medias de las estimaciones de los parámetros; y la covarianza $\tilde{\mathbf{C}}_{\mathrm{M}}$ combina la incerteza de la detección y de los parámetros a través del Jacobiano $\mathbf{J}^{\mathcal{G}}$.

La figura 5.3 ilustra el cálculo de la distribución predictiva para una representación unidimensional de las variables $X_{\mathrm{I}}, X_{\mathrm{M}}$. A diferencia del cálculo de retro-proyección resuelto en el capítulo 3 que toma coordenadas puntuales en la imagen y valores puntuales de los parámetros y devuelve un valor puntual de coordenadas mundo (figura 5.2a), se agrega a todas las variables una cuantificación de incerteza en forma de matriz de covarianza. Como muestra la figura 5.2b se hace una aproximación lineal de la función de retro-proyección para propagar incertezas.

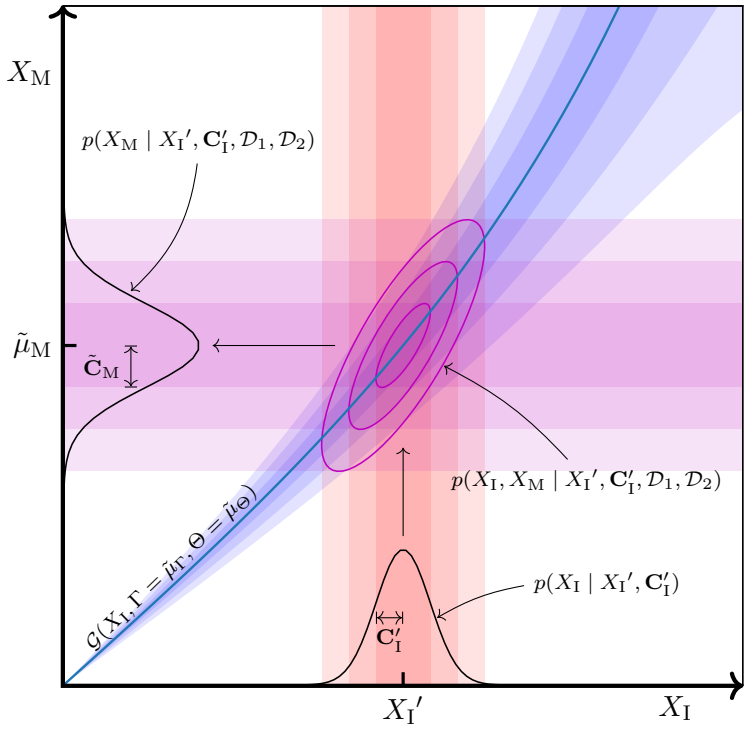


Figura 5.3: Diagrama unidimensional que ilustra cómo la distribución de probabilidad de la entrada detectada $p(X_{\mathrm{I}} \mid X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}')$ (en rosa) se combina con la distribución de probabilidad a posteriori de los parámetros dados los datos, $p(\Gamma, \Theta \mid \mathcal{D}_1, \mathcal{D}_2)$ (en azul) dando como resultado la probabilidad a posteriori de la variable de salida $p(X_{\mathrm{M}} \mid \tilde{\mu}_{\mathrm{M}}, \tilde{\mathbf{C}}_{\mathrm{M}}, \mathcal{D}_1, \mathcal{D}_2)$ (en violeta).

5.2.1. Calibración intrínseca

Los datos para calibración intrínseca se denotan $\mathcal{D}_1 = \{(X'_{\mathrm{M}}{}^{(i,j)}, X_{\mathrm{I}}{}'^{(i,j)}, \mathbf{C}'^{(i,j)}_{\mathrm{I}})\}_{i,j}$ donde cada tupla j, i consiste en coordenadas mundo $X'_{\mathrm{M}}{}^{(i,j)}$ y sus proyecciones correspondientes sobre la imagen $X_{\mathrm{I}}{}'^{(i,j)}$ (ver figura 5.1b). Donde $j \in [1, m]$ es el índice de las imágenes y $i \in [1, n]$ es el índice de las esquinas internas del tablero. Las detecciones podrían no tener todas la misma precisión entonces se incluye explícitamente $\mathbf{C}'^{(i,j)}_{\mathrm{I}}$ como parte de los datos de calibración.

Todos los parámetros del dataset son $\Omega = \{\Gamma, \{\Theta^{(j)}\}_j\}$, donde $\{\Theta^{(j)}\}_j$ es la lista de los parámetros extrínsecos de cada imagen tomada. La densidad de probabilidad a posteriori dados los datos es

$$p(\Omega \mid \mathcal{D}_1) = \frac{p(\mathcal{D}_1 \mid \Omega)\; p(\Omega)}{p(\mathcal{D}_1)}.$$

En la práctica no se tiene información a priori acerca de Ω entonces el posterior es igual a la verosimilitud, que a su vez es el producto de la probabilidad de cada tupla de datos dados los parámetros. En símbolos

$$p(\Omega \mid \mathcal{D}_1) \propto \prod_{j,i} p(X'_{\mathrm{M}}{}^{(i,j)}, X_{\mathrm{I}}{}'^{(i,j)}, \mathbf{C}'^{(i,j)}_{\mathrm{I}} \mid \Gamma, \Theta^{(j)}). \tag{5.4}$$

Cada término en la productoria es la probabilidad de los datos medidos (i, j)-ésimos condicionados en los parámetros. Tomando $p(X'_{\mathrm{M}}{}^{(i,j)}, X_{\mathrm{I}}{}'^{(i,j)}, \mathbf{C}'^{(i,j)}_{\mathrm{I}} \mid \Gamma, \Theta^{(j)})$ y aplicando la definición de probabilidad condicional para dejar $X_{\mathrm{I}}{}'^{(i,j)}, \mathbf{C}'^{(i,j)}_{\mathrm{I}}$ del lado derecho de la condicional rápidamente nos lleva a

$$p(\Omega \mid \mathcal{D}_1) \propto \prod_{j,i} \mathcal{N}(X'_{\mathrm{M}}{}^{(i,j)} \mid \tilde{\mu}^{(i,j)}_{\mathrm{M}}, \tilde{\mathbf{C}}^{(i,j)}_{\mathrm{M}}) \tag{5.5}$$

$$\text{donde } \tilde{\mu}^{(i,j)}_{\mathrm{M}} = \mathcal{G}(X_{\mathrm{I}}{}'^{(i,j)}, \Gamma, \Theta^{(j)}),$$

$$\text{y } \tilde{\mathbf{C}}^{(i,j)}_{\mathrm{M}} = \mathbf{J}^{\mathcal{G}}_{X_{\mathrm{I}}} \mathbf{C}'^{(i,j)}_{\mathrm{I}} \mathbf{J}^{\mathcal{G}}_{X_{\mathrm{I}}}{}^{\top}.$$

Que parece ser una versión simplificada de la ecuación (5.3) porque se toman valores puntuales de $\Gamma, \Theta^{(j)}$ en lugar de una media y varianza. La probabilidad de $\Omega \mid \mathcal{D}_1$ en la ecuación (5.4) puede ser evaluada numéricamente para cualquier valor de Ω, solo requiere los datos de calibración y calcular $\mathcal{G}$ y sus derivadas con respecto a X_{I} como se muestra en la ecuación (5.5). Métodos como Metropolis-Hastings [47] pueden estimar la media y varianza de $\Omega \mid \mathcal{D}_1$. Recordar que de $\Omega = \{\Gamma, \{\Theta^{(j)}\}_j\}$ las posiciones con respecto al patrón de calibración no son de utilidad más adelante,

el objetivo de esta calibración extrínseca es solamente estimar Γ porque la distorsión óptica es una constante intrínseca de la cámara. La estimación de $\{\Theta^{(j)}\}_j$ es colateral. Marginalizando con respecto a $\{\Theta^{(j)}\}_j$ es trivial suponiendo que $\Omega \mid \mathcal{D}_1$ es aproximadamente gaussiana y que las variables Γ, $\{\Theta^{(j)}\}_j$ son independientes. La calibración intrínseca resulta en

$$p(\Gamma \mid \mathcal{D}_1) = \mathcal{N}(\Gamma \mid \tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma). \tag{5.6}$$

Donde $\tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma$ son las componentes de la media y varianza de $\Omega \mid \mathcal{D}_1$ correspondientes a Γ obtenidas por Metropolis-Hastings.

5.2.2. Calibración extrínseca

Luego de hacer la calibración intrínseca en condiciones controladas donde se estimó la PDF de Γ, se instala la cámara en alguna locación urbana apuntando a alguna región de interés (figura 5.1c). El set de datos de calibración extrínseco se denota $\mathcal{D}_2$, consiste en n coordenadas mundo y sus correspondientes detecciones en la imagen, $\{({X'_{\mathrm{M}}}^{(i)}, {X_{\mathrm{I}}}'^{(i)}, \mathbf{C}'^{(i)}_{\mathrm{I}})\}_i$ con $i \in [1, n]$. La calibración extrínseca tiene el objetivo de estimar la media y varianza de la pose de la cámara Θ.

Por la ley de probabilidad total y suponiendo independencia entre Θ y $\mathcal{D}_1$ y también entre Γ y $\mathcal{D}_2$ la probabilidad a posteriori de Θ es

$$p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1) = \int p(\Theta \mid \mathcal{D}_2, \Gamma)\; p(\Gamma \mid \mathcal{D}_1) \mathrm{d}\Gamma, \tag{5.7}$$

donde $p(\Gamma \mid \mathcal{D}_1) = \mathcal{N}(\Gamma \mid \tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma)$ (ecuación (5.6)). Por la regla de Bayes

$$p(\Theta \mid \mathcal{D}_2, \Gamma) \propto p(\mathcal{D}_2 \mid \Theta, \Gamma) p(\Theta \mid \Gamma)$$

pero como Θ y Γ son independientes $p(\Theta \mid \Gamma) = p(\Theta)$. La distribución a posteriori de Θ es

$$p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1) \propto p(\Theta) \int p(\mathcal{D}_2 \mid \Theta, \Gamma)\; p(\Gamma \mid \mathcal{D}_1) \mathrm{d}\Gamma. \tag{5.8}$$

Nótese que se puede incorporar información a priori $p(\Theta)$, después de todo la pose de la cámara es una magnitud física sobre la que puede tenerse información a partir de la instalación, a diferencia de los parámetros intrínsecos que dependen del modelo de distorsión elegido y son mucho más difíciles de interpretar físicamente.

Tal como en ecuación (5.5) la verosimilitud $p(\mathcal{D}_2 \mid \Theta, \Gamma)$ se puede calcular como

el producto de la verosimilitud de cada tupla de datos resultando en

$$p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1) \propto p(\Theta) \prod_i \mathcal{N}({X'_{\mathrm{M}}}^{(i)} \mid \tilde{\mu}_{\mathrm{M}}^{(i)}, \tilde{\mathbf{C}}_{\mathrm{M}}^{(i)}), \tag{5.9}$$

$$\begin{aligned} \text{donde} \quad \tilde{\mu}_{\mathrm{M}}^{(i)} &= \mathcal{G}({X_{\mathrm{I}}}'^{(i)}, \Theta, \tilde{\mu}_\Gamma), \\ \text{y} \quad \tilde{\mathbf{C}}_{\mathrm{M}}^{(i)} &= \mathbf{J}_{X_{\mathrm{I}}}^{\mathcal{G}} \mathbf{C}_{\mathrm{I}}'^{(i)} {\mathbf{J}_{X_{\mathrm{I}}}^{\mathcal{G}}}^T + \mathbf{J}_\Gamma^{\mathcal{G}} \tilde{\mathbf{C}}_\Gamma \mathbf{J}_\Gamma^{\mathcal{G}T}. \end{aligned}$$

Ahora también se necesita la derivada de $\mathcal{G}$ respecto a Γ, es decir $\mathbf{J}_\Gamma^{\mathcal{G}}$. Nuevamente, por métodos computacionales aplicados a la ecuación (5.9) se puede estimar la media $\tilde{\mu}_\Gamma$ y varianza $\tilde{\mathbf{C}}_\Gamma$ de $\Theta \mid \mathcal{D}_2, \mathcal{D}_1$ tal que

$$p(\Theta \mid \mathcal{D}_2, \mathcal{D}_1) = \mathcal{N}(\Theta \mid \tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma).$$

5.2.3. Resumen de calibración y predicción

El procedimiento es el siguiente. Tomar imágenes de un patrón de calibración en laboratorio como se ve en la figura 5.1a produce los datos de calibración $\mathcal{D}_1$ y el resultado de la calibración es la PDF a posteriori de los parámetros $p(\Gamma \mid \mathcal{D}_1)$ de media $\tilde{\mu}_\Gamma$ y varianza $\tilde{\mathbf{C}}_\Gamma$. Se las obtiene computando la posteriori con la ecuación (5.5) usando algún método estándar de integración numérica como Metropolis-Hastings. Cuando se instala la cámara en su posición de monitoreo se extraen los puntos de calibración extrínseca $\mathcal{D}_2$ (figura 5.1b) que se usan para estimar la media y varianza $\tilde{\mu}_\Theta, \tilde{\mathbf{C}}_\Theta$ de la distribución de probabilidad a posteriori $p(\Theta \mid \mathcal{D}_1, \mathcal{D}_2)$ calculada según la ecuación (5.9)). Esto completa el proceso de calibración. Con la detección de un vehículo, $X_{\mathrm{I}}', \mathbf{C}_{\mathrm{I}}'$, la PDF predictiva de sus coordenadas mundo se calculan con la ecuación (5.3) como se ilustra en la figura 5.1c. Esta predicción puede hacerse online porque tiene un costo computacional muy bajo.

5.3. Notas sobre la implementación en Python

En esta sección se explica cómo se implementó en Python 3 [68] con el paquete PyMC3 [48] un modelo probabilístico que incluya la propagación de incerteza, cómo se aprovechó la capacidad de vectorización y broadcasting del paquete NumPy [54] para que las operaciones matriciales sean rápidas, y cómo se calcularon analíticamente los Jacobianos para la propagación de incerteza.

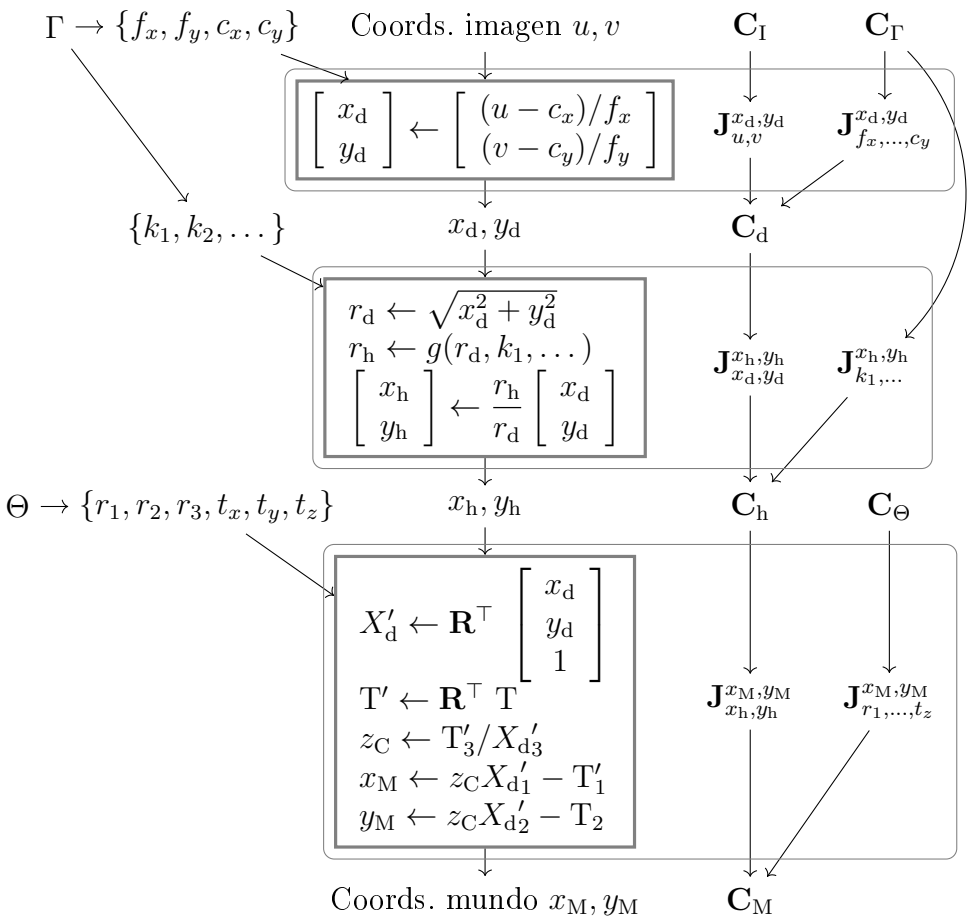


Figura 5.4: Diagrama de flujo de retro-proyección.

5.3.1. Acumulación de covarianza

Para propagar incertezas linealmente como se mostró en la ecuación (5.3) se necesita calcular los Jacobianos de $\mathcal{G}$ respecto a todas las variables. La función $\mathcal{G}$ se define como en el algoritmo 2. En la figura 5.4 se representan los pasos del algoritmo en forma de diagrama de flujo. El lado izquierdo del diagrama son los cálculos de retro-proyección que se separan en tres bloques. La parte intrínseca lineal (retorna $x_\mathrm{d}, y_\mathrm{d}$), la extrínseca no lineal (retorna $x_\mathrm{h}, y_\mathrm{h}$) y la proyección al plano mundo (retorna $x_\mathrm{M}, y_\mathrm{M}$). Cada bloque tiene dos entradas: una variable de coordenada y los parámetros necesarios para calcular las coordenadas de salida a partir de las coordenadas de entrada.

Del lado derecho del diagrama se esquematiza el cálculo de las covarianzas. Cada bloque de retro-proyección viene acompañado por los Jacobianos de las coordenadas de salida respecto a las coordenadas de entrada y los parámetros del bloque. En el

diagrama una flecha que va de una covarianza a un Jacobiano:

$$\begin{array}{c} \mathbf{C}_{\mathrm{I}} \\ \downarrow \\ \mathbf{J}_{u,v}^{x_{\mathrm{d}},y_{\mathrm{d}}} \end{array}$$

representa propagar la covarianza $\mathbf{C}_{\mathrm{I}}$ a través del Jacobiano $\mathbf{J}_{u,v}^{x_{\mathrm{d}},y_{\mathrm{d}}}$ mediante la operación

$$\mathbf{J}_{u,v}^{x_{\mathrm{d}},y_{\mathrm{d}}}\ \mathbf{C}_{\mathrm{I}}\ \mathbf{J}_{u,v}^{x_{\mathrm{d}},y_{\mathrm{d}}\top}.$$

Mientras que dos flechas entrantes a una covarianza:

$$\begin{array}{ccc} \mathbf{C}_A & & \mathbf{C}_B \\ \downarrow & \swarrow & \\ \mathbf{C}_{\mathrm{d}} & & \end{array}$$

representa sumar las covarianzas $\mathbf{C}_{\mathrm{d}} = \mathbf{C}_A + \mathbf{C}_B$.

Al recorrer el diagrama de arriba hacia abajo se comienza por las covarianzas en imagen $\mathbf{C}_{\mathrm{I}}$ y los parámetros intrínsecos $\mathbf{C}_\Gamma$ y se incorpora la covarianza de los parámetros extrínsecos $\mathbf{C}_\Theta$ hasta obtener la covarianza que representa la incerteza en las coordenadas mundo $\mathbf{C}_{\mathrm{M}}$. Este proceso de acumular progresivamente la incerteza en cada etapa es equivalente a la aplicación de la regla de la cadena que permitiría calcular los Jacobianos de X_{M} respecto a $X_{\mathrm{I}}, \Gamma, \Theta$ como se hizo en la ecuación (5.3).

Un detalle que no se muestra en el diagrama es cómo propagar los componentes cruzados de $\mathbf{C}_\Gamma$. Si se descompone la matriz en cuatro bloques

$$\mathbf{C}_\Gamma = \left[\begin{array}{c|c} \mathbf{C}_f & \mathbf{C}_{fk} \\ \hline \mathbf{C}_{fk}^\top & \mathbf{C}_k \end{array}\right]$$

queda claro cómo sumar la incerteza de los parámetros intrínsecos lineales $\{f_x, f_y, c_x, c_y\}$, denotada $\mathbf{C}_f$, para obtener $\mathbf{C}_{\mathrm{d}}$ y también la de los no lineales $\{k_1, k_2, \dots\}$, denotada $\mathbf{C}_k$, para obtener $\mathbf{C}_{\mathrm{h}}$. Pero en el caso de las componentes de covarianza cruzada entre los lineales y no lineales $\mathbf{C}_{fk}$ se incorporan a $\mathbf{C}_{\mathrm{h}}$ sumando el término

$$2\ \mathbf{J}_{f_x,\dots,c_y}^{x_{\mathrm{h}},y_{\mathrm{h}}}\ \mathbf{C}_{fk}\ \mathbf{J}_{k_1,\dots}^{x_{\mathrm{h}},y_{\mathrm{h}}\top}.$$

5.3.2. Derivadas analíticas

El cálculo de las derivadas con que se construyeron los Jacobianos se hizo analíticamente. La ventaja de tenerlas en forma analítica es que se pueden implementar en funciones de Python que aprovechan la eficiencia de Numpy. Volviendo a la figura 5.4 el primer bloque de distorsión lineal se puede derivar a mano.

El segundo bloque también puede derivarse a mano (es particularmente fácil si g es el modelo estereográfico). Recordando del capítulo 2 la función de distorsión radial puede ser una simple tangente (estereográfico) que es fácil de tratar analíticamente o un polinomio o peor un cociente de polinomios. Para resolver la retro-proyección para todos los modelos se usó la propiedad de que el producto de la derivada de una función y la derivada de la función inversa es la identidad. Es decir que el Jacobiano en el sentido de retro-proyección $\mathbf{J}^{x_\mathrm{h},y_\mathrm{h}}_{x_\mathrm{d},y_\mathrm{d}}$ puede calcularse como la inversa del Jacobiano de la proyección $\mathbf{J}^{x_\mathrm{d},y_\mathrm{d}}_{x_\mathrm{h},y_\mathrm{h}}$ que sí se puede tratar analíticamente.

Finalmente las derivadas del tercer bloque del diagrama en la figura 5.4 se calcularon con asistencia de la librería Sympy [69]. Fue fundamental la detección y colección de subexpresiones comunes que permitió que la derivada calculada simbólicamente se tradujera muy fácilmente en una expresión ejecutable numéricamente de bajo costo computacional.

5.3.3. Vectorización y broadcasting

La retro-proyección requiere hacer operaciones de la forma $\mathbf{JCJ}^\top$. En los pasos de calibración cada imagen tiene n puntos de calibración que deben ser retro-proyectados con su incerteza para calcular la probabilidad a posteriori de los parámetros. La propagación de incerteza es lineal e implica repetir ese producto para todos los puntos. Más aún, como se va a usar un método de optimización tipo Monte Carlo (MC) entonces la operación va a repetirse para una población de realizaciones para cada iteración del método. Por eso es importante que la operación se haga con el menor costo computacional posible. Para eso se aprovechó la propiedad de broadcasting y de vectorización de Numpy.

Por ejemplo, se tienen dos arrays de Numpy, `Jac` de tamaño $(n, 2, 6)$ y `Cov` de tamaño $(n, 6, 6)$ y se quiere obtener el resultado de propagar las n matrices de covarianzas de 6×6 con los correspondientes n Jacobianos de 2×6. El resultado que se quiere obtener es un array de tamaño $(n, 2, 2)$ que contiene las n matrices de covarianza de 2×2. Las funciones de producto punto o producto tensorial de Numpy retornan el producto de todos los Jacobianos con todas las covarianzas, mientras que se quiere solo el producto entre los pares correspondientes de matrices. Para evitar hacer un bucle for que es lento en Python se recurrió a las propiedades de indexado y *broadcasting* de arrays de Numpy [70],

```
Jac.shape # -> (54, 2, 6)
Cov.shape # -> (54, 6, 6)

# indices extra para broadcasting
Jac_aux = Jac.reshape((-1, 2, 6, 1, 1))
Cov_aux = Cov.reshape((-1, 1, 6, 6, 1))

# se aprovecha el broadcasting en el producto
Cov_ret = (Jac_aux *
           Cov_aux *
           Jac_aux.transpose((0,4,3,2,1))
           ).sum((2,3)) # suma del producto punto

Cov_ret.shape # -> (54, 2,2)
```

De esta manera se aprovecha la velocidad de las operaciones vectorizadas de Numpy para hacer la propagación de incerteza.

5.3.4. El modelo probabilístico en PyMC3

Luego de definir el modelo probabilístico en papel hay que hacer uso de alguna librería para inferencia bayesiana, en Python el paquete recomendado es PyMC3 [48] que implementa métodos de Cadena de Markov Monte Carlo. Por ejemplo la

definición de un modelo probabilístico en PyMC3

```
import pymc3 as pm
projectionModel = pm.Model()

parShape = (6) # tamaño del array de parámetros
data # array que contiene los datos de calibración

with projectionModel:
    # Priors para parámetros del modelo
    params = pm.Normal('params', mu=0, sigma=1, shape=parShape)

    # Aplicar el modelo a los parámetros
    mu = ...(params)... # una serie de operaciones

    Y_obs = pm.Normal('Y_obs', mu=mu, sd=1, observed=data)
```

Donde se definió un modelo probabilístico empezando por la probabilidad a priori sobre los parámetros. Con esos parámetros se calculó `mu` y se incluyeron los datos observados `data` con los que evaluar la probabilidad a posteriori de los parámetros. Importante: se definió que los datos observados deben seguir una distribución normal de esperanza `mu` (que se calcula a partir de los parámetros) y desviación estándar 1.

Una limitación de PyMC3 es que no hay una manera de que el set de datos observados sigan distribuciones normales bivaluadas y además que cada dato tenga una varianza distinta a los demás [71] Se formuló el modelo de una manera alternativa, inspirada en la idea de normalización de elipses que se explicó en la figura 6.2 para comparar todas las proyecciones en pie de igualdad. Theano [72] es el soporte computacional de PyMC3. Se definió una operación a medida usando `theano.Op()`. Dentro de esa operación de Theano se hace la retro-proyección y propagación de incerteza y *se normalizan los residuos entre la retro-proyección y los datos observados usando la covarianza predicha.* El objetivo es que si el modelo es correcto entonces los residuos normalizados sigan una distribución normal unitaria de media cero. En pseudo código la operación definida es

```
# puntos de calibración son variables globales
imagePoints # puntos en imagen
objpoints2D # en plano mundo


class ProjectionT(theano.Op):
    # Operación de Theano
    itypes = [T.dvector] # tipo de entrada
    otypes = [T.dtensor3]  # tipo de salida

# Python implementation:
def perform(self, node, inputs_storage, output_storage):
# inputs_storage contiene los parámetros del modelo
# retro-projeccion y propagacion incerteza
xy, Cm <- ...retroproyeccion(imagePoints, inputs_storage)...

# los residuos
        xy -= objpoints2D

# singular value decomposition para obtener la matriz A
# que transforma el espacio tal que las Cm sean la identidad
        S = np.linalg.inv(cm)
        u, s, v = np.linalg.svd(S)
        s = np.sqrt(s)
        A <- ... u * s * v ...
# normalizo residuos según A
        xy <- ... xy.dot(A) ...

# valores de salida
        output_storage[0][0] = xy

    # optional:
```

```
    check_input = True
```

Los datos empíricos de los puntos de calibración se definen como variables globales. Dentro de la operación se usa los parámetros que se quiere calibrar que están almacenados en el vector `inputs_storage` (para la calibración intrínseca es una concatenación de los intrínsecos y de todos los extrínsecos de todas las imágenes del tablero de calibración, es un vector de 201 componentes). Se calcula la retro-proyección con propagación de incerteza. Se usa las covarianzas de la propagación para calcular matriz `A` que sirve para hacer un cambio de base. El cambio de base es tal que los residuos de la retro-proyección deberían tener media cero y covarianza unitaria. De esta manera la salida de esta operación de Theano es fácil de incorporar

```
# variable global dummy de datos observados
observedNormed = np.zeros((nIm * nPt * 2))

# operacion de Theano
projTheanoWrap = ProjectionT()

with projectionModel:
    # Priors for unknown model parameters
    xAl = pm.Uniform('xAl', lower=allLow, upper=allUpp,
     shape=allLow.shape, transform=None)

    # apply numpy based function wrapped in Theano
    xyMNor = projTheanoWrap(xAl)

# residuos normalizados
    mu = T.reshape(xyMNor, (-1, ))
    Y_obs = pm.Normal('Y_obs', mu=mu, sd=1,
      observed=observedNormed)
```

De este modo fue posible definir el modelo probabilístico tal que los datos de

calibración se compararan con la distribución de probabilidad predicha por la propagación de incertezas.

5.3.5. Differential Evolution Metropolis

Entre los algoritmos implementados en PyMC3 para inferencia el más recomendado es NUTS (No-U-Turn-Sampler) [73], basado en el gradiente de la probabilidad a posteriori, es una mejora sobre Monte Carlo Hamiltoniano. Es decir que se necesita poder calcular la derivada de la PDF a posteriori respecto a los parámetros. En esta tesis la probabilidad a posteriori ya implica hacer la derivada del modelo de retro-proyección, está fuera de las posibilidades hacer una segunda de derivada de la retro-proyección. Incluso se dificulta la implementación de derivadas automáticas en PyMC3 porque las operaciones involucradas en la retro-proyección no están todas disponibles en Theano.

Entre los métodos no basados en el gradiente se encuentra DEM (Differential Evolution Metropolis)[74] que difiere del tradicional Metropolis-Hastings en la manera de generar muestras candidatas. Tiene la ventaja de ser muy simple, proponer saltos que son de la escala y orientación de la distribución objetivo y buen rendimiento en espacios de dimensión alta [75]. Se lo eligió porque funciona adecuadamente en espacios de dimensión alta (en este caso de 201 dimensiones) y porque no necesita el gradiente de la probabilidad a posteriori.

Una muestra candidata se genera de la forma

$$\mathbf{x}_p = \mathbf{x}_i + \lambda(\mathbf{x}_{R1} - \mathbf{x}_{R2}) + \mathbf{e}$$

Como se muestra en la figura 5.5 $\mathbf{x}_i$ es la posición actual, $\mathbf{x}_{R1}, \mathbf{x}_{R2}$ son dos muestras obtenidas anteriormente elegidas al azar. $\mathbf{e}$ es un vector extraído de una distribución de propuestas como Metropolis-Hastings, de una varianza menor a la distribución objetivo. Se debe proveer a DEM una medida de desviación estándar S para parametrizar la distribución de propuestas. Por defecto $\lambda = 2.38/\sqrt{(2 \text{ ndim})} \sim 0.119$ y la escala de $\mathbf{e}$ es 0.001 veces la desviación estándar de la distribución de propuestas.

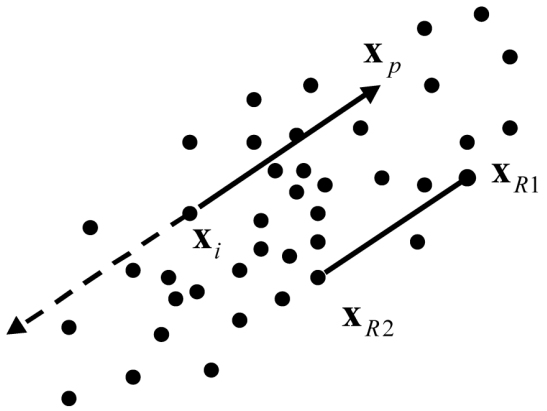


Figura 5.5: Extraído de [74], ilustra cómo se proponen los saltos de muestreo en DEM.

Los resultados mostrados en el capítulo 6 fueron obtenidos usando DEM. Se definió S haciendo simulaciones cortas de prueba y error hasta observar que las covarianza de las trazas había llegado a un estacionario. La escala de $\mathbf{e}$ y λ se definieron en 0.04.

Capítulo 6

Resultados del enfoque bayesiano

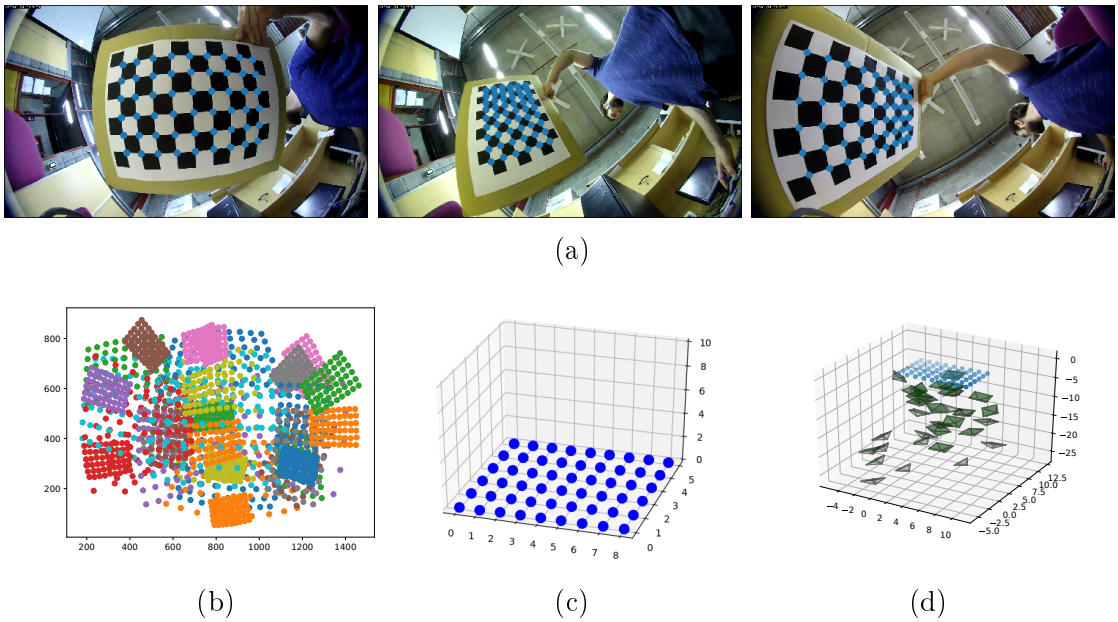
En este capítulo se muestra que la aproximación lineal para propagación de incertezas es suficientemente precisa, comparándola con una propagación usando Monte Carlo, que es computacionalmente más costoso pero que funciona en casos de propagación no lineal.

La calibración en dos pasos y la predicción se evalúan primero sobre datos simulados. Se generan datos realistas de fotos de tableros de calibración para la calibración intrínseca y un total de seis posiciones de instalación final de la cámara para evaluar la calibración extrínseca.

Luego, se usan datos de calibración reales obtenidos en condiciones controladas de laboratorio para estimar los parámetros intrínsecos. La cámara después se instaló en un sitio de testeo y se marcaron manualmente puntos de calibración para estimar los parámetros extrínsecos.

Finalmente se muestra la incerteza predicha de las posiciones mundo de un vehículo detectado en la secuencia de video.

Se usó un tablero cuadriculado de 37cm de largo con 9×6 esquinas internas como patrón de calibración intrínseco. Se obtuvieron $m = 33$ imágenes del patrón sobre las que se aplicó el detector de esquinas de OpenCV como se muestra en la figura 6.1a. Las imágenes se obtuvieron tal que se abarcara todo el campo de visión [76], las esquinas detectadas se muestran en la figura 6.1b. La función de OpenCV



(a) (b) (c) (d)

Figura 6.1: (a) Tres de las 33 imágenes del patrón de calibración, un tablero cuadriculado de 9×6. (b) Una superposición de todas las esquinas internas en la imagen. (c) Las coordenadas 3D de las esquinas del tablero en el marco de referencia mundo. (d) Las 33 poses de la cámara que se obtiene de la calibración provista por OpenCV.

`calibrateCamera` toma las esquinas detectadas y sus correspondientes posiciones en 3D (figura 6.1c) y retorna 33 poses de cámara, mostradas en la figura 6.1d. Como se explicará oportunamente las esquinas internas del tablero y la pose estimada de la cámara se usan como condiciones iniciales para algoritmos de sampleo o como *ground truth* a partir de donde fabricar datos sintéticos.

Tanto la adquisición de imágenes y video como el procesamiento off line de datos se realizaron en una computadora de escritorio con sistema operativo Ubuntu Linux usando scripts de Python 3 [68] y las librerías NumPy [54], SciPy [77], Matplotlib [78], y el entorno de desarrollo Spyder IDE [79]. Los algoritmos de calibración desarrollados por Bouguet [36] (fueron el punto de partida de muchos de los cálculos) están implementados en OpenCV [34] que también se usó para manipulación general de imágenes. La librería PyMC3 [48] se usó para las simulaciones Monte Carlo.

6.1. Validación de propagación lineal de incerteza con Monte Carlo

En esta sección se compara la propagación de incerteza mediante aproximación de primer orden contra la propagación no lineal de Monte Carlo. En el corazón del modelo estereográfico la función de distorsión radial es altamente no lineal porque describe la distorsión óptica que caracteriza a las cámaras ojo de pez. Además, cualquier modelo de formación de imagen debe incluir una proyección de perspectiva que es no lineal con respecto a los parámetros de pose de la cámara. Por esto no es inmediatamente evidente que la aproximación de Taylor a primer orden sea útil en la práctica.

De manera similar a lo hecho por Criminisi et al. [80] se genera una población de tuplas $(X_{\mathrm{I}}, \Gamma, \Theta)$ que obedecen distribuciones gaussianas. Las variables cumplen $X_{\mathrm{I}} \sim \mathcal{N}(\mu_{\mathrm{I}}, \mathbf{C}_{\mathrm{I}})$, $\Gamma \sim \mathcal{N}(\mu_{\Gamma}, \mathbf{C}_{\Gamma})$ y $\Theta \sim \mathcal{N}(\mu_{\Theta}, \mathbf{C}_{\Theta})$. Cada tupla generada se usa para retro-proyectar a coordenadas mundo. El set de puntos resultante en el plano del suelo será comparado con PDF gaussianas obtenidas por propagación lineal de incertezas. Si la aproximación lineal es válida entonces la media y varianza de las partículas MC estarán cerca de la media y varianza obtenidas por propagación lineal.

Los datos recabados para la calibración intrínseca son una fuente de coordenadas realistas. En lugar de definir arbitrariamente una cantidad de poses que imiten la calibración se tomaron prestadas las poses de la cámara estimadas por OpenCV ya que cubren una abanico razonable de posiciones. Se toman las 33 poses estimadas por OpenCV como $\{\Theta^{(j)}\}_j$; junto con las posiciones 3D (mundo) de las esquinas internas y los parámetros intrínsecos Γ que surgieron de trabajos previos con la cámara. Se define $\Gamma = [800, 465, 800]^{\top}$. El centro óptico se supone en el centro de la imagen, por eso c_x y c_y son la mitad del alto y ancho de la imagen respectivamente. Para proponer un valor de k se observa que el campo visual de 183° se termina al llegar aproximadamente al borde de la imagen, es decir que se puede aproximar el parámetro como la mitad de ancho de la imagen que es 1600 píxeles.

Para generar una población de muestras para Monte Carlo primero se generan simulaciones de las esquinas internas del tablero en la imagen. Usando las ecuacio-

nes (2.1) a (9) se proyectan las 54 coordenadas mundo de las esquinas internas del tablero ${X_{\mathrm{M}}}^{(i,j)}$ a la imagen, a su vez para las 33 poses. Finalmente se tienen 33 sets de 54 coordenadas imagen (en píxeles) $\mu_{\mathrm{I}}^{(i,j)}$. Las coordenadas imagen se supone que se detectan con una incerteza de 1 píxel de desviación estándar, es decir $\mathbf{C}_{\mathrm{I}} = \mathbf{I}_2$. Se adicionó ruido gaussiano de esa intensidad (desviación estándar de un píxel) a las coordenadas en la imagen para simular el error de detección. Se considera razonable (mas adelante será confirmado empíricamente) que los parámetros intrínsecos y extrínsecos pueden estimarse con precisión en tres cifras significativas. Es decir que la desviación estándar es de 10^{-3} veces el valor del parámetro, quedando definidas $\mathbf{C}_\Gamma, \mathbf{C}_\Theta$.

El siguiente paso es generar la población de $N_{MC} = 5000$ muestras Monte Carlo, produciendo detecciones en imagen, parámetros intrínsecos y extrínsecos según las medias y varianzas anteriores. Las N_{MC} tuplas son alimentadas a la función de retro-proyección. No se examina solamente la salida final en coordenadas mundo sino también las coordenadas intermedias X_{h}, correspondientes a una cámara sin distorsión.

En la figura 6.2a las coordenadas imagen se muestrean de una distribución gaussiana, 5000 muestras para cada esquina detectada, el inserto en la derecha muestra la comparación entre las muestras de una esquina, las elipses de covarianza de 90 % de probabilidad, asociadas a los puntos Monte Carlo (en negro) y al análisis a primer orden (en rojo). Notar la curvatura por la distorsión óptica.

En la figura 6.2b las mismas muestras pasan por la corrección de distorsión intrínseca (en el algoritmo 2 son las líneas 3 a 6) y se puede ver que la grilla de 54 puntos ya no está curvada. Las elipses se han estirado en la dirección radial, también la discrepancia entre las elipses Monte Carlo y de aproximación lineal se ha incrementado levemente por el error de linealización en la dirección radial.

En la figura 6.2c a esos mismos puntos se les ha realizado la proyección de perspectiva sobre el plano $z_{\mathrm{M}} = 0$ (algoritmo 2 líneas 10 a 14). La incerteza en los seis parámetros de pose se incorpora en el mapeo y aumenta el tamaño de las elipses. Las elipses de covarianza de Monte Carlo y de propagación lineal se mantienen virtualmente indistinguibles.

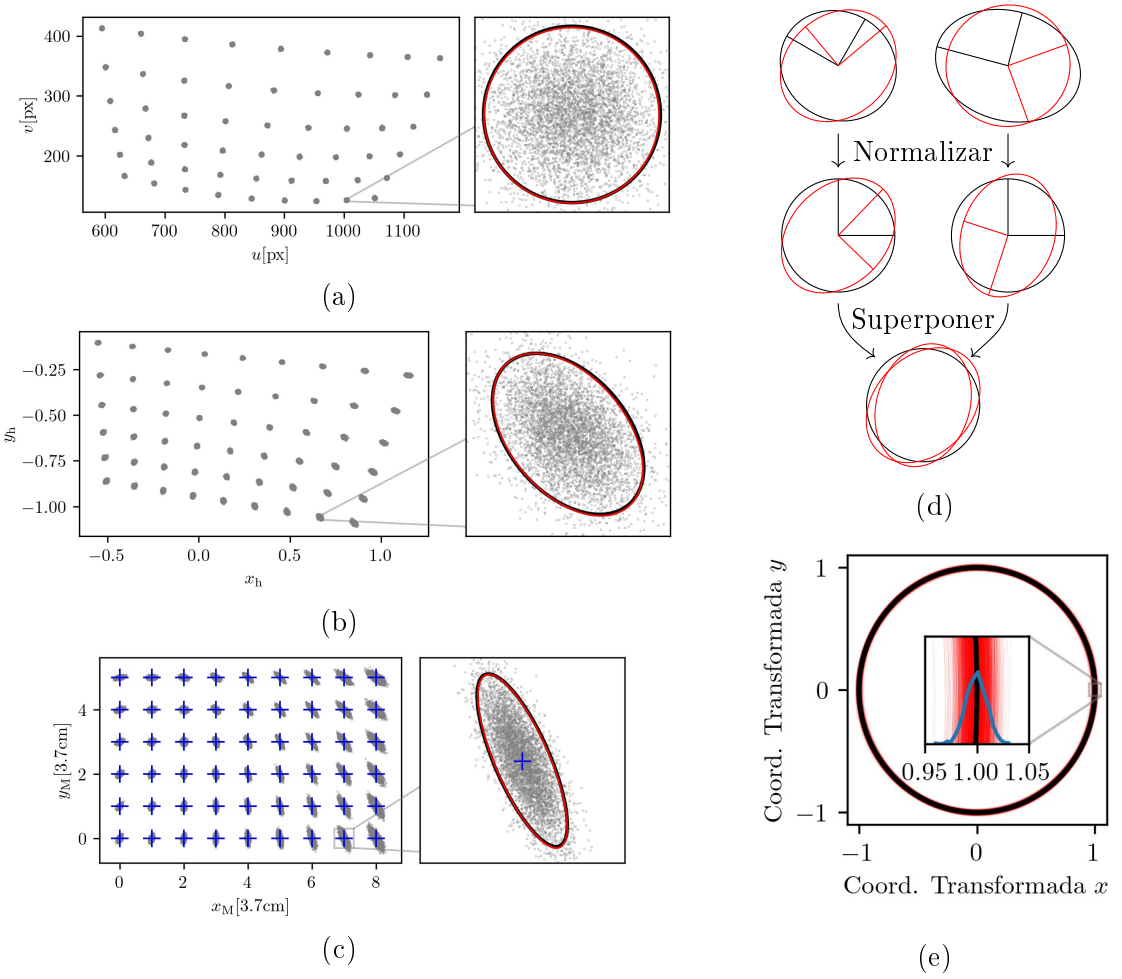


Figura 6.2: (a) Población MC en coordenadas imagen. (b) Misma población convertida a coordenadas homogéneas. (c) Retro-proyección a coordenadas mundo. (d) Esquema de normalización de elipses. (e) Comparación entre MC (círculo negro) con las elipses predichas analíticamente.

Hay $1782 = 33 \times 54$ puntos de calibración, para cada uno se obtuvo un par de PDFs de predicción. La primera PDF por propagación lineal de incertezas y las segunda ajustando una gaussiana a las N_{MC} muestras Monte Carlo. Para evaluar cualitativamente que las elipses hayan resultado muy similares en todos los 1782 puntos de calibración, la elipse por propagación lineal es transformada a una nueva base donde su contraparte calculada por Monte Carlo es una distribución gaussiana unitaria de media cero. Ilustrado en la figura 6.2d, se le resta a la elipse de propagación lineal (trazo rojo) el centro de la elipse Monte Carlo (en trazo negro) y se rota y escalea tal que la elipse Monte Carlo se convierte en el círculo unitario.

Las elipses transformadas de esta manera pueden graficarse solapadas en una misma figura porque todas deben compararse con el mismo círculo de referencia. En

la figura 6.2e las líneas rojas son las elipses de propagación lineal que forman un halo rojo congruente con el circulo unitario, es decir que en los 1782 casos testeados la propagación lineal de incertezas da virtualmente el mismo resultado que haber hecho una simulación por Monte Carlo. El inserto central muestra un detalle más fino de la coincidencia de las elipses con el círculo.

6.1.1. Calibración intrínseca con datos sintéticos

Para evaluar el proceso de calibración intrínseca se estima la distribución de probabilidad a posteriori de los tres parámetros intrínsecos con los mismos datos sintéticos de la sección anterior. La figura 6.1d muestra las 33 posiciones de la cámara con respecto al tablero de calibración y en la figura 6.1b todas las detecciones de las esquinas internas en una sola imagen.

Los tres parámetros intrínsecos y los 33×6 parámetros intrínsecos (6 por cada imagen) forman un vector aleatorio de 201 componentes. La probabilidad de este vector se evalúa como en la ecuación (5.4). Se extrajeron 442 cadenas de Markov de 50 muestras con el algoritmo Differential Evolution Metropolis (DEM, ver sección 5.3 para detalles de su implementación) [48, 74]. Los valores de inicio de las cadenas se definieron *ad hoc* para reducir el tiempo de *burn-in*, es decir el tiempo que tarda la simulación en llegar un estado estacionario de MCMC. Los histogramas de las muestras extraídas eran unimodales y en forma de campana.

	Valor real	Media Muestral ($\tilde{\mu}_\Gamma$)	Desv. Estándar Muestral
c_x	800	800.001	0.05
c_y	452	451.999	0.06
k	800	799.999	0.13

$$\tilde{\mathbf{C}}_\Gamma = \begin{bmatrix} 0.00247 & 0.00020 & -0.00030 \\ 0.00020 & 0.00379 & 0.00123 \\ -0.00030 & 0.00123 & 0.01793 \end{bmatrix} \tag{6.1}$$

Cuadro 6.1: Comparación entre los valores *ground truth* de los parámetros intrínsecos y los estimados muestreando la probabilidad a posteriori.

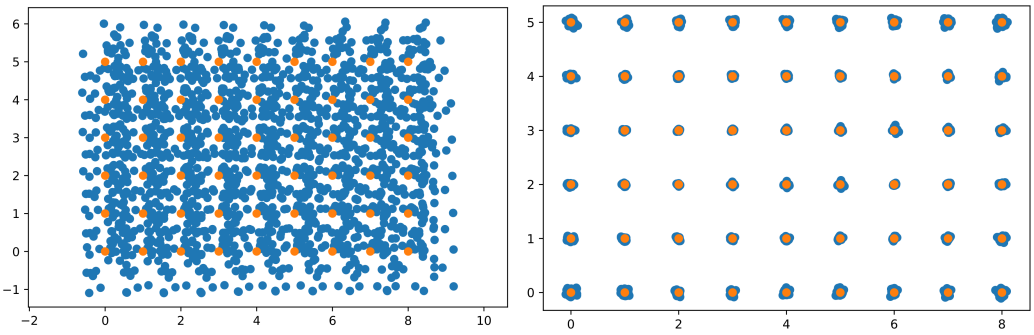
La tabla 6.1 compara los valores reales (*ground truth*) de los parámetros con las estimaciones resultantes de las muestras. La discrepancia aparece en la sexta

cifra significativa y se debe a las fluctuaciones estadísticas del ruido que se adicionó artificialmente al crear los datos sintéticos (de 1 píxel de desviación estándar). La covarianza de las muestras extraídas surge de la intensidad de ese ruido, si fuera más intenso, el posterior sería menos informativo (tendría mayor covarianza). Esto muestra que la esperanza de la probabilidad a posteriori es un buen estimador de los valores reales de los parámetros.

6.1.2. Calibración intrínseca con datos reales

Para estimar los parámetros intrínsecos de la cámara ojo de pez con datos reales se sigue el mismo procedimiento. Se usan las esquinas internas detectadas en las 33 imágenes de tablero en lugar de las que se fabricaron artificialmente suponiendo parámetros conocidos.

El muestreo por cadena de Markov requiere semillas donde iniciar la cadena, se las define haciendo una optimización no lineal para que estén más cerca de los valores de alta probabilidad. En la figura 6.3a los puntos naranja son las posiciones de los puntos de calibración en el plano mundo. Si se retro-proyecta con los parámetros extrínsecos retornados por `calibrateCamera` de OpenCV y los parámetros intrínsecos usados como *ground truth* en la sección anterior se obtienen los puntos en azul, difieren mucho de las posiciones reales de las esquinas del tablero. Se interpreta que los parámetros que producen esa retro-proyección están lejos de la zona de alta probabilidad que muestrea DEM. Para proveerle a DEM mejores semillas que acorten el proceso de *burn-in* se usan rutinas estándar de optimización no lineal de Scipy [77] que corrigen la retro-proyección acercándola a las posiciones mundo correctas. La función a optimizar es la probabilidad a posteriori de los parámetros (ecuación (5.4)). La retro-proyección con los valores de parámetro antes de optimizar se muestra en la figura 6.3a y en la figura 6.3b se muestra la retro-proyección con los parámetros optimizados, hay una mejora significativa. OpenCV estima los parámetros minimizando el error de proyección (sobre la imagen) por eso son malas estimaciones para retro-proyección. La clara mejora reduce drásticamente el tiempo de *burn-in* de DEM.



(a) (b)

Figura 6.3: (a) Retro-proyección con los parámetros obtenidos con las funciones de OpenCV. (b) Usando los parámetros luego de minimizar el error de retro-proyección.

Se extraen 500 cadenas cadenas de Markov de 2000 muestras cada una con DEM. Se supuso que la detección de esquinas internas tiene una incerteza de 1 píxel de desviación estándar. Las muestras obtenidas tienen una media y varianza que son:

$$\begin{aligned} \tilde{\mu}_\Gamma &= [816.45, 472.64, 795.19], \\ \tilde{\mathbf{C}}_\Gamma &= \begin{bmatrix} 0.628 & -0.051 & 0.016 \\ -0.051 & 0.640 & 0.367 \\ 0.016 & 0.367 & 3.746 \end{bmatrix}. \end{aligned} \tag{6.2}$$

Notar que la varianza es mucho mayor que la que se estimó con los datos simulados. La incerteza en los parámetros debe absorber el error del modelo estereográfico. También la matriz de covarianza tiene elementos no diagonales significativos.

6.1.3. Calibración extrínseca y predicción con datos sintéticos

Continuando lo hecho en la sección 6.1.1 donde se resolvió la calibración para una cámara simulada, ahora se simula la instalación de la cámara en una escena urbana para hacer la calibración extrínseca, evaluando el procedimiento para posicionamientos plausibles de la cámara.

El interés principal es evaluar la calibración en un conjunto de condiciones realistas en el contexto de monitoreo de vehículos y peatones en escenas urbanas, se definió:

- Dos alturas de la cámara sobre el suelo $\{7.5\text{m}, 15\text{m}\}$.

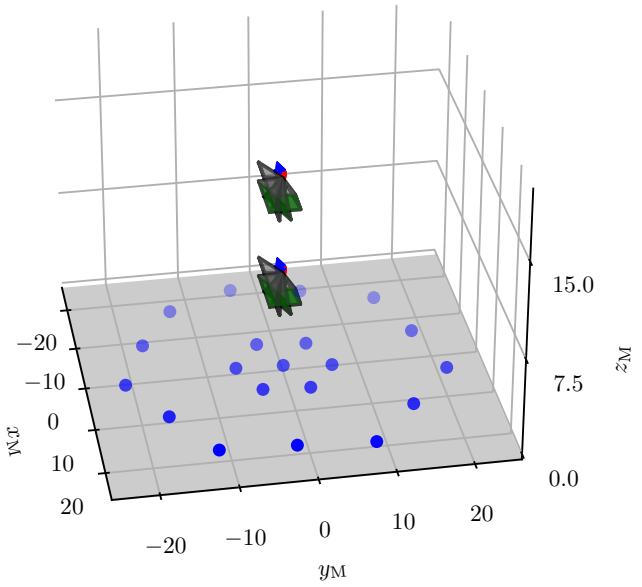


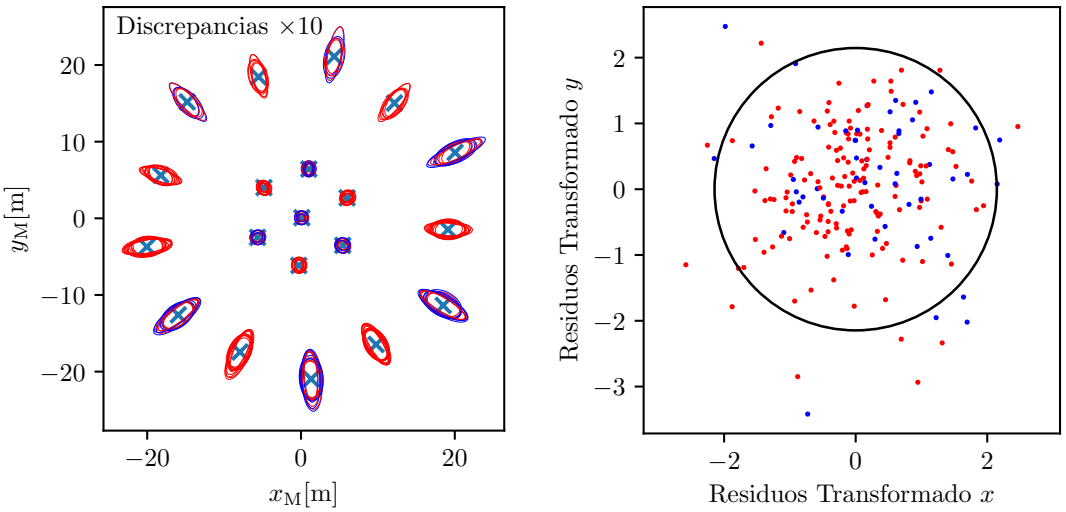
Figura 6.4: Los 20 puntos de calibración y seis casos de pose de cámara.

- El eje óptico (dirección de apunte) formando tres ángulos posibles $\{0°, 30°, 60°\}$ con respecto a la vertical.
- 20 puntos de calibración sobre el plano mundo $z_M = 0$ que se distribuyen homogéneamente dentro de un radio de 50m alrededor de la cámara.

En total abarcando seis posibles situaciones que se muestran en la figura 6.4. Los 20 puntos se proyectaron sobre la imagen que captura la cámara para simular detecciones de puntos de calibración, diez de ellos se usarán para hacer la calibración extrínseca y los restantes se usarán para evaluar cuantitativamente la calibración.

Ya hecha la estimación de la probabilidad a posteriori de los parámetros intrínsecos (media y varianza $\tilde{\mu}_\Gamma, \tilde{\mathbf{C}}_\Gamma$), ahora se aplica el procedimiento para muestrear la probabilidad a posteriori en el espacio 6-dimensional de los parámetros extrínsecos. En las seis situaciones de pose de la cámara se extraen 30 cadenas de Markov de 1000 muestras. Las seis medias ($\tilde{\mu}_\Theta$) y varianzas ($\tilde{\mathbf{C}}_\Theta$) obtenidas se usan para retroproyectar las detecciones sintéticas en la imagen con sus incertezas a sus posiciones mundo. La figura 6.5a muestra las elipses de varianza proyectadas en el marco de referencia mundo, el tamaño de las elipses y el error con respecto a las posiciones reales se ha magnificado $\times 10$ para que la disparidad sea visible. En rojo si el punto

fue usado para calibración y en azul si fue usado para testear la predicción. La incerteza retro-proyectada es menor cerca de la cámara y las elipses están menos estiradas porque esas regiones tienen un mejor factor de vista. De los puntos retro-proyectados más lejos de la cámara, la incerteza aumenta especialmente en la dirección radial por efecto de la perspectiva. Para evaluar visualmente el error de retro-proyección se procede como en la figura 6.2d transformando la diferencia entre la posición real y la elipse retro-proyectada tal que la elipse se convierta en un círculo. Se grafica su superposición en la figura 6.5b sobre el círculo de 90 % de probabilidad. El error de los puntos de entrenamiento (rojos) y los de testeo (azules) tiene una dispersión en acuerdo con el circulo de referencia que representa la incerteza de predicción.



(a) (b)

Figura 6.5: (a) Retro-proyección al plano mundo luego de calibrar. (b) Comparación de todas las retro-proyecciones con los puntos de calibración en una base donde las elipses predichas son círculos.

Para cuantificar el error en metros la tabla 6.2 reporta la raíz cuadrada del promedio de las desviaciones cuadráticas (root mean squared deviations) entre las posiciones reales en el mundo y las retro-proyecciones (solo los centros de las elipses, sin tomar en cuenta la varianza) de los puntos de calibración y de testeo. El error de predicción en los puntos de testeo siempre es mayor que el error en los puntos de calibración, ambos son del orden de 10cm.

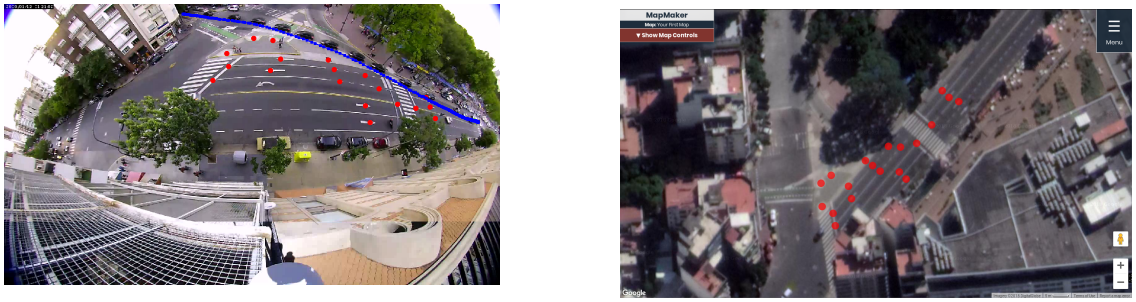
Solo calibrando con 10 puntos

ang[°]	h[m]	N_{Calib}	N_{Test}	Calib RMSD[m]	Test RMSD[m]
0	7.5	10	10	0.076	0.136
0	15	10	10	0.083	0.103
30	7.5	8	8	0.096	0.078
30	15	10	10	0.070	0.113
60	7.5	7	6	0.126	0.131
60	15	8	8	0.057	0.119

Cuadro 6.2: Para cada combinación de ángulo y altura se indican los puntos que estaban dentro del campo visual de la cámara para realizar la calibración y el testeo de predicción. Se reporta la desviación media cuadrática.

6.1.4. Calibración extrínseca y predicción con datos reales

Continuando la sección 6.1.2 se usan puntos de calibración para estimar la pose de la cámara en situaciones reales y geolocalizar la trayectoria de un vehículo.



(a) (b)

Figura 6.6: Calibración extrínseca con datos reales, en rojo los puntos de calibración y en azul la traza de un vehículo. (a) Imagen de cámara omnidireccional. (b) Imagen satelital.

La cámara se instaló a una altura de 15.7m sobre el suelo. El punto en el suelo que está directamente debajo de la cámara se marcó manualmente en el mapa y se uso como origen de coordenadas (0m, 0m) en el plano del suelo. Ésta es información de la pose de la cámara disponible a priori, antes de hacer ninguna calibración. la PDF a priori se definió gaussiana con media en esos valores y con desviación estándar de 0.3m en la altura y 2m para las coordenadas horizontales.

Se definen manualmente $n = 19$ puntos de calibración que consisten en pares correspondientes de coordenadas imagen-mundo. El terreno donde se realizó el ex-

perimento es horizontal y nivelado, tal que se puede suponer que los puntos están en $z_{\mathrm{M}} = 0$. Esto sirve también para convertir coordenadas mundo en sus diferentes representaciones (metros, ángulos de latitud-longitud, píxeles en una imagen satelital) usando un simple cambio de escala.

A las localizaciones en la imagen se les asignó una incerteza de 1 píxel de desviación estándar (fueron hechas a mano y se observó que la repetibilidad tenía un error de 1 píxel). La figura 6.6 muestra en rojo los puntos de calibración tanto en la imagen como en latitud-longitud (marcados sobre una imagen satelital). Se marcó a mano también la trayectoria de un vehículo a medida que atravesaba la escena, en azul en la figura 6.6a y 6.7. Los puntos marcados manualmente corresponden al punto medio inferior del lateral del vehículo, que está cerca del piso, para evitar error de paralajes.

Figura 6.7: Posiciones detectadas del vehículo.

Con la información a priori de la pose de la cámara y las probabilidades a posteriori de los parámetros intrínsecos obtenidos en la sección 6.1.2, DEM retornó 60 cadenas de Markov de 9500 muestras de los vectores de pose 6-dimensionales. La media y varianza de las muestras es

$$\begin{aligned} \tilde{\mu}_{\Theta} &= [2.7441, 1.1450, -0.1767, -3.2703, -2.2972, 17.6410], \\ \tilde{\mathbf{C}}_{\Theta} &= \begin{bmatrix} 4.1 & 0.1 & 1.0 & -14.4 & -56.9 & 52.2 \\ 0.1 & 2.7 & -0.6 & 47.7 & -1.0 & 26.3 \\ 1.0 & -0.6 & 1.8 & -42.5 & -14.3 & -3.5 \\ -14.4 & 47.7 & -42.5 & 1554.9 & 183.6 & 532.1 \\ -56.9 & -1.0 & -14.3 & 183.6 & 1027.9 & -921.9 \\ 52.2 & 26.3 & -3.5 & 532.1 & -921.9 & 1306.9 \end{bmatrix} \times 10^{-5}. \end{aligned} \tag{6.3}$$

La parte de $\tilde{\mu}_\Theta$ asociada a la rotación (los tres primeros elementos del vector) está en radianes, con una desviación estándar de $\sim 0.3°$ y la parte asociada a la traslación en metros con ~ 0.1m de desviación estándar. Pero $\tilde{\mu}_\Theta$ representa la posición del marco de referencia del mundo en coordenadas de la cámara, la posición de la cámara en el marco de referencia del mundo se calcula haciendo $-\mathbf{R}(\hat{r}_x, \hat{r}_y, \hat{r}_z)^\top \cdot [\hat{t}_x, \hat{t}_y, \hat{t}_y]^\top$. Que devuelve una altura de 17.0m y una desviación estándar de 0.14m, que está en el orden de magnitud de la altura real.

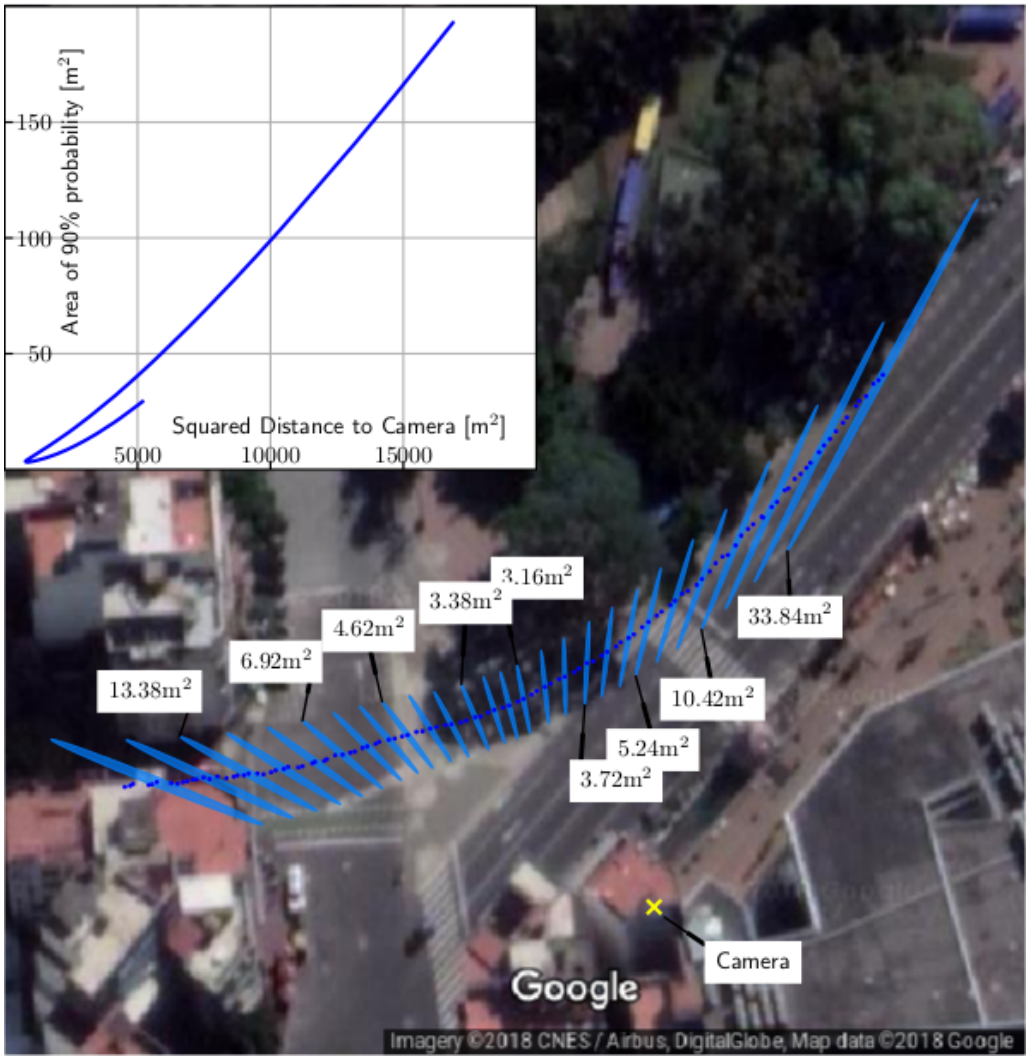


Figura 6.8: La retro-proyección de la traza del vehículo al plano mundo con incerteza (se grafica una elipse cada tantas).

La traza predicha del vehículo en coordenadas del mundo tiene una incerteza que combina las incertezas de los parámetros intrínsecos, extrínsecos y la detección en la imagen. En la figura 6.8 las elipses azules son las regiones predichas de 90 % de probabilidad, dibujadas una cada varios puntos retro-proyectados (puntos azules)

para simplificar la visualización.

El efecto de la perspectiva puede pensarse en dos componentes, el primero es la distancia neta a la cámara: la varianza predicha aumenta con la distancia al cuadrado. En el inserto de la figura 6.8 se muestra la dependencia empírica del área de la elipse de 90 % con la distancia al cuadrado. El área de 90 % de probabilidad mas pequeña es de 3.15m^2 y corresponde a cuando el auto está más cerca de la cámara.

La segunda componente es el factor de vista del punto retro-proyectado en el suelo con respecto a la cámara, que estira la elipse en dirección radial respecto al punto mas cercano a la cámara. Además está la distorsión óptica que estira radialmente pero respecto al eje óptico de la cámara. En este caso la elongación debido al factor de vista y a la distorsión óptica se combinan estirando las elipses en aproximadamente la misma dirección.

Capítulo 7

Discusión y conclusión

En la primera parte de esta tesis se estudiaron los modelos de formación de imagen, los métodos estándares de calibración, se formuló la retro-proyección y se aplicaron estos conocimientos para resolver algunos problemas concretos. Se destacan, por su importancia conceptual, en esta primera mitad el estudio de la distorsión *fisheye* para georreferenciar vehículos y el planteo del problema de manejo de incertezas. En la segunda parte se reformularon los métodos de calibración desde principios bayesianos y se aplicó a datos simulados y a datos reales. Para finalizar, se comentan las diferencias entre este método y otros propuestos en la literatura, y las rutinas que provee OpenCV.

7.1. Estudios de fotogrametría con cámara *fisheye*

Elección de cámaras omnidireccionales y modelo de distorsión

A pesar de las ventajas mencionadas en el capítulo 2, las cámaras omnidireccionales no se usan tan comúnmente en aplicaciones de videovigilancia urbana; la solución tradicional sigue siendo la instalación de múltiples cámaras (fijas o móviles) pero de campo visual reducido. La principal limitación de las cámaras omnidireccionales es la marcada distorsión óptica; pero ésta puede corregirse computacionalmente [81, 82], mientras que tiene la ventaja de poder observar en todo momento un hemisferio completo de la escena. La cámara ojo de pez es muy útil en aplicaciones relacionadas al transporte en las que se quiere analizar el movimiento de vehículos o

peatones en una región amplia, por ejemplo en intersecciones [83] como la mostrada en la figura 6.8.

De los modelos de formación de imagen se eligió el estereográfico porque, a la vez que describe cualitativamente el perfil de distorsión radial de la cámara, minimiza la cantidad de parámetros y éstos tienen una interpretación física. En el capítulo 3 se resolvió la retro-proyección que mapea desde coordenadas en la imagen (u, v) en píxeles a coordenadas en el espacio mundo en el plano del suelo $(x_{\mathrm{M}}, y_{\mathrm{M}})$ donde $z_{\mathrm{M}} = 0$.

Retro-proyección de *fisheye* y apunte de PTZ

Para la cámara PTZ se usa el modelo *pinhole* con una rotación extra asociada al movimiento de pan-tilt. Se resolvió el problema de apuntar la cámara de esta manera: conocida la posición de un objeto de interés en el espacio mundo X_{M} se estiman los ángulos de pan y tilt tal que la cámara PTZ ubique al objeto en su centro óptico.

En el capítulo 4, en el que se desarrollan aplicaciones de la retro proyección, se combinaron ambos resultados (retro-proyección *fisheye* y apunte PTZ) en un sistema *fisheye*+PTZ. Un operario identifica con un clic en la imagen *fisheye* un objeto de interés y el sistema automáticamente resuelve retro-proyectar esa coordenada clicada al plano del mundo y usarla para calcular el apunte de la PTZ. Se demostró el funcionamiento en una situación real, validando los modelos de proyección de imagen y retro-proyección. Se encontró que la calibración debe hacerse teniendo en cuenta si los parámetros estimados se usarán luego para proyectar o retro-proyectar. Es decir que si se minimiza el error de proyección la calibración retorna valores distintos que si se minimiza el error de retro-proyección. Este hallazgo llevó a replantear el manejo de incertezas, lo que desembocó en la formulación probabilística bayesiana de calibración y predicción del capítulo 5.

Retro-proyección a superficies parametrizadas

Es fácil extender el método para retro-proyectar hacia superficies curvas. Sean q_1, q_2 los parámetros de la superficie donde se quiere localizar los objetos. La retro-

proyección solo difiere de la forma propuesta antes en la solución de las ecuaciones de colinealidad (ecuación 3.1). Ahora toman la forma

$$\begin{bmatrix} x_{\mathrm{M}}(q_1, q_2) \\ y_{\mathrm{M}}(q_1, q_2) \\ z_{\mathrm{M}}(q_1, q_2) \end{bmatrix} = z_{\mathrm{C}}\ X'_{\mathrm{d}} - \mathrm{T}'.$$

Las incógnitas del sistema son $(z_{\mathrm{C}}, q_1, q_2)$. Para las formas paramétricas de superficies esféricas y cilíndricas es posible hallar la solución analíticamente[1].

Esto permitiría fácilmente lidiar con casos donde el terreno no sea un plano horizontal, incluso teniendo una curvatura pronunciada.

Ajuste de las trayectorias retro-proyectadas y cálculo de velocidad

Se exploró primero aplicar la retro-proyección para estimar la velocidad de un vehículo usando un manejo de incertezas simple. Para ello se detectaron vehículos en un video *fisheye* combinando una sustracción de fondo y seguimiento de puntos de interés y se retro-proyectaron sus coordenadas al plano mundo. Haciendo validación cruzada se encontró que las trazas debían ajustarse con polinomios de orden dos. Los ajustes de polinomio retornaron los coeficientes de la parábola que mejor ajusta a la traza y además una covarianza de los coeficientes. Con esos parámetros y sus covarianzas se calculó la velocidad como la derivada de los polinomios ajustados respecto al tiempo. La velocidad se obtuvo también con su covarianza usando la fórmula cerrada de propagación de incerteza para la combinación lineal de variables aleatorias gaussianas.

Se ajustaron los polinomios offline por cuadrados mínimos usando todos los datos de la trayectoria y online por cuadrados mínimos recursivos.

Además de polinomios, se exploraron otros modelos de trayectoria: modelos simples de la cinemática de vehículos, como el robot móvil diferencial o de la bicicleta [82, 86]. La ventaja de estos modelos en comparación con un polinomio es que capturan de manera más adecuada y realista la forma cualitativa de las trazas. La desventaja es que introducen variables de control cinemático que deben ser estimadas, por ejemplo en el caso de la bicicleta hay que estimar el ángulo de la rueda delantera

[1] Resultados no incluidos en esta tesis, forman parte de [84, 85].

y la velocidad de la rueda trasera, equivalentes al giro del volante y la posición del pedal acelerador en un automóvil. Para ello habría que parametrizar la dependencia de dichas variables en el tiempo. Quedaron claras dos opciones: proponer un modelo paramétrico de las mismas o ajustar una forma genérica como polinomios. Proponer un modelo paramétrico sería poco viable porque son variables (giro del volante y posición del pedal acelerador) que difícilmente tengan un comportamiento modelable. Por otro lado, se observó que las trayectorias de los vehículos no eran de gran complejidad y que en general trazaban una curva bastante suave. Entonces resultó preferible evitar el enfoque cinemático, porque agregaba una capa de complejidad analítica adicional que no sería significativamente mejor que un simple ajuste de polinomio a la trayectoria. Otra posibilidad sería modelar la trayectoria por trozos, combinando secciones rectas y secciones curvas, si el vehículo realizara una maniobra de giro.

Cualquiera sea la forma en que se decida modelar la traza de los vehículos, tiene que permitir una estimación con incerteza de sus parámetros. Pensando en una aplicación en tiempo real, es más adecuado el método online porque es causal. Si fuera necesario, puede usarse el Filtro de Kalman Extendido como método de estimación online que permite usar un modelo de trayectoria no lineal.

Probabilidad de infracción de velocidad

Hasta aquí, se logró una estimación de la velocidad del vehículo tiempo a tiempo, con una medida de incerteza que surge principalmente de los errores de detección de los puntos de interés. Finalmente, se mostró cómo aplicar esa estimación de velocidad con incerteza a un proceso de toma de decisión sobre si hubo o no infracción de velocidad. Para ello se calculó la probabilidad de que el módulo de la velocidad supere un valor umbral por integración numérica. Este cálculo mostró que, partiendo de una estimación de posición, no hay dificultad en calcular la probabilidad de infracción. El paso siguiente sería tomar una decisión discreta 'hay infracción' ó 'no hay infracción', pero este proceso escapa a los objetivos de este trabajo y se considera que es un problema para el que existe un gran abanico de técnicas y abundante material bibliográfico [87, 88, 89].

Detección de vehículos con redes neuronales convolutivas

En cuanto a la detección de vehículos, se considera que lo mejor es usar una red neuronal convolutiva entrenada apropiadamente. Si bien la segmentación frente-fondo y el seguimiento de puntos de interés permiten detectar vehículos en un video de tránsito, su funcionamiento es bastante limitado. Sólo sirve si las cámaras están fijas, además las sombras de los vehículos en un día soleado son detectadas también como objetos en movimiento, y también falla cuando un vehículo ocluye mínimamente la visión de otro. Se evaluó el rendimiento de Yolo-v3 (entrenada en el dataset COCO) en un dataset de imágenes de monitoreo de tránsito MIO-TCD. Los resultados son alentadores y tiene un error de localización de 8.1 píxeles (es el radio de la región de confianza del 90 %). A su vez, evaluando en un dataset propio de imágenes *fisheye* se obtiene un error de localización un poco mayor, 13.0 píxeles. La conclusión de la prueba fue que para detectar y localizar vehículos de manera confiable en un video de monitoreo de tránsito es necesario usar una red neuronal especialmente diseñada para localización y entrenada en un dataset de monitoreo de tránsito.

Las componentes de un sistema de monitoreo por visión

La figura 4.13 ilustra las componentes que tendría un sistema de visión monocular de monitoreo de tránsito cuyo objetivo es detectar infracciones basadas en la localización de los vehículos. Cada componente corresponde a un enunciado de lo aprendido en la primera mitad de la tesis:

- Es necesario hacer las calibraciones intrínseca y extrínseca para obtener los parámetros de la cámara. Se realizan una sola vez y resultan en los parámetros que se necesita para retro-proyectar.
- La localización de los vehículos en la imagen tiene asociada una incerteza. Puede hacerse en tiempo real sobre un video de tránsito con una red neuronal convolutiva entrenada para tal fin.
- La retro-proyección al plano mundo debe transformar la información de las variables anteriores a coordenadas mundo.

- En base a la posición en el plano mundo y su incerteza se calcula la probabilidad de que esté cometiendo una infracción (por ejemplo, superar la velocidad máxima permitida).
- sobre esa probabilidad se fundamenta (se asienta) una toma de decisión.

La detección de vehículos con redes neuronales, el cálculo de probabilidad de infracción y la toma de decisión son áreas que ya se pueden considerar resueltas, en términos generales. Lo que no está resuelto es el método de calibración de cámaras *fisheye* y de retro-proyección *que considere incertezas apropiadamente en todas las variables involucradas.* A partir de esta conclusión intermedia, se desarrolló y formuló un andamiaje general de tratamiento de incertezas con un enfoque bayesiano.

7.2. Abordaje bayesiano a la calibración de cámara

Ventajas del enfoque bayesiano

Queda claro que la calibración de la cámara es una parte crítica de todo sistema de fotogrametría. El método bayesiano permite formular *ab initio*, en términos probabilísticos, el problema de calibración y predicción, la incorporación de información a priori acerca de la cámara y su posicionamiento. Sundareswara and Schrater [90] demostraron que la predicción bayesiana a posteriori es menos susceptible a fluctuaciones estadísticas que una simple estimación puntual por maximización de la verosimilitud. El enfoque bayesiano también provee una manera natural de comparar modelos para elegir el más adecuado o, incluso, de combinarlos para una predicción unificada [91]. Esto sería aplicable tanto a los modelos de distorsión que caracterizan la cámara como a los modelos para ajustar las trazas de los vehículos.

Calibración bayesiana

El método propuesto en el capítulo 5 consiste en dos pasos de calibración y un paso de predicción de posición que es computacionalmente eficiente. El primer paso de calibración estima la PDF a posteriori de los parámetros de distorsión

óptica en laboratorio. El segundo paso estima la PDF a posteriori de los parámetros extrínsecos de la cámara. El abordaje bayesiano permite introducir información a priori sobre la posición de la cámara obtenida al momento de instalarla. Si hubiera pocos puntos de calibración entonces la información a priori ayudaría a disminuir la incerteza de calibración y a eliminar posibles ambigüedades típicas de algunos patrones de calibración [92]. El paso de predicción estima la PDF de la posición en coordenadas mundo condicionando en la detección de las coordenadas imagen de un vehículo y las PDF a posteriori de los parámetros.

La distribución a posteriori de los parámetros dados los datos se estimó con Differential Evolution Metropolis. Fue el método elegido porque no necesita el gradiente de la probabilidad a posteriori y porque funciona adecuadamente en espacios de dimensión alta. Se obtuvo una población de muestras acampanada y unimodal, es decir que es razonable tratar las PDF de las variables involucradas como gaussianas. Esta observación justificaría reemplazar DEM por un método más rápido de optimización (no lineal) de la probabilidad a posteriori. Es decir, en lugar de generar una población de muestras que luego se ajustarían por gaussianas se buscaría el punto de máxima probabilidad a posteriori (tomándoselo como la media de la distribución) y con la aproximación de Laplace se estimaría la varianza de la distribución, disminuyendo drásticamente el costo computacional [47].

Comparación con Criminisi et al. [46]

Parte importante de lo propuesto en esta tesis está inspirado en el trabajo de Criminisi et al. [46]. En él, se usa información de la geometría de la escena disponible en la imagen (puntos y líneas de fuga) para: medir la distancia entre planos paralelos, calcular áreas y longitudes, y determinar la posición de la cámara. El método acompaña cada medición con una medida de su incerteza. Se aplica el método para reconstrucción 3D, el cálculo de las dimensiones de muebles y edificios, y la determinación de la altura de personas. El área de aplicación es en análisis forense y modelado 3D.

Al igual que en esta tesis, Criminisi et al. [46] trabajan con escenas que contengan planos sobre los que se quiere realizar mediciones. Analizan en profundidad la

incerteza de las medidas calculadas en el plano, validan la propagación lineal comparándola con una predicción MC. Describen cómo los errores de las medidas en la imagen se propagan a las mediciones 3D y computan intervalos de confianza.

Lo que diferencia a Criminisi et al. [46] respecto a la presente tesis es:

- Ellos suponen imágenes obtenidas por proyección de perspectiva, no tratan distorsión no lineal.
- La información para calibración es la línea de fuga de un plano de referencia y un punto de fuga de la dirección no paralela al plano (aquí se usan puntos fiduciarios en el plano).
- No se necesita tener hecha la calibración intrínseca (aquí proponemos que la calibración sea parte del método porque es necesaria);
- pero sólo se obtienen mediciones a salvo de un factor de escala, excepto que se provea una referencia de tamaño en la imagen (aquí desde un principio se plantea la parametrización y la predicción con el objetivo de obtener medidas físicas en la escala correcta).
- La geometría se parametriza en términos de puntos y líneas de fuga (aquí se describen en términos iguales a los de OpenCV, que es un estándar: el vector de Rodrigues y el vector de traslación).
- Al calibrar no se incorpora información aparte de los datos de calibración (aquí se usa la probabilidad a priori sobre los parámetros).

Criminisi et al. [46] hacen un tratamiento completo de la incerteza y usan un método computacionalmente eficiente basado en parametrización de las distribuciones de probabilidad y propagación lineal de incerteza. Pero no consideran la distorsión óptica no lineal, su descripción de la geometría no es estándar y no aprovechan el aporte de información de la probabilidad a priori.

Comparación con Sundareswara and Schrater [90]

En el trabajo de Sundareswara and Schrater [90] se propone un método bayesiano de calibración y se aplica a un problema de reconstrucción 3D a partir de movimiento. Su objetivo es encontrar la escena 3D más probable que dio origen a las imágenes capturadas. La calibración calcula explícitamente la PDF a posteriori de los parámetros mediante métodos de muestreo similares a DEM. Luego, la reconstrucción 3D se hace marginalizando sobre la distribución de parámetros y se muestra que la marginalización mejora la precisión de la reconstrucción. Al igual que en esta tesis, aprovechan la posibilidad de introducir priors en los parámetros cuando se tiene información previa a la calibración; caso contrario usan priors uniformes.

Esta tesis sigue en términos generales ideas muy similares al trabajo de Sundareswara and Schrater [90] pero con las siguientes diferencias:

- Ellos usan un modelo *pinhole*, no tratan distorsiones no lineales (aquí se considera una fuerte distorsión radial).
- Hacen la calibración en un solo paso, a partir de múltiples observaciones del objeto de interés desde varios ángulos (aquí se usan dos pasos de calibración y se supone una única observación monocular de la escena).
- Estiman la probabilidad a posteriori de los parámetros y de la reconstrucción de la escena al mismo tiempo (aquí la calibración intrínseca debe hacerse previa a la instalación de la cámara y luego se hace la calibración extrínseca).
- El resultado de la calibración es una población de muestras de parámetros (aquí la calibración implica estimar los parámetros de media y varianza).
- Al hacer la reconstrucción 3D con la población de parámetros, la marginalización se hace promediando la población de reconstrucciones 3D (aquí la predicción es una propagación lineal que automáticamente incorpora la marginalización).
- No plantean un cálculo computacionalmente eficiente usando linealización y una descripción paramétrica de las distribuciones de probabilidad.

Sundareswara and Schrater [90] usan el enfoque bayesiano pero su problema es de reconstrucción 3D a partir del movimiento de la cámara (no aplicable a cámaras de monitoreo) y usa una descripción computacionalmente costosa en base a una población de muestras que deben usarse para generar una población de reconstrucciones 3D.

Posibilidad de extender la propagación de incerteza con más términos de Taylor y momentos de orden superior de la PDF

La propagación de incerteza supone una expansión en Taylor a primer orden (los Jacobianos usados en la propagación se calcularon con la regla de la cadena) y solo trata hasta el segundo momento (varianza) de la PDF. Si se quisiera, la propagación de incerteza podría hacerse más precisa tomando en cuenta momentos de orden superior de la PDF y propagándolos a través de términos de Taylor de orden superior. Mekid and Vaja [93] resuelven la propagación de incerteza a través de una función no lineal $\mathbb{R}^2 \rightarrow \mathbb{R}^1$. La PDF de la variable aleatoria se trata hasta su cuarto momento (además de la varianza se incluye *skewness* y *kurtosis*). La función no lineal se expande como serie de Taylor incluyendo hasta el tercer término para el caso de variables 2D. La principal dificultad radicaría en calcular las derivadas segunda y tercera de la función de retro-proyección pero podría resolverse usando algoritmos de derivación automática que hay disponibles en lenguajes de programación de alto nivel [94].

La incerteza en la posición retro-proyectada puede usarse para calcular la velocidad con una variedad de métodos

En la sección 4.2.1 se ajustó un polinomio a las posiciones y su derivada se usó para calcular la velocidad. Ahora, con la cuantificación de incerteza de posición como una matriz de varianza se podría usar los mismos métodos extendidos a cuadrados mínimos ponderados o cuadrados mínimos recursivos ponderados. Las matrices de varianza de la posición son adecuadas para calcular la velocidad con incerteza también por: diferencias finitas, filtro Kalman, cuadrados mínimos recursivos, o algún método alternativo.

Considerar X_{M} y z_{M} como variables aleatorias con incerteza

Se supuso que las mediciones de calibración en el mundo X_{M} no tienen error y que están perfectamente ubicados en el plano del suelo $z_{\mathrm{M}} = 0$. Pero sería razonable suponer que las mediciones de X_{M} son imprecisas debido a error instrumental. También suponer que la hipótesis de z_{M} no se cumple perfectamente por dos motivos: Que los puntos de calibración se marcaron sobre objetos que en realidad están ligeramente fuera del plano del suelo por desniveles de la calle o vereda y que el suelo en realidad es una superficie curva que no se puede aproximar por un plano. No modelar estas fuentes de incerteza aumenta la incerteza a posteriori de los parámetros Θ (el error en el modelo sería "absorbido" allí). Y la propagación de incerteza, ecuación (5.3), retornaría una incerteza de geolocalización incorrecta. Expandir el modelo para tratar la incerteza en X_{M} y z_{M} completaría el tratamiento bayesiano del problema, no quedando ya ninguna variable no considerada como aleatoria.

Extender el tratamiento de incertezas a z_{M} es factible agregando un término más a la propagación de incerteza. Una posible línea de investigación futura es incluir incerteza en las mediciones de calibración de X_{M} ya que en la bibliografía siempre supone que dichas mediciones puntuales no tienen asociada una incerteza.

7.3. Resultados del abordaje bayesiano con datos reales y simulados

En el capítulo 6 se muestran resultados de aplicar el método propuesto a datos reales y simulados.

La propagación de incerteza considerando una PDF sólo hasta segundo momento y linealizando la retro-proyección es válida

La calibración simulada mostró que los parámetros intrínsecos se estimaron con buena precisión y que con la calibración extrínseca se pueden predecir posiciones georreferenciadas con una correcta cuantificación de incerteza (figura 6.5). Aún habiendo severas distorsiones ópticas se mostró que es una aproximación válida mediante simulación Monte Carlo (figura 6.2).

Precisión de calibración en datos reales

Al calibrar con datos reales los parámetros intrínsecos se estiman con una precisión relativa de 10^{-3} (ecuación (6.2)). El incremento de incerteza respecto al caso simulado puede deberse principalmente a que el modelo de distorsión no describe perfectamente la distorsión real y las imperfecciones en la óptica de la cámara no fueron capturadas por el modelo estereográfico. La calibración extrínseca retorna una estimación de la pose con una incerteza de menos de 1° para la orientación y 0.1m para la posición (ecuación (6.3)).

Comportamiento de la incerteza retro-proyectada

La predicción de la posición mundo de un vehículo se muestra en la figura 6.8 como elipses de 90 % de confianza. El área de las elipses es proporcional al cuadrado de la distancia a la cámara, que es el comportamiento esperado de la retro-proyección. Es decir, la incerteza predicha es mayor si se retro-proyecta lejos de la cámara y si el factor de vista es pequeño. Las elipses tienden a estirarse en dirección radial. Ambos efectos se ven más pronunciados cuando el punto de retro-proyección está en el plano del suelo lejos de la cámara.

Puede aumentarse la precisión principalmente tomando mediciones de calibración más precisas y mejorando el modelo de distorsión óptica. Adicionalmente se podría considerar la curvatura del suelo y la incerteza en z_{M} pero no se espera que esto tenga un efecto significativo. En definitiva, es muy difícil mantener una incerteza de predicción acotada para vehículos que están fuera de la zona central de visión de la cámara. Por más precisa que sea la calibración, la cámara *fisheye* observa una zona tan amplia (de horizonte a horizonte) que al alejarse el punto de proyección de la zona central, en algún momento la propagación de incerteza diverge. Es una consecuencia ineludible de la geometría del sistema.

7.4. Comparación del método propuesto con rutinas existentes de OpenCV

OpenCV (*Open Source Computer Vision Library*) es una librería de software de fuente abierta que incluye un conjunto bastante exhaustivo de algoritmos clásicos y de estado del arte de procesamiento de imágenes, visión artificial y machine learning [34]. Es la herramienta insignia de la comunidad de desarrolladores de sistemas de visión. Por este motivo, en este trabajo se tomó como referencia su proceso de calibración y los modelos de formación de imagen (excepto por el modelo de distorsión radial).

El tratamiento de las incertezas en OpenCV no es completo

OpenCV incluye rutinas de calibración de cámaras pero con un tratamiento incompleto de la incerteza. La función `calibrateCamera` estima los parámetros intrínsecos minimizando el error de proyección, es un estimador de cuadrados mínimos [95] siguiendo Zhang [37], Bouguet [36]. También calcula el Jacobiano de las coordenadas imagen con respecto a los parámetros, pero no con el propósito de propagación de incerteza sino para usarse durante la optimización de calibración. `calibrateCamera` retorna un vector de desviaciones estándar de los parámetros estimados, lo calcula mediante una símil propagación inversa de incerteza: multiplica la varianza del error de proyección con la inversa Moore-Penrose del Jacobiano. No se tratan los términos de no diagonales de la covarianza (incerteza correlacionada entre los parámetros), forzando que los parámetros tengan una incerteza descorrelacionada. En este trabajo la calibración resultante (ecuación (6.2) y (6.3)) tiene términos de varianza cruzada no despreciables indicando claramente que el método de OpenCV está descartando componentes significativas de incerteza. Además `calibrateCamera` no toma en cuenta la incerteza de los puntos usados para la calibración. La función `solvePnP` estima la posición y orientación de un objeto dadas correspondencias 3D-2D y `warpPerspective` puede convertir coordenadas imagen a coordenadas en el plano del suelo si se tiene la matriz de transformación. Ninguna de las dos funciones hace un tratamiento de la incerteza.

Aplicar la calibración bayesiana a los modelos de OpenCV

Se usó el modelo estereográfico en este trabajo porque es una descripción apropiada de la distorsión de la cámara usada [39]. Pero se deja resuelta la retro-proyección a coordenadas mundo para los modelos polinómico, cociente de polinomios y polinómico en el ángulo ('*fisheye*' según OpenCV) [16]. De este modo queda la posibilidad de aplicar el presente método de calibración y predicción bayesiana con dichos modelos. Sería necesario implementar el cálculo del Jacobiano de las funciones de distorsión radial correspondientes, que es factible gracias a librerías de manipulación simbólica. Gracias a la modularidad del sistema es fácil agregarlos a la librería de Python donde se implementó la retro-proyección con incerteza.

7.5. Conclusión Final

Este trabajo trata el tema de la cuantificación de incerteza de posición para aplicaciones de sistemas de visión de campo amplio. La cámara *fisheye* es ideal para observar una región urbana amplia pero se debe lidiar con la fuerte distorsión óptica de sus lentes. Se resolvió la retro-proyección de coordenadas imagen a coordenadas mundo para los modelos de distorsión de OpenCV y también el modelo estereográfico. Se decidió usar este último porque describe adecuadamente la distorsión de la cámara y tiene pocos parámetros, lo que facilita su manipulación. Los principios de inferencia bayesiana son generales y permitieron encontrar un procedimiento de calibración en dos etapas (intrínseca y extrínseca) y un método de predicción. Logrando así, usar la cámara de monitoreo como sensor de posición con una correcta cuantificación de la incerteza en la posición del vehículo.

Bibliografía

[1] Mashrur Chowdhury. *Data Analytics for Intelligent Transportation Systems*. Elsevier, Amsterdam, 2017. ISBN 9780128097151.

[2] Rodolfo I. Meneguette, Robson E. De Grande, and Antonio A. F. Loureiro. *Intelligent Transport System in Smart Cities*. Urban Computing. Springer International Publishing, 2018. ISBN 978-3-319-93331-3. doi: 10.1007/978-3-319-93332-0. URL https://www.ebook.de/de/product/33055190/rodolfo_i_meneguette_robson_e_de_grande_antonio_a_f_loureiro_intelligent_transport_system_in_smart_cities.html.

[3] Robert P Loce, Raja Bala, and Mohan Trivedi, editors. *Computer Vision and Imaging in Intelligent Transportation Systems*. Wiley, IEEE Press, 2017. ISBN 9781118971604. URL https://www.ebook.de/de/product/28557989/computer_vision_and_imaging_in_intelligent_transportation_systems.html.

[4] Wilfried Linder. *Digital Photogrammetry: A Practical Course*. Springer, 4ed. edition, 2016.

[5] Davide Scaramuzza. *Omnidirectional Camera*, pages 552–560. Springer, Boston, MA, 04 2014. ISBN 978-0-387-31439-6. doi: 10.1007/978-0-387-31439-6_488. URL https://doi.org/10.1007/978-0-387-31439-6_488.

[6] Eric-Jan Wagenmakers, Michael Lee, Tom Lodewyckx, and Geoffrey J. Iverson. Bayesian versus frequentist inference. In *Bayesian Evaluation of Informative Hypotheses*, pages 181–207. Springer New York, 2008. doi: 10.1007/978-0-387-09612-4_9.

[7] Jordi Vallverdú. *Bayesians Versus Frequentists: A Philosophical Debate on Statistical Reasoning*. SpringerBriefs in Statistics. Springer-Verlag Berlin Heidelberg, 1 edition, 2016. ISBN 978-3-662-48636-8,978-3-662-48638-2.

[8] Damián Oliva, Agustín Yabo, Lilián García, Sebastián I Arroyo, and Félix G Safar. Implementación de un sistema para la medición del flujo de tránsito y detección de embotellamientos en autopistas. In *Argentine Symposium on Artificial Intelligence (ASAI 2015)-JAIIO 44 (Rosario, 2015)*, 2015.

[9] Sebastian I. Arroyo, Felix Safar, and Damian Oliva. Georeferenced feature tracking in wide field images. In *2015 16th Workshop on Information Processing and Control, RPIC 2015*, pages 1–6, Cordoba, 2016. IEEE. ISBN 9781467384667. doi: 10.1109/RPIC.2015.7497125.

[10] Damián Ezequiel Stanganelli. Implementación de un sistema de seguimiento de objetos múltiples. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Director: Dr. Damián Oliva, Co-Director: Mg. Sebastián I. Arroyo, Universidad Nacional de Quilmes, 2016.

[11] Agustín Yabo, Sebastián I Arroyo, Félix G Safar, and Damián Oliva. Vehicle classification and speed estimation using computer vision techniques. In *XXV Congreso Argentino de Control Automático (AADECA 2016)(Buenos Aires, 2016)*, 2016.

[12] Sebastián I. Arroyo, Félix Safar, and Damián Oliva. Probabilidad de infracción de velocidad de vehículos utilizando visión artificial en cámaras de campo amplio. In *ARGENCON 2016, 3rd Biennial Congress of IEEE Argentina*, pages 1–6, Buenos Aires, 2016.

[13] Lilián García Rojas. Sistema de videovigilancia integrado por una cámara fisheye y una cámara pan-tilt-zoom. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Director: Mg. Sebastián I. Arroyo, Co-Director: Dr. Damián Oliva, Universidad Nacional de Quilmes, October 2018.

[14] Sebastián I. Arroyo, Ulises Bussi, Félix Safar, and Damián Oliva. A monocular wide-field vision system for geolocation with uncertainties in urban scenes.

Engineering Research Express, 2(2):025041, 06 2020. doi: 10.1088/2631-8695/ab9b36.

[15] Juan Manuel Castellano. Incerteza de localización de vehículos con redes neuronales en videos de tránsito. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Director: Mg. Sebastián I. Arroyo, Co-Director: Dr. Damián Oliva, Universidad Nacional de Quilmes, March 2019.

[16] Sebastián Arroyo, Lilian Garcia, Felix Safar, and Damian Oliva. Sistema de video-detección basado en cámaras fisheye y ptz. *IEEE Latin America Transactions*, 100(1e), February 2021. (Early Access).

[17] Consejo Nacional de Investigaciones Científicas y Técnicas. Becas internas de postgrado tipo I, September 2013.

[18] Consejo Nacional de Investigaciones Científicas y Técnicas. Beca interna doctoral para temas estratégicos. Resolución D Nº 3423, September 2014.

[19] Secretaría de Políticas Universitarias. Sistemas de Video Detección Vehicular con Visión de Campo Amplio, y su aplicación a los Sistemas Inteligentes de Transporte (SIT). 1ra. Convocatoria de Proyectos de Investigación Básica y Aplicada "Universidad y Transporte", 2014. Director: Mg. Félix Safar, Co-Director: Dr. Damián Oliva.

[20] Secretaria de Investigación Universidad Nacional de Quilmes. Programa: Estrategias de ingeniería en automatización, computación y procesos industriales apicados a la resolución de problemas tecnológicos; proyecto: Desarrollo de sistemas autónomos basados en visión. Archivo Público de Actos Resolutivos RR-1076-15 (#11084), May 2015. URL `https://apar.unq.edu.ar/archivo/detalle.php?idArchivo=11084`. Director de Programa: Safar, Félix Gustavo. Director de Proyecto: Oliva, Damián.

[21] Secretaria de Investigación Universidad Nacional de Quilmes. Programa: Estrategias de ingeniería en automatización, computación y procesos industriales apicados a la resolución de problemas tecnológicos; proyecto: Desarrollo de sistemas autónomos basados en visión. EXPTE 1406/15 (02/05/2017

- 30/04/2019), 2017. URL http://secretariadeinvestigacion.web.unq.edu.ar/programas-proyectos/. Director de Programa: Safar, Félix Gustavo. Director de Proyecto: Oliva, Damián.

[22] Waze Mobile Ltd.,. Waze app, 2006-2021. URL https://www.waze.com. All Rights Reserved.

[23] Benjamin Coifman and SeoungBum Kim. Speed estimation and length based vehicle classification from freeway single-loop detectors. *Transportation Research Part C: Emerging Technologies*, 17(4):349–364, 2009. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2009.01.004. URL https://www.sciencedirect.com/science/article/pii/S0968090X09000072.

[24] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer, 2011. ISBN 1848829345,9781848829343.

[25] D. Forsyth and J. Ponce. *Computer vision: a modern approach*. Always learning. Pearson Education, One Lake Street, Upper Saddle River, New Jersey 07458, 2012. ISBN 9780136085928.

[26] Rafael C. Gonzalez and Richard E. Woods. Digital image processing. *Publishing house of electronics industry*, 141(7), 2002.

[27] Sayanan Sivaraman and Mohan Manubhai Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, 2013. ISSN 15249050. doi: 10.1109/TITS.2013.2266661.

[28] Jun Wei Hsieh, Shih Hao Yu, Yung Sheng Chen, Wen Fong Hu, J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu. Automatic Traffic Surveillance System for Vehicle Tracking and Classification. *IEEE Intelligent Transportation Systems Magazine*, 7(2):175–187, 06 2006. ISSN 15580016. doi: 10.1109/TITS.2006.874722.

[29] Omnibond. Trafficvision, 2021. URL http://www.trafficvision.com.

[30] Cubic. Gridsmart technologies, inc. © copyright. Technical report, 10545 Hardin Valley RoadKnoxville, Tennessee 37932United States of America, 2020. URL http://gridsmart.com.

[31] Zi Yang and Lilian S.C. Pun-Cheng. Vehicle detection in intelligent transportation systems and its applications under varying environments: A review. *Image and Vision Computing*, 69:143–154, 01 2018. doi: 10.1016/j.imavis.2017.09.008. URL https://doi.org/10.1016/j.imavis.2017.09.008.

[32] L Sobel. Camera Models and Machine Perception. Technical report, Computer Science Department, Technion, 1972.

[33] John J. Craig. *Introduction to robotics : mechanics and control.* Pearson Higher Education, 3rd international ed. edition, 2014. ISBN 9781292040042,1292040041.

[34] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[35] T Schenk. Introduction to Photogrammetry. *Department of Civil and Environmental Engineering and Geodetic Science, The Ohio State University*, pages 79–95, 2005.

[36] Jean-Yves Bouguet. Camera calibration toolbox for matlab. *http://www.vision.caltech.edu/bouguetj/calib_doc/index.html*, 2004.

[37] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. ISSN 01628828. doi: 10.1109/34.888718.

[38] David Claus and Andrew W Fitzgibbon. A Rational Function Model for Fish-eye Lens Distortion. *Proc. {CVPR} 2005*, 1:0–6, 2005. ISSN 1063-6919. doi: 10.1109/CVPR.2005.43. URL https://www.robots.ox.ac.uk/~dclaus/publications/claus05rf_model.pdf.

[39] Damián Stanganelli, Damián Oliva, Martín Noblía, and Félix Safar. Calibración de una cámara fisheye comercial con el modelo unificado para la observación de objetos múltiples. In *2014 IEEE Biennial Congress of Argentina (ARGENCON)*, pages 147–152, 2014. doi: 10.1109/ARGENCON.2014.6868487.

[40] Xianghua Ying and Zhanyi Hu. Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model. In *Lecture Notes in Computer Science*, pages 442–455. Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-540-24670-1_34.

[41] Diogo Filipe Cabral de Sousa Leite. *Target Tracking with Pan-Tilt-Zoom Cameras*. PhD thesis, M. Sc. thesis, Instituto Superior Technico, 2011.

[42] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Texts in Statistical Science. Chapman and Hall/CRC, 3 edition, 2014. ISBN 1439840954, 9781439840955.

[43] Steven M Kay. *Fundamentals of statistical signal processing*. Prentice Hall PTR, 1993.

[44] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81 (2):155, 2009.

[45] Tong Ke and Stergios I. Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4618–4626, 2017. doi: 10.1109/CVPR.2017.491.

[46] Antonio Criminisi, Ian Reid, and Andrew Zisserman. A Plane Measuring Device. *Image and Vision Computing*, 17(8):625–634, 1999. ISSN 1098-6596. doi: 10.1016/S0262-8856(98)00183-8.

[47] Christopher M. Bishop. *Pattern recognition and machine learning*. Information science and statistics. Springer Science+Business Media, Berlin/Heidelberg, Germany, 2006. ISBN 9780387310732.

[48] John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55, 2016.

[49] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with microsoft kinect sensor: A review. *IEEE transactions on cybernetics*, 43(5):1318–1334, 2013.

[50] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19 (2):4–10, 2012.

[51] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco Madrid-Cuevas, and Rafael Medina-Carnicer. Generation of fiducial marker dictionaries using mixed integer linear programming. *Pattern Recognition*, 51, 10 2015. doi: 10.1016/j.patcog.2015.09.023.

[52] Francisco Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76, 06 2018. doi: 10.1016/j.imavis.2018.05.004.

[53] Roger A Horn and Charles R Johnson. *Matrix analysis*, pages 146–7. Cambridge University Press, 1999.

[54] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[55] Rita Cucchiara, Costantino Grana, Massimo Piccardi, and Andrea Prati. Detecting moving objects, ghosts, and shadows in video streams. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1337–1342, 2003.

[56] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3): 346–359, 2008. ISSN 10773142. doi: 10.1016/j.cviu.2007.09.014.

[57] Marius Muja and David G Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *International Conference on Computer Vision Theory and Applications (VISAPP '09)*, 2:1–10, 2009. ISSN 00301299.

[58] Eric W. Weisstein. "vandermonde matrix.", October 2021. URL `https://mathworld.wolfram.com/VandermondeMatrix.html`.

[59] Eiji Mizutani Jyh-Shing Roger Jang, Chuen-Tsai Sun. *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, 1997. ISBN 0132610663,9780132610667.

[60] Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4:40–79, 2010.

[61] Marco Taboga. *Lectures on probability and statistics*. https://www.statlect.com, 2010.

[62] Mykel J. Kochenderfer, Tim A. Wheeler, and Kyle H. Wray. *Algorithms for Decision Making*. Massachusetts Institute of Technology, 2022. URL `https://algorithmsbook.com/`. 2021-03-13 17:19:38-08:00, revision 1ee8a78.

[63] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *Cvpr*, jun 2016. ISSN 01689002. doi: 10.1109/CVPR.2016.91. URL `https://doi.org/10.1109/CVPR.2016.91`.

[64] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[65] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[66] Zhiming Luo, B Frederic, Carl Lemaire, Janusz Konrad, Shaozi Li, Akshaya Mishra, Andrew Achkar, Justin Eichel, Pierre-Marc Jodoin, and Others. Miotcd: A new benchmark dataset for vehicle classification and localization. *IEEE Transactions on Image Processing*, 2018.

[67] Andrew Gordon Wilson. The case for Bayesian deep learning. *arXiv preprint arXiv:2001.10995*, 2020. URL `https://cims.nyu.edu/~andrewgw/caseforbdl/`.

[68] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[69] Aaron Meurer, Christopher P Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K Moore, Sartaj Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, jan 2017. doi: https://doi.org/10.7717/peerj-cs.103.

[70] Wes McKinney. *Python for Data Analysis. Data Wrangling with Pandas, NumPy, and IPython*. O'Reilly, 2nd edition, 2017. ISBN 978-1-491-95766-0.

[71] PyMC3 Discourse. Using multivariate pdfs when defining the model's likelihood and sampling step method. https://discourse.pymc.io/t/using-multi-multivariate-pdfs-when-defining-the-models-likelihood-and-sampling-step-method/883, February 2018. Forum Post by user Sebalander.

[72] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.0, may 2016.

[73] Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(47):1593–1623, 2014. URL `http://jmlr.org/papers/v15/hoffman14a.html`.

[74] Cajo J F Ter Braak. A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: Easy Bayesian computing for real parameter spaces. *Statistics and Computing*, 16(3):239–249, 2006. ISSN 09603174. doi: 10.1007/s11222-006-8769-1.

[75] PyMC3 Discourse. Metropolis: Ideal balance between slow-mixing and high rejection rate? `https://discourse.pymc.io/t/metropolis-ideal-balance-between-slow-mixing-and-high-rejection-rate/1205`, May 2018. Forum Post by user Sebalander.

[76] Clive S. Fraser. Automatic Camera Calibration in Close Range Photogrammetry. *Photogrammetric Engineering & Remote Sensing*, 79(4):381–388, 2013. ISSN 00991112. doi: 10.14358/PERS.79.4.381.

[77] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckes-

ser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17: 261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[78] John D Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95, 2007.

[79] Spyder Project Contributors. Spyder-documentation. https://www.spyder-ide.org/, 2019.

[80] Antonio Criminisi, Ian Reid, and Andrew Zisserman. Single view metrology. *International Journal of Computer Vision*, 40(2):123–148, 2000. ISSN 09205691. doi: 10.1023/A:1026598000963.

[81] J S Chahl and M V Srinivasan. Reflective surfaces for panoramic imaging. *Applied Optics*, 36(31):8275–8285, 1997.

[82] Peter Corke. *Robotics, Vision and Control - Fundamental Algorithms in MATLAB*. Springer, 2011. ISBN 3540221085. doi: 10.1007/978-3-540-73958-6_2.

[83] Wei Wang, Tim Gee, Jeff Price, Hairong Qi, Wei Wang, Tim Gee, Jeff Price, and Hairong Qi. Real time multi-vehicle tracking and counting at intersections from a fisheye camera. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 17–24. IEEE, IEEE, 01 2015. doi: 10.1109/wacv.2015.10.

[84] Leonardo J. Turkovic. Automatización del control de cierre de latas de conservas mediante técnicas de visión artificial. Trabajo Final de la Carrera Ingeniería en Automatización y Control Industrial, Directora: Virginia Mazzone, Co-Director: Mg. Sebastián I. Arroyo, Universidad Nacional de Quilmes, 2021. Unpublished.

[85] Hernan Guimaraynz. Automatización del control de cierre de latas de conservas mediante técnicas de visión artificial. Informe Final de Beca EVC CIN (Convocatoria 2018), Director: Sebastián I. Arroyo, Co-Directora: María Carolina Reid, Universidad Nacional de Quilmes, Consejo Interuniversitario Nacional, 2020.

[86] Ulises Bussi, Virginia Mazzone, and Damián Oliva. Control strategies analysis using ekf applied to a mobile robot. In *2017 XVII Workshop on Information Processing and Control (RPIC)*, pages 1–6. IEEE, 2017.

[87] David Rios Insua, Fabrizio Ruggeri, Refik Soyer, and Simon Wilson. Advances in bayesian decision making in reliability. *European Journal of Operational Research*, 282(1):1–18, 2020. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2019.03.018. URL `https://www.sciencedirect.com/science/article/pii/S0377221719302577`.

[88] Konrad P. Körding and Daniel M. Wolpert. Bayesian decision theory in sensorimotor control. *Trends in Cognitive Sciences*, 10(7):319–326, 2006. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2006.05.003. URL `https://www.sciencedirect.com/science/article/pii/S1364661306001276`. Special issue: Probabilistic models of cognition.

[89] Jim Q Smith. *Bayesian decision analysis: principles and practice*. Cambridge University Press, November 2010. ISBN 9780521764544. doi: 10.1017/CBO9780511779237.

[90] Rashmi Sundareswara and Paul R Schrater. Bayesian modelling of camera calibration and reconstruction. In *Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05)*, pages 394–401. IEEE, 2005. doi: 10.1109/3dim.2005.24.

[91] David J. C. MacKay. Bayesian Interpolation. *Neural Computation*, 4(3):415–447, 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.415.

[92] Shen Cai and Zhuping Zang. A deformed chessboard pattern for automatic camera calibration. In *2013 International Conference on Advanced ICT and Education (ICAICTE-13)*. Atlantis Press, 2013. doi: 10.2991/icaicte.2013.117.

[93] S. Mekid and D. Vaja. Propagation of uncertainty: Expressions of second and third order uncertainty with third and fourth moments. *Measurement*, 41(6): 600–609, 07 2008. doi: 10.1016/j.measurement.2007.07.004.

[94] Jarrett Revels, Miles Lubin, and Theodore Papamarkou. Forward-mode automatic differentiation in julia. *arXiv preprint arXiv:1607.07892*, 2016.

[95] Sara A. van de Geer. *Least Squares Estimation*. American Cancer Society, Atlanta, Georgia, U.S., 2005. ISBN 9780470013199. doi: 10.1002/0470013192.bsa199.