UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

IMPLEMENTACIÓN DE UN SISTEMA DE MULTITARGET TRACKING CON UNA CÁMARA INFRAROJA DE ALTA RESOLUCION.

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICO

BENJAMIN IGNACIO THEDY CORNEJO

PROFESOR GUÍA:
Martin Adams

Profesor Co-Guia:
Leonardo Cament Riveros

Comisión:
Andres Caba Rutte

SANTIAGO DE CHILE
2024

Resumen: Implementación de un sistema de multitarget tracking con una cámara infraroja de alta resolucion.

El objetivo principal de este trabajo de título es implementar un sistema de seguimiento de múltiples objetos (MTT) utilizando una cámara infrarroja de alta resolución. Para esto se adaptó exitosamente un algoritmo de seguimiento de múltiples objetos del estado del arte (δ-GLMB) para su funcionamiento con una cámara térmica. Se evaluó el sistema propuesto mediante la comparación de las predicciones del algoritmo de seguimiento con la posición real de los objetos, obteniendo resultados satisfactorios en términos de precisión y eficacia. La evaluación del sistema mostró un desempeño prometedor en condiciones adversas, como la presencia de neblina. Como trabajo futuro, se propone la integración de una segunda cámara para mejorar la percepción de profundidad y la creación de un dataset con condiciones climáticas adversas para evaluar la robustez del sistema.

"... You're not happy; you want what you don't have and forget what you do have.

You're not consistent; you change your mind as often as the moon changes its shape.

If you're rich, you're poor, because, like a donkey carrying heavy gold on his back, you can only carry your riches in life. You will have to leave them behind when you die...

You're neither young nor old; you're always dreaming about the one that you're not.

When you're young, you're like an old man who has to beg for money from older folks. And when you're old, you no longer have the desire, agility, and good looks to enjoy your wealth.

So what's so good about the thing we call "life?" There's a thousand things to suffer from in life, yet it's death we fear..."

-William Shakespeare (Measure for Measure)

Agradecimientos

Agradezco fuertemente a mi familia Elsa, Jorge, Cristobal, Fernanda, Vicente, Alvaro, Amanda, Emilia, Marina, Arturo, Ricardo y Raul. A ustedes les debo todo lo que soy, espero que esta memoria pueda hacerle justicia a todo el cariño que me han entregado a lo largo de mi vida.

Gracias a mis amigos y a todos los que me acompañaron en este arduo camino, particularmente a Felipe, Martin, Eduardo, Almendra y Matias.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Identificación y formulación del problema

La detección y el seguimiento visual de objetos representan desafíos fundamentales en los campos de la visión por computadora, con una amplia gama de aplicaciones que abarcan desde la vigilancia inteligente hasta la conducción autonoma. En el ámbito del seguimiento de objetivos en cámaras térmicas infrarrojas (TIR), se presentan ventajas significativas, ya que no se ven afectadas por cambios en la iluminación y pueden realizar un seguimiento durante la noche, en días lluviosos, nubosos y otras condiciones climáticas extremas. Esto las convierte en una herramienta ampliamente utilizada en vehículos de conducción asistida, la vigilancia con vehículos aéreos no tripulados y otras situaciones similares. Sin embargo, el seguimiento de objetivos en cámaras TIR todavía enfrenta desafíos, como la oclusión y la deformación de los bordes de detección.

En los últimos años, se han desarrollado diversos métodos para abordar estos desafíos, especialmente en aplicaciones críticas como la prevención de colisiones en vehículos autónomos. La detección térmica, aprovechando la firma térmica distintiva de los seres vivos, ha demostrado ser especialmente útil en esta tarea. En este contexto, el seguimiento de multiples objetivos *multitarget* (MTT) en cámaras térmicas representa un área de investigación emocionante en el campo de la visión por computadora.

Por lo tanto, esta memoria se propone explorar y desarrollar un enfoque de seguimiento *multitarget* en cámaras térmicas, con el objetivo de contribuir al avance de la tecnología y al desarrollo de sistemas más seguros y eficientes en aplicaciones como la conducción autónoma.

Para lograr este objetivo, se llevaron a cabo tres etapas principales: la calibración de la cámara térmica, la implementación y entrenamiento de una red neuronal para la detección de personas en imágenes térmicas, y la implementación de algoritmos de seguimiento de objetos, comenzando con un algoritmo de seguimiento de un solo objetivo (STT) como base, y finalmente adaptando el algoritmo δ-GLMB como un enfoque de MTT específicamente diseñado para cámaras térmicas.

Se espera que este trabajo contribuya al avance de la tecnología y al desarrollo de sistemas más seguros y eficientes en el campo de la conducción autónoma y otras aplicaciones relacionadas.

1.2. Objetivos del trabajo de título

El objetivo general del presente trabajo de titulo es implementar un sistema de "seguimiento de múltiples objetos"(de ahora en adelante MTT por sus siglas en ingles *Multi Target Tracking*) utilizando una cámara infrarroja de alta resolución. El problema a resolver consiste en implementar algoritmos de seguimiento de múltiples objetos (*multitarget*) adecuados para imágenes térmicas, abordando desafíos como la oclusión y la deformación.

Para esto se seguirán los siguientes objetivos específicos.

- Calibrar la cámara termal a utilizar a lo largo de la presente memoria con el fin de poder integrar la información del mundo real a imágenes térmicas en 2D.
- Implementar y entrenar una red neuronal capaz de detectar los objetos de interés (en particular personas) presentes en una imagen termal.
- Implementar un algoritmo del estado del arte capaz de llevar a cabo el seguimiento de múltiples objetos detectados mediante la red neuronal mencionada anteriormente.
- Realizar una comparación entre las predicciones generadas por el algoritmo de seguimiento de múltiples objetos y la posición espacial de los objetos en el entorno real, con el objetivo de evaluar la precisión y efectividad del sistema propuesto.

Capítulo 2

Marco Teórico y Estado del Arte

2.1. Contexto

El *multitarget tracking* ("MTT" de ahora en adelante) es la tarea que se preocupa de "seguimiento de múltiples objetos", para esto se desarrollaron modelos basados en movimientos.

Entre los algoritmos de *tracking* mas "primigenios"se encuentran aquellos que utilizan:

- Kalman Filter [1]
- Particle Filter [2]

Las implementaciones de los filtros mencionados anteriormente (dentro del contexto de *target tracking*), asumen implícitamente que hay un solo objeto y siempre existe una medición presente. Sin embargo, en muchas aplicaciones, es necesario realizar el seguimiento simultáneo de varios objetos (MTT). Por lo tanto, se han desarrollado diversos enfoques para extender estos métodos de seguimiento de un solo objeto (STT) al seguimiento de múltiples objetos. En términos generales la adaptación de STT a MTT se lleva a cabo mediante el siguiente algoritmo.

Primero, se identifican los objetos a rastrear basados en un vídeo o muestra inicial de entrada. Debido a las detecciones perdidas y/o las alarmas falsas, el número de mediciones entregadas por el proceso de detección o segmentación generalmente no es igual al número de objetos rastreados. Por lo tanto, se requiere un algoritmo de asociación de datos que asigne las mediciones recibidas a los rastreadores individuales de objetos individuales disponibles en ese momento (este tópico es conocido como " *Data asociation*" y es uno de los mayores problemas presentes en el área del MTT. Posteriormente para los algoritmos tradicionales basados en movimiento se estima el estado de cada objeto rastreado. Finalmente, la lista de trayectorias se pasa a la etapa de clasificación y validación, que decide si un objeto realmente existe y determina la clase del objeto.

Los pasos del algoritmo para pasar de "STT" a "MTT" descritos anteriormente se pueden apreciar en la figura 4.3

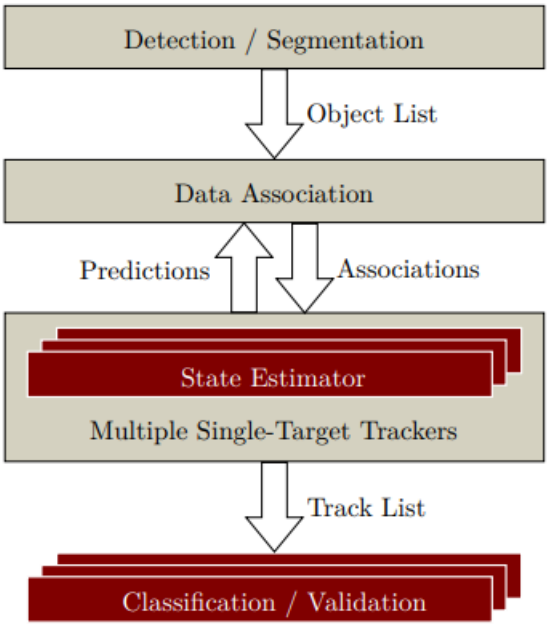

Figura 2.1: Algoritmo para pasar de "STT" a "MTT". Fuente: [3]

La mayoría de algoritmos de MTT basan su funcionamiento ampliamente en el algoritmo descrito anteriormente, incluso algoritmos del estado del arte como *δ-Generalised Labelled Multi-Bernoulli* , *Poisson multi-Bernoulli mixture*. Ambos algoritmos a pesar de diferir en su enfoque para calcular las probabilidades y asignar etiquetas a las hipótesis de trayectoria mantienen la estructura global del MTT explicada anteriormente. Es importante notar esto pues el enfoque que se propone usar en la presente memoria también toma en cuenta este algoritmo en el sentido de que usa algoritmos de "STT" y detección para llegar a crear un algoritmo exitoso de "MOT".

2.2. Calibración de la Cámara

2.2.1. Vectores Homogéneos y Matriz de Proyección

Dado que es necesario para el algoritmo de STT proyectar los puntos x,y,z predichos sobre la imagen (3D a 2D) para el tiempo K, es imperativo tener una manera de llevar a cabo este proceso, para esto en primera instancia se presentará el concepto de vector homogéneo para luego presentar los conceptos de matriz de parámetros intrínsecos y matriz de proyección.

2.2.1.1. Vector Homogéneo

Dado un vector $\mathbf{V} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ correspondiente a las coordenadas 3D de un punto en el mundo real.

Se define el vector homogéneo del vector $\mathbf{V}$ como $\mathbf{V}_{\text{homogéneo}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$ [4].

Esta definición surge de la necesidad de facilitar la proyección hacia el infinito o en general ponderar el vector por un escalar sin afectar el valor de estos mismos.

Dada la definición se puede transformar un vector homogéneo a su símil cartesiano dividiendo el vector homogéneo por su ponderador y eliminando este mismo de la siguiente forma:

$$\mathbf{V}_{\text{homogéneo}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}, \quad \mathbf{V}_{\text{no homogéneo}} = \begin{bmatrix} x/w \\ y/w \\ z/w \end{bmatrix}$$

2.2.1.2. Matriz de Parámetros Intrínsecos y Matriz de Proyección

Es necesario notar que para el desarrollo teórico de la presente memoria con respecto a matriz de proyección , proyecciones de puntos sobre el plano de imagen y matriz de la cámara se usaron mayoritariamente los capítulos 6 y 7 del libro "Multiple View Geometry in Computer Vision" por R. I. Hartley y A. Zisserman, [5] [6] respectivamente.

Dado que el modelo de cámara termal con el que se trabajó a lo largo de esta memoria corresponde al modelo de tipo *Pinhole* se definirá de ahora en adelante las propiedades para este tipo de cámaras.

El modelo *Pinhole* corresponde (para efectos de esta memoria) a un modelo de cámara con una apertura menor a 180º, es decir con un Campo de Visión (FOV de ahora en adelante por sus siglas en ingles *Field of View*) acotado.

Para este tipo de cámaras la matriz que proyecta un punto del espacio sobre el plano de imagen de la cámara esta dado por:

$$P = K[R|t] \tag{2.1}$$

Donde:

- P es la matriz de proyección 3×4.
- K es la matriz intrínseca 3×3.
- R|t es la matriz extrínseca compuesta por la matriz de rotación R y el vector de traslación t , esta matriz se encarga de :
  - Trasladar los puntos globales del origen a la posición del centro de la cámara.
  - Rotar los puntos globales conforme la disposición del plano de imagen de la cámara.

La matriz intrínseca (K) corresponde a:

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Donde:

- f_x y f_y son las longitudes focales en las direcciones x e y, respectivamente.
- s es un factor de escala de píxel, dado que los píxeles son cuadrados y no hay deformación en la escala en ninguna dirección se tiene s = 0 para el caso de estudio de la presente memoria.
- c_x y c_y son las coordenadas del punto principal (centro del sensor) en la imagen.

Finalmente, la proyección 3D a 2D se realiza multiplicando las coordenadas 3D del punto (X, Y, Z, 1) por la matriz de proyección P:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Donde:

u y v son las coordenadas 2D del punto proyectado en la imagen.

w es el término de escala mencionado en la subsección de "Vector Homogeneo".

El vector [X,Y,Z, 1]' se encuentra dentro del FOV de la cámara en cuestión.

2.3. Algoritmos de detección de objetos sobre imagenes

Los algoritmos de detección de objetos sobre imagenes pueden emplear enfoques estadísticos o redes neuronales para identificar patrones en señales y asignar etiquetas a lo largo del tiempo.

Los algoritmos de detección de objetos más avanzados se basan en redes neuronales convolucionales [7] (CNN por sus siglas en ingles *Convolutional Neural Networks*). Estas CNN son una arquitectura de red neuronal inspirada en la estructura visual del ser humano. La información de la imagen se procesa mediante filtros convolucionales, que detectan patrones visuales como bordes, texturas y formas. Estos filtros son seguidos por capas de agrupación, que reducen la dimensionalidad de las características extraídas, y capas completamente conectadas, que combinan estas características para realizar la clasificación de objetos.

Estas redes son entrenadas con extensos conjuntos de datos que contienen imágenes etiquetadas con información de clase. Esto permite que el algoritmo, una vez entrenado, pueda

realizar inferencias sobre nuevas imágenes, detectando objetos pertenecientes a las clases con las que fue previamente entrenado.

Para las labores de detección de objetos llevadas a cabo en la presente memoria se empleó un tipo particular de CNN llamada R-CNN [8] (por sus siglas en ingles *Region Based Convolutional Neural Network*). Este enfoque se caracteriza por utilizar cuadros delimitadores en las regiones de la imagen, los cuales son evaluados de forma independiente por redes convolucionales para cada Región de Interés, con el fin de clasificar múltiples regiones de la imagen en las clases propuestas. La arquitectura R-CNN fue diseñada con el propósito de resolver tareas de detección de imágenes de manera rápida y eficiente.

En la figura 2.2 se presenta el diagrama donde se ejemplifica el funcionamiento de una red R-CNN, en esta imagen se puede apreciar los siguientes pasos:

- 1. Imagen como entrada de la red.
- 2. Cuadros delimitadores sobre la imagen "extraen" la región de interés.
- 3. Se procesa cada region de interes independientemente.
- 4. Se clasifica el objeto dentro de la region de interés.

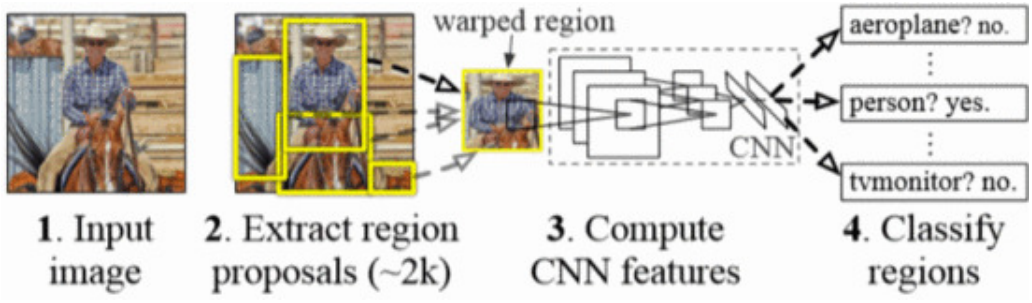

Figura 2.2: Ejemplo de detección de objeto sobre imagen usando R-CNN. Imagen obtenida de [8]

2.4. Filtro de Kalman

El filtro de Kalman [1] fue inventado como una técnica para el filtrado y la predicción en sistemas lineales. El filtro de Kalman implementa el cálculo de creencias para estados continuos. No es aplicable a espacios de estados discretos o híbridos [9].

En el filtro de Kalman, la estimación inicial del estado del objeto se asume dada por una distribución gaussiana:

$$N(x; \hat{x}, P) = \frac{1}{\sqrt{\det(2\pi P)}} \exp\left(-\frac{1}{2}(x-\hat{x})^T P^{-1}(x-\hat{x})\right) \tag{2.3}$$

donde $\hat{x}$ es el valor medio y P es la covarianza. Las ecuaciones de predicción están dadas por:

$$\begin{aligned} x_+ &= Fx + v \\ z_+ &= Hx^+ + w \end{aligned} \tag{2.4}$$

donde F es la matriz de transición , H es la matriz de medición, v es el ruido del proceso y w es el ruido de la medición, que se asumen no correlacionados. Si el ruido del proceso v sigue una distribución gaussiana con media cero, la transición de estado de una distribución gaussiana 2.3 produce el estado del objeto predicho y la covarianza del error de estimación:

$$\begin{aligned} \hat{x}_+ &= F\hat{x} \\ P_+ &= FPF^T + Q \end{aligned} \tag{2.5}$$

donde Q es la matriz de covarianza del ruido del proceso.

Dado que en general, un sensor no puede medir todas las componentes del estado predicho $\hat{x}_+$ se suele definir un ruido de medición gaussiano de media cero w y dado un modelo de medición lineal, el estado predicho $\hat{x}_+$ con covarianza P_+ induce una medición predicha distribuida de manera gaussiana:

$$z_+ = H\hat{x}^+ \tag{2.6}$$

con la covarianza correspondiente:

$$R_+ = HP_+H^T \tag{2.7}$$

Dado que el filtro de Kalman requiere que todas las señales sigan una distribución gaussiana, la incertidumbre de la medición de una medición obtenida z está representada por la matriz de covarianza R. En la etapa de actualización de la medición del filtro de Kalman, se requiere la innovación del estado del objeto, dada por el residuo de la medición z y la medición predicha z_+:

$$\gamma = z - z_+ \tag{2.8}$$

La covarianza de innovación correspondiente del residuo está dada por:

$$S = HP_+H^T + R \tag{2.9}$$

La ganancia K del filtro de Kalman modela la dependencia del estado posterior del objeto $\hat{x}$ en las incertidumbres del modelo de proceso y medición:

$$K = P_+ + H^T S^{-1} \tag{2.10}$$

$$\hat{x} = \hat{x}_+ + K(z - z_+) = \hat{x}_+ + K(\gamma) \tag{2.11}$$

Por un lado, un alto ruido de medición R en combinación con una pequeña covarianza de error de estimación predicha P_+ implica un valor pequeño de K y el estado posterior está dominado por el estado predicho $\hat{x}_+$. Por otro lado, el estado posterior está dominado por la medición actual z para un pequeño ruido de medición R y una alta covarianza de error de estimación predicha P_+. Finalmente, la covarianza del error de estimación posterior está dada por:

$$\begin{aligned} P &= P_+ - KSK^T \\ &= (I - KH)P_+(I - KH)^T + KRK^T \end{aligned} \tag{2.12}$$

2.5. Filtro de partículas

Como se menciono en la subseccion anterior el filtro de Kalman requiere que todas las señales sigan una distribución gaussiana. En consecuencia, el filtro de Kalman no puede manejar distribuciones arbitrarias. Es en este contexto (casos no paramétricos) que se vuelve útil la implementación del filtro de partículas, es decir, cuando las distribuciones de densidades, por ejemplo, no pueden caracterizarse mediante gaussianas. En líneas generales, el filtro de partículas aproxima funciones de densidad de probabilidad arbitrarias mediante un número finito de muestras, conocidas como partículas. Este enfoque es especialmente valioso para manejar distribuciones multimodales, las cuales son necesarias, por ejemplo, en la implementación de algoritmos de seguimiento de objetos múltiples (MTT). En tales casos, donde puede haber dos o más objetos en una escena, resulta crucial parametrizar distribuciones multimodales capaces de coincidir con las densidades de los objetos en cuestión.

En un filtro de partículas, la función de densidad de probabilidad (PDF) $p(x|z_{1:k})$ se aproxima mediante ν partículas $x^{(1)}, \ldots, x^{(\nu)} \sim p(x|z_{1:k})$, donde cada partícula $x^{(i)}$ es un "posible" valor de la variable aleatoria x. A cada partícula se le asocia un peso normalizado $w^{(i)} > 0$, de modo que $\sum_{i=1}^{\nu} w^{(i)} = 1$. En consecuencia, la aproximación de la PDF $p(x|z_{1:k})$ mediante ν partículas se da por la suma ponderada de las partículas:

$$p(x|z_{1:k}) \approx \sum_{i=1}^{\nu} w^{(i)} \delta_{x^{(i)}}(x) \tag{2.13}$$

donde la función delta de Kronecker se define como:

$$\delta_{x^{(i)}}(x) = \begin{cases} 1 & \text{si } x = x^{(i)}, \\ 0 & \text{en otro caso.} \end{cases} \tag{2.14}$$

Utilizando un gran número de partículas $\nu \to \infty$, es factible obtener una representación exacta de una PDF. La predicción de las partículas al tiempo de la siguiente medida se realiza muestreando las partículas predichas a partir de la densidad de transición f^+:

$$x_+^{(i)} \sim f^+(\cdot|x^{(i)}, z_{1:k}) \tag{2.15}$$

Los pesos de las partículas no se modifican durante el paso de predicción.

En el paso de actualización del filtro de partículas, la función de verosimilitud $p(z|x^{(i+)})$ debe evaluarse para cada una de las N partículas. El peso de las partículas se actualiza mediante:

$$w^{(i)} = \frac{p(z|x_t^{(i)}) \cdot w_t^{(i)}}{\sum_{j=1}^{N} p(z|x_t^{(j)}) \cdot w_t^{(j)}} \tag{2.16}$$

Donde el denominador asegura que los pesos actualizados sumen uno nuevamente.

Finalmente, la estimación del estado a *posteriori* del objeto se obtiene mediante la suma ponderada de las partículas:

$$x = \sum_{i=1}^{\nu} w^{(i)} x^{(i)} \tag{2.17}$$

Una desventaja del filtro partículas es la degeneración de las partículas, es decir, los pesos se concentran en un número muy pequeño de partículas. La degeneración se debe al hecho de que la varianza de los pesos aumenta durante el paso de actualización y nunca disminuye. Para evitar la degeneración, se eliminan las partículas con pesos muy pequeños, mientras que las partículas con un peso alto se seleccionan varias veces durante el llamado proceso de remuestreo. En particular el remuestreo implementado para el algoritmo de MTT basado en filtro de partículas en la presente memoria se lleva a cabo mediante las lineas de codigo:

Código 2.1: Resampling

```
% ---resampling
idx = randsample(length(w_update), filter.J_max, true, w_update);
w_update = ones(filter.J_max, 1) / filter.J_max;
x_update = x_update(:, idx);
```

Donde en la primera línea se hace una muestra aleatoria de los pesos de las partículas tomando tantos índices como numero de partículas hay, estos índices pueden ser repetidos y mientras mayor sea el peso de *w_update* mayor es la probabilidad de que se tome ese índice.

Luego se vuelven a inicializar los pesos con valor igual para todos.

Finalmente se genera el *x_update* para la nueva iteración , donde el *x_update* corresponde a los valores del mismo vector de estados (*x_update*) según los índices idx, así repitiendo las partículas con pesos elevados para la nueva iteración.

2.6. Random Finite Sets

En el contexto del seguimiento de múltiples objetos, una desventaja de los vectores aleatorios es la falta de representación del número incierto de objetos. Un RFS [10]

$$X = \{x^{(1)}, \ldots, x^{(n)}\} \tag{2.18}$$

consta de n vectores no ordenados con estados de objeto aleatorios $x^{(1)}, \ldots, x^{(n)}$, donde $n \geq 0$ es un número aleatorio, por lo tanto, el conjunto {a,b,c} es igual al conjunto {b,a,c}. En consecuencia, un RFS X representa naturalmente la incertidumbre sobre el número de objetos en un estado multi-objeto (los objetos pueden aparecer o desaparecer el campo de vision del sensor en cuestion). Además, el RFS

$$Z = \{z^{(1)}, \ldots, z^{(m)}\} \tag{2.19}$$

2.19 se utiliza convenientemente para describir el proceso de medición que devuelve un número aleatorio de mediciones cuyos valores $z^{(1)}, \ldots, z^{(m)}$ también son aleatorios. Aquí, la aleatoriedad en el número de medidas recibidas se debe a la posibilidad de falsas alarmas y

detecciones perdidas.

2.6.1. *Labeled-Random Finite Sets*

Para un *Labeled-Random Finite Set* [11] se considera un espacio de estado $\mathbb{X}$ y un espacio discreto $\mathbb{L}$. Sea Sea $L : \mathbb{X} \times \mathbb{L} \to \mathbb{L}$ la proyección definida por $L((x, \ell)) = \ell$. Luego, $L(x)$ se llama la etiqueta del punto $x \in \mathbb{X} \times \mathbb{L}$, y se dice que un subconjunto finito X de $\mathbb{X} \times \mathbb{L}$ tiene etiquetas distintas si y solo si X y sus etiquetas $L(X) = \{L(x) : x \in X\}$ tienen la misma cardinalidad. El indicador de etiquetas distintas se define como:

$$\Delta(X) = \delta_{|X|}(|L(X)|) \tag{2.20}$$

Entonces 2.20 toma valor 1 si cada objeto tiene un label diferente y toma valor 0 si es que no.

Mediante la implementación de la función anterior se define y asegura de que cada estado tenga una etiqueta única, esto permite identificar trayectorias individuales asignadas a estados x , lo que a su vez es crucial para la implementación de un algoritmo de MTT ya que, una trayectoria no es mas que una secuencia de estados bajo la misma etiqueta ℓ.

2.7. δ-*Generalized Labeled Multi-Bernoulli Multi-Target Tracking*

Dado que en un algoritmo de MTT no se sabe a priori cuantos objetos se cruzaran por el campo de visión del sensor es conveniente usar *Random Finite Sets* (RFS) , debido a que estos permiten trabajar con un numero de elementos pertenecientes un conjunto con cardinalidad variable, en desmedro del uso de matrices por ejemplo, cuyas dimensiones son constantes y deben ser definidas a priori lo que dificulta la representación de objetos variables y sus respectivos estados.

2.8. Filtro δ-GLMB

Un δ-GLMB por sus siglas en ingles δ-*Generalized Labeled Multi-Bernoulli* [12] es un *Labeled RFS* con estado de espacios $\mathbb{X}$ y estado de etiquetas $\mathbb{L}$ ($\mathbb{L}$ es discreto pues corresponde a las etiquetas asignadas a los estados).

Este conjunto distribuye dada la funcion:

$$\pi(X) = \Delta(X) \sum_{\substack{\xi \in \Xi \\ I \in F(L)}} \omega^{I,\xi} \delta_I[L(X)][p^{(\xi)}]^X \tag{2.21}$$

Dada la ecuación (2.21) se tiene:

- $\Delta(X)$ es 1 si las etiquetas en X son únicas para cada elemento del conjunto, y 0 de lo contrario
- Cada $\xi \in \Xi$ representa una historia de mapas de asociación, $\xi = (\theta_1 : k)$

- Cada $I \in F(L)$ representa un conjunto de etiquetas de objetos
- $w^{(I,\xi)} = w^{(\xi)}(I)$ es un peso asociado a la asignación específica de etiquetas I según ξ
- δ_I es la función delta de Kronecker para la partición en particular que comprueba la cardinalidad de las etiquetas de X con respecto a la cardinalidad de la partición I
- $[p^{(\xi)}]^X$ es la probabilidad de tener el conjunto de objetos X dado una una historia de mapas de asociación, $\xi = (\theta_1 : k)$ del conjunto Ξ

Por lo tanto la suma en la expresión 2.21 implica que se están sumando sobre todas las posibles "historias" de asociación y todos los posibles conjuntos de etiquetas, teniendo en cuenta sus respectivos pesos y funciones de probabilidad. Esto entrega una densidad de probabilidad de tener un conjunto de objetos con ciertas etiquetas a lo largo del tiempo.

Expuesta la función de densidad de un conjunto GLMB se tiene el algoritmo del filtro GLMB para cada hipótesis presente en la siguiente figura:

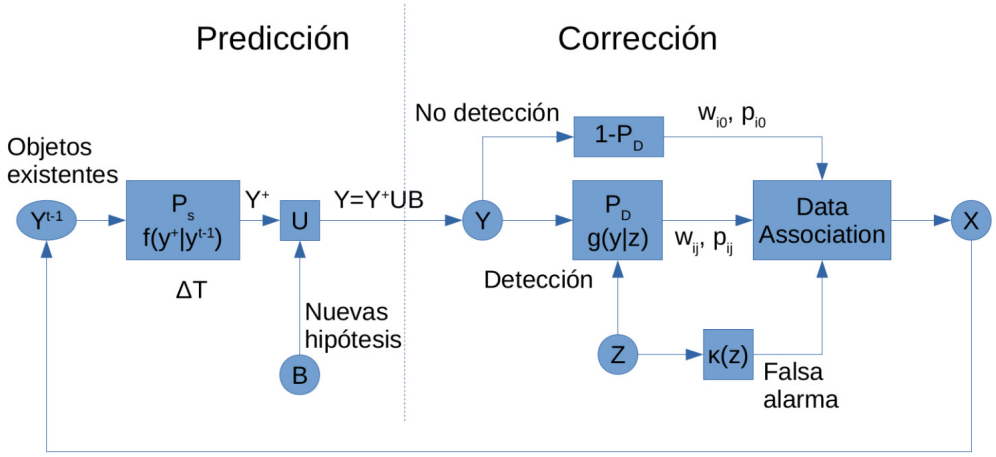

Figura 2.3: Diagrama Algoritmo Filtro δ-GLMB desde el punto de vista de la implementación

En el algoritmo expuesto en el diagrama presente en la figura 2.3 es necesario notar que:

$$Y = Y^+ \cup B = \{r_\ell, p_\ell\}_{\ell=\ell_1}^{\ell_N} \cup \{r_\ell, p_\ell\}_{\ell=\ell_{N+1}}^{\ell_{N+B}} \tag{2.22}$$

A demás se tiene para cada hipótesis las siguientes etapas del algoritmo :

2.8.1. Predicción

Probabilidad de supervivencia para tiempo t:

$$r_\ell = r_\ell^{t-1} \langle p_\ell^{t-1}(x), P_S(x) \rangle \tag{2.23}$$

Pero por definición $r_\ell^{t-1} = 1$ para una δ-GLMB, entonces:

$$r_\ell = \langle p_\ell^{t-1}(x), P_S(x) \rangle \tag{2.24}$$

En 2.24 para calcular la probabilidad de supervivencia de la hipotesis ℓ para t se pondera la probabilidad de existencia anterior dada por r_ℓ^{t-1} por el producto interno entre la densidad del objeto x y la probabilidad de supervivencia del objeto x, donde el producto interno$\langle$f, g$\rangle$ esta definido como: $\int$f(x)g(x)dx

A demás la densidad de la hipótesis ℓ para el objeto x esta dada por

$$p_\ell(x) = \frac{\int p_\ell^{t-1}(x^{t-1}) P_s(x^{t-1}) p(x|x^{t-1})\, dx^{t-1}}{\langle p_{(\ell)}^{t-1}(x), P_s(x) \rangle} \tag{2.25}$$

Finalmente para el nacimiento de nuevas hipótesis se tiene que estas nacen con el par probabilidad de {existencia, densidad} dado por:

$$\{r_\ell, p_\ell\}_{\ell=\ell_{N+1}}^{\ell_{N+B}} \tag{2.26}$$

Es necesario notar que los las etiquetas de nacimiento (*Birth*) se asignan de la forma (t,1)...(t,B) Con N correspondiente al número de hipótesis de una multi-bernulli en el tiempo anterior, B el numero de nuevas hipótesis para el tiempo actual y t el indice del tiempo actual.

2.8.2. Corrección

Luego en la etapa de corrección se tienen las probabilidades de existencia de la hipótesis en cuestión r_ℓ corresponden a 1.

Para las densidades se tiene:

- No detección:

$$p_{\ell 0}(x) = \frac{p_{\ell 0}(x)(1 - P_D(x))}{w_{\ell 0}} \tag{2.27}$$

  Donde P_D corresponde a la probabilidad de detección, $w_{\ell 0}$ corresponde al peso asignado a la no detección

- Detección:

$$p_{\ell j}(x) = \frac{p_\ell(x) P_D(x) g(z_j|x)}{w_{\ell j}} \tag{2.28}$$

Donde $p_{\ell j}(x)$ corresponde a la densidad (update) del objeto para la medición z_j, y $g(z_j|x)$ corresponde al *likelihood* de la medición z_j.

2.8.3. Pesos $w_{\ell,j}$

Los pesos usados en la sección de predicción asignados al par etiqueta (ℓ), detección (j) se presentan a continuación:

- *Miss-Detection*

$$w_{\ell 0} = \langle p_\ell(x), 1 - P_D(x) \rangle \tag{2.29}$$

- Detección:

$$w_{\ell j} = \langle p_\ell(x), P_D(x) g(z_j|x) \rangle \tag{2.30}$$

- *Cluter* (falsa alarma)

$$\kappa_j \approx \frac{\lambda_\kappa}{V} \tag{2.31}$$

Con κ_j numero esperado de falsas alarmas , V volumen del FOV

2.8.4. Otros Componentes del algoritmo

Dado que como resultado del algoritmo hasta el momento no se sabe de qué objeto proviene una medición, surge un desafío de asociación de datos. La consideración de todas las asociaciones posibles resulta impracticable en términos computacionales cuando se trata de grandes volúmenes de objetos y mediciones, debido al elevado costo computacional involucrado en cada iteración subsiguiente. Por lo tanto, aunque un RFS originalmente no aborda directamente la asociación de datos, ya que simplemente explora todas las combinaciones posibles, se recurre a métodos de "optimizacion y o muestreo" para calcular las asociaciones más relevantes.

Para llevar a cabo este "optimizacion y o muestreo" comúnmente hay dos ramas de algoritmos a elegir. En primer lugar, el Algoritmo de Murty [13],[14], que calcula las T mejores asociaciones, y el algoritmo de muestreo de Gibbs [15] que genera asociaciones basadas en su verosimilitud condicional, este algoritmo esta basado en la generación de muestras aleatorias y por lo tanto produce resultados no deterministas. Se atribuye un menor peso computacional al algoritmo de muestreo de Gibbs como recompensa a su *trade off* de resultados no deterministas [16].

Finalmente, en la extracción de estados del algoritmo de filtro GLMB de este informe se emplea el algoritmo MAP, cuyos pasos principales se describen de la siguiente manera:

El algoritmo genera una probabilidad de cardinalidad, asignando una probabilidad para cada número posible de etiquetas.

Además, para cada tiempo K, se obtiene la probabilidad asociada a cada conjunto de estados posibles, como se indica en la ecuación 2.21.

Posteriormente, en cada extracción de estados, se selecciona la cardinalidad de etiquetas con la mayor probabilidad y se extrae un número de estados equivalente a la cardinalidad determinada. Los estados extraídos representan los estados con la mayor probabilidad para el tiempo K.

Capítulo 3

Metodología

En la presente sección se presenta la metodología para llevar a cabo los experimentos necesarios para la implementación de un algoritmo de MTT usando una cámara infrarroja.

Es necesario mencionar que la implementación de los algoritmos STT Y MTT usaron como base el *toolbox* presente en: https://ba-tuong.vo-au.com/codes.html el que a su vez esta basado en las publicaciones [12], [16] y [17] principalmente.

Adicionalmente los códigos necesarios para llevar a cabo la implementación de los algoritmos expuestos en la presente memoria se encuentran disponibles en https://github.com/thedyyy/Tesis.

Finalmente en la figura 3.1 se presenta un diagrama de flujo de los hitos mas relevantes llevados a cabo en la presente memoria, los que posteriormente serán explorados en el presente capitulo

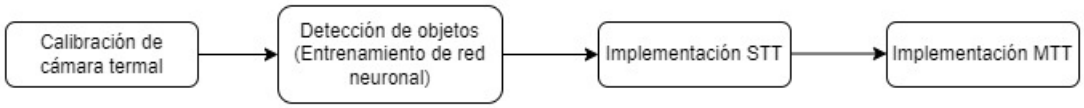

Figura 3.1: Flujo grama de la metodología

3.1. Dataset

Para la presente memoria los experimentos de detección fueron llevados a cabo usando un dataset creado por el Laboratorio de Visión Computacional de la Universidad de Chile el año 2018 donde se dispuso de:

- Cámara RGB
- Velodyne LiDAR
- Cámara Termal

En particular la cámara termal utilizada para la grabación del dataset es el modelo ”FLIR Lepton 3.5 , 160 x 120 57º HFOV with shutter”.

Figura 3.2: Disposición de los sensores montados sobre un vehículo eléctrico "Husky".

La base de la cámara termal se dispuso a aproximadamente 35 [cms] de la base del lidar Velodyne, lo que se traduce a que los valores de la matriz de traslación para la cámara termal están dados por:

$$[T_x, T_y, T_z] = [0, 0, 0.35] \tag{3.1}$$

Asumiendo que la base del Velodyne es el origen global, esto es útil pues se pueden usar los *Ground Truths* 3D (verdad base de las posiciones y características reales de los objetos en el entorno) marcados desde las posiciones de los objetos vistas a partir de la nube de puntos generada por el LIDAR sin necesitar llevar a cabo cambios sobre la matriz de proyección de la cámara termal.

En el conjunto de datos, se realizaron mediciones simultáneas utilizando varios sensores y se almacenaron en archivos rosbag (formato de archivo utilizado en ROS para capturar y almacenar datos generados por los nodos de ROS durante la ejecución de un sistema robótico) . En total, se obtuvieron 6 secuencias de datos, cada una representando una toma de datos conjunta con múltiples sensores. Para realizar pruebas específicas, se seleccionaron 2 de estas secuencias. Estas dos secuencias fueron elegidas debido a que presentan características valiosas, como la presencia de múltiples objetivos en movimiento, situaciones donde los objetivos son la única etiqueta presente en la imagen, y la introducción de perturbaciones visuales mediante el uso de una máquina de humo.

Este conjunto de pruebas es especialmente significativo para destacar la utilidad de la cámara térmica. La cámara térmica ha demostrado ser resiliente en la visualización de imágenes en condiciones adversas, como la presencia de humo o niebla. Estos escenarios desafiantes

subrayan la capacidad de la cámara térmica para proporcionar información valiosa incluso en entornos visuales difíciles, destacando su robustez y aplicabilidad en situaciones del mundo real adversas para otros tipos de sensores como camaras RGB o LIDARS.

3.2. Matriz de Proyección Cámara Termal

3.2.1. Matriz de parámetros intrínsecos

Para calcular la matriz de parámetros intrínsecos de la cámara térmica, también conocida como matriz de la cámara o matriz de calibración, se procedió a calentar una impresión de un tablero de ajedrez con dimensiones de 11 filas por 10 columnas, cada casilla con un lado de 20[mm]. Este calentamiento se realizó utilizando simultáneamente dos ampolletas incandescentes de 150[W], asegurándose de calentar el tablero de manera uniforme [18]. Durante este proceso, se grabó un rosbag del cual se extrajeron imágenes, como la que se muestra en la Figura 3.3

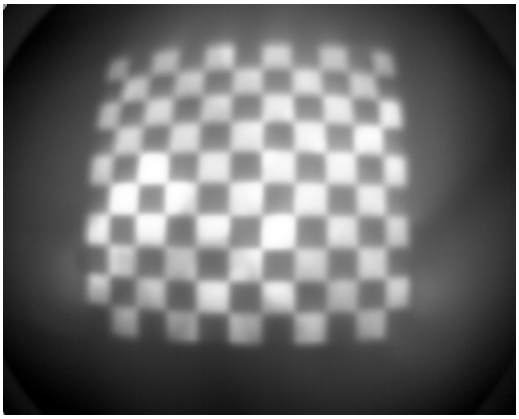

Figura 3.3: Imagen del patrón de tablero de ajedrez obtenida de la cámara termal en formato de 8bits

Una vez obtenidas suficientes imágenes se uso la aplicación de MATLAB 2018 "Camera Calibrator", esta aplicación recibe las imágenes de patrón de tablero de ajedrez , solicita el tamaño de las celdas del patrón del tablero de ajedrez como entrada y entrega como resultado los parámetros intrínsecos de la cámara.

El algoritmo implementado por la aplicación consta de :

- Se detectan las esquinas del tablero en 2D.
- Se generan el mismo número de esquinas para un tablero de ajedrez creado en el mundo 3D.
- Mediante algoritmos matemáticos se busca la matriz de parámetros intrínsecos que mejor proyecte los puntos 3D generados artificialmente sobre los puntos 2D detectado en la imagen. para esto se busca minimizar el error de distancia entre el punto proyectado y el punto de la esquina encontrada mas cercana (la relacion de dos puntos es unica y no se puede usar un punto proyectado para 2 esquinas encontradas)

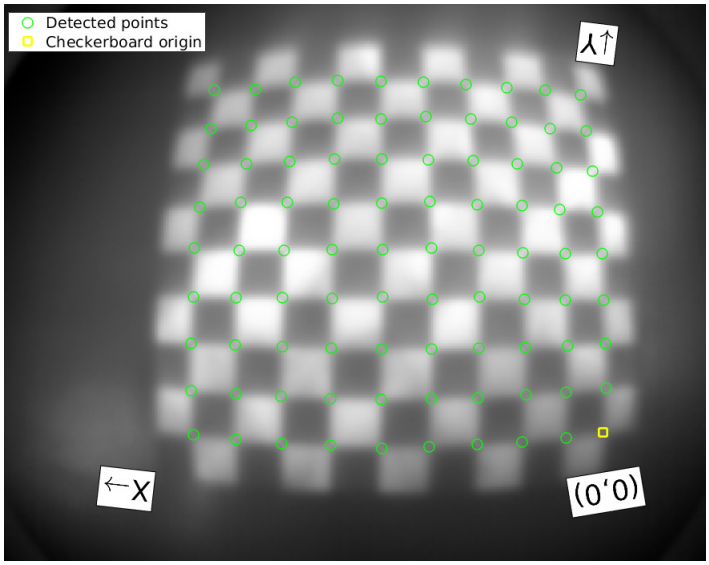

Figura 3.4: Esquinas detectadas sobre el tablero de ajedrez

El proceso de evaluar las imágenes en el algoritmo se llevó a cabo hasta llegar a un error de reproyeccion inferior a 0.2[pixeles] en promedio para las imágenes procesadas.

El resultado obtenido del algoritmo para la región de interés con alfa=0 (i.e. al desdistorsionar la imagen la región de interés se limita solamente a los píxeles de la imagen original y no al relleno) esta dado por la matriz de cámara :

$$\text{Matriz de parámetros intrínsecos} = \begin{bmatrix} 155.4981 & 0.0000 & 74.7454 & 0.0000 \\ 0.0000 & 157.1137 & 55.5787 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 \end{bmatrix} \tag{3.2}$$

La cuarta columna se lleno de ceros pues posteriormente se debe ponderar por la matriz de parámetros extrínsecos R|t como se menciono en el marco teórico.

Cabe destacar que la dificultad al llevar a cabo este algoritmo radica en que debido a que la cámara termal captan luz emitida en el espectro infrarrojo lo que corresponde a firmar termales, debido a la difusión del calor la resolución de las esquinas en las imágenes termales ”aguda” espacialmente hablando lo que dificulta la detección de las esquinas del tablero de ajedrez al momento de calibrar.

3.2.2. Matriz de parámetros extrínsecos

Dado que se trata de una cámara modelo *pinhole* como se estableció anteriormente para pasar del plano de la imagen al plano global es necesario rotar la imagençomo se presenta en la figura 3.5

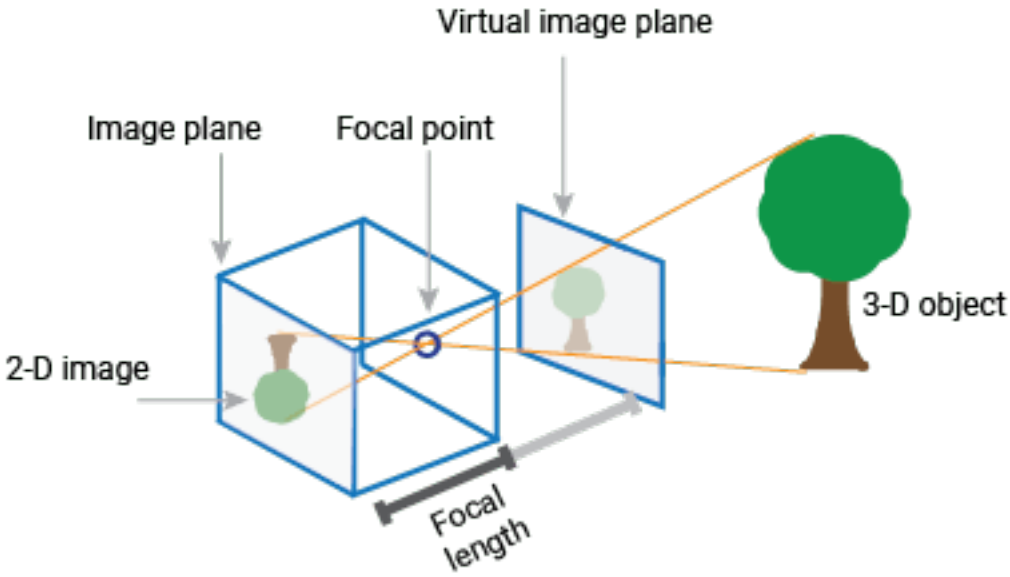

Figura 3.5: Relación entre el plano imagen y el "mundo 3D" (imagen obtenida de [19])

Esta rotación corresponde a tf= [0.0, 0.0, 0.0, -1.5708, 0.0, -1.5708] en ros pues el eje X en esta plataforma es normal a la cámara (en este caso).

Siendo los valores tf correspondientes a [$traslacion_x, traslacion_y, traslacion_z$, roll, pitch, yaw].

Con [roll, pitch, yaw] rotaciones con respecto a los ejes [x, y, z] respectivamente.

Luego tomando en cuenta la traslación de la cámara con respecto al origen global definida en 3.1, se define la función en MATLAB que transforma estos vectores traslación y rotación en vectores homogéneos.

La función esta definida como :

```
function H = createHomogeneousMatrixFromROS(tx, ty, tz, rx, ry, rz)
H = eul2tform([rx,ry, rz], 'ZYX');  %final
H(1:3, 4) = [tx; ty; tz];
```

Esta función convierte ángulos en formato "euler" a una matriz de transformación homogénea, luego al obtener las rotaciones como matriz de transformación homogénea y luego asignar para la cuarta columna el vector[tx; ty; tz] se obtiene la matriz de parámetros extrínsecos [R|t]. Cabe destacar que dada la forma como se definió, esta matriz convierte los objetos desde la cámara al "mundo 3D" por lo tanto para proyectar los puntos 3D sobre el plano imagen es necesario ponderar por:

$$inv([R|t]) \tag{3.3}$$

Así se llevan los puntos a proyectar "del mundo a la camara" y luego de proyectan usando la matriz de parámetros intrínsecos 2.2.

3.3. Deteccion de Objetos

Dado que para implementar el algoritmo de STT y posteriormente el algoritmo de MTT se requieren de mediciones para cada tiempo t y en este caso las mediciones son detecciones de los objetos en las imágenes termales. Se re-entrenó una red neuronal con un *Dataset* de imágenes termales para mejorar su desempeño en este campo (ya que como se ha mencionado anteriormente detectar objetos en imágenes termales puede resultar engorroso debido a lo difuminados que pueden resultar los bordes de los objetos a causa de la convección térmica.

Para esta tarea se uso el *framework* OpenMMLab ya que resulta sencillo entrenar y personalizar redes con el fin de mejorar sus desempeños.

En particular con fines experimentales se eligió arbitrariamente una red ya implementada con pesos resultantes del entrenamiento con imágenes RGB y se re entreno usando el dataset FLIR ADAS [20] que contiene 9711 imágenes termales separadas en conjuntos de entrenamiento y validación, dentro de este dataset contiene aproximadamente 375,000 etiquetas en imágenes termales visibles (8bits), las clases disponibles en el dataset son 15 , sin embargo en la presente memoria se enfoco la detección en la clase 1 correspondiente a humanos.

La red elegida fue:

"mask-rcnn_r50-caffe_fpn_ms-poly-3x_coco"

Los pesos a partir de los que se inicio el reentrenamiento con la red fueron:

"mask_rcnn_r50_caffe_fpn_1x_coco_bbox_mAP-0.38_segm_mAP-0.344_20200504_231812-0ebd1859"
Ambos obtenidos del *framework* OpenMMLab.

Se eligieron estos archivos de configuración de la red y pesos iniciales en desmedro de una red excelsa en detección de humanos debido a que se desea apreciar en la implementación del algoritmo MTT como se desempeña ante "*miss-detections*" y "*Clutter*".

La red se entrenó usando una tarjeta : NVIDIA GeForce RTX 4070 y las versiones de los siguientes entornos:

- TorchVision: 0.16.2
- OpenCV: 4.8.1
- MMEngine: 0.10.1

Se definió un *learning rate* de 2.0000e-02 , un *batch size* de 16, se entreno por 12 épocas y la cantidad de imágenes usada por época fue de 2049, la época con mejor desempeño en

todas las métricas para el conjunto de validación fue la epoca numero 12 (se atribuye al bajo *learning rate* establecido que no haya habido *overfitting*)

Las métricas obtenidas para las detecciones i.e. *bboxes* para los "mejores pesos" fueron:

Tabla 3.1: Resultados de la evaluación en la época 12.

Epoch(val)	bbox_mAP_50	bbox_mAP_s	bbox_mAP_m	bbox_mAP_l
12	0.4900	0.2360	0.5360	0.5620

Es necesario tomar en cuenta que el dataset FLIR ADAS para la cámara termal se grabo usando una cámara "Teledyne FLIR Tau 2 640x512, 13mm f/1.0 (HFOV 45º, VFOV 37º)" , es decir con una resolución superior a la cámara usada durante los experimentos llevados a cabo en la presente memoria i.e. "FLIR Lepton 3.5 , 160 x 120 57º HFOV with shutter" lo que implica que el desempeño alcanzado por la red luego del entrenamiento no se puede equiparar al desempeño logrado por la detección de la misma red para imágenes obtenidas desde la cámara FLIR Lepton, debido principalmente a que una baja resolución relativa a la usada para el entrenamiento exacerba los problemas intrínsecos de las detecciones termales que corresponden a que los bordes de los objetos no están bien definidos.

3.4. Ground Truth 3D

Mediante la app de MATLAB 2022 "ground truth labeler" se realizo el etiquetado del "*Ground Truth* para los objetos visibles en la nube de puntos resultante de las mediciones del Lidar Velodyne a lo largo de la toma de muestra del Dataset con el que se llevan a cabo los experimentos.

Este etiquetado se llevo a cabo mediante el algoritmo de interpolación presente en la misma app, este algoritmo interpola las detecciones a lo largo del tiempo creando así un etiquetado para todos los *Timeframes* de las mediciones a partir de las posiciones marcadas por el usuario.

Es crucial la obtención de este conjunto de *Ground Truths* debido a que con estos se puede medir el desempeño de el rastreo de objetos en el mundo 3D.

Finalmente , debido a que las traslaciones del la cámara termal con el "mundo 3D" se llevaron a cabo con respecto a la base del LIDAR Velodyne VLP-16 se pueden usar estos puntos en el algoritmo sin hacer cambios a la matriz de proyección de la cámara termal.

3.5. STT mediante Filtro de Partículas

El algoritmo de STT implementado está dado por un filtro de partículas cuyos parámetros corresponden a:

- $x_+ = Fx + V$, donde:

$$F = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$V = \begin{bmatrix} \frac{T^2}{2} & 0 & 0 \\ T & 0 & 0 \\ 0 & \frac{T^2}{2} & 0 \\ 0 & T & 0 \\ 0 & 0 & \frac{T^2}{2} \\ 0 & 0 & T \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \tag{3.4}$$

- T, tiempo de muestreo, corresponde al tiempo de muestreo de la cámara térmica (1/6) [s].

- F, matriz de transición de estados.

- V, matriz que modela la dinámica de los estados, donde $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$ es un vector aleatorio de 3 filas por N columnas, con N igual al número de partículas. Cada elemento de la matriz tiene distribución gaussiana con media cero y desviación estándar uno, ponderado escalarmente por un valor arbitrario σ_V para cada una de sus componentes, simulando así velocidad y aceleración aleatoria para el modelo de movimiento de las partículas.

- FOV se define como la sección esférica en el eje x, y de radio $r = 30$ entre los ángulos -31.25^o y 31.25^o, con altura 3. Este cono delimita dónde se generan las nuevas partículas y elimina las partículas que, después de la predicción, yacen fuera de sus límites, asignándoles un peso 0. Es necesario eliminar las partículas que yacen fuera de sus límites, ya que las proyecciones llevadas a cabo por la ponderación de la matriz de proyección y un vector posición de una partícula cualquiera solo tienen sentido matemático si esta partícula se encuentra dentro del FOV de la cámara.

- Matriz de proyección, dada la fórmula presente en 2.1 y lo mencionado en 3.3, donde:

$$K = \begin{bmatrix} 155.4981 & 0.0000 & 74.7454 & 0.0000 \\ 0.0000 & 157.1137 & 55.5787 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 \end{bmatrix} \tag{3.5}$$

,

$$\text{inv}(R|t) = \begin{bmatrix} 0.0500 & -0.9988 & 0.0000 & -0.0000 \\ -0.0000 & -0.0000 & -1.0000 & 0.3500 \\ 0.9988 & 0.0500 & -0.0000 & 0.0000 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{3.6}$$

Entonces,

$$P = \begin{bmatrix} 82.4231 & -151.5684 & -0.0003 & 0.0001 \\ 55.5087 & 2.7775 & -157.1139 & 54.9899 \\ 0.9988 & 0.0500 & -0.0000 & 0.0000 \end{bmatrix} \tag{3.7}$$

- Matriz de covarianza del ruido de la observación *model.R*: $model.D = diag([5; 5])$; $model.R = model.D * model.D'$

Es importante destacar el núcleo de la implementación del presente algoritmo que consiste en que luego de cada predicción se transforman los vectores de estados que se encuentran dentro del FOV a vectores presentes en el plano de la imagen , es decir sus posiciones en el espacio pasan de 3D a 2D mediante la multiplicación de los vectores de la posición de los estados homogéneos por la matriz de proyección , luego una vez proyectadas las partículas sobre la imagen se calcula la verosimilitud de estas con respecto a la detección z dada la función:

```
qz1 = mvnpdf(z_hat', z', model.R)
```

Donde mvnpdf es una función que calcula la función de densidad de probabilidad multivariante (pdf) para una distribución normal multivariante (en este caso con respecto a la detección z' dada la matriz de covarianza model.R. La función mvnpdf toma como entrada una matriz de muestras , en este caso las partículas proyectadas z_hat' y devuelve los valores de la función de densidad de probabilidad correspondientes. En otras palabras esta función mide "que tan cerca" están los objetos de la detección desde la perspectiva de la cámara y el ruido asociado a ella.

Finalmente el remuestreo se lleva a cabo como se mencionó en 2.1.

Adicionalmente se destaca que para la presente implementación no se le agregó la opción de manejar *miss detections* ya que se tomo el desarrollo del presente algoritmo como un paso para poder implementar el algoritmo de MTT y es en este algoritmo donde de maneja la posibilidad de *miss detections*.

3.6. MTT mediante filtro GLMB

Para la implementación del filtro GLMB se uso el filtro STT desarrollado en la presente memoria con el fin de que las partículas de aquel filtro modelen las densidades de probabilidades de los objetos del MTT, de esta forma en términos generales se "corre" un STT para representar cada densidad de probabilidad de cada objeto en el MTT. Para el esquema del algoritmo presente en 2.3 se tienen sus parámetros:

3.6.1. Parametros filtro GLMB

- La probabilidad de existencia para las nuevas hipótesis (*birth*)(presente en 2.27 esta dada por 0.02
- Para la proyección de las partículas sobre la imagen con el fin de generar la verosimilitud se utiliza la misma matriz de proyección que para el STT

- Para el proceso de calculo de verosimilitud de las partículas se mantiene el *observation noise covariance* dado por [25,0];[0, 25]

- T=1/6 [s] al igual que en STT pues se comparte el mismo periodo "muestreo" ya que se usa la misma cámara para las detecciones.

- Modelo de movimiento: para cada partícula para cada predicción se mantiene el modelo de movimiento utilizado en el algoritmo STT

- FOV , se utiliza el mismo FOV que para el STT , pues se usa la misma cámara

- Parámetros de *clutter* presentes en 2.31:

  - $\lambda_k = 0.2$, corresponde al numero esperado de falsas alarmas , se determinó empíricamente que el numero de falsas alarmas con la red de detección de objetos implementada corresponde aproximadamente a un 20 %

  - Volumen del FOV= multiplicación de los limites del FOV , en este caso 30*(12--12)* 3.7 = 2664, por lo tanto la densidad de probabilidad de falsas alarmas para una medición j es: 0.00007

- Para el *birth* se definió un solo *birth state density* para cada tiempo t dado por: una distribución con posición aleatoria dentro del FOV en cuestión. Las velocidades x',y',z' fueron creadas mediante la ponderación del parámetro sigma_velo definido como 1.3 por una variable n variables aleatorias con distribución gaussiana con media en 0 y desviación estándar 1 para cada componente de velocidad x',y',z'. Cabe destacar que la velocidad sigma_velo se definió como 1.3 debido a que esta es aproximadamente la velocidad de la caminata de un humano en [m/s].

- Empíricamente se determinó que con la red de detección implementada para la resolución de la cámara implementada la probabilidad de detección corresponde a P_D=0.8.

- Arbitrariamente se determino la probabilidad de supervivencia P_s como 0.99, por lo tanto la probabilidad de no supervivencia esta dada por $1 - P_s$.

- Con el fin de reducir el costo computacional del algoritmo se definió el numero de *births* por iteración como 1

Adicionalmente es necesario notar que a un objeto no se le asigna una probabilidad de detección si es que esta fuera del FOV.

Capítulo 4

Resultados

4.1. STT

Para comprobar la correcta implementación del algoritmo de filtro de partículas implementado para llevar a cabo el STT se definieron 2 escenas de detección, la primera es descrita por una persona caminando en dirección constante a lo largo del tiempo ver figura 4.1 , cuyo archivo CSV con las correspondientes coordenadas de las detecciones se encuentra disponible en el enlace https://github.com/thedyyy/Tesis/blob/main/results3.csv del CSV se aprecia que la detección se pierde en el tiempo 73.

Figura 4.1: Caso de estudio 1 para STT

Para la secuencia de imágenes perteneciente a las detecciones presentes en el archivo CSV indicado se tienen 2 vídeos presentes en los siguientes links

https://youtu.be/_pQH8yA7-Tk en el presente link se muestra el gráfico de las predicciones para la secuencia de imágenes en cuestión. Este gráfico está representado luego de que se truncaran las predicciones de estados fuera del FOV. Como se puede apreciar el vídeo parte desde un ”dimante” que corresponde a lo visible desde la perspectiva superior (observando al plano (x,y) posteriormente las partículas convergen a una ”línea recta” constantemente hasta que se pierde la medición en el tiempo 72. Sin embargo se espera de un algoritmo de tracción que no solo ”proyecte” en la dirección del objeto, si no que acote la posición de sus partículas en la dirección del eje X también, sin embargo esto no es válido para la presente

implementación, debido a que se esta usando una sola cámara no hay punto de comparación y por lo tanto no hay percepción de profundidad. Es por esto ultimo que el algoritmo solamente proyecta en la dirección donde esta el objeto.

En el siguiente link https://youtu.be/UfyTIKmXFhk se encuentran las figuras de las proyecciones de las partículas sobre el plano imagen (puntos en azul) y la proyección del rectángulo de detección sobre el plano imagen también. Es inmediato notar que el calculo de la verosimilitud se lleva a cabo bien, pues para cada estado las partículas se proyectan muy cercanamente al rectángulo de la detección. De no llevarse a cabo correctamente la etapa de calculo de verosimilitud (*likelihood*) no seria posible notar tan evidentemente la cercanía de los puntos proyectados (que son resultado directo de el cálculo de verosimilitud para el tiempo K anterior del algoritmo y por lo tanto producto de un buen algoritmo de verosimilitud).

En la segunda secuencia de imágenes presente en el link https://github.com/thedyyy/Tesis/blob/main/results4.csv se tiene una persona caminando de izquierda a derecha. Según la perspectiva de la cámara, no existe pérdida de la detección.

El vídeo de las proyecciones de los puntos sobre el plano de la imagen, en conjunto con la proyección del rectángulo de detección presente en el link https://youtu.be/RnCk6gHV7Hw, denota una vez mas la capacidad el algoritmo implementado para seguir las direcciones de un objeto dentro del plano imagen. Sin embargo, al estudiar el vídeo presente en que corresponde a las partículas luego de filtrar los que se encuentran fuera del FOV se evidencia una vez mas que el algoritmo es incapaz de detectar profundidad.

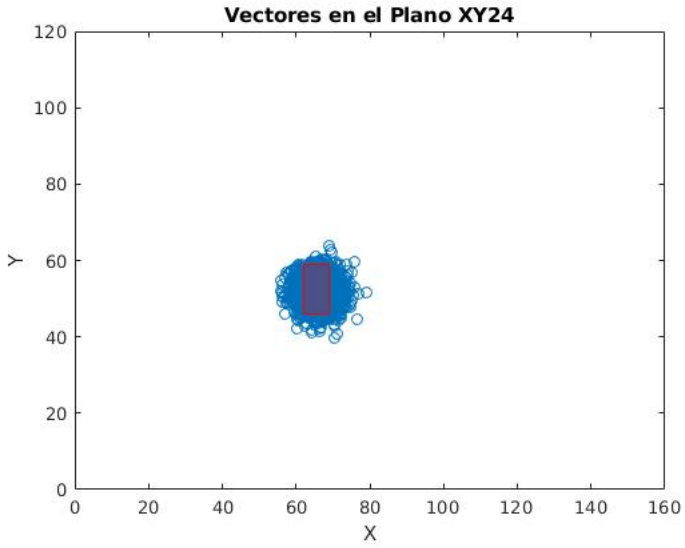

Figura 4.2: Proyección de las predicciones y rectángulo de detección sobre el plano de la imagen

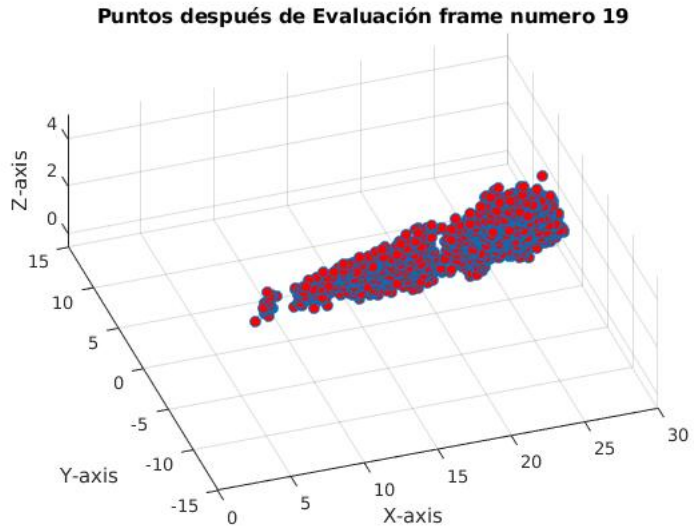

Figura 4.3: Vista de 3 ejes del conjunto de puntos X_predict

4.2. MTT

Para el algoritmo MTT de filtro δ-GLMB se llevó a cabo el "seguimiento" sobre las imágenes termales presentes en los dos rosbags adjuntos al presente documento. Teniendo ya las detecciones de los objetos a lo largo del tiempo y los *Ground truths* para cada uno de los objetos presentes en las imágenes se procede a graficar las trayectorias de los *Ground truths* (desde ahora GT) y compararlo con las trayectorias de los objetos calculadas por el algoritmo δ-GLMB

4.2.1. MTT caso 1

Se graficaron dados los GT las posiciones de interés (x,y) y los ángulos , esto último debido a que como el algoritmo carece de un punto de referencia no es capaz de llevar a cabo el proceso de triangulación necesario para rastrear bien a los objetos con respecto a la profundidad, es por esto que dentro de una dirección dada el algoritmo elige algún punto x,y que tenga alta verosimilitud, sin embargo , como se mencionó anteriormente estos pares no necesariamente indican la posición exacta del objeto, sino que indican la dirección de éste , es por esto que se graficó "Direccion del objeto" que consta de el ángulo dado por el arctan(y/x) de esta forma se tiene una métrica que indica que tan bien está "rastreando" con respecto a la dirección x,y el algoritmo.

En las figuras 4.4 y 4.5 se presentan los gráficos de GT para la primera secuencia de imágenes estudiada:

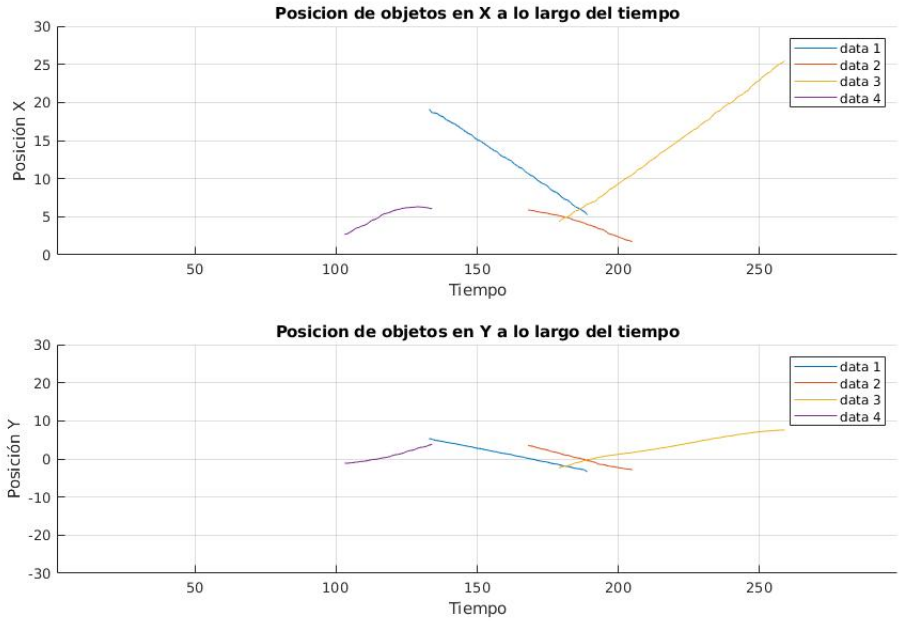

Figura 4.4: Posiciones de los objetos (GT) en el tiempo para el primer caso de estudio

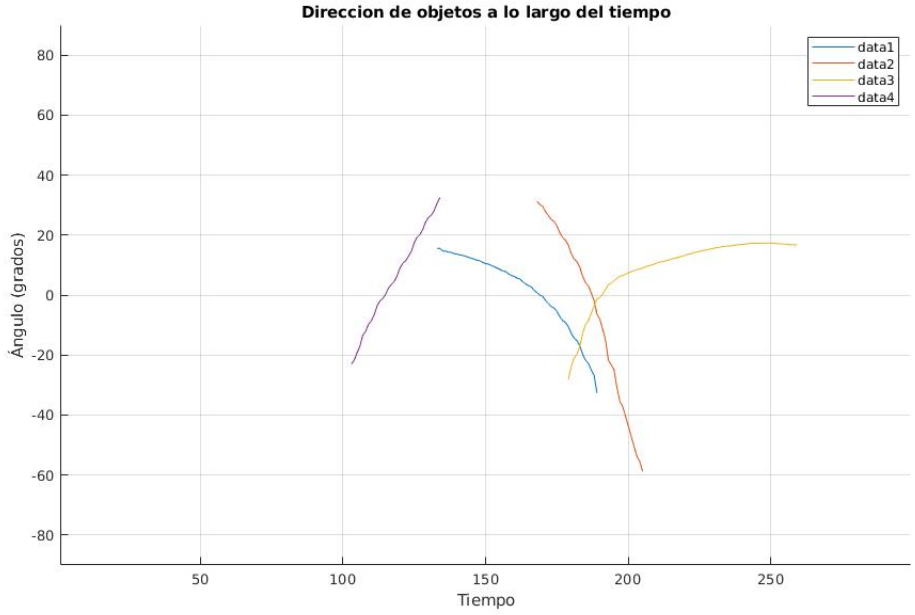

Figura 4.5: Dirección de los objetos (GT) en el tiempo para el primer caso de estudio

Para comparar con un caso base sólido se etiquetaron las detecciones de los objetos en las imágenes termales (i.e. 2D) simulando un 100 % de efectividad en las detecciones, estas ultimas pueden ser víctimas de la oclusión por objetos inertes como árboles o incluso otras personas, sin embargo siempre que no ocurra una oclusión el objeto será detectado. Los resultados de la implementación del algoritmo para estas detecciones ”perfectas” presentan

en las figuras 4.6 y 4.7:

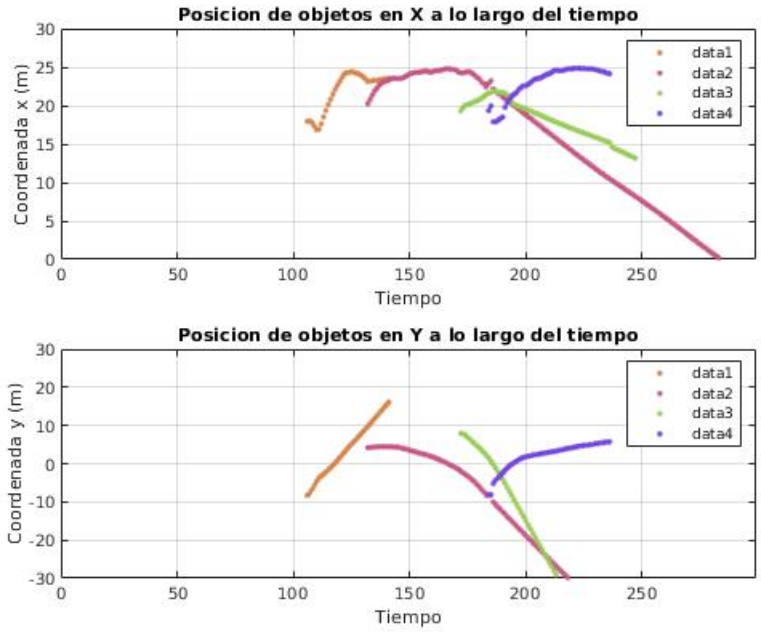

Figura 4.6: Posiciones de los objetos en el tiempo para el caso de detecciones "perfectas"

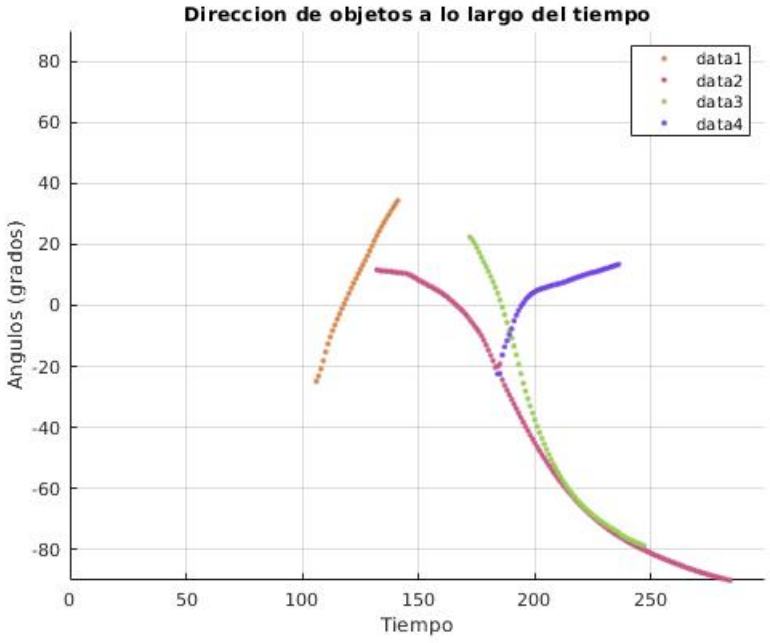

Figura 4.7: Dirección de los objetos en el tiempo para el caso de detecciones "perfectas"

Como se mencionó anteriormente, dado que el algoritmo no puede medir la profundidad con precisión, se requiere evaluar su desempeño mediante los gráficos de dirección de los ob-

jetos y sus similitudes. Para lograr esto, se procedió a calcular la diferencia entre los ángulos de las direcciones de los objetos identificados manualmente por el usuario. En este proceso, se seleccionaron dos etiquetas asociadas y se determinaron las discrepancias de ángulos correspondientes. Se observó que, para todos los casos de estudio, la diferencia máxima fue de aproximadamente 10º, lo cual concuerda con la comparación visual de los gráficos presentados en las figuras 4.5 y 4.7. Es importante destacar que este cálculo se realizó teniendo en cuenta el período de tiempo durante el cual el Ground Truth (GT) en 3D estuvo presente, y no al revés.

Finalmente notar que la cardinalidad total de objetos seguidos en la escena por el algoritmo coincide con el número de objetos presentes en el GT.

4.2.2. MTT caso 2

Para el segundo caso de estudio se tienen los gráficos correspondientes al GT presentes en las figuras 4.8 y 4.9:

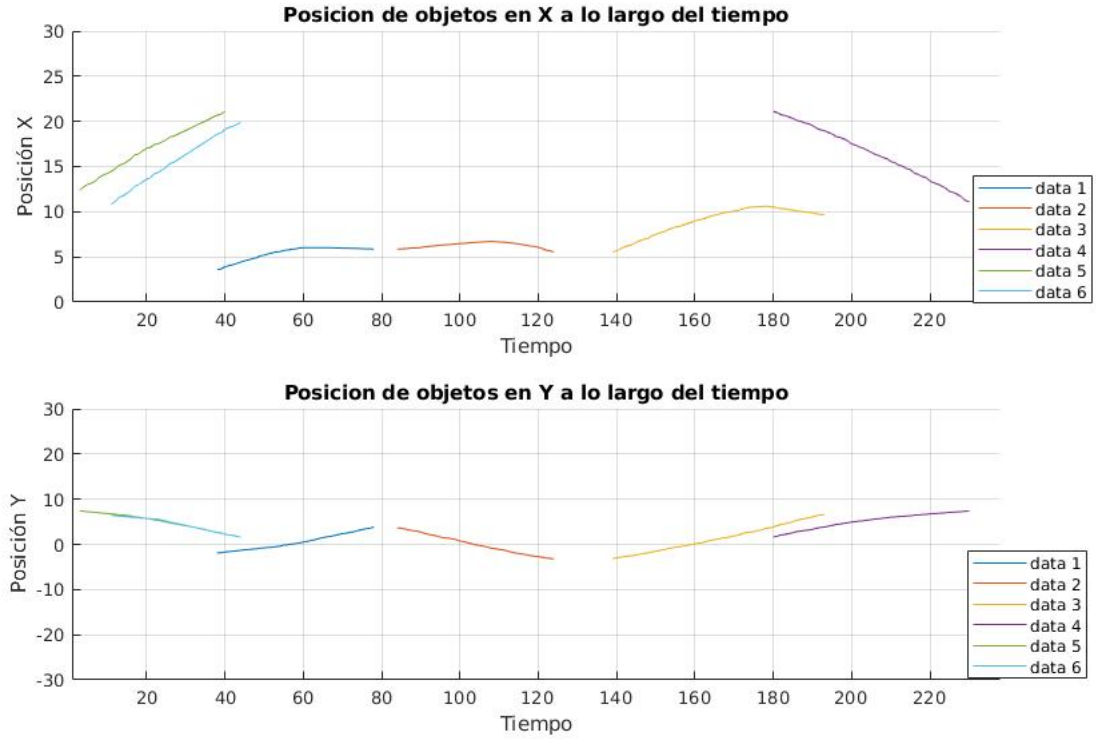

Figura 4.8: Posiciones de los objetos (GT) en el tiempo para el segundo caso de estudio

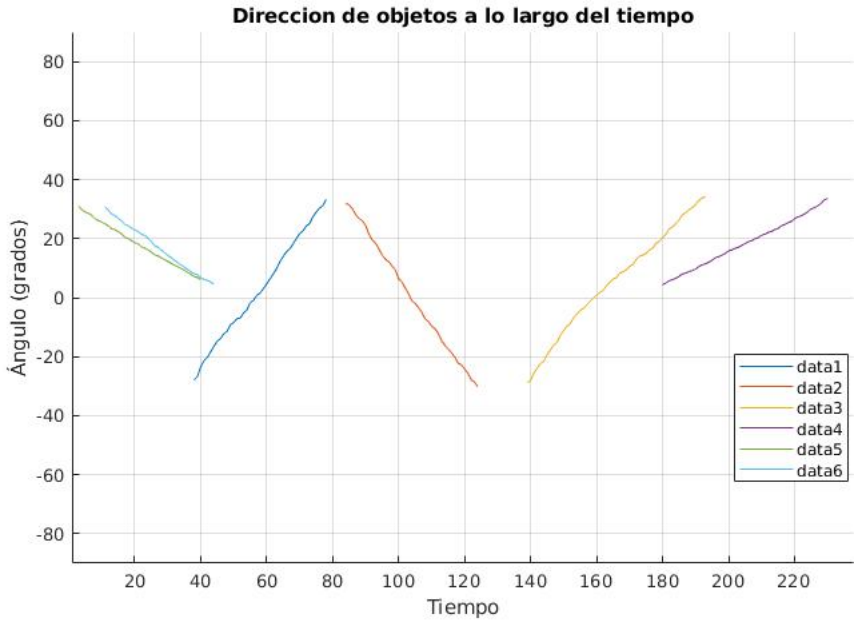

Figura 4.9: Dirección de los objetos (GT) en el tiempo para el segundo caso de estudio

Al igual que en el caso de estudio anterior se estudió un caso con detecciones perfectas, presentes en los gráficos de las figuras 4.10 y 4.11

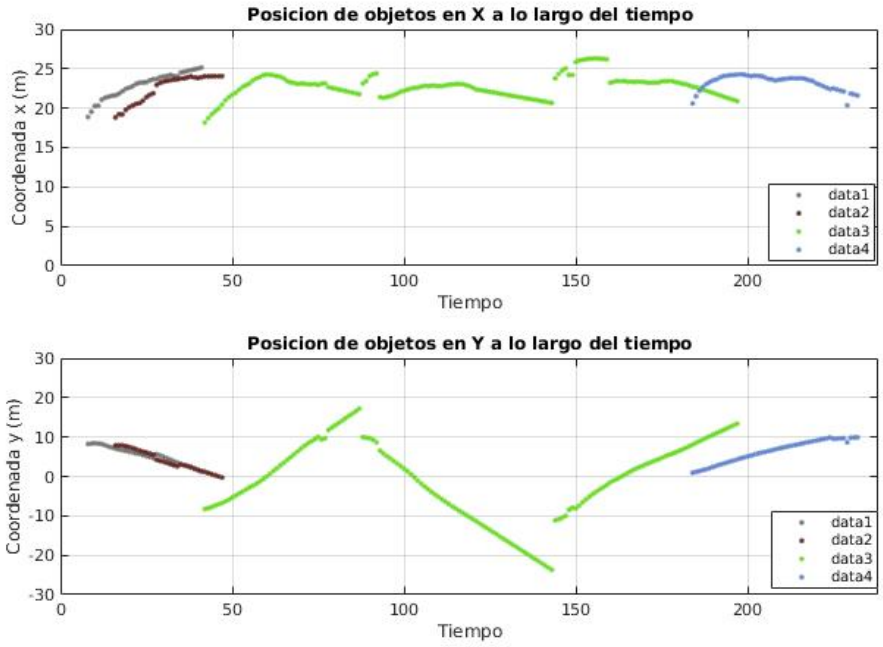

Figura 4.10: Posiciones de los objetos en el tiempo para el segundo caso de estudio con detecciones ”perfectas”

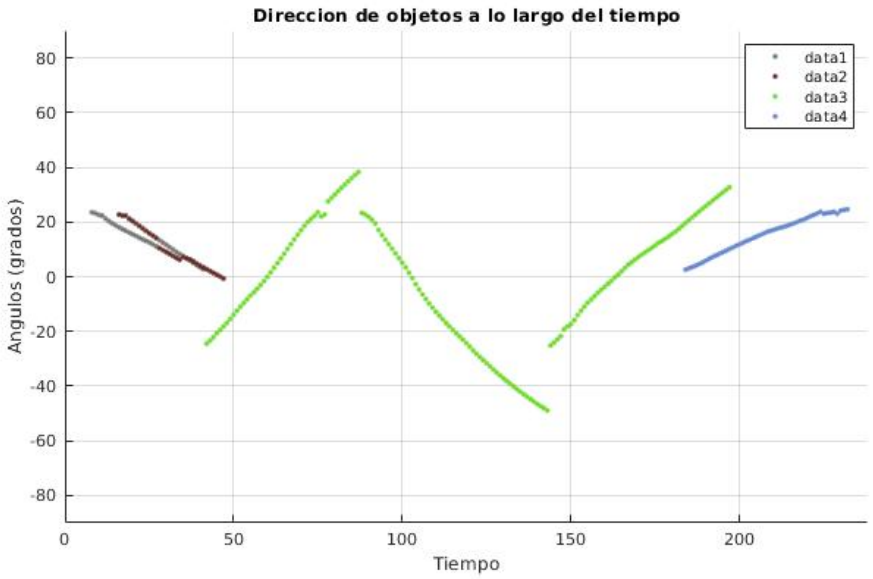

Figura 4.11: Dirección de los objetos en el tiempo para el segundo caso de estudio con detecciones "perfectas"

Finalmente para el presente caso se agrego el estudio del desempeño del algoritmo sin etiquetado adicional, es decir se usó solo la red neuronal entrenada para llevar a cabo la detección de los objetos. Los gráficos correspondientes al seguimiento de objetos para este estudio se encuentran presentes en las figuras 4.12 y 4.13 .

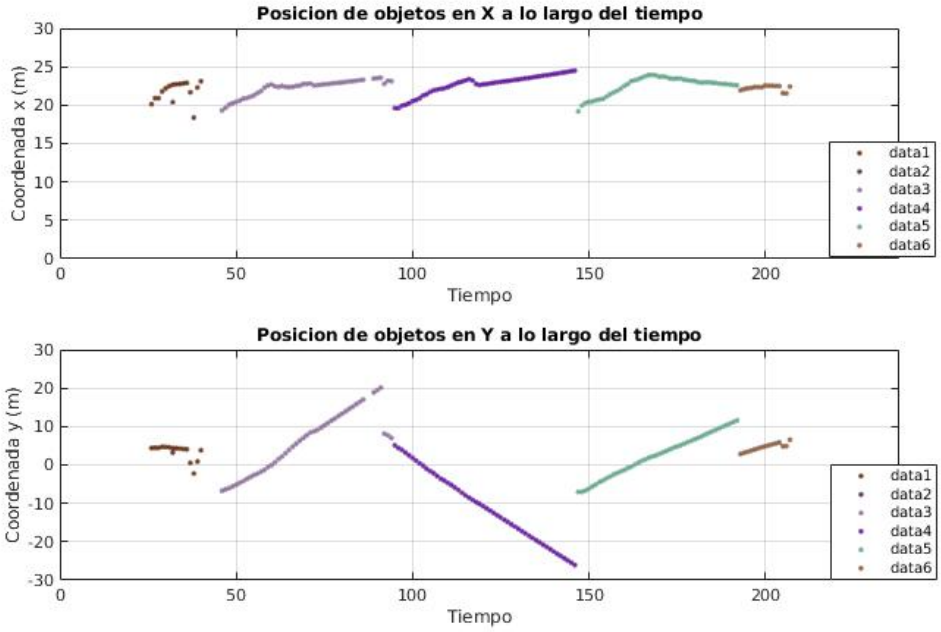

Figura 4.12: Posiciones de los objetos en el tiempo para el segundo caso de estudio sin detecciones "perfectas"

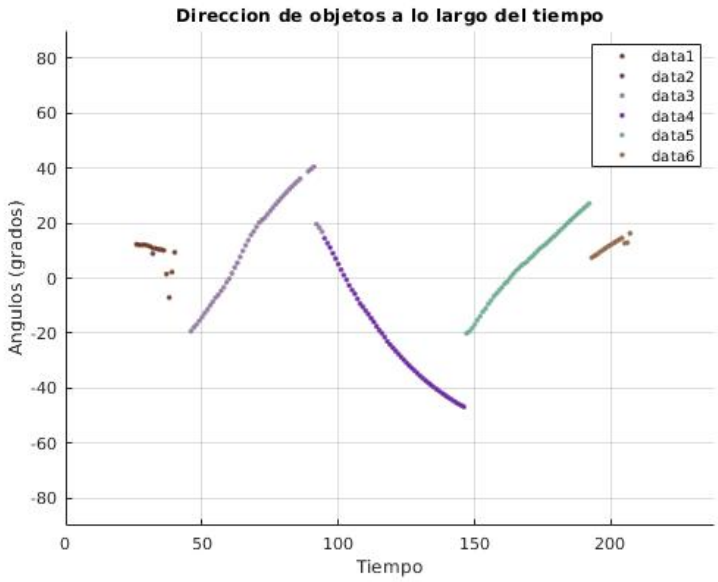

Figura 4.13: Dirección de los objetos en el tiempo para el segundo caso de estudio con detecciones "perfectas"

Es interesante notar como resultado del presente caso de estudio que:

- La cardinalidad disminuyó para el caso de la detección perfecta, esto se debe a que a pesar de que el algoritmo se desempeño bien llevando cabo el seguimiento de los objetos (una vez mas no superando los 10º de diferencia entre predicción y GT) no fue capaz de asignar correctamente las trayectorias a objetos diferentes. Al contrario, la cardinalidad fue correcta para el caso sin etiquetado a mano, sin embargo , este resultado no se condice con un buen desempeño, pues por simple inspección se puede apreciar que el algoritmo pudo llevar a cabo de manera correcta el seguimiento de 4 objetos solamente,esto se atribuye a que las detecciones no fueron lo suficientemente buenas, evitando así un flujo correcto del algoritmo.

- Mediante inspección se puede apreciar que el seguimiento de los objetos comparte cotas superiores e inferiores en los gráficos de "direcciones", lo que indica una vez mas que el algoritmo implementado es eficiente al momento de seguir las direcciones en el plano x,y.

Finalmente es necesario notar que para el presente casos de estudios la imagen fue "obstruida" con una maquina de humo aproximadamente un 70 % del tiempo demostrando así la robustez de las detecciones termales ante condiciones ambientales adversas.

Figura 4.14: Correspondencia RGB con humo

Figura 4.15: Correspondencia Termal con humo

Como se puede apreciar en las imágenes 4.15 y 4.14 el humo en la escena no afecta la visibilidad de la imagen termal y por lo tanto no incide en las detecciones de esta, traduciéndose así en un buen desempeño de los algoritmos usan este tipo de mediciones.

Capítulo 5

Conclusiones

5.1. Conclusiones sobre el trabajo

Se logró llevar a cabo la calibración de una cámara térmica de alta resolución, logrando un error de proyección inferior a 0.2 píxeles. Siendo este logro fue fundamental para transformar con precisión las posiciones de los objetos del entorno tridimensional a coordenadas bidimensionales en las imágenes térmicas.

Se logró implementar y entrenar una red neuronal del estado del arte capaz de detectar exitosamente seres humanos en imágenes termales.

Se implementó un algoritmo del estado del arte (δ-GLMB) capaz de llevar a cabo el seguimiento de las direcciones de múltiples objetivos utilizando una cámara termal como fuente de información. Adicionalmente se evaluó el desempeño del algoritmo implementado comparándolo con el *Ground Truth* obteniendo resultados satisfactorios tanto cualitativa como cuantitativamente.

Se realizó una contribución significativa al estado del arte al adaptar con éxito el algoritmo δ-GLMB para su uso con una cámara térmica como fuente de información. Esta adaptación permitió alcanzar un buen desempeño en tareas de seguimiento de múltiples objetivos (MTT), incluso en escenarios adversos a la detección, como la presencia de neblina o humo en una imagen. Este avance, debido a la resiliencia de las detecciones llevadas a cabo por cámaras térmicas, amplía el alcance de aplicación del algoritmo implementado en una variedad de áreas, incluyendo conducción autónoma, vigilancia de seguridad y detección de incendios forestales, entre otros campos de aplicación.

5.2. Trabajo Futuro

A modo de trabajo futuro se plantea el uso de una segunda cámara con el fin de entregar al algoritmo percepción de profundidad. Se plantea el uso de la cámara FLIR AX5 o el uso de la cámara RGB presente en el dataset sobre el que se evaluó el desempeño del algoritmo. Para esto es necesario en primera instancia implementar un T variable y ”etiquetar” las mediciones provenientes de la segunda cámara debido a que esta usaría una diferente matriz de proyección. De este trabajo sería interesante ver como se complementa el buen desempeño de los algoritmos de detección para las imágenes RGB con la robustez de la detección de la

cámara termal ante escenarios adversos.

Crear un dataset con ”lluvia artifical” con el fin de evaluar la robustez de las detecciones de objetos usando cámaras termales (ya que se comprobó el buen desempeño ante el humo).

Finalmente, implementar un algoritmo de MTT usando 2 cámaras para tener percepción de profundidad y así poder usar métricas de evaluación de desempeño de MTT que en un principio se tenían contempladas para la presente como lo es CLEAR MOT.

Bibliografía

[1] Kalman, R., "A new approach to linear filtering and prediction problems," Transactions of the ASME–Journal of Basic Engineering, vol. 82, pp. 35–45, 1960.

[2] Arulampalam, M. S., Maskell, S., Gordon, N., y Clapp, T., "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," IEEE Transactions on Signal Processing, vol. 50, no. 2, pp. 174–188, 2002, doi:10.1109/78.978374.

[3] Reuter, S., Multi-object tracking using random finite sets. 2014. doi: 10.18725/OPARU-3204.

[4] Hartley, R. I. y Zisserman, A., "Multiple view geometry in computer vision," cap. 2, p. 26, Cambridge University Press, second ed., 2004. ISBN: 0521540518.

[5] Hartley, R. I. y Zisserman, A., "Multiple view geometry in computer vision," cap. 6, p. Número de páginas relevantes para el capítulo 6, Cambridge University Press, second ed., 2004. ISBN: 0521540518.

[6] Hartley, R. I. y Zisserman, A., "Multiple view geometry in computer vision," cap. 7, p. Número de páginas relevantes para el capítulo 7, Cambridge University Press, second ed., 2004. ISBN: 0521540518.

[7] Krizhevsky, A., Sutskever, I., y Hinton, G. E., "Imagenet classification with deep convolutional neural networks," en Proceedings of the 25th International Conference on Neural Information Processing Systems, 2012.

[8] Hmidani, O. y Alaoui, E. M. I., "A comprehensive survey of the r-cnn family for object detection," en 2022 5th International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–6, 2022, doi:10.1109/CommNet56067.2022.9993862.

[9] Thrun, S., Burgard, W., y Fox, D., Probabilistic Robotics. Cambridge, Massachusetts: The MIT Press, 2005.

[10] Mahler, R. P. S., Advances in Statistical Multisource-Multitarget Information Fusion. Artech House, 2014. Pages 43–55.

[11] Mahler, R. P. S., Advances in Statistical Multisource-Multitarget Information Fusion. Artech House, 2014. Pages 445–449.

[12] Vo, B.-N., Vo, B.-T., y Phung, D., "Labeled random finite sets and the bayes multi-target tracking filter," IEEE Trans. Signal Process., vol. 62, no. 24, pp. 6554–6567, 2014.

[13] Murty, K. G., "An algorithm for ranking all the assignments in order of increasing cost," Operations Research, vol. 16, no. 3, pp. 682–687, 1968, http://www.jstor.org/stable/168595.

[14] He, X., Tharmarasa, R., Pelletier, M., y Kirubarajan, T., "Accurate murty's algorithm for multitarget top hypothesis extraction," Fusion 2011 - 14th International Conference on Information Fusion, 2011.

[15] Arnold, S. F., "18 gibbs sampling," en Computational Statistics, vol. 9 de Handbook of Statistics, pp. 599–625, Elsevier, 1993, doi:https://doi.org/10.1016/S0169-7161(05)80142-7.

[16] Vo, B.-N., Vo, B.-T., y Hoang, H. V., "An efficient implementation of the generalized labeled multi-bernoulli filter," IEEE Transactions on Signal Processing, vol. 65, no. 8, pp. 1975–1987, 2017, doi:10.1109/tsp.2016.2641392.

[17] Vo, B.-T. y Vo, B.-N., "Labeled random finite sets and multi-object conjugate priors," IEEE Trans. Signal Process., vol. 61, no. 13, pp. 3460–3475, 2013.

[18] ElSheikh, A., Abu-Nabah, B. A., Hamdan, M. O., y Tian, G.-Y., "Infrared camera geometric calibration: A review and a precise thermal radiation checkerboard target," Sensors, vol. 23, no. 7, p. 3479, 2023, doi:10.3390/s23073479.

[19] MathWorks, "Camera calibration," Año de la última actualización o acceso, https://la.mathworks.com/help/vision/ug/camera-calibration.html. Consultado el dd de mes de año.

[20] Systems, F., "FLIR ADAS Dataset," 2024, https://www.flir.com/oem/adas/adas-dataset-form/.