Proyecto Fin de Máster
Máster en Ingeniería Electrónica, Robótica y Automática

Algoritmo de localización visual del perfil de un carril en una imagen adquirida con sistemas de visión de luz estructurada

Autor: Juan Manuel Amador Olivares
Tutor: Hipólito Guzmán Miranda

Departamento de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Dedicado a
mi familia

II

Agradecimientos

Quiero agradecer en primer lugar la guía de mi tutor y el apoyo de mis compañeros de trabajo, sin los que hubiera sido imposible llevar a cabo este proyecto.

Agradecer la inestimable ayuda de mis profesores

Agradecer a mi familia y a María Jesús, por su apoyo incondicional durante el día a día.

Resumen

En este trabajo final de Máster se presenta un algoritmo de visión por computador capaz de extraer la posición y orientación del perfil de un carril en imágenes tomadas por un sistema de visión con luz estructurada.

El algoritmo presentado es una parte de un proyecto mucho mayor con el objetivo de la auscultación de vías y medida del estado dinámico de un vehículo ferroviario.

Serán introducidos algunos conceptos sobre la auscultación de vías y descrito el contexto dónde trabaja el algoritmo de visión desarrollado. Se profundizará en el diseño del algoritmo, así como su desarrollo e implementación. También serán presentados y analizados los resultados del algoritmo en un ensayo real.

Para finalizar se presentarán mejoras del algoritmo, que pueden aumentar su funcionalidad o mejorar su rendimiento.

Abstract

In this final Master's thesis a computer vision algorithm is presented capable of extracting the position and orientation of the profile of a rail in images acquired by a vision system with structured light.

The presented algorithm is a part of a much larger project with the objective of rail auscultation and measurement of the dynamic state of a railway vehicle.

Some concepts about path auscultation will be introduced and the context where the developed vision algorithm works will be described. The design of the algorithm will be deepened, as well as its development and implementation. The results of the algorithm will also be presented and analyzed in a real test.

Finally, algorithm improvements will be presented, which may increase its functionality or improve its performance.

Índice general

Índice de figuras

Capítulo 1

Introducción

Este proyecto se centra en el desarrollo de un algoritmo de visión por computador capaz de localizar en una imagen el perfil de un carril iluminado con luz estructurada. Este algoritmo es sólo una pequeña parte de un proyecto de mucho mayor alcance: el proyecto SegFer.

El proyecto SegFer trata del desarrollo de un dispositivo industrial que una vez fijado en la parte inferior un ferrocarril, es capaz de realizar la auscultación de vías para su mantenimiento e incluso conocer el estado dinámico del vehículo de forma que puedan predecirse eventos de descarrilamiento.

El ferrocarril, como medio de transporte, tiene numerosas ventajas frente a otros medios de transporte, a saber: capacidad, eficiencia, velocidad, seguridad, menor ocupación del suelo y menor contaminación. La sencilla electrificación de este medio de transporte tenderá a aumentar su uso a medida que disminuya la disponibilidad de combustibles fósiles económicamente rentables en las siguientes décadas. Es por esto que el desarrollo de un dispositivo capaz de mejorar la seguridad del ferrocarril y de reducir costes de mantenimiento de infraestructuras es vital para ayudar a la evolución y mejora de este medio de transporte.

En este capítulo se describirá el problema de la auscultación de vías y obtención de la dinámica del ferrocarril. Es imprescindible conocer este problema para entender la necesidad y finalidad del algoritmo desarrollado, así como sus requisitos y especificaciones de diseño.

1.1. Auscultación de vías

Esta sección está basada en una memoria del proyecto *Desarrollo de nuevas tecnologías de auscultación ferroviaria basadas en la simulación en tiempo real* [5].

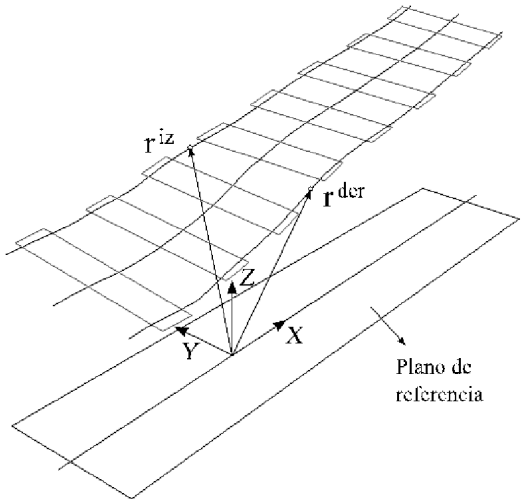

Figura 1.1: Medida de irregularidad de los carriles [5]

El ferrocarril requiere de una insfraestructura en buenas condiciones cuyo mantenimiento es muy costoso. El mal estado de las infraestructuras puede acabar reduciendo la eficiencia tanto energética como económica de este medio de transporte. La seguridad de este medio de transporte también puede verse afectada.

Un elemento con alto costo de mantenimiento de dicha infraestructura son las vías, que pueden extenderse hasta cientos de miles de kilómetros. Su auscultación (evaluar en las condiciones en las que se encuentra) es imprescindible para mantener la seguridad y el confort de este medio de transporte.

Por ejemplo, España tienen una red ferroviaria de alta velocidad de más de 3100 km, la segunda más grande por detrás de china. En España las vías han sido construidas en balasto, por lo que tienen un alto coste de mantenimiento, entre 100.000 y 150.000 euros al año por km de vía.

Un sistema que optimice el proceso de auscultación puede traducirse en un enorme ahorro en mantenimiento.

Durante la auscultación de las vías se mide la geometría de los carriles. Existen dos tipos de medidas en la auscultación: geométricas y dinámicas. Las geométricas miden las desviaciones con respecto a una geometría de referencia de los carriles. Las dinámicas miden la respuesta dinámica del vehículo ante dichas irregularidades.

La figura 1.1 muestra el sistema de referencia respecto al que se miden las irregularidades de la vía. Este sistema de referencia se sitúa en cada uno de los puntos de la línea media de la vía de referencia. Cada carril tendrá su propio vector de irregularidad con coordenadas x, y, z.

Los defectos de los carriles se clasifican por las desviaciones de la geometría de estos, de acuerdo a los siguientes parámetros:

1. **Alineamiento**. De define como $(y^{izq} + y^{der})/2$.
2. **Nivelación**. De define como $(z^{izq} - z^{der})/2$.
3. **Ancho de vía**. De define como $(y^{izq} - y^{der})/2$.
4. **Perfil vertical**. De define como $(z^{izq} + z^{der})/2$.
5. **Peralte**. En los tramos curvos de la vía se eleva la parte exterior respecto a la interior, creándose un ángulo respecto a la horizontal. Por tanto, el peralte puede ser interpretado como un "defecto"de nivelación intencionado en los tramos curvos. El peralte permite a los ferrocarriles recorrer las curvas con menor peligro.
6. **Alabeo**. Ritmo de cambio de la nivelación de los carriles.
7. **Desgaste de perfil**. Diferencia entre el perfil del carril real y el ideal.

Las irregularidades pueden ser puntuales (como el caso de la producida en una frenada brusca). Sin embargo, suelen tener asociada una longitud de onda característica. Conocer la magnitud de las irregularidades así como la frecuencia (longitud de onda) es de vital importancia para el diseño un sistemas de adquisición de datos cuya función sea medir estas irregularidades.

En la figura 1.2 puede observarse la magnitud de la nivelación y el alineamiento en función de la distancia recorrida por el vehículo. Es posible conocer la longitud de onda característica mediante el espectro de potencia (*Power Espectral Density*, PSD) calculado con la FFT (*Fast Fourier Transform*), este espectro se muestra en la figura 1.3.

Normalmente los defectos de alineamiento y nivelación suelen tener una PSD aproximadamente lineal, siendo mayor la amplitud del defecto para mayores longitudes de onda. Como puede observarse en las gráficas de la figura 1.3 las longitudes de onda de interés de las irregularidades es de 1 m debido a que por debajo de esta longitud de onda las irregularidades son muy pequeñas. Este valor de longitud de onda junto a la máxima velocidad de avance del ferrocarril permiten calcular la mínima frecuencia de trabajo del sistema de auscultación, y por tanto del algoritmo de visión descrito en este documento.

Para conocer las irregularidades de la vía el sistema embarcado procesa los datos de la dinámica del vehículo. Esta información puede ser utilizada con otros fines de gran importancia, como la prevención de accidentes o la medición del confort.

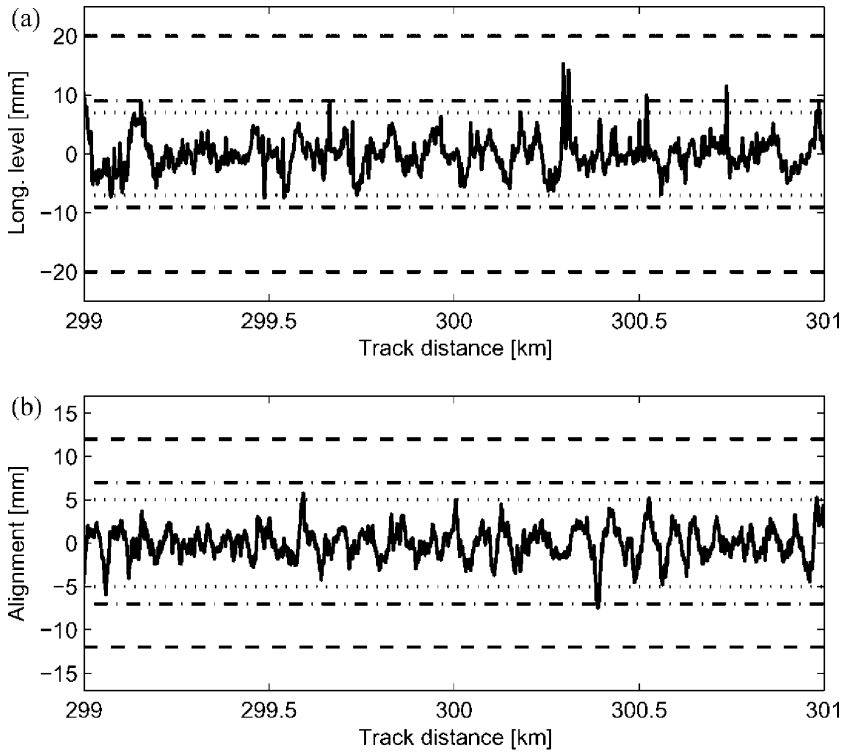

Figura 1.2: Medidas de los defectos en función de la distancia [6]

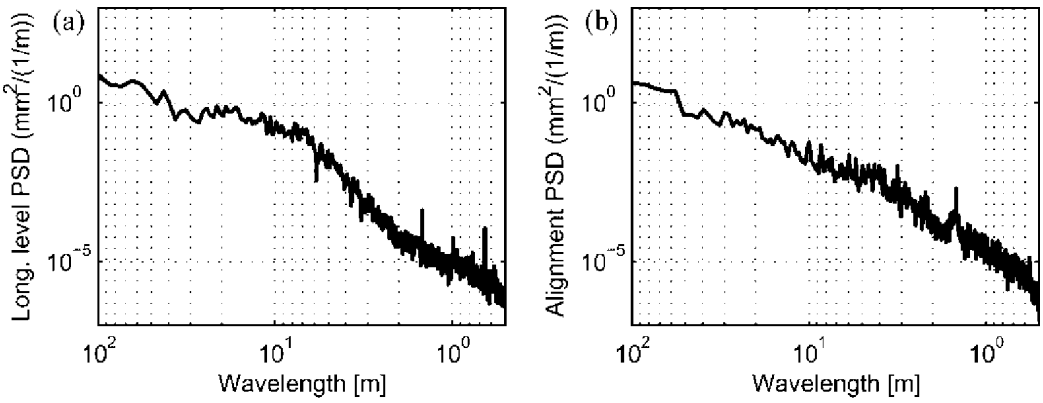

Figura 1.3: Espectro de potencia de los defectos [6]

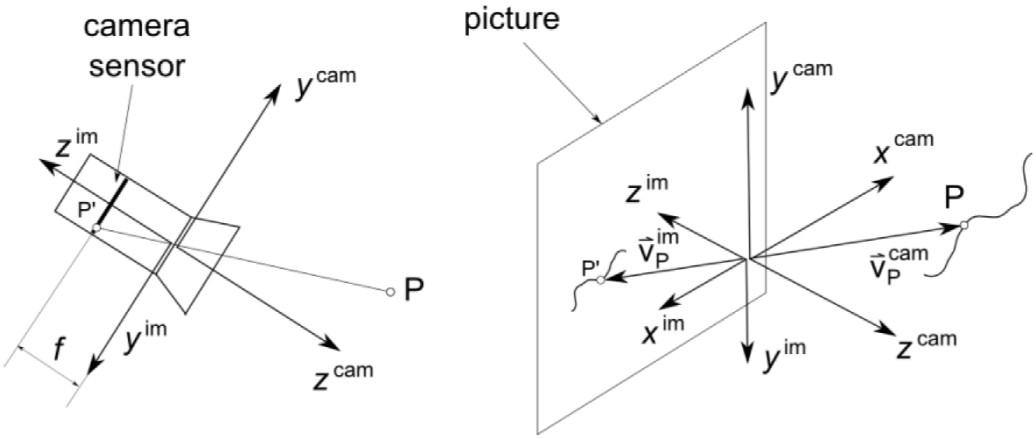

Figura 1.4: Cinemática de la visión artificial [4]

1.2. Formulación del problema

Esta sección introduce al lector en problema matemático de la auscultación de vías y la obtención del estado dinámico del ferrocarril. Se trata sólo de una introducción para que el lector sea capaz de comprender las entradas y salidas del algoritmo de visión desarrollado. Para un desarrollo matemático más detallado puede consultarse el trabajo de José Luís Escalona Franco, *Cinemática ferroviaria para su aplicación a sistemas de visión artificial* [4].

El problema que se plantea para la obtención de la dinámica vehicular y las irregularidades de la vía es fundamentalmente cinemático.

Se van a describir las ecuaciones utilizadas de visión artificial, lo que permite comprender con precisión las entradas y salidas del algoritmo desarrollado y cómo será utilizada la salida de este.

1.2.1. Cinemática de la visión artificial con proyección láser lineal

El uso de cámaras de vídeo es indispensable para una reconstrucción y caracterización precisa de las irregularidades de las vías y el desgaste del perfil de los carriles.

El modelo de cámara utilizado es el de cámara estenopeica o cámara oscura tal y como puede observarse en la figura 1.4.

La posición $\mathbf{v}_P^{cam}$ es la posición real de un punto del carril respecto al sistema de referencia de la cámara $< x^{cam}, y^{cam}, z^{cam} >$. La posición $\mathbf{v}_{P'}^{im}$ es la posición de la proyección del punto P en el plano sensor respecto al sistema de referencia de la imagen $< x^{im}, y^{im}, f >$. Estas dos posiciones pueden

relacionarse mediante la ecuación 1.1, deducida del modelo de cámara oscura de la cámara y los triángulos semejantes que forman las componentes de los vectores $\mathbf{v}_P^{cam}$ y $\mathbf{v}_{P'}^{im}$.

$$\frac{(\mathbf{v}_P^{cam})_z}{f} = \frac{(\mathbf{v}_P^{cam})_x}{(\mathbf{v}_{P'}^{im})_x} = \frac{(\mathbf{v}_P^{cam})_y}{(\mathbf{v}_{P'}^{im})_y} \tag{1.1}$$

donde $\mathbf{v}_P^{cam} = [(\mathbf{v}_P^{cam})_x, (\mathbf{v}_P^{cam})_y, (\mathbf{v}_P^{cam})_z]^T$ y $\mathbf{v}_{P'}^{im} = [(\mathbf{v}_{P'}^{im})_x, (\mathbf{v}_{P'}^{im})_y, (\mathbf{v}_{P'}^{im})_z]^T$ son las componentes de los puntos P, en la realidad, y P', en la imagen, respectivamente. La ecuación 1.1 puede reescribirse de la forma:

$$\frac{f}{(\mathbf{v}_P^{cam})_z}\mathbf{v}_P^{cam} = \mathbf{v}_{P'}^{im} \tag{1.2}$$

Las componentes x e y de $\mathbf{v}_{P'}^{im}$ son datos que se pueden extraer de la imagen grabada de la siguiente forma:

$$\begin{aligned}(\mathbf{v}_{P'}^{im})_x &= a(n_x - n_{x_0}) \\ (\mathbf{v}_{P'}^{im})_y &= b(n_y - n_{y_0})\end{aligned} \tag{1.3}$$

donde a y b son la anchura y altura de los píxeles de la cámara respectivamente. n_x y n_y son las posiciones x e y del píxel a procesar. Las constantes n_{x_0} y n_{y_0} indican la posición del centro de la imagen (donde se encuentra el origen del sistema de coordenadas de la imagen).

La ecuación 1.2 contiene dos ecuaciones independientes para el cálculo de las tres componentes de $\mathbf{v}_P^{cam}$ (sistema indeterminado). De la imagen grabada por una cámara (2D) no se puede extraer la posición de los puntos que aparecen en el espacio (3D). Una técnica común para atajar este problema es utilizar iluminación estructurada. Este tipo de iluminación se sirve de la proyección de puntos, franjas o rejillas sobre la superficie de trabajo. En función de cómo se deforme este patrón de luz sobre la superficie se puede detectar las singularidades de la pieza objeto de análisis.

En el proyecto de auscultación de vías se ha conseguido la iluminación estructurada utilizando un láser de proyección lineal que genera un haz plano de luz además de la cámara, como puede observarse en la figura 1.5. Este tipo de sistema son comunes en la industria, incluso existen productos comerciales desarrollados [10].

Gracias al uso del láser se tiene la información de que todos los puntos iluminados por el láser están contenidos en un plano, es decir, que los vectores posición de dichos puntos cumplen la siguiente condición:

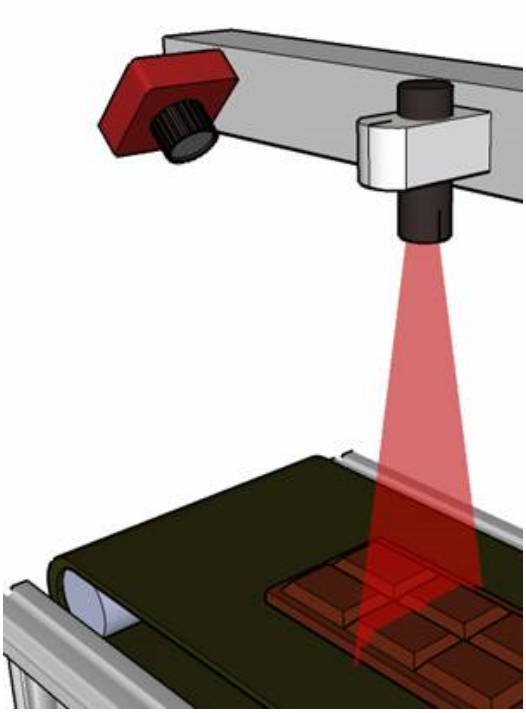

Figura 1.5: Reconstrucción con cámara y láser de proyección lineal [3]

$$a(\mathbf{v}_Q^{cam})_x + b(\mathbf{v}_Q^{cam})_y + c(\mathbf{v}_Q^{cam})_z - d = 0 \Rightarrow [a,b,c] \begin{bmatrix} \mathbf{v}_Q^{cam})_x \\ \mathbf{v}_Q^{cam})_y \\ \mathbf{v}_Q^{cam})_z \end{bmatrix} - d = 0$$
$$\Rightarrow \triangle^T \mathbf{v}_Q^{cam} - d = 0 \quad (1.4)$$

donde a, b, c, y d son los parámetros de la ecuación general del plano del láser y $\mathbf{v}_Q^{cam}$ es el vector posición de los puntos Q iluminados por el láser en el sistema de referencia de la cámara. Los puntos Q forman el perfil del carril cuya posición pretende conocerse.

Cuando la cámara y el láser son estacionarios, las dos ecuaciones independientes que se extraen de la ecuación 1.2 junto con la ecuación 1.4 dan lugar a un sistema determinado de ecuaciones (tres ecuaciones y tres incógnitas) del que se puede extraer la posición (3D) $\mathbf{v}_P^{cam}$ de los puntos grabados en la imagen e iluminados por el láser. El problema que se quiere resolver en el presente proyecto de auscultación es mucho más complejo que este porque:

1. Ni la cámara ni el láser son estacionarios, sino que están fijos a un vehículo en movimiento, por lo que la orientación del objeto grabado respecto a la cámara y el láser puede cambiar con el tiempo.

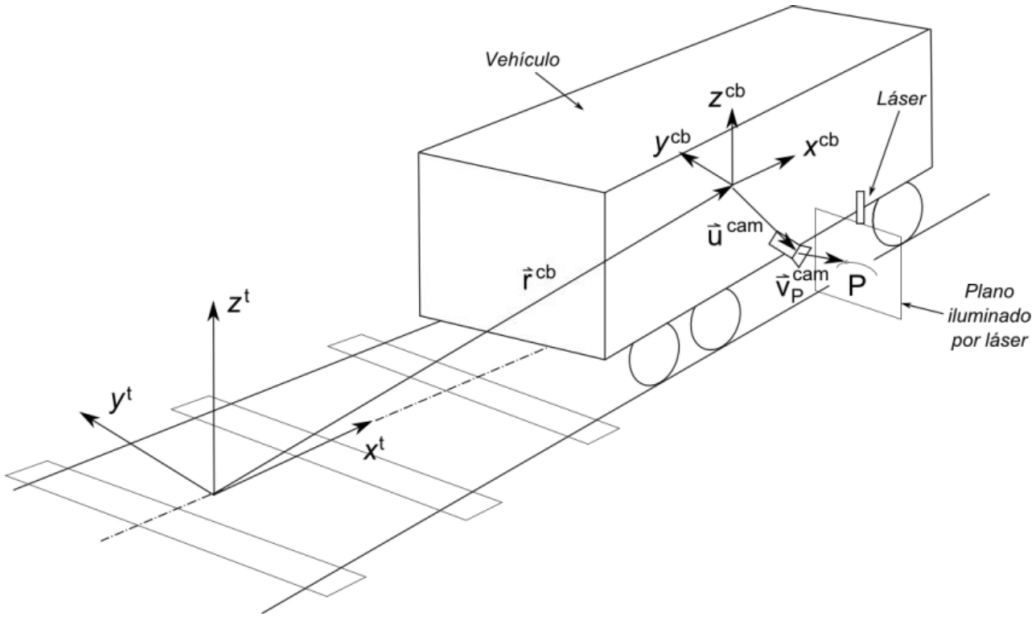

Figura 1.6: Cinemática del vehículo instrumentado [4]

2. El objeto cuyas coordenadas se pretende identificar no esta fijo en el espacio ni con respecto al vehículo y además tiene geometría cambiante. Este objeto es la sección transversal del carril ferroviario.

3. A partir de la imagen grabada se pretende obtener, además de la geometría de la sección transversal del carril, la medida de los movimientos del vehículo con respecto a la vía. Además, debe tenerse en cuenta que el desplazamiento de los carriles respecto al vehículo grabados por la cámara pueden deberse a la propia dinámica del vehículo o a las irregularidades de los carriles. Ambos problemas están acoplados.

Los aspectos matemáticos relacionados con estos movimientos y geometrías se exponen en la figura 1.6. En ella se muestran un único láser y una cámara. Sin embargo, el vehículo tendrá dos sistemas de visión como este, de manera que habrá uno para cada carril de la vía.

En el siguiente apartado se combinan las ecuaciones de la cinemática de visión con las de la cinemática vehicular, para dar con el sistema de ecuaciones del problema completo de auscultación y obtención de la dinámica del vehículo. Sin embargo, ya es posible comprender cual es la función del algoritmo de visión desarrollado en el presente proyecto.

Figura 1.7: Fotograma del perfil de la cabeza del carril

1.2.2. Objeto del algoritmo

Obsérvese como los datos de entrada de toda la cinemática de visión (conocidos los parámetros intrínsecos y extrínsecos de la cámara y el láser) son las coordenadas de un punto en la imagen obtenida por la cámara. Estas aparecen en la ecuación 1.3 como n_x y n_y. Este punto debe ser el punto central del perfil que es tomado como referencia para el cálculo de las irregularidades de las vías. También es necesario conocer la orientación del carril. Este punto será calculado por el algoritmo de visión desarrollado, teniendo como única entrada la imagen tomada por las cámaras. La cinemática de visión permite la reconstrucción 3D del perfil del carril respecto ala sistema de referencia de la cámara, a partir de la información que se obtiene mediante el algoritmo de visión que analiza las imágenes.

En la figura 1.7 podemos ver un ejemplo de una imagen grabada por una de las cámaras. En la imagen se ve un fondo oscuro con una nube de puntos de alto nivel de intensidad que corresponde al reflejo del láser de proyección lineal sobre el carril. La oscuridad de la imagen se debe a que son tomadas con muy baja exposición. Esto facilita la posterior umbralización durante la segmentación de la imagen realizada por el algoritmo.

El objetivo de este documento es el de describir el algoritmo desarrollado capaz de realizar precisamente esta tarea: obtener la posición y orientación del perfil en la imagen.

El problema completo a resolver en el proyecto SegFer es recogido por un complejo sistema de ecuaciones de 12 ecuaciones y 15 incógnitas (indeterminado). No es objeto de este documento abordarlo. Sin embargo, es importante resaltar que el algoritmo de visión no será el único que será ejecutado por el

ordenador de auscultación. Será necesario ejecutar también el algoritmo que resuelve el sistema de ecuaciones completo.

1.3. Equipo de Auscultación

A continuación se presenta brevemente el equipo de auscultación utilizado en el proyecto SegFer. Es importante conocerlo porque es el utilizado en los ensayos realizados para obtener los resultados experimentales del algoritmo mostrados en el capítulo 3.

El sistema está formado por un ordenador de cómputo (PC de auscultación), dos sistemas de visión, una IMU de 3 ejes y un acelerómetro piezoeléctrico de un eje.

1.3.1. PC de auscultación

La elección del PC de auscultación es de suma importancia. Idealmente sería escogido un PC cuya CPU tenga un elevado rendimiento y la mayor capacidad de paralelización posible, de forma que puedan ejecutarse algoritmos de visión lo más rápidamente posible.

Por otro lado, es imperativo un PC de bajo coste, ya que se persigue un equipo completo de auscultación cuyo coste sea reducido para hacerlo competitivo en la industria. Además, una mayor potencia se traduce en un mayor consumo, lo que puede ser una limitación en aplicaciones en las que no es posible acceder a una toma de alimentación en el ferrocarril.

Por estas razones el PC de auscultación elegido es de gama media con las siguientes características:

1.3.2. Sistema de percepción

El sistema de percepción completo consta de los siguientes elementos: cámara de vídeo, lente, y un láser de proyección lineal.

1.3.2.1. Cámara de vídeo

La cámara elegida [7] para esta función es la MQ003CG-CM de XIMEA. Se trata de una cámara $CMOS$ a color con una resolución de 648x488 píxeles. Es una cámara muy pequeña (26x26x25 mm), como puede observarse en la figura 1.8. Con un peso de sólo 26 gramos y con la capacidad de hacer más de 500 fotogramas por segundo (FPS en adelante), es ideal para el proyecto.

Figura 1.8: Modelo de cámara

La cámara se comunica mediante una conexión USB (3.0). Dado el enorme flujo de datos transmitido por la cámara el fabricante recomienda conectar una sola cámara por tarjeta USB.

Para recibir y procesar las imágenes, el fabricante ofrece *xiAPI*, una completa librería de funciones para la configuración, control y lectura de imágenes de la cámara desde nuestra propia aplicación. Esta API es además compatible con algunas de las más famosas librerías de visión artificial y procesado de imágenes como **OpenCV**.

1.3.2.2. Lente

La cámara de vídeo es una matriz de sensores ópticos con un sistema electrónico que controla, lee y transmite la información que llega a cada uno de los elementos de dicha matriz. El objetivo es generar fotogramas de un objeto (el perfil del carril iluminado por el láser de proyección lineal), y con la mayor precisión posible, por lo que también es necesario que al sensor de la cámara llegue la luz proyectada por dicho objeto. Para ello necesitamos la óptica o lente.

La lente utilizada es la Fujinon HF12.5HA-1B, con una distancia focal de 12,5 *mm*. Esta lente fue recomendada por el fabricante para el enfoque correcto de objetos entre 30 y 45 *cms*, distancia existente entre la cámara y el perfil de la cabeza del carril iluminado por el láser. La lente permite un enfoque preciso de manera manual con una apertura relativa máxima de 1:1.4. También permite cambiar la apertura del iris con un rango F1.4-F16.

1.3.2.3. Láser de proyección lineal

El láser de proyección lineal permite crear la iluminación estructurada. El láser de proyección lineal elegido tiene una potencia de 50 mW alimentado con una tensión de 5 V. Genera un haz pano láser a 120º desde su cabezal. El láser puede ser ajustado para conseguir una proyección nítida y de un espesor de aproximadamente 1 mm en objetos a distinta distancia (el rango es de varios metros).

1.3.2.4. Inertial Measurement Unit (IMU)

Las IMU serán las encargadas de proporcionar la mayoría de los datos necesarios para conocer la dinámica del vehículo.

La IMU elegida es la 3DM-GX4-25 [9]. Estos sensores integran un acelerómetro, un giróscopo y un magnetómetro, todos ellos de 3 ejes.

El acelerómetro tiene un rango máximo de $\pm 16g$ y una resolución de $< 0,1mg$ con un ancho de banda máximo de 225 Hz.

El giróscopo tiene un rango máximo de $\pm 900^{o}/s$ y una resolución de $<$0,008º$/s$ con un ancho de banda máximo de 250 Hz.

El magnetómetro no va a ser utilizado en este proyecto.

Tanto el acelerómetro como el giróscopo tiene una máxima frecuencia de envío de paquetes de datos de 1000 Hz.

Estos sensores tienen además la posibilidad de estimar medidas mediante la aplicación del Filtro de Kalman. Sin embargo, esta funcionalidad no va a ser utilizada en el proyecto SegFer.

1.3.2.5. Acelerómetro piezoeléctrico

Como se ha visto anteriormente la frecuencia de adquisición depende de la velocidad del ferrocarril y de la longitud de onda de las irregularidades. Las gráficas muestran que las irregularidades por debajo de $1m$ de longitud de onda tiene magnitudes menores, y el sistema de adquisición en general no ha sido diseñado para captar estas irregularidades. Sin embargo, empresas de la industria han mostrado interés por estas irregularidades de menor longitud de onda (corrugación), por lo que finalmente se ha añadido al sistema un acelerómetro capaz de captar frecuencias mayores.

Los acelerómetros MEMS como la IMU utilizada en el proyecto tienen limitada su frecuencia de adquisición a cientos o pocos miles de medidas por segundo. Es por tanto necesario utilizar acelerómetros de otra tecnología, como los acelerómetros piezoeléctricos.

El acelerómetro piezoeléctrico de un sólo eje utilizado en el proyecto es capaz de responder ante aceleraciones de alta frecuencia provocadas la corrugación.

Capítulo 2

Diseño del algoritmo

La visión por computador se diferencia de otras ramas de la ingeniería por no tener una solución cerrada excepto en aplicaciones específicas comunes. El diseño de un nuevo algoritmo de visión suele pasar por la revisión de la bibliografía y la posterior aplicación de distintos métodos ya existentes para la obtención de un algoritmo que cumpla las especificaciones. En ocasiones puede ocurrir que las soluciones existentes no sean extrapolables a una aplicación completa, por lo que es necesaria la creación de un nuevo algoritmo de visión. Este es el caso del algoritmo desarrollado en este proyecto.

Como se verá en el siguiente apartado, las especificaciones de velocidad de cómputo son más restrictivas que en la mayoría de aplicaciones de vídeo, por tanto, en el diseño se han tomado métodos, ideas y conceptos de la bibliografía que permitan un análisis de imágenes lo más rápido posible.

A la hora de implementar el algoritmo diseñado, la idea inicial era comenzar con pruebas de funcionamiento en Matlab, y una vez probado ser implementado en C++ (para ser ejecutado en procesadores con arquitectura común) o en un FPGA, para comprobar si cumple las especificaciones de velocidad de cómputo.

2.0.1. Localización del perfil en la imagen

El objetivo principal del algoritmo es la de localizar dentro la imagen el perfil del carril dentro de la imagen. Esto se traduce en 3 datos de salida:

- Coordenadas x e y de un punto característico del perfil.
- Orientación del perfil (ángulo respecto al eje horizontal de la imagen).

Se entiende por punto característico un punto a partir del cual (y conociendo la orientación) pueda ser reconstruido el perfil. Por ejemplo, la cabeza

del perfil, que puede ser observada en la figura 2.1, está formada por circunferencias de distinto radio que intersectan en ciertos puntos. Conocida las coordenadas de cualquiera de estos puntos y la orientación del perfil respecto a la horizontal es posible dibujar (reconstruir) el perfil.

2.1. Planteamiento del problema

La selección del punto característico es vital ya que condiciona tanto el diseño del propio algoritmo de visión como la posición de los elementos de visión (cámara y láser). Estos deben ser colocados de forma que la zona del perfil que contiene al punto característico sea grabado con nitidez e iluminado por el láser de proyección lineal.

Existen distintas consideraciones a tener en cuenta a la hora de elegir el punto característico del perfil:

- El punto característico debe encontrarse en la parte superior del perfil(cabeza). El pie del perfil suele estar enterrado y no es posible verlo. También pueden impedir su visión elementos de la vía como sujeciones, o incluso rocas u hojas del entorno, como puede observarse en la imagen 2.2.

- Cualquier punto de la cabeza del perfil no puede ser escogido. Se va a distinguir entra la zona interior y la zona exterior del carril. La zona interior de la cabeza del perfil es la que entra en contacto con las ruedas del ferrocarril. La zona interior se desgasta con el tiempo por el rozamiento con el ferrocarril. Sin embargo, el desgaste de la cabeza del carril tambien debe ser grabado de alguna forma, ya que es una información muy valiosa que debe acompañar a la información de la irregularidad de la vía.

- El sistema de visión, tal y como está concebido (con una cámara y un láser para cada carril) sólo es capaz de grabar uno de los lados del carril, o el lado interior o el lado exterior, un ejemplo se puede observar en la figura 2.3. Otra posibilidad es colocar la cámara y el láser justo encima del carril, por lo que se graba toda la parte central de la cabeza, perdiéndose parte de la zona lateral, como puede observarse en la figura 1.7. Es posible grabar el carril completo pero es necesario duplicar el número de cámaras y láser de proyección lineal, encareciendo y complicando el sistema.

- La reconstrucción del carril en verdadera magnitud a partir de la información que proporciona el algoritmo tras analizar la imagen es realizada

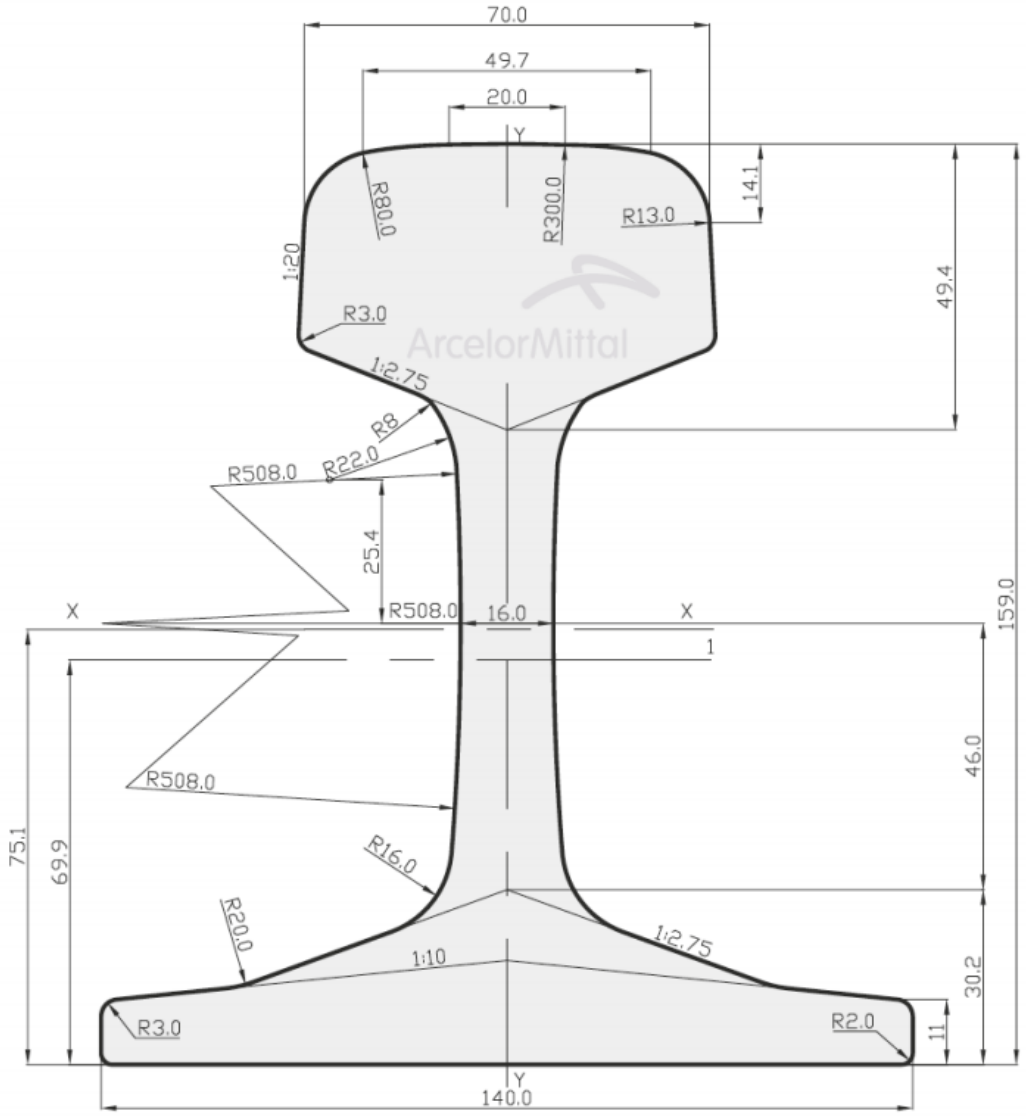

Figura 2.1: Perfil del carril UIC54 [12].

Figura 2.2: Fotografía de una vía.

utilizando las expresiones anteriormente vistas en la sección 1.2.1. Ya que la cámara está girada (la curva del perfil no es grabada justo desde el frente, sino que existe un ángulo entre el plano del perfil y el plano de la imagen) la curva del perfil aparecerá deformada en la imagen. Por ello, es imprescindible conocer la posición relativa entre el plano de la imagen y el plano del perfil del carril. Dicho ángulo no es fijo, sino que va variando con el movimiento del ferrocarril o incluso con las irregularidades del carril, que son justo los datos por obtener del sistema de auscultación. Para poder tener en cuenta estos cambios del ángulo entre el plano de la imagen y el del perfil del carril sería necesario un proceso iterativo. Sin embargo, se ha decidido despreciar este cambio de ángulo, ya que se trata de cambios realmente pequeños y no tienen un efecto perceptible sobre la imagen.

Por tanto, se van a tomar los ángulos (realmente son dos ángulos) entre el plano de la imagen y el plano del perfil del carril iguales a los medidos entre el plano de la imagen y el plano de proyección láser (plano de proyección láser y plano del perfil del carril coincidentes).

Finalmente fue decidido que se grabaría la parte interior del carril, (la parte que sufre desgaste) por las siguientes razones:

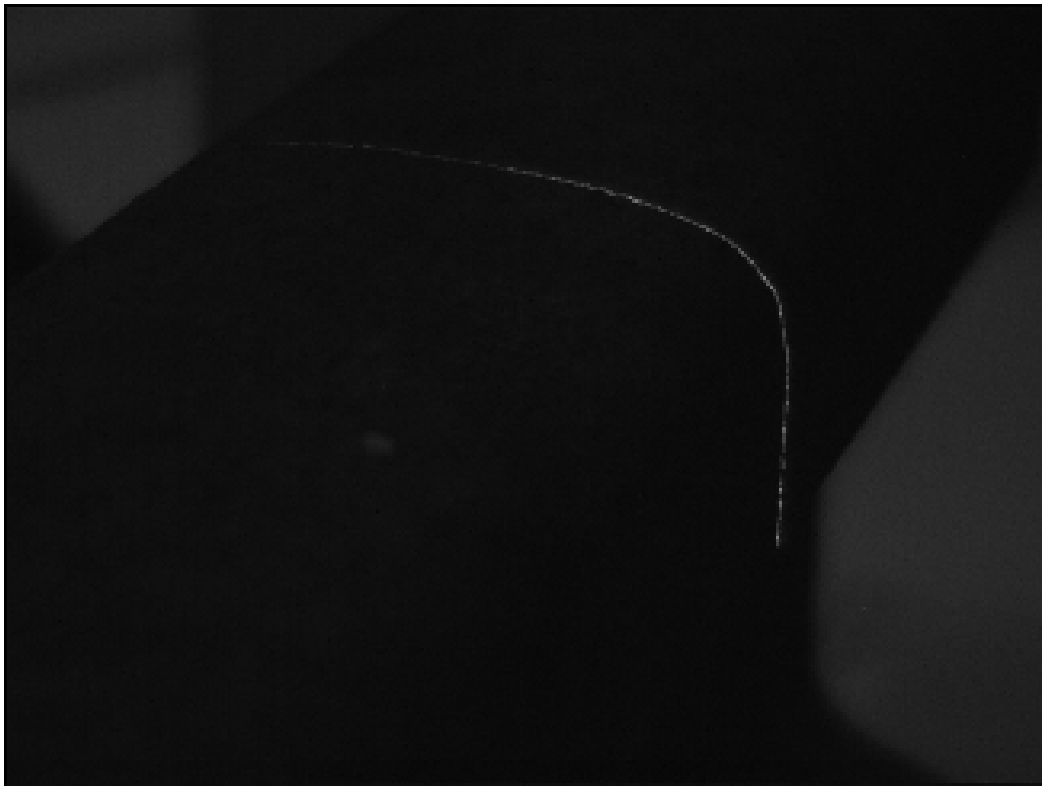

Figura 2.3: Imagen de la parte interior del perfil del carril.

- Como se ha mencionado anteriormente, el desgaste no afecta a la parte final de la recta de los laterales de la cabeza del perfil. Es es así porque el punto de contacto rueda-carril nunca llega a esta parte del perfil. Esta es por tanto una buena zona en la que elegir el punto característico del perfil. No se puede tomar como punto de referencia (punto característico) para posicionar el carril un punto que sufra desgaste.

- Puede ser obtenida la información del desgaste.

- El diseño mecánico del dispositivo que contiene los elementos de visión es mas simple si este es colocado en la parte interior.

Por esto, se ha elegido como punto característico el punto final de la recta de la cabeza del perfil de la parte interior. La propia recta (su parte final) facilitará la obtención de la orientación del perfil, a partir del cálculo de la pendiente de la recta.

2.1.1. Especificaciones

Algunas de las especificaciones del sistema de adquisición son heredadas por el algoritmo de visión.

La primera especificación hace referencia a la velocidad de cómputo del algoritmo. Como se va a justificar más adelante, el sistema de adquisición en su conjunto debe ser capaz de adquirir medidas a una frecuencia de $250Hz$. Se trata de una frecuencia de adquisición alta y el sistema de adquisición no ofrece la posibilidad de guardar las imágenes provenientes de la cámara a dicha frecuencia por cuestiones de capacidad de almacenamiento y velocidad de grabación en disco. Es por tanto imperativo que el algoritmo sea capaz de trabajar en tiempo real, de forma que analice la imagen y sólo sea necesario guardar el resultado del análisis (coordenadas del punto característico y la orientación del perfil).

Esta es la especificación que más complica el diseño del algoritmo. Los algoritmos de análisis de vídeo suelen trabajar a 30 o 60 Hz, por lo que las soluciones encontradas en otras aplicaciones son complicadas de extrapolar a esta aplicación. Existen un gran número de métodos y algoritmos que permiten encontrar formas en una imagen. Por ejemplo, la transformada de Hough permite encontrar rectas o circunferencias, sin embargo es complicado que puedan trabajar a la frecuencia requerida en este proyecto.

La selección de la frecuencia de muestreo no es arbitraria, sino que ha sido calculada en función de la velocidad máxima que pueden alcanzar los ferrocarriles con los que se va a probar el dispositivo, y la longitud de onda

mínima de las irregularidades. En el capítulo anterior se vió como las irregularidades de mayor magnitud se producían para una longitud de onda mayor que $1m$. Las irregularidades de menor longitud de onda (corrugación) son captadas por un acelerómetro piezoeléctrico como fue descrito en el capítulo anterior. La velocidad máxima de los ferrocarriles dónde va a funcionar esta versión del dispositivo es de $70km/h$ ($19{,}44m/s$). Con estos datos y la fórmula que relacciona la frecuencia temporal con la longitud de onda es posible calcular la máxima frecuencia con la que el sistema podrá encontrar las irregularidades:

$$f = \frac{v}{\lambda} = \frac{19{,}44m/s}{1m} \simeq 20Hz$$

Por el Teorema de Nyquist es conocido que para reconstruir una señal la frecuencia del adquisición debe ser al menos el doble que el ancho de banda de la señal. En la práctica no se usa un factor dos, sino uno más alto. En este caso se ha usado un factor algo mayor de 10, por tanto la frecuencia de funcionamiento queda por encima de $200Hz$. Finalmente se tomó la decisión de tomar el valor de $250Hz$ como frecuencia de adquisición del sistema.

Otra de las especificaciones es la robustez. El sistema completo debe conocer en cada instante la posición del perfil del carril en la imagen para ser capaz de realizar el cálculo de las irregularidades y la dinámica vehicular. Por tanto, el algoritmo no puede fallar en ningún instante.

2.1.2. Hipótesis de partida

Las hipótesis de partida forman el contexto bajo el que el algoritmo puede funcionar correctamente de forma robusta. A continuación se listan las hipótesis y su justificación:

1. Existe un contraste suficiente entre el fondo y los píxeles iluminados en la imagen por el láser de proyección lineal como para poder ser diferenciados. Esto se consigue con un bajo tiempo de exposión. De esta forma sólo los píxeles iluminados con una gran intensidad tomarán tonos claros en la imagen, el resto permanecerán oscuros.

   Tras las pruebas realizadas se confirma que esta hipótesis se va a cumplir excepto ante exposición directa a la luz solar durante un día soleado. Esto no significa que esta hipótesis no se cumpla en días soleados, ya que los carriles generalmente quedan bajo el tren, por lo que no reciben directamente radiación del sol. En cualquier caso, se planean realizar cambios en el sistema de visión para que el algoritmo sea robusto en cualquier situación (ver capítulo 4).

2. En la imagen es visible una curva como se observa en la imagen 2.3, en la que se puede apreciar la parte recta del lado interior de la cabeza del carril, la curva cerrada y el principio de la parte superior de la cabeza del carril.

3. La curva del perfil del carril está correctamente enfocada y es delgada y continua. Puede observarse un ejemplo de curva del perfil del carril con demasiado espesor en la imagen 2.4 (curva del perfil del carril difuminado). Aunque pueda parecer que este espesor es debido a que la proyección lineal del láser no está enfocada, esto no tiene porqué ser cierto, probablemente la línea proyectada sea muy fina. En grabaciones en movimiento, durante el tiempo que la luz entra en el sensor de la cámara, un mismo punto en la realidad se proyecta en distintos píxeles, por lo que la intensidad de luz recibida de un mismo punto se reparte entre varios píxeles y la forma del objeto se deforma, produciéndose el conocido efecto de desenfoque de movimiento. Sin embargo, este efecto puede evitarse reduciendo el tiempo de exposición a la luz del sensor de la cámara. Esta es precisamente la forma de configurar las cámaras en este proyecto, por este y otros motivos (alta intensidad en la imagen únicamente de los píxeles de la curva del perfil del carril).

4. El tamaño de la curva del perfil en la imagen es aproximadamente constante.

5. Existe ruido producido por la adquisición de la imagen.

6. Pueden aparecer nubes de píxeles iluminadas o curvas con forma arbitraria. En ningún caso estas curvas tendrán el tamaño y la forma que la curva del carril, ni estarán demasiado cerca de este. Las nubes de píxeles pueden aparecer por elementos del fondo iluminados de la escena grabada por las cámaras o por reflejos de objetos metálicos en la vía. En la cabeza del perfil no debe existir ninguno de estos objetos por lo que no es posible que aparezcan reflejos cerca de la curva del perfil del carril.

   Esta hipótesis puede ser relajada si la información de la imagen anterior es utilizada (ver 4).

7. Aunque aparezcan reflejos con forma de curva continua, ninguna tendrá el espesor, el tamaño o la forma de la curva del perfil del carril en la imagen.

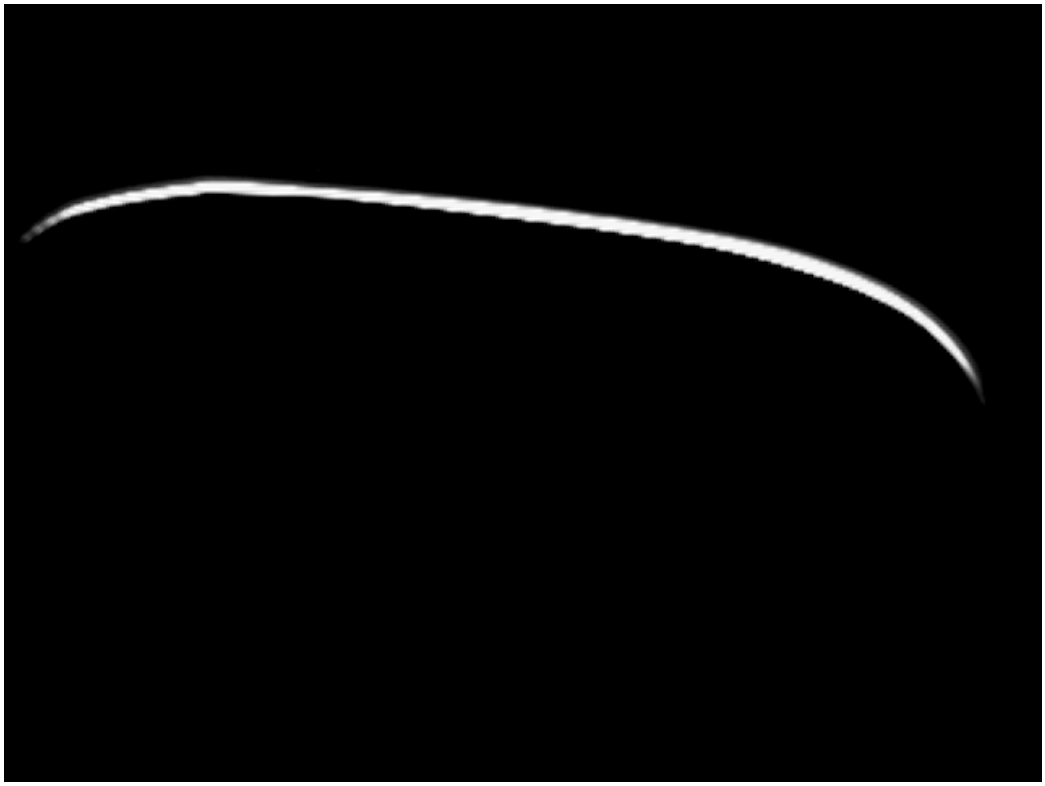

Figura 2.4: Fotografía del perfil del carril iluminado con el tiempo de exposición ligeramente elevado (perfil difuminado por movimiento).

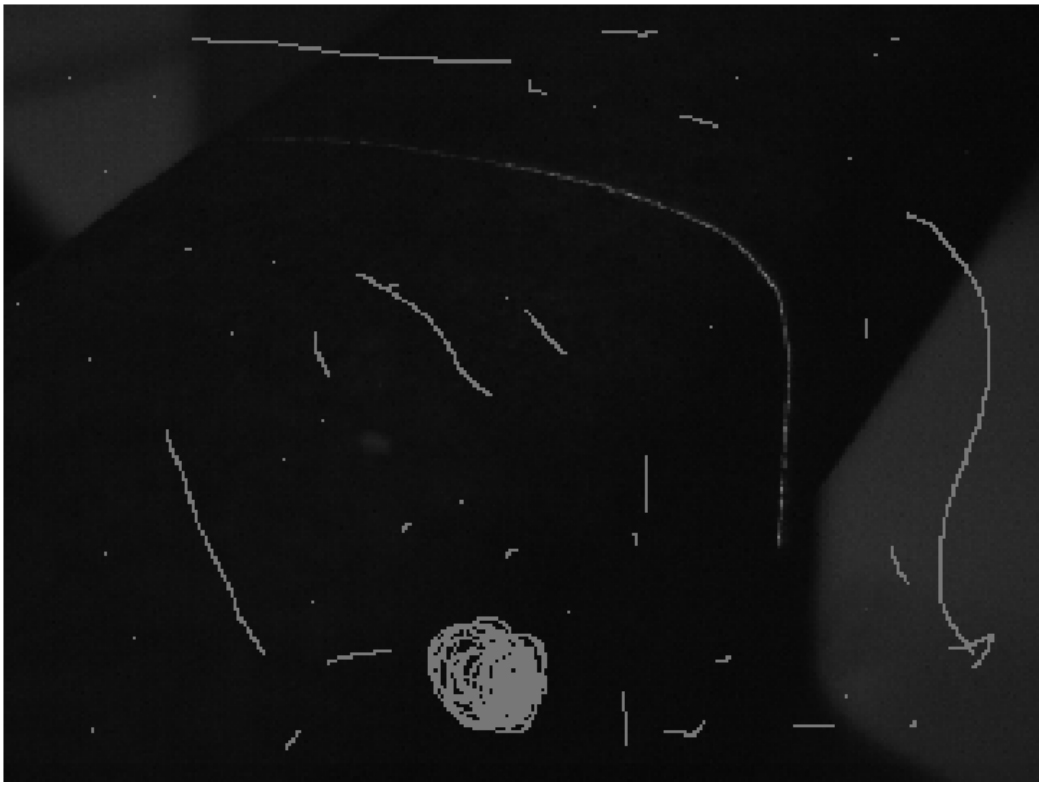

Figura 2.5: Imagen de prueba del algoritmo de visión.

2.2. Solución propuesta

A continuación se presenta el algoritmo de visión desarrollado. Este pretende satisfacer las especificaciones considerando las hipótesis de partida para conseguirlo. Para mostrar el funcionamiento del algoritmo, se va ha utilizado la imagen de la figura 2.5. Se trata de una imagen real tomada por el sistema de visión, retocada con software para añadirle de forma exagerada píxeles iluminados (ruido), curvas y otras formas. El objetivo de elegir esta imagen para las pruebas es la de demostrar la robustez del algoritmo y de describir su funcionamiento en todos los casos posibles.

2.2.1. Descripción del algoritmo

El algoritmo se divide en una serie de pasos sucesivos. A continuación se describen y justifican todos los pasos.

2.2.1.1. Paso 1. Fragmentación y primera selección de recuadros

La tarea que cumple el algoritmo de visión desarrollado es en concepto sencillo: encontrar en una imagen monócrona con fondo oscuro una forma muy definida y conocida, y establecer su posición en la propia imagen. Sin embargo, lo que podía complicar la implementación del algoritmo es la especificación de frecuencia de ejecución. Conseguir una tasa de 250 imágenes por segundo ha sido la principal preocupación durante el diseño del algoritmo.

Este primer paso intenta ayudar en ese sentido. Un concepto básico para conseguir algoritmos más rápidos en visión se basa en reducir al máximo el número de veces que se recorre la imagen. Por ello en este paso se fragmenta la imagen, se divide en distintos recuadros. La imagen contendrá muchos recuadros que sería inútil recorrer una y otra vez ya que no contienen nada de información, solo el color oscuro del fondo y ruido. Diferenciar los recuadros con información útil de los que no la contienen, y guardar en memoria estos recuadros que serán utilizados en el futuro es un primer paso para reducir en gran medida el número de operaciones que realizará el algoritmo, y por tanto aumentará la velocidad en la que éste se ejecuta.

En la figura 2.6 se presenta el resultado de aplicar a la imagen de prueba el paso 1. Los recuadros pintados en rojo son los que han cumplido los requisitos para ser guardados. La selección de los recuadros se basa en el cálculo de ciertas métricas en cada uno de ellos y su inclusión o no en la lista de posibles recuadros que contiene a la curva del perfil en función de la aplicación de unos umbrales a estas métricas. Las métricas son el contraste y el nivel de intensidad (nivel de blanco total):

1. **Contraste mínimo**: Un recuadro que contiene al perfil debe tener un contraste mínimo. Si no probablemente se trate de un recuadro del fondo.

2. **Nivel de blanco total máximo**: Se entiende por nivel de blanco total la suma de la intensidad de todos los píxeles del recuadro. Un recuadro que contiene al perfil contiene un fondo negro con una delgada curva de píxeles blancos o grises que lo atraviesa, por tanto el nivel de blanco no puede ser mayor que cierto valor. En caso de un nivel de blanco demasiado alto podría tratarse de un reflejo de un objeto o similar, por tanto ese recuadro no será guardado.

Estos dos umbrales forman parte de una lista de parámetros del algoritmo. Ajustar estos parámetros es fundamental para un correcto funcionamiento de éste.

Los recuadros que son guardados serán analizados en los siguientes pasos. El objetivo final es una lista que guarde únicamente los recuadros que con-

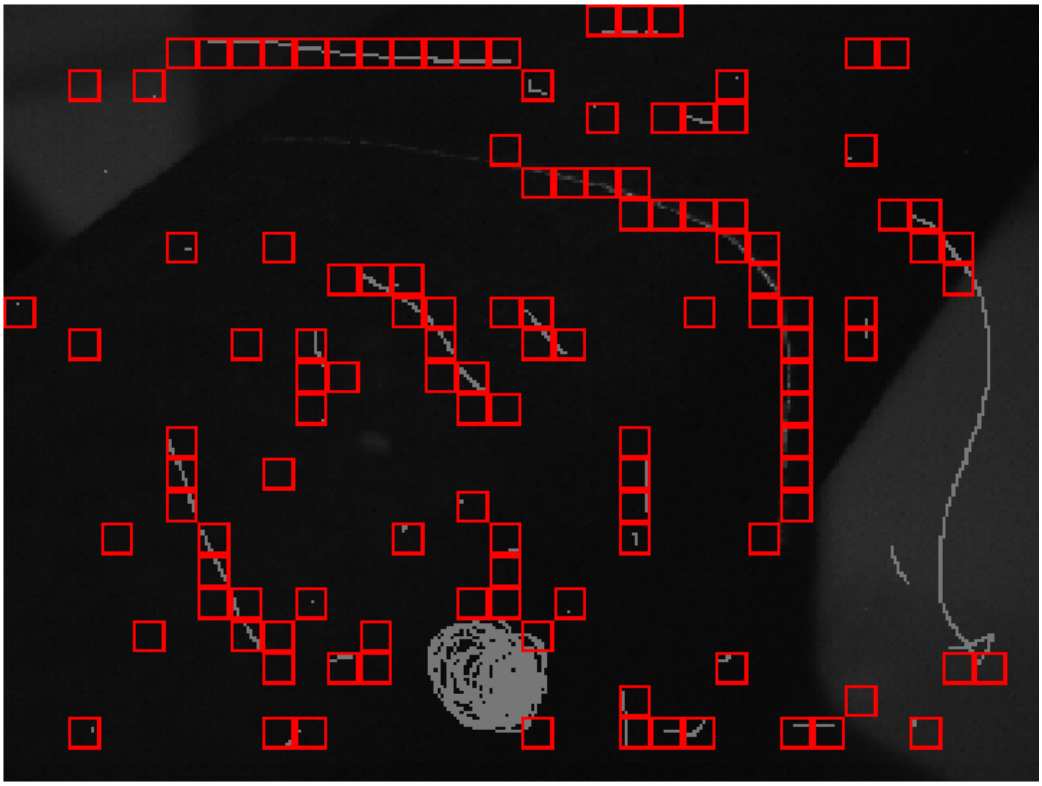

Figura 2.6: Paso 1. Fragmentación y primera selección de recuadros (color rojo).

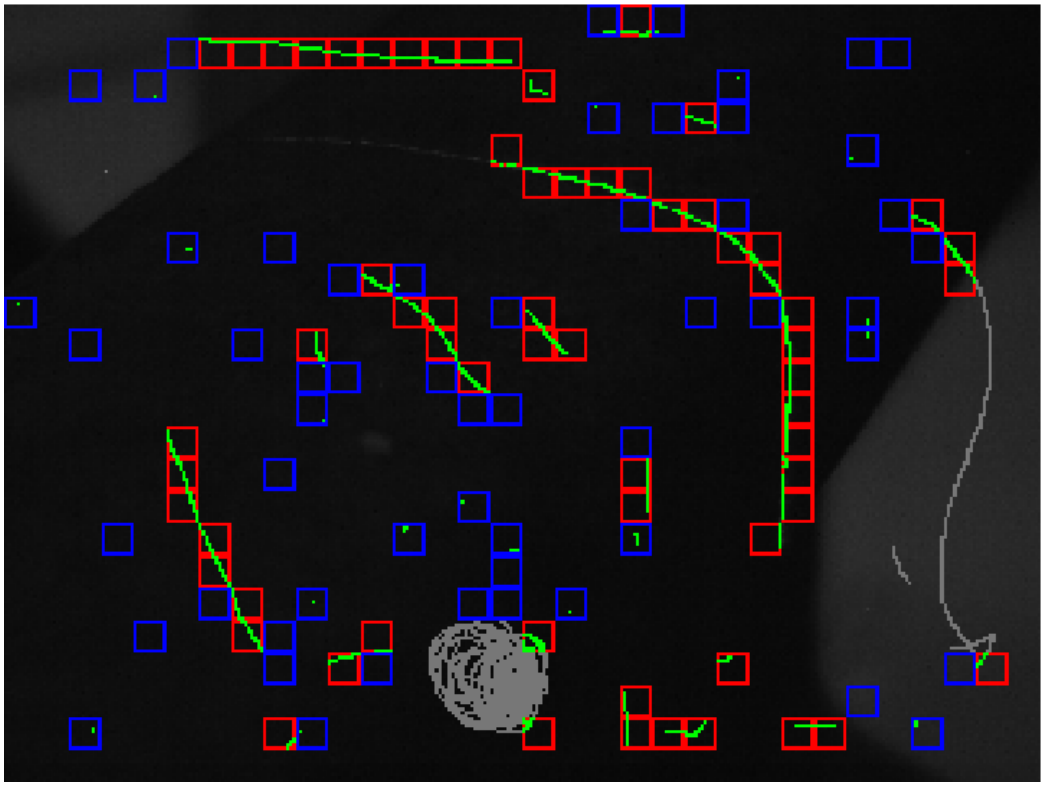

Figura 2.7: Paso 2. Umbralización y descarte de recuadros por número de píxeles (recuadros azules).

tienen al perfil, ninguno más. De esta forma se podría con relativa facilidad localizar el perfil con la mayor precisión posible, como se verá más adelante.

2.2.1.2. Paso 2. Umbralización y descarte de recuadros por número de píxeles

Del paso 1 sólo han quedado recuadros con algunos píxeles iluminados diferenciables del color oscuro de fondo del recuadro. En este paso se van a guardar los píxeles del recuadro que potencialmente pertenecen al carril (para cada uno de los recuadros que han sido guardados en el paso 1), haciendo de nuevo un filtrado de ellos. Para ello se hace pasar a los píxeles del recuadro por un umbral de intensidad. Los píxeles que pasen el umbral serán guardados y asociados a su recuadro para ser utilizados en pasos posteriores.

El resultado de este paso se puede observar en la figura 2.7. Los píxeles verdes son los píxeles cuya intensidad es mayor que la del umbral elegido.

En este mismo paso se puede aplicar otro requisito a los recuadros guar-

dados. El número de píxeles en el recuadro que han pasado el umbral debe ser mayor que cierto número. Los recuadros en los que sólo un muy reducido número de píxeles que han pasado el umbral no pueden contener al carril. Probablemente se trata de un recuadro con un par de píxeles que ha tomado un alto nivel de intensidad por el ruido, o se trata del final de alguna curva. Es cierto que se puede perder el recuadro que contiene precisamente al punto final de la curva del carril. Sin embargo, este será recuperado de otra forma en los siguientes pasos.

Una cuestión a aclarar es el por qué no se pone también un número máximo de píxeles que han pasado el umbral dentro de un recuadro. Esta operación sería redundante, ya que los recuadros con demasiados píxeles con alta intensidad son descartados en el paso 1. En la figura 2.7 se muestran en azul los recuadros descartados.

Quizás el detalle de mayor importancia de este paso es como elegir el valor del umbral. En la bibliografía se denomina *umbral local o adaptativo* a la aplicación de un umbral distinto a distintas partes de una imagen, lo que puede proporcionar un mejor resultado que un umbral único para todos los píxeles de una imagen, ya que los valores de intensidad van variando en conjunto a lo largo de la imagen. Este método de aplicar el umbral es perfecto para este algoritmo, ya que se puede aplicar un umbral a cada recuadro por separado.

El valor del umbral se obtiene calculando el valor de intensidad medio del recuadro. Esto es directo ya que se ha calculado la suma de los valores de intensidad de los píxeles de un recuadro en el paso anterior, por tanto sólo es necesario dividir entre el número total de píxeles en el recuadro para obtener la media. Este valor medio es generalmente demasiado oscuro y no da un buen resultado ya que la mayor parte de los píxeles del recuadro son obscuros, por tanto debe ser multiplicado por un factor, en la práctica de alrededor de 1,5, para que el resultado obtenido sea el buscado.

Aplicando de esta forma el umbral se obtienen con bastante precisión los píxeles del perfil del carril (píxeles iluminados por la luz reflejada del perfil del carril iluminado por el láser), independientemente de que la intensidad de los píxeles del carril vayan cambiando a lo largo de la curva completa. Esto es algo que puede ocurrir en la imagen por los distintos ángulos con los que se refleja la luz hasta la cámara desde los puntos iluminados de la superficie del carril.

2.2.1.3. Paso 3. Recta de mejor ajuste

En el paso anterior se han obtenido unos píxeles por cada recuadro que potencialmente contienen al perfil del carril. Estos píxeles, si pertenecen al

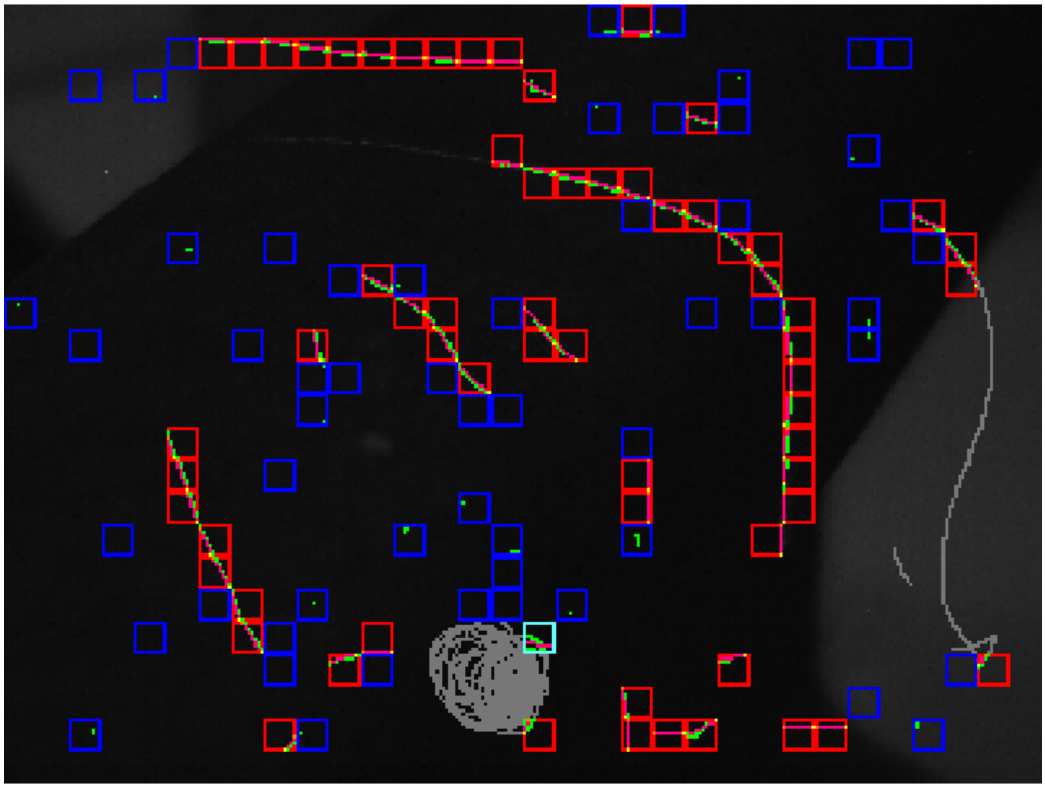

Figura 2.8: Paso 3. Recta de mejor ajuste (morado). Recuadros descartados (azul claro). Puntos inicial y final (amarillo y naranja, respectivamente)

carril, deben cumplir un nuevo requisito: formar una línea. Aunque el perfil sólo tiene una parte recta, los pequeños pedazos de la curva que quedan dentro de un recuadro son aproximadamente rectos.

En este paso se utilizan los píxeles de los recuadros guardados anteriormente para calcular la recta de mejor ajuste (por el método de mínimos cuadrados) de dichos píxeles dentro del recuadro. Junto a los parámetros de la recta también se calcula el error de regresión, que indica lo lejanos que quedan los píxeles de la recta calculada. Este error es un indicativo de si realmente los píxeles forman una recta o no. Si el error es mayor que un umbral probablemente se trate de una nube de píxeles inconexa o de una curva muy cerrada (lo suficiente para que parezca una curva incluso en el tamaño de un recuadro). Estos recuadros no pueden contener al carril por lo que son descartados.

Realmente el valor utilizado para evaluar la curvatura de la nube de píxeles es el error de regresión medio (error de regresión entre el número de píxeles tomados para calcularla) de forma que el número de píxeles no afecte

a la hora de descartar o no un recuadro, sólo la forma de los píxeles de este. El umbral utilizado para descartar recuadros en este paso es otro de los parámetros del algoritmo.

En la figura 2.8 se han pintado en morado en cada recuadro las rectas de mejor ajuste calculadas para los píxeles que fueron guardados en el Paso 2. También se han pintado los píxeles iniciales y finales de cada recta en cada recuadro. Uno de los recuadros guardados hasta este paso ha sido descartado por un error demasiado alto en la recta de regresión calculada en la imagen de ejemplo.

En realidad el objetivo principal de este paso no es descartar más recuadros, sino calcular la información que será necesaria para pasos posteriores del algoritmo.

Para los recuadros que siguen sin ser descartados se guarda la pendiente de la recta y los puntos finales e iniciales de ésta dentro de los límites del recuadro. Esta información será usada en el siguiente paso para encontrar los recuadros que contienen la perfil.

2.2.1.4. Paso 4. Seguimiento de formas

En este paso se van a intentar unir recuadros para intentar encontrar los que contienen al perfil, si es que el perfil es visible en la imagen. Este es el paso más interesante del algoritmo. Se trata además de una bifurcación en la ejecución del algoritmo. Los anteriores pasos se aplican siempre, los pasos posteriores a este dependen de que en este paso se encuentre un grupo de recuadros que contiene al perfil.

En las hipótesis de partida (sección 2.1.2), se encuentra la de continuidad de la curva del carril. Esto significa que la curva del carril no contiene de forma intermitente zonas demasiado oscuras como para no ser detectada la curva y no ser guardado el recuadro en los tres primeros pasos. En la figura 2.8 se puede observar como la cadena de recuadros que contienen al perfil es continua. Dicho de otra forma, todo recuadro de esta cadena, tiene su vecino de la cadena dentro de la vecindad 8 de recuadros. Este paso va a consistir en encontrar cadenas de recuadros que cumplen esta condición de continuidad y evaluar si se trata o no del carril.

Una forma ordenada de búsqueda de cadenas de recuadros puede consistir en buscar con una dirección determinada. Cómo los carriles son grabados desde el interior, el recuadro extremo (superior) de la cadena que contiene al carril se encuentra a la izquierda del resto de recuadros de la cadena si el carril grabado es el izquierdo (desde el punto de vista de las cámaras) y se encontrará a la derecha en el caso de las imágenes del carril derecho. Este recuadro también es el que se encuentra en la parte superior de la

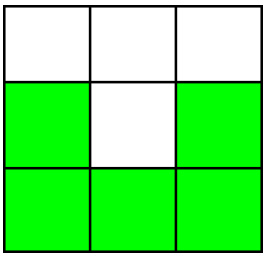

Figura 2.9: Recuadros vecinos (en verde) al último recuadro de la cadena (recuadro central) en los que en los que se buscará continuarla.

cadena. Es por esto que el algoritmo comienza buscando los recuadros con la posición más arriba a la izquierda (derecha en el caso del carril derecho), para comprobar si este tiene vecinos y comenzar a buscar cadenas.

Una vez encontrado el primer recuadro debe comprobarse si existe algún recuadro que lo continúa, un recuadro en su vecindad. No se busca en la vecindad completa, sino en los recuadros laterales o inferiores como se aprecia en la figura 2.9. Para que un recuadro vecino sea considerado como continuación de la cadena del recuadro actual, debe cumplir las siguientes condiciones:

1. El recuadro no puede haber sido descartado en pasos anteriores.

2. La diferencia entre la pendiente de la recta de mejor ajuste del recuadro actual y el recuadro vecino debe ser menor que cierto valor umbral máximo.

3. La distancia entre el pixel final (de su recta de mejor ajuste) del recuadro actual y el pixel inicial del recuadro vecino no puede superar un valor umbral máximo.

No debe olvidarse que aunque se están formando cadenas de rectángulos, estos deben contener la curva del perfil. Por tanto las rectas de mejor ajuste de dos recuadros vecinos que contienen una misma curva continua tendrán una pendiente parecida. Podría ocurrir que uno de los vecinos conteniese el extremo de otra curva, con lo que la pendiente será distinta y por tanto no puede continuar la cadena de recuadros actual.

La condición de pendiente no es suficiente, pueden darse casos en los que existan varios vecinos con la pendiente de su recta de mejor ajuste similar al del recuadro actual. Además esta condición de pendiente no puede ser demasiado ajustada ya que la curva del carril contiene zonas de pequeño radio en los que el cambio de pendiente de las rectas de mejor ajuste de un recuadro a otro no es menor. Es necesario por tanto que se cumpla también la condición de cercanía de pixel final e inicial de las rectas.

En este punto del algoritmo pueden darse tres casos:

1. Uno de los recuadros vecinos cumple las condiciones. Este vecino se añade a la cadena, se borra de la lista de búsqueda de recuadros guardados (para que no pueda ser elegido de nuevo como recuadro continuador de cadena) y se convierte en el nuevo recuadro actual. Se vuelven a buscar recuadros vecinos a este.

2. Existe más de un vecino que cumple con las condiciones. Este caso puede darse incluso con un buen ajuste de los parámetros del algoritmo. Si la curva del carril no es lo suficientemente delgada y contiene partes totalmente verticales (en la zona de la parte recta) la misma curva podría ser encuadrada por dos recuadros, cuando parte recta de la curva queda justamente en la interfaz de ambos recuadros. En cualquier caso, sólo uno de los recuadros debe ser elegido. De todos los recuadros aspirantes que han cumplido las condiciones, será elegido el cual cuya recta de mejor ajuste tenga una menor diferencia de pendiente con la del recuadro actual. Este recuadro vecino es añadido a la cadena, borrado de la lista de búsqueda de recuadros que fueron guardados en los pasos 1-3 y elegido como el nuevo recuadro actual desde el que seguirá buscándose un nuevo vecino que siga la cadena.

3. No existen recuadros vecinos (al recuadro actual) guardados en los pasos anteriores o ninguno de los vecinos cumple las condiciones para continuar en la cadena de recuadros. En este caso la cadena actual ha finalizado y debe comprobarse si es la cadena que contiene al carril del perfil. Aunque no es común, podría haberse encontrado una cadena que contiene una curva que no es el perfil, o simplemente tratarse de un recuadro aislado sin vecinos con los que formas una cadena.

   La evaluación de la cadena se basa en una hipótesis de partida muy concreta: no existirá ninguna otra curva en la imagen además de la del carril que tenga el tamaño y la forma que la cadena que contiene al perfil del carril. Por tanto se van a aplicar condiciones de longitud y tamaño para descartar la cadena encontrada, o elegirla como la cadena que contiene al perfil del carril y finalizar la búsqueda, ejecutándose directamente el siguiente paso. Es por esto que el ancho, y el alto de la cadena (medidos en número de recuadros) deben encontrarse dentro de un intervalo de valores determinados por cuatro parámetros. El algoritmo también comprobará que la longitud o número de recuadros de la cadena está entre un valor mínimo y un valor máximo. Este paso es es el más eficiente a la hora de descartar recuadros que no contienen

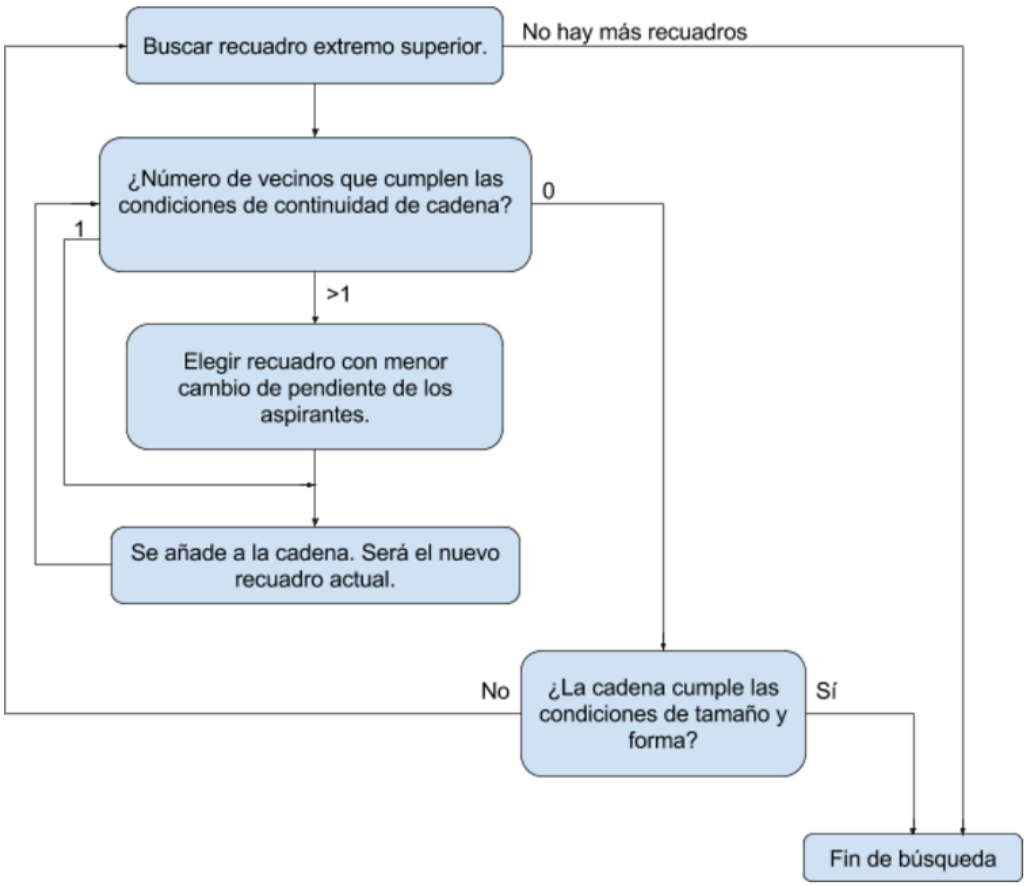

Figura 2.10: Diagrama de bucle de búsqueda de la cadena de recuadros que contiene la curva del perfil en la imagen.

al carril, ya que se está usando información sobre la forma y tamaño esperado del carril en la imagen, información que es conocida. Sin embargo, la aplicación de condiciones en pasos anteriores ayuda a reducir el tiempo de computación del algoritmo, permitiendo que este paso se ejecute muy rápido y que en el paso anterior no se tengan que calcular demasiadas rectas de regresión. No se debe perder de vista que el algoritmo debe ser ejecutado 500 (250 por cada carril) veces por segundo en un PC con una capacidad de cómputo media.

Este bucle de búsqueda de cadenas se muestra en el diagrama de flujo de la figura 2.10. En la figura 2.8 se muestra como el algoritmo ha encontrado la cadena del perfil.

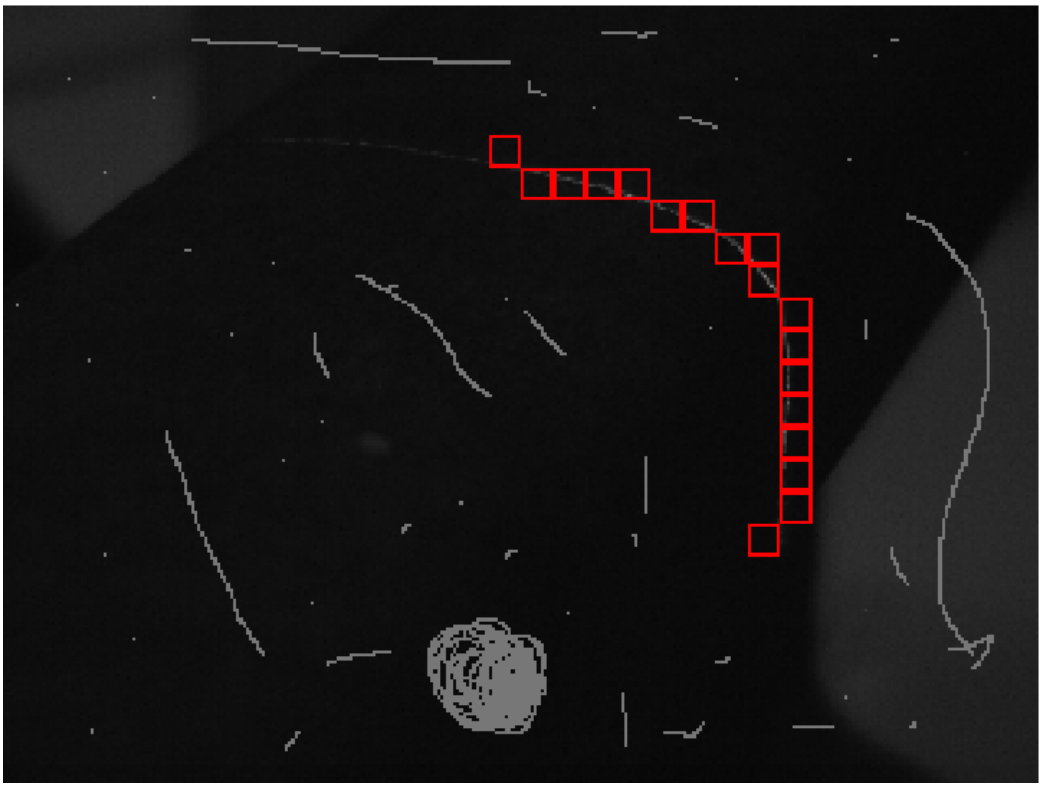

Figura 2.11: Paso 4. Cadena de recuadros que contiene la curva del perfil en la imagen encontrada.

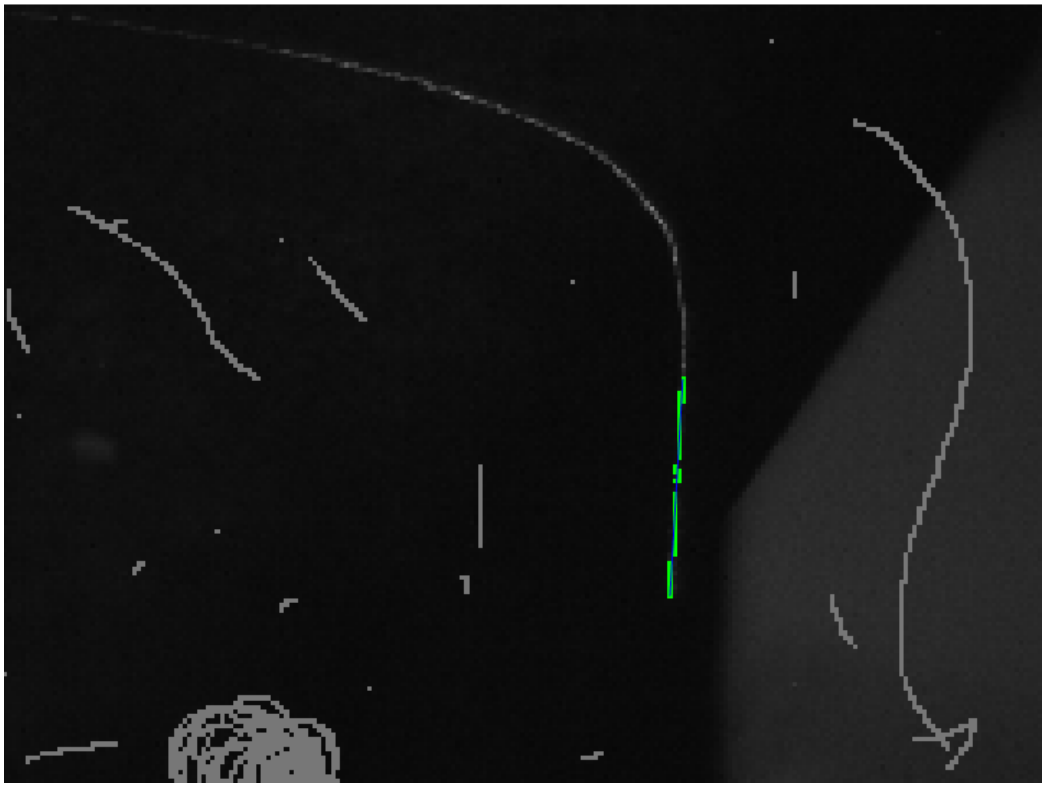

Figura 2.12: Paso 5. Cálculo de la pendiente de la parte recta del perfil del carril en la imagen.

2.2.1.5. Paso 5. Obtención de la pendiente de la recta

Si el paso anterior encontró la cadena de recuadros que contiene el perfil del carril en la imagen, es posible calcular los datos de salida del algoritmo. En este paso este paso se va a calcular la pendiente de la parte recta. El cálculo de la pendiente de la recta va a permitir además calcular la posición de su extremo final, como se verá en el siguiente paso.

El valor aproximado de esta pendiente ya se conoce, y es el de la pendiente de las rectas de últimos recuadros de la cadena que contienen al perfil. Estos últimos recuadros contienen la parte recta. Para que la medida sea lo más precisa posible, en vez de utilizar las pendientes de estas rectas de los últimos recuadros, se recuperan los píxeles que pasaron la umbralización dentro de ellos y se vuelve a calcular la recta de mejor ajuste. El resultado es bastante preciso, al menos todo lo que permite la resolución de la cámara. El número de recuadros de la parte recta elegido para realizar el cálculo es un parámetro del algoritmo.

En la figura 2.12 se muestran los píxeles utilizados para el cálculo de la

recta de mejor ajuste (en verde) y el resultado del cálculo (en azul). En este caso han sido utilizados los píxeles de los últimos recuadros de la cadena y sus correspondientes vecinos laterales.

Ya se ha comentado anteriormente la posibilidad de que pequeñas partes de la curva del carril no entren dentro de los recuadros de la cadena del carril. Esto ocurre si la parte recta es totalmente vertical, dónde los píxeles de la curva se dividen entre dos recuadros, el izquierdo y el derecho. También puede ocurrir que la curva cruze dos recuadros vecinos pero una pequeña parte de ellos no entren en la cadena. Esto ocurre porque la cadena de recuadros que contiene al perfil tiene espesor 1.

En las primeras versiones del algoritmo esto creaba problemas de precisión porque la pérdida de un grupo de píxeles cambia ligeramente la pendiente, empeorando la precisión. La solución ha consistido en intentar agregar todos los píxeles de la parte recta aunque no pertenezcan a los recuadros de la cadena encontrada en el paso anterior. Como los píxeles que pasan el umbral son guardados en memoria a pesar de ser descartado el recuadro que lo contiene en pasos posteriores, es posible usar los píxeles de los recuadros laterales a los de la cadena en su parte recta para realizar el cálculo de la recta de mejor ajuste. En cierta forma, es como si en la parte recta final realmente la cadena pudiera tener un espesor de 3 recuadros. La inclusión de todos los píxeles sin excepción en el cálculo de la recta de mejor ajuste ha permitido que la precisión en el cálculo de la pendiente sea elevado en todos los casos.

2.2.1.6. Paso 6. Obtención del punto final de la parte recta

Una vez obtenida la pendiente de la recta sólo resta obtener las coordenadas x e y del punto fin la de la recta. Aunque parezca que esta tarea es tan simple como escoger el pixel más bajo del último recuadro de la cadena del perfil que pasará el umbral en los primeros pasos, esta forma de localizar el pixel no funcionaría frecuentemente.

En el paso 2 del algoritmo, tras umbralizar los recuadros, los que tuvieran menos de un determinado número de píxeles que pasaran el umbral eran descartados. Por esto, si la parte final de la parte recta de la curva del carril quedaba en un recuadro con pocos píxeles, este recuadro podía ser descartado, a pesar de contener al carril. Esto puede ocurrir con mucha frecuencia, por tanto el método diseñado permite seguir los píxeles de la recta incluso fuera del último recuadro de la cadena.

Como la pendiente de la cadena es conocida, y se conocen los píxeles de ella, es posible reconstruirla, incluso aunque su parte final no esté dentro del recuadro final de la cadena de recuadros encontrada. En esta idea, y en la idea

de que cuando la recta se acaba los píxeles tendrán un nivel de intensidad bajo, se basa este paso. Se va a ir recorriendo la recta hasta encontrar su pixel final, se encuentre o no dentro del último recuadro de la cadena.

Para comenzar la búsqueda del pixel, se elige el pixelo más alto de los píxeles que pasaron el umbral del último recuadro de la cadena. Una vez elegido el pixel, se comprueba si el mismo y su vecindad 8 (píxeles vecinos) se encuentran por debajo de un nivel de intensad máximo. Este nuevo umbral es un parámetro del algoritmo que debe ser ajustado para su correcto funcionamiento. Si todos los píxeles tienen un valor de intensidad menor al del umbral, la recta ha dejado de verse en la imagen y por tanto ha sido encontrado el final de esta.

Si la condición no se cumple se debe elegir el siguiente pixel de la recta justo un pixel inferior al actual. Por tanto el nuevo elegido para aplicar la condición tendrá su coordenada y una unidad mayor que la actual. Su coordenada x es calculada con la ecuación general de la recta con los parámetros de pendiente e intercepción obtenidos en el paso anterior de la parte recta. El valor devuelto por el cálculo es truncado y elegido como la nueva coordenada x. Con este nuevo pixel elegido, se aplica de nuevo la condición. De esta forma se va recorriendo la recta hasta que esta desaparece en la imagen. Este método es independiente de los recuadros, excepto por la selección el pixel de partida. Por tanto aunque el pixel final de la recta queda fuera del último recuadro, este será encontrado.

La condición de fin nunca se cumple exactamente en el pixel final de la recta, sino dos píxeles más abajo. Esto ocurre porque cuando el pixel final es elegido, el mismo no cumple la condición debido a que su nivel de intensidad supera el umbral. Una vez elegido el pixel inferior, el pixel final seguirá en la vecindad y por tanto seguirá sin cumplirse la condición. No es hasta que el algoritmo se encuentra dos píxeles más abajo del pixel final cuando este no se encuentra en la vecindad y la condición se cumple. Es por esto la coordenada y del pixel final es dos veces inferior a la del pixel dónde su cumplió la condición (el eje y crece hacia abajo). La coordenada x es calculada aplicando de nuevo la ecuación general de la recta con la coordenada y definitiva del pixel final. En la figura 2.13 se observa como el algoritmo ha encontrado el punto final con precisión de un pixel.

En el capítulo 4 se describe una forma de mejorar este paso permitiendo una precisión de subpixel a la hora de obtener la posición del punto final de la recta del perfild el carril.

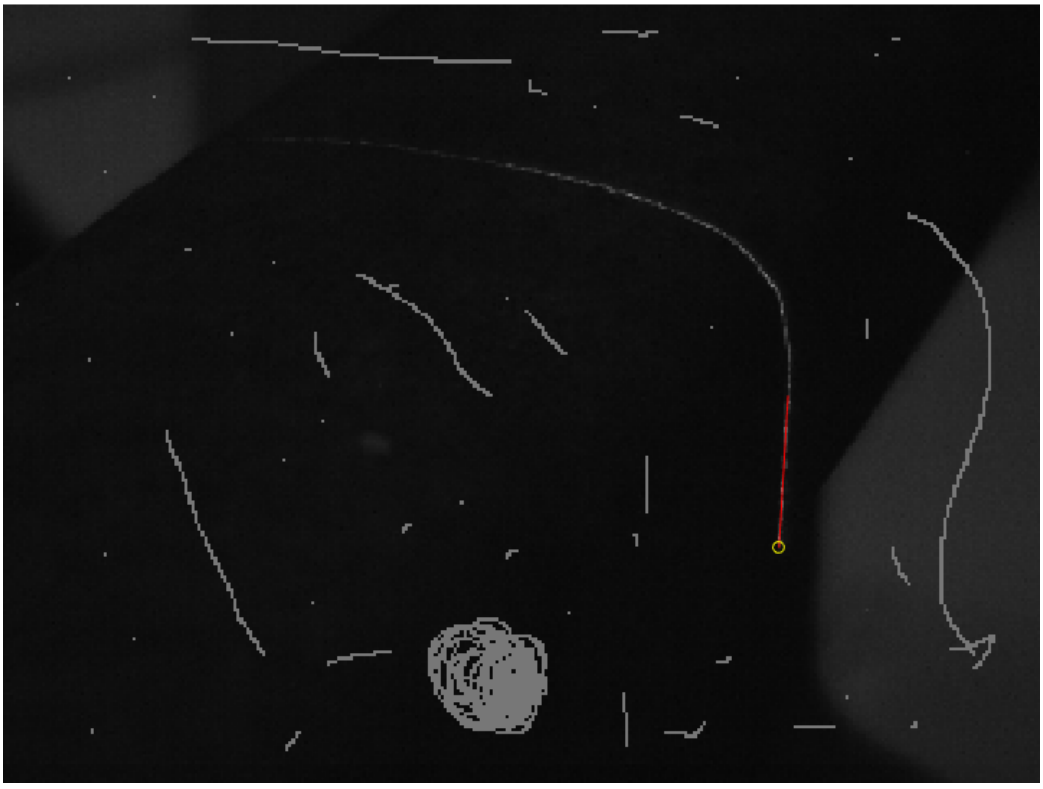

Figura 2.13: Paso 6. Obtención del punto final de la parte recta.

Figura 2.14: Carril de prueba.

2.2.2. Implementación del algoritmo

El algoritmo fue diseñado e implementado para comprobar la validez de las ideas propuestas en Matlab. El funcionamiento del algoritmo fue puesto a prueba con vídeos realizados en las oficinas de Virtualmech con un trozo de carril real como el que se observa en la figura 2.14.

Las pruebas de funcionamiento en Matlab eran satisfactorias. Sin embargo, era necesario realizar pruebas de rendimiento. Para ello el algoritmo fue escrito en C++, que permite alcanzar velocidades muy superiores a las de un lenguaje interpretado como Matlab. En caso de que las pruebas no cumpliesen con la especificación de 250 ejecuciones por segundo, el algoritmo debería ser implementado en una FPGA para alcanzar un mayor paralelismo y alcanzar la velocidad de ejecución de la especificación.

El código de la implementación en matlab no va a ser descrito por tratarse de una versión implementada para el diseño y pruebas de funcionamiento del algoritmo. Las velocidades de ejecución alcanzadas en matlab son muy bajas y por tanto no tiene sentido ser explicadas en detalle. Si será explicada la implementación en C++. En cualquier caso, junto a este documento pueden encontrarse los ficheros que componen el código de Matlab totalmente comentados.

En esta sección se describe como ha sido implementado el algoritmo en C++. Se entiende que las secciones anteriores en los que se explica el funcionamiento del algoritmo han sido leídos y entendidos.

2.2.2.1. Implementación en C++

Los códigos C++ que implementan el algoritmo se encuentran adjuntos a este documento. Están totalmente comentados, el código puede ser seguido simplemente leyéndolo. Concretamente los archivos que forman el código son los siguientes:

1. *algoritmoBuscaCarril.cpp* y *algoritmoBuscaCarril.h*: Contienen el código de la función que implementa el algoritmo. Contiene dos versiones de la función: una analiza la imagen y calcula los resultados y la otra adicionalmente pinta los resultados en varias imágenes. La función que pinta el resultado puede no cumplir las especificaciones de velocidad de cómputo.

2. *funcionesAlgoritmo.cpp* y *funcionesAlgoritmo.h*: Contienen el código de las funciones que realizan el cálculo de la recta de mejor ajuste, así como sus puntos iniciales iniciales y finales al enmarcarla en un recuadro. También se incluyen funciones para pintar las rectas obtenidas.

3. *funcionesListas.cpp* y *funcionesListas.h*: Contiene el código para encontrar los valores máximos y mínimos en vectores de enteros o de variables numéricas de punto flotante.

En las siguientes secciones se va a detallar la implementación en código C++ de cada uno de los pasos anteriormente descritos, de forma que sea posible comprender el código de la forma más sencilla posible, leyendo este capítulo y los comentarios del código.

La declaración de la función que implementa el algoritmo es la siguiente (versión sin dibujado):

```
int algoritmoBuscaCarril(Mat ima, OutAlgoritm *res, int dir,
    AlgoritmParam param);
```

La función recibe en total 4 argumentos:

1. Imagen del perfil del carril. La imagen debe estar representada en valores de gris entre 0 y 255. En este caso el tipo de dato es *Mat*, usado en OpenCV.

2. Referencia a una estructura tipo *OutAlgoritm* dónde se guardará el resultado del algoritmo. A continuación se muestra la definición de la estructura:

```
typedef struct OutAlgoritmStruct{
        float m, x, y;
} OutAlgoritm;
```

3. Dirección del carril. Indica si la imagen tomada es del carril izquierdo o el derecho de la vía. El algoritmo buscará en una dirección u otra en función de este parámetro.

4. Copia de una estructura del tipo *AlgoritmParam*. Esta estructura contiene el valor de todos los parámetros del algoritmo.

   Los parámetros del algoritmo son valores constantes elegidos de forma manual que son aplicados en distintas condiciones del algoritmo. Las pruebas realizadas muestran la necesidad de elegir acertadamente los parámetros para que el algoritmo pueda funcionar correctamente.

   En el capítulo 4 se introduce una mejora del algoritmo en la que los parámetros son elegidos de forma automática analizando las imágenes que llegan del sistema de adquisición. Además de esta forma los parámetros pueden dejar de ser constantes durante la ejecución del algoritmo, para poder ir siendo adaptados a las nuevas condiciones con la que es tomada la imagen.

   Estos parámetros deben ser sintonizados para un correcto funcionamiento del algoritmo como se muestra en la sección 2.2.3. A continuación se muestra la definición de la estructura:

```
typedef struct AlgoritmParamStruct{
// Generales

// Pasos 1-3
int SQminContrast;                  // Contraste mínimo
    que debe haber en un cuadrado
int SQmaxWhite;                     // Máximo blanco que
     puede haber en un cuadrado
float meanTHmult;                   // Al hacer el
    umbral en cada recuadro se usa la media
    multiplicada por esta variable
int minSQpoint;                     // Número mínimo de
    píxeles en un recuadro que deben haber pasado el
     umbral
float maxMeanErrorSQLR; // Máximo error medio de
    error entre los puntos de un recuadro y la recta
     de mejor ajuste calculada
```

```
    // Paso 4
    float maxSlopeJump;              // Máximo salto de
        pendiente entre recuadros contiguos del perfil
        en grados
    float maxDextremes;              // Máxima distancia
        entre el pixel inicial y final de recuadros
        contiguos
    float maxNBrectD;                // Máxima distancia
        para considerar un vecino directamente continuo
        sin comprobar el resto
    float maxNBrectM;                // Máximo ángulo
        para considerar un vecino directamente continuo
        sin comprobar el resto
    float marginSlopeJump;  // Salto máximo para elegir
        siguiente vecino por distancia
    int minNSQprofile;               // Mínimo número de
        recuadros que contienen al perfil
    int maxNSQprofile;               // Máximo número de
        recuadros que contienen al perfil
    int minNSQwidth;                 // Mínimo número de
        recuadros que debe medir a lo ancho el conjunto
        del perfil
    int minNSQheight;                // Mínimo número de
        recuadros que debe medir a lo alto el conjunto
        del perfil

    // Paso 6
    int THendRect;                   // Por debajo de
        este valor se considera que los puntos no
        pertenencen a la recta del perfil
    // Calculado como el valor del final de la bajada
        del primer
    // y mayor pico del histograma de la imagen (ya sea
        blanco y
    // negro o tonos rojos)

    } AlgoritmParam;
```

La función devuelve un valor entero que indica si se ha encontrado el perfil en la imagen, o un código de error en caso negativo. Todas las declaraciones de estructuras mostradas se encuentran en el archivo *algoritmoBuscaCarril.h*. En este mismo archivo se encuentran definiciones utilizadas por el algoritmo como el tamaño de la imagen, el tamaño del recuadro en píxeles, etc.

A continuación se va a profundizar en la implementación en C++ de los distintos pasos del algoritmo. Algunos de estos pasos requieren el uso de gran cantidad de memoria durante su ejecución. Por ejemplo, es necesario

guardar arrays de una gran cantidad de puntos. Aunque inicialmente parece que estas variables deben ser locales a la función del algoritmo ya que no van a ser utilizadas fuera de esta, debe recordarse que al llamar una función (que no es la función principal del programa) las variables creadas en esta función se añadirán en la pila. Si el número de variables creadas supera el tamaño de la pila se producirá un error de ejecución. En este caso el número de variables locales es demasiado grande y los bloques de memoria mayores se han declarado de forma global. Es por esto que en el archivo *algoritmoBuscaCarril.h* aparecen grandes arrays y matrices tipo *extern*, ya que estas han sido declaradas en el archivo principal del programa para que sean globales. De no hacerse de esta forma se obtenía el error de ejecución descrito.

2.2.2.2. Pasos 1-3

Los tres primeros pasos están implementados en un mismo bloque. En el primer paso se recorre toda la imagen para comprobar que recuadros cumplen las condiciones de dicho paso. Sólo si el recuadro cumple las condiciones se realizan los cálculos y comprobaciones del paso 2. Lo mismo se hace con el paso 3. De esta forma se evita tener que recorrer todos los recuadros de la imagen cada vez que se aplica un nuevo paso.

Como se ha descrito anteriormente, primero se aplica un filtro de nivel de intensidad en el paso 1. Sólo a los pixeles de los recuadros que cumplen las condiciones se le aplica un umbral. Los pixeles que pasan el umbral son guardados en una lista temporal de puntos. Esta lista está compuesta por dos arrays dónde se guardan las coordenadas X e Y de los puntos.

Los recuadros que tienen un mínimo de puntos pasan al paso 3. En este paso se calcula la recta de mejor ajuste con los puntos del recuadro que pasaron el umbral. Sólo si el error de regresión es lo suficientemente bajo se guardan los datos de la recta que serán usados en pasos posteriores. También son guardados los puntos que pasaron el umbral del recuadro en una lista de puntos y el índice del primer pixel de dicho recuadro en la lista. Esto permite usar la información del umbral sin tener que volver a calcular el umbral de los recuadros en pasos posteriores.

2.2.2.3. Paso 4. Seguimiento de formas

En el paso 4 se busca la cadena de recuadros que contiene la curva del perfil. Este paso se implementa con dos bucles *while*. Cada iteración del bucle externo comienza una cadena que puede contener al perfil, para ello busca el primer recuadro por el que se comenzará a intentar formar la cadena. Cada

iteración del bucle interno pretende buscar un nuevo recuadro vecino que cumpla las condiciones para continuar la cadena actual.

La lista de recuadros en la que se busca es la formada por los recuadros que cumplieron las condiciones en los anteriores pasos. Cuando un recuadro es utilizado por este paso, su identificador de recuadro es igualado a 0 indicando que no puede volver a ser utilizado. Es por esto que un recuadro no será utilizado más que una vez en el intento de formar cadenas.

Los recuadros que forman la cadena actual se añaden a una lista, que será utilizada por pasos posteriores en el caso en el que se encuentre la cadena del perfil.

2.2.2.4. Pasos 5 - 6. Obtención de la pendiente de la recta y el punto extremo

Los dos últimos pasos del algoritmo sólo son ejecutados si en el paso anterior fue encontrada la cadena que contiene al perfil. El paso anterior indica con la variable $profileFinded$ si el perfil ha sido encontrado con éxito.

El paso 5 pretende calcular la pendiente de la parte recta. Este paso es implementado con un bucle for que recorre de forma descendente (comenzando por el recuadro final) la lista de los recuadros que forman la cadena. Los píxeles interiores a los recuadros que pasaron el umbral en los anteriores pasos son añadidos a una nueva lista que será utilizada para calcular la pendiente de la parte recta.

Como ya fue descrito en la sección 2.2.1.5 también son añadidos los píxeles de los recuadros laterales, para ello se comprueba si los recuadros laterales pertenecen a la lista de recuadros que cumplieron las condiciones de los pasos 2 y 3 (recuadros que contienen una nube píxeles de mayor intensidad que el fondo en forma de recta). Sólo para esos recuadros fueron guardados los píxeles que pasaron el umbral.

El paso 6 es implementado mediante un bucle *while* que va recorriendo la parte recta hasta encontrar el extremo. Para ello se va incrementando la coordenada y y se calcula la coordenada x mediante la ecuación de la recta (es necesario truncar el resultado para seleccionar un pixel). El bucle acaba si el pixel examinado cumple la condición de fin o si las coordenadas del nuevo pixel lo sitúan fuera de la imagen.

Antes de finalizar el algoritmo, este guarda los resultados en la estructura pasada como referencia.

2.2.2.5. Pruebas de rendimiento

Como se ha comentado anteriormente, en los inicios del proyecto parecía ser inevitable el uso de una FPGA que permitiera la ejecución del algoritmo a la frecuencia requerida. El paso intermedio de implementar el algoritmo en C++ tiene su interés como primera medida de la eficiencia del algoritmo.

Una vez implementado el algoritmo en C++, se probaron varias imágenes para verificar a priori su correcto funcionamiento. Una vez hecho, se preparó un programa que ejecuta el algoritmo a la máxima velocidad posible pasándole un grupo de imágenes cíclicamente. Las pruebas fueron realizadas en un portátil Lenovo Z50-70 que cuenta con las siguientes características:

1. Procesador: Core i7-4510U, 2 Núcleos a 2GHz.
2. RAM: 16GB.

A continuación se muestran los resultados obtenidos para imágenes de distinta resolución:

1. Resolución: 320x240 píxeles, frecuencia: 2808 Hz.
2. Resolución: 640x480 píxeles, frecuencia: 525 Hz.
3. Resolución: 1280x960 píxeles, frecuencia: 159 Hz.

En los ensayos a realizar las cámaras tienen una resolución de $640x480$ (sección 1.3). Sin embargo, se trata de cámaras a color de las que sólo se reutiliza el canal rojo sin tratamiento por lo que queda una imagen monócrona de 320x240 píxeles. Tanto en ese caso como en el 640x480 la frecuencia obtenida supera la frecuencia requerida (250 Hz). Posteriormente se realizaron pruebas con éxito con el equipo de auscultación descrito en el capítulo 1 y e imágenes reales de las cámaras del proyecto.

Tras la obtención de estos resultados fue decidido no implementar en FPGA el algoritmo, evitando de esta forma añadir mayor complejidad al proyecto. En el capítulo 4 se proponen cambios en el algoritmo que mejoran su rendimiento manteniendo la funcionalidad intacta, por lo que sería posible aumentar la frecuencia de trabajo del algoritmo en el caso del uso de cámaras de mayor precisión.

2.2.3. Sintonización

El algoritmo utiliza una gran cantidad de parámetros durante su funcionamiento para obtener un buen resultado. Darle valor a estos parámetros buscando el buen funcionamiento del algoritmo es llamado sintonización.

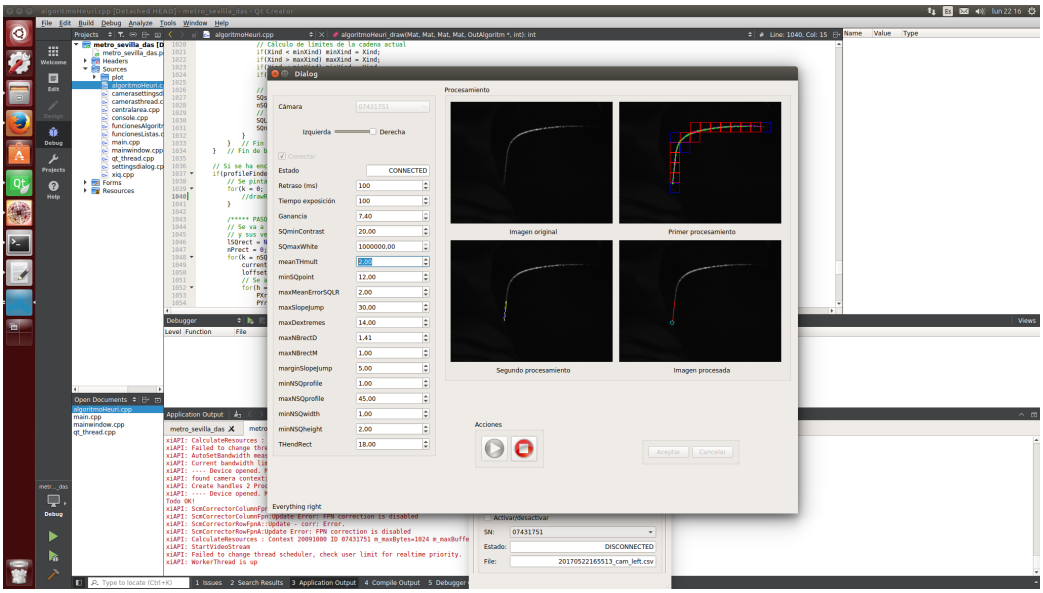

Figura 2.15: Sintonizador de parámetros.

Como se describe en el capítulo 4 es posible el uso de un algoritmo inteligente que sea capaz de seleccionar automáticamente los parámetros del algoritmo, e incluso ir cambiándolos durante la ejecución del programa para adaptarse a ciertas condiciones que pueden cambiar. Sin embargo, este algoritmo inteligente es un pequeño proyecto en sí mismo. Esto unido a que los ensayos dónde se va a probar por primera vez el algoritmo se producen por la noche (no hay cambios de condiciones de iluminación) llevó a la idea de realizar estas primeras pruebas con los parámetros elegidos a mano.

En las figuras 2.15 2.16 se puede observar la interfaz del sintonizador desarrollada en Virtualmech supervisada por el autor del presente proyecto.

La interfaz está integrada en el software que controla el sistema de auscultación. Durante el propio montaje y preparación de los ensayos se realizaba la sintonización, una vez fijadas las cámaras y el láser de proyección lineal.

2.3. Conclusiones del diseño e implementación del algoritmo

Durante el desarrollo e implementación del algoritmo se visitaron las instalaciones del Metro de Sevilla, lugar dónde se han realizado los ensayos del sistema completo. Durante las visitas, entre otras tareas, se pudieron tomar vídeos con las cámaras similares a los que se tomarían en los ensayos. Estos

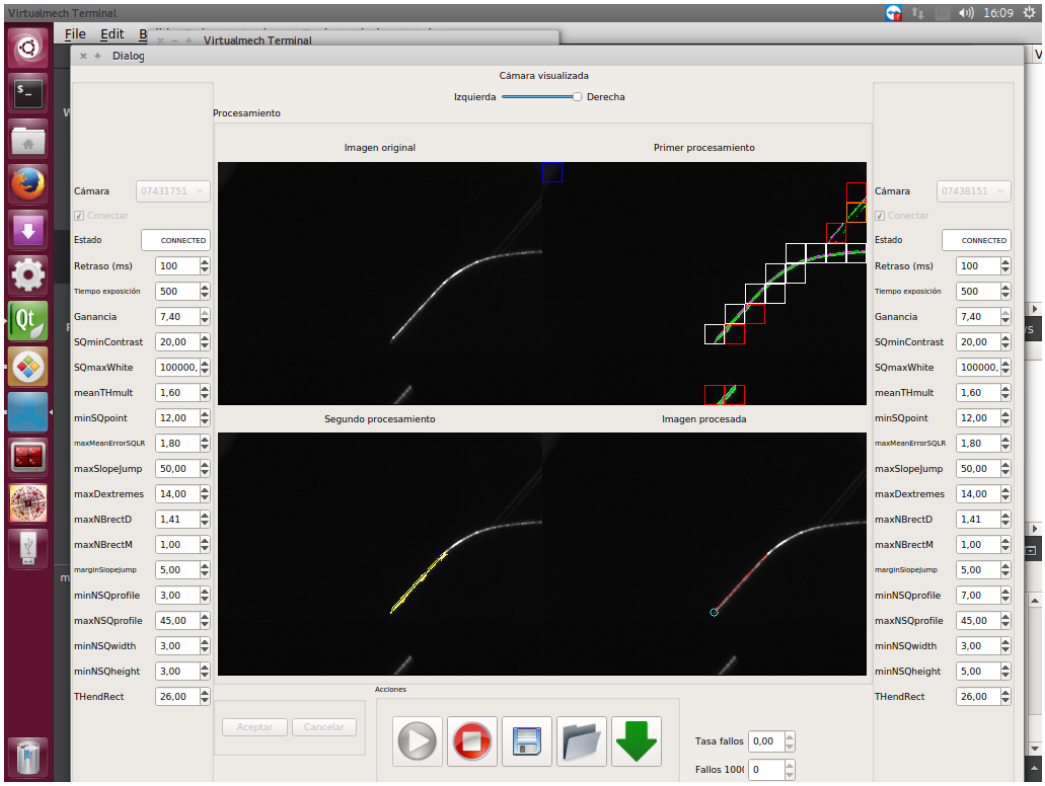

Figura 2.16: Sintonizador de parámetros.

han sido los vídeos utilizados durante las pruebas del algoritmo en la oficina. Esto ha permitido poder sacar conclusiones a priori realistas sobre el funcionamiento del algoritmo.

Las conclusiones del desarrollo e implementación del algoritmo son las siguientes:

1. El algoritmo es capaz de encontrar tanto el perfil del carril en la imagen, como la posición del punto final de la parte recta y la pendiente de ésta.

2. Si el perfil del carril no se encuentra en la imagen el algoritmo es capaz de detectarlo e indicarlo. Aunque no está pensado para ello, el algoritmo es capaz de encontrar el perfil del carril aunque una pequeña parte de su extremo se encuentre fuera de la imagen, siempre que el extremo cortado no sea el que contiene la parte recta.

3. El correcto funcionamiento del algoritmo es sensible a la sintonización de los parámetros.

4. El algoritmo no se ve afectado por el ruido de la imagen a la hora de encontrar el perfil del carril.

5. El algoritmo es capaz de rechazar reflejos de otros objetos que pueden aparecer en la imagen, siempre que se cumplan las hipótesis de la sección 2.1.2.

6. El algoritmo es estable.

7. El algoritmo no requiere ser implementado en FPGA. Es posible implementarlo en un PC con un procesador de arquitectura común para las especificaciones del proyecto SegFer. Si fuese necesario un mayor rendimiento es posible la implementación del algoritmo usando paralelismo. Si no fuera suficiente pueden utilizarse sistemas DSP antes de dar el salto a las FPGA, que complicarían la implementación.

8. El algoritmo cumple con la especificación de la velocidad de cómputo requerida por el proyecto SegFer.

9. El algoritmo tiene un consumo de memoria no despreciable.

10. El algoritmo utiliza técnicas comunes en visión como división de la imagen, umbral adaptativo y cálculo del contraste. También utiliza conceptos menos frecuentes como el cálculo de la pendiente de una recta.

11. El algoritmo ha sido desarrollado, implementado y testeado a tiempo para el ensayo programado. Sin embargo, no se han implementado las mejoras propuestas en el capítulo 4, las que puedan mejorar el algoritmo en varios aspectos.

En el capítulo 3 se sacarán conclusiones del funcionamiento del algoritmo provenientes del análisis de los resultados del algoritmo en los ensayos reales.

Capítulo 3

Resultado de los ensayos

Todo el desarrollo e implementación del algoritmo culmina en los ensayos. Es en ellos dónde realmente se demuestra si el algoritmo es capaz de funcionar. En este capítulo se van a presentar gráficas de los resultados de los ensayos y las conclusiones que se obtienen de estas.

3.1. Realización de ensayos

Los ensayos han sido realizados por un equipo de Virtualmech (entre los que se encuentra el autor de este proyecto) en las instalaciones del Metro de Sevilla. Aunque el montaje de los dispositivos se llevaba a cabo durante el día, los ensayos con el ferrocarril en movimiento fueron realizados por la noche. Esto ha sido muy conveniente para el algoritmo de visión, ya que no se producen cambios de iluminación y sólo es iluminado el perfil del carril mediante el láser de proyección lineal.

La preparación del ensayo por parte del algoritmo de visión es relativamente sencilla. Una vez fijadas las cámaras y el láser se procede a elegir los valores de los parámetros del algoritmo mediante la herramienta de sintonización del software de auscultación.

Los ensayos consistieron en la grabación de datos de todo el sistema de adquisición en varios viajes en un pequeño tramo de vía de referencia (previamente auscultado mediante otro sistema). Al tratarse del primer ensayo, surgieron problemas, todos afectando gravemente a la visión:

1. Para mejorar el contacto rueda-carril, los carriles son engrasados por el propio ferrocarril. La grasa provoca que la suciedad pueda quedarse pegada al carril con mayor facilidad, impidiendo la visión de parte de este, generalmente la parte recta de la cabeza del carril, que es

precisamente la que se busca como referencia para posicionar el carril en la imagen.

2. Un ferrocarril no es un sólido rígido, sino que está formado por 3 partes principales unidas por suspensiones que pueden sufrir movimientos relativos entre ellas. Este movimiento ha provocado que ciertas piezas se coloquen entre las cámaras y el carril en ciertos instantes de los ensayos, aunque no lo hicieran con el ferrocarril en reposo.

3. Las vibraciones hicieron ceder la sujeción del enfoque de las cámaras, provocando que fueran desenfocadas progresivamente a partir de cierto ensayo.

Todos estos problemas provocan que el carril no aparezca en la imagen, incluso en los ensayos en los que los carriles están correctamente enfocados. A pesar de ello pueden intentar sacarse algunas conclusiones a partir de los datos obtenidos debido a que estos problemas sólo aparecen de forma intermitente.

3.2. Análisis de resultados

A continuación se muestra el análisis de los datos de uno de los viajes de los ensayos. Cuando el algoritmo no encuentra el perfil del carril en la imagen, este devuelve el valor -1 en los campos de salida. De esta forma el primer paso al analizar los datos del algoritmo es encontrar estos datos y colocar en su lugar el último valor válido. Esto funciona bien en tramos cortos dónde se ha perdido el carril por la grasa o los contracarriles. Sin embargo, en las zonas dónde por cualquiera de estas causas el perfil se pierde en la imagen de forma sostenida, aparece una recta de valor constante, habiéndose tomado el último valor válido durante un largo intervalo de tiempo.

Durante el viaje analizado, el algoritmo fue ejecutado 240000 veces. De todas ellas la cámara izquierda perdió 90000 veces al carril, mientras que el carril derecho 20000. En las figuras 3.1 y 3.2 se muestran los datos de posición generados por el algoritmo de las cámaras izquierda y derecha respectivamente. Se muestran la versiones filtradas mediante un filtro paso baja que elimina ruido del algoritmo o del sistema de adquisición. En las figuras 3.3 y 3.4 se muestran los datos de orientación generados.

La salida del algoritmo da una idea de las magnitudes del movimiento del perfil del carril en las imágenes durante el viaje del ensayo (recordar que las imágenes tienen una resolución de 320x240 píxeles). Sin embargo, no

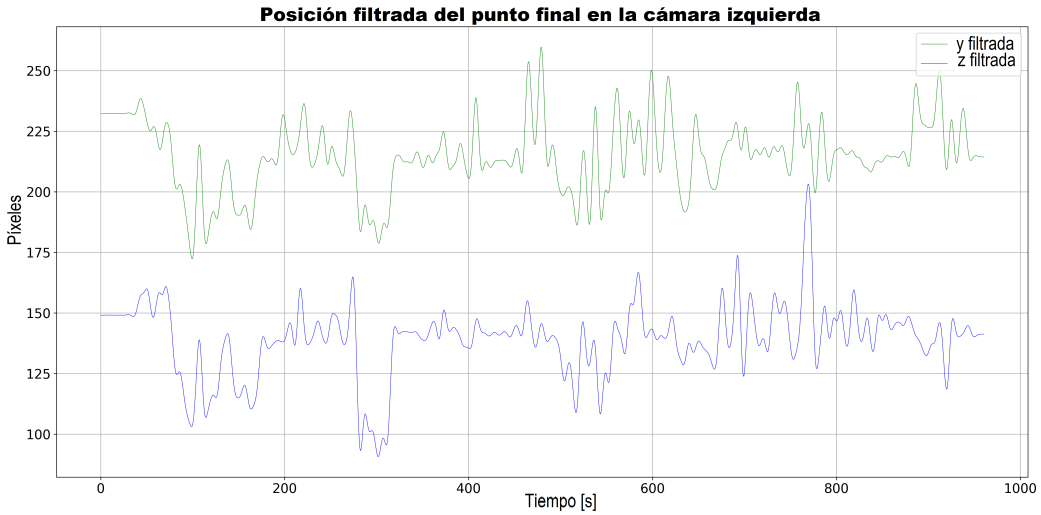

Figura 3.1: Salida (posición) del algoritmo, cámara izquierda.

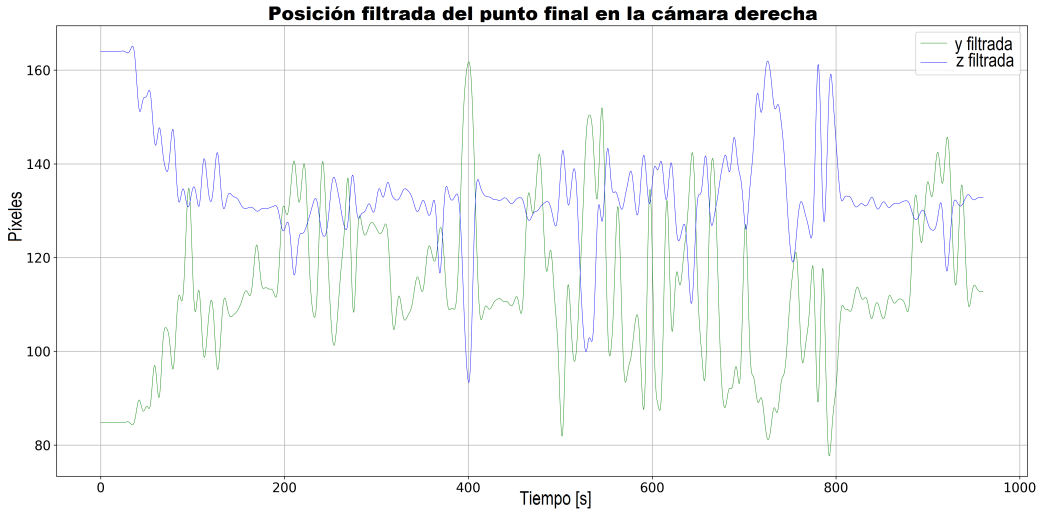

Figura 3.2: Salida (posición) del algoritmo, cámara derecha.

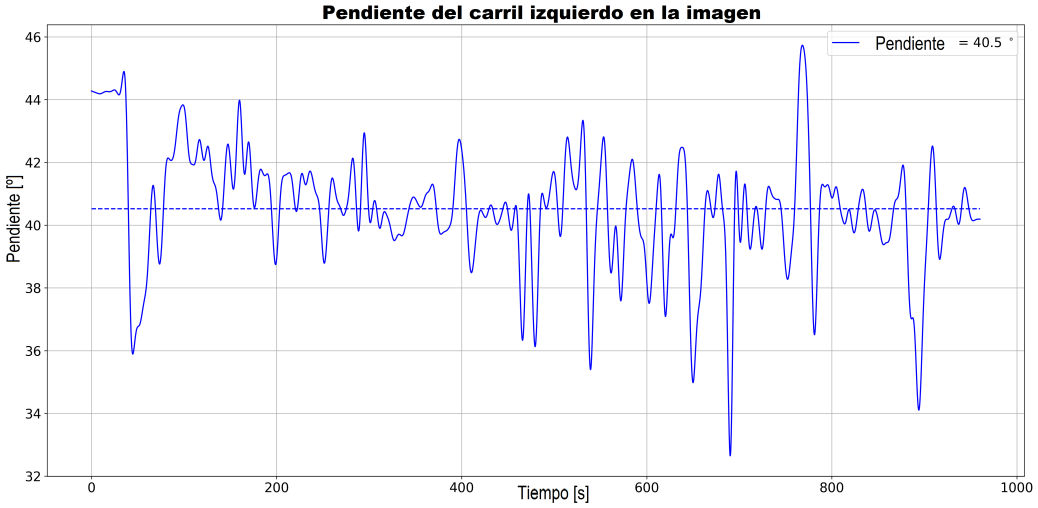

Figura 3.3: Salida (orientación) del algoritmo, cámara izquierda.

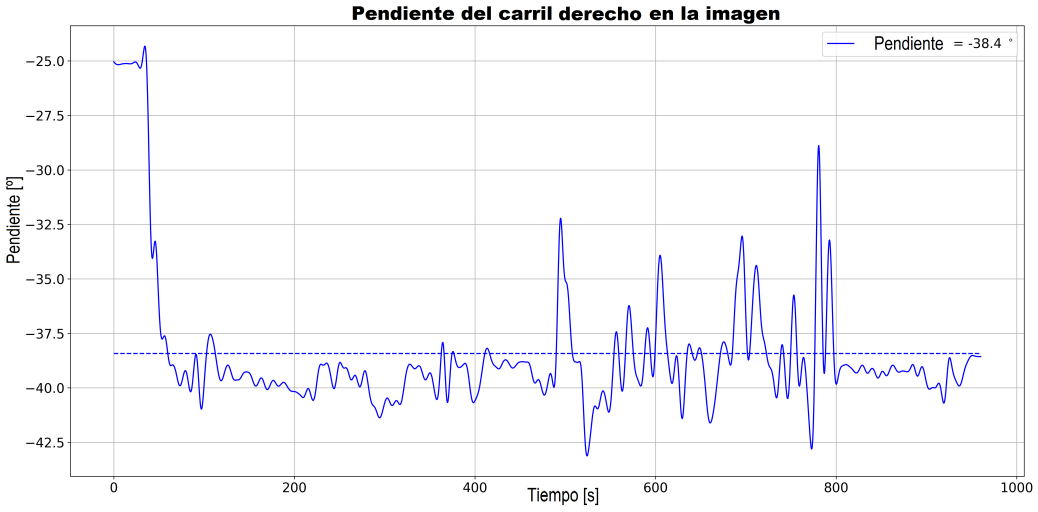

Figura 3.4: Salida (orientación) del algoritmo, cámara derecha.

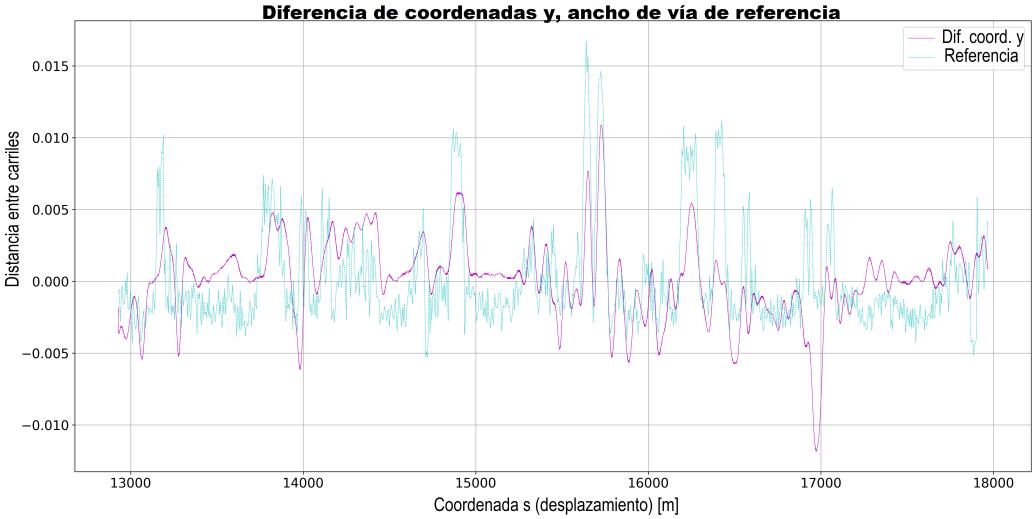

Figura 3.5: Distancia entre carriles y medida de referencia.

son suficientes por si solas para comprobar si el algoritmo está funcionando correctamente; deben ser comparadas con alguna referencia.

No se tienen datos de la posición y orientación que debería tomar el perfil del carril en las imágenes a lo largo del ensayo. Lo que si se obtiene durante la auscultación de referencia es la medida de la distancia entre carriles. Cómo la distancia entre las cámaras izquierda y derecha es conocida es posible calcular la distancia entre los carriles de forma sencilla, y comparar esta medida con la medida de referencia tomada durante la auscultación de las vías anterior a la realización del ensayo. Estas medidas de referencia fueron proporcionadas por el Metro de Sevilla.

En la figura 3.5 se muestra la comparación entre la distancia entre los carriles según los datos del algoritmo (morado) y según las medidas de referencia (celeste). No es inmediata la comparación de la distancia de las cámaras y la distancia de referencia. Por un lado la posición dada por el algoritmo está en píxeles mientras que la medida de referencia está en milímetros. Es necesario aplicar las ecuaciones de la sección 1.4 para obtener la posición del perfil en coordenadas 3D (medidas en metros). Por otro lado, las medidas de referencia vienen dadas junto a su posición longitudinal en la vía (distancia recorrida desde el kilómetro 0), mientras que el sistema de adquisición le pone una marca de tiempo al resultado del algoritmo. Gracias a la odometría del sistema de adquisición es posible asignar a cada marca de tiempo una posición longitudinal en la vía, y de esta forma poder comparar el resultado del algoritmo con las medidas de referencia. El resultado de este tratamiento de los datos se muestra en la figura 3.5.

La gráfica muestra como la tendencia de la distancia entre carriles es captada por el algoritmo de visión. En general, coinciden las subidas y las bajadas en ambas curvas. Idealmente, las dos curvas deberían ser prácticamente iguales, mientras que el resultado obtenido resta bastante de ser ideal. Sin embargo, no deben olvidarse los factores que incluyen negativamente en esta comparativa:

1. Los problemas anteriormente comentados: aparición de grasa y contracarriles, que no permiten grabar el carril por parte de las cámaras, y problemas de enfoque. En la cámara izquierda se pierden un tercio de los datos.

2. La reconstrucción 3D de la posición y orientación del perfil del carril dependen de una precisa calibración de la cámara.

3. La aplicación de filtros provoca desfases temporales.

4. Las medidas de referencia fueron tomadas meses antes del ensayo. En ese intervalo de tiempo la distancia entre carriles puede haber cambiado (aunque los cambios deben ser menores).

En el siguiente ensayo serán solucionados lo problemas de visibilidad del carril grabando la parte exterior de la cabeza del perfil (a costa de perder la posibilidad de medir el desgaste de los carriles). Esto permitirá un mejor análisis del funcionamiento del algoritmo. En cualquier caso, los resultados obtenidos indican que el algoritmo tiene potencial para cumplir el objetivo para el que ha sido creado y cumplir las especificaciones requeridas por el proyecto, y por tanto parece razonable continuar trabajando en él.

Capítulo 4

Conclusiones y Trabajo futuro

En este capítulo se presentan las conclusiones del trabajo realizado y se describen diferentes ideas que se proponen implementar en el futuro.

4.1. Conclusiones

Las conclusiones que se obtienen del algoritmo de visión desarrollado tras analizar los datos de los ensayos son los siguientes:

1. El algoritmo cumple la especificación de velocidad de cómputo.
2. El algoritmo es capaz de seguir la posición del perfil del carril en la imagen. No se pueden obtener medidas de la precisión de los resultados de los ensayos.
3. El algoritmo es estable (no deja colgado el software de auscultación).

4.2. Trabajo futuro

Las siguientes propuestas pueden ayudar a mejorar el rendimiento del algoritmo o aumentar su funcionalidad.

4.2.1. Autosintonización

El algoritmo desarrollado tiene una cantidad no despreciable de parámetros que deben ser sintonizados. Además de afectar esto negativamente a la facilidad de uso del algoritmo, es un problema de funcionalidad. El algoritmo, tal y como ha sido presentado, es incapaz de adaptarse a cambios que pueden afectar a las imágenes como cambios de iluminación.

Por ello se propone diseñar e implementar un algoritmo independiente que sea capaz de analizar las primeras imágenes provenientes de las cámaras para sintonizar automáticamente el algoritmo de búsqueda de carril. Podría tratarse de un algoritmo iterativo que utilice ciertas métricas de la aplicación del algoritmo de búsqueda de carriles para cambiar el valor de los parámetros de éste, comparando los valores de estas métricas con valores de referencia. Por ejemplo, el número de puntos que pasan el umbral dentro de cada recuadro debe situarse alrededor de un valor que puede ser conocido a priori, conociendo el tamaño de los recuadros y el espesor de la curva del perfil del carril en la imagen. El funcionamiento sería el siguiente:

1. Al recibir las primera imagen del perfil del carril se ejecuta una versión modificada del algoritmo de búsqueda de carril (con unos parámetros por defecto) que calcular ciertas métricas de la ejecución del propio algoritmo.

2. Se ejecuta un algoritmo iterativo. El algoritmo calcula los cambios en los parámetros de la siguiente ejecución del algoritmo de búsqueda del carril, en función de los valores de las métricas calculadas anteriormente y la referencia de dichas métricas, y proceder ejecutando el algoritmo de búsqueda de carril con la siguiente imagen recibida. De nuevo se calculan los valores de las métricas para obtener nuevos cambios en los parámetros.

Un algoritmo iterativo cuidadosamente diseñado podría obtener rápidamente el valor de los parámetros que permitan encontrar con precisión la posición y orientación del perfil del carril. Uno de los mayores problemas podría ser que varias métricas podrían estas acopladas, por lo que el cambio en uno de los parámetros podría cambiar varias métricas simultáneamente.

Este esquema también permite realizar cambios en la configuración de las cámaras, como cambiar el tiempo de exposición o la ganancia, parámetros que afectan directamente en la imagen generada.

4.2.2. ROI

El algoritmo desarrollado carece de inteligencia, no es capaz de aprender. Sin embargo, dotar al algoritmo con algo de inteligencia y que sea capaz de utilizar la información del pasado en sus futuras ejecuciones podría traducirse en una notable mejora de rendimiento.

Realmente la única parte de la imagen que interesa analizar es la que contiene la parte recta. Una vez encontrado el carril en la primera imagen,

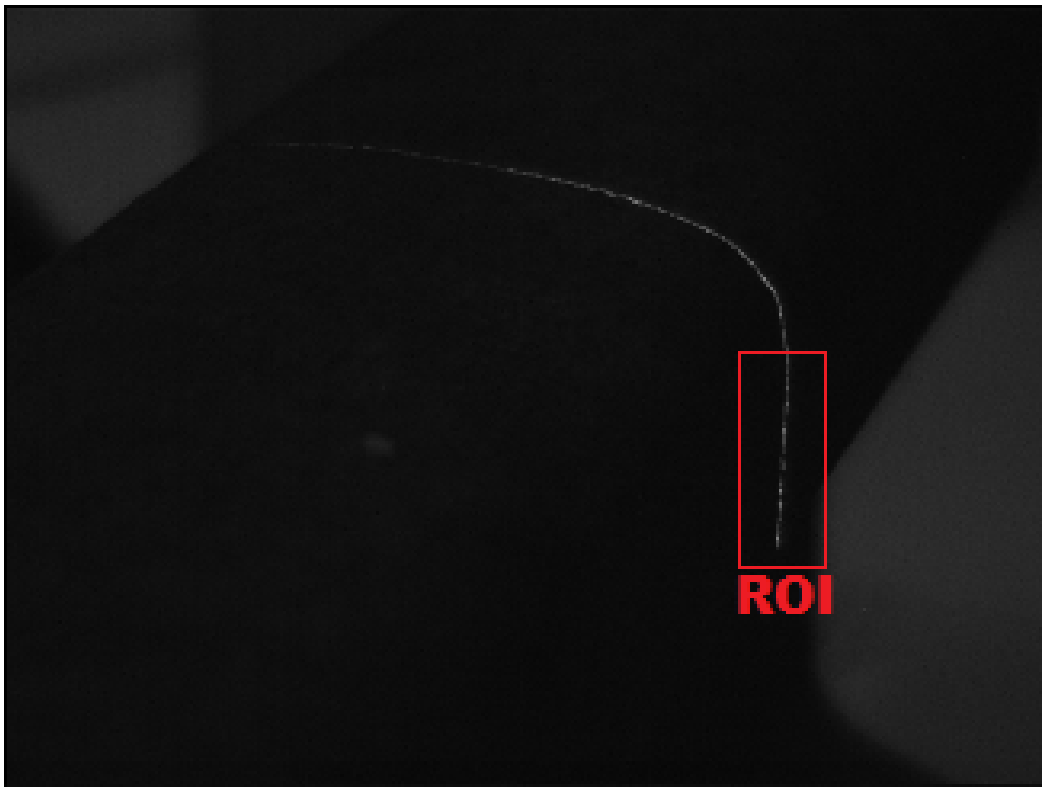

Figura 4.1: Ejemplo de Región de interés (ROI).

es posible definir una región de interés (ROI, Region Of Interest) centrada en la última posición del carril (como puede observarse en la imagen 4.1) para la siguiente ejecución del algoritmo. Esta región de interés puede ser muy ajustada al tamaño que ocupa la parte recta gracias a que el algoritmo se ejecuta con una alta frecuencia respecto a la velocidad de movimiento del perfil, y por tanto el perfil va a sufrir pequeños cambios en posición y orientación entre cada ejecución del algoritmo.

La ROI puede ser aún más ajustada si se calcula la velocidad del carril en la imagen (que podría ser muy precisa si se utiliza algún filtrado como el Filtro de Kalman), para predecir la posición en la siguiente imagen. La velocidad puede calcularse ya que es conocido el periodo de ejecución del algoritmo y la posición y orientación en las ejecuciones anteriores.

Este método puede ser utilizado sin pérdida de robustez, porque en el caso de que el algoritmo no sea capaz de encontrar el carril en la ROI, puede ser ejecutado con la imagen completa.

4.2.3. Comenzar buscando recuadros. Mejora de rendimiento

El algoritmo, en un intento de reducir el número de veces que recorre la imagen, descarta la mayoría de los recuadros de la imagen en los 3 primeros pasos. Sin embargo, en el paso 1 se recorren todos los píxeles de la imagen una vez. Aunque la mayoría de los recuadros son descartados en el primer paso, en el segundo paso se recorren los píxeles de los recuadros que no fueron descartados. En los siguientes pasos se reutiliza la información guardada en los pasos anteriores, con la excepción del paso 6 que puede acceder a los píxeles de algunos recuadros adicionales.

Con todo esto, el número de veces que se recorre la imagen es menor de 2. Sin embargo cambiando la estructura de ejecución de los primeros 4 pasos es posible reducir este número, incluso por debajo de 1. Esto puede ser conseguido comenzando primero por el paso de búsqueda de cadenas de recuadros (paso 4), ejecutando los pasos 1-3 únicamente con los recuadros que analiza el paso 4. Esta nueva estructura de ejecución de los primeros 4 pasos es mostrada en la figura 4.2.

La clave para reducir el número de píxeles recorridos de la imagen es restringir la búsqueda del primer recuadro de la cadena. El paso 4 decide si una cadena es o no la del perfil del carril atendiendo a razones de tamaño y longitud de la cadena de recuadros. Por tanto es inútil elegir como primer recuadro de cadena a recuadros que no van a permitir cumplir el mínimo de altura o anchura. Esta idea es representada en la figura 4.3.

De esta forma existirán recuadros de la imagen que nunca serán recorridos, aumentando el rendimiento. El aumento de rendimiento de este método respecto al original depende de la imagen y de las condiciones de tamaño de la cadena del perfil del carril.

4.2.4. Precisión sub-pixel

En la sección 2.2.2.1 se muestra que el tipo de dato de la posición y orientación calculadas por el algoritmo son valores en coma flotante. Sin embargo, la precisión que permite este tipo de variables sólo es utilizada por la orientación y por la coordenada x. La coordenada y tiene una precisión de un pixel, ya que es calculada aplicando una condición pixel a pixel.

El hecho de que la coordenada y no tenga mayor precisión afecta directamente a la precisión en la coordenada x, que es calculada utilizando la ecuación general de la recta. Por estas razones se propone aumentar la precisión en posición del algoritmo, calculando el punto final de la parte recta con precisión sub-pixel.

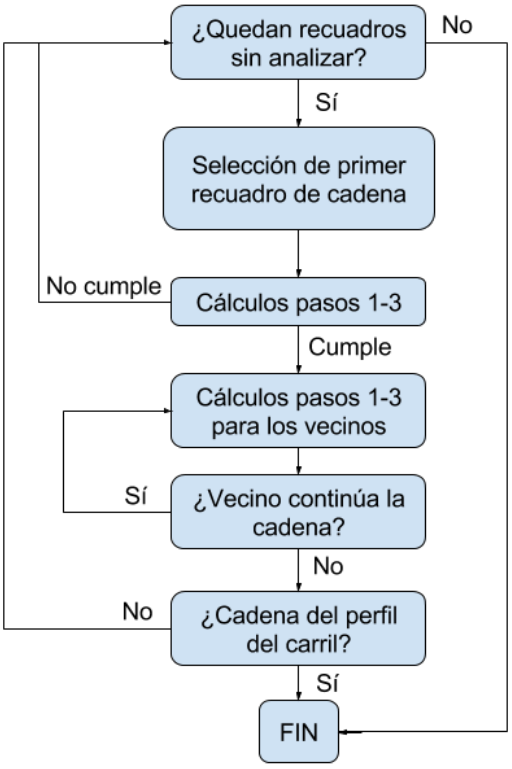

Figura 4.2: Diagrama de flujo de la nueva estructura para la mejora de rendimiento.

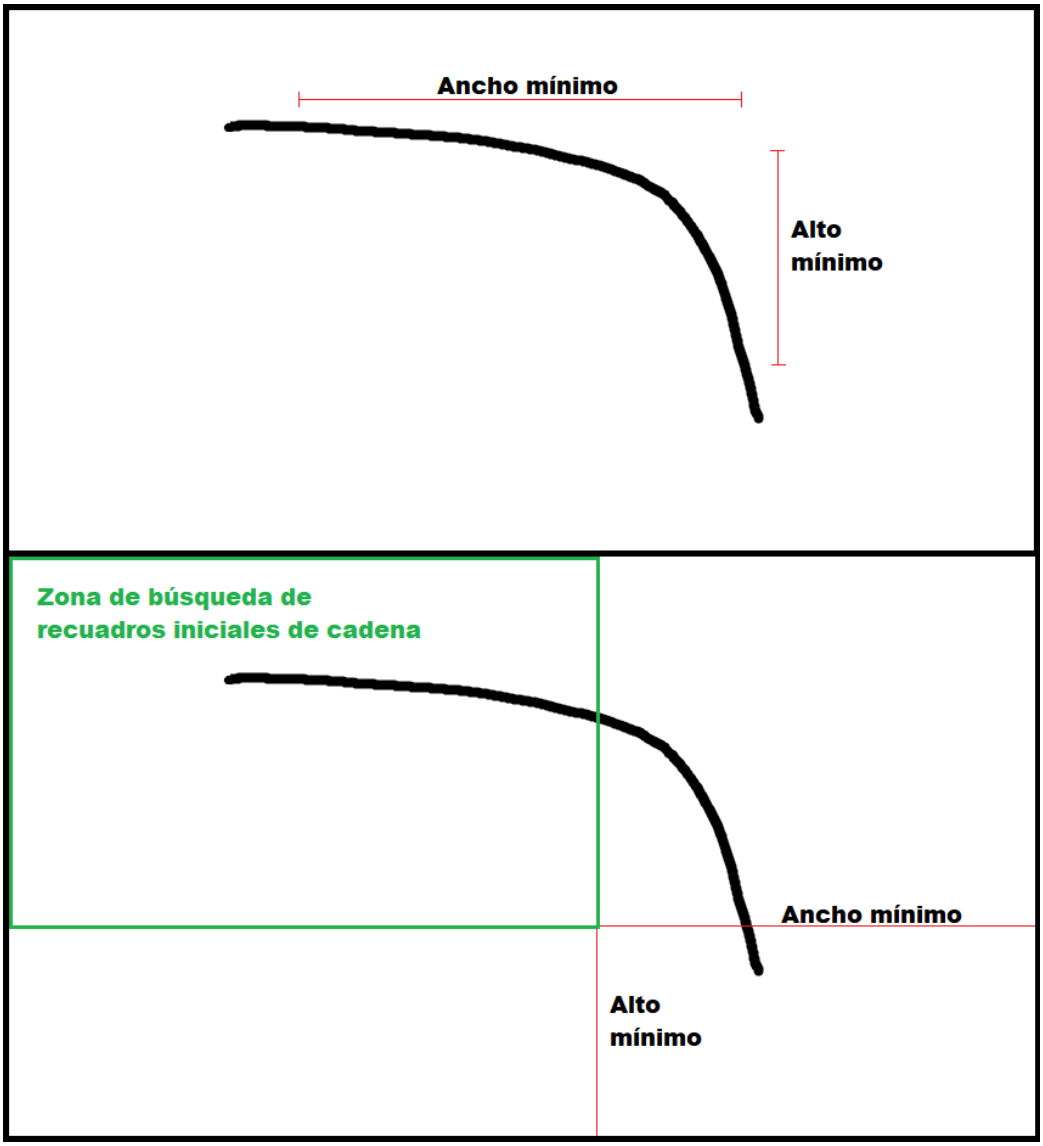

Figura 4.3: Zona de búsqueda de recuadros iniciales de cadena.

La precisión sub-pixel puede ser obtenida utilizando la información de los píxeles cercanos. Por ejemplo, podría ser calculado el ritmo de cambio de la intensidad de los últimos píxeles de la parte recta, y usar esta información para calcular en que punto del interior del pixel final ya calculado acaba la parte recta.

4.2.5. Cálculo de desgaste

Conocidas la posición y orientación del punto final de la parte recta de la cabeza del perfil en la imagen es posible reconstruir en el espacio 3D una curva completa del perfil del carril ideal (sin desgaste). Esto es utilizado para resolver el problema completo de irregularidades de la vía y dinámica del vehículo. También puede ser utilizada esta información para el cálculo del desgaste en el carril, pero para ello el algoritmo de visión debe proporcionar datos sobre la geometría de toda la curva del perfil del carril, no sólo del extremo de la parte recta. Con esta información la curva completa podría ser reconstruida en el espacio 3D, para poder ser comparada con la curva ideal, tras ser ambas curvas re-proyectadas en un plano paralelo a las curvas 3D.

El proceso de extracción de la geometría del perfil debe ser muy rápido para no afectar al rendimiento del algoritmo completo. En trabajos anteriores, antes del desarrollo del algoritmo, se realizaba un umbral a la imagen completa, de forma que quedara una imagen binaria, en la que sólo quedara el fondo de un valor y los puntos del perfil del carril de otro. Para cada columna de la imagen binaría se calculaba la posición y media de todos los puntos en dicha columna. De esta forma se calculaba una curva de espesor de un pixel que coincidía con la geometría del perfil del carril como se observa en la figura .

Esta forma de obtener la curva del perfil del carril tiene dos inconvenientes:

1. Se accede 1 vez a la imagen original y 1 vez a la imagen binaria. Se podría complicar el cumplimiento de la especificación de velocidad de cómputo.

2. Haciendo únicamente la media por columnas no se obtiene precisión cuando la curva se vuelve vertical (parte recta). No es posible realizar la media por filas porque es posible que el perfil (al ser curvo) corte en dos posiciones a una misma fila de la imagen.

A pesar de los inconvenientes de hacer una media de posición de los píxeles, es la base de la mejora aquí propuesta. Se puede utilizar la información del

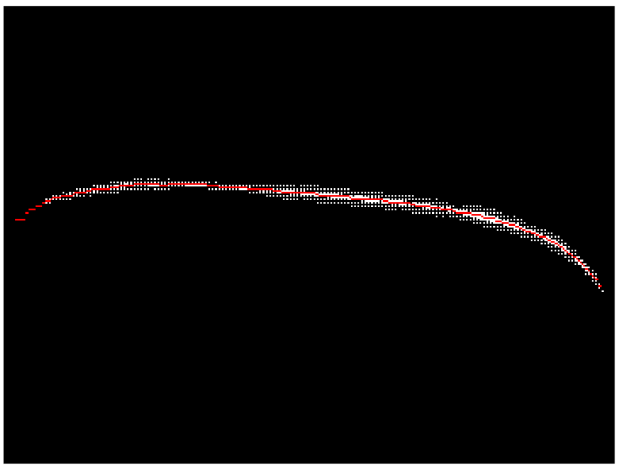

Figura 4.4: Curva del perfil del carril con espesor de un pixel.

algoritmo desarrollado para obtener la curva del perfil de una forma mucho más rápida y precisa:

1. El algoritmo ha obtenido los recuadros en los que se encuentra el perfil. Puede realizarse el cálculo de la media únicamente en estos recuadros. El umbral ya ha sido aplicado por el algoritmo.

2. Para cada recuadro de la cadena que contiene al perfil el algoritmo tiene información sobre la dirección de la curva. Esta información indica si una curva está más cerca de ser horizontal u vertical. Por tanto, puede calcularse la media por filas o por columnas dependiendo de si la curva está en cada recuadro mas cerca de ser vertical u horizontal, respectivamente. De esta forma no se pierde precisión ni densidad de puntos sea cual sea la dirección de la curva en cada tramo.

La función que ejecuta del algoritmo devolvería en este caso un vector adicional con los puntos hallados.

4.2.6. Sistema infrarrojo

En este caso no se trata de una mejora del algoritmo, sino una mejora del sistema de visión.

Aunque el sistema de visión ha dado buenos resultados, se encuentra que cuándo el carril está iluminado por luz solar directa se vuelve muy complicado

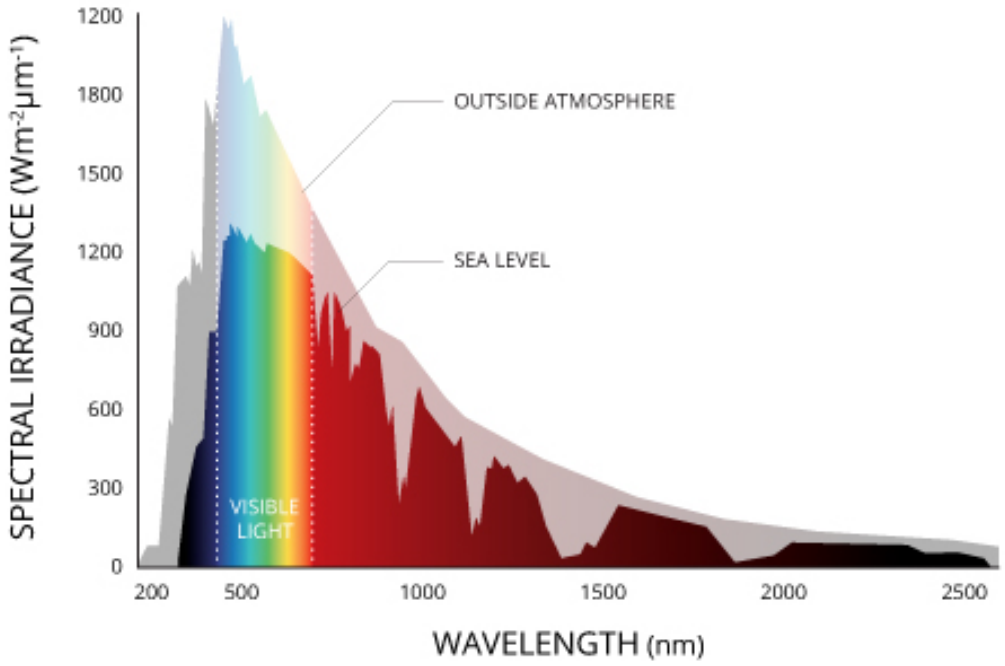

Figura 4.5: Radiación solar en la tierra [13].

distinguir el láser en la imagen, incluso mirando la escena a simple vista. Para solucionarlo se propone cambiar la longitud de onda de trabajo de rojo (700 nm) a infrarrojo (850 nm). Las ventajas de trabajar a esta nueva longitud de onda se debe a la forma del espectro de la radiación solar que llega a la superficie terrestre (figura 4.5). La zona de mayor potencia de radiación solar se da en la parte visible del espectro, justo dónde se está trabajando actualmente. Sin embargo, la potencia cae a medida que se avanza en la zona infrarroja. Por ello se pretende mover la longitud de onda de trabajo a los 850 nm con el fin de mejorar la robustez en días soleados.

Además de cambiar la frecuencia de trabajo para la cámara y el láser, se propone el uso de un filtro de paso banda centrado en la longitud de onda de 850 nm. De esta forma, la gran parte de la potencia electromagnética que reciba el sensor de la cámara será proveniente de un emisor a la frecuencia de trabajo, reduciendo las posibilidades que los píxeles del sensor de la cámara sean excitados por otra fuente de luz que no sea el láser de proyección lineal.

Bibliografía

[1] Depaoli, Roberto, Díaz, Daniel, Fernández, Luis y Stockli, Roberto, *El tratamiento de la distorsión radial en metrología efectuada con cámaras digitales*, Departamento de Ingeniería, Universidad Nacional de La Matanza.

[2] Platero, Carlos, *Apuntes de Visión Artificial*, Universidad Politécnica de Madrid, Dpto. Electrónica, Automática e Informática Industrial.

[3] MERCÈ BRUNED, *Tecnología de un escáner 3D con visión artificial*, http://www.measurecontrol.com/tecnologia-de-un-escaner-3d-con-vision-artificial/, 2009.

[4] Escalona Franco, José Luis, *Cinemática ferroviaria para su aplicación a sistemas de visión artificial*, comunicación personal, 2015.

[5] Escalona Franco, José Luis, *Desarrollo de nuevas tecnologías de auscultación ferroviaria basadas en la simulación en tiempo real*, comunicación personal, 2015.

[6] Li, M.X.D, Berggren, E.G, Berg, M. y Persson, I., *Assessing track geometry quality based on wavelength spectra and track-vehicle dynamic interaction*, Vehicle System Dynamics, 46, Suppl., 261 -276, 2008.

[7] XIMEA, *USB 3.0 camera series, Technical Manual*, proporcionado por el fabricante, XIMEA, 2015.

[8] Ruiz Arahal, Manuel, *Estimación de la posición de objetos*, Universidad de Sevilla, Escuela Técnica Superior de Ingeniería, apuntes Sistemas de Percepción.

[9] Lord MicroStrain, *3DM-GX4-25, User Manual*, proporcionado por el fabricante, Lord MicroStrain, 2015.

[10] Beena Vision Systems, *A Vehicle Based Rail and Track Profile Measurement System*, http://www.beenavision.com/, 2016.

[11] Pajares Martisanz, Gonzalo y De La Cruz garcía, Jesús M. *Visión por computador. Imágenes Digitales y Aplicaciones. 2ª Edición*, 2008.

[12] ArcelorMittal. *http://rails.arcelormittal.com/system/calculator_items/pdfs/000/000* Acedido online, Noviembre, 2017.

[13] Fundamentals of environmental measurements. *http://www.fondriest.com/environmental-measurements/parameters/weather/photosynthetically-active-radiation/*, Acedido online, Octubre, 2017.