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

ESTIMACIÓN DEL MAPA LOCAL PARA UN VEHÍCULO AUTÓNOMO

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA MENCIÓN ELÉCTRICA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

DANIEL HERRMANN PRIESNITZ

PROFESOR GUÍA:
JAVIER RUIZ DEL SOLAR SAN MARTÍN

MIEMBROS DE LA COMISIÓN:
MARCOS ORCHARD CONCHA
MIGUEL TORRES TORRITI

SANTIAGO DE CHILE
2015

Resumen

El objetivo de esta tesis es diseñar e implementar un sistema de modelamiento del entorno. Este debe recibir como entradas la morfología del terreno, imágenes segmentadas y el la pose del auto y generar en base a estas variables un modelo del entorno (*Mapa Local)* que permita a otro módulo posterior la generación de trayectorias seguras para el vehículo. Este sistema corresponde al módulo *Estimador Mapa Local* del proyecto *Vehículo Autónomo* del AMTC.

El diseño del sistema toma ideas de diversas fuentes, muchas de ellas de vehículos participantes del DARPA Grand Challenge 2005, ya que este desafío es muy similar al problema que se desea resolver. Se utilizan datos de segmentación visual y de sensores de rango, fusionando ambas fuentes de información para obtener un modelo del entorno, denominado *Mapa Local*, más preciso y robusto que cualquiera de las dos entradas de manera independiente.

Se comparan dos variantes de detecciones de obstáculos en base a las mediciones de sensores de rango. El primer método, denominado método de frecuencias, toma la información y calcula la energía en diferentes bandas de frecuencia e intenta estimar la navegabilidad del terreno en base a estos valores. El segundo método, de diferencias de altura, tiene un enfoque similar, sin embargo calcula la mayor diferencia de altura de un punto central con una vecindad de diferentes radios, luego se utilizan dichos valores de la misma manera que en el método anterior. En ambos casos se estudia el efecto de utilizar o no la información de la segmentación visual para complementar la caracterización. Tomando las características disponibles para cada método, se estima el costo de que el vehículo navegue por cada lugar del entorno, costos altos implican peligro, mientras que bajos indican zonas seguras para la navegación. Es justamente esta caracterización del entorno, en base a costos, la que se denomina *Mapa Local*.

Para probar las diferentes variantes del sistema se graban 4 diferentes bases de datos, correspondientes a mediciones realizadas por el vehículo mientras era conducido manualmente. Se diseñan medidas de desempeño para comparar el rendimiento de las variantes. Estas intentan reflejar la cantidad de eventos fatales (choques), eventos indeseados (paradas frente a detecciones falsas de obstáculos) y conducción normal, que el vehículo hubiera encontrado si fuese conducido por el sistema. Estas características se calculan para diferentes velocidades de conducción, por lo que para cada variante probada se tienen 4 curvas que caracterizan su desempeño.

Al comparar los métodos propuestos se logra concluir que la información visual enriquece el *Mapa Local* y logra un mejor desempeño, siempre y cuando se logre una fusión con bajos errores, en caso contrario esta información puede producir bajas de rendimiento. En cuanto a los métodos de detección de obstáculos, el método de diferencias de altura obtuvo una ligera ventaja en el desempeño, pero esta es muy poco significativa para seleccionar uno por sobre el otro.

Dedicatoria

A todos mis seres queridos.

Agradecimientos

Me gustaría agradecer a mi familia por el cariño, el apoyo incondicional y su paciencia en estos largos años.

Quisiera agradecerle a mi polola Tamara Sanhueza por aguantarme en los tiempos de estrés y por entender mis horarios irregulares cuando tenía alguna fecha límite. Tampoco puedo dejar de mencionar su invaluable rol en alegrarme, animarme y empujarme todos los días durante todos estos años para que tuviera las energías que me hacían falta para terminar con esta etapa.

También me gustaría agradecer a Javier Ruiz del Solar, mi profesor, mentor y algún día colega por toda su ayuda y motivación en el desarrollo de este trabajo. Gracias también son extendidas a los profesores Marcos Orchard y Miguel Torres por aceptar ser miembros de la comisión evaluadora.

Un especial reconocimiento a mis amigos de siempre, que estuvieron ahí para mi formación y siguieron ahí siempre que los necesité. Gracias por las incontables veces que discutimos ideas que ayudaron a llegar a la versión final del sistema desarrollado. Y por último, pero no menos importante, gracias por aportarme con las distracciones ocasionales, las que fueron indispensables para que lograra no colapsar y mantener mi cordura.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

En esta tesis, se mostrará el trabajo del alumno Daniel Herrmann, bajo la supervición del profesor Javier Ruiz del Solar, referente al diseño e implementación de un sistema de caracterización del entorno para la navegación no-supervisada de un vehículo autónomo, utilizando información de segmentación visual y sensores de rango (LIDAR). El trabajo forma parte de un proyecto del centro de investigación Advanced Mining Technology Center (AMTC) de la Universidad de Chile. El proyecto corresponde al área de la robótica de campo y tiene com objetivo la automatización total de un automóvil, para permitir su navegación no supervisada en terreno no pavimentado. En las siguientes secciones de este documento se describe y da a conocer la motivación detrás del proyecto, junto con los objetivos del sistema implementado. Luego, en el capítulo 3, se define el problema específico que se quiere resolver, la solución propuesta y la metodología utilizada para la implementación de la solución. En el capítulo 4 se presentan los resultados experimentales y en el capítulo 5, se presentan las conclusiones.

1.1 Motivación

La robótica de campo, es un área de la robótica, especializada en ambientes exteriores dinámicos y no controlados. Sus aplicaciones están orientadas, pero no limitadas, a las industrias de la construcción, agricultura, minería, militar y aeroespacial. Esta área se origina de la necesidad de la industria de automatizar procesos complejos en ambientes altamente dinámicos, donde múltiples interferencias no controlables pueden interferir con el funcionamiento adecuado del proceso, desde viento u cambios de iluminación, hasta personas, vehículos u otros obstáculos. Todas estas variables aumentan la complejidad de las soluciones, ya que deben ser robustas a múltiples interferencias.

Un área particular de interés dentro de la robótica de campo, es la navegación autónoma de vehículos terrestres, en la cual el objetivo es reemplazar a los operadores humanos, especialmente en ambientes peligrosos. Es sabido que el ser humano no esta diseñado para ejecutar tareas repetitivas por largos

periodos de tiempo, ya que mientras mas tiempo pasa, la concentración disminuye, aumentando de manera considerable la probabilidad de accidentes. Específicamente en el caso de la conducción de vehículos, la gran mayoría de los accidentes se producen debido a errores humanos, tal como se muestra en (INE & Carabineros, 2011). Es por esto que en este caso, reemplazar los operadores humanos con sistemas autónomos es necesario, para asegurar una conducción sin accidentes, minimizar los riesgos para el vehículo, el ambiente y las personas involucradas. Esto se aplica tanto a vehículos industriales como a vehículos de uso personal en ciudades.

1.1.1 Proyecto vehículo autónomo

Una de las industrias en las que se aplica la robótica de campo es la minería, la cual se beneficia enormemente de los avances tecnológicos, debido a que el uso óptimo de recursos y procesos ininterrumpidos son esenciales para tener éxito. Es por esto que la automatización genera avances enormes en esta industria, en cuanto a seguridad y eficiencia.

En Chile, la minería es uno de los pilares fundamentales del desarrollo de la economía, por lo que, es esencial buscar formas de incorporar nuevas tecnologías que permitan no solo a mejorar la productividad y la eficiencia, sino también la seguridad de todas las personas y equipos involucrados en el proceso, y minimizar el impacto ambiental que estos tienen.

El proyecto *Vehículo Autónomo* es un proyecto del centro de investigación AMTC de la Universidad de Chile. Este centro busca diseñar soluciones tecnológicas a la par con el estado del arte, para lograr reducir los costos, el riesgo del personal, el impacto ambiental y aumentar la producción. El proyecto está centrado en la automatización de un vehículo, para generar una iniciativa de investigación y sus objetivos son generar nuevas tecnologías tanto como el desarrollo del centro de investigación, junto con sus investigadores.

El proyecto consta de las siguientes etapas:

- *Etapa 1* - Sistemas de asistencia al operador (detección de obstáculos, detección de camino, entre otros)(1er-3er año).
- *Etapa 2* - Sistema de tele-operación (manejo del vehiculo a distancia, con información de todos los sistemas de asistencia y de las camaras, para permitir la operacion remota)(4to-6to año).
- *Etapa 3* - Operación autónoma (Sistemas de tele-operación en base a objetivos, evasión autónoma de obstáculos, generación y seguimiento de trayectorias)(7-10 año).

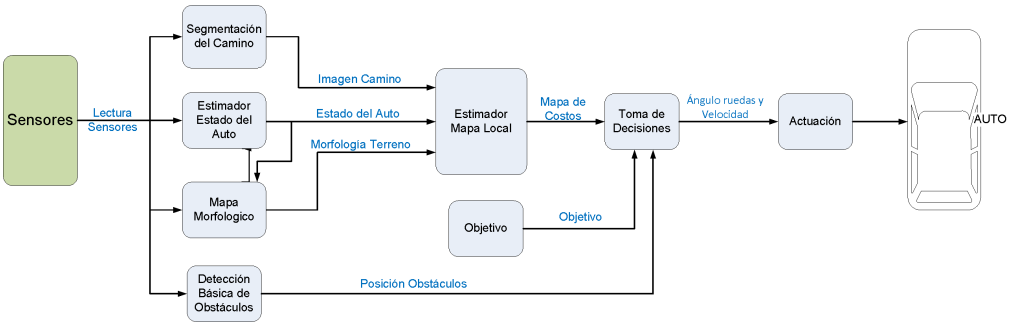


Figura 1.1: Diagrama General Proyecto Vehículo Autónomo

En la Figura 1.1 se ve el diagrama de bloques de la tercera etapa en donde se muestra el esquema de control del vehículo. A continuación se da una breve descripción de la función que cada módulo debe cumplir.

- *Segmentación del Camino* - En base a información visual de cámaras calcula diversas características en cada pixel y segmenta el camino en la imagen. Entrega una imagen binaria en la cual se marca el camino.
- *Estimador Estado del Auto* - En base a unidades de medición inerciales (IMU), GPS, la posición de los puntos de apoyo de los neumáticos (Mapa Morfológico), la posición del volante y del acelerador, se calcula el estado del vehículo, lo que incluye posición, orientación, velocidad y aceleración, todo en 3 dimensiones.
- *Mapa Morfológico* - A partir de las mediciones de los LASER y el estado del auto, se genera un mapa de elevación, el cual muestra la altura de la superficie del terreno para cada punto X,Y.
- *Detección Básica de Obstáculos* - Toma las lecturas de los LASER y los radares y detecta los obstáculos de manera rápida, la posición de los obstáculos detectados son enviados directamente al bloque *Toma de Decisiones*.
- *Estimador Mapa Local* - Se encarga de generar un Mapa Local, el cual se utiliza en el bloque *Toma de Decisiones* para generar la trayectoria óptima del vehículo. Este mapa se crea a partir de la información de la Morfología del Terreno, la Imagen segmentada del camino y el Estado del Auto.
- *Objetivo* - Define el modo de conducción y el objetivo (Ir a un punto especifico, seguir la trayectoria, entre otros). Se envía el objetivo al bloque Toma de Decisiones de manera que se tomen las decisiones adecuadas para lograr el objetivo.

- *Toma de Decisiones* - Como primera tarea, tiene que verificar si hay algún Obstáculo, detectado por el módulo de Detección Básica de Obstáculos, en la trayectoria del vehículo y evadirlo. En caso contrario, se analiza el Mapa local y se genera una trayectoria óptima a seguir, para minimizar los riesgos y lograr el objetivo. Como segunda tarea, se traducen las trayectorias a una serie de ángulos de las ruedas y velocidades, las que son enviadas al módulo siguiente.
- *Actuación* - Es el bloque encargado del control de bajo nivel. Hace el control necesario para que las ruedas cumplan con las velocidades y ángulos requeridos.

1.2 Definición del problema y objetivos

Para poder lidiar con las cambiantes condiciones del ambiente y las diversas perturbaciones que este causa al sistema, es fundamental medir o estimar con alta precisión todos los factores que podrían afectar las acciones del robot. Esto hace que la estimación de la vecindad del robot sea esencial en este tipo de aplicaciones, ya que cada decisión a tomar tiene que considerar cualquier obstáculo cercano, los caminos por los que hay que maniobrar y el destino. En el caso de un vehículo terrestre, sensores inerciales, junto con un GPS y encoders en las ruedas, son comúnmente usados para estimar la posición real del robot, mientras que lasers, radares y cámaras, son utilizados para caracterizar las inmediaciones del vehículo, áreas navegables y ubicación de obstáculos. En el caso del proyecto Vehículo Autónomo del AMTC esto no es diferente, el bloque Estimador Mapa Local es el encargado de resolver este problema, y como se mostró anteriormente este recibe información de lasers, imágenes y el estado del auto, el cual contiene sensores inerciales, encoders y GPS.

1.2.1 Objetivos generales

El objetivo de esta tesis es diseñar e implementar un sistema que reciba la información asociada a la morfología del terreno, imágenes segmentadas y el estado del auto, generando un modelo del entorno, llamado mapa local, que permita a otro sistema posterior, la generación de trayectorias seguras para la conducción y que logren cumplir con el objetivo actual del vehículo. Este sistema será utilizado en el proyecto Vehículo Autónomo del AMTC.

1.2.2 Objetivos específicos

Para lograr diseñar e implementar el sistema deseado, es necesario que este cumpla con diversos requisitos:

- Integración del sistema con el proyecto Vehículo Autónomo del AMTC: El sistema debe estar programado en C++, utilizar ROS (Robot Operating system) y las entradas y salidas al sistema deben estar diseñadas para ser compatibles con los bloques previos y posteriores del diagrama general mostrado en la Figura 1.1.

- Evitar la dependencia completa a las imágenes segmentadas y lograr ser robusto frente a falta o inestabilidad de estas. Ya sea por falla de la cámara, malas condiciones de iluminación, polvo, brillos o muchas otras posibles interferencias, las imágenes segmentadas por si solas no son una medición confiable para definir lo que es camino u obstáculo. Por esto mismo se debe diseñar el sistema para que sea capaz de funcionar cuando no hay imágenes nuevas y capaz de ignorar mediciones erróneas.

- Lograr lidiar con las mediciones incompletas de los sensores laser, de manera de no provocar demasiadas paradas innecesarias ni choques. Debido a irregularidades que provocan oclusiones, no es posible obtener datos completos de la superficie. El sistema debe ser capaz de lidiar con zonas de la superficie de la cual no se tiene información.

- Diseñar la salida del sistema de manera que permita encontrar trayectorias seguras para el vehículo, marcando zonas obstaculizadas, zonas seguras y zonas desconocidas.

- Diseñar e implementar una metodología de evaluación para la salida del sistema que permitan validar o rechazar los resultados obtenidos. Estas deben lograr definir cual de las posibles opciones probadas da una mejor solución al problema.

1.2.3 Hipótesis

El uso de información visual segmentada en la caracterización del entorno del vehículo complementa y enriquece el modelo de la morfología del terreno, permitiendo suplir vacíos de información intrínsecos de este. Gracias a esta fusión de información el sistema logrará entregar un modelo del entorno más completo y preciso, permitiendo una conducción más segura para el vehículo, a pesar de la naturaleza poco robusta de las imágenes.

1.2.4 Estructura del documento

En este documento tiene la siguiente estructura. En el Capítulo 2 se muestra una breve revisión bibliográfica que muestra vehículos autónomos de buen desempeño y que tienen objetivos similares a los del proyecto. El Capítulo 3 detalla la metodología utilizada para resolver el problema, junto con

detalles de la implementación de éste. Luego en el Capítulo 4 se muestran los experimentos realizados, sus resultados y un análisis de estos. Finalmente en el Capítulo 5 se exponen las conclusiones obtenidas y se dan ideas que lograrían mejorar los resultados obtenidos.

Capítulo 2

Estado del arte

Para lograr un diseño eficiente que cumpla con los requisitos y que sea competitivo con otras aplicaciones de vehículos autónomos, es necesario estudiarlas y analizar sus ventajas y desventajas. A continuación se presenta resumidamente el estudio de diferentes sistemas de navegación autónoma, sus metodologías de modelamiento del entorno, modelamiento del camino y sus sensores utilizados.

2.1 Sistemas de navegación autónoma

Para lograr los objetivos, primero se investigaron diversos sistemas de navegación autónoma ya funcionales y se analizaron los bloques equivalentes al desarrollado en esta Tesis. Con este objetivo, se centró la investigación en los participantes de la iniciativa DARPA Grand Challenge (Defense Advanced Research Projects Agency). Esta competencia fue impulsada el 2003 para fomentar la innovación en navegación de vehículos terrestres no tripulados. El objetivo de esta era la creación de vehículos autónomos capaces de navegar a través de terreno desconocido no-pavimentado. La primera competencia se desarrolló en marzo del 2013 y el premio al primer lugar era de 1 millón de dolares. La competencia consistía en navegar por una ruta de 142 millas a través del desierto Mojave en no mas de 10 horas. 107 equipos se registraron y 15 corrieron, pero ninguno de los robots participantes navegó mas de un 5% de la ruta, tal como se ve en (Buehler, Iagnemma, & Eds, 2006). La competencia se repitió en Octubre del 2005, esta vez con un premio de 2 millones de dolares. De los 195 equipos que se registraron, 23 corrieron y 5 lograron terminar la ruta. De los primeros lugares se detallarán las secciones relevantes para el sistema que se desarrollara.

2.1.1 Stanley

Figura 2.1: Stanley: Primer lugar DARPA Grand Challenge

El robot Stanley (Figura 2.1), de la conocida universidad Estadounidense Stanford, obtuvo el primer lugar del DARPA Grand Challenge 2005 con un tiempo de 6 h, 53 min y 58 s, tal como se muestra en (Thrun et al., 2006). El equipo que lo diseñó y construyó estaba dividido en 4 grupos principales:

- *Grupo del Vehículo:* Se encarga de todo lo relacionado al vehículo, incluyendo el mando a distancia, los computadores, sensores y sus respectivas monturas, entre otros.
- *Grupo de Software:* Desarrolla todo el software, incluyendo el software de navegación y los diversos sistemas de monitoreo disponibles.
- *Grupo de Pruebas:* Prueba cada cada componente del sistema y el sistema completo de acuerdo a un riguroso calendario.
- *Grupo de Comunicaciones:* Encargado de todas las relaciones con los medios y los eventos de recaudación de fondos.

En esta sección, trabajo se investigará una parte específica del trabajo del Grupo de Software, los módulos encargados de modelar el entorno del vehículo y que permiten a módulos posteriores tomar decisiones y controlar al vehículo de manera óptima y segura.

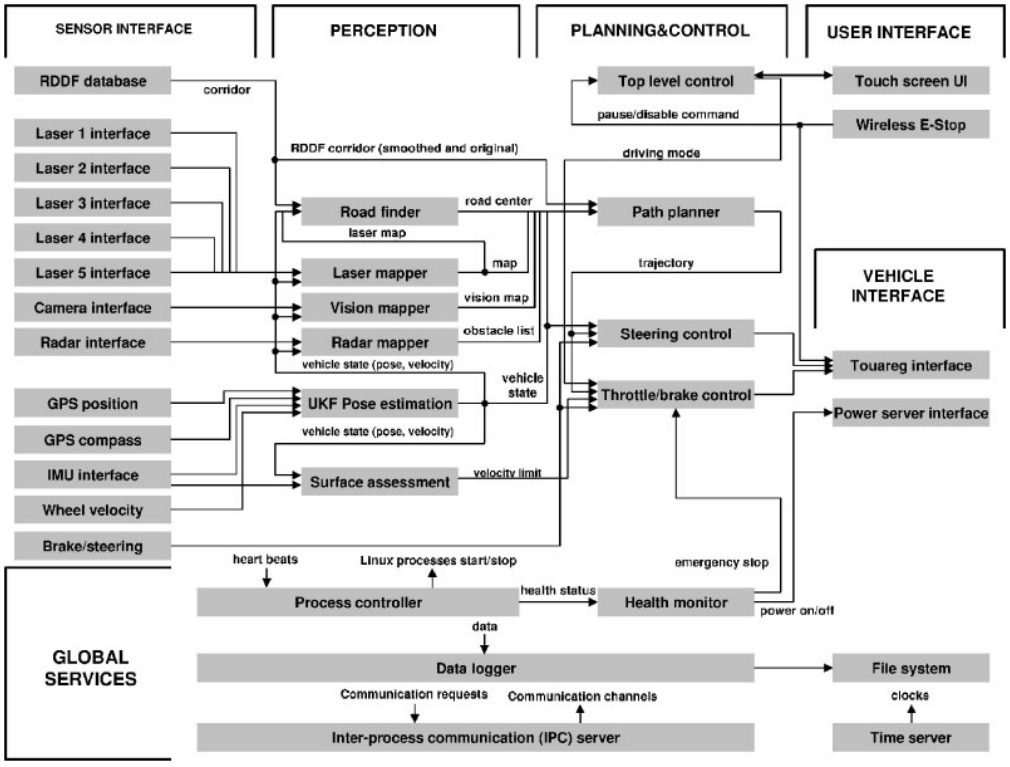
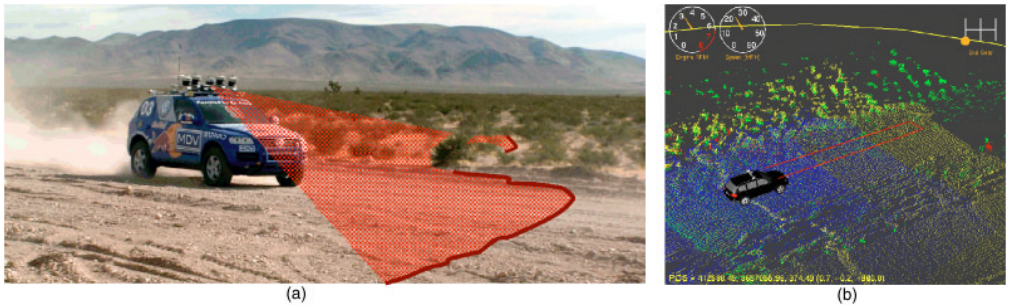


Figura 2.2: Diagrama de Bloques del Sistema de Control de Stanley, obtenido de (Thrun et al., 2006)

Tal como se observa en la Figura 2.2, existen diversos módulos de modelación del entorno. A continuación se listan y explican los módulos módulos mas relevantes para el trabajo:

- *Lasser Mapper:* Este módulo, se encarga de modelar el entorno del robot en base a mediciones de laser. Tal como se observa en la Figura 2.2, este recibe información de los 5 sensores laser montados en el vehículo y de el módulo de estimación de pose. Este último, tal como su nombre lo indica, estima la pose del vehículo, lo cual es esencial para poder integrar las mediciones de cada sensor a medida que el robot se mueve.



(a) (b)

Figura 2.3: obtenida de (Thrun et al., 2006).(a) Ilustración de un sensor laser: El sensor está orientado hacia abájo para sensar el terreno en frente del vehículo a medida que este se mueve. Stanley posee 5 de estos sensores, montados en 5 direcciones diferentes. (b) A lo largo del tiempo, se obtiene una nube de puntos 3D para cada sensor. Esta nube es analizada para encontrar terreno navegable y obstáculos potenciales.

En la Figura 2.3 se observan los 5 sensores laser montados en el techo del automovil y se ilustra la orientación de uno de estos en la Figura 2.3-(a). Como se ve, los sensores laser están inclinados ligeramente hacia abajo. Cada uno de estos sensores estan orietados en diferentes ángulos, de manera que observan secciones diferentes del terreno. En la Figura 2.3-(b) se muestran las nubes de puntos generadas por los sensores en diferentes colores. Estas nubes se general al integrar las mediciones de los sensores a lo largo del tiempo y tal como se observa, a medida que el vehículo avanza, muchas secciones medidas por un sensor, eventualmente es observada por otro, lo que permite reafirmar observaciones o descartarlas.

El algoritmo utilizado por este módulo de Stanley para poder encontrar obstáculos y zonas libres consiste en lo siguente:

- Una locación se define como obstáculo si se encuentran 2 mediciones cercanas cuya distancia vertical sea mayor a un umbral.
- Una locación se define como navegable si se encuentran 2 mediciones cercanas cuya distancia vertical sea menor a un umbral.
- Una locación se define como desconocida si se encuentran menos de 2 mediciones en una vecindad determinada.

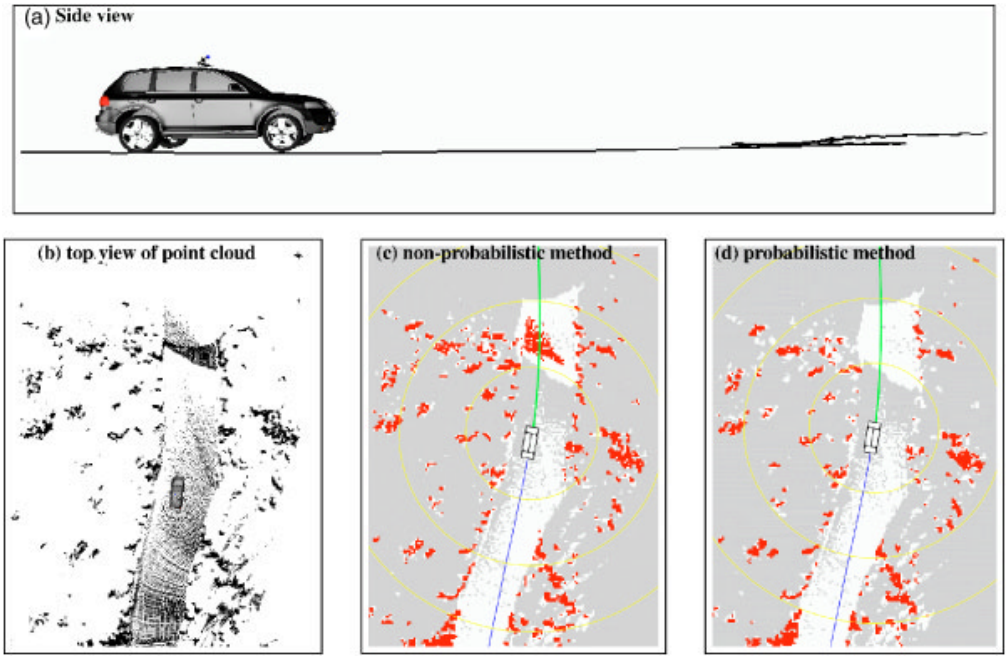


Figura 2.4: Ejemplo de pequeños errores en la orientación del robot: (a) Lectura del haz central de uno de los lasers. Al realizar la integración en el tiempo, algunas secciones de terreno son escaneadas 2 veces. (b) Muestra la nube de puntos. (c) Mapa resultante sin análisis probabilístico, el rojo muestra obstáculos, blanco camino libre y gris zonas sin datos. (d) Mapa con análisis probablilístico, en donde el obstáculo falso ya no es detectado.

El mapa generado debe ser robusto a pequeños errores en la pose del robot, pero tal como se observa en la Figura: 2.4, cuando hay pequeños errores en la orientacion del robot, genera que mediciones de diferentes secciones de terreno se perciban como el mismo, generando una especie de traslape de mediciones tal como se observa en las Figuras 2.4-(a) y 2.4-(b). Debido al algoritmo utilizado, estos errores producen obstáculos ficticios, mostrados en la Figura 2.4-(c). Para remediar estos problemas, utilizan metodo probabilístico, el cual utiliza la correlación entre la diferencia de altura de 2 mediciones cercanas y el tiempo transcurrido entre estas como se ilustra en la Figura 2.5. Los resultados de este metodo probabilístico se observan en la Figura 2.4-(d).

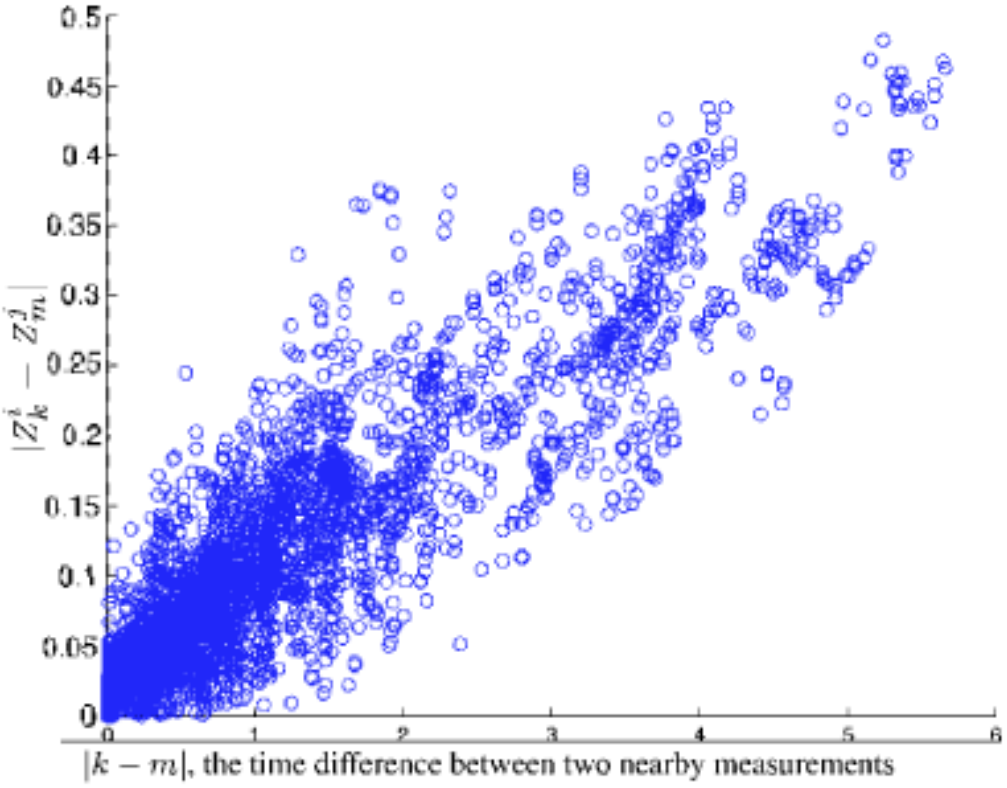


Figura 2.5: Correlación entre diferencia temporal de medidas cercanas y la distancia vertical entre estas.

Para el ajuste de parámetros, es necesario contar con datos etiquetados con las clases correctas (Obstáculos y zonas navegables), para así poder calibrar el sistema. Etiquetar una base de datos a mano es un trabajo monumental, por lo que el equipo de Stanford optó por usar una estrategia diferente. Generaron una base de datos con el vehículo conducido manualmente y se etiquetó como terreno navegable toda área atravesada por el vehículo, mientras que se etiquetó como obstáculo el terreno en 2 franjas a una distancia fija del vehículo. Esto se puede observar en la Figura 2.6

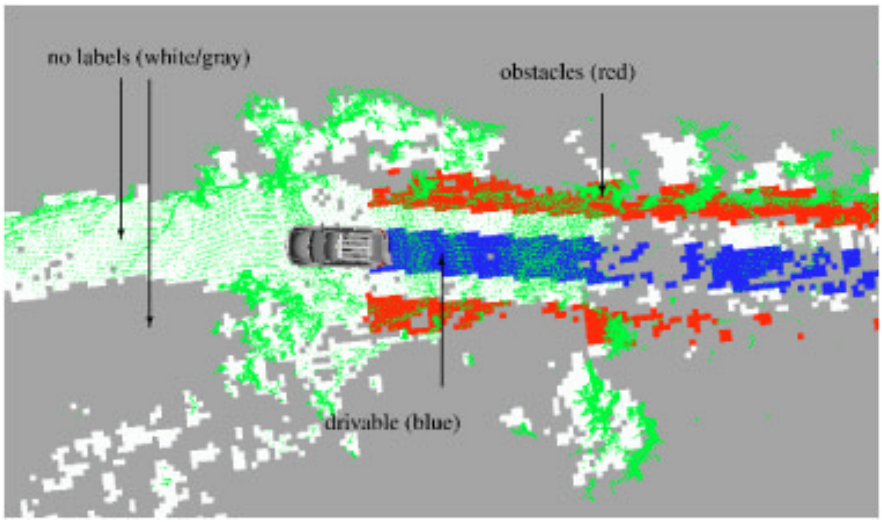


Figura 2.6: Etiquetado de terreno para ajuste de parámetros del detector de obstáculos. El área recorrida por el vehículo es etiquetada como navegable (azul), mientras que 2 franjas a una distancia fija del vehículo a la derecha e izquierda, son etiquetadas como (obstáculos).

- *Vision Mapper:* Este bloque se encarga de segmentar el camino, generar un mapa local proyectando el camino y utilizar este mapa solamente para limitar la velocidad del vehículo cuando no se encuentre camino manejable hasta los 40 m.

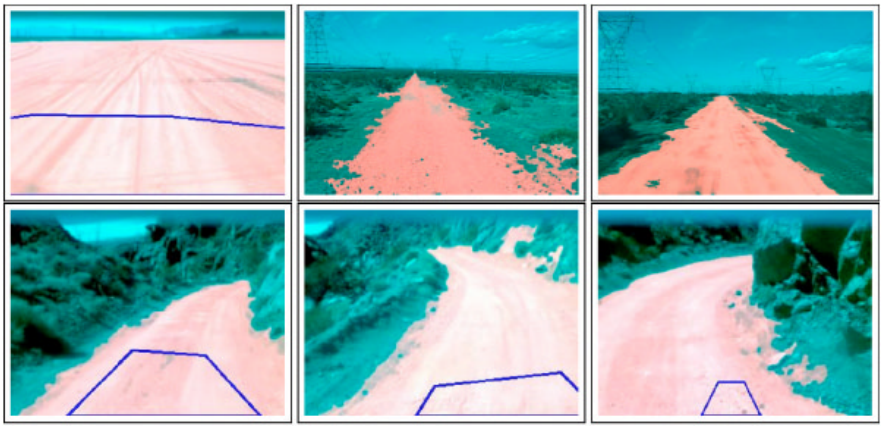


Figura 2.7: Ejemplos del algoritmo de segmentación procesando diferentes terrenos enfrentados en el DARPA Grand Challenge

En la Figura 2.7 se muestran ejemplos del algoritmo de segmentación funcionando. Este algoritmo esta basado en segmentación de color en el espacio RGB. Se observa en la figura mencionada recientemente unos cuadriláteros azules, estos representan el área de entrenamiento. El cuadrilátero es una proyección de una sección navegable en el mapa generado por el laser. El algoritmo de aprendizaje mantiene un modelo de múltiples gaussianas definidas en el espacio

de color RGB. El numero total de gaussianas es denominado n, para cada una se mantiene un color medio μ_i, una covarianza σ_i y una cuenta m_i del numero total de pixeles usados para entrenar esa gaussiana. Este modelo se debe adaptar a cambios en iluminación y también a cambios en el tipo terreno. Cada vez que se observa una nueva imagen los pixeles del cuadrilátero manejable se utilizan para entrenar un k, menor que n, gaussianas locales. Luego estas se fusionan con las gaussianas del modelo utilizando una de 2 estrategias con cada gaussiana local:

- *Adaptación lenta* - Si la distancia de la nueva gaussiana local con la gaussiana del modelo mas cercana es menor a un umbral ϕ se cambia la gaussiana del modelo por la suma ponderada de ambas.
- *Adaptación rápida* - Si la distancia de la nueva gaussiana local con la gaussiana del modelo mas cercana es mayor a ϕ agrega una nueva gaussiana al modelo. Si el máximo numero de gaussianas estan siendo utilizadas, la con menor numero de pixeles es reemplazada por la nueva.

Ambos métodos son esenciales para la correcta adaptación del modelo. En la Figura 2.8 se observa la adaptación rápida actuando y adaptando el modelo del camino a un nuevo terreno.

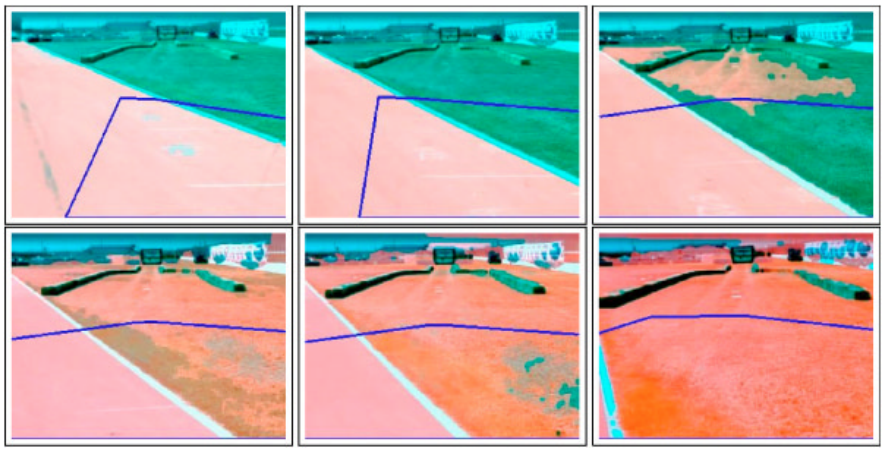


Figura 2.8: Ejemplos del algoritmo de aprendisaje adaptando el modelo de multiples gaussianas a un nuevo terreno.

Posteriormente esta detección de camino es proyectada a un plano y en este nuevo mapa se busca si el camino detectado permite la conducción segura hasta al menos 40 m. Si este no fuera el caso, se limita la velocidad del vehículo a 25 km/h velocidad a la que el mapa generado por los LIDAR es mas que suficiente para una conducción segura.

2.1.2 Sandstorm y H1lander

Figura 2.9: Sandstorm (izquierda) y H1lander (derecha), robots creados por *Red Team*, obtuvieron el segundo y tercer lugar respectivamente.

Los robots "Sandstorm" y "H1lander" (fig:2.9), creados por el equipo de róbotica (*Red Team*) de la universidad *Carnegie Mellon* en Pittsburgh, Estados Unidos. Obtuvieron el segundo y tercer lugar respectivamente en el DARPA Grand Challenge 2005 con tiempos de 7 h, 4 min y 7 h, 14 min, tal como se muestra en (Urmson et al., 2006). Ambos robots son muy diferentes en sus actuadores y en las estrategias de control de bajo nivel, pero son muy similares en el hardware y software de alto nivel. El equipo decidió tener 2 robots en lugar de uno, a pesar del tiempo extra, esfuerzo y personal que requiere, debido a que esto les permitiría siempre tener al menos un robot disponible para probar software, lo que consideraron esencial para la robustez del sistema.

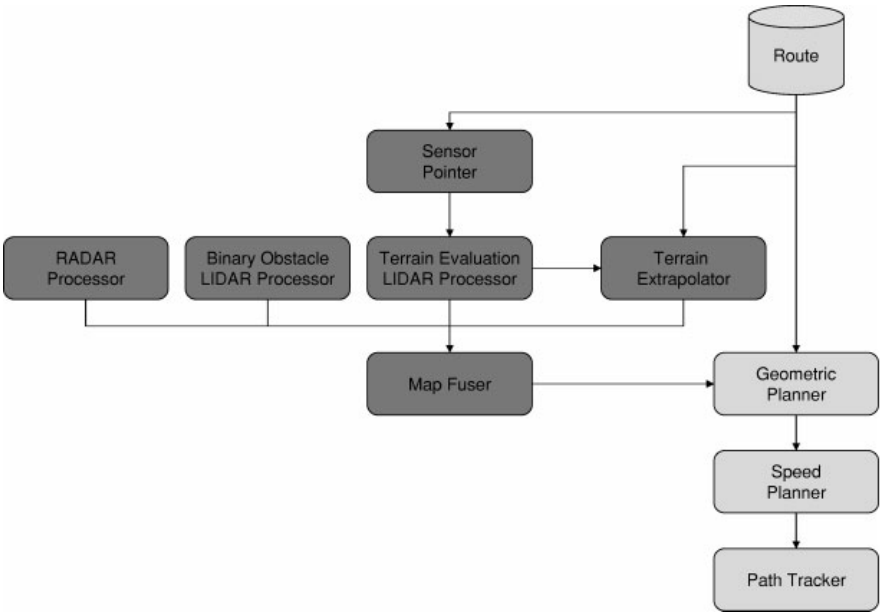


Figura 2.10: Diagrama de bloques del software de Sandstorm y H1lander.

En la Figura 2.10 se muestra el diagrama de bloques del software de ambos robots. Se ve que se utilizan múltiples entradas de sensores que luego son fusionadas. Para esto cada sensor o agrupación de sensores genera un mapa local, cada uno con su confidencia. Cada vehículo cuenta con un RADAR y con 5 LIDAR, los cuales están divididos en 2 grupos. En la Figura 2.11 se muestran los sensores montados en ambos robots. El RADAR se utiliza por si solo. El Riegl LIDAR y ambos SICK LMS LIDAR montados en la parte superior de los vehículos son utilizados para analizar el terreno. Finalmente los 2 SICK LMS LIDAR montados en el parachoques son utilizados para la detección binaria de obstáculos.

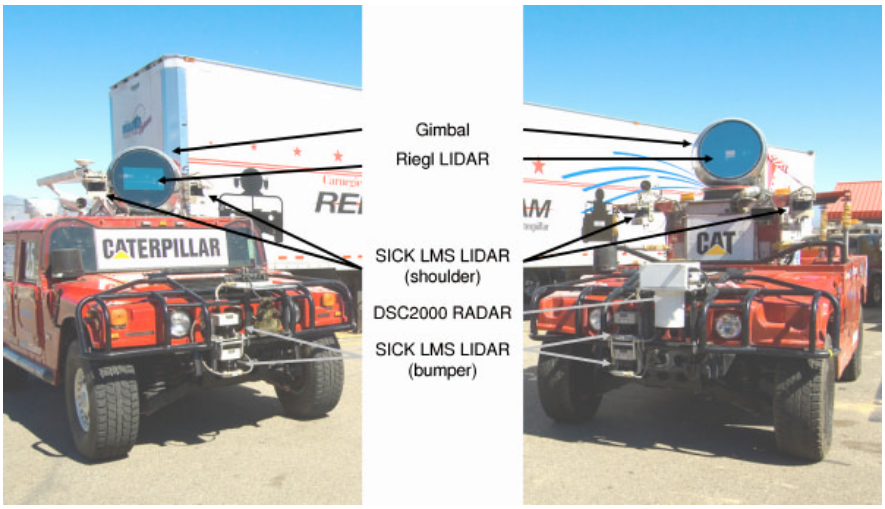


Figura 2.11: Posición de sensores montados en ambos robots.

Con cada uno de los grupos de sensores mensionados anteriormente se estimas mapas describiendo el entorno, los cuales son fusionados por el bloque *Map Fuser*. El funcionamiento de este bloque es muy relevante para esta tesis.

El bloque *Map Fuser* es bastante simple, realiza simplemente una suma ponderada de los mapas calculados por cada agrupación de sensores. El peso de cada sensor es la confidencia de cada mapa de entrada. Luego de diversas pruebas decidieron utilizar confidencias constantes para cada sensor, ya que esto daba buenos resultados.

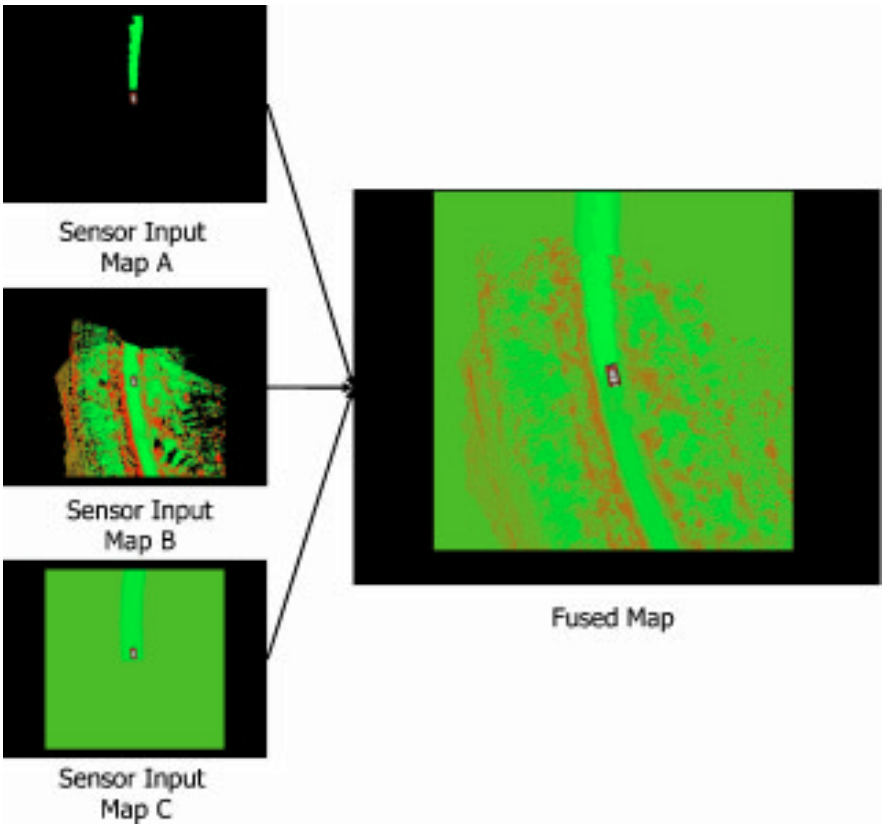


Figura 2.12: Ejemplo del bloque de *Map Fuser* realizando la fusión de 3 mapas de entradas.

En la Figura 2.12 se observa el módulo *Map Fuser* en funcionamiento. En este, los 3 mapas de entrada provienen de cada uno de los 3 grupos de sensores descritos previamente. Cabe destacar que las zonas desconocidas no se consideran al momento de realizar la suma ponderada, tal como se observa en la figura anterior.

2.2 Modelamiento del Entorno

En los robots analizados en la sección anterior y en otros diversos sistemas analizados ((Crane III et al., 2006), (Häselich, Arends, Lang, & Paulus, 2011) y (Papadakis, 2013), entre otros) se utilizan mapas locales para modelar el entorno. En el caso de las aplicaciones más similares, como en la mayoría de los vehículos del DARPA Grand Challenge 2005, se utilizan mapas de costos para modelar la navegabilidad de cada celda del mapa local y así poder decidir por una trayectoria que evite zonas peligrosas. A continuación se describen algunos de estos sistemas.

2.2.1 NaviGATOR

NaviGATOR es un vehículo autónomo del Centro para Máquinas Inteligentes y Robótica (CIMAR) de la Universidad de Florida.

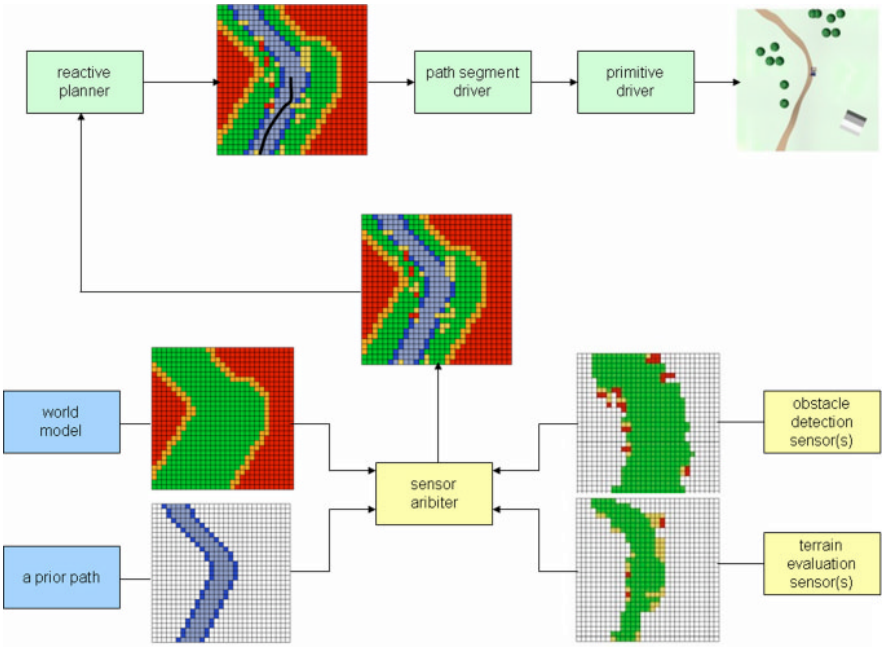


Figura 2.13: Sistema de modelamiento del entorno de NaviGATOR vehículo autónomo del equipo CIMAR.

En la Figura 2.13, se ve el funcionamiento del sistema de modelamiento del entorno de NaviGATOR. En este sistema se observa que se realiza una fusión de diferentes fuentes de información. *World Model* y *Prior Path* provienen de información previa del terreno. *Obstacle Detection Sensor(s)* y *Terrain Evaluation Sensor(s)* se generan en base a mediciones de diversos sensores. Es importante destacar que la salida del sistema se interpreta como un mapa de navegabilidad, esto quiere decir

que en cada celda contiene un número que representa la navegabilidad de esta.

2.2.2 Clasificación de Terreno, por Haselich

En (Häselich et al., 2011) se diseña un sistema de modelamiento del entorno que fusiona información visual con sensores laser. A partir de esta fusión se genera un modelo del entorno una grilla 2D que clasifica cada celda con las clases *desconocido*, *camino*, *disparejo* y *obstáculo*. Esto se muestra en la Figura 2.14.

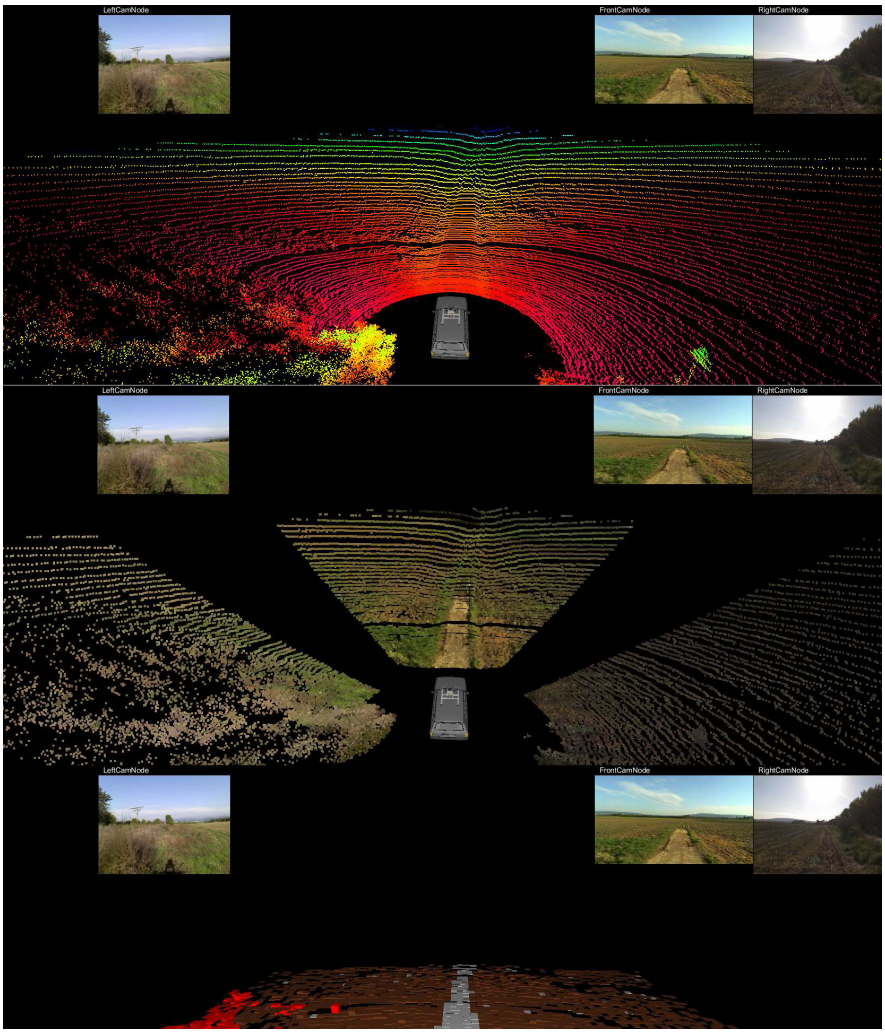
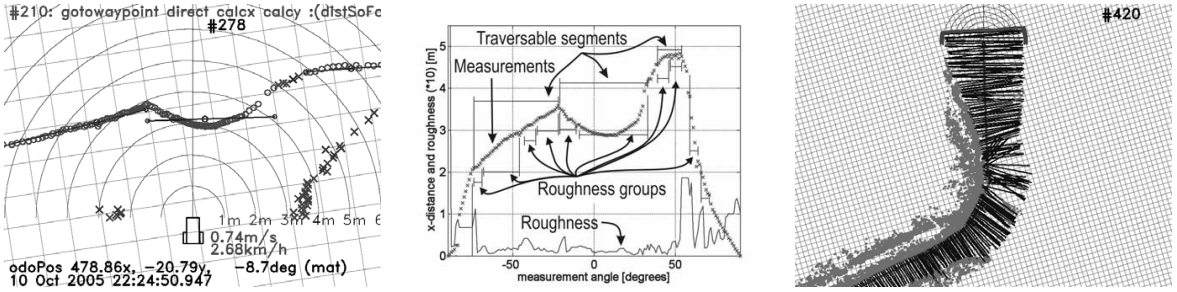


Figura 2.14: Fusión sensorial y modelamiento del entorno, obtenido de (Häselich et al., 2011).

2.2.3 Papadakis

El sistema diseñado en (Papadakis, 2013) se basa en el análisis de un solo escaneo de Laser. Tal como se muestra en la Figura 2.15, se encuentra la rugosidad de cada punto en el escaneo. Luego se generan y clasifican segmentos, de acuerdo a su rugosidad. En la Figura 2.15-(c) se muestran los segmentos navegables, con los cuales se genera un modelo del entorno.



(a) Vista Superior Datos Laser **(b)** Datos Laser y Rugosidad **(c)** Segmentos Navegables

Figura 2.15: Sistema de detección de segmentos navegables, obtenido de (Papadakis, 2013).

2.2.4 Urmson

Ya se analizó en bastante detalle el funcionamiento de los sistemas de navegación autónoma de Sandstorm y H1lander, pero en esta sección se quiere destacar el sistema de fusión sensorial y modelación del entorno que utilizan estos 2 vehículos. En la Figura 2.16 se observa la fusión de diversas entradas y el mapa consolidado. Este mapa, se interpreta como un mapa de costos, donde cada celda contiene un número que representa el costo que el vehículo navegue a través de esta.

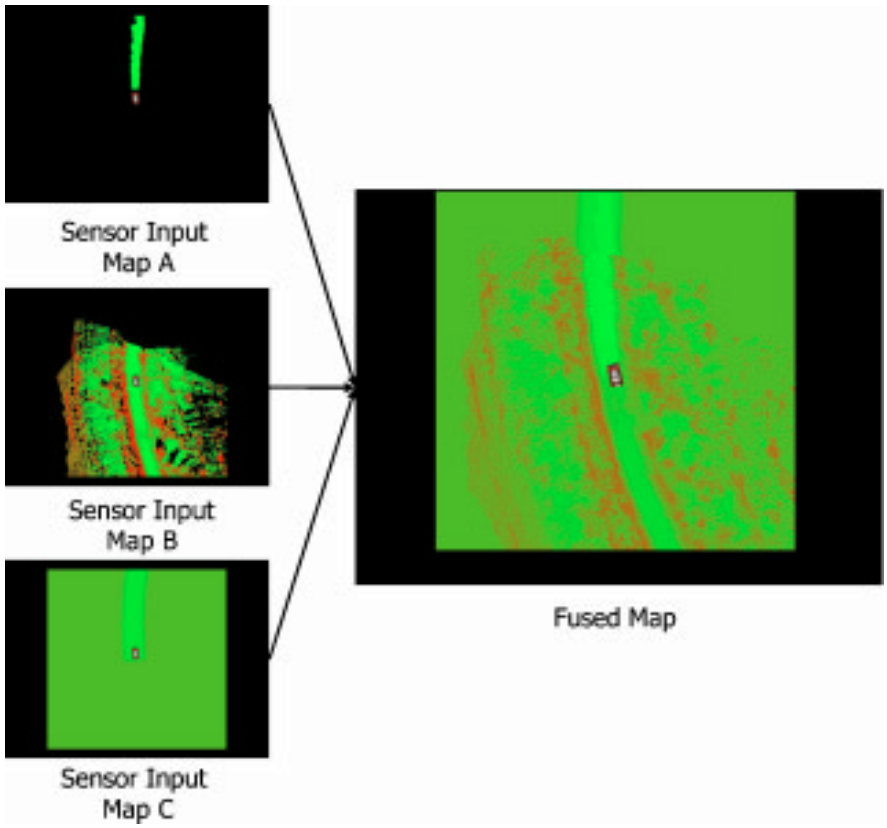


Figura 2.16: Fusión Sensorial y mapa consolidado de Sandstorm y H1lander.

2.3 Modelamiento del Camino

Luego de investigar, se notó que el utilizar modelos paramétricos de caminos se utiliza en general para vehículos navegando por calzadas, tal como se muestra en la Figura 2.17, obtenida de (Tapia-Espinoza & Torres-Torriti, 2013). Se optó por no usar ningún modelo de camino, sino que utilizar el mapa de costos local para representar zonas navegables y no navegables, ejemplos de esto se muestran en la sección anterior. De esta manera no es necesario tener un modelo del camino.

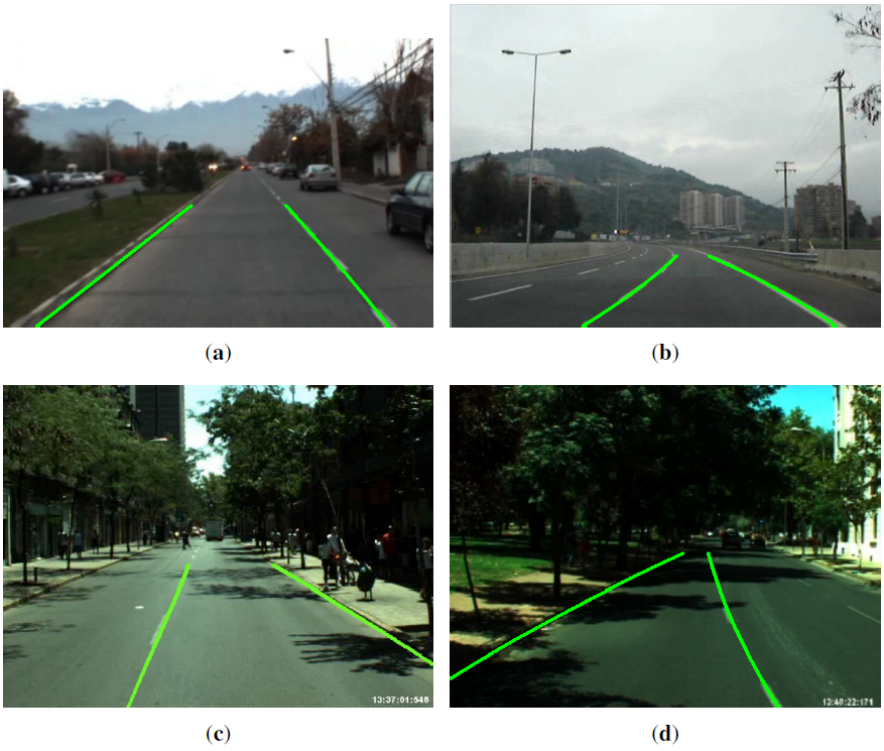


Figura 2.17: a.

2.4 Sensores Utilizados

En los diversos sistemas analizados se observa que los sensores mas utilizados y mas confiables son los LIDAR, utilizados en todos los vehículos autónomos analizados. Luego las cámaras son un sensor muy utilizado, pero la verdad en la gran mayoría de los sistemas este sensor es solo un instrumento secundario, de apoyo, como el caso de Stanley, en el cual las imágenes solo se utilizan para limitar la velocidad máxima en caso de no encontrar camino. Existen algunos casos en los que las imágenes tienen un rol mas importante (Braid, Broggi, & Schmiedel, 2006), pero utilizan visión estéreo con mas de una cámara, tal como se muestran en la Figura 2.18. También existe alguna utilización de RADAR, pero es claramente minoritaria. Para el propósito de esta tesis se utilizarán varios LIDAR y una cámara ya que estos son los sensores instalados en el vehículo del proyecto.

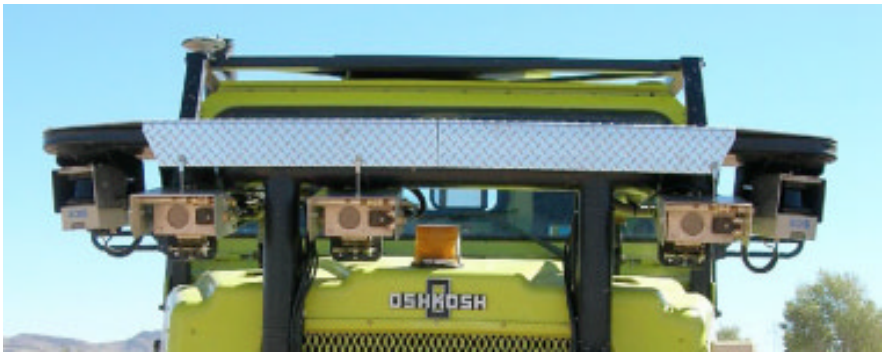


Figura 2.18: a.

Capítulo 3

Sistema Propuesto para Modelamiento del Entorno

3.1 Sistema desarrollado

Tomando en cuenta el estudio del estado del arte, se diseño el sistema de estimación del mapa local para un vehículo autónomo. Este se describe detalladamente en las siguientes secciones, junto con todas las variantes que se probaron. Luego se muestran y describen los resultados de los experimentos para todas las variantes.

3.1.1 Definición de entradas y salidas del sistema

Se comenzará analizando las entradas y salida del sistema. La Figura 3.1 muestra todos los bloques del Diagrama General del Proyecto Vehículo Autónomo (Figura 1.1) que tienen comunicación directa con el bloque *Estimador Mapa Local*, el cual representa el sistema desarrollado.

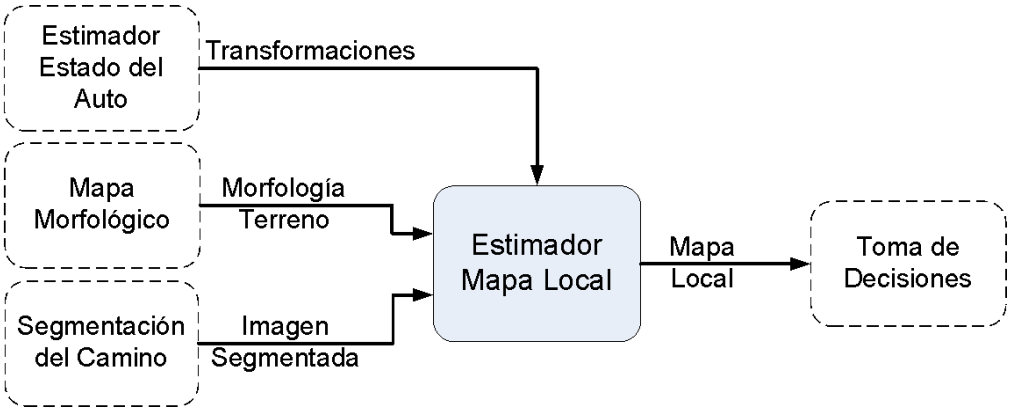


Figura 3.1: Entradas y salida del sistema desarrollado

Las entradas y salida al sistema son las siguientes:

- *Morfología del Terreno* - Describe el entorno del terreno, dando información detallada de la elevación en función de las coordenadas x e y relativas al vehículo. En este caso se representa por una matriz, en donde filas y columnas representan las coordenadas X e Y respectivamente. Cada valor en la matriz representa la elevación en dichas coordenadas, medida por los sensores LIDAR. Este mapa de elevación representa un área de 40x40 metros centrada en el vehículo. Es importante destacar que se reserva un valor especial para representar elevaciones desconocidas. Cada celda de la matriz representa un área de 20x20 centímetros, un ejemplo se puede ver en la Figura 3.2.

Figura 3.2: Ejemplo de la Morfología del Terreno.

- *Imagen Segmentada* - Es una imagen de 640x480 pixeles, en donde cada uno de estos se encuentra clasificado como clase *"Camino"* o *"No-Camino"*. Esta imagen es el resultado de un módulo desarrollado por Fernando Bernuy, parte de su trabajo se muestra en (Bernuy, Ruiz del Solar, Parra, & Vallejos, 2011). Este procesa una imagen con diversas técnicas de filtrado y segmentación. La cámara que obtiene las imágenes originales, se encuentra instalada sobre el techo del vehículo y esta orientada de tal manera que en el borde inferior de la imagen no se alcanza a ver el capo del vehículo. La imagen tiene el área en frente del vehículo centrada, un ejemplo de la imagen original y la segmentación se muestra en la Figura 3.3.

(a) Imagen Original **(b)** *Imagen Segmentada*

Figura 3.3: Se muestra la imagen original (a) y la *Imagen Segmentada* (b).

- *Transformaciones* - El vector de estados del auto comprende a una agrupación de variables, en este caso las variables que se utilizan son las transformaciones entre un sistema de referencia fijo origen y los sistemas de referencias del vehículo y la cámara, ambas pueden ser obtenidas en cualquier momento que se necesite. Estas transformaciones son necesarias para poder integrar la información de la *Morfología del Terreno* y la *Imagen Segmentada*.

- *Mapa Local* - La salida del sistema es, tal como lo indica su nombre, un mapa local. Este mapa guarda información en una estructura de matriz y al igual que la *Morfología del Terreno* filas y columnas representan las coordenadas X e Y respectivamente. El mapa representa un área de 80x80 metros centrados en el vehículo y cada celda representa un área de 20x20 centímetros. El valor de cada celda representa la transitabilidad del terreno, donde 255 se define como *Desconocido*, 220 *Obstáculo*, 100 *Camino no Verificado*, 50 *Camino Disparejo* y 0 *Camino Libre*. En la Figura 3.4 se ejemplifica esto.

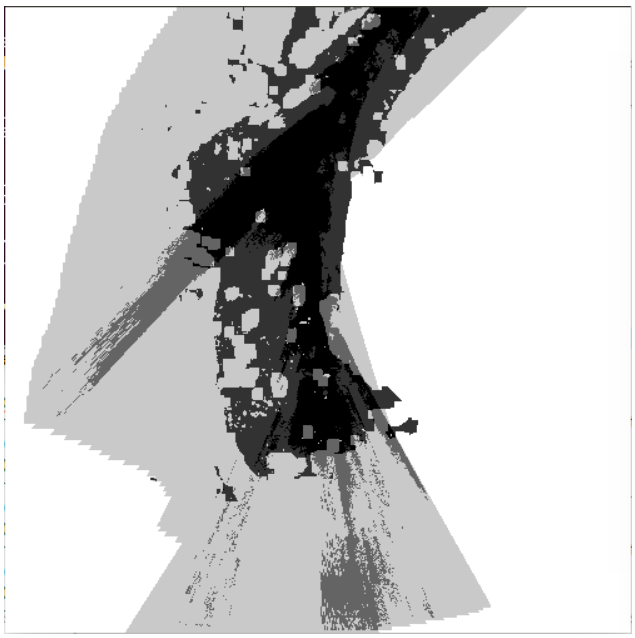


Figura 3.4: Ejemplo del Mapa Local.

3.1.2 Comunicación Mediante ROS

Para lograr la integración con el proyecto *Vehículo Autónomo* es necesario, dentro de muchas otras cosas, usar ROS, ya que este es el sistema que utilizan los módulos contiguos, mostrados en la Figura 1.1. En esta sub-sección no se darán detalles del funcionamiento de ROS, pero se expondrá lo básico y necesario para entender su funcionamiento.

ROS es un meta-sistema operativo para robots, es decir que provee una capa estructurada de comunicación sobre el sistema operativo. ROS proporciona la capacidad pasar mensajes entre procesos y de abstraerse del hardware, pero también permite el control de dispositivos de bajo nivel. Gracias a que es un sistema con código abierto, muchas funcionalidades comunes ya se encuentran disponibles. También provee herramientas y librerías para obtener, construir, escribir y ejecutar código en múltiples computadores.

Los conceptos fundamentales de ROS, junto con los objetivos de diseño de éste, pueden encontrarse en (Quigley et al., 2009). De todas las utilidades que provee ROS, solo se utilizará una pequeña porción. Los conceptos básicos de dichas funcionalidades se explican brevemente a continuación:

- *Nodos* - Son procesos que ejecutan algún tipo cómputo, puede ser tan simple como una suma u otra operación matemática o tan compleja como el reconocimiento de rostros en una multitud. ROS esta diseñado para ser modular, por lo que un sistema esta típicamente compuesto por diversos nodos. En este contexto el termino nodo es sinónimo de módulo.

- *Mensaje* - Es una estructura de datos estrictamente definida, que se utiliza como medio de comunicación entre nodos. Tipos datos estándar son soportados, al igual que arreglos de estos. Un mensaje puede estar compuesto por otros mensajes o arreglos de estos, anidados con profundidades arbitrarias.

- *Tópico* - Son simplemente una cadena de caracteres usada para representar y referirse a cierta información. Los nodos envían mensajes publicándolos en tópicos. Cualquier nodo, si tienen interés en cierta información, pueden subscribirse a tópicos y recibir mensajes publicados en estos. Para cada tópico pueden existir múltiples publicadores y subscriptores.

- *Transformaciones* - En ROS, realizar el seguimiento de la relación espacial entre diferentes sistemas esta implementado mediante el sistema *tf*. Este construye un árbol de transformación dinámico, que relaciona todos los sistemas de referencia del robot. Este árbol se actualiza a medida que recibe información de diversos sub-sitemas.

3.1.3 Módulos Comunes del Sistema

A continuación se describen detalladamente todos los módulos comunes que componen el *Estimador del Mapa Local*, sin incluir los bloques específicos de las variantes que se probaran en los experimentos. El diagrama de bloques detallado se muestra a continuación, en la Figura 3.5.

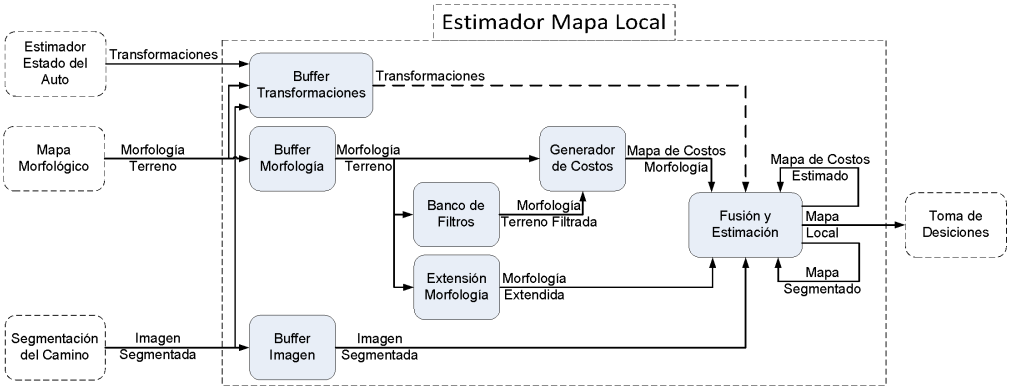


Figura 3.5: Diagrama de Bloques del Sistema Modelamiento del Entorno

Antes de proceder a la descripción detallada de cada módulo, se dará una descripción general del funcionamiento del sistema.

Como primer paso, el sistema guarda la información de los datos de entrada a medida que estas llegan, de forma asíncrona. Esta información se guarda en los nodos de tipo *Buffer*. Adicionalmente cada vez que llega un dato nuevo, se guarda el tiempo en el cual el dato fue tomado, lo que será de vital importancia al momento de relacionar espacialmente los diferentes datos.A grandes rasgos, el resto del sistema realiza 3 funciones: Estimar el *Mapa de Costos* (llamado *Mapa de Costos Estimado*), estimar el *Mapa Segmentado* y realiza una fusión de estos mapas y generar el *Mapa Local*. Todas estas etapas se ejecutan de manera síncrona cada con cada dato de *Morfología del Terreno*. La parte final de estas 3 etapas son realizadas en el bloque de *Fusión y estimación*, pero toda la información requerida para estas operaciones son generadas en los nodos previos. A continuación se describirá de manera general los pasos que sigue la información para lograr generar las 3 estructuras de datos mencionadas anteriormente:

- *Mapa de Costos Estimado* - Primero en el bloque *Banco de Filtros*, se aplican diversos filtros sobre la *Morfología del Terreno* para caracterizar cada celda. Luego en el *Generador de Costos* se utiliza dicha caracterización para definir la dificultad de atravesar cada celda, información guardada en el *Mapa de Costos*. Finalmente, en el bloque *Fusión y estimación* se agrega la información del nuevo *Mapa de Costos* a la estimación histórica, el *Mapa de Costos Estimado*.

- *Mapa Segmentado* - Se comienza por estimar la altura respecto del vehículo de las celdas desconocidas, o más allá del rango, de la *Morfología del Terreno*. Esto es necesario ya que en el bloque *Fusión y estimación* se llevará a cabo la fusión entre la nueva *Imagen Segmentada* con el *Mapa Segmentado*. Para esto se requiere de la relación espacial entre los sistemas de referencia de ambos datos, pero además, es necesario realizar una proyección, ya que la *Imagen Segmentada* esta en un espacio plano, mientras que las celdas de los mapas tienen una representación en 3 dimensiones. Es vital poder proyectar fuera del rango del *Mapa Segmentado*, ya que de otra manera no se utilizaría una gran parte de la información de las imágenes.
- *Mapa Local* - Finalmente se realiza la fusión del *Mapa de Costos Estimado* con el *Mapa Segmentado*, esta se lleva a cabo con unas simples reglas, las cuales serán detalladas más adelante.

Los bloques que se detallarán en esta sección son los siguientes: *Buffer de Morfología*, *Buffer de Imagen* y *Buffer de Transformaciones*.

Buffer de Morfología

Cuando llega un nuevo mensaje que tiene la información de la *Morfología del Terreno*, se guarda esta en una variable local de tipo *grid_map_2d*, una clase implementada especialmente para este sistema, detalles de esta clase se incluirán en el anexo. Esta clase se diseñó para ser compatible con OpenCV, la librería de imágenes que se utilizará en algunos bloques posteriores para filtrar de diversas maneras la morfología.

Buffer de Imagen

Cuando llega un nuevo mensaje que tiene la información de la *Imagen Segmentada*, se guarda esta en una variable local de una clase de imágenes de ROS, compatible con OpenCV.

Buffer de Transformaciones

Antes de comenzar con la descripción del bloque, se mostrarán los sistemas de referencias (denominados frames) utilizados. En la Figura 3.6 se muestra un muy simplificado modelo del vehículo y los 3 frames utilizados en diversos bloques del *Estimador del Mapa Local*. En esta imagen el vehículo esta viajando hacia la esquina superior izquierda y la linea roja muestra la trayectoria que

ha recorrido el vehículo. En cada frame de referencia, el eje X se muestra en rojo, Y en verde y Z en azul.

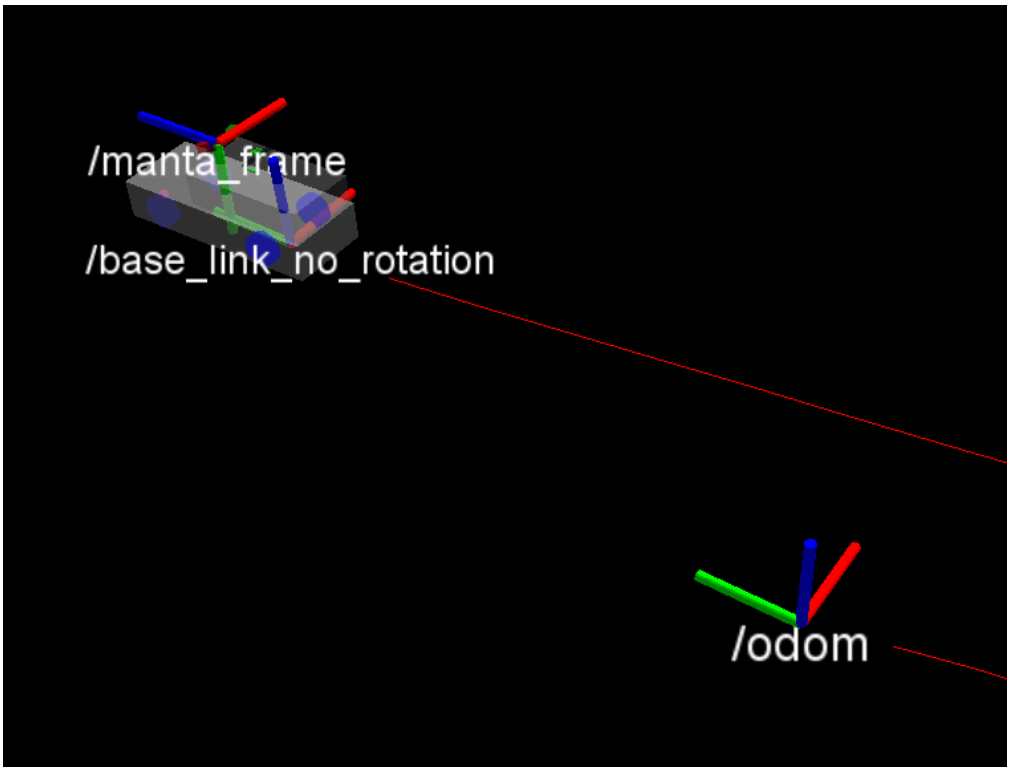


Figura 3.6: Modelo del vehículo mostrando los diferentes sistemas de referencia utilizados por el *Estimador del Mapa Local*

A continuación se describe cada sistema de referencia:

- *odom:* Este sistema esta fijo, todos los demás sistemas están referenciados a él.
- *base_link_no_rotation:* Este sistema es solo una traslación respecto del sistema odom, tal como se observa en la imagen. El punto (0,0,0) de este sistema esta fijo al centro del eje trasero del vehículo. Tal como se menciono antes la orientación del sistema es fija, por lo tanto el vehículo rota en torno al eje Z de este.
- *manta_frame:* Este es el sistema de referencia de la cámara de la cual provienen las imágenes originales que entran al *Estimador del Mapa Local* luego de ser segmentadas por la *Segmentación del Camino*. Tal como se ve en la Figura 3.6, el eje Z de este sistema apunta en la misma dirección de la cámara y del vehículo.

Cada una de las entradas al sistema (*Morfología del Terreno* y *Imagen Segmentada*) tienen un tiempo asociado. Este representa el momento en el que se obtuvo la medición original a partir de la cual se generó la entrada en cuestión. Por ejemplo, el tiempo asociado a una *Imagen Segmentada* corresponde al tiempo en el que se sacó la foto que se segmentó para obtenerla. De aquí en adelante, a este tiempo se le llamará tiempo original de algún dato, por ejemplo tiempo original de la *Imagen Segmentada*. Esto no se limitará a las entradas, también para otras variables internas del código.

Este bloque se encarga de guardar los tiempos y sistemas de referencia asociados ambas entradas al sistema. En cualquier momento que se necesite una transformación mas adelante, simplemente se utilizará la clase *tf* de ROS para encontrar la transformación deseada. Para obtener una transformación en particular simplemente hay que proporcionarle a una función un tiempo, un sistema de referencia de origen y otro de destino. Para obtener una transformación de un sistema en un tiempo a otro en un tiempo diferente, es necesario usar 2 transformaciones y utilizar como sistema intermedio a *odom* ya que este es un sistema fijo. Por ejemplo, para obtener la transformación del sistema **A** en el tiempo **a** al sistema **B** en el tiempo **b** es necesario lo siguiente:

1.- Pedir la transformación de **A** a *odom* en el tiempo **a**. Esta se llamará *Tranformación 1*.

2.- Pedir la transformación de *odom* a **B** en el tiempo **b**. Esta se llamará *Tranformación 2*.

3.- La transformación deseada, denominada *Transformación 3*, corresponderá a aplicar, en este orden, las transformaciones 1 y 2.

Este procedimiento no se volverá a mencionar mas adelante, simplemente se dirá que se requiere la transformación del sistema **A** en el tiempo **a** al sistema **B** en el tiempo **b**.

Extensión Morfología

Debido al acotado rango de la *Morfología del Terreno* en comparación con el camino visible en las imágenes de una cámara (imagen original de la *Imagen Segmentada*) es de sumo interés el lograr tener información sobre el terreno fuera del rango de la morfología, para lograr proyectar la *Imagen Segmentada* y así lograr usar la información de esta para estimar el *Mapa Local* en regiones donde de otra manera no se tendría información.

Para poder saber si la extensión aporta información útil, se utilizarán 2 versiones de este bloque, una que entregará simplemente la misma morfología de entrada y otra, que estimará el plano del terreno. Para encontrar el plano que mejor logre representar el terreno hay que tener en cuenta que no todas las celdas representan puntos pertenecientes al plano, sino que hay varias celdas, que pertenecen a obstáculos aislados.

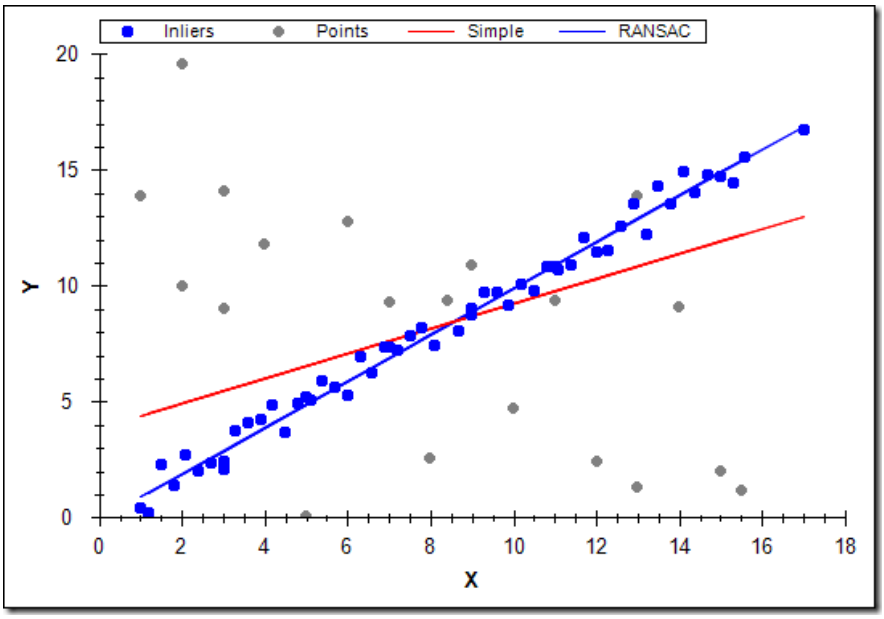


Figura 3.7: Diferencias entre RANSAC y mínimos cuadrados para encontrar una linea dentro de un conjunto de puntos.

Para encontrar el plano, se podría simplemente calcular los mínimos cuadrado de la distancia entre el modelo del plano y la morfología del terreno, pero esto generaría una desviación del modelo, ya que se estaría incluyendo la información de valores atípicos que no debieran ser considerados. En la Figura 3.7 se muestra un ejemplo de datos pertenecientes a una linea (azul), los valores atípicos (gris), modelo estimado utilizando mínimos cuadrados (rojo) y el modelo estimado utilizando RANSAC (linea azul), algoritmo que se explicará en breve. Se observa que debido a la inclusión de datos atípicos, el modelo estimado por mínimos cuadrados no representa bien a los datos deseados.

RANSAC (Fischler & Bolles, 1981) o *Random Sample Consensus*, que en español significa *consenso de muestras aleatorias* es un método que esta diseñado para hacer exactamente lo que se requiere en este caso. Es uno de los métodos iterativos para encontrar los parámetros de un modelo matemático en un conjunto de datos con valores atípicos, tal como se muestra en la Figura 3.7. Lo único que cambia al utilizar diferentes modelos matemáticos (lineas, parábolas, circulos, planos, esferas, etc...), Son las ecuaciones del modelo mismo y la medida de distancia entre dicho modelo con parámetros específicos y un conjunto de puntos cualquiera. Tal como lo indica su nombre, este es un método no determinístico, debido a que las muestras iniciales con las que se arman los modelos son aleatorias, por lo que no es posible asegurar que se encontrará el mejor modelo, ya que es posible que diferentes muestras iniciales lleven a modelos finales diferentes. En la Figura 3.8 se muestra otro ejemplo del uso del algoritmo, esta vez para estimar una parábola.

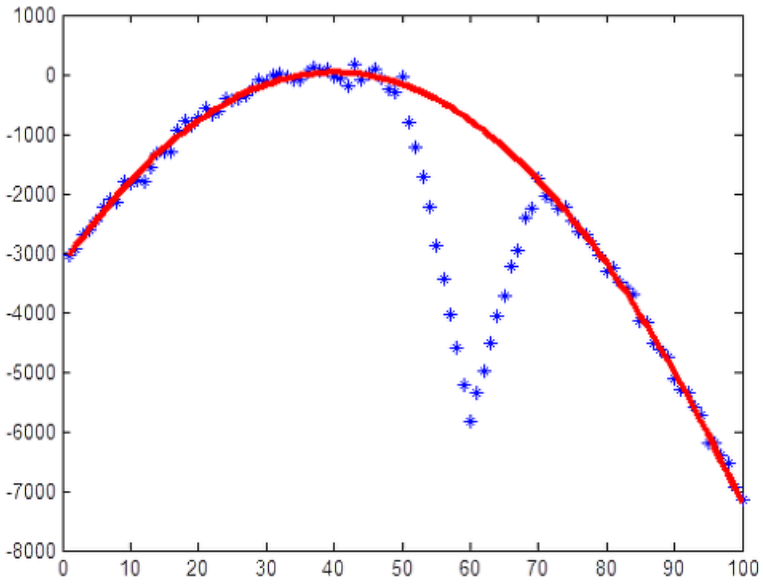


Figura 3.8: Ejemplo de RANSAC utilizado para encontrar el modelo de una parábola.

Se ejemplificarán los pasos básicos necesarios para utilizar el algoritmo RANSAC, utilizando como ejemplo un plano, tal cual como está implementado en este bloque:

1.- Escoger el modelo matemático a utilizar, se eligió la ecuación general de un plano, tal como se muestra en la Ecuación: 3.1.

$$ax + by + cz + d = 0 \tag{3.1}$$

2.- Implementar la función que genere los parámetros del modelo con un mínimo de datos. En este caso el mínimo necesario son 3 y se decidió utilizar mínimos cuadrados para encontrar los parámetros del modelo. Como ya se tiene cierta información de la orientación del plano, orientado similar a los ejes x e y, un conjunto de puntos generará una ecuación del siguiente

tipo:

$$X\beta = Z$$

$$\text{Donde, } X = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix}, \beta = \begin{bmatrix} -a/c \\ -b/c \\ -d/c \end{bmatrix}, Z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}. \tag{3.2}$$

A partir de la Ecuación 3.2, que no tiene solución general para más de 3 puntos linealmente independientes, se utiliza mínimos cuadrados para estimar los parámetros "β", tal como lo muestra la Ecuación 3.3.

$$\hat{\beta} = (X^T X)^{-1} X^T Z. \tag{3.3}$$

3.- Ahora falta implementar una función que calcule la distancia de modelo particular a un punto cualquiera $X_0 = (x_0, y_0, z_0)$. Para lograr esto, se parte de la ecuación general del plano, mostrada anteriormente (ec: 3.1), y se toma el vector normal al plano:

$$n = \begin{pmatrix} a \\ b \\ c \end{pmatrix}, \tag{3.4}$$

luego la Ecuación de un vector desde el plano al punto en cuestión:

$$\delta = - \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix}. \tag{3.5}$$

Ahora se procede a proyectar "δ" sobre "n", lo que da como resultado la distancia del punto al plano:

$$
\begin{aligned}
D &= |proj_n \delta| && (3.6)\\
&= \frac{|n \cdot \delta|}{|n|} && (3.7)\\
&= \frac{|a(x - x_0) + b(y - y_0) + c(z - z_0)|}{\sqrt{a^2 + b^2 + c^2}} && (3.8)\\
&= \frac{|ax + by + cz - ax_0 - by_0 - cz_0|}{\sqrt{a^2 + b^2 + c^2}} && (3.9)\\
&= \frac{|-d - ax_0 - by_0 - cz_0|}{\sqrt{a^2 + b^2 + c^2}} && (3.10)\\
D &= \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}}. && (3.11)
\end{aligned}
$$

Se utiliza la Ecuación 3.11 para encontrar la distancia de un punto arbitrario a un modélo específico.

Con esto se tienen los requisitos para el algoritmo de RANSAC. A continuación se describirán los pasos que sigue el algoritmo:

1.- Se inicializan los parámetros del modelo con dátos seleccionados, ya que se tiene algo de información de la orientación del plano, se escogieron 4 celdas cercanas a los vértices del mapa, verificando que tengan valores medidos y escogiendo otra en la vecindad en caso contrario.

2.- Se mide la distancia de cada dato al modelo recién calculado y se definen como datos válidos los que estén a menos de cierta distancia. Esta distancia depende del problema y es sintonizada al momento de ajustar los parámetros del sistema.

3.- A partir de los datos validados en el paso anterior, se calcula un nuevo modelo.

4.- Si se cumplen con los requisitos de mínimo número de datos y del error del modelo, se finaliza el algoritmo entregando el modelo. En caso contrario, se vuelve al paso 2. Si la cantidad de iteraciones supera cierto número (sintonizado al ajustar parámetros) se entrega el último modelo calculado.

Fusión y Estimación

Este bloque calcula simultáneamente el *Mapa de Costos Estimado* y el *Mapa Segmentado*. Luego con estas 2 estimaciones se calcula el *Mapa Local*. Cabe destacar que para lograr una estimación consistente hay que tomar en cuenta el movimiento del vehículo, esto quiere decir que al comparar

un nuevo mapa de entrada con la estimación anterior, estos se encontraran en posiciones diferentes respecto del vehículo. Debido a esto la estimación hecha en este bloque corresponde a fusionar las regiones de traslape y rellenar la información nueva en donde no se tenía información. También importante notar que este bloque se ejecuta cada vez que llega un nuevo *Mapas de Costos de la Morfología* y que ambos mapas estimados de salida estarán centrados en la misma pose que el *Mapas de Costos de la Morfología*. A continuación se procederá a describir las 3 funciones principales de este bloque.

- **Mapa de Costos Estimado**

  Como primer paso se define que el sistema de referencia del *Mapa de Costos Estimado* de salida tiene la misma pose que el sistema de referencia del *Mapa de Costos de la Morfología* entrante, por lo que en realidad ambos sistemas son el mismo. Luego se necesita la transformación entre el *Mapa de Costos Estimado* anterior y el
  textitMapa de Costos Estimado de salida. Ambas variables tienen el mismo sistema de referencia, pero cada uno tiene un tiempo original diferente. Por lo tanto es necesario encontrar la transformación utilizando el método explicado en la sección 3.1.3. Es necesaria la transformación desde *base_link_no_rotation* en el tiempo original del *Mapa de Costos Estimado* anterior a *base_link_no_rotation* en el tiempo original del *Mapa de Costos Estimado* de salida.

  Ahora que se tiene la transformación necesaria, simplemente se recorre *Mapa de Costos Estimado* de salida. Mediante la transformación descrita anteriormente, con las coordenadas correspondientes a la celda actual, se calculan las coordenadas correspondientes a la celda del *Mapa de Costos Estimado* anterior. Las coordenadas de la celda correspondiente al *Mapa de Costos de la Morfología* son las mismas que las del *Mapa de Costos Estimado* de salida, ya que tienen el mismo sistema de referencia, solo hay que tener cuidado con no salirse del rango, ya que el *Mapa de Costos de la Morfología* es más pequeño. Cos esto se tienen las coordenadas de las celdas correspondientes a las 2 entradas y a la salida, solo resta seguir las reglas de en la Tabla 3.1, la que muestra la salida para cada combinación de entradas.

Tabla 3.1: Reglas para la estimación del nuevo *Mapa de Costos Estimado.*

Nuevo costo \ Estimación anterior	Desconocida	Conocida
Desconocido	Desconocido	Solo anterior
Conocido	Solo nuevo	Promedio ponderado

A la clase *Desconocido* del *Mapa de Costos Estimados* se le asocia un valor "1", al igual que al *Mapa de Costos*.

- **Mapa Segmentado**

Las transformaciones necesarias son en parte las mismas que las utilizadas para calcular el *Mapa de Costos Estimado*, ya que el sistema de referencia del *Mapa Segmentado* de salida es el mismo que el del *Mapa de Costos Estimado* de salida. Sumado a eso se necesita la transformación desde el sistema de la *Imagen Segmentada* anterior en su tiempo original al sistema del *Imagen Segmentada* de salida en su tiempo original.

Esta ultima transformación es necesaria para obtener los puntos en el sistema de la cámara y luego poder proyectarlos y encontrar los índices de los pixeles en los que estos caen. Para esto se utiliza una aproximación esférica, en donde se asume que el sensor de la cámara es una sección de una esfera, lo que produce una pequeña deformación en las esquinas. Si bien esta aproximación no es la mejor, facilita bastante el cálculo. Si se deseara cambiar a una proyección más precisa en futuras aplicaciones, es posible realizarlo cambiando sólo 2 funciones en el código.

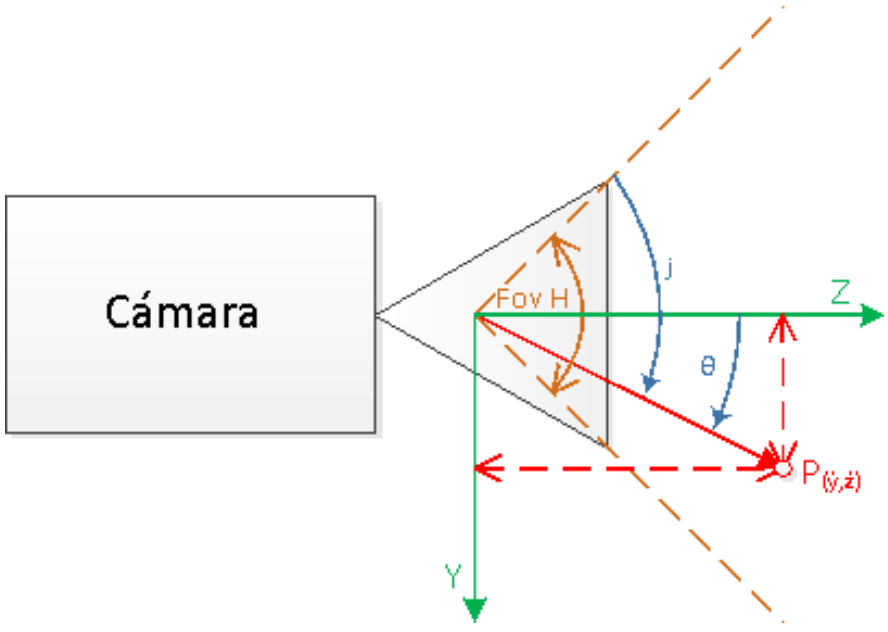


Figura 3.9: Representación 2d de la cámara, su sistema de referencia y el índice j que se busca.

En la Figura 3.9 se muestra una representación bidimensional de la cámara, su sistema de

referencia y el angulo de visión de esta. Es importante destacar que la posición del sistema de referencia, así como las dimensiones y todas las otras variables mostradas en la figura no se encuentran a escala, ni en las posiciones precisas dentro de la cámara, pero están dispuestos de tal manera que facilite su explicación. A continuación se listan las variables mostradas en la figura y las que serán usadas en las ecuaciones que seguirán:

- Y: Eje Y del sistema de referencia de la cámara, apunta hacia abajo de esta misma.
- Z: Eje Z del sistema de referencia de la cámara, apunta hacia adelante de esta misma.
- Fov_H: "Horizontal field of view" o ángulo de visión horizontal, parte de los parámetros de la cámara.
- P: Punto de coordenadas $(\hat{x}, \hat{y}, \hat{z})$ que se desea proyectar a la imagen para encontrar el índice correspondiente. En la Figura 3.9 solo se muestran las coordenadas correspondientes.
- θ: Variable auxiliar que representa el ángulo, en el plano YZ, ente el eje Z y el punto P.
- j: Índice que indica en que pixel de la imagen se encuentra el punto P. La imagen dimensiones conocidas, en este caso es de 640*480 pixeles, por lo que el indice j toma valores entre 0 y 479.

$$\theta = atan(\frac{\hat{y}}{\hat{z}}) \tag{3.12}$$

$$j = floor((Fov_H/2 + \theta) * \frac{480 - 1}{Fov_H}) \tag{3.13}$$

Tal como se muestra en la Ecuación 3.13, se puede obtener el índice j en función de Fov_H, $\hat{y}$, $\hat{z}$ y el número de filas (480). Análogamente, haciendo el mismo análisis para i se obtiene que:

$$\psi = atan(\frac{\hat{x}}{\hat{z}}) \tag{3.14}$$

$$i = floor((Fov_V/2 + \psi) * \frac{640 - 1}{Fov_V}) \tag{3.15}$$

Con las Ecuaciones 3.15 y 3.13 pueden encontrarse los índices (i, j) de cualquier punto en el sistema de referencia de la cámara. Es importante notar que si i o j dan como resultado números menores a 0 o superiores a sus límites 639 y 479 respectivamente, entonces el punto que se intenta proyectar no aparece en la imagen.

Ahora que ya se tienen todas las transformaciones necesarias, se recorre el nuevo *Mapa Segmentado*, nuevamente se transforman las coordenadas de la celda actual para encontrar la

celda correspondiente del *Mapa Segmentado Anterior* y el pixel de la *Imagen Segmentada*. Con esto se tiene acceso a ambas entradas y a la salida, la cual toma valores de acuerdo a la tabla 3.2, que muestra las reglas que sigue el algoritmo implementado.

Tabla 3.2: Reglas para la estimación del nuevo *Mapa Segmentado*, $\alpha = 0.7$.

Imagen Segmentada \ Estimación anterior	Desconocida	Conocida
Desconocido	Desconocido	Solo anterior
No Camino	*No Camino*	*No Camino* $\alpha + (1 - \alpha)$ Anterior
Camino	Camino No Verificado	*Camino* $\alpha + (1 - \alpha)$ Anterior

Para el algoritmo representado en al Tabla 3.2, se definieron los siguientes valores:

- *Desconocido* = 255
- *No Camino* = 200
- *Camino No Verificado* = 100
- *Camino* = 0
- $\alpha = 0.7$

El algoritmo descrito, tiene el efecto de llevar una estimación de la confiabilidad de las clases *Camino* y *No Camino*. VSalores cercanos a 0 implican que la gran mayoría de las mediciones han votado por la clase *Camino*, mientras que valores cercanos a 200 implican que la mayoría de las muestras han votado por la clase *No Camino*. Una desición importante que se tomó, fue asignarle un valor de baja confiabilidad (100) a las celdas en donde la estimación anterior tenga un valor *Desconocido* y la *Imagen Segmentada* dice *Camino*. En cambio, en el caso Desconocido/*No Camino* la salida es simplemente *No Camino*. Esta asimetría provoca que falsas detecciones de camino sean menos probables y al mismo tiempo hace que detecciones iniciales de *No Camino* sean más difíciles de cambiar a *Camino*.

- **Mapa Local**

Ambas entradas contienen valores en sus celdas que representan el costo de transitar por esta. En la Tabla 3.3 se muestran las definiciones de las clases de las entradas:

- *Mapa de Costos-Desconocido*: 1, tal como se mencionó anteriormente.
- *Mapa de Costos-Obstáculo*: todos los valores entre 0.5 y 1.
- *Mapa de Costos-Libre*: todos los valores mayores o iguales a 0 y menores a 0.5.
- *Mapa Segmentado-Desconocido*: 255 como ya se ha mencionado previamente.
- *Mapa Segmentado-No Camino*:Valores entre 60 y 200. Para evitar falsas detecciones de camino, se asume como *No camino* los valores de baja confiabilidad también.
- *Mapa Segmentado-Camino*:Valores entre 0 y 60 inclusive.

En este caso no se necesitan transformaciones, debido a que el *Mapa Local* se calcula directamente del *Mapa de Costos Estimado* y del *Mapa Segmentado*, por los tanto los sistemas de referencia son los mismos. Nuevamente se recorren las celdas de la salida, el *Mapa Local* en este caso. Las celdas correspondientes del las entradas ya mencionadas tienen los mismos indices que la del *Mapa Local*. La Tabla 3.3 muestra las reglas para calcular el *Mapa Local*.

Tabla 3.3: Reglas para el cálculo del *Mapa Local*.

Mapa Segmentado \ Mapa de Costos	Desconocido	Obstáculo	Libre
Desconocido	Desconocido	Obstáculo	Camino Disparejo
No Camino	Obstáculo	Obstáculo	Camino Disparejo
Camino	Camino No Verificado	Obstáculo	Camino

Para explicar mejor las reglas representadas por la Tabla 3.3, a continuación se describen en forma de pseudocódigo:

```
1: if MapadeCostosEstimado == Obstáculo then
2:     MapaLocal = Obstáculo
3: else if MapadeCostosEstimado == Libre then
4:     if MapaSegmentado == Camino then
5:         MapaLocal = Camino
6:     else
7:         MapaLocal = Camino Disparejo
8:     end if
9: else
10:    if MapaSegmentado == Camino then
```

```
        MapaLocal = Camino No Verificado
    else if MapaSegmentado == No Camino then
        MapaLocal = Obstáculo
    else
        MapaLocal = Desconocido
    end if
end if
```

Es importante destacar que una de las variantes probadas en los experimentos consiste en no utilizar las imágenes para la estimación, lo que sería representado por un valor *Desconocido* para cada celda del *Mapa Segmentado*. En este caso la salida sería calculada exclusivamente del *Mapa de Costos*, tal como lo muestra la Tabla 3.4.

Tabla 3.4: Reglas para el cálculo del *Mapa Local* sin información de las imágenes.

Mapa de Costos	Desconocido	Obstáculo	Libre
Mapa Local	Desconocido	Obstáculo	Camino

Esta solución es puramente académica y se necesita para poder tener una base con la cual comparar el sistema completo para ver como se comporta al agregar la información proveniente de las imágenes.

3.1.4 Módulos del Método de Frecuencias

Banco de Filtros

Este módulo se encarga de filtrar la *Morfología del Terreno* y utilizar los resultados de dichos filtros como características en el bloque *Generador de Costos* para generar el *Mapa de Costos de la Morfología*. Se diseñaron dos soluciones diferentes para este bloque. La primera opción utiliza filtros de diferentes frecuencias de corte, con los cuales se calcula la energía en diferentes bandas de frecuencias para cada celda. Con estos valores se intenta caracterizar el terreno. La segunda variante utiliza una filtro no lineal de diferencias de altura para lograr el mismo objetivo. A continuación se dan los detalles de implementación para la primera variante.

El objetivo de este método es lograr calcular la energía de diferentes bandas de frecuencia para cada celda y luego utilizar estos valores para estimar la navegabilidad de cada celda. Este método esta

inspirado en (Hoffman & Krotkov, 1990).

Para lograr calcular la energía de diferentes bandas de frecuencias, se partirá por definir la densidad espectral. Para ejemplificar se analizará el caso de una *Morfología del Terreno* unidimensional, en donde cada celda esta representado por $i(x)$. En la Ecuación 3.16, $S_{xx}(f)$ corresponde a la densidad espectral mientras que $I(f)$ es la transformada de fourier de $i(x)$.

$$S_{xx}(f) = |I(f)|^2. \tag{3.16}$$

Para estimar $S_{xx}(f)$ para una frecuencia específica f, basta con pasar la señal (*Morfología del Terreno*) por un filtro pasa banda muy angosto centrado en f y luego calcular la energía de la señal resultante de la siguiente manera:

$$S_{xx}(f) \approx 1/\Delta f \int_{-\infty}^{\infty} |I(f) \cdot H(f)|^2 \, \mathrm{d}f. \tag{3.17}$$

Como $1/\Delta f$ es una constante y las densidades espectrales calculadas sólo serán comparados con otras densidades espectrales, podemos eliminarla de la ecuación, obteniendo asi:

$$S_{xx}(f) \approx \int_{-\infty}^{\infty} |I(f) \cdot H(f)|^2 \, \mathrm{d}f. \tag{3.18}$$

El problema es que en la Ecuación 3.18 es necesario realizar el cálculo en el espacio de Fourier, pero se está buscando una aproximación que permita realizar el calculo en el espacio de la imagen. Para esto se utiliza el teorema de Parseval, que se muestra en la Ecuación 3.19.

$$\int_{-\infty}^{\infty} i(x) \, \mathrm{d}x = \int_{-\infty}^{\infty} I(f) \, \mathrm{d}f. \tag{3.19}$$

Con esto, es posible reescribir la Ecuación 3.18 de la siguiente manera:

$$S_{xx}(f) \approx \int_{-\infty}^{\infty} |i(x) * h(x)|^2 \, \mathrm{d}x, \tag{3.20}$$

en donde $*$ corresoponde a la operación de convolución, la cual es utilizada para aplicar filtros. Con la Ecuación (3.20) se puede aproximar la densidad espectral para una frecuencia f específica, pero también se busca caracterizar esta en función de la posición. Para esto, se realizan estas mismas

operaciones pero sobre una ventana móvil $w_{x_0}(x)$. Con esto se obtiene:

$$S_{xx}(f, x_0) \approx \int_{-\infty}^{\infty} |(i(x) \cdot w_{x_0}(x)) * h(x)|^2 \, \mathrm{d}x. \tag{3.21}$$

Ahora lo único que resta por hacer es aproximar la integral de la Ecuación (3.21) por una suma, ya que los datos que tenemos son discretos. Además, no es necesario hacer una suma de $-\infty$ a∞ ya que se utiliza una ventana Gaussiana, por lo que sus valores tienden a cero a medida que se alejan del centro.

$$S_{xx}(f, x_0) \approx \sum_{x \in A} |(i(x) \cdot w_{x_0}(x)) * h(x)|^2 \, \mathrm{d}x. \tag{3.22}$$

Donde A es el conjunto de índices en los que los valores de la suma son relevantes. Para generalizar para 2 dimensiones, solo es necesario reemplazar x_0, un escalar, por X_0, un vector. Claramente también se necesita que A ahora represente un área en 2 dimensiones, $i(X)$ corresponde a la celda de coordenadas X de la imagen, $h(X)$ es el valor del filtro h evaluado en las coordenadas X y $w_{X_0}(X)$ es la ventana móvil centrada en X_0, evaluada en X.

Aplicando la ecuación (3.22) al *Estimador del Mapa Local*, se tiene que todo lo que hay que hacer para calcular la densidad espectral de la celda (x, y) para una frecuencia f_i es lo siguiente:

- Extraer una ventana centrada en (x, y) de la *Morfología del Terreno.*
- Aplicar un filtro pasa banda de frecuencia media f_i a la ventana.
- Elevar al cuadrado cada elemento de la ventana.
- Sumar todos los elementos de la ventana.

Se sigue este algoritmo para 5 bandas de frecuencias. Como ya se impuso la restricción de que los anchos de las bandas (Δf) deben ser iguales, las bandas se definen como:

- *Baja:* Frecuencia de 0.5[1/m].
- *Media Baja:* Frecuencia de 1[1/m].
- *Media:* Frecuencia de 1.5[1/m].
- *Media Alta:* Frecuencia de 2[1/m].
- *Alta:* Frecuencia de 2.5[1/m].

Para llegar a estos valores, se tomó en cuenta que el ancho de las celdas es de 0.2[m], por lo que el menor periodo que es posible detectar es de 0.4[m]. Esto implica que la frecuencia máxima que es posible detectar es de 2.5[1/m]. En el extremo opuesto, se asumió que variaciones de alturas con periodos de más de 2[m] generarían no generarían obstáculos, por lo que la menor frecuencia que se quiere detectar es de 0.5[1/m]. El resto de las frecuencias se obtienen al dividir regularmente el espacio de frecuencias.

El ultimo paso es transformar las frecuencias/periodos que se tienen en los sigma (σ) de los fitros deseados. Los resultados de los filtros pasa banda se calcularan en base a la resta de la salida de 2 filtros pasa bajo, por lo que se necesita calcular 2 σ para cada filtro. Los filtros pasa bajo que conformarán los filtros pasa banda, serán filtros Gaussianos, por lo que se necesita estudiar estos para lograr definir el σ de cada Gaussiana. La Ecuación (3.23) muestra la función de distribución de una Gaussiana unidimensional:

$$f(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{(x-u)^2}{2\sigma^2}}, \tag{3.23}$$

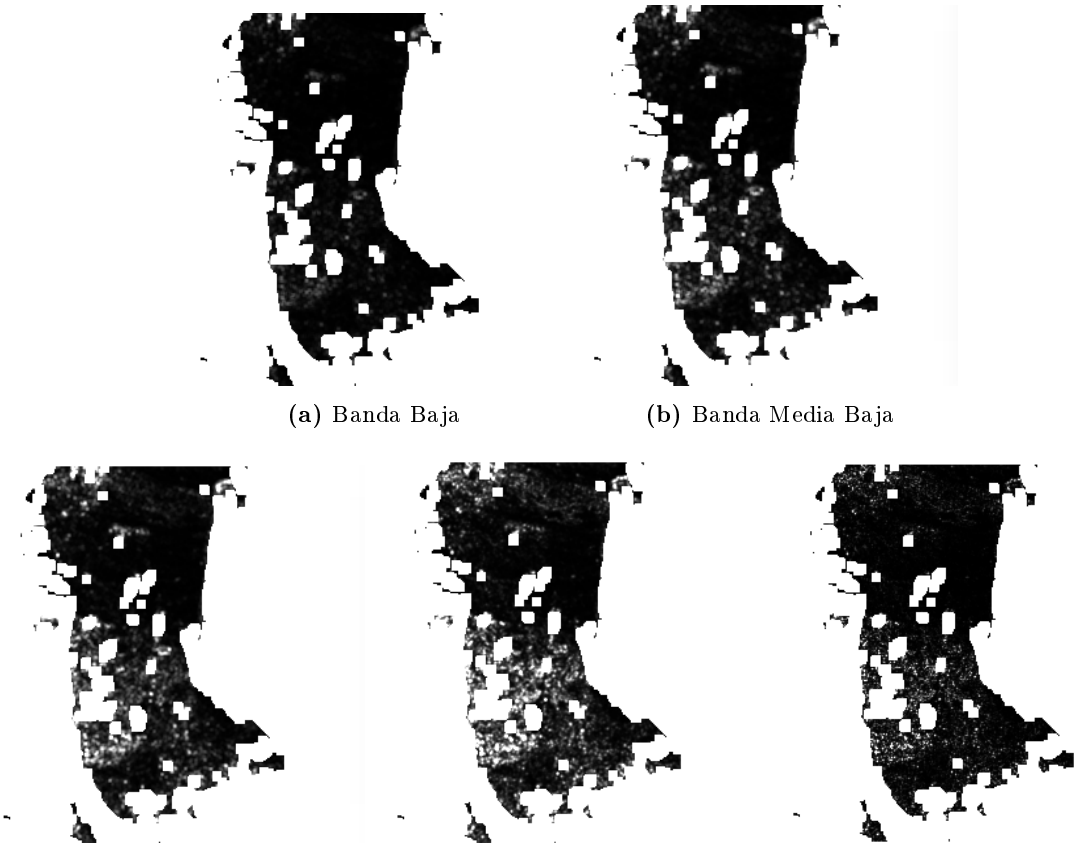
donde σ^2 corresponde a la varianza y u a la media. También es sabido que la frecuencia de corte de una gaussiana esta dada por:

$$f_c = \frac{1}{2\pi\sigma}. \tag{3.24}$$

Con el fin de minimizar la cantidad de computo necesario, se calcularán 6 $\sigma's$ de manera que los filtros medios compartirán los filtros pasa bajo con los vecinos. Con la distribución de las frecuencias centrales de las bandas, es claro que el ancho de las bandas debe ser de 1[1/m], por lo tanto, en la Tabla 3.5 se muestran las frecuencias de corte de cada banda, los periodos y sigmas asociados a estas.

Tabla 3.5: Filtros pasa banda con sus respectivas frecuencias, periodos y sigma.

Banda	f_{inf}[1/m]	f_{sup}[1/m]	T_{inf}[m]	T_{sup}[m]	σ_{inf}[rad/m]	σ_{sup}[rad/m]
Baja	0.25	0.75	4	1.333	0.636	0.212
Media Baja	0.75	1.25	1.333	0.8	0.212	0.127
Media	1.25	1.75	0.8	0.571	0.127	0.09
Media Alta	1.75	2.25	0.571	0.444	0.09	0.07
Alta	2.25	2.75	0.444	0.363	0.07	0.057



(a) Banda Baja **(b)** Banda Media Baja

(c) Banda Media **(d)** Banda Media Alta **(e)** Banda Alta

Figura 3.10: Se muestra un ejemplo de la densidad espectral calculada a partir de la Ecuación 3.22, para cada una de las bandas definidas en la Tabla 3.5 (a-e)

Generador de Costos

Este bloque toma la salida del bloque *Banco de Filtros* y estima en base a esta el costo de transitar por cada celda, generando el *Mapa de Costos de la Morfología*. Este mapa también es guardado en la clase *grid_map_2d*. Se define como camino libre el valor o costo 0 y como obstáculo el costo 0.9. Las celdas también pueden tener cualquiera de los valore intermedios. Se reserva el valor 1 para el valor *Desconocido*, que representa falta de información en la celda correspondientes de alguna de las entradas.

La entrada para este método son 5 *grid_map_2d*, uno para cada una de las 5 bandas de frecuencia. Para estimar el costo de cada celda, se utiliza el valor de la energía espectral de cada una de las bandas frecuencia para esa celda.

Se decidió que una simple combinación lineal de la energía de cada una de las bandas de frecuencia es suficiente para estimar rugosidad del terreno y por consiguiente estimar el costo. Para lograr obtener los pesos iniciales de la combinación lineal se estudiarán los filtros y sus períodos asociados, para intentar definir cuanto influyen a la conducción del vehículo.

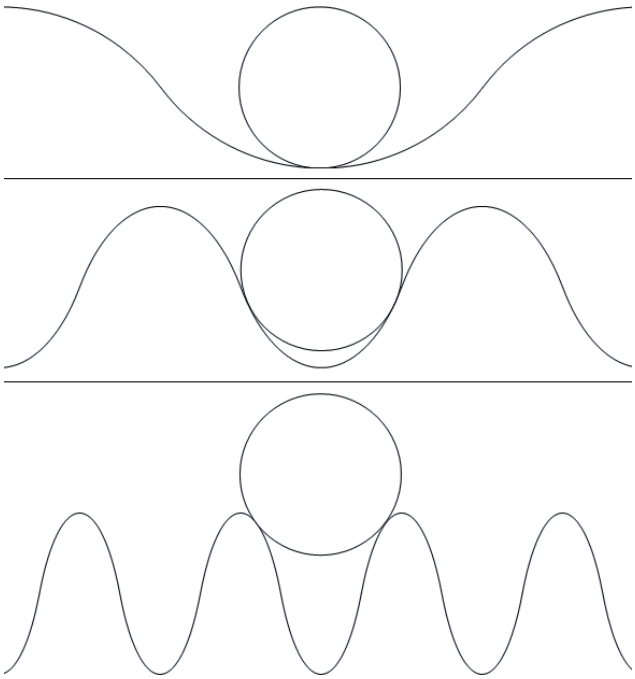


Figura 3.11: Representación de un neumático sobre diferentes terrenos sinusoidales. Arriba: $T = 4 \times Diametro$, medio: $T = 2 \times Diametro$, abajo: $T = 1 \times Diametro$.

Los agujeros o irregularidades de tamaño similar al neumático serán los que mas afecten la conducción del vehículo. Para ejemplificar se dará el ejemplo con terrenos sinusoidales. En la Figura 3.11 se ve una representación de un neumático sobre diferentes terrenos sinusoidales de la misma amplitud pero diferente periodo. Si el neumático pasa sobre un terreno sinuoso que tiene el periodo (Figura 3.11 - abajo) mucho mas pequeño que el diámetro del neumático, este simplemente pasa fácilmente de montículo en montículo, sin que importen los agujeros, el efecto del terreno pasan a ser vibraciones. En el caso contrario, si el periodo es muy grande en comparación al diámetro de los neumáticos (Figura 3.11 - arriba), los cambios en la altura serán demasiado suaves para que afecten la conducción de manera significativa. En cambio si el periodo es comparable a 2 veces el diámetro del neumático (Figura 3.11 - medio) el efecto en la conducción es muy grande, debido a las pendientes abruptas que se generan.

Tabla 3.6: Filtros pasa banda con el rango de periodos correspondientes, periodo medio y relación entre este y el diámetro de los neumáticos, estimado en $0.5[m]$.

Banda	$T_{max}[m] - T_{min}[m]$	T_{medio}	T_{medio}/D
Baja	$4 - 1.333$	2.666	5.333
Media Baja	$1.333 - 0.8$	1.066	2.133
Media	$0.8 - 0.571$	0.685	1.371
Media Alta	$0.571 - 0.444$	0.507	$1,014$
Alta	$0.444 - 0.363$	0.403	0.806

Tal como se menciono en el párrafo anterior, terrenos con el periodo cercano al doble de el diámetro de los neumáticos son las que provocan mayores problemas para la conducción. En la Tabla 3.6 se muestra el cuociente entre el periodo medio y el diámetro del neumático para cada banda. Es importante destacar que la banda con un cuociente más cercano a dos es la banda media baja.

El siguiente paso consta simplemente de realizar una suma ponderada del resultado de cada filtro en cada celda. Los pesos de cada banda deben sintonizarse para cumplir 2 objetivos. Uno ajustar los pesos relativos entre las bandas para que estos reflejen las diferencias de los costos de transitar sobre los diferentes terrenos. Dos, calibrar la ganancia general para que la suma ponderada en las celdas correspondientes a obstáculos en valores cercanos a 0.9. Cabe destacar que esta suma se satura en 0.9, exceptuando las celdas con el valor *desconocido*, las que se les asigna un valor 1.

3.1.5 Módulos del Método de Diferencias de Altura

Banco de Filtros

Este método es mucho más sencillo que su análogo. Se basa en encontrar cambios bruscos de alturas en la vecindad de cada celda. Para lograr esto se define un radio de búsqueda r y se busca la diferencia máxima entre la celda central con las demás celdas dentro del radio definido. Este valor es el que se guarda y se utiliza luego en el bloque *Generador de Costos* para estimar la navegabilidad de esta.

Tras estudiar este método y hacer el paralelo con el método de filtrado, se decidió que no es suficiente hacer este análisis para un solo radio, sino que se decidió calcular la diferencia máxima en anillos de 3 radios diferentes y utilizar estos 3 valores para caracterizar la navegabilidad en bloques siguientes.

Cada anillo define una vecindad A, en la cual se realiza la búsqueda. La Ecuación (3.25) se muestra como se calcula la diferencia de alturas.

$$H_{dif}(X_0) = Max_{X \in A}(|X - X_0|) \tag{3.25}$$

Por motivos prácticos, se decidió definir los anillos en términos de la distancia medida en celdas. En la Figura 3.12 se muestran las 3 vecindades en forma de anillos utilizadas, cada una en un tono de gris diferente, mientras que la posición del vehículo ve marcada.

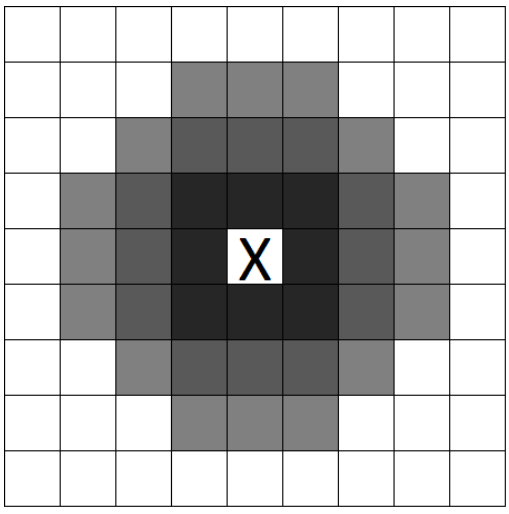
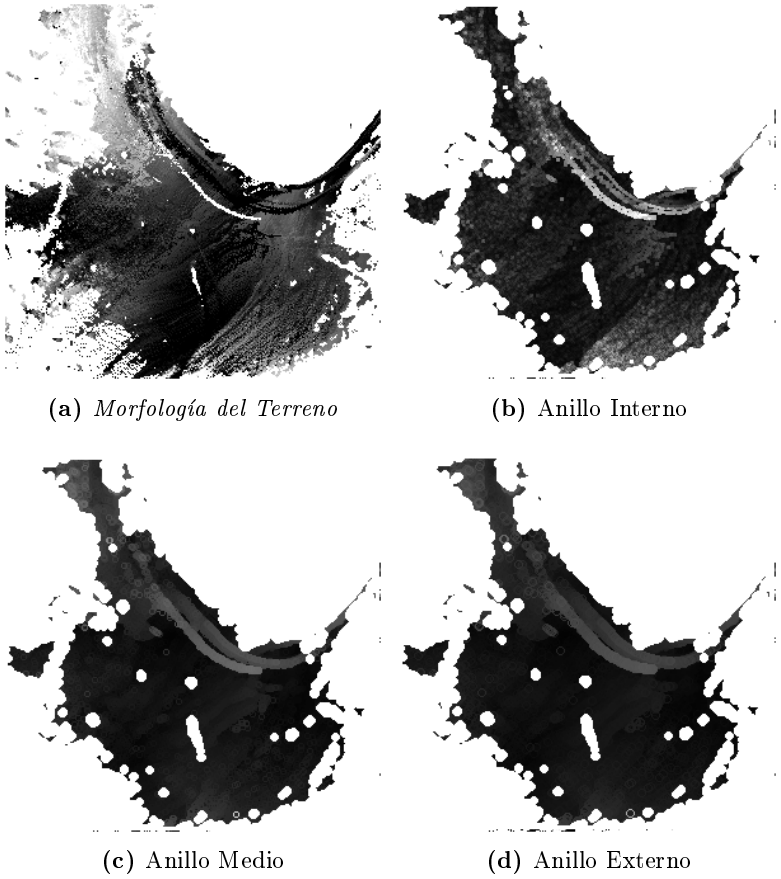


Figura 3.12: Cada uno de los anillos representa una vecindad diferente en la cual se busca la diferencia de altura máxima

En la Tabla 3.7 se detallan los anillos, caldas que los componen y radios de estos. Se espera que las diferencias máximas entre la celda central y cada una de estas 3 vecindades logren caracterizar el terreno en torno a la celda y así poder estimar la navegabilidad de esta.

Tabla 3.7: Anillos de vecindad, número de celdas y distancia promedio a la celda central.

Anillo	N_{celdas}	$D_{promedio}$
Interno	8	0.241
Medio	12	0.431
Externo	16	0.607



(a) *Morfología del Terreno* **(b)** Anillo Interno

(c) Anillo Medio **(d)** Anillo Externo

Figura 3.13: Se muestra un ejemplo de una *Morfología del terreno* (a) y la diferencia de altura, para cada uno de los anillos definidos en la tabla 3.7 (b-d)

Con esto se tienen dos soluciones, las que toman la *Morfología del Terreno*, la filtran de maneras diferentes y entregan 2 conjuntos de *grid_map_2d* que guardan la información de los filtros aplicados

por ambos métodos. Es importante destacar todos los cálculos se hacen en celdas para las que todas las celdas que participarían de la operación son conocidas. Esto provoca que ambos métodos realicen una operación de apertura sobre las celdas con valor desconocido. Se reserva un valor alto, fuera de los rangos posibles del sensor, para identificar las celdas con valores desconocidos.

Generador de Costos

Al igual que en el bloque análogo se estima el costo de transitar por cada celda en base a la salida del *Banco de Filtros*. Se utilizan las mismas definiciones que en el bloque análogo: para el camino libre el costo 0 y como obstáculo el costo 0.9. Las celdas también pueden tener cualquiera de los valore intermedios. Se reserva el valor 1 para el valor *Desconocido*, que representa falta de información en la celda correspondientes de alguna de las entradas.

La entrada para este método son 3 *grid_map_2d*, uno para cada uno de los 3 anillos de vecindad. Para este método se utiliza un enfoque muy similar al del *método de Filtros*, en donde se planea calcular una suma ponderada de los valores de estas 3 entradas para cada celda y así estimar la navegabilidad de esta.

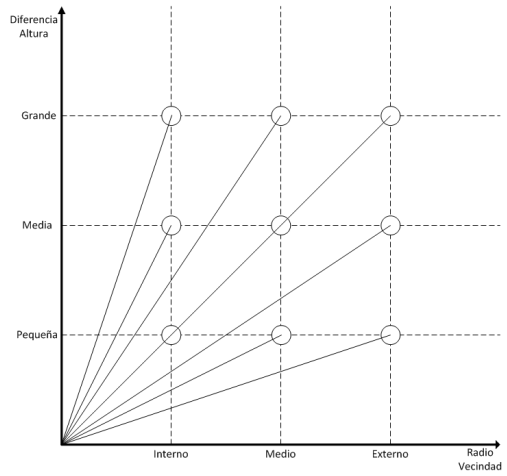


Figura 3.14: Representación de 3 diferencias de altura para cada una de las 3 vecindades y las pendientes que estas generan.

En la Figura 3.14 se observa claramente que para puntos con una misma diferencia de altura, a medida que el radio se hace más pequeño, las pendientes de las rectas entre el origen (celda central) y el punto en cuestión se vuelven más abruptas. Por lo tanto el peso de los anillos mas cercanos debiera ser mayor. Se debe recordar que los pesos utilizados deben cumplir las mismas condiciones

que en el método anterior, los pesos relativos deben reflejar los costos de transitar por terrenos con las características representadas y la ganancia general debe ser tal que la suma ponderada de los 3 valores asociados a los anillos debe ser cercana a 0.9 para las celdas que representan obstáculos. También es importante destacar que los pesos estimados en esta sección son solo primeras aproximaciones y que en la secciones posteriores se encontrarán valores óptimos para estos mediante un entrenamiento.

Capítulo 4

Experimentos, Resultados y Análisis

Para lograr cumplir y verificar los objetivos impuestos en el primer capítulo, es necesario realizar experimentos y comparar los resultados de las diferentes variantes disponibles. Para lograr una buena comparación, también es necesario diseñar e implementar medidas de desempeño que logren estimar de alguna forma la calidad de cada una de las variantes probadas. También se compararán los resultados con el fin de determinar cual es la mejor variante.

4.1 Experimentos

Antes de realizar los experimentos es necesario cumplir con ciertos requisitos. A continuación se describirán estos, de forma quue se logren realizar los experimentos de una forma rápida y que permita sacar información confiable y precisa:

- Es necesario tener una base de datos que permita correr experimentos múltiples veces sin necesidad de hardware adicional o salidas a terreno.
- Obtener o generar otra datos con los que se logre calcular que tan buenas o malas son las estimaciones de los mapas locales. Este punto de comparación se denominará *Ground_Truth* ya que este representa la verdad del terreno.
- Diseñar medidas de desempeño que logren cuantificar y representar visualmente la calidad de las estimaciones de los mapas locales.

4.1.1 Base de datos

Para lograr obtener una base de datos práctica, se utiliza otra de las grandes ventajas de ROS. Se utiliza package (unidad básica de la librería de ROS) disponible en su librería llamado *Rosbag*. Este consiste en un conjunto de herramientas para grabar y reproducir topicos de ROS. Los tópicos necesarios para lograr que el *Estimador Mapa Local* funcione correctamente son los siguientes:

- *z_map_array* - contiene la información de la *Morfología del terreno*, tal como se ha mencionado anteriormente.
- *RoadSegmentation\output* - tiene la información de la *Imagen Segmentada*.
- *tf* - este se maneja internamente por la clase *tf*, por lo que nunca se interactúa directamente con estos mensajes, solo a través de la clase mencionada.
- *Clock* - utilizado por *Rosbag* para simular los tiempos en los que se publico cada mensaje originalmente.
- *borde_camino* - no es usado directamente por el sistema implementado, sinó que es utilizado para generar el *Ground Truth*. En la siguiente subsección se darán mas detalles de esto.

Estos tópicos son suficientes para que el sistema funcione correctamente. por lo que se grabó una base de datos con estos, lo que permitirá su reproducción. Existen diversas opciones a la hora de reproducir un *bag*, desde cambiar la velocidad de reproducción, hasta definir tiempos iniciales y finales para la reproducción. Esta base de datos será dividida en 2 conjuntos *entrenamiento* y *prueba* tomas de datos realizadas con el vehículo.

En la Figura 4.1 se ve el recorridos de la bases de datos grabada y se muestra en diferentes colores la sección de entrenamiento (rojo) y pruebas (azul).

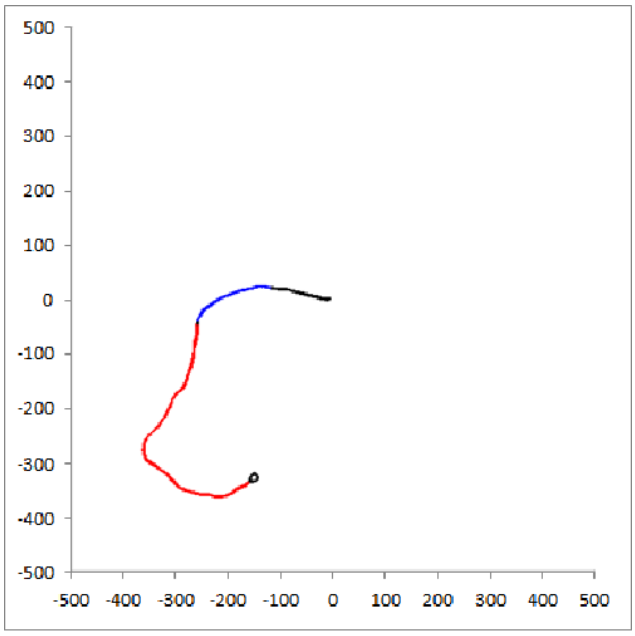


Figura 4.1: Se muestran los recorridos de la base de datos utilizada, se muestra en rojo el conjunto de entrenamiento y en azul el de prueba.

4.1.2 Generación Ground Truth

Como ya se mencionó anteriormente el *Ground Truth* se obtendrá para cada base de datos a partir del tópico *borde_camino*. Este contiene información procesada a partir de mediciones de un sensor LASER ubicado en el parachoques delantero del vehículo y que se encuentra apuntando hacia abajo. Claramente un sensor con esta pose tiene poca utilidad para asegurar una navegación sin fallas y por los mismo esta no se utiliza de ninguna manera en el sistema de estimación del mapa local.

Un sistema previo procesa los escaneos de este sensor y detecta las coordenadas de los puntos en donde comienzan a haber obstáculos a los costados del vehículo. Luego esa información se envía por mensaje de ROS en el tópico ya mencionado. Este mensaje es de tipo *sensor_msgs/PointCloud* lo que quiere decir que es un listado de puntos, en este caso cada mensaje contiene solamente 2 puntos.

Para hacer uso de esta información se diseña el siguiente sistema:

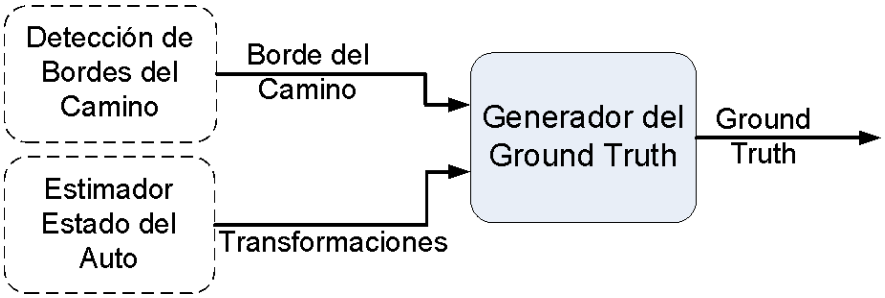


Figura 4.2: Diagrama general del sistema de generación del ground truth implementado.

en donde las entradas y salidas son las siguientes:

- *Borde del Camino* - 2 puntos que representan los límites del terreno navegable hacia la izquierda y derecha del vehículo.
- *Transformaciones* - como ya se ha visto antes, estas viene a través de los *tf* y permiten encontrar transformaciones entre diferentes sistemas de referencia.
- *Ground Truth* - se define como un gran mapa, de tipo *grid_map_2d*, que tiene todas sus celdas marcadas como *Desconocido*, a excepción de los puntos que se definen como *Camino* en el sistema.

El *Generador del Ground Truth* funciona de la siguiente manera:

- Se inicializa este gran mapa local, al que llamaremos *Ground Truth*, de 1x1 km^2, con valor *Desconocido* en todas sus celdas. Al igual que en otros mapas locales, *Desconocido* corresponde al valor "255" y las celdas tienen un tamaño de 20x20 centímetros.
- Cuando llega un nuevo *Borde del Camino*, se calcula la transformación desde el frame *base_link* hacia *odom*. Donde *base_link* es sistema de referencia que tiene el mismo cero que *base_link_no_rotation* solo que su eje X siempre apunta hacia adelante del vehículo, a diferencia del último, que mantiene siempre sus ejes solidarios a los ejes de *odom*.
- Se transforman ambos puntos de borde hacia el sistema de *odom*, el sistema del *Ground Truth*.
- Se recorre la linea definida por ambos puntos de borde y se marcan todas las celdas del *Ground Truth* por las que se pasa como *Camino*, que como ya se ha mencionado se le asocia un valor "0".

- Se guarda el *Ground Truth* como una imagen para ser utilizado posteriormente. solo para visualización, se destaca el *Ground Truth* del conjunto de entrenamiento en rojo y el de pruebas en azul.

En la Figura 4.3 se muestra el *Ground Truth* para ambos conjuntos.

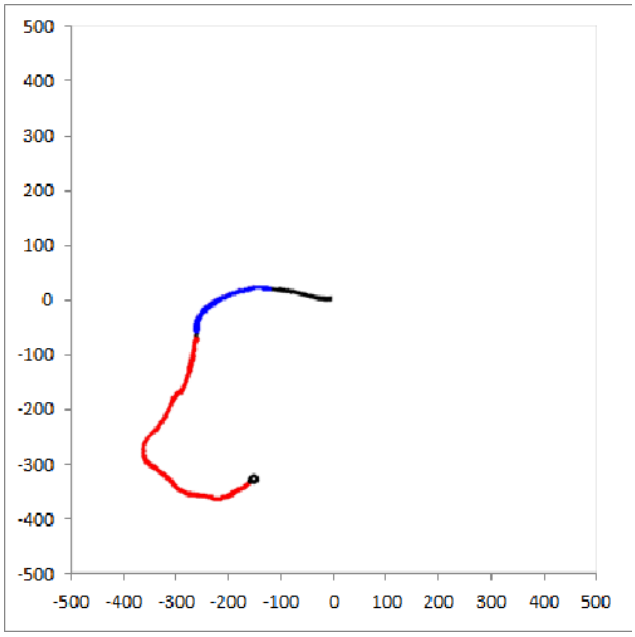


Figura 4.3: Se muestran el Ground Truth de la base de datos utilizada, se muestra en rojo el conjunto de entrenamiento y en azul el de prueba.

4.1.3 Metodología de Evaluación

Es necesario diseñar una medida de desempeño, que permita caracterizar, valga la redundancia, el desempeño de cada una de las variantes del sistema para la base de datos. Para lograr esto hay que volver a los objetivos del sistema, particularmente a: "La salida debe permitir encontrar trayectorias seguras para el vehículo, marcando zonas obstaculizadas, seguras y desconocidas". El sistema a diseñar debe generar trayectorias y cuantificar la seguridad de estas.

Es importante destacar que el sistema generara trayectorias para cada salida del *Mapa Local* y como estos están limitados en su rango, por el rango de los sensores y de las proyecciones estimadas, las trayectorias generadas siempre terminarán llegando a un obstáculo u a una zona desconocida. Esto no es algo malo, ya que si se detectan obstáculos o zonas desconocidas a tiempo, se pueden tomar medidas lidiar con ellos. Por esto, se define como trayectoria libre una que logre navegar

al vehículo hasta distancias iguales o superiores a la distancia de detención del vehículo, sin pasar por obstáculos en el *Mapa Local* o en el *Ground Truth*. La ecuación 3.73 de (Wong, 2001) que se muestra a continuación muestra la ecuación de detención de un vehículo en movimiento.

$$S = \frac{W}{2gC_{ae}} \log\left(1 + \frac{C_{ae}V^2}{\mu W + f_r W\cos(\theta_s) + W\sin(\theta_s)}\right) \tag{4.1}$$

Donde:

- S: Es la distancia de detención.
- g: Es la aceleración de gravedad, aproximada en 0.8 m/s.
- W: Es el peso del vehículo, en este caso es de aproximadamente $1550Kg \cdot g = 15190N$.
- C_{ae}: Es una agrupación de constantes que caracterizan la resistencia aerodinámica del vehículo. Se muestran mas detalles en la ecuación 4.2.

  $$C_{ae} = \frac{\rho}{2} C_D A_f \approx 0.7 \tag{4.2}$$

  $\rho \approx 1.225$ es la densidad del aire, aunque varía bastante con la presión atmosférica y la temperatura. $C_D \approx 0.37$ es el coeficiente de arrastre, el valor es aproximado. $A_f \approx 1.7 \cdot 1.8 = 3.06m^2$ es el área transversal del vehículo, aproximada del ancho y alto de este mismo.
- V: Velocidad del vehículo en m/s.
- μ: Coeficiente de roce entre los neumáticos y el terreno. Para este caso se utiliza un coeficiente de 0.5 lo cual es bastante pesimista, tal como se ve en la tabla 4.1 que muestran ejemplos de terrenos y sus coeficientes de roce.

Tabla 4.1: Coeficiente de roce de diferentes terrenos

Terreno	Asfalto	Asfalto mojado	Tierra	Tierra Mojada	Nieve	Hielo
Coeficiente de Roce	0.9	0.7	0.65	0.55	0.15	0.08

- f_r: Se denomina resistencia rotacional, la cual es la fuerza que se opone al movimiento cuando un objeto (redondo) rueda por una superficie. Es muy pequeña en comparación a la fuerza de roce ya que en general es alrededor de un orden de magnitud menor. Por esta razón, se desprecia este factor.

- θ_s: Angulo del plano inclinado. Para esta aplicación se asume que se navegarán terrenos relativamente planos, por lo que $\theta_s \approx 0$.

Con las aproximaciones mencionadas anteriormente la ecuación 4.1 queda de la siguiente forma:

$$S \approx \frac{W}{2gC_{ae}} \log\left(1 + \frac{C_{ae}V^2}{\mu W}\right) \tag{4.3}$$

$$S \approx 1125 \log\left(1 + 0.00009V^2\right) \tag{4.4}$$

Los valores de V nunca superarán los $70km/h \approx 19.44m/s$. Con este valor máximo se puede aproximar $\log(1+\alpha) \approx \alpha$ que es solo válido para valores cercanos a 0 de α. Para el valor máximo de $V = 19.44m/s$ se aproxima $\log\left(1 + 0.00009V^2\right) = 0.0339 \approx 0.00009V^2 = 0.0333$ con una diferencia del 2% la que será mas pequeña para velocidades menores. Notamos en la ecuación 4.8 que luego de hacer la aproximación recién mencionada es posible hacer que esta no dependa del coeficiente de roce o del peso.

$$S \approx \frac{W}{2gC_{ae}} \log\left(1 + \frac{C_{ae}V^2}{\mu W}\right) \tag{4.5}$$

$$S \approx \frac{W}{2gC_{ae}} \frac{C_{ae}V^2}{\mu W} \tag{4.6}$$

$$\approx \frac{V^2}{2g\mu} \tag{4.7}$$

$$S \approx \frac{V^2}{9.8} \tag{4.8}$$

Ahora que se tiene la distancia de frenado, es necesario tomar en cuenta el tiempo de reacción del sistema, ya que durante ese tiempo el vehículo avanzará a velocidad constante sin reaccionar. Como el sistema funcionará a alrededor de $10Hz$, el retardo en ver un obstáculo no puede ser superior a $0.1s$ ahora se le suma a esto retardos en la toma de decisiones y en la ejecución de estas y se toma un valor conservador del tiempo de reacción de $t_r = 0.3s$. Por motivos de seguridad, se toma un tiempo extra de $t_s = 0.2$. Con estas nuevas variables se define la distancia segura, mientras que la distancia de detención total queda:

$$S_{tot} = t_r V + \frac{V^2}{9.8} \tag{4.9}$$

$$S_{segura} = t_s V + S_{tot} \tag{4.10}$$

$$S_{segura} = (t_r + t_s)V + \frac{V^2}{9.8} \tag{4.11}$$

$$S_{segura} = 0.5V + \frac{V^2}{9.8} \tag{4.12}$$

Con la distancia segura, se definen diferentes tipos de trayectorias posibles:

- *Camino Libre*: Trayectoria que logra recorrer la distancia segura sin encontrar obstáculos en el *Ground Truth* ni en el *Mapa Local*.
- *Parada correcta*: Trayectoria que encuentra un obstáculo en el *Mapa Local* a una distancia D_{ML} y otro en el *Ground Truth* a una distancia D_{GT}. Se verifica que la distancia entre dichos obstáculos sea pequeña, $D_{GT} \cdot (1 - \alpha) < D_{MP} < D_{GT} \cdot (1 + \alpha)$ con $\alpha = 0.1$. Esto indica una detención correcta y proporciona la información suficiente para que el vehículo evite el obstáculo.
- *Parada Falsa*: Trayectoria que encuentra obstáculos en el *Mapa Local*, pero no en el *Ground Truth*. Este obstáculo no existe en la realidad, por lo que se denomina obstáculo ficticio. Este provoca detenciones indeseadas del vehículo.
- *Choque*: Trayectoria que encuentra algún obstáculo en el *Ground Truth* sin encontrarlo en el *Mapa Local*. Esto provocaría una colisión con el obstáculo no detectado por el sistema.

Para cada *Mapa Local* se genera una trayectoria, el método de generación que se utilizó es una variación del método de campos potenciales, el cual se detallará más adelante. Esta trayectoria se clasificará en una de las 4 opciones mostradas anteriormente para velocidades entre 0 km/h y 100 km/h. Al variar la velocidad de conducción, la distancia segura irá cambiando, por lo que su clasificación también cambiará. A velocidades bajas, la distancia segura es baja también, por lo que es más fácil encontrar un *Camino Libre*. Para cada trayectoria se generará un vector con la clase de esta para cada velocidad. Luego de obtener dichos vectores, se recorren las velocidades y se cuentan los eventos de cada tipo y se obtienen las curvar mostradas en la Figura 4.4.

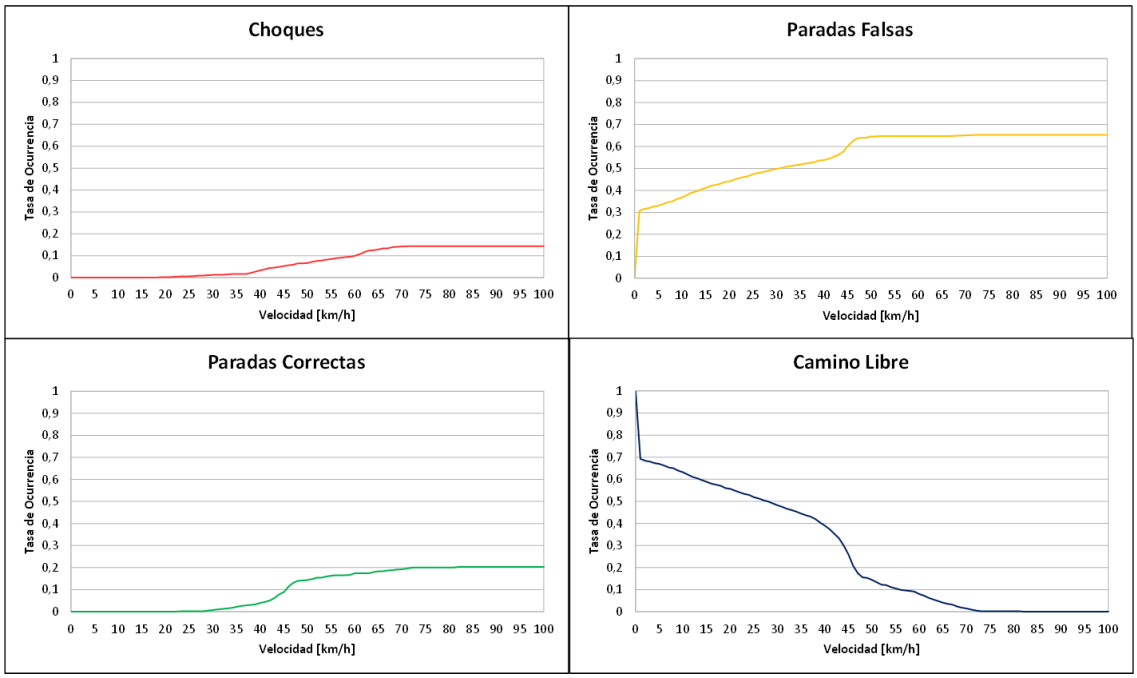


Figura 4.4: Representación del desempeño mediante numero de eventos en función de la velocidad.

Las curvas mostradas en la Figura anterior fueron obtenidas de la base de datos de entrenamiento. En estas se ve que a velocidades menores a los 25 km/h no se observa ningún posible choque. Esto quiere decir que, para velocidades menores a esta, no se encuentra ningún obstáculo en el *Ground Truth* o bien todos los que se encuentran son detectados también en el *Mapa Local*. En cuanto a las otras curvas, se observa que a medida que aumenta la velocidad, hay cada vez menos camino libre, en cambio las paradas falsas y las paradas correctas aumentan con la velocidad.

4.1.4 Generación de Trayectorias

Para lograr generar las trayectorias se utiliza una variación del método de campos potenciales. El método se basa en los siguientes principios:

- Cada obstáculo genera un campo potencial que repele al vehículo.
- El punto objetivo genera vacío potencial que atrae al vehículo.
- Las derivadas en X e Y del campo potencial definen las fuerzas que actúan sobre el vehículo.

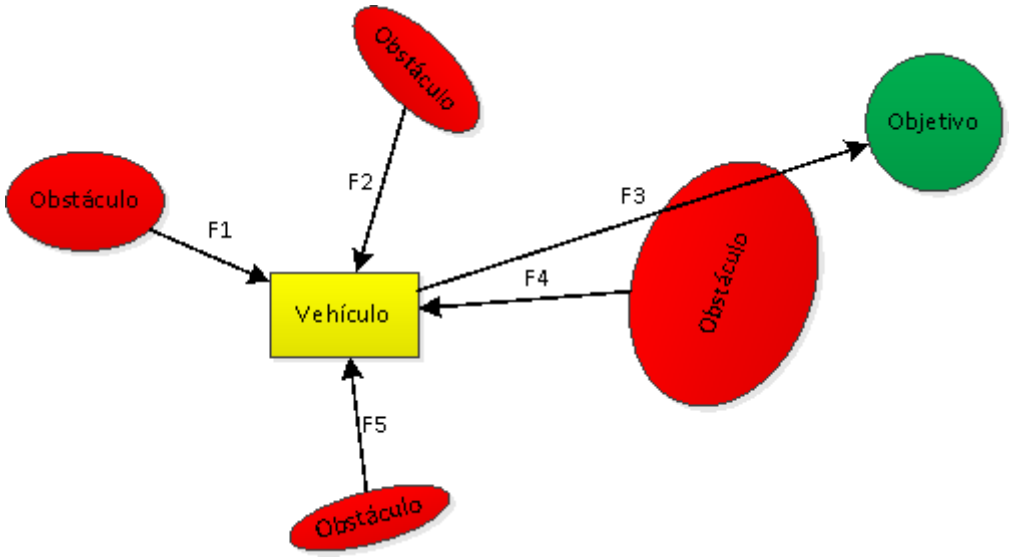


Figura 4.5: Ejemplo del clásico algoritmo de campos potenciales

En la Figura 4.5 se muestran las fuerzas que actúan sobre el vehículo en ejemplo de campos potenciales. Estas fuerzas se deben a la acción de los campos potenciales de cada obstáculo y del objetivo. Los tipos de funciones que se pueden utilizar para los campos potenciales son variadas:

- *Inversa* - $U(d) = 1/(\alpha d)$
- *Exponencial* - $U(d) = Ae^{-\frac{(d)^2}{2\sigma^2}}$
- *Lineal* - $U(d) = max(A - \alpha d, 0)$

Las funciones *Inversa* y *Exponencial* son las que normalmente se utilizan en campos potenciales. La primera se utiliza ya que hace que la partícula, vehículo en este caso, nunca entre a los obstáculos, debido a que el potencial se hace infinito cerca de estos. La segunda se utiliza debido a su simplicidad, cuando se utiliza una grilla de ocupancia, el campo potencial para cada celda se puede calcular simplemente filtrando la grilla con una gaussiana.

En este caso se utiliza una función *Lineal* para el campo de los obstáculos, mientras que para el objetivo se utiliza una fuerza constante, sin importar la ubicación del vehículo. Esto tiene el efecto de hacer que las trayectorias que se generan sean mas suaves hacia el objetivo, pero con la desventaja que es posible que el vehículo intente cruzar obstáculos, momento en el cual se terminará el algoritmo.

Para cada *Mapa Local* se realizarán los siguientes pasos:

1 Calcular el campo potencial de los obstáculos, esto se hace asignando un valor base a los obstáculos, luego utilizar resta de filtros morfológicos para encontrar los bordes siguientes, asignar valores menores al paso anterior, volver a encontrar bordes, etc... Repitiendo hasta llegar a 0.

2 Encontrar la derivada en X e Y del campo, esto se realiza con simples filtros pasa altos direccionales. Como ya se mencionó antes, estas derivadas representan la fuerza aplicadas al vehículo en X e Y respectivamente.

3 Calcular la fuerza total sobre el vehículo en la posición actual. Tomando en cuenta la fuerza en X e Y, además de la fuerza constante que empuja al vehículo hacia el objetivo.

4 Mover la posición del vehículo en el *Mapa Local* de acuerdo a la fuerza total.

5 Repetir los pasos 3 y 4 hasta encontrar un obstáculo en el *Mapa Local* y en el *Ground Truth*.

Es importante destacar que el paso 5 no es trivial. Si se encuentra un obstáculo en el *Ground Truth* primero, no es ningún problema, simplemente se guarda la distancia hasta dicho obstáculo y se sigue la trayectoria hasta encontrar un obstáculo en el *Mapa Local*. En el caso contrario, se encuentra el primer obstáculo en el *Mapa Local*. Como la trayectoria se genera con el *Mapa Local* al encontrarnos con un obstáculo ya no se cuenta con información para seguir generando la trayectoria. En este caso, se asume que el vehículo sigue en linea recta hasta que se encuentre un obstáculo en el *Ground Truth*. Un punto importante, es que siempre se encuentran obstáculos en ambos mapas, ya que el *Mapa Local* tiene un rango acotado y el *Ground Truth* muestra un recorrido finito, tal como se muestra en la Figura 4.6.

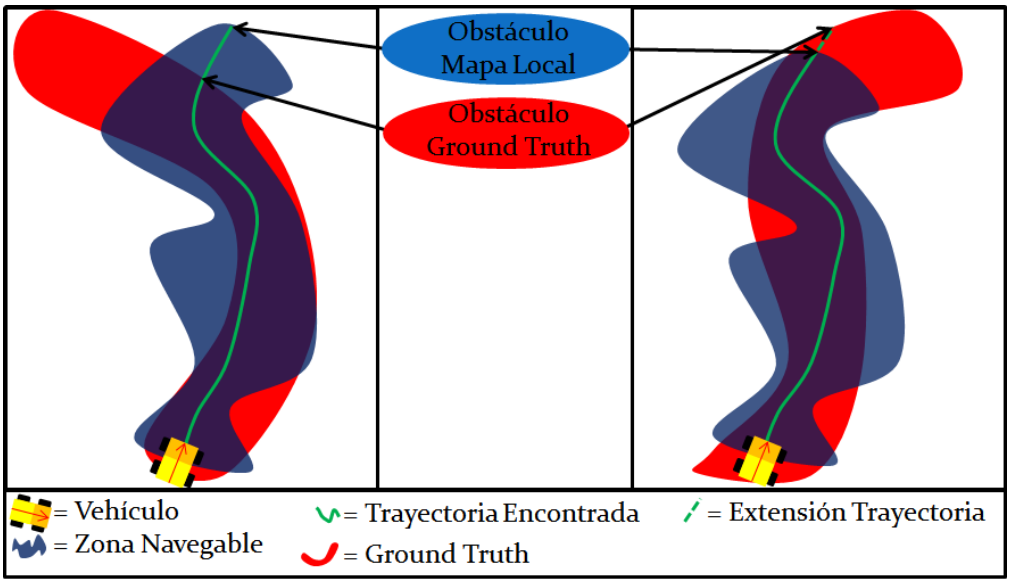


Figura 4.6: Ejemplo de las trayectorias generadas. Ambas son iguales en su mayor parte, salvo al final, ya que una es mas larga que la otra. El caso de la izquierda se encuentra un obstáculo en el *Ground Truth* primero, mientras que a la derecha se encuentra uno en el *Mapa Local*

Con este algoritmo, se obtiene la distancia recorrida hasta encontrar un obstáculo en cada mapa. De esta manera, para una velocidad específica, se puede saber si clasificar la trayectoria como *Camino Libre*, *Parada Correcta*, *Parada Falsa* o *Choque*, siguiendo las definiciones dadas anteriormente.

4.2 Resultados

Todo lo explicado anteriormente en este documento se aplica para poder encontrar los resultados experimentales. Cabe destacar que todos los parámetros dados en la sección de *Metodología*, son un punto de partida, pero que luego fueron ajustados, utilizando el conjunto de entrenamiento. Para realizar dicho ajuste, se utilizo un método exhaustivo. Es decir que para ambos métodos, se dividió el espacio de parámetros en intervalos regulares y se probaron todas las combinaciones posibles. Luego, se calcularon las trayectorias para cada resultado y a partir de estas se obtuvieron las curvas descritas anteriormente. A continuación se muestran las curvas encontradas para ambas variantes del algoritmo, cada una usando y sin usar la información de las imágenes. Se ejemplifica mostrando la curva del punto de operación seleccionado, parámetros muy altos (casi todo se detecta como obstáculo) y parámetros muy bajos (casi todo se ve como camino libre).

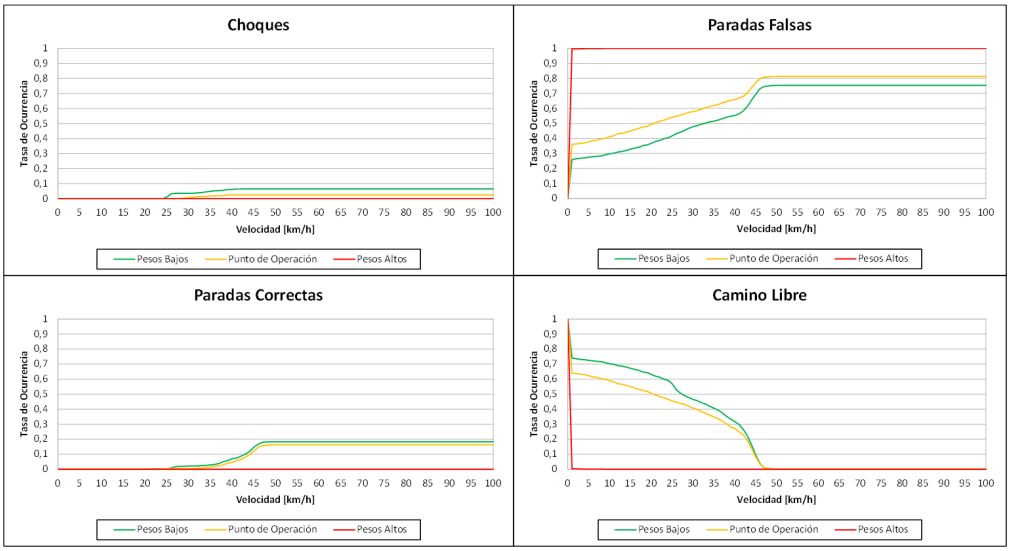


Figura 4.7: Comparación de los experimentos del método de frecuencias con diferentes pesos en la detección de obstáculos.

En la Figura 4.7 se observa el efecto de cambiar los pesos de los filtros, siendo más tolerante (bajando los pesos) o siendo más estricto (subiendo los pesos). Como es de esperar, los choques disminuyen al ser mas restrictivo, pero aumentan las paradas falsas. Esto implica que que el punto de operación se tiene que seleccionar intentando llegar a un equilibrio. Cabe destacar que la elección de estos parámetros (p) depende de la aplicación, en este caso, el parámetro que se maximizó ($Puntaje(p)$) fue el área bajo la curva del camino libre (A_{cl}), sumada al área bajo la curva de las paradas correctas (A_{pc}), menos el área bajo la curva de los choques (A_c).

$$Puntaje(p) = A_{cl} + A_{pc} - A_c \tag{4.13}$$

Con esta ecuación se encontró el punto de operación para cada una de las variantes. En las Figuras siguientes se muestran estas mismas curvas para las otras variantes.

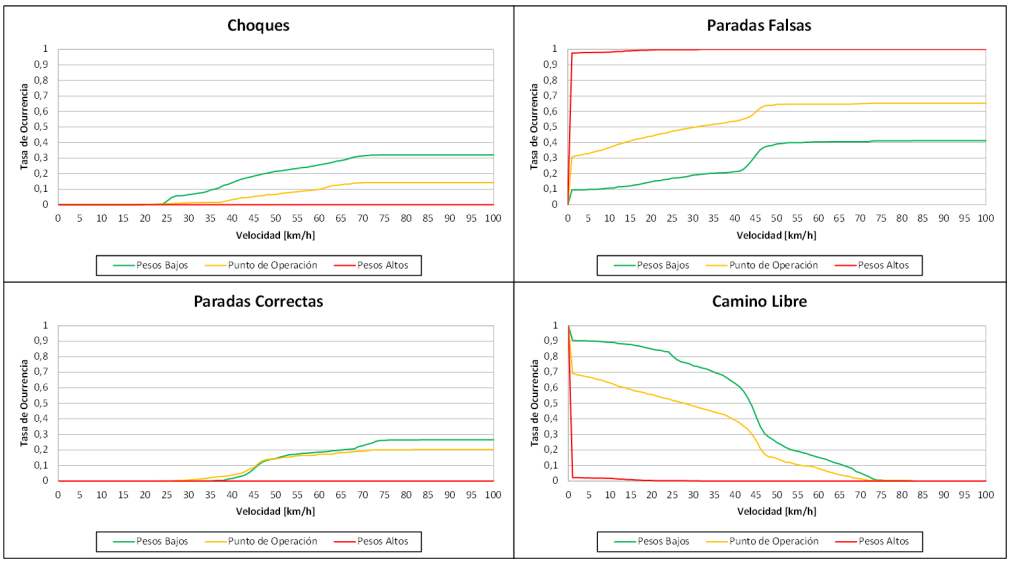


Figura 4.8: Comparación de los experimentos del método de frecuencias, utilizando las imágenes, con diferentes pesos en la detección de obstáculos.

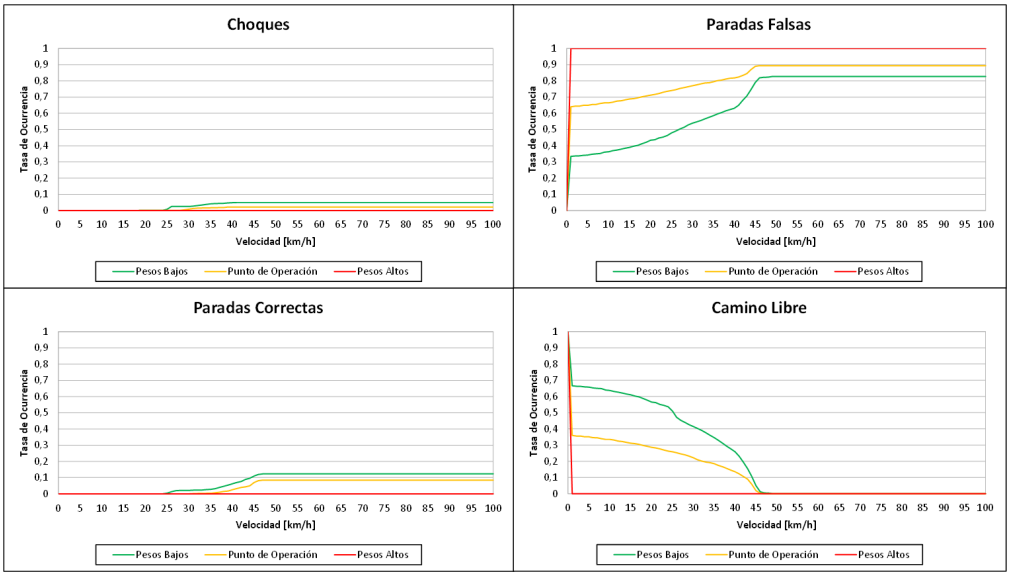


Figura 4.9: Comparación de los experimentos del método de diferencias de altura con diferentes pesos en la detección de obstáculos.

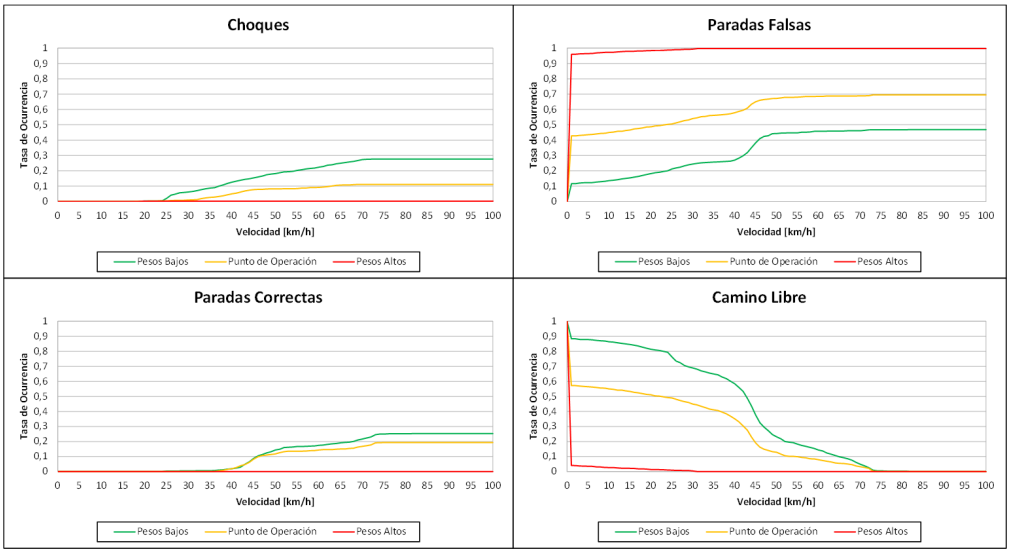


Figura 4.10: Comparación de los experimentos del método de diferencias de altura, utilizando las imágenes, con diferentes pesos en la detección de obstáculos.

Es importante notar que estos parámetros afectan solamente el *Mapa de Costos de la Morfología*, pero se realizó la optimización del cada sistema completo. En las Figuras 4.7 y 4.9 se nota claramente el rango de la *Morfología del Terreno*, ya que alrededor de los $45[km/h]$ se observa que ya no hay mas camino libre, para cualquier peso. La distancia segura asociada a los $45[km/h]$ corresponde a $22.2[m]$, mientras que el rango de la *Morfología del Terreno* es de $25[m]$. A continuación se muestran los resultados del punto de operación para todas las variantes.

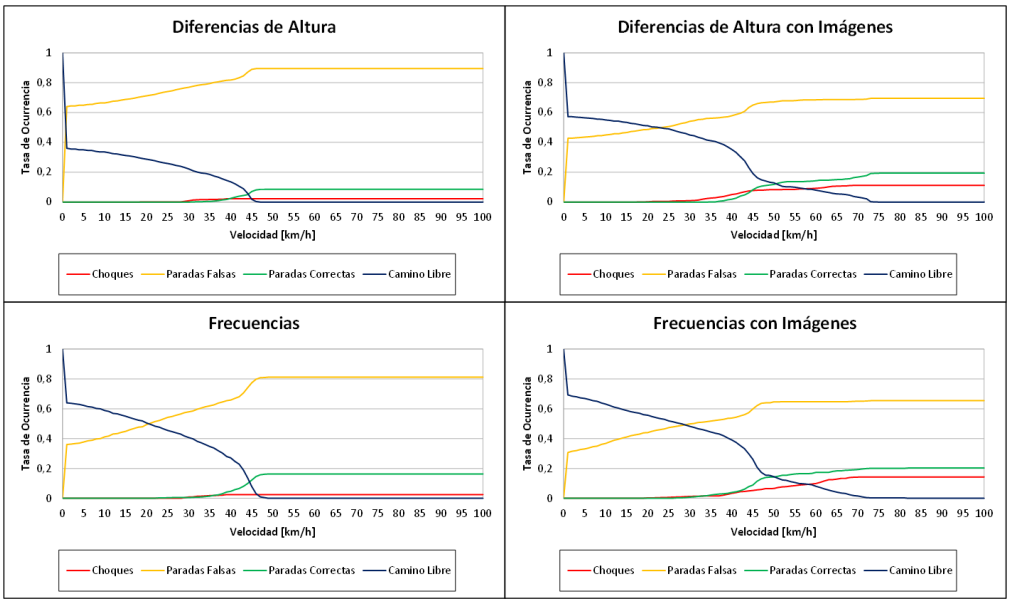


Figura 4.11: Resultados experimentales en el conjunto de entrenamiento.

Se observa que las curvas tienen el mismo comportamiento general en todos los casos, los *Choques*, *Paradas Falsas* y *Paradas Correctas* aumentan con la velocidad mientras que el *Camino Libre* baja. Lo que cambia es el ritmo de crecimiento u decrecimiento de estas curvas.

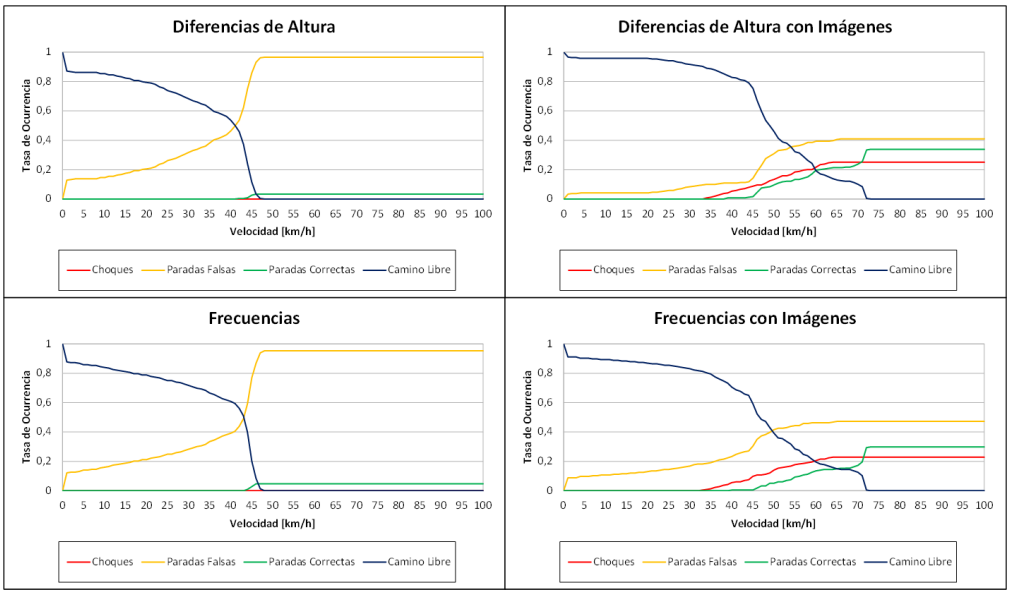


Figura 4.12: Resultados experimentales en el conjunto de prueba.

Es importante notar que la suma de estas 4 curvas siempre da 1 para cualquier velocidad, por lo

que solo 3 de estas son representativas del rendimiento del sistema. Pero tener las 4 curvas facilitará la comparación de sistemas que se realizará mas adelante.

4.3 Análisis de Resultados

En esta sección se compararán y analizarán los resultados obtenidos, intentando determinar los aportes de cada uno de los métodos utilizados. Para lograr una mejor comparación se muestran todas las curvas de un tipo en el mismo gráfico, de forma que se puede visualizar mejor las diferencias de desempeño de las variantes probadas.

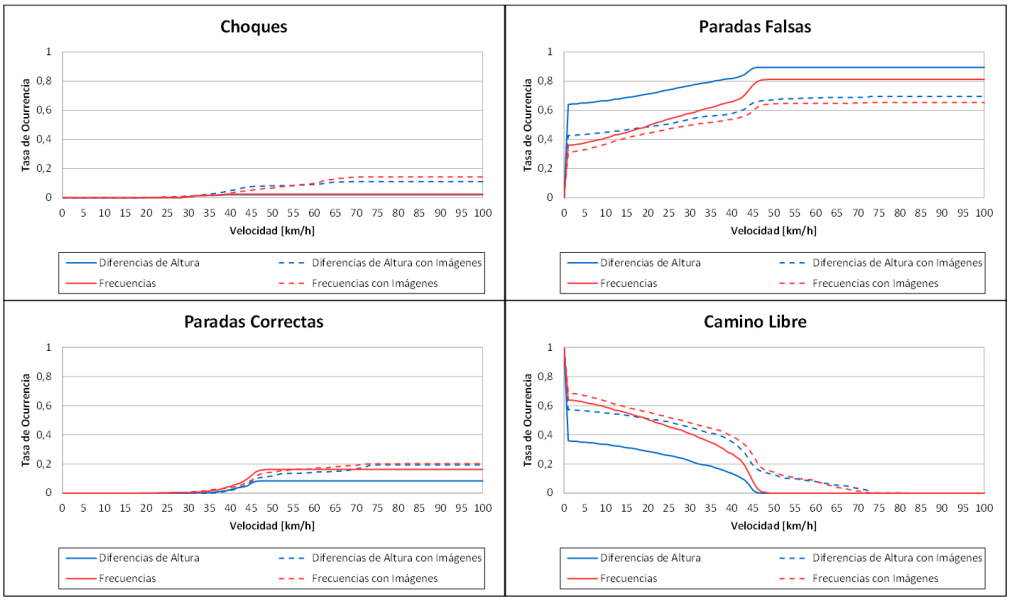


Figura 4.13: Comparación de resultados en el conjunto de entrenamiento.

En la Figura 4.13 se observa que no hay una clara jerarquía en las curvas debido a que esas curvas no son son estrictamente mayores o menores que las otras. Sin embargo se pueden comparar las curvas cada método con su análogo con imágenes en las curvas de *Paradas Falsas* y *Camino libre*. En estas se ve claramente que el utilizar imágenes disminuye las paradas falsas y aumenta el camino libre. Esta conclusión podría llevar a pensar que el uso de las imágenes implica un mejor desempeño, pero se observa en las curvas de *Choques* que los métodos con imágenes provocan mas choques a partir de los $35[km/h]$. Es importante notar que esta base de datos corresponde al conjunto de entrenamiento, por lo que no se deben sacar demasiadas conclusiones de estos resultados. Se continuará analizando el conjunto de pruebas.

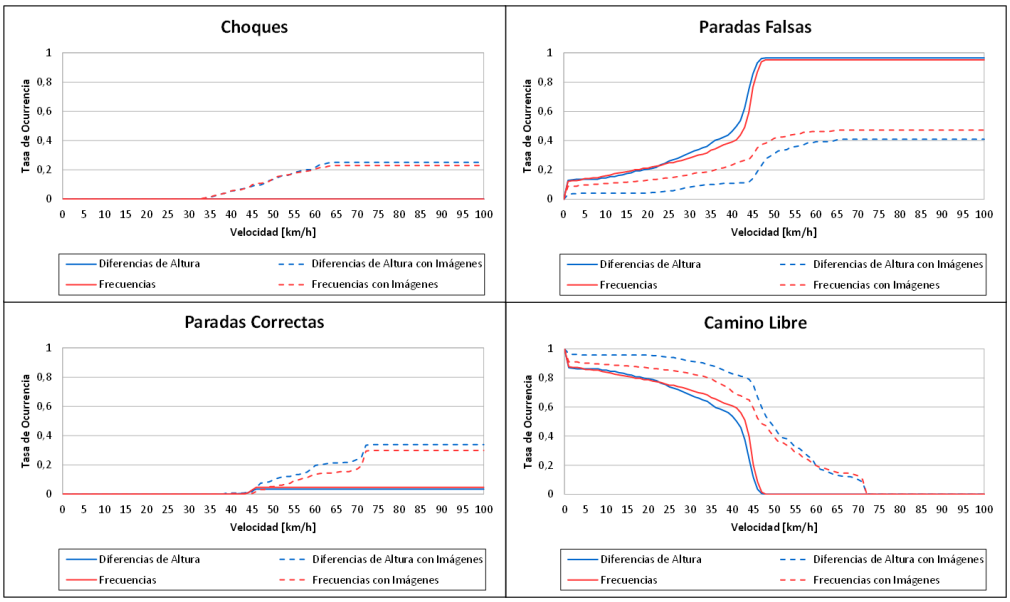


Figura 4.14: Comparación de resultados en el conjunto de prueba.

Las curvas mostradas en la Figura 4.14 muestran un caso diferente, las diferencias entre los métodos son menores, pero hay muy pocos cruces entre las curvas, por lo que se puede definir una clara jerarquía para cada curva. En las curvas de *Paradas Falsas*, *Paradas Correctas* y *Camino Libre* se observa la siguiente jerarquía, de mejor a peor:

1 *Diferencias de altura con imágenes*

2 *Frecuencias con imágenes*

3 *Frecuencias*

4 *Diferencias de altura*

Mientras que en las curvas de *Choques* se observa que los métodos sin imágenes nunca chocan. En cambio los métodos con imágenes comienzan a chocar a partir de los $35[km/h]$. Lo que también ocurrió con el conjunto de entrenamiento. En estas curvas el método de *Frecuencias con imágenes* presenta ligeramente menos choques que el de *Diferencias de altura con imágenes*.

Otro punto importante a notar es que en los métodos con imágenes el *Camino Libre* llega a cero alrededor de los $72[km/h]$, lo cual implica una distancia segura de $50.8[m]$. El rango del *Mapa Local* es de $50[m]$, lo que explica muy bien este número.

En resumen, la utilización de imágenes tiene muy buen impacto sobre el rendimiento del sistema, siempre que se utilicen para velocidades inferiores a los $35[km/h]$. Para velocidades superiores, es

necesaria información mas lejana. En el caso de las imágenes mientras más lejana la información esta es más ruidosa, debido a la proyección. Como esta no es perfecta, cualquier error es amplificado considerablemente al proyectar sobre puntos lejanos. Por esto se observa un aumento en los choques al utilizar la información de las imágenes a velocidades mayores. Otro punto importante es que la densidad de las mediciones del LIDAR es baja para distancias altas y por lo tanto se tiene que completar la información faltante confiando exclusivamente de las imágenes. Estos 2 efectos combinados son los que provocan los aumentos de los *Choques* y aumentar las *Paradas Correctas* y *Camino Libre*. Por lo tanto las imágenes son una gran herramienta para suplementar la información adoptada por los LIDAR, pero no dan resultados confiables por si mismas.

Tomando todo en cuenta, se puede afirmar que el método de *Diferencias de altura con imágenes* es el que tiene el mejor desempeño para velocidades menores a los $35[km/h]$. No se recomienda navegar a velocidades superiores, ya que los *Choques* no son aceptables en este tipo de aplicaciones.

Capítulo 5

Conclusiones

Se diseño e implementó un sistema de estimación del entorno de un vehículo autónomo utilizando la morfología del terreno, las imágenes segmentada y el estado del vehículo. Se evaluó su uso para la planificación de trayectorias seguras para la conducción autónoma.

El análisis de los resultados muestra que el método que mejor se comporto fue el de diferencias de altura con imágenes, siempre y cuando se limite la velocidad a $35[km/h]$. para evitar los *Choques*. Cabe destacar que en el sistema se tomó un enfoque conservador al interpretar las zonas desconocidas de la *Morfología del Terreno* como obstáculos, a diferencia de otros enfoques que interpretan estos como terreno navegable (Urmson et al., 2006). Con este método se logran cumplir los objetivos específicos en su gran mayoría.

Se logró la integración con el proyecto Vehículo Autónomo del AMTC, ya que el sistema se encuentra programado en C++, utiliza ROS y las entradas y salidas son compatibles con los demás bloques del proyecto. Lo único que falta para la integración total es el juntar los módulos en el vehículo en lugar de probar el software con datos grabados (los que tienen exactamente el mismo formato y el sistema responde a ellos sin saber si son grabados o en vivo).

La robustez frente a la inestabilidad de las imágenes se logro mediante la estimación del *Mapa Segmentado*, el cual requiere de múltiples observaciones antes de cambiar su estimación. Esto produce una reacción ligeramente mas lenta, pero que es muy necesaria, debido a la inestabilidad de las imágenes. Sumado a esto, si por cualquier razón (polvo, mala iluminación, brillos u otras interferencias) la calidad de las imágenes es muy pobre por mucho tiempo y el *Mapa Segmentado* esta vacío (todo el mapa desconocido), el sistema puede funcionar perfectamente utilizando solamente la información de la *Morfología del Terreno*.

Tal como se ve en los diversos ejemplos de la *Morfología del Terreno*, esta no es completa y contiene diversos vacíos, los que provocan zonas desconocidas, las cuales son magnificadas al momento de procesar, ya que los métodos utilizados para estimar la navegabilidad son basados en análisis de la

vecindad. Aun con estos vacíos, el sistema logra lidiar ellos cuando hay información de las imágenes, en caso contrario simplemente se asumen desconocidos y se tratan como obstáculos. Nuevamente esta es una estrategia pesimista, la cual se elige para minimizar el riesgo del vehículo.

Las medidas de desempeño diseñadas, utilizan un generador de trayectorias básico, lo cual demuestra que las salidas del sistema permiten encontrar trayectorias seguras para el vehículo, aunque en algunas ocasiones estas trayectorias impliquen frenar para evitar posibles accidentes. Estas mismas medidas de desempeño permitieron evaluar el rendimiento de las diferentes variantes del sistema implementadas y escogió uno de estos métodos probados como la mejor solución. El método escogido es el de diferencias de alturas con imágenes, restringiendo la velocidad a un máximo de $35[km/h]$.

También es importante destacar que la información visual es utilizada para mejorar el rendimiento del sistema mejorando la calidad del *mapa local* entregado. La calidad de este mapa y el aporte de las imágenes serían mucho mayores si se contara con varias cámaras y se utilizara visión estéreo, esto permitiría disminuir de manera considerable la inestabilidad de las imágenes y su proyección, lo que llevaría a mejores resultados y un mejor desempeño del sistema en general.

Capítulo 6

Bibliografía

Andersen, J. C., Blas, M. R., Ravn, O., Andersen, N. A., & Blanke, M. (2006). Traversable terrain classification for outdoor autonomous robots using single 2d laser scans. *Integrated Computer-aided engineering*, *13*(3), 223-232.

Bernuy, F., Ruiz del Solar, J., Parra, I., & Vallejos, P. (2011, Dec). Adaptive and real-time unpaved road segmentation using color histograms and ransac. *2011 9th IEEE International Conference on Control and Automation (ICCA)*. Retrieved from `http://dx.doi.org/10.1109/ICCA.2011.6138056` doi: 10.1109/icca.2011.6138056

Braid, D., Broggi, A., & Schmiedel, G. (2006, Sep). The terramax autonomous vehicle. *Journal of Field Robotics*, *23*(9), 693-708. Retrieved from `http://dx.doi.org/10.1002/rob.20140` doi: 10.1002/rob.20140

Buehler, M., Iagnemma, K., & Eds, S. S. (2006). *The 2005 darpa grand challenge: The great robot race* (M. Buehler, S. Singh, & K. Iagnemma, Eds.). Springer.

Crane III, C. D., Armstrong II, D. G., Touchton, R., Galluzzo, T., Solanki, S., Lee, J., ... et al. (2006, Aug). Team cimar's navigator: An unmanned ground vehicle for the 2005 darpa grand challenge. *Journal of Field Robotics*, *23*(8), 599-623. Retrieved from `http://dx.doi.org/10.1002/rob.20127` doi: 10.1002/rob.20127

Fischler, M. A., & Bolles, R. C. (1981, Jun). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381-395. Retrieved from `http://dx.doi.org/10.1145/358669.358692` doi: 10.1145/358669.358692

Hoffman, R., & Krotkov, E. (1990). Terrain roughness measurement from elevation maps. In *1989 advances in intelligent robotics systems conference* (p. 104-114).

Häselich, M., Arends, M., Lang, D., & Paulus, D. (2011). Terrain classification with markov random fields on fused camera and 3d laser range data. In *Proceedings of the 5th european conference on mobile robotics (ecmr)*.

INE, & Carabineros. (2011). *Carabineros: informe anual.* (Tech. Rep.). Instituto Nacional de

Estadísticas (Chile), en convenio con Carabineros de Chile.

Langer, D., Rosenblatt, J., & Hebert, M. (1994). A behavior-based system for off-road navigation. , *10*(6), 776–783. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=338532 doi: 10.1109/70.338532

Papadakis, P. (2013, Apr). Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence*, *26*(4), 1373-1385. Retrieved from http://dx.doi.org/10.1016/j.engappai.2013.01.006 doi: 10.1016/j.engappai.2013.01.006

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T. B., Leibs, J., ... Ng, A. Y. (2009). Ros: an open-source robot operating system. In *Icra workshop on open source software.*

Tapia-Espinoza, R., & Torres-Torriti, M. (2013, Mar). Robust lane sensing and departure warning under shadows and occlusions. *Sensors*, *13*(3), 3270-3298. Retrieved from http://dx.doi.org/10.3390/s130303270 doi: 10.3390/s130303270

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., ... et al. (2006, Sep). Stanley: The robot that won the darpa grand challenge. *Journal of Field Robotics*, *23*(9), 661-692. Retrieved from http://dx.doi.org/10.1002/rob.20147 doi: 10.1002/rob.20147

Trepagnier, P. G., Nagel, J., Kinney, P. M., Koutsougeras, C., & Dooner, M. (2006, Aug). Kat-5: Robust systems for autonomous vehicle navigation in challenging and unknown terrain. *Journal of Field Robotics*, *23*(8), 509-526. Retrieved from http://dx.doi.org/10.1002/rob.20128 doi: 10.1002/rob.20128

Urmson, C., Ragusa, C., Ray, D., Anhalt, J., Bartz, D., Galatali, T., ... Struble, J. (2006, Aug). A robust approach to high-speed navigation for unrehearsed desert terrain. *Journal of Field Robotics*, *23*(8), 467–508. Retrieved from http://dx.doi.org/10.1002/rob.20126 doi: 10.1002/rob.20126

Wong, J. Y. (2001). *Theory of ground vehicles.* John Wiley & Yung.

Anexo

La estructura del mensaje *MapElevationArray* es la siguiente:

- *Header header:* Es el encabezado estándar de todos los mensajes de ROS, entre otra información, contiene el ID del frame de referencia y el tiempo asociado al mensaje. Es importante destacar que no hay que confundir este tiempo con el tiempo de emisión, ya que en general el tiempo de referencia indica el tiempo cuando el dato original fue tomado y este se mantiene durante todas las etapas de procesamiento.
- *float32 cell_width:* El ancho en metros de cada celda. El sistema funciona con un ancho de 0.2 metros.
- *float32 cell_height:* El alto en metros de cada celda. El sistema funciona con un alto de 0.2 metros.
- *uint32 width:* La cantidad de celdas a lo ancho de la matriz. El sistema funciona con 200 celdas de ancho.
- *uint32 height:* La cantidad de celdas a lo alto de la matriz. El sistema funciona con 200 celdas de alto.
- *float32*[] *z:* Vector que guarda los valores de cada una de las celdas de la matriz. Tiene un largo de *width*height*.

La clase *grid_map_2d* se utiliza para guardar la información del mensaje de la siguiente manera:

- *float cellDimensionX:* Se guarda directamente la información de *cell_width*.
- *float cellDimensionY:* Se guarda directamente la información de *cell_height*.
- *float mapDimensionX:* Representa el tamaño de dimensión X del mapa en metros. Se guarda *cell_width*width*.

- *float mapDimensionY:* Representa el tamaño de dimensión Y del mapa en metros. Se guarda *cell_height*height*.
- *cv::Map data:* Clase para manejar imágenes y matrices de la librería OpenCV. El tipo de dato guardado puede elegirse, en este caso se utilizan valores de tipo float. El uso de esta clase permite utilizar todos los filtros y funciones disponibles en la librería.