uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería Informática
Curso Académico 2020-2021

Trabajo Fin de Grado

"DETECCIÓN DE OBJETOS EN MOVIMIENTO CON SENSOR LIDAR"

Alberto Ramos González

Tutor/es

Juan Pedro Llerena Caña

Colmenarejo, Julio 2021

RESUMEN

La detección de objetos en movimiento es una tarea fundamental en el mapeado, navegación y seguimientos de objetos. LIDAR es uno de los sensores más reconocidos en la tarea de detectar objetos en movimiento mediante la captura de nubes de puntos de una escena, destacando su función en aplicaciones como robótica, videovigilancia, automatización de vehículos, etc.

El presente trabajo consiste en crear un sistema computacional capaz de detectar objetos en movimiento con un sensor LIDAR.

Para alcanzar este objetivo, se diseña y desarrolla un dispositivo que permita la recopilación de información del entorno. Se utiliza una Raspberry Pi que conecta un LIDAR de corto alcance y un conjunto de sensores (GPS, cámara, barómetro, acelerómetro, giroscopio y magnetómetro). Con este dispositivo se puede registrar información del entorno proporcionada por los distintos sensores con el fin de ser tratado en diferido. Se realiza una agrupación de las nubes de puntos con el algoritmo de *clustering* DBSCAN, una segmentación de los datos utilizando la técnica de SMOTE y una asociación de las nubes de puntos mediante algoritmo ICP con el fin de detectar el movimiento. Además, se realiza una transformación de los puntos obtenidos por el LIDAR al sistema de coordenadas geodésicas mundial WGS-84, de esta manera se puede expresar los datos obtenidos en posición con respecto al elipsoide de referencia.

Para verificar el movimiento generado en escena se ha desarrollado un sistema comparativo que detecta el movimiento en las imágenes capturas por la cámara que integra la Raspberry Pi, esta detección se basa en la sustracción de imágenes.

Este trabajo puede ser utilizado como punto de partida para aplicaciones relacionadas con detección y escaneado de escenas mediante un dispositivo.

El resultado obtenido finalmente es un dispositivo capaz de registrar conjuntos de datos y un sistema que detecta el movimiento en nubes de puntos pudiendo representarlas en WGS-84.

Palabras clave: LIDAR, detección de objetos en movimiento, nubes de puntos, WGS-84, ICP, DBSCAN, SMOTE.

ABSTRACT

The detection of moving objects is a fundamental task in the mapping, navigation and tracking of objects. LIDAR is one of the most recognized sensors in the task of detecting moving objects by capturing point clouds from a scene, highlighting its role in applications such as robotics, video surveillance, vehicle automation, etc.

The present work consists of creating a computational system capable of detecting moving objects with a LIDAR sensor.

To achieve this goal, a device that allows the collection of information from the environment is designed and developed. A Raspberry Pi is used to connect a short-range LIDAR and a set of sensors (GPS, camera, barometer, accelerometer, gyroscope and magnetometer). With this device, information from the environment provided by the different sensors can be recorded in order to be processed in deferred. A clustering of the point clouds is performed using the DBSCAN clustering algorithm, a segmentation of the data using the SMOTE technique and an association of the point clouds using ICP algorithm in order to detect motion. In addition, a transformation of the points obtained by the LIDAR to the WGS-84 world geodetic coordinate system is performed, in this way it is possible to express the data obtained in position with respect to the reference ellipsoid.

To verify the movement generated in the scene, a comparative system has been developed to detect the movement in the images captured by the camera that integrates the Raspberry Pi, this detection is based on the subtraction of images.

This work can be used as a starting point for applications related to detection and scanning of scenes through a device.

The final result is a device capable of recording data sets and a system that detects movement in point clouds and can represent them in WGS-84.

Keywords: LIDAR, moving object detection, point clouds, WGS-84, ICP, DBSCAN, SMOTE.

AGRADECIMIENTOS

El presente trabajo pone fin a una etapa enriquecedora tanto a nivel personal como profesional. Por ello, me gustaría expresar mi más sincero agradecimiento a todas las personas que me han ayudado y apoyado durante esta etapa de mi vida.

En primer lugar, a mis padres, por apoyarme en todas las decisiones que he tomado en mi vida y ayudarme en todo momento.

A mis amigos que siempre han estado ahí, tanto en los momentos buenos como en los malos.

A mi novia, la persona que más me ha acompañado y apoyado en esta etapa, ayudándome a dar mi mejor versión.

A mi tutor, Juan Pedro Llerena, por haberme dado la posibilidad de emprender este trabajo, orientándome y ayudándome durante su realización.

Por último, agradecer a la Biblioteca Universidad Carlos III de Madrid y su docencia por las orientaciones recibidas.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ACRÓNIMOS

LIDAR	*Light Detection And Ranging.*
2D	*2 Dimension.*
RGB	*Red Green Blue.*
GPS	*Global Positioning System.*
IMU	*Inertial Measurement Unit.*
UAV	*Unmanned Aerial Vehicles.*
DL	*Deep Learning*
ENU	*East North Up.*
WGS 84	*World Geodetic System 1984.*
ECEF	*Earth-centered, earth-fixed.*
ISO	*International Standardization Organization.*
IEEE	*Institute of Electrical and electronics Engineers.*
Wi-Fi	*Wireless Fidelity.*
IVA	Impuesto sobre el Valor Añadido.
ICP	*Iterative Closest Point.*
DBSCAN	*Density-Based Spatial Clustering of Applications with Noise*.
HDBSCAN	*Hierarchical Density-Based Spatial Clustering Application with Noise.*
OPTICS	*Ordering Points To Identify the Clustering Structure*.
DENCLUE	*DENsity-based CLUstEring*.
INS	*Inertial Navigation System*.
RMSE	*Root-mean-square error.*
SMOTE	*Synthetic Minority Oversampling Technique.*
ML	*Machine Learning*
UART	*Universal Asynchronous Receiver-Transmitter.*
I2C	*Inter-Integrated Circuit.*
HOG	*Histogram of Oriented Gradients*.
DPM	*Deformable Part-based Model.*
ADAS	*Advanced Driver Assistance Systems*.
CV	*Computer Vision*.
CSV	*Comma-Separated Values*.

1. INTRODUCCIÓN.

La tecnología informática está vinculada con numerosas áreas del conocimiento, un claro ejemplo es la detección de objetos, la cual se encuentra estrechamente relacionada con técnicas de aprendizaje automático. La detección de objetos tiene numerosas aplicaciones en diferentes áreas, y es el punto de partida del presente trabajo. Se pretende realizar una investigación sobre la detección de objetos en movimiento con sensor LIDAR (*Light Detection And Ranging*). Esta tecnología se encuentra en continuo avance, hasta el punto de detectar objetos en movimiento en diferentes entornos, no obstante, sigue presentando dificultades y problemas que son la motivación de estudios de vanguardia.

1.1. Motivación.

La detección de objetos en movimiento es definida por Jaya S. Kulchandani y Kruti J. Dangarwala [1] como la tarea de identificar el movimiento físico de un objeto en una región o área determinada. En las últimas décadas, la detección de objetos en movimiento ha sufrido grandes avances, reconociéndose como un paso crucial en una amplia gama de aplicaciones como, la robótica, automatización de vehículos, videovigilancia y numerosos campos dentro de la industria, donde se debe detectar objetos determinados como personas, animales, vehículos u otros. La detección de objetos en movimiento ha demostrado ser un problema desafiante debido a diferentes motivos como la clasificación de sombras como objetos, el fondo de una escena con movimiento, cambios de iluminación y mapas o entornos desconocidas, donde los objetos dinámicos desaparecen de la escena parcialmente, es decir, entre observaciones sucesivas se puede detectar objetos estáticos que no han desaparecido de la escena.

La realización de este trabajo permite contribuir en el campo de la detección de objetos en movimiento, enfocado a descubrir objetos dinámicos a partir de nubes de puntos obtenidas por un sensor LIDAR con escáner 2D (*2 Dimension*). Además, el sistema se combina con otros sensores como cámara RGB (*Red, Green, Blue*), GPS (*Global Positioning System*) e IMU (*Inertial Measurement Unit*), pudiendo realizar una transformación de los datos recogidos por el sensor láser a un sistema de coordenadas geodésicas mundial. En la actualidad existen numerosas investigaciones sobre la detección de objetos basadas en los sensores mencionados anteriormente, como se muestra en el artículo sobre detección de objetos con vehículos aéreos no tripulados UAVS (*Unmanned Aerial Vehicles*) pequeños dotados de un LIDAR, publicado por el Instituto Americano de Aeronáutica y Astronáutica [2].

La relevancia que puede tener realizar la investigación sobre detección de objetos en movimiento puede llegar desde la creación del dispositivo que realice dicha detección hasta el estudio de diferentes técnicas y mecanismos para conseguir dicho objetivo. Este documento plasma la evolución de la detección de objetos en movimiento y el gran potencial que tiene para el desarrollo y evolución en distintas aplicaciones.

1.2. Formulación del problema.

La detección de objetos en movimiento con sensor LIDAR es utilizado en muchos campos, gracias al escaneado del sensor láser se registran nubes de puntos detalladas de la escena, estas nubes de puntos son generadas con la distancia del sensor al objeto, obteniendo un error ínfimo. Sin embargo, en las nubes de puntos podemos encontrar ruido que dificulta la detección de los objetos en movimiento pudiendo detectar un movimiento donde no se ha producido. Una de las cuestiones que se tiene que tratar en la detección de objetos en movimiento es diferenciar entre el movimiento producido en los objetos que se encuentran en escena y el movimiento del observador.

La detección de objetos en movimiento en aplicaciones como la conducción autónoma, robótica, navegación y detección de obstáculos en UAVS, donde el desplazamiento del sistema es inevitable tienen un factor fundamental el posicionamiento absoluto, debido a que la nube de puntos obtenida por el LIDAR tiene un posicionamiento relativo.

Tener presentes estos problemas es importante para la realización de un sistema de detección de objetos en movimiento. Estos problemas son estudiados en la literatura.

1.3. Objetivos.

El presente trabajo plantea los siguientes objetivos:

I. Detectar objetos en movimiento con un sensor LIDAR.
   - Realizar una agrupación, segmentación de los datos obtenidos por el sensor láser, aplicando diferentes algoritmos.
   - Implementar un sistema apto para realizar una asociación de nubes de puntos y medir el nivel de error del sistema con el fin de ser evaluado.

II. Implementar un sistema capaz de recopilar los datos escaneados por un sensor LIDAR conectado a una Raspberry Pi.
   - Implementar un sistema de control para corregir el ángulo de escaneado.
   - Filtrar las medidas que registra el LIDAR.
   - Visualización en tiempo real de la nube de puntos generada.

III. Implementar un sistema computacional capaz de recoger en tiempo real los datos que están siendo obtenidos por el sensor láser, cámara, GPS y el módulo que integra acelerómetro, giróscopo, magnetómetro y barómetro, para posteriormente ser tratados en diferido.
   - Identificar y almacenar las imágenes pertenecientes a cada escena escaneada por el láser.
   - Obtener y almacenar la posición actual donde se está realizando la detección, junto a los datos que capta el módulo que integra acelerómetro, giróscopo, magnetómetro y barómetro.
   - Almacenar en un archivo todos los datos para poder ser tratados en diferido.

IV. Implementar un sistema computacional que sea capaz de trasladar la información registrada por un sensor LIDAR a un sistema de coordenadas geodésicas mundial WGS-84 (*World Geodetic System 1984*). A su vez, el sistema computacional será capaz de combinarlo con el segundo objetivo.
    - Redefinir las medidas del LIDAR a un sistema de referencia local ENU (*East North Up*), con el eje X apuntando al este, el eje Y apuntando al norte y el eje Z arriba.
    - Trasladar el sistema de referencia local ENU al sistema WGS84 a través de ECEF (*Earth-Centered, Earth-Fixed*).
    - Implementar un sistema computacional en tiempo real capaz de realizar simultáneamente la recogida de datos obtenidos por los diferentes sensores y realizar las transformaciones expuestas anteriormente.

V. Realizar un procesado de los datos recogidos por los diferentes sensores y llevar a cabo un análisis de los datos.
    - Implementar un sistema que permita visualizar las nubes de puntos 2D con su correspondiente imagen.
    - Implementar un sistema que facilite la visualización de los datos geodésicos WGS84 en un mapa.
    - Detectar el movimiento entre nubes de puntos adyacentes.

1.4. Estructura del documento.

En esta sección, se detalla la estructura establecida del presente trabajo sobre la detección de objetos en movimiento mediante un sensor LIDAR. La estructura del documento se divide en 6 capítulos distinguidos.

- Capítulo 1. Introducción: Se da comienzo con una introducción del trabajo, donde se expone la motivación de la realización de este proyecto. Prosigue con un apartado sobre la problemática existente en la detección de objetos en movimiento. Por último, se exponen los principales objetivos del proyecto pudiendo focalizar la manera de abordar el proyecto y los puntos que son tratados.

- Capítulo 2. Estado del arte: El documento continuo con un capítulo acerca del estado del arte, el cual se divide en varios apartados. Comienza con un apartado donde se realiza un recorrido de la historia de la detección de objetos, donde se exponen trabajos relacionados con la detección en movimiento y otras soluciones basadas en CV (*Computer Vision)*. Continúa destacando los tipos de sensores LIDAR y sus principales características, mostrando investigaciones relacionadas con el modelo del sensor utilizado en este proyecto. Prosigue con un apartado sobre principales algoritmos de *clustering,* segmentación y asociación de nubes de puntos que se utilizan en la literatura. Posteriormente, un apartado sobre la fusión de datos, se muestran las principales técnicas junto a proyectos y aplicaciones donde se realiza una fusión de datos multi-sensor con sensores semejantes a los utilizados en el proyecto. Este capítulo contiene un

apartado donde se indican los diferentes aspectos legislativos que afectan al desarrollo del presente trabajo, así como, los principales estándares y normas relacionadas. Finalmente, el capítulo concluye con un análisis del entorno socio-económico, evaluando cómo afecta la detección de objetos en movimiento en la sociedad y economía actual.

- Capítulo 3. Propuesta: En este capítulo se presenta la propuesta del trabajo sobre la detección de objetos en movimiento con sensor LIDAR. Comienza con un apartado donde se realiza una descripción detallada del sistema. Una vez descrito el sistema se exponen las tecnologías hardware y software utilizadas para el desarrollo del proyecto. Por último, se explica las conexiones del dispositivo diseñado.
- Capítulo 4. Experimentación y resultados obtenidos: En este capítulo se presenta una serie de experimentos y los resultados obtenidos en los mismos. Además, se presenta un sistema comparativo para la verificación del movimiento en el sistema.
- Capítulo 5. Conclusiones y trabajos futuros: Este capítulo presenta dos apartados distinguidos. En el primero, se describe los objetivos conseguidos que son explicados en el capítulo 1 y las conclusiones del desarrollo del proyecto. Posteriormente se muestra un apartado de líneas futuras del proyecto, proponiendo posibles mejoras.
- Capítulo 6. Análisis del proyecto: Para finalizar se presenta un capítulo sobre el análisis del proyecto, exponiendo la planificación inicial del trabajo y su presupuesto, donde se muestran las variaciones de tiempo y costes que se han teniendo en el desarrollo del proyecto, indicando el error cometido en la estimación inicial y su corrección para alcanzar la meta en el tiempo de entrega.

2. ESTADO DEL ARTE.

En este capítulo, se presenta la situación actual junto a la problemática a la que nos enfrentamos sobre la detección de objetos en movimiento. Durante las últimas décadas se han realizado continuas investigaciones y estudios, dando lugar a grandes avances en la actualidad. El análisis y detección de objetos en movimiento está muy relacionado con aplicaciones en tiempo real como la detección de obstáculos, navegación autónoma y la recopilación de datos que aporten información sobre objetos estáticos y dinámicos en una escena.

La utilización de sensores LIDAR para este propósito, han demostrado ser eficientes y precisos, logrando obtener nubes de puntos detalladas del entorno donde se realiza dicha detección. Sin embargo, la generación de ruido en las nubes de puntos puede dar soluciones erróneas, detectando un objeto en movimiento donde no lo hay. A diferencia de la toma de imágenes digitales con cámaras u otros sensores de imágenes que emplean un filtro óptico, LIDAR es un sensor muy susceptible a los objetos de muestreo, produciendo bastante incertidumbre sobre la escena donde se está realizando dicho escaneo. La resolución espacial en la dirección vertical de un LIDAR es una tarea problemática para obtener la forma del objeto, pudiendo solamente adquirir información a una determinada altura. Por el contrario, nos encontramos con el láser de barrido horizontal los cuales tienen la dificultad de llegar a medir los puntos que se encuentran ocluidos detrás de otros puntos, sumándose al movimiento y balanceo del escáner.

2.1. Historia de la detección de objetos.

En este apartado se presenta la trayectoria que ha sufrido la detección de objetos con el paso del tiempo, mostrando dos periodos que marcaron un hecho histórico en la detección de objetos, exponiendo técnicas basadas en visión por computador como otra variante a la detección de objetos en movimientos. Además, se muestra trabajos relacionados con otras técnicas basadas en sensor LIDAR cercanas a la dirección del presente trabajo.

La detección de objetos ha sido uno de los problemas más desafiantes de la visión por computador. Su evolución en las últimas décadas puede considerarse primordial en la historia del DL (*Deep Learning)*, destacando significativamente la evolución que ha tenido desde 1990.

Se podría diferenciar dos periodos históricos: “detección de objetos tradicional anterior al año 2014 (Pre-AlexNet)” y “detección de objetos basado en aprendizaje profundo posterior al año 2014 (Post-AlexNet)” como define Zhengxia Zou y Zhenwei Shi en el siguiente artículo [3].

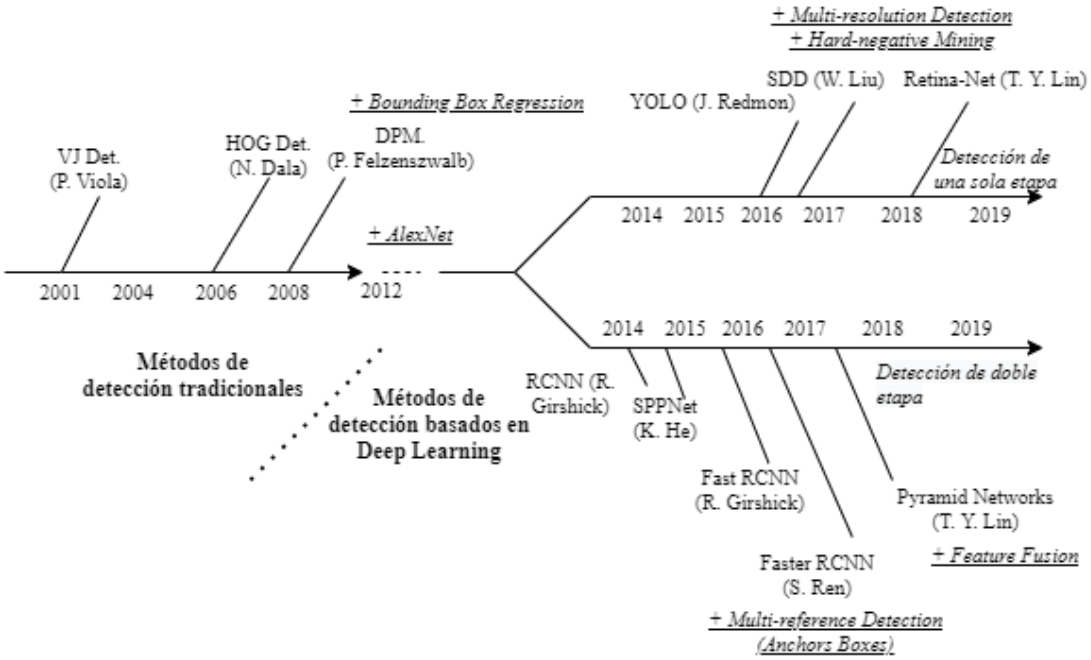


Fig. 2.1. Cronología de la detección de objetos.

Detección de objetos tradicional:

- **Detectores Viola Jones**, se remontan 18 años aproximadamente, consiguieron por primera vez detectar en tiempo real rostros, gracias a P. Viola y M. Jones, de ahí el nombre del algoritmo "*Detector Viola-Jones*". Este algoritmo está basado en funciones de Haar, donde las propiedades que se tienen que reconocer están incluidas en fragmentos rectangulares donde se realiza la suma de píxeles oscuros y claros.
- **Detector HOG**, "*Histogram of Oriented Gradients*" en 2005 se generalizó su uso por B. Triggs y Dalal, se centró en el problema de detección de peatones y posteriormente fue generalizándose para detectar diferentes clases de objetos. Este detector realiza un reescalado de la imagen de entrada varias veces calculando un descriptor HOG para cada escala, manteniendo el tamaño de entrada sin cambios.
- **DPM**, "*The Deformable Part-based Model*" en 2008 fue propuesto basándose en el detector HOG de B. Triggs y Dalal en el que se utiliza un filtro para representar una categoría de objeto. DMP está basado en un aprendizaje levemente supervisado, utilizando un modelo que consta de un filtro raíz y un conjunto de filtros piezas.

ImageNet en 2012, destacando la labor de AlexNet, provocó una evolución significativa debido a las redes de aprendizaje profundo, pudiendo aprender características más robustas y complejas de visión por computador.

Detección de objetos basado en aprendizaje profundo:

- **Detección de doble etapa**, la primera etapa se desarrollan las regiones de interés de los objetos, mientras que la segunda etapa se clasifican las regiones y se ubican los objetos mediante regresión de cuadro delimitador. Podemos destacar

R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN, FPM y SPP-net. Esta fase sigue siendo un modelo de referencia en la detección de objetos en el presente.

- **Detección de una sola etapa**, a diferencia de la anterior fase este detector obvia la etapa de desarrollar regiones, realizando directamente la clasificación y ubicación de los cuadros delimitadores. Cobró gran importancia con la propuesta de YOLO y posteriormente con la llegada de SSD y RetinaNet.

Estos dos periodos presentados están centrados en la detección de objetos con sistemas de visión por computador abriendo un abanico de posibilidades para abordar el problema planteado. Se puede afirmar que estos periodos han dado lugar a otras técnicas que no están centradas en la visión por computador. La aparición de otras técnicas también se debe a las limitaciones que tiene la visión por computador y las dificultades que presentan los sensores de filtro óptico, estos son sensibles a cambios de iluminación dificultando la detección en determinados momentos.

Un ejemplo de estas técnicas puede ser el método de detección de obstáculos con LIDAR 2D en un UAV como define Lanxiang Zheng en el siguiente artículo [4], utilizando diferentes sensores para complementar los datos del LIDAR y poder garantizar un vuelo seguro obteniendo información del entorno y los diferentes obstáculos que aparecen en el vuelo. Esta información sobre los obstáculos puede ser planteada como información global sobre los obstáculos o información local. LIDAR es uno de los sensores más útiles para la detección de obstáculos, debido a que no se ve tan afectado como otros sensores ópticos ante condiciones climatológicas adversas, cambios de luz o nubosidad. En este proyecto se utiliza un IMU y un GPS para transformar correctamente las nubes de puntos en un sistema de coordenadas global. Las tareas que se realizan para detectar obstáculos generalmente es obtener información con el sensor LIDAR 2D sobre los obstáculos presentes en el entorno de vuelo, procesar previamente las nubes que se están registrando y, por último, realizar un agrupamiento de las nubes de puntos. LIDAR tiene un movimiento extrínseco que es la razón por lo que las nubes de puntos pueden aparecer distorsionadas, para corregir este problema el autor plantea un método de compensación de la posición de los datos utilizando el algoritmo ICP (*Iterative Closest Point*) para encontrar los movimientos del LIDAR entre nubes de puntos adyacentes. Otra forma de resolver este problema es utilizar INS (*Inertial Navigation System*) para estimar la posición del LIDAR en el registro de cada punto. Con la ayuda de un IMU y un GPS se realiza la transformación de los puntos a una posición real en un sistema de coordenadas global. Por último, para el agrupamiento de nubes de puntos se utiliza el algoritmo DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) un método de agrupamiento euclidiano basado en la densidad de puntos, dividiendo los puntos en diferentes grupos pudiendo detectar obstáculos.

La navegación autónoma supone un desafío debido a que en el entorno pueden aparecer objetos móviles, por lo que los sistemas deben estar realizando continuamente estimaciones para configurar el espacio de navegación. En el trabajo de Julien Moras se define un método de caracterización del espacio conducible utilizando LIDAR e información de mapas georreferenciados [5]. En este artículo se presenta un sistema con

LIDAR para la caracterización del espacio conducible, otras técnicas basadas en sistemas de visión han sido estudiadas en diferentes proyectos como navegación urbana o en ADAS (*Advanced Driver Assistance Systems*) utilizando una visión frontal para conseguir su objetivo, pero estos sistemas de visión son sensibles a los cambios de iluminación por lo que en este proyecto se opta por sistemas basados en láser. El método que se presenta es la percepción del entorno con los datos obtenidos por un sensor LIDAR como sensor principal de percepción y detección de obstáculos junto a mapas digitales, el vehículo integra un LIDAR, GPS y un IMU, lo precisa que sea la información recibida de los mapas digitales depende de estos dos últimos sensores, consiguiendo un espacio de conducción limitado por la estimación de la información. Una de las principales tareas que se trata en este proyecto es la transformación del mapa a un sistema de coordenadas mundial WGS-84 donde el eje Z está apuntado hacia arriba de manera que se encuentra ortogonal a la superficie del elipsoide de referencia.

Como hemos podido observar en este apartado existen dos enfoques distintos para solucionar el problema de la detección de objetos en movimiento. Encontrando investigaciones relacionadas con la temática tratada en este apartado, brindando un abanico de soluciones para resolver este problema.

2.2. Sensor LIDAR.

LIDAR es un sensor basado en una tecnología láser, que permite conocer la distancia que existe entre el sensor y una superficie u objeto, utilizando un haz de luz láser pulsado. Cuando el láser impacta con un objeto, rebota y retrocede a la posición donde está situado el sensor LIDAR, consiguiendo calcular la distancia entre el sensor y el objeto o superficie con la diferencia del tiempo transcurrido entre la emisión del láser y la recepción del rebote, obteniendo gran precisión y un error reducido.

Este sensor tiene diferentes usos y aplicaciones en la actualidad como topografía de terrenos, inspección del tendido eléctrico, agricultura, minería, robótica y vehículos autónomos entre otras, pudiendo destacar un gran reconocimiento dentro de la detección de objetos en movimiento.

Se puede diferenciar dos clasificaciones distintas de los sensores LIDAR en la actualidad, en función del tipo de láser o del escaneado como se detalla en el siguiente artículo [6].

Dependiendo del tipo de láser podemos destacar:

- **LIDAR de pulsos**. Funciona emitiendo pulsos de luz láser y se utiliza para medir la distancia entre el sensor y la superficie, midiendo el tiempo que tarda el láser desde que se realiza la emisión hasta el momento que es recibido.
- **LIDAR de medición de fase**. Emite continuamente un haz láser y en el instante en el que el sensor recibe la señal calcula la diferencia de fase emitida y reflejada.

Según el tipo de escaneado podemos destacar:

- **Lineal**. Contiene un espejo que va rotando y desviando el haz láser. Generando una serie de líneas paralelas sobre la superficie. Una de las principales desventajas de este tipo de escaneado es que al estar girando el espejo continuamente en una sola dirección no asegura tener siempre mediciones.

- **Fibra óptica**. A través de una serie de pequeños espejos y con un cable de fibra óptica, el láser se va desviando a las fibras laterales que se montan alrededor del eje. Este mecanismo genera una huella con una serie de formas circulares solapadas entre sí. El principal inconveniente es que el ángulo de escaneo es mucho menor.
- **Elíptico**. El láser es desviado por dos espejos realizando un escaneado elíptico. Este escaneado aumenta la dificultad, debido a la utilización de dos medidores angulares.
- **Zigzag**. Contiene un espejo rotatorio en ambos sentidos produciendo de esta manera un escaneado en forma de zigzag. A diferencia del escaneado de líneas este siempre está realizando mediciones, pero en las zonas cercanas a los límites laterales la densidad de puntos que se recogen es mayor que en el centro.

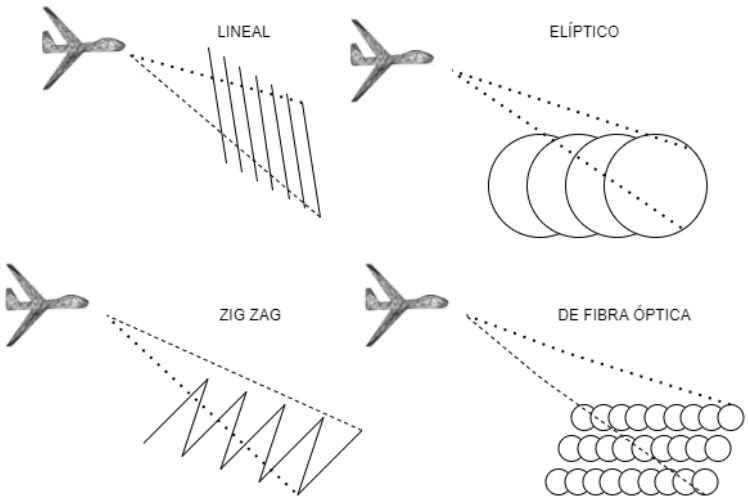


Fig. 2.2. Clasificación de láser según el tipo de escaneado.

El sensor láser utilizado en este proyecto es el modelo Hokuyo URG-04LX, es un LIDAR 2D de bajo costo diseñado para aplicaciones de robótica. Este sensor rotacional produce un solo haz a medida que va girando el espejo en su interior mientras se toman medidas con el desplazamiento de fase entre el haz láser emitido y su reflexión, obteniendo un alto volumen de datos producidos. El modelo de sensor utilizado es muy reconocido en el campo de la robótica debido a su reducido tamaño y las prestaciones que ofrece a pesar de su bajo coste. En el trabajo de Tapio Taipalus y Juhana Ahtiainen [7], se define la detección y seguimiento de personas de un robot móvil que integra el LIDAR 2D Hokuyo URG-04LX, resolviendo la problemática de detectar a personas en entornos concurridos.

Los fundamentos físicos que se involucran en este sensor LIDAR 2D son la interacción del haz del láser con la superficie y el proceso de medir la distancia como se define en el siguiente artículo [8]. Interacción del láser con la superficie, cuando el láser incide en un punto de la superficie puede ser transmitido, reflejado y dispersado, dependiendo del tipo de superficie y su ángulo. La distancia para el sensor utilizado se calcula con la diferencia de fase entre la luz transmitida desde el sensor y la reflejada por la superficie.

$$L = \frac{1}{2} * \left\{ \frac{\emptyset}{\left(\frac{F}{v} * 2\pi\right)} \right\} \quad (2.1)$$

L es la distancia a la superficie desde el sensor en milímetros, $\emptyset$ es la diferencia de fase en radianes, F la frecuencia en Hz y v la velocidad de la luz en mm/seg.

Podemos encontrar otros sensores ópticos que podrían sustituir el sensor láser que se plantea en este trabajo, pero encontramos algunas desventajas frente a este, como es la sensibilidad a los cambios de iluminación, como se ha comentado anteriormente.

2.3 Otros sensores.

Otros sensores que desempeñan una función en un sistema de detección de objetos en movimiento y complementan la información que obtiene el sensor LIDAR son GPS, barómetro, IMU. El sensor GPS proporciona la latitud y longitud del sistema en todo momento, el barómetro registra la altitud del sistema sobre el nivel del mar y un IMU que contiene (acelerómetro, giroscopio y magnetómetro), pudiendo realizar la transformación de la información registrada por el sensor LIDAR a un sistema de coordenadas geodésicas.

- **GPS**. Este sensor devuelve continuamente la posición del sistema basándose en el principio matemático de la trilateración, calculando la distancia al satélite en base al tiempo que tarda la señal en desplazarse. Los satélites envían señales de radio que contienen la información sobre su radio y el instante de tiempo en el que se envió, estas señales viajan a la velocidad de la luz. El sensor receptor registra el instante de tiempo que tardó en llegar y calcula la distancia con $d = c(t_2 - t_1)$ (2.2). El sensor necesita conocer la distancia a cuatro satélites para calcular la posición en la tierra en tres dimensiones.
- **Barómetro**. El sensor barométrico digital proporciona la altitud a la que se encuentra el dispositivo sobre el nivel del mar con precisión. Para el cálculo de la altitud se necesita la temperatura y la presión atmosférica, debido a que la presión desciende a medida que ganamos altitud. El sensor utilizado en este proyecto tiene un rango de medición de 300 hPa a 1110 hPa lo que equivale a una altitud de -500 metros a 9000 metros sobre el nivel del mar. Con una precisión de 1 hPa y un error de altitud en 1 metro de diferencia [6].
- **IMU**. La unidad de medición inercial utilizada en este trabajo dispone de tres sensores diferentes (acelerómetro, giroscopio y magnetómetro). El giroscopio mide la velocidad angular, que es el número de grados que se gira por segundo. El acelerómetro nos proporciona la aceleración lineal. Por último, el magnetómetro obtiene el campo magnético. Todos los sensores registran información en las tres componentes (x, y, z).

2.4. Algoritmia.

En este apartado se presentan algunos algoritmos que pueden ser aplicados como solución en el trabajo. Se estudian los principales algoritmos de *clustering*, segmentación y asociación de nubes de puntos aplicables a la información registrada por el sensor LIDAR.

2.4.1. Algoritmos de clustering.

El principal objetivo de los algoritmos de *clustering* es agrupar los objetos de un conjunto de datos según unas determinadas características y similitudes, normalmente basada en la distancia. Existen multitud de algoritmos de *clustering*, cada uno adaptándose mejor a una distribución de datos en particular, abordando diferentes problemáticas como la escalabilidad, atributos, dimensión, distancia, ruido, entre otros. Los distintos tipos de agrupación que se definen [9] en este articulo son:

- Agrupación basada en conectividad, es un tipo de agrupación que establece una jerarquía predefinida de clústeres, pudiendo realizar una descomposición de los datos en función de dicha jerarquía, obteniendo de esta manera los diferentes clústeres.

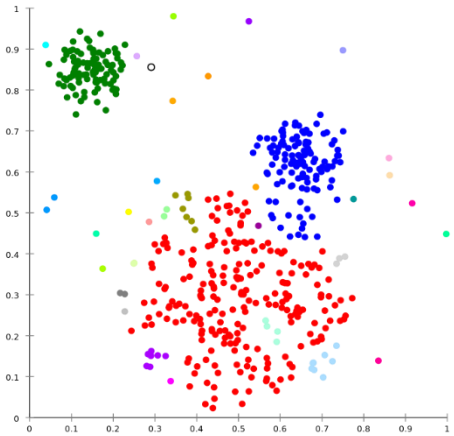


Fig. 2.3. Gráfica de agrupación basada en conectividad [10].

- Agrupación basada en centroides, es una de las técnicas más simples y eficaces de crear agrupaciones, se caracteriza por ser representadas por un vector central donde se realizarán respectivamente las agrupaciones según la cercanía a estos vectores.

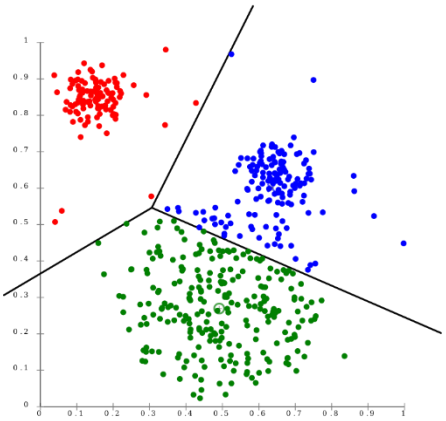


Fig. 2.4. Gráfica de agrupación basada en centroides [11].

- Agrupación basada en densidad se realiza la detección de las regiones que tienen mayor concentración de datos y se separan por áreas con escasos datos o nulos. Los datos o puntos que no están contenidos en la agrupación son etiquetados como ruido.

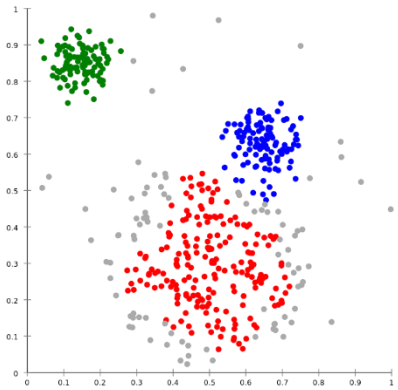


Fig. 2.5. Gráfica de agrupación basada en densidad [12].

- La agrupación basada en distribución, en este tipo de *clustering* se tiene en cuenta una métrica totalmente diferente como es la probabilidad, está estrechamente relacionada con la estadística, se agrupan conjuntos de datos en función de la probabilidad que existe de corresponder a dicha distribución de datos.

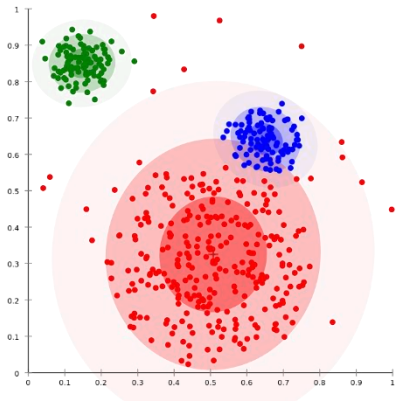


Fig. 2.6. Gráfica de agrupación basada en distribución [13].

- Agrupación difusa, los objetos pertenecen a un determinado grupo según el grado de confianza o pertenencia, este varía entre 0 y 1. Se utiliza con un conjunto de datos donde la variable tiene un alto nivel de superposición.

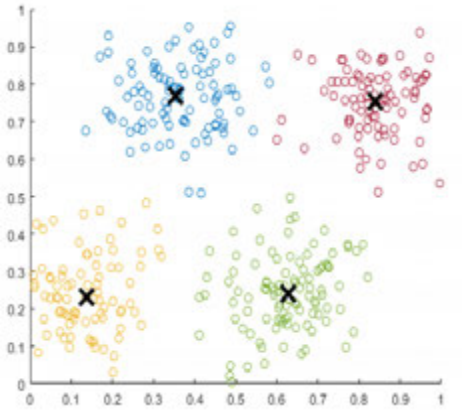


Fig. 2.7. Gráfica de agrupación difusa [14].

Una vez visto que es un algoritmo de agrupación y los diferentes tipos que existen, podemos observar que los algoritmos basados en densidad son los más oportunos para las nubes de puntos 2D recogidas por el sensor LIDAR. El *clustering* de nubes de puntos es esencial para la reconstrucción de escenas y detección e identificación de objetos. Los algoritmos basados en densidad son capaces de detectar agrupaciones de formas arbitrarias como los generados por el LIDAR. En el *clustering* de los datos obtenidos por el sensor láser podemos hacer uso de diferentes algoritmos de agrupación:

- **HDBSCAN** (*Hierarchical Density-Based Spatial Clustering Application with Noise)* es un algoritmo de agrupación que tiene la capacidad de detectar de manera arbitraria utilizando una técnica de *clustering* plano basada en la estabilidad del *clustering*, sin requerir tantos parámetros como su competidor más directo DBSCAN. El parámetro principal que necesita para efectuar el *clustering* es (*min_cluster_size*), indica el tamaño mínimo de un grupo. Este algoritmo hace uso de una variedad de distancias para realizar una separación en regiones de densidades variables al ruido. Mientras se realiza el procesamiento, este algoritmo crea una jerarquía de agrupaciones basadas en diferentes umbrales de

densidad, obteniendo el *cluster* plano. Por lo tanto, el número de *clusters* queda definido por las propias características de los datos.

- **OPTICS** (*Ordering Points To Identify the Clustering Structure*) es un algoritmo muy similar a DBSCAN buscando resolver las principales debilidades de este. Este algoritmo busca muestras centrales con alta densidad y a partir de estas muestras se forman las agrupaciones. Este algoritmo tiene una jerarquía de agrupaciones para un radio de vecinos variable. Los datos se ordenan linealmente de manera que los puntos más cercanos se convierten en vecinos, se almacena una distancia para cada punto del conjunto de datos, representando la distancia que se necesita para que los dos puntos pertenezcan al mismo grupo.
- **DENCLUE** (*DENsity-based CLUstEring*) es un algoritmo que identifica regiones de densidad en el conjunto de datos. Este algoritmo crea una red de regiones en el conjunto de datos usando una función de influencia, obteniendo puntos con el mismo máximo local, de esta manera se describen los datos que se encuentran en el mismo grupo. Utiliza un método que combina agrupaciones jerárquicas y particiones de datos.
- **DBSCAN**. Este algoritmo se utiliza en este proyecto para establecer diferentes grupos en las nubes de puntos, es un algoritmo de *clustering* basado en densidad cuyo funcionamiento es simple, un punto p forma un cluster de densidad con el punto q si p está dentro del radio de vecindad ε definida como $N_\varepsilon(\rho) = \{q | \, d(p,q) \leq \varepsilon\}$ y además tiene el mínimo de vecinos requeridos para formar un grupo $MinPts$. Un punto p es conectado directamente por densidad desde un punto q si $p \in N_\varepsilon(q)$ y $|N_\varepsilon(p)| \geq MinPts$. Definimos nuestro conjunto de puntos como $x = \{x_1, x_2 \ldots . x_n\}$, DBSCAN necesita dos parámetros, el radio de vecindad y el mínimo de vecinos para formar un grupo. Primero se escoge un punto arbitrario que no haya sido visitado, se obtiene la vecindad de ese punto con ε y se calcula si existen el número mínimo de vecinos en el radio de dicho punto, será etiquetado como escogido si cumplen estas condiciones, de lo contrario es etiquetado como ruido. Este proceso se repite hasta terminar de definir todos los grupos de datos o que todos los puntos hayan sido etiquetados. Los grupos etiquetados con "-1" son *outliers*.

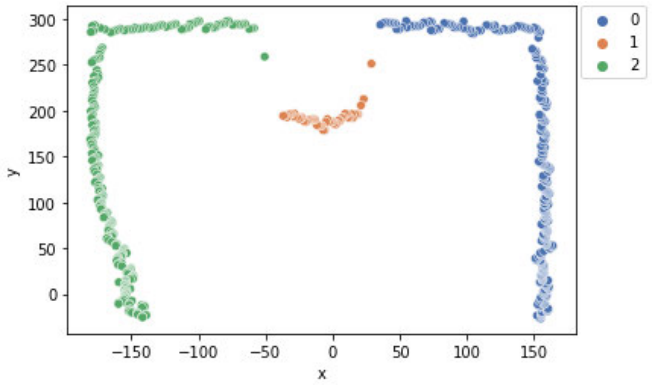


Fig. 2.8. Clustering de una nube de puntos 2D con DBSCAN.

En la Fig. 2.8 se observa una nube de puntos capturada por el dispositivo desarrollado. Esta nube de puntos ha sido agrupada por el algoritmo DBSCAN formando 3 agrupaciones distintas.

2.4.2. Segmentación.

Una vez realizada la técnica de *clustering*, se tiene la nube de punto agrupada según la densidad del conjunto de datos. Se procede a la segmentación de los grupos, estableciendo una homogeneidad entre los *clusters* adyacentes con el objetivo de poder realizar correctamente una asociación de las nubes de puntos, por tanto, si existe un desequilibrio en las diferentes agrupaciones se aplica un modelo predictivo para sintetizar puntos y obtener la misma dimensionalidad en ambos conjuntos, para ello se utiliza una técnica de sobre muestreo de minorías sintéticas SMOTE (*Synthetic Minority Oversampling Technique*). Esta técnica permite equilibrar el número de puntos de cada *cluster*, sintetizando nuevos puntos en el conjunto de datos desequilibrado, utilizando la interpolación lineal del grupo minoritario. Los pasos que se realizan en esta técnica son los siguientes [15]:

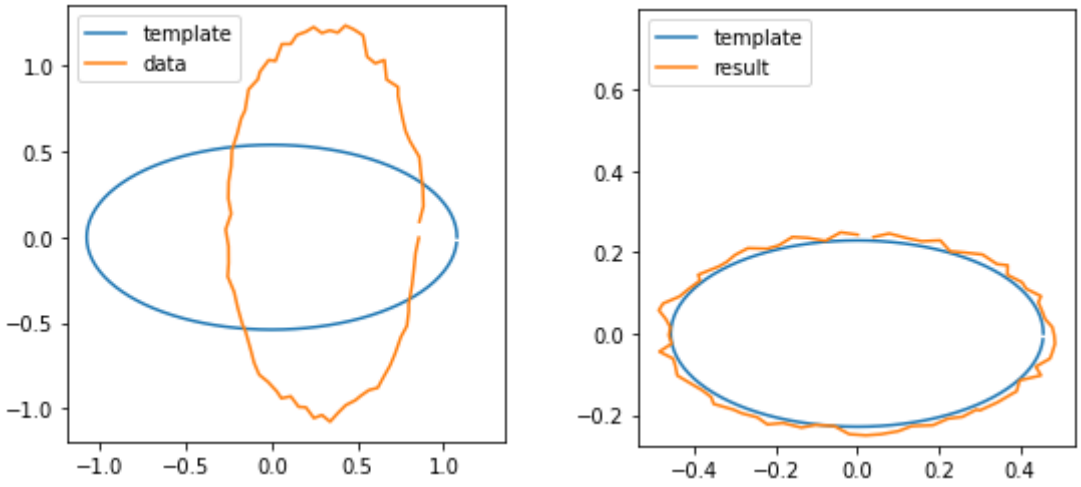
- Este método encuentra las distancias existentes entre el conjunto de datos mayoritario en el que se ha realizado un submuestreo y tiene un desequilibrio, utilizando la distancia euclídea.
- Se seleccionan los puntos k de la clase mayoritaria que tienen la menor distancia a la clase desequilibrada.
- Si hay n puntos en el grupo de la clase minoritaria, el resultado que se obtiene viene dado por la siguiente operación, $n \times k$ (2.3) puntos de la clase mayoritaria.

Esta técnica nos proporciona un correcto balanceado de los conjuntos de datos.

2.4.3. Algoritmo de asociación de nubes de puntos.

Una vez realizada el *clustering* y segmentación en las nubes de puntos es primordial realizar una asociación de los *clusters* obtenidos entre los pares de nubes de puntos adyacentes, con el objetivo de obtener la correlación entre las diferentes áreas de una escena. Un algoritmo que da solución a este problema es ICP. Este algoritmo iterativo

es utilizado para minimizar la diferencia entre dos nubes de puntos, obteniendo una correlación y mapeado óptimo. Este método se basa en aplicar una rotación y translación entre los pares de puntos más cercanos entre sí, utilizando la distancia euclídea. Este proceso es iterativo hasta conseguir minimizar el error.

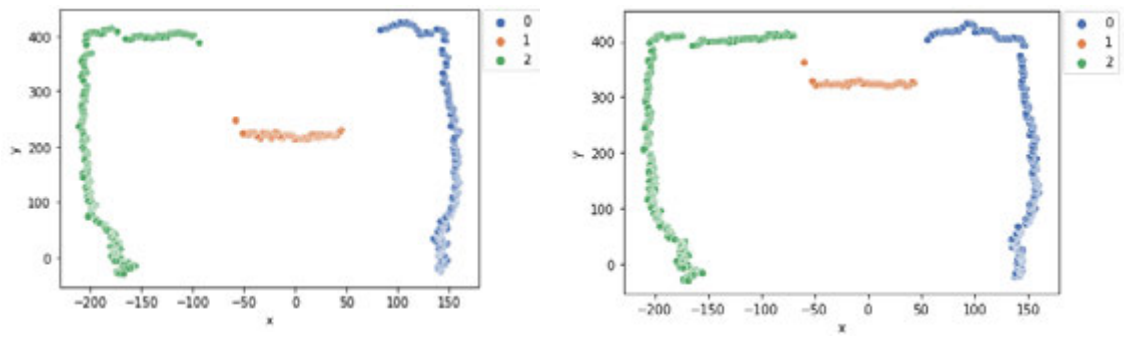


(a) (b)

Fig. 2.9. Representación gráfica de un conjunto de datos ajustado por el algoritmo ICP.

En la Fig. 2.9 se observa dos conjuntos de datos representados gráficamente, en el grafico (a) se muestra el conjunto dato el cual se tiene que ajustar al conjunto etiquetado como *template,* en el grafico (b) se muestra el resultado de aplicar el algoritmo ICP, después de realizar la rotación y traslación se ha obtenido un ajuste de dos conjuntos de datos, minimizando el error entre ellos.

El dispositivo registra nubes de puntos sucesivas de un entorno con el fin de ser tratados en diferido. Una vez realizado el *clustering* y la segmentación de los datos se procede a realizar una asociación de las nubes de puntos para obtener una correlación entre las distintas áreas de las escenas y observar variaciones en los objetos, detectando posibles movimientos.



(a) (b)

Fig. 2.10. *Clustering* aplicado a dos escenas adyacentes durante una secuencia en la que un objeto se aleja del sensor. a) Registro de datos en t-1. b) registro de datos en t.

En la Fig. 2.10 se puede observar dos escenas sucesivas donde el conjunto de puntos naranjas, etiquetados con un "1" representa un objeto que ha sufrido un desplazamiento alejándose del sensor.

El algoritmo de iteración de puntos más cercanos, se caracteriza por encontrar un mapeado óptimo entre dos nubes de puntos en base a una rotación y traslación que minimice la distancia entre los pares de puntos correspondientes de las nubes de puntos origen y destino. Los conjuntos de entrada son A (nube de puntos de origen) y B (nube de puntos de destino) con P_A y P_B puntos, ambos conjuntos tienen que tener la misma dimensión para poder aplicar este algoritmo. Los pasos del algoritmo ICP son los siguientes:

- Cálculo del punto más cercano de A para cada punto de B, utilizando la distancia euclídea $d_j = Min\left(\sqrt{B_j^2 - A_i^2}\right)$ (2.4) $\quad i = 1,2,3, \dots \dots, P_A$, los pares de puntos se eliminarán si la distancia obtenida $d_j > treshold$, por el contrario, los pares de puntos que la distancia sea menor pasarán a construir un conjunto, $M = C(A, B)$.
- Rotación y traslación del conjunto B y M, se calcula la matriz de rotación R y el vector de traslación T utilizando el método de mínimos cuadrados. Una vez se ha realizado la roto traslación, se calcula la transformación del conjunto de puntos de A utilizando la matriz de rotación R y el vector de traslación T, el vector de puntos de partida $A' = A \times (R|T)$ (2.5), siendo $(R|T)$ la matriz de roto traslación, es decir, $A' = RA + T$ (2.6).
- Cálculo del error de la rotación y traslación, al ser un algoritmo iterativo se calcula en la primera iteración el error, si el error es mayor que el umbral continuará iterando hasta que se minimice.

$$E = (R, T) = \sum_{i=1}^{P_B} \sum_{i=1}^{P_A} w_{ij} \| B_i - \left(R_{A_j} + T\right) \|^2 \qquad (2.7)$$

w_{ij} es la suma de los pesos correspondientes a cada par de puntos.

2.5. Fusión de datos multi-sensor.

La fusión de datos multi-sensor es el proceso que se realiza para combinar los datos recogidos de varios sensores distintos con el objetivo de facilitar una descripción completa y precisa de un entorno o proceso. En la actualidad se utiliza en diferentes áreas como, reconocimiento de objetos, mapeo del entorno, control de tráfico, robótica, automatización de vehículos, entre otros.

La fusión de datos es un área que involucra varios campos y es complicado establecer una clasificación clara y precisa. Por lo tanto, podemos dividir en diferentes técnicas y métodos de acuerdo a los siguientes criterios como define Federico Castanedo [16]:

- Según las relaciones existentes entre las fuentes de datos de entrada, como propuso Durrant-Whyte [17]. Estas relaciones pueden ser datos cooperativos, complementarios o redundantes como se observa en la Fig. 2.11.

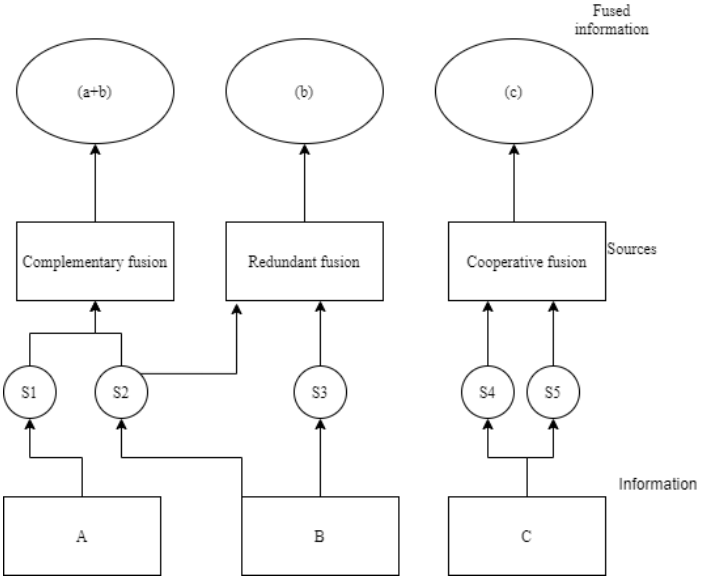


Fig. 2.11. Clasificación basada en las relaciones entre las fuentes de datos.

- Atendiendo a los tipos de datos de entrada / salida y sus características, como propone Dasarathy [18]. La clasificación está compuesta de 5 categorías diferentes como se puede ver en la Fig. 2.12.

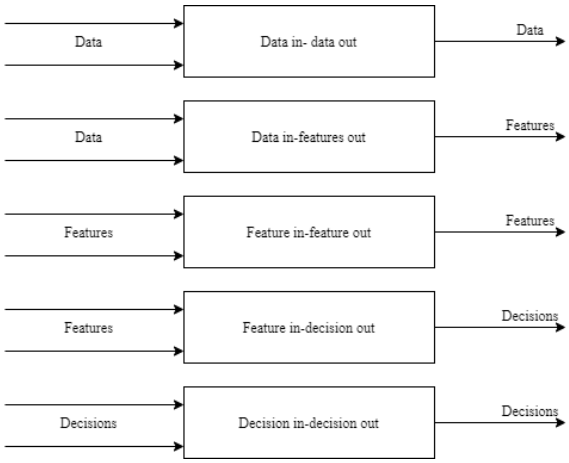


Fig. 2.12. Clasificación de Dasarathy.

- En función del nivel de abstracción de los datos que están siendo utilizados en la fusión de datos, como medidas, decisiones, señales o características. Esta clasificación atiende a la fusión de datos multi-sensor como propone Luo [19].
- Según los niveles de procesamiento de datos, fueron propuestos por Joint Directors of Laboratories [20]. La clasificación consta de 5 niveles diferentes como se puede observar en la Fig. 2.13.

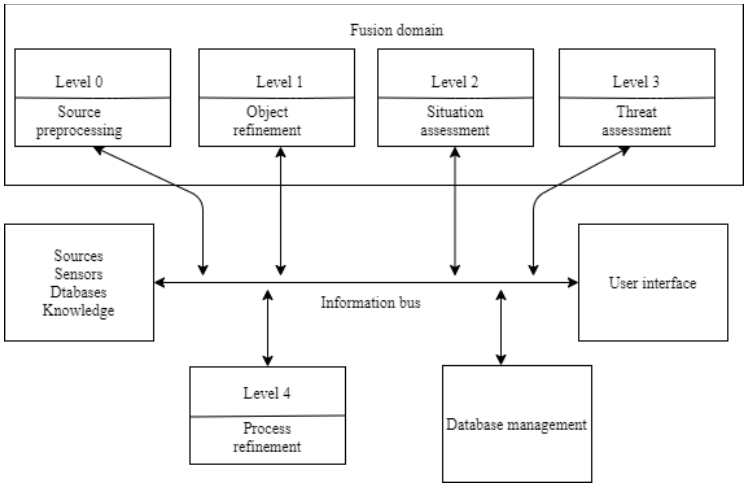


Fig. 2.13. Clasificación de fusión de datos JDL.

- Clasificación basada en el tipo de arquitectura, pudiendo destacar 3 grupos distintos, centralizada, distribuida y descentralizada, como se observa en la Fig. 2.14.

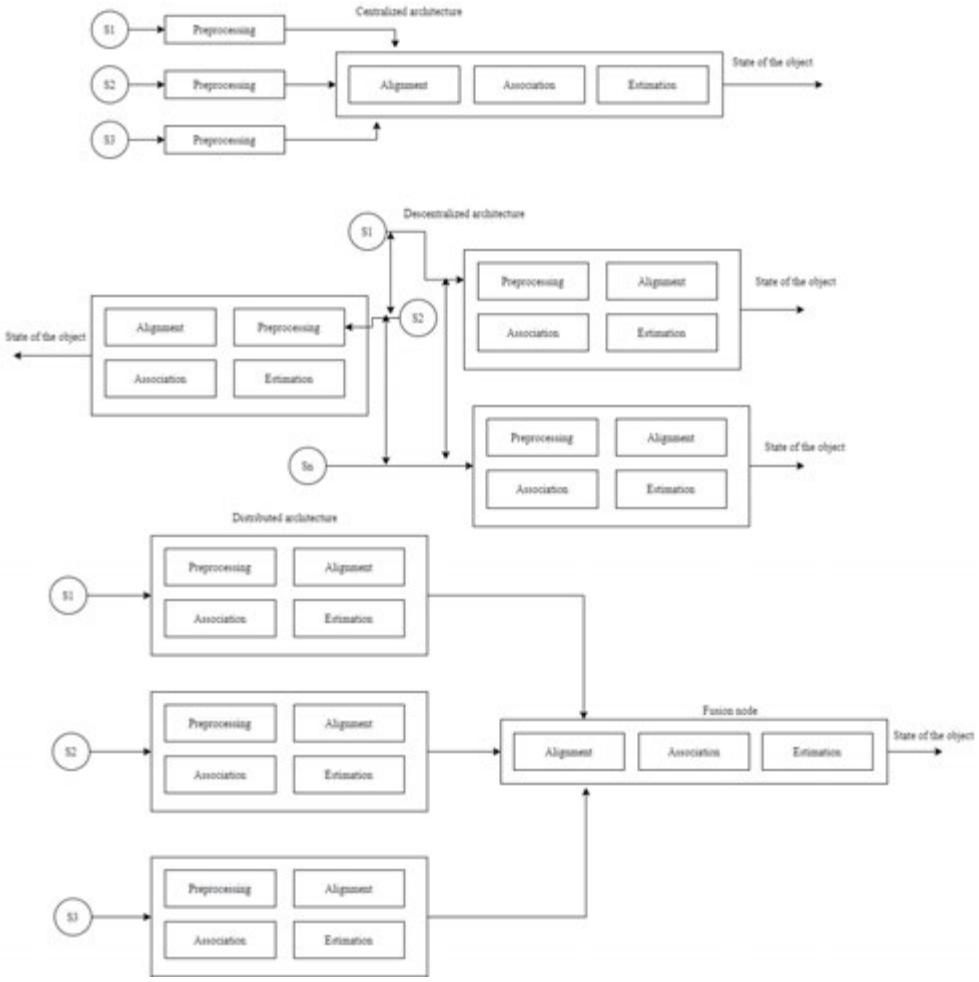


Fig. 2.14. Clasificación basada en el tipo de arquitectura.

El presente proyecto hace uso de 4 sensores distintos, LIDAR, cámara, GPS y un módulo GY-91 que integra acelerómetro, giróscopo, magnetómetro y barómetro. En la actualidad existen distintos centros de investigación y organismo que realizan estudios de la fusión de datos multi-sensor con estos mismos sensores, como es definido por los autores de este artículo [21], que utilizan la fusión de datos multi-sensor para estimar el estado 3D del UAV. Para realizar la estimación se utiliza una fusión de los datos recogidos por IMU, GPS, LIDAR, un sensor de flujo óptico y una cámara de profundidad consiguiendo obtener el estado del UAV en todo momento.

2.6. Marco regulador.

En este apartado se lleva a cabo un repaso de la legislación española sobre las regulaciones legales en las que se ve involucrada la realización de este TFG. En el periodo que se está redactando el presente documento, la legislación española no recoge ninguna regulación sobre la detección de objetos en movimiento con LIDAR. Debido a la ausencia de legislación al respecto, se va a realizar un repaso sobre los dispositivos utilizados para dicha detección.

El único dispositivo del trabajo que está involucrado dentro de unas regulaciones legales es la cámara, debido a que muchas pruebas son realizadas en el exterior. Por

tanto, hay que tener en cuenta la Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal, que dice textualmente: *"La presente Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar"* [22]. Para asegurar el derecho a la protección de datos, se necesita comunicar a las personas involucradas y requerir su consentimiento para poder realizar un tratamiento de sus datos.

En la realización de este proyecto se ha hecho uso de softwares y librerías de terceros, haciendo referencia a la Ley de Propiedad Intelectual [23]. El software está definido en el artículo 96, que dice textualmente: "*toda secuencia de instrucciones o indicaciones destinadas a ser utilizadas, directa o indirectamente, en un sistema informático para realizar una función o una tarea o para obtener un resultado determinado, cualquiera que fuere su forma de expresión y fijación*". Por lo tanto, todo software de terceros utilizado en este proyecto contiene una licencia que acepta su uso.

Una vez finalizados los aspectos legales del proyecto realizaremos un análisis sobre los principales estándares y normas que se aplican en el proyecto.

Representación estándar de la ubicación del punto geográfico por coordenadas, ISO 6709: 2008 (*International Standardization Organization*) [24]. Este estándar especifica la representación de coordenadas, entre ella la latitud, longitud y altura, incluyendo orden de coordenadas y unidades de medida.

La Raspberry Pi Model 3B+ cumple con los estándares IEEE (*Institute of Electrical and electronics Engineers*) 802.11 a/b/g/n/ac corresponde con la velocidad de transmisión de datos Wi-Fi (*Wireless Fidelity*) [25]. La letra de cada estándar corresponde con diferentes velocidades de transmisión de datos a distintas frecuencias.

2.7. Entorno socio-económico.

En este apartado se comenta el impacto que tiene la detección de objetos en la sociedad actual. Tras el transcurso de los años, todos los campos y aplicaciones en los que se ve involucrada la detección de objetos han ido evolucionando sirviendo de gran ayuda al ser humano y pasando desapercibido en numerosas ocasiones por gran parte de la sociedad. Un claro ejemplo del avance son los vehículos autónomos y la robótica que emplean técnicas de detección de objetos en movimiento y se puede afirmar que ha sido un paso crucial para dicha evolución.

En cuanto se refiere al sector económico y laboral, la detección de objetos en movimiento y su rápido avance se están aplicando en el sector de vehículos autónomos. Grandes empresas como Ford y Apple entre otras, se han lanzado en una carrera de investigación por conseguir una plataforma completamente autónoma e independiente capaz de detectar objetos tridimensionales en un entorno, consiguiendo una conducción totalmente autónoma. Otras marcas como DJI Automotive se centran en la investigación y producción de sistemas de conducción autónoma con años de experiencia en la detección de movimiento, mapeo y detección de obstáculos. Por lo que están apareciendo puestos de trabajo que requieren profesionales cualificados que se encarguen de implementar nuevos sistemas capaces de detectar objetos.

Otro sector donde la detección de objetos ha supuesto un gran desarrollo es la robótica, todo robot con movilidad necesita obtener información de su entorno para llevar a cabo determinadas tareas y objetivos. Empresas como Ecovacs Robotics se dedican al desarrollo y fabricación de robótica de servicio doméstico, hacen uso de tecnologías de detección y evitación de obstáculos.

Las tecnologías en las que la detección de objetos está involucrada han conseguido una evolución que hace unos años parecían inalcanzables, brindando nuevas posibilidades a la sociedad.

También se debería de hacer hincapié que al haber trabajos sustitutos como es el caso de la robótica o la conducción autónoma, existe la probabilidad de aumentar la tasa de destrucción de empleo una de sus principales desventajas. En cambio, tener un vehículo autónomo como vehículo público o un robot al cargo de un determinado trabajo es tener su disposición las 24 horas del día para realizar dicha labor, sin días de descanso. Sería necesario encontrar un equilibrio dentro de las empresas para paliar la destrucción de empleo, asegurando un puesto diferente para el empleado que sea sustituido.

Todos los sectores en los que la detección de objetos se ve implicada están ofreciendo nuevos empleos, pero también es inevitable tomar decisiones en un futuro ante el problema de la destrucción de empleo causado por sistemas autónomos. Algunas de las soluciones que ya han sido propuestas es el reentrenamiento de los trabajadores con el objetivo de capacitarlos para otros puestos de trabajo diferentes. Todo este desarrollo que se está planteando a la sociedad en la actualidad debe tener una base ética y es necesario educar en este aspecto, ofreciendo nuevas posibilidades de trabajo a la sociedad.

3. PROPUESTA.

A lo largo de este capítulo se presenta la propuesta que se plantea para solucionar el problema de la detección de objetos en movimiento con sensor LIDAR. Se propone diseñar y desarrollar un sistema con un conjunto de sensores de bajo coste conectados a un ordenador reducido (Raspberry Pi), con el objetivo de capturar información del entorno y detectar posibles movimientos en objetos. Este sistema recogerá información en tiempo real almacenándola con el fin de ser tratada en una primera aproximación en diferido. La información consta de nubes de puntos 2D generadas por el sensor LIDAR, latitud y longitud proporcionado por el GPS, una imagen capturada por una cámara que identifica la escena de detección, altitud recogida por el barómetro y por último aceleración, giro y campo magnético proporcionado por un acelerómetro giroscopio y magnetómetro. Las nubes de puntos 2D serán complementadas con los datos recogidos por los demás sensores, pudiendo realizar una fusión sensorial del entorno. Los datos del LIDAR serán transformados en un sistema de referencia geodésico (WGS-84) con la finalidad de representar la información como puntos sobre la superficie terrestre. Además, se procesan los datos mediante una agrupación, segmentación y asociación de las nubes de puntos 2D aplicando diferentes algoritmos con el fin de detectar objetos en movimiento en el conjunto de datos.

3.1. Descripción del sistema.

Tras la formulación de la propuesta planteada sobre la detección de objetos en movimiento, este apartado describe el sistema establecido como solución planteada de alto nivel. La Fig. 3.1 muestra un diagrama de cajas del sistema.

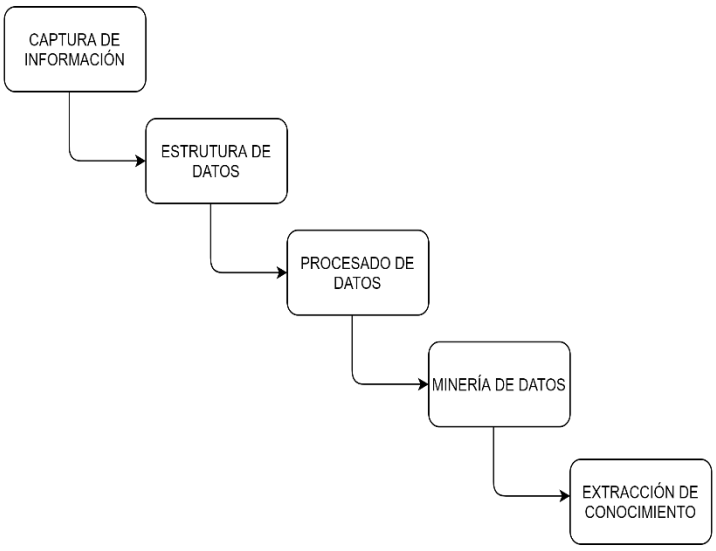


Fig. 3.1. Descripcion de alto nivel del sistema planteado.

Las diferentes secciones que se muestran en la anterior Fig. 3.1 son explicadas a continuación en los siguientes apartados.

3.1.1. Captura de información.

En el diseño e implementación del dispositivo se ha utilizado una Raspberry Pi 3. La codificación de esta Raspberry se ha realizado con el lenguaje de programación Python permitiendo controlar el conjunto de sensores y el registro de su información. El dispositivo lanza el programa indicando el ángulo de escaneado que se necesita para el entorno donde se encuentre, el dispositivo obtendrá información de los diferentes sensores y la almacenará en un archivo con extensión (.csv).

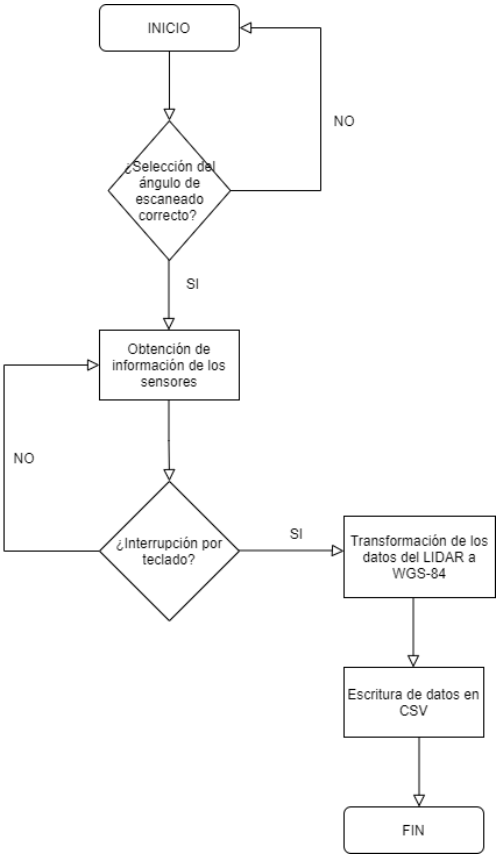


Fig. 3.2. Diagrama de flujo del proceso de captura de información.

El sistema antes de registrar la información realiza las transformaciones de los datos LIDAR al sistema de coordenadas geodésicas WGS-84. La información obtenida por el sensor LIDAR está referida sobre los ejes del sensor, es necesario rotarlo a un sistema de coordenadas locales con los ejes apuntando al este en el eje X, al norte en el eje Y, arriba en el eje Z (ENU). El sensor LIDAR realiza un escaneado en 2D obteniendo un conjunto de puntos (x,y), por lo tanto, la rotación de los ejes se realiza únicamente en el eje X e Y, la coordenada Z es representada por la altitud sobre el nivel del mar a la que se encuentra el dispositivo. Una vez realizada las rotaciones de los ejes en (ENU), se transforma los datos a un sistema de coordenadas geocéntricas (ECEF), desempeñando una función de marco intermediario entre el marco cartesiano local y WGS-84, de esta manera se podrá expresar los datos obtenidos en posición con respecto al elipsoide de referencia (sistema geodésico mundial).

Para la transformación de los datos del LIDAR se utiliza un GPS y un módulo que integra diferentes sensores (acelerómetro, giróscopo, magnetómetro y barómetro). La transformación de los datos consta de diferentes pasos tal y como definen los autores de este artículo [26], estos son los siguientes:

- Rotación de los ejes del LIDAR a un marco local ENU, el sensor LIDAR obtiene un conjunto de datos en 2D referidos sobre sus propios ejes con las coordenadas (x_{LIDAR}, y_{LIDAR}), como se puede observar en la Fig. 3.3. Se necesita rotar los ejes X e Y del LIDAR para obtener un sistema de referencia local (ENU).

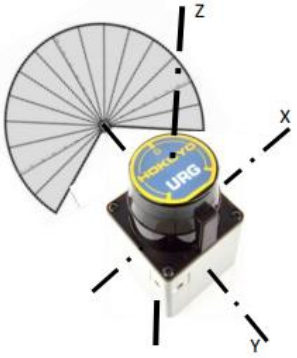


Fig. 3.3. Ejes del sensor LIDAR.

Se parte de las coordenadas $(x_{LIDAR}, y_{LIDAR}, z_{ALTITUDE})$ para realizar la rotación.

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = (R) \times \begin{pmatrix} x_L \\ y_L \\ z_A \\ 1 \end{pmatrix} \tag{3.1}$$

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\,(\sigma) & -sin\,(\sigma) & 0 \\ 0 & sin\,(\sigma) & cos\,(\sigma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_y = \begin{pmatrix} cos\,(\sigma) & 0 & sin\,(\sigma) & 0 \\ 0 & 1 & 0 & 0 \\ -sin\,(\sigma) & 0 & cos\,(\sigma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

$$R_z = \begin{pmatrix} cos\,(\sigma) & -sin\,(\sigma) & 0 & 0 \\ sin\,(\sigma) & cos\,(\sigma) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Primero se realiza una rotación sobre el eje x:

$$\begin{aligned} x'_x &= x_L \\ y'_x &= y_L cos\,(\sigma) - z_A sin\,(\sigma) \\ z'_x &= y_L sin\,(\sigma) + z_A cos\,(\sigma) \end{aligned} \tag{2.11}$$

Por último, se rota sobre el eje y:

$$\begin{aligned} x'_{xy} &= x'_x cos\,(\sigma) + z'_x sin\,(\sigma) \\ y'_{xy} &= y'_x \\ z'_{xy} &= -x'_x sin\,(\sigma) + z'_x cos\,(\sigma) \end{aligned} \tag{3.3}$$

Las coordenadas $(x'_{xy}, y'_{xy}, z'_{xy})$ se encuentran en un sistema de referencia local (ENU).

- Transformación cartesiana local a ECEF, este sistema de coordenadas geocéntricas se utiliza como un marco intermedio en el sistema cartesiano local y WGS-84, definiendo la latitud y altura geodésica con respecto al plano tangente local. Las coordenadas locales (ENU) $(x'_{xy}, y'_{xy}, z'_{xy})$ obtenidas anteriormente están referidas al origen del plano (Lat, Lon, Alt) se transforman a coordenadas globales. En la transformación se hace uso de los vectores unitario del sistema de referencia local ENU, $\vec{x}_i, \vec{y}_i, \vec{z}_i$, se transforma en las coordenadas ECEF:

$$\begin{aligned} \vec{x}_i &= \begin{bmatrix} -sin(Lon) \\ \cos(Lon) \\ 0 \end{bmatrix} \\ \vec{y}_i &= \begin{bmatrix} -sin(Lat)\cos(Lon) \\ sin(Lat)sin(Lon) \\ cos\,(Lat) \end{bmatrix} \\ \vec{z}_i &= \begin{bmatrix} \cos(Lat)\cos(Lon) \\ \cos(Lat)\sin(Lon) \\ sin\,(Lat) \end{bmatrix} \end{aligned} \tag{3.4}$$

Una vez calculados los vectores unitarios, se necesita calcular la curvatura del elipsoide, en la latitud geodésica del origen local:

$$R_{E_{orig}}(Lat) = \frac{R_{max}}{(1 - ecc^2 sin^2(Lat))^{\frac{1}{2}}} \tag{3.5}$$

$Rmax$ es el radio ecuatorial que son 6378137 metros, Rp es el radio polar y la excentricidad es definida como:

$$ecc = \sqrt{-\frac{R_p^2}{R_{max}} + 1} \tag{3.6}$$

Expresamos el origen local en coordenadas globales:

$$orig \rightarrow_{ECEF} = \begin{bmatrix} R_{E_{orig}}\cos(Lon) \\ R_{E_{orig}}\sin(Lon) \\ R_{E_{orig}}(1 - ecc^2)sin(Lat) \end{bmatrix} + Alt \times \bar{z}_i \tag{3.7}$$

Por último, se calcula la posición de ECEF sumando las coordenadas locales al origen:

$$pos_{ECEF} = \begin{bmatrix} x_{ECEF} \\ y_{ECEF} \\ z_{ECEF} \end{bmatrix} = orig \rightarrow_{ECEF} + x'_{xy}\vec{x}_i + y'_{xy}\vec{y}_i + z'_{xy}\vec{z}_i \tag{3.8}$$

- Conversión de ECEF a coordenadas geodésicas, para la conversión de ECEF a coordenadas geodésicas es necesario realizar una aproximación para evitar un proceso iterativo calculando la transformación, debido a que sobre el plano de la tangente local es definida la altura geodésica, al mismo tiempo el punto tangente depende del eje local vertical. Por tanto, la latitud geodésica se aproxima como una corrección sobre la latitud y longitud geocéntrica:

$$R = \sqrt{(x_{ECEF})^2 + (y_{ECEF})^2 + (z_{ECEF})^2}$$
$$D = \sqrt{(x_{ECEF})^2 + (y_{ECEF})^2} Lat_{geoc} = a\frac{z_{ECEF}}{D}) \tag{3.9}$$

La latitud geodésica se aproxima con los siguientes términos:

$$f = 1 - ecc^2 x_a = \frac{(1-f)R_{max}}{(tan^2(Lat) + (1-f)^2)^{\frac{1}{2}}}$$
$$y_a = (1-f)\left({R_{max}}^2 - {x_a}^2\right)^{\frac{1}{2}}$$
$$\mu_a = a\ tan\frac{\left({R_{max}}^2 - {x_a}^2\right)^{\frac{1}{2}}}{(1-f)\ x_a}$$
$$r_a = \frac{x_a}{cos\ cos\ (\ Lat)}$$
$$l = R - r_a$$
$$\delta\lambda = \mu_a - Lat_{geoc}$$
$$h = l\ cos\ (\ \delta\lambda)$$
$$\rho_a = \frac{(1-f)R_{max}}{(1.0 - (2f - f^2)sin^2(\mu_a))^{\frac{1}{2}}}$$
$$Lat = \mu_a - a\ tan\ \frac{lsin(\delta\lambda)}{\rho_a + h} \tag{3.10}$$

Para finalizar, la altura y la latitud se calculan utilizando la curvatura local del elipsoide:

$$R_E = \frac{R_{max}}{(1.0 - ecc^2 sin^2(Lat))^{\frac{1}{2}}}$$
$$Alt = \frac{D}{cos\ (\ Lat)} - R_E \tag{3.11}$$
$$lon = atan\ \frac{y_{ECEF}}{x_{ECEF}}$$

Una vez finalizados los pasos se habrán transformado los datos del LIDAR en posición con respecto al elipsoide de referencia (sistema geodésico mundial). En la Fig. 3.4 se observa esta representación.

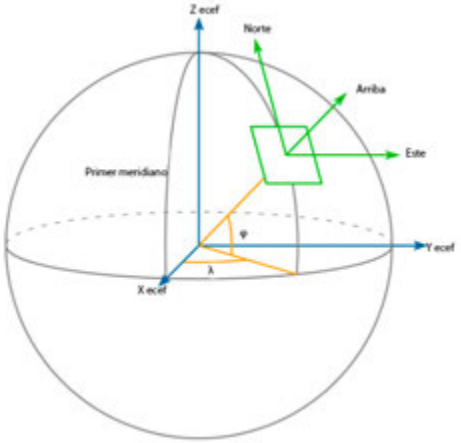


Fig. 3.4. Representación de ENU, ECEF y WGS-84.

3.1.2. Estructura de datos.

El registro de la información capturada por los diferentes sensores es almacenado en dos carpetas, una carpeta con el nombre CSV (*Comma-Separated Values*) contiene los csv con el siguiente nombre "2D_DataX.csv" donde x es el número del csv, cada escaneado se almacena en un csv con el número sucesiva al último que se encuentra en la carpeta. En segundo lugar, se encuentra una carpeta con el nombre "image" que contiene las imágenes capturadas del entorno donde se realiza el escaneado, las imágenes se almacenan siguiendo un procedimiento similar al de los csv, el nombre con el que se almacenan es "my_imageX.jpg" donde x es el número que identifica a cada imagen, este es el sucesor al último que se encuentre en la carpeta. Finalmente, el .csv consta de 21 atributos que se describen de manera resumida a continuación:

- **Schedule**: identifica la fecha y hora a la que se ha realizado cada iteración de escaneado.
- **Time**: identifica la iteración de escaneado.
- **X**: es la distancia en el eje X a la que se encuentra el objeto con respecto al sensor, la unidad se encuentra en mm.
- **Y**: es la distancia en el eje Y a la que se encuentra el objeto con respecto al sensor, la unidad se encuentra en mm.
- **Latitude**: identifica la latitud en la que se encuentra el dispositivo.
- **Longitude**: identifica la longitud a la que se encuentra el dispositivo.
- **Image**: en este atributo se almacena el nombre de la imagen correspondiente a cada iteración.
- **Gyro_X**: este atributo contiene la velocidad angular en la componente X.
- **Gyro_Y**: este atributo contiene la velocidad angular en la componente Y.
- **Gyro_Z**: este atributo contiene la velocidad angular en la componente Z.
- **Mag_X**: este atributo identifica el campo magnético en la componente X.

- **Mag_Y**: este atributo identifica el campo magnético en la componente Y.
- **Mag_Z**: este atributo identifica el campo magnético en la componente Z.
- **Accel_X**: identifica la aceleración lineal en la componente X.
- **Accel_Y**: identifica la aceleración lineal en la componente Y.
- **Accel_Z**: identifica la aceleración lineal en la componente Z.
- **AltitudeBaro**: este atributo contiene la altitud que nos proporciona el barómetro.
- **AltitudeGPS**: este atributo contiene la altitud que nos proporciona el GPS.
- **WGS84_X**: este atributo contiene la transformación del dato que contiene el atributo X al sistema de coordenadas geodésicas WGS84.
- **WGS84_Y**: este atributo contiene la transformación del dato que contiene el atributo Y al sistema de coordenadas geodésicas WGS84.
- **WGS84_Z**: este atributo contiene la altitud que proporciona el barómetro obteniendo el eje Z del sistema de coordenadas geodésicas WGS84.

3.1.3. Procesado de datos.

El procesado de datos que se realiza en el sistema comienza con una etapa de preparación de los datos donde se buscan errores y descarte de información incompleta. El dispositivo de detección finaliza la ejecución mediante una interrupción por teclado, lo que puede producir que la última iteración contenga datos incompletos o no se capture correctamente la imagen pertinente que identifica la escena de escaneado. Si los datos contienen algún error o están incompletos serán eliminados dejando la muestra correctamente. Los datos se almacenan en distintos *DataFrames,* estos son estructuras de datos que permiten guardar información de distinto tipo con su correcta dimensión para la tarea concreta que se realiza, estos son los siguientes:

- **Data_Clustering**, este *DataFrame* almacena los datos que van a ser agrupados mediante el algoritmo de agrupamiento. Contiene únicamente los atributos Time para identificar la iteración del escaneado y los puntos en dos dimensiones registrados por el sensor LIDAR, se encuentran en los atributos X e Y, de esta manera se genera un conjunto de datos con menos atributos facilitando la aplicación del algoritmo de clustering.
- **Data_Segmentation**, este *DataFrame* almacena los datos que van a ser segmentados. Contiene los atributos X e Y (los puntos) y un atributo *Label* que identifica el grupo al que pertenece cada punto. De esta manera podremos aplicar la técnica de sobre muestreo de minorías sintéticas entre los grupos pertenecientes al atributo *Label*.
- **Data_WGS84**, este *DataFrame* almacena los datos para la representación de las coordenadas geodésicas en un mapa. Contiene el atributo Time para identificar la iteración y los atributos WGS84_X, WGS84_Y, WGS84_Z, de esta manera se genera un conjunto más sencillo para representar los puntos en el mapa.

3.1.4. Minería de datos.

La minería de datos es el proceso de encontrar correlaciones y patrones en el conjunto de datos con el objetivo de poder extraer información. Se han utilizado

diferentes técnicas en los datos que han sido registrados previamente. La primera técnica utilizada en el conjunto de datos es el *clustering*, mediante el algoritmo DBSCAN, un algoritmo de agrupamiento basado en densidad con el que encontramos diferentes regiones en las nubes de puntos registradas en base a la distancia euclídea entre pares de puntos. DBSCAN es un algoritmo de ML (*Machine Learning)* no supervisado que consigue generar de manera automática las diferentes agrupaciones de datos. Este algoritmo utiliza dos parámetros que son importantes para su correcta agrupación ($MinPts$ y ε).

En la selección del valor $MinPts$ no hay un método que determine el valor preciso para un determinado conjunto de datos, para conjunto de puntos bidimensionales se utiliza el valor predeterminado de DBSCAN ($MinPts = 4$), este es el valor que hemos utilizado para todas las agrupaciones realizadas en este trabajo.

El valor de épsilon puede ser distinto para cada conjunto de datos, debido a que las distancias son variables dependiendo de la escena donde se realice el escaneado. Se ha utilizado el "método del codo", esta técnica calcula la distancia media entre cada punto y sus vecinos más cercanos, donde $n = MinPts$, las n-distancias se muestran en una gráfica de menor a mayor, orden ascendente, encontrando el valor de ε en el punto de curvatura máxima. La Fig. 3.5 muestra una prueba realizada para un conjunto de datos obtenidos por el sensor LIDAR, donde se puede observar que el punto máximo de curvatura está en torno a 21, en concreto $\varepsilon = 21{,}09$.

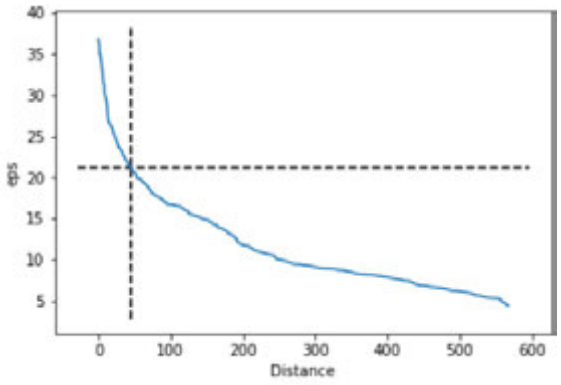


Fig. 3.5. Representación gráfica del método del codo para obtener épsilon.

La segunda técnica de minería de datos utilizada es SMOTE una técnica de sobre muestreo de minorías sintéticas, al tratar con un algoritmo automático da lugar a conjuntos de datos desbalanceados lo que dificulta la asociación de nubes de puntos. El algoritmo ICP busca la correlación entre pares de puntos, por lo tanto, únicamente se puede aplicar con conjuntos de puntos balanceados. SMOTE permite establecer un equilibrio balanceando los conjuntos minoritarios.

La tercera técnica utilizada es ICP para establecer una asociación entre los conjuntos de *clusters* definidos por DBSCAN sobre las nubes de puntos adyacentes registradas por el dispositivo. Con este algoritmo iterativo conseguimos una asociación punto a punto minimizando la diferencia entre dos nubes de puntos.

Por último, se calcula el RMSE (*Root-Mean-Square Error*) de los *clusters* pertenecientes a A y B. Se realizarán dos cálculos diferentes, primero se calcula el RMSE con el conjunto de datos sin aplicar el algoritmo ICP, de esta manera verificaremos si existe algún movimiento en las nubes de puntos sucesivas, si no existe movimiento el RMSE de todos los *clusters* será de la misma magnitud, de lo contrario podríamos verificar que existe un movimiento. Posteriormente, se aplica el algoritmo ICP encontrando un mapeado óptimo entre las nubes de puntos y se realiza un segundo cálculo del RMSE entre el conjunto de entrada y el resultado de aplicar la roto traslación, pudiendo observar que los RMSE se han aproximado hasta el punto de pertenecer a la misma magnitud en todos los conjuntos de puntos. De esta manera se verifica que cuando hay un movimiento detectado en el primer cálculo del RMSE, reafirmamos con este algoritmo que existe un movimiento, debido a que al aplicar la roto traslación los errores se reducen.

$$RMSE = \sqrt{\sum \frac{(y' - y)^2}{n}} \tag{3.1}$$

Este conjunto de técnicas permite obtener información del escenario, detectando el movimiento de los objetos.

3.1.5. Extracción de conocimiento.

Una vez se ha realizado la minería de datos podemos dar paso a la extracción de conocimiento de las diferentes técnicas realizadas sobre el conjunto de datos. En este proceso vamos a obtener información de cada una de las técnicas pudiendo detectar el movimiento en las diferentes escenas registradas. En las nubes de puntos 2D se puede apreciar la forma del objeto, en los bordes de un objeto los puntos están menos concentrados y según la distancia a la que se encuentre el objeto respecto al sensor LIDAR se observan diferentes concentraciones de puntos.

El algoritmo de *clustering* DBSCAN proporciona información de los objetos, agrupando diferentes conjuntos de puntos, de esta manera se obtienen las regiones que equivalen a los objetos puestos en escena junto a los *outliers* que se producen en la captura de datos. Las diferentes regiones pertenecientes a un objeto se pueden observar en la imagen capturada por la cámara RGB que integra el dispositivo. Además, se puede detectar un tipo de movimiento, si el número de *clusters* se ha reducido es un indicio de que un objeto ha salido de la escena lo que indica que se ha producido un movimiento, por el contrario, si se tiene una escena A con un número determinado de agrupaciones y en la escena sucesiva B el número de agrupaciones ha aumentado respecto A, se puede afirmar la entrada de un nuevo objeto en escena, por lo que podemos que en el entorno se ha producido un movimiento. Con la segunda técnica (SMOTE) se observa el desequilibrio en las diferentes agrupaciones entre dos escenas sucesivas, el conjunto minoritario puede denotar que ha ocurrido un movimiento, ya sea del propio dispositivo o del objeto que está en escena. En la tercera técnica (algoritmo ICP) podemos extraer diferente información, primero aplicando la rotación, con el ángulo que nos proporciona

el IMU podemos determinar la rotación que ha sufrido el dispositivo, si la rotación fuera insignificante determinamos que el dispositivo no se ha movido. La traslación que efectúa el algoritmo iterativo determina la distancia que se han desplazado los *clusters* adyacentes de las nubes de puntos, lo que indica el movimiento que ha sufrido un objeto en la escena. Por último, con esta técnica se realiza el cálculo del RMSE, si realizamos una comparación de los errores en las regiones donde no hay movimiento el error debe ser menor que las regiones que existe un desplazamiento, una vez se aplica el algoritmo ICP volvemos a calcular los RMSE, donde se puede observar que las regiones que han sufrido un desplazamiento tienen un error similar a las regiones donde no existe movimiento.

3.2. Tecnologías empleadas.

Una vez descrito el sistema propuesto, se expone las tecnologías hardware y software que se utilizan para la implementación y desarrollo del sistema de detección de objetos en movimiento.

3.2.1. Hardware.

Los componentes que forman el dispositivo de detección de objetos en movimiento son los siguientes:

- Raspberry Pi 3 Model B+, es un ordenador de placa reducida con unas dimensiones de 82 mm x 56 mm x 19,5 mm, utilizado como controlador de los diferentes sensores. Este modelo de Raspberry integra un procesador de 64 bits con 4 núcleos, bluetooth, Wi-Fi y puerto Ethernet. Se ha integrado una micro SD de 64GB donde se instala el sistema operativo, brindando suficiente capacidad para realizar la captura de datos de los sensores conectados a la Raspberry Pi.
- Sensor LIDAR Hokuyo URG-04LX, es uno de los LRF de corto alcance más populares en aplicaciones de robótica y navegación autónoma. Destaca por su peso y dimensión reducida con un rango máximo de medición de 4 metros. Este sensor cuenta con una precisión de $\pm$10 mm y un ángulo máximo de 240º. El LIDAR nos proporcionará nubes de puntos 2D de la escena donde se realice el escaneado. Dependiendo del ángulo de escaneado y la distancia a la que se encuentre la superficie las densidades de puntos varían.
- Módulo GPS NEO6MV2, es un módulo GPS de tamaño reducido 36 mm x 25.9 mm que conecta una antena devolviendo en tiempo real la posición del dispositivo.
- Módulo Pi Camera V2.1, es una cámara de dimensión reducida 25mm x 23mm x 9mm, ofrece una resolución nativa de 8MP y una resolución de imagen fija de 3280 x 2464. Esta cámara captura imágenes de la escena donde se realiza dicha detección.
- Módulo GY-91 10 DOF, este módulo integra dos módulos MPU9250 y el barómetro BMP280. Combina un giroscopio, acelerómetro, magnetómetro y barómetro en el mismo chip con una dimensión de 14,3mm x 20,5 mm, pudiendo

realizar las transformaciones de las nubes de puntos 2D obtenidas con el sensor LIDAR.

- Ordenadores, se han utilizado dos ordenadores para el desarrollo de este proyecto. Un ordenador portátil utilizado como ordenador cliente para el control remoto de la Raspberry Pi, de esta manera se puede desplazar para ejecutar el escaneado en diferentes escenarios. Un segundo ordenador de sobremesa donde se ha realizado el tratamiento de los datos y el desarrollo del documento.

3.2.2. Software.

En este apartado se muestra el *software* y *frameworks* empleados, se ha utilizado principalmente el sistema operativo Raspberry Pi OS para la programación y recogida de datos. Además, se ha utilizado Windows 10 pro instalados en un portátil y ordenador de sobremesa. En todos los equipos, se ha programado utilizando el lenguaje Python, utilizando las siguientes librerías:

- **NumPy**: Se utiliza para crear arrays y matrices de una manera más sencilla.
- **Pandas**: Librería para facilitar la creación de gráficos.
- **Seaborn**: Biblioteca para la visualización de gráficos estadísticos.
- **Matplotlib.pyplot**: Librería para la creación de gráficos en dos dimensiones.
- **Scikit-learn**: Sirve para aplicar ciertos algoritmos de inteligencia artificial de manera rápida y sencilla.
- **Hokuyo**: Librería de desarrollo del sensor láser URG-04LX-UG01.
- **Sertial_port**: Se utiliza para proporcionar acceso al puerto serie.
- **Os**: Permite realizar operaciones con el sistema.
- **Glob**: Librería de operaciones con directorios.
- **Time**: Se utiliza para trabajar con fechas y horas mediante un conjunto de funciones.
- **Picamera**: Biblioteca que proporciona una interfaz al módulo de la cámara de la Raspberry Pi.
- **Csv**: Librería para la importación y exportación de hojas de cálculo y *databases*.
- **Keyboard**: Biblioteca que permite tener el control total del teclado.
- **Pynpunt**: Se utiliza para controlar y monitorear dispositivos por el teclado.
- **Math**: Librería que ofrece funciones matemáticas.
- **Sys**: Encargado de proveer funcionalidades con el intérprete.
- **BMP280**: librería del sensor barométrico del módulo GY-91.
- **FaBo9Axis_MPU9250**: librería del sensor IMU del módulo GY-91.
- **Pynmea2**: Permite el acceso a los datos del GPS.
- **RPi.GPIO**: Biblioteca que permite leer y escribir los pines de entrada / salida de la Raspberry Pi.
- **Folium**: Librería que permite visualizar datos geoespaciales.

En la Raspberry se ha utilizado el sistema operativo PIXEL un *software* libre y de código abierto, conocido anteriormente como Raspbian. Es un sistema operativo basado es una distribución de GNU/Linux con el nombre de Debian, recomendado para

Raspberry Pi, debido a su optimización para dicho *hardware*. PIXEL es un entorno de escritorio simple y sencillo que viene dotado con paquetes de aplicaciones enfocadas a la productividad y herramientas de programación que se ha utilizado para la implementación de código del sistema de obtención de información.

El dispositivo de detección de objetos no integra periféricos para la ejecución y visualización del escaneado en escena, por lo que se ha optado por una arquitectura cliente-servidor a través del *software* VNC *Connect*. Este *software* nos permite el control remoto desde un ordenador cliente con la aplicación VNC *Viewer*, pudiendo controlar la interfaz conectándose a la IP de la Raspberry Pi mediante la aplicación VNC *Server*.

3.3. Conexión del sistema.

En este apartado se expone la implementación del circuito para el desarrollo del dispositivo, todas las conexiones giran en torno a una Raspberry Pi sobre la cual se conectarán los sensores. Un esquema de los puertos y pines que integra la placa que se ha utilizado nos facilita la explicación de las conexiones que se han realizado. La Fig. 3.6 muestra el diagrama de pines y puertos de la placa Raspberry Pi 3 Model B+.

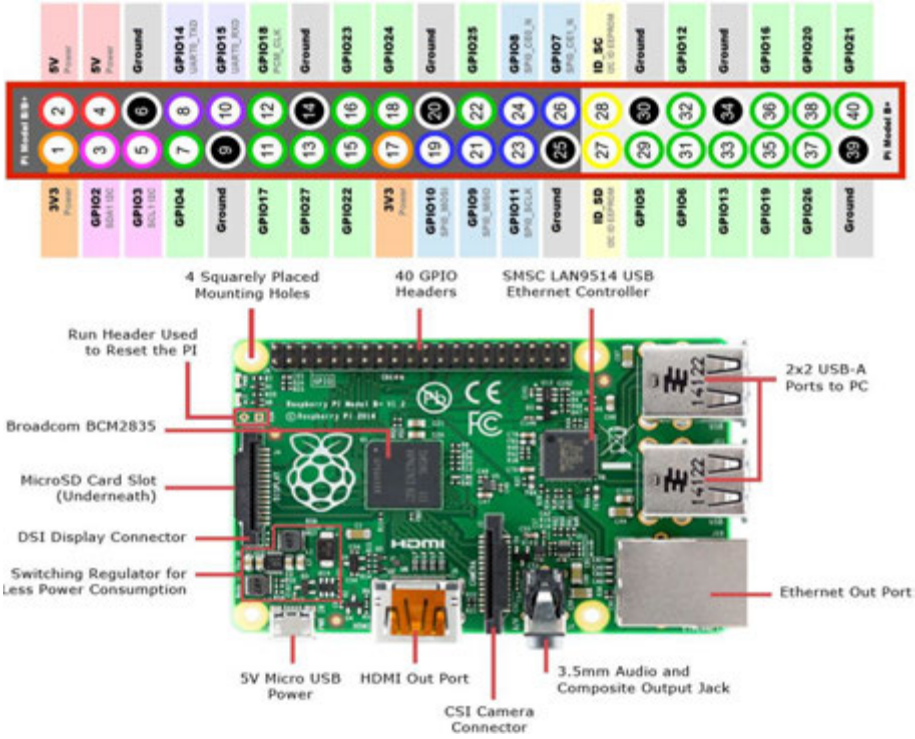


Fig. 3.6. Diagrama de puertos y pines Raspberry Pi 3 Model B+ [27].

Las conexiones de los diferentes sensores son las siguientes:

- Sensor LIDAR Hokuyo URG-04LX, el dispositivo cuenta con dos conexiones, un cable de alimentación de 5V y 0,5 A, conectado directamente a una batería y un cable de transmisión de datos USB 2.0 conectado a un puerto USB-A de la Raspberry Pi. En la Fig. 3.7 se observa las dos conexiones del sensor.

(a) (b)

Fig. 3.7. Conexiones del Sensor LIDAR Hokuyo URG-04LX [28].

- Pi Camera Module V2.1, la cámara se comunica por el estándar CSI-2 bus serie que define una interfaz entre la Pi Camera y el procesador de la placa, conectado directamente al puerto CSI de la Raspberry Pi, como se puede observar en la Fig. 3.8.

Fig. 3.8. Conexión de la Pi Camera Module V2.1.

- GPS NEO6MV2, la conexión se realiza mediante el protocolo de comunicación UART (*Universal Asynchronous Receiver-Transmitter*), el módulo se alimenta conectando Vcc y Gnd al pin 2 de 5V y el 6 Gnd de la Raspberry Pi, para la transmisión de datos conectaremos RX y TX con sus opuestos en la Rasberry utilizando el pin 8 GPIO_TXD y el pin 10 GPIO_RXD, como se puede observar en el siguiente esquema de conexión.

(a) (b)

Fig. 3.9 Conexión del GPS NEO6MV2 [29].

- Módulo GY-91 10 DOF, el módulo se comunica mediante la interfaz I2C (*Inter-Integrated Circuit*) utilizando dos líneas de información, una para la señal de reloj y otra dedicada a los datos. Este módulo se alimenta conectando 3V3 y Gnd al pin 1 3V3 y el 9 Gnd de la placa, para la trasmisión de los datos conectamos SCL y SDA con los pines I2C 3 y 5 de la Raspberry PI, en la Fig. 3.10 se muestra el esquema de conexión de este módulo.

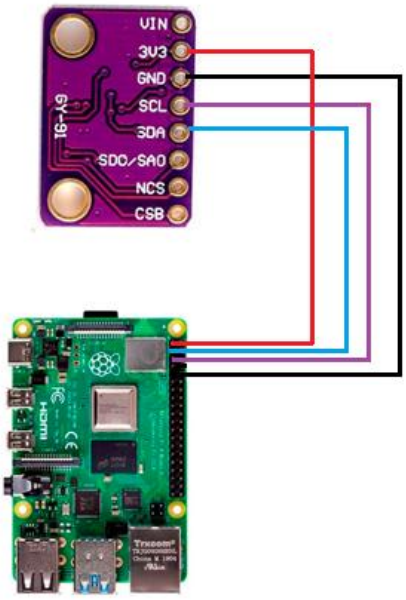


Fig. 3.10. Conexión del Módulo GY-91 10 DOF.

- Batería XTRON 5000, la Raspberry Pi y el conjunto de sensores son alimentados por esta batería, pudiendo desplazar el dispositivo por cualquier escenario.

Fig. 3.11. Batería XTRON 5000.

En la Fig. 3.12 se puede observar el dispositivo final, esta disposición no interfiere en el eje óptico del sensor LIDAR. La cámara se encuentra en el frontal del LIDAR alineados con sus ejes y lo más cercano al eje óptico del sensor láser. El módulo GY-91 que incluye magnetómetro y giroscopio están alineados con los ejes del LIDAR encontrándose en la parte superior de este.

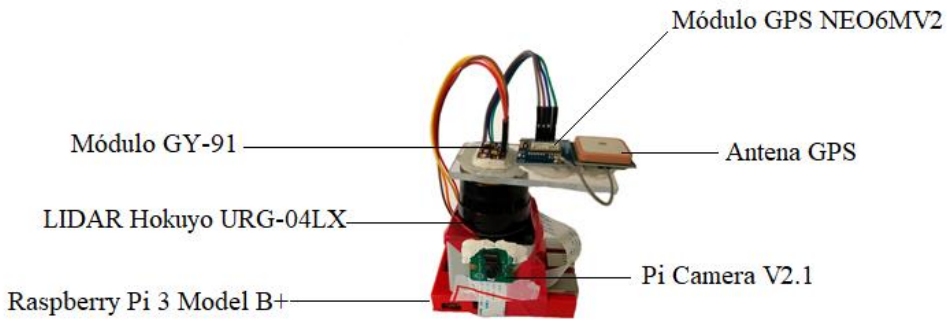


Fig. 3.12. Dispositivo de detección de objetos en movimiento.

4. EXPERIMENTACIÓN Y RESULTADOS OBTENIDOS.

En este capítulo se exponen los experimentos realizados con el objetivo de verificar el sistema desarrollado de detección de objetos en movimiento.

Debido a que el objetivo del trabajo es la detección de objetos en movimiento con sensor LIDAR, se muestran experimentos con las nubes de puntos registradas por este sensor. Se realizan diferentes experimentos donde se observa la imagen de la escena con el tratamiento de la nube de puntos, detectando el movimiento de los objetos. Además, se representan las transformaciones de las nubes de puntos obtenidas por el LIDAR en el sistema de coordenadas geodésicas WGS-84.

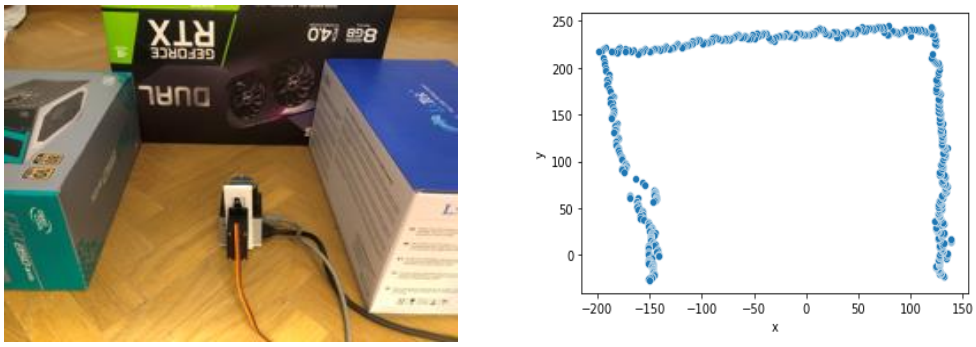
Por último, se presenta un sistema comparativo con sustracción de imágenes detectando el movimiento en las imágenes capturadas por la cámara RGB que integra el dispositivo, verificando el movimiento que obtiene el sistema en las nubes de puntos.

4.1. Experimentación.

Debido a que el objetivo principal del trabajo es la detección de objetos en movimiento con sensor LIDAR, la mayoría de los experimentos que aparecen en este capítulo contendrán nubes de puntos registradas por este sensor.

Los experimentos iniciales que se exponen son representaciones gráficas de las nubes de puntos 2D capturadas por el sensor LIDAR a diferentes distancias y con distintos objetos geométricos, observando las concentraciones de puntos en las distintas áreas del objeto o superficie. Estos experimentos se han realizado en recintos controlados con un ángulo de escaneado de ± 100º, estos son los siguientes:

- Escena 2D, distancia cercana, geometría recta. En la Fig. 4.1 se puede observar como el sensor LIDAR no está centrado en el recinto delimitado por los objetos rectangulares, se encuentra situado a unos 22 cm respecto a la superficie que se encuentra frontalmente, estos objetos tienen una ligera inclinación como se puede muestra en la nube de puntos, los laterales se encuentran más cercanos al sensor, lo que produce una mayor concentración de puntos.

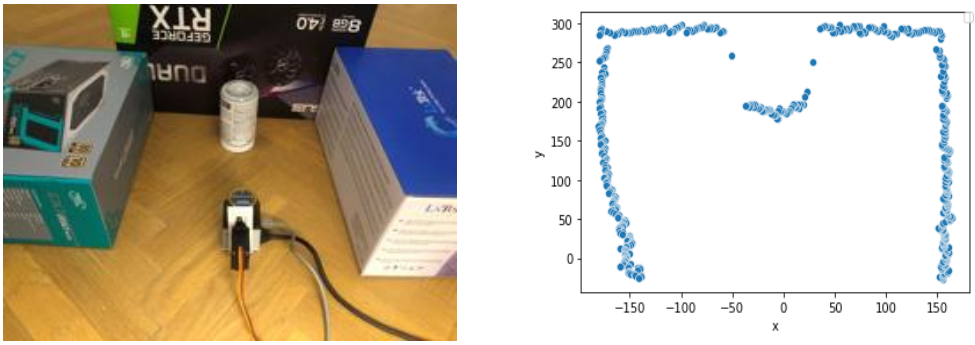


(a) (b)

Fig. 4.1. Representación gráfica de una escena 2D a distancia cercana con geometría recta.

- Escena 2D, distancia cercana, geometría curva. En la Fig. 4.2 se incluye un objeto curvo a una distancia de 20 cm respecto al sensor, en la gráfica se puede

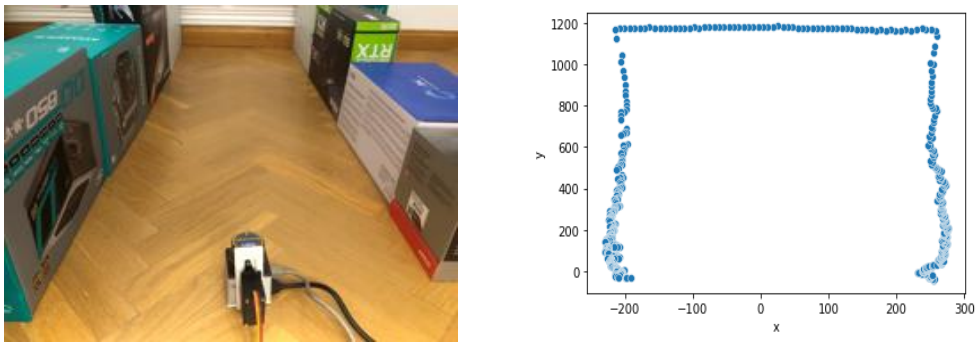
observar como en la zona del objeto curvo la nube de puntos tiene una ligera curvatura donde los puntos se encuentran más concentrados en el centro del objeto y a medida que nos acercamos a su borde la superposición de puntos disminuye.



(a) (b)

Fig. 4.2. Representación gráfica de una escena 2D a distancia cercana con geometría curva.

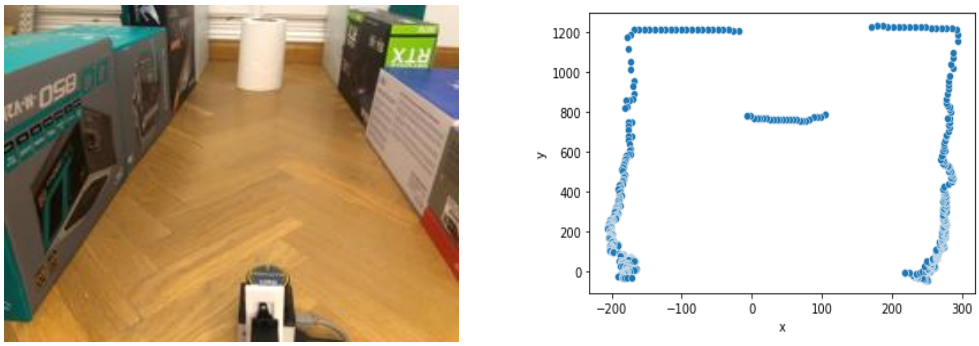
- Escena 2D, distancia media geometría recta. En la Fig. 4.3 se aumenta la distancia del experimento colocando el sensor a 120 cm respecto a la superficie frontal, se observa que la superficie lateral más cercana al sensor la densidad de puntos es superior y a medida que la distancia aumenta la densidad disminuye.



(a) (b)

Fig. 4.3. Representación gráfica de una escena 2D a distancia media con geometría recta.

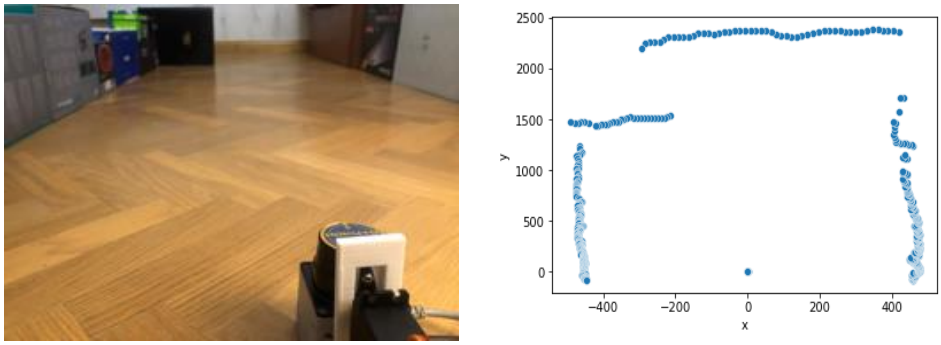
- Escena 2D, distancia media, geometría curva. En la Fig. 4.4 se ha incluido un objeto curvo a una distancia 80 cm en el mismo recinto que el anterior experimento. Se puede observar que la densidad de puntos en el objeto curvo es mucho menor que el de la Fig. 4.2, esto se debe a que la distancia se ha aumentado.



(a) (b)

Fig. 4.4. Representación gráfica de una escena 2D a distancia media con geometría curva.

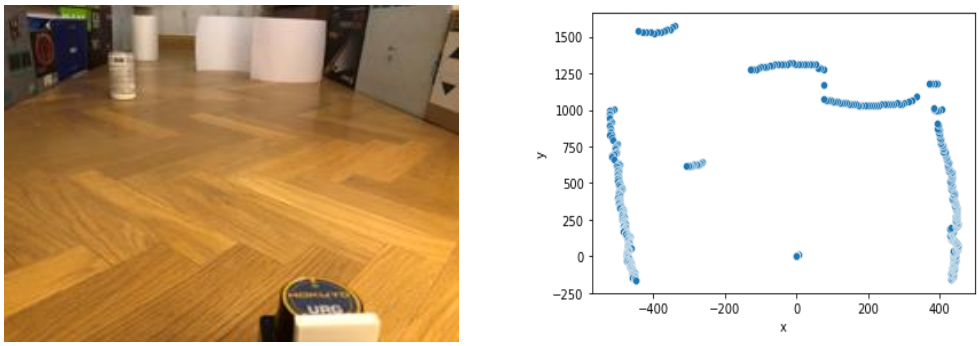
- Escena 2D, distancia media-lejana, geometría recta. En la Fig. 4.5 la distancia del escenario se ha aumentado a 250 cm. Se ha colocado un objeto rectangular sobresaliendo a los 150 cm y se observa que la densidad de puntos es mayor respecto a los puntos pertenecientes a la superficie que se encuentra a 250 cm.



(a) (b)

Fig. 4.5. Representación gráfica de una escena 2D a distancia media-lejana con geometría recta.

- Escena 2D, distancia media-lejana, geometría curva. En la Fig. 4.6 se han colocado varios objetos curvos a diferentes distancias, tres convexos con diferentes diámetros y uno cóncavo. Se observa que la nube de puntos representa perfectamente la escena que aparece en la imagen, mostrando correctamente la geometría de los objetos. En la imagen no se aprecia de forma definida las geometrías de los diferentes objetos, esta escena es una clara muestra de cómo el sensor LIDAR no se ve afectado por la iluminación a diferencia de los sensores de filtro óptico.



(a) (b)

Fig. 4.6. Representación gráfica de escena 2D a distancia media-lejana con geometría curva.

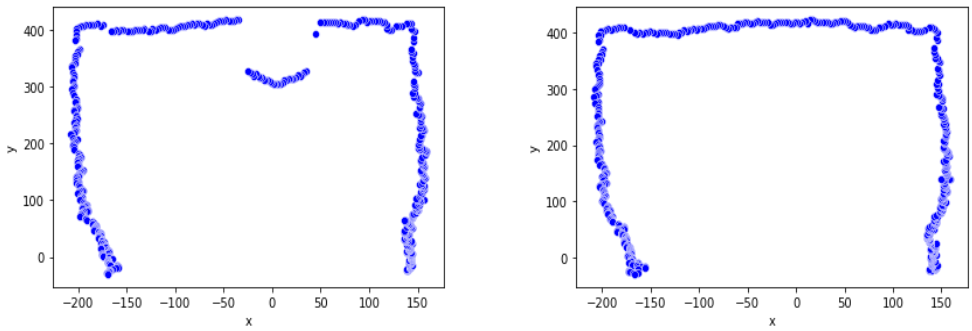
Los siguientes experimentos que se exponen se tratan de escenarios controlados donde el sensor LIDAR se encuentra estático y aparecen objetos con un determinado movimiento. Se aplicarán las diferentes técnicas comentadas en los anteriores capítulos para detectar este movimiento. Estos experimentos son los siguientes:

- Escenario donde un objeto curvo sale de la escena. En la Fig. 4.7 se puede observar un escenario donde un objeto curvo ha sufrido un desplazamiento. Se ha realizado un movimiento debido a que el objeto no se encuentra en la escena.

(a) (b)

Fig. 4.7. Imagen representativa de dos escenas adyacentes durante una secuencia en la que un objeto curvo sale de la escena. a) Imagen en t-1. b) Imagen en t.

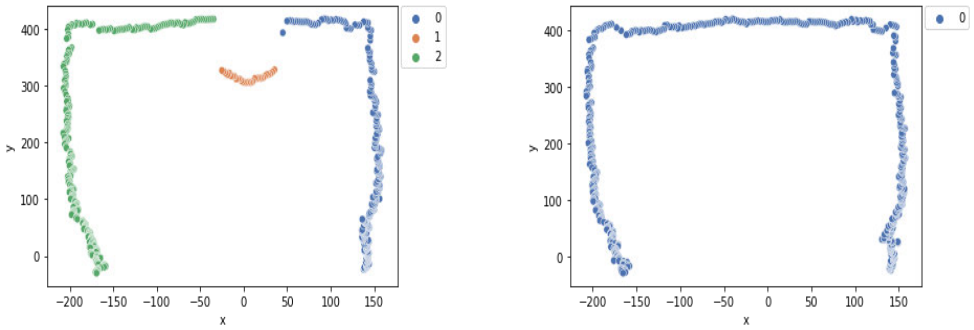
En la Fig. 4.8 se observa que el sensor LIDAR ha capturado un escenario con un objeto curvo, en la nube de puntos adyacente el objeto no se encuentra en dicha escena, por lo tanto, el objeto ha salido de la escena desempeñando un movimiento.



(a) (b)

Fig. 4.8. Representación gráfica de dos nubes de puntos correspondientes a dos escenas adyacentes durante una secuencia en la que un objeto curvo sale de la escena. a) Nube de puntos en t-1. b) Nube de puntos en t.

La Fig. 4.9 muestra el *clustering* de las nubes de puntos adyacentes, debido a que el número de *clusters* entre los dos escenarios adyacentes varia se determina que se ha producido un movimiento en el entorno. El número de *clusters* de la escena a) es superior al de la escena b) por lo que el sistema determina que se ha producido un movimiento en el objeto curvo que se muestra en la Fig. 4.7.



(a) (b)

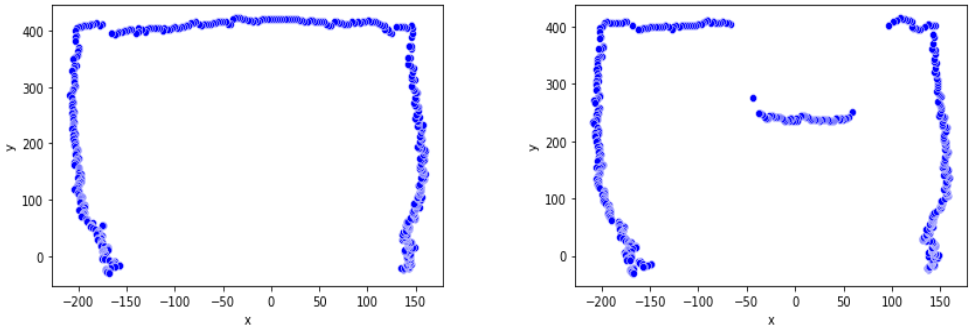
Fig. 4.9. *Clustering* aplicado a dos escenas adyacentes durante una secuencia en la que un objeto curvo sale de la escena. a) Registro de datos en t-1. b) registro de datos en t.

- Escenario donde un objeto rectangular entra en la escena. En la Fig. 4.10 se puede observar un escenario donde un objeto rectangular ha sufrido un movimiento. El desplazamiento se ha producido debido a que el objeto no se encontraba en la escena a) y en la escena b) se observa un nuevo objeto rectangular.

(a) (b)

Fig. 4.10. Imagen representativa de dos escenas adyacentes durante una secuencia en la que un objeto rectangular entra en escena. a) Imagen en t-1. b) Imagen en t.

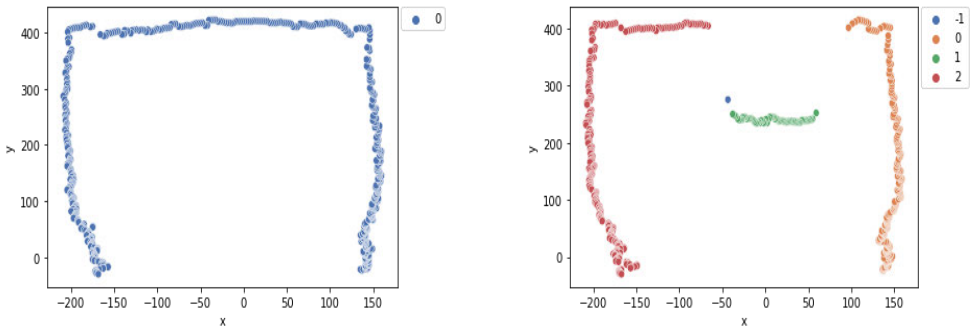
En la Fig. 4.11 se observa que el sensor LIDAR ha capturado un escenario a) en el que no se encuentra el objeto rectangular que aparece en la escena adyacente b).



(a) (b)

Fig. 4.11. Representación gráfica de dos nubes de puntos correspondientes a dos escenas adyacentes durante una secuencia en la que un objeto rectangular entra en escena. a) Nube de puntos en t-1. b) Nube de puntos en t.

La Fig. 4.12 muestra el *clustering* de las nubes de puntos adyacentes, debido a que el número de *clusters* entre los dos escenarios adyacentes ha variado se determina que se ha producido un movimiento en el entorno. El número de *clusters* de la escena a) es inferior al de la escena b) por lo que el sistema determina que se ha producido un movimiento en el objeto rectangular que se muestra en la Fig. 4.10, este ha entrado en escena.



(a) (b)

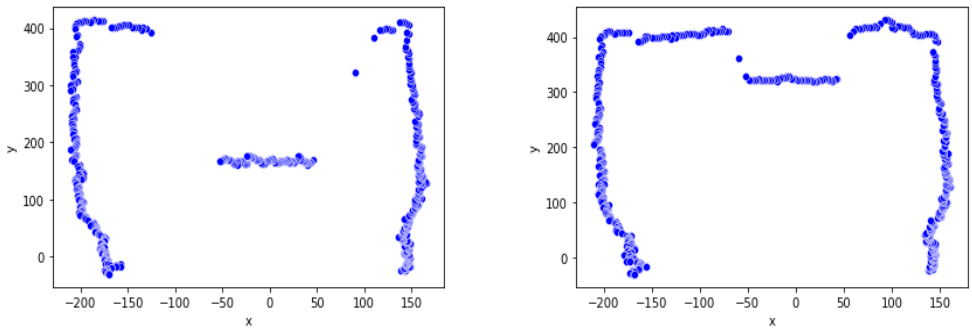
Fig. 4.12. *Clustering* aplicado a dos escenas adyacentes durante una secuencia en la que un objeto rectangular entra en escena. a) Registro de datos en t-1. b) registro de datos en t.

- Escenario donde un objeto rectangular se aleja del sensor LIDAR. En la Fig. 4.13 se puede observar un escenario donde un objeto rectangular ha sufrido un movimiento. El desplazamiento ha sido hacia detrás alejándose del sensor LIDAR.

(a) (b)

Fig. 4.13. Imagen representativa de dos escenas adyacentes durante una secuencia en la que un objeto rectangular se aleja del sensor LIDAR. a) Imagen en t-1. b) Imagen en t.

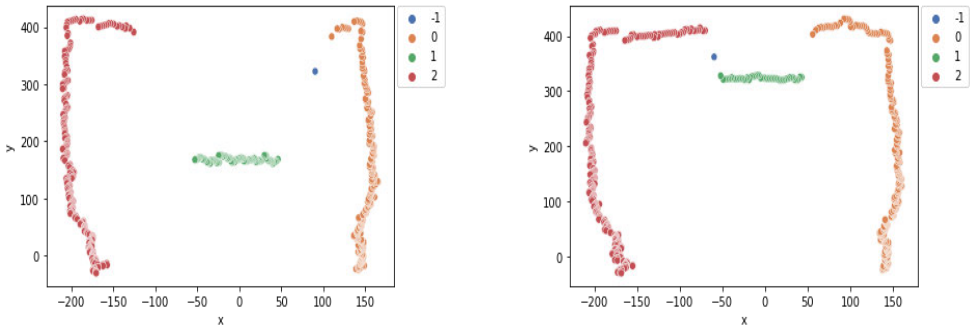
En la Fig. 4.14 se observa que el sensor LIDAR ha capturado perfectamente el movimiento ocurrido. Muestra una nube de puntos a una distancia de 18 cm, su forma denota que es un objeto rectangular, observando en la nube de puntos adyacente que dicho objeto se ha trasladado a una distancia de 32 cm.



(a) (b)

Fig. 4.14. Representación gráfica de dos nubes de puntos correspondientes a dos escenas adyacentes durante una secuencia en la que un objeto rectangular se aleja del sensor LIDAR. a) Nube de puntos en t-1. b) Nube de puntos en t.

La Fig. 4.15 muestra el *clustering* de dos nubes de puntos adyacentes, debido a que el objeto no ha desaparecido del escenario el número de *clusters* que se obtienen en ambas nubes de puntos es el mismo. El grupo etiquetado con "-1" se trata de *outliers*.



(a) (b)

Fig. 4.15. *Clustering* aplicado a dos escenas adyacentes durante una secuencia en la que un objeto se aleja del sensor. a) Registro de datos en t-1. b) registro de datos en t.

La Fig. 4.16 se observa la asociación de las nubes pertenecientes a cada *cluster*. El *cluster* "1" de ambas nubes de puntos es la región de color rojo y verde que representa el desplazamiento de un objeto rectangular. Se muestra el ajuste de las nubes de puntos de color verde y rojo donde el error se ha minimizado. El escenario a) es la plantilla donde el escenario b) adyacente se ha ajustado perfectamente minimizado el error entre ambos escenarios.

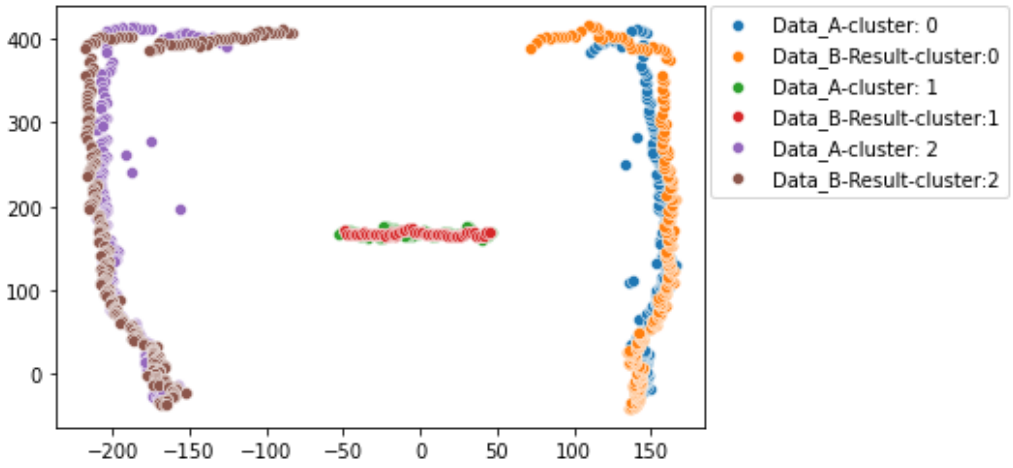


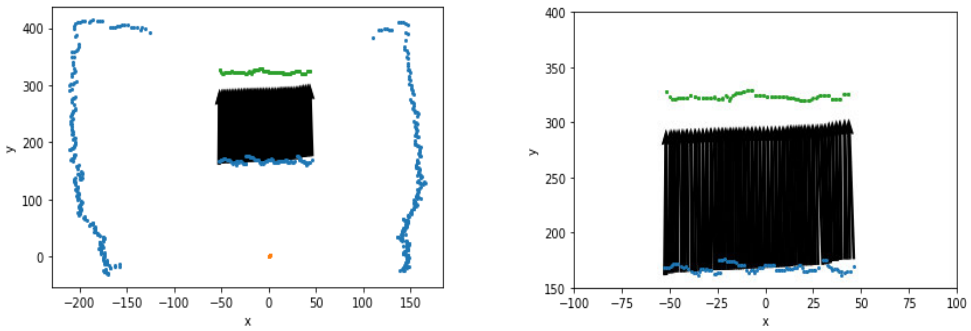
Fig. 4.16. Asociación de las nubes de puntos de un escenario donde un objeto rectangular se aleja del sensor.

La TABLA. 4.1 contiene el error cuadrático medio de los datos de este experimento, antes y después de aplicar el algoritmo ICP. El error cuadrático medio se ha calculado con las nubes de puntos pertenecientes a las regiones correspondientes del escenario a) y b). Se observa que la región que ha sufrido un movimiento el RMSE es mayor que las zonas donde no se desempeña ningún movimiento y se mantiene estático. Una vez aplicado el algoritmo ICP sobre los conjuntos de datos se obtiene que los errores se aproximan entre sí, de esta manera verificamos que se ha desempeñado un movimiento en el escenario escaneado.

TABLA. 4.1. RMSE, EXPERIMENTO DONDE UN OBJETO RECTANGULAR SE ALEJA DEL SENROR.

	Cluster 0	**Cluster 1**	**Cluster 2**
RMSE datos	0.1245	0.8245	0.2577
RMSE ICP	0.0765	0.0975	0.0796

En la Fig. 4.17 se muestra la dirección del movimiento, una vez el sistema identifica la región que ha sufrido un movimiento realiza el cálculo del gradiente de dicha región, obteniendo la dirección del movimiento. Como se puede observar la dirección del movimiento se ha representado gráficamente con una flecha entre los pares de puntos asociados entre sí.



(a) (b)

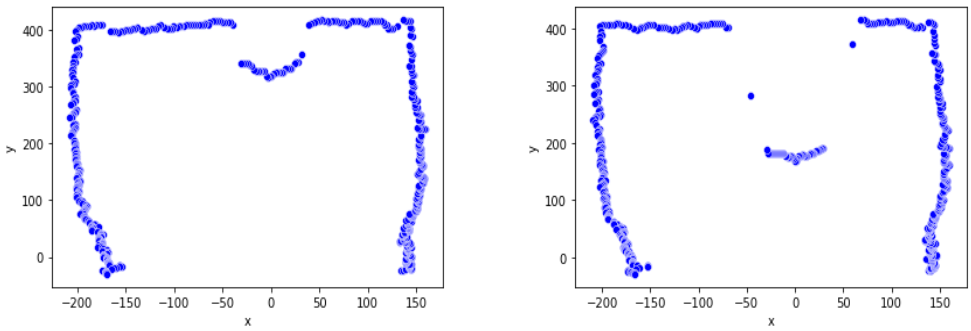
Fig. 4.17. Representación gráfica de la dirección del movimiento, escenario donde un objeto rectangular se aleja del sensor.

- Escenario donde un objeto curvo se acerca al sensor LIDAR. En la Fig. 4.18 se observa un escenario donde un objeto curvo ha sufrido un movimiento. El desplazamiento ha sido hacia delante acercándose al sensor LIDAR.

(a) (b)

Fig. 4.18. Imagen representativa de dos escenas adyacentes durante una secuencia en la que un objeto curvo se acerca al sensor LIDAR. a) Imagen en t-1. b) Imagen en t.

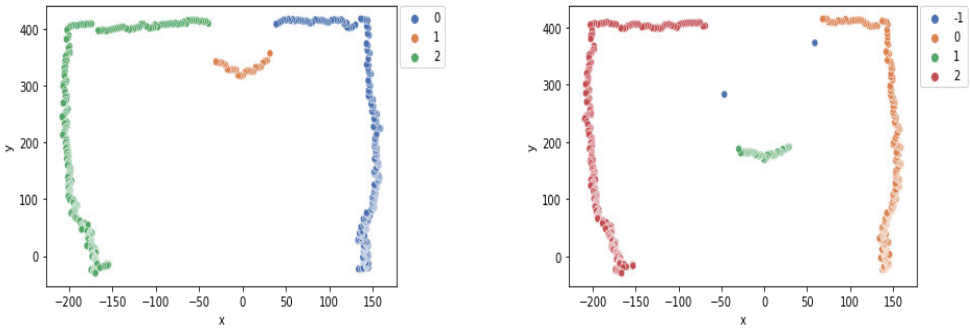
En la Fig. 4.19 se observa que el sensor LIDAR ha capturado perfectamente el movimiento. Se muestra en la primera escena una nube de puntos circular sobre los 33 cm de distancia, en la siguiente escena dicha nube de punto se ha traslado a 18 cm de distancia.



(a) (b)

Fig. 4.19. Representación gráfica de dos nubes de puntos correspondientes a dos escenas adyacentes durante una secuencia en la que un objeto curvo se acerca al sensor LIDAR. a) Nube de puntos en t-1. b) Nube de puntos en t.

La Fig. 4.20 muestra el *clustering* de dos nubes de puntos adyacentes, el número de *clusters* que se obtienen en ambas nubes de puntos es el mismo, debido a que el objeto no ha desaparecido del escenario, recordando que el grupo etiquetado con "-1" que se muestra en la escena b) son *outliers* que serán eliminados de la nube de puntos.



(a) (b)

Fig. 4.20. *Clustering* aplicado a dos escenas adyacentes durante una secuencia en la que un objeto curvo se acerca al sensor. a) Registro de datos en t-1. b) registro de datos en t.

La Fig. 4.21 se observa la asociación de las nubes pertenecientes a cada *cluster*. El *cluster* "1" de ambas nubes de puntos es la región de color rojo y verde que representa el desplazamiento de un objeto curvo. Se muestra el ajuste de las nubes de puntos de color verde y rojo donde el error se ha minimizado. El escenario a) es la plantilla donde el escenario b) adyacente se ha ajustado perfectamente minimizado el error entre ambos escenarios.

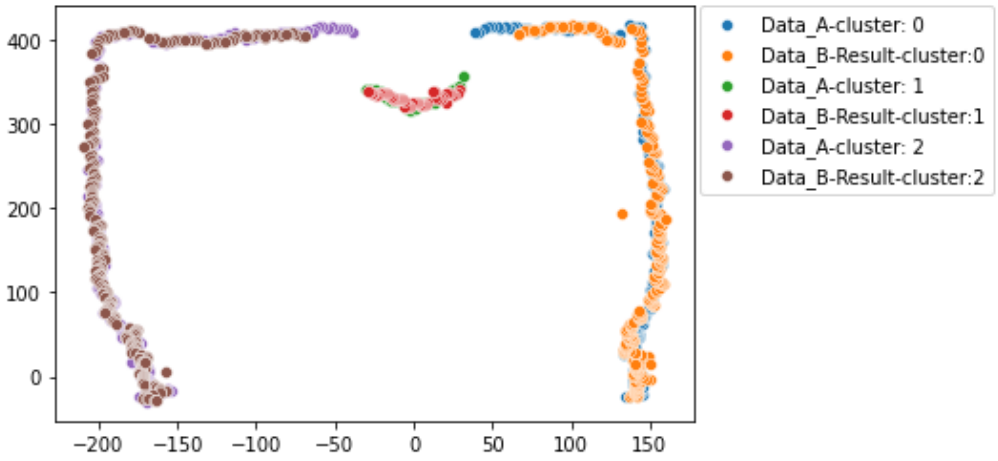


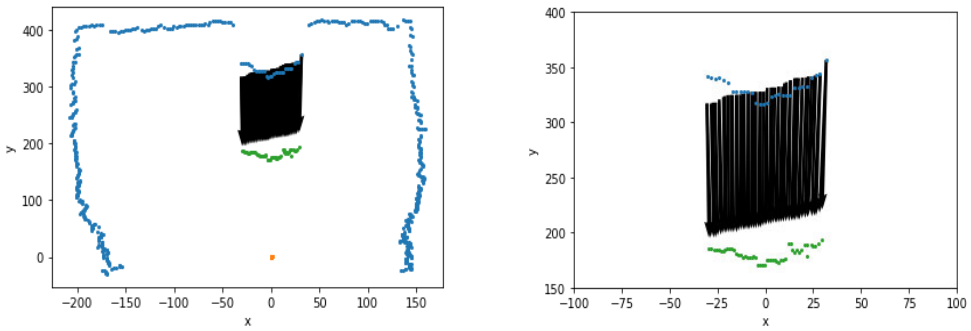
Fig. 4.21. Asociación de las nubes de puntos de un escenario donde un objeto curvo se acerca al sensor.

La TABLA. 4.2 presenta el RMSE de este escenario. Lo que se puede observar es lo mismo que en la TABLA. 4.1, el error cuadrático medio de la región donde hay movimiento es superior que las regiones que se mantienen estáticas, de esta manera podemos verificar que existe un movimiento en el escenario. Tras aplicar el algoritmo ICP sobre el conjunto de datos, observamos que los errores se aproximan verificando que en este escenario la región del *cluster* "1" ha sufrido un desplazamiento.

TABLA. 4.2. RMSE, EXPERIMENTO DONDE UN OBJETO CURVO SE ACERCA AL SENSOR.

	Cluster 0	**Cluster 1**	**Cluster 2**
RMSE datos	0.0668	0.7584	0.0945
RMSE ICP	0.0345	0.0531	0.0432

En la Fig. 4.22 se muestra la dirección del movimiento, una vez el sistema identifica la región que ha sufrido un movimiento realiza el cálculo del gradiente. Como se puede observar la dirección del movimiento se ha representado gráficamente con una flecha entre los pares de puntos asociados entre sí.

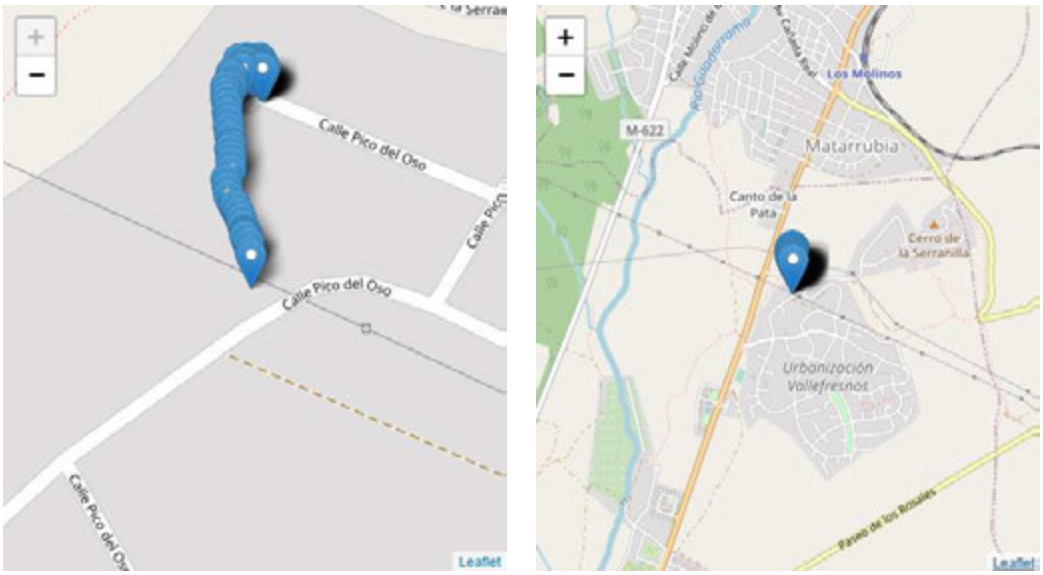


(a) (b)

Fig. 4.22. Representación gráfica de la dirección del movimiento escenario donde un objeto curvo se acerca.

Los siguientes experimentos que se muestran son la representación de las coordenadas obtenidas al realizar la transformación de los datos del LIDAR a coordenadas geodésicas WGS-84:

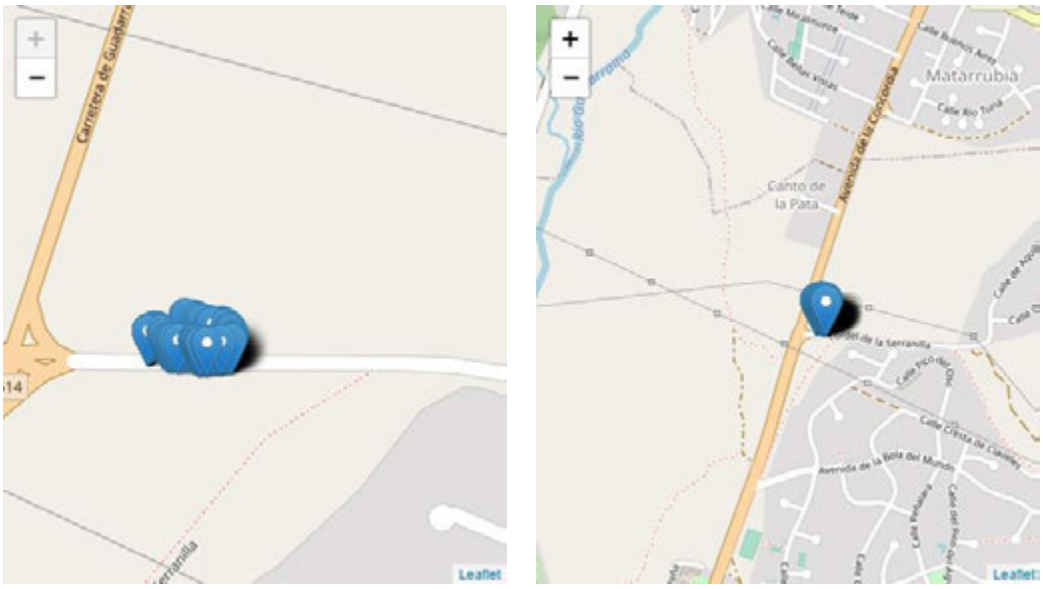
- Representación del primer experimento en el mapa. En esta prueba se realiza el escaneado por el recorrido que se muestra en la Fig. 4.23, coincidiendo con la localización del entorno donde se realiza el escaneado.



(a) (b)

Fig. 4.23. Representación en un mapa de la transformación de los datos del LIDAR a coordenadas geodésicas WGS-84. Experimento 1.

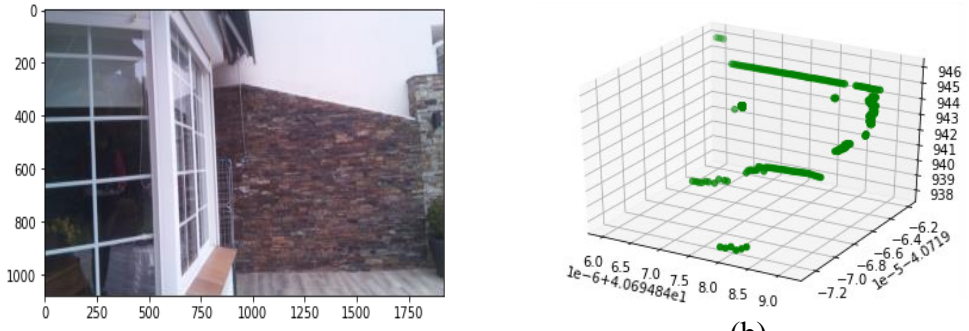
- Representación del segundo experimento en el mapa. En esta prueba se realiza el escaneado por una carretera como se observa en la Fig. 4.24, coincidiendo con la localización del entorno donde se realiza el escaneado.



(a) (b)

Fig. 4.24. Representación en un mapa de la transformación de los datos del LIDAR a coordenadas geodésicas WGS-84. Experimento 2.

- Representación del tercer experimento en una grafico 3D. En esta prueba se muestra las coordenadas (WGS84_X, WGS84_Y, WGS84_Z), representadas en un gráfico 3D. En la Fig. 4.25 se muestra la representación en 3D de la escena a).



(a) (b)

Fig. 4.25. Representación gráfica en 3D de las coordenadas WGS-84.

4.2. Sistema comparativo.

Se presenta un sistema comparativo en la detección de objetos en movimiento utilizando la cámara que integra la Raspberry, de esta manera se desempeña otra función a parte de la captura de la escena en una imagen en cada iteración de escaneado, pudiendo detectar el movimiento en las diferentes regiones de la imagen. Este sistema es complementario y se presenta como una verificación de los movimientos obtenidos en las nubes de puntos, sin desviar la atención del objetivo principal que es la detección de objetos con un sensor LIDAR.

Uno de los problemas de este sistema son los cambios de iluminación, cuando la iluminación varía se detecta movimiento que no se ha producido. Además, este sistema no cuantifica el movimiento, a diferencia del sensor LIDAR que se puede cuantificar el

desplazamiento ocurrido en el mundo real, por lo que se presenta como una verificación del movimiento capturado por el LIDAR

Este sistema desarrollado se basa en detección de objetos en movimiento con sustracción de imágenes utilizando la biblioteca OpenCV (*Open Computer Vision*). Este proceso se basa en la sustracción del fondo, obteniendo únicamente la región donde se desempeña el movimiento, descartando el resto de la escena que se mantiene estática, para ello se hace eso de un algoritmo de segmentación de fondo que se basa en Gaussian-Mixture. El proceso de desarrollo de esta técnica empleada es el siguiente:

- Transformación de las imágenes RGB a escala de grises.
- Sustracción de imágenes, este proceso es iterativo y se aplica la diferencia entre imágenes.
- El siguiente paso es emplear el método de umbralización simple, las imágenes fueron transformadas a escala de grises, de esta manera podemos transfórmalas a una imagen binaria, es decir, una imagen en blanco y negro. La umbralización simple permite encontrar los contornos de los objetos.
- Discriminación de contornos, con el proceso anterior se han encontrado los contornos, por lo que se deben descartar aquellos que su tamaño sea inferior al área establecido y no representen un movimiento en la imagen. Por tanto, los que no superen una determinada área serán descartados, por el contrario, se recoge en un rectángulo la zona de la imagen donde se ha producido un movimiento.

En la Fig. 4.26 se observa un experimento donde se detecta el movimiento con la técnica de sustracción de imágenes, apareciendo en escena un objeto cilíndrico con un desplazamiento paralelo al dispositivo, el sistema detecta el movimiento en el objeto y lo recoge en un rectángulo.

(a) (b)

Fig. 4.26. Representación de la detección de movimiento basada en sustracción de imágenes.

En la Fig. 4.27 escenario a) se muestra el objeto en movimiento en escala de grises y en el escenario b) el objeto en blanco y negro, debido a que la imagen ha sido transformada a imagen binaria mediante la tecnica de umbralización simple.

(a) (b)

Fig. 4.27. Representación de la imagen en escala de grises y transformada a imagen binaria.

La Fig. 4.28 muestra el contorno del objeto en rojo obtenido tras el proceso de umbralización simple.

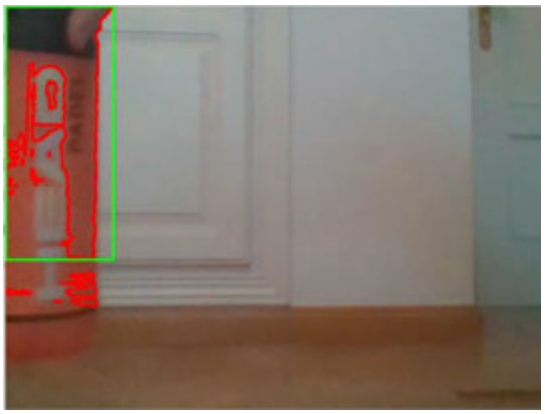


Fig. 4.28. Representación del contorno del objeto en movimiento.

En la Fig. 4.29 muestra el objeto en movimiento que se detectó con la nube de puntos obtenida por el LIDAR, experimento que se muestra en la Fig. 4.13.

(a) (b)

Fig. 4.29. Imagen representativa de dos escenas adyacentes durante una secuencia en la que se ha detectado un objeto en movimiento mediante la técnica de sustracción de imágenes. a) Imagen en t-1. b) Imagen en t.

Resumen general de los experimentos:

En la TABLA. 4.3 se muestra un resumen de los resultados obtenidos en todos los experimentos realizados de detección de objetos en movimiento con LIDAR.

TABLA. 4.3. RESUMEN DE LOS EXPERIMENTOS REALIZADOS.

Experimento	**Descripción**	**Detección con el dispositivo LIDAR**	**Verificación del sistema comparativo**
Desplazamiento lateral de un objeto.	Escenario donde se produce un movimiento lateral de un objeto curvo.	Sí	Sí
Cruce de dos objetos de diferente tamaño.	Escenario donde se produce un movimiento lateral de objetos de diferente dimensión en sentido opuesto.	Sí	Sí
Desplazamiento en el eje Z del dispositivo.	El dispositivo se desplaza en el eje Z elevándose, mientras un objeto se mueve hacia detrás.	Sí	Sí
Desplazamiento en el eje Y del dispositivo.	El dispositivo se desplaza en el eje Y acercándose al objeto, mientras el objeto sufre un movimiento formando aproximadamente 45º con respecto al eje Y del LIDAR.	Sí	Sí
Desplazamiento en el X del dispositivo.	El dispositivo se desplaza en el eje X paralelo al objeto que se acerca al dispositivo.	Sí	Sí
Desplazamiento de un objeto curvo hacia delante.	El dispositivo se encuentra estático y el objeto curvo se desplaza acercándose al dispositivo.	Sí	Sí
Desplazamiento de un objeto rectangular hacia delante.	El dispositivo se encuentra estático y el objeto rectangular se desplaza acercándose al dispositivo.	Sí	Sí
Desplazamiento de un objeto curvo hacia detrás.	El dispositivo se encuentra estático y el objeto curvo se desplaza alejándose al dispositivo.	Sí	Sí

Desplazamiento de un objeto rectangular hacia detrás.	El dispositivo se encuentra estático y el objeto rectangular se desplaza alejándose al dispositivo.	Sí	Sí
Objeto curvo entra en escena	El dispositivo se encuentra estático y el objeto curvo sale de la escena.	Sí	Sí
Objeto rectangular sale de escena.	El dispositivo se encuentra estático y el objeto rectangular sale de la escena.	Sí	Sí
Objeto curvo entra en escena.	El dispositivo se encuentra estático y un objeto curvo entra en escena.	Sí	Sí
Objeto rectangular sale de la escena.	El dispositivo se encuentra estático y un objeto rectangular entra en escena.	Sí	Sí
Giro del dispositivo en el eje Z.	El dispositivo gira sobre su eje Z mientras un objeto se desplaza hacia detrás alejándose.	Sí	No
Desplazamiento de un vehículo acercándose al dispositivo.	El dispositivo se encuentra en movimiento y un vehículo se acerca al dispositivo.	No	Sí
Desplazamiento en todas las componentes del LIDAR.	El dispositivo se mueve en los 3 ejes (x,y,z) en un escenario controlado, donde se desempeña un movimiento por un objeto.	Sí	No

Del conjunto de experimentos realizados el 80% se han verificado mientras que el 20% restante no se ha podido verificar si se ha producido movimiento. Esto se debe a que el sistema comparativo no detecto el movimiento por los cambios de iluminación sufridos en el entorno, sin embargo, el sistema detecto en las nubes de puntos con el cálculo del error que se había producido un movimiento. En uno de los experimentos no se detectó el movimiento en las nubes de puntos, esto se debe al brillo y reflexión del material del objeto, pasando desapercibido ante el escaneado del sensor LIDAR.

5. CONCLUSIONES Y TRABAJOS FUTUROS.

En este capítulo se presentan las conclusiones del trabajo realizando un repaso de los resultados obtenidos tras los experimentos expuestos en el apartado anterior y se proponen líneas futuras de trabajo para la mejora de este proyecto.

5.1. Conclusiones.

En este proyecto se ha implementado un sistema computacional capaz de detectar objetos en movimiento utilizando un sensor LIDAR. Teniendo presente los objetivos expuestos en la introducción del manuscrito, a continuación, se evalúa cada uno de ellos y el grado de cumplimiento.

La propuesta de este trabajo es la detección de objetos en movimiento con sensor LIDAR. El proyecto se conforma por el diseño y desarrollo de un dispositivo capaz de registrar información de un entorno mediante un conjunto de sensores (LIDAR, cámara RGB, GPS, IMU y barómetro). Además, se implementa un sistema que permite la detección de movimiento en las nubes de puntos registradas por el sensor LIDAR y la representación de las transformaciones de las nubes de puntos al sistema de coordenadas geodésicas mundial WGS-84.

- Como primer objeto cumplido que se puede destacar, es el desarrollo e implementación de un sistema capaz de registrar la información de un entorno mediante un sensor LIDAR junto a otros sensores (GPS, cámara, IMU y barómetro) conectados a una Raspberry Pi, con el fin de ser tratados en diferido. Este sistema permite previsualizar las nubes de puntos capturadas por el LIDAR en tiempo real. La implementación de este sistema recoge las transformaciones de la información obtenida por el sensor LIDAR a un sistema de coordenadas geodésicas WGS-84.
- Posteriormente, en el procesado de los datos se ha logrado eliminar errores e información incompleta, como ha sido descrito en el apartado 3.1.3 de este documento.
- Se ha conseguido la implementación de un sistema que permita visualizar las nubes de puntos 2D con su correspondiente imagen, junto a la visualización de los datos geodésicos WGS84 en un mapa.
- Gracias a los experimentos recogidos en el capítulo 4 se puede observar el cumplimiento del objetivo de detectar objetos en movimiento con un sensor LIDAR. Se muestra la aplicación de las técnicas de segmentación agrupación y asociación de nubes de puntos, verificando la detección del movimiento con un sistema comparativo.

Recapitulando se ha conseguido de manera satisfactoria todos los objetivos planteados inicialmente. Se ha conseguido capturar información del entorno mediante el dispositivo desarrollado, detectando movimiento en las diferentes regiones de las nubes de puntos y transformando dichas nubes obtenidas con el sensor LIDAR a un sistema de coordenadas geodésicas mundial. Además, se ha presentado un sistema comparativo de

verificación, detectando el movimiento en las imágenes capturadas por la cámara RGB que integra el dispositivo. Globalmente, se ha estudiado cómo y por qué se usan sensores LIDAR para la detección de objetos, recopilando el proceso y técnicas utilizadas en este manuscrito.

5.2. Trabajos futuros y posibles mejoras.

El proyecto desarrollado en este TFG permite ser el punto de partida de nuevas líneas de investigación, mejorando el mismo. Durante el periodo de diseño y desarrollo han aparecido diferentes alternativas que pueden dar lugar a posibles mejoras y aplicaciones futuras.

Una de las posibles mejoras sería el reemplazo del sensor LIDAR 2D por uno que realice el escaneado en 3D, pudiendo obtener mayor información de los objetos y el entorno.

El sistema de verificación propuesto en el capítulo de experimentación (detección de objetos en movimiento basado en sustracción de imágenes) no es posible cuantificar el desplazamiento en el mundo real, por tanto, un trabajo futuro sería encontrar la transformación de espacios que relacionan el sistema de visión con el del sensor LIDAR.

Finalmente, se puede desarrollar una mejora en la implementación del dispositivo para que el sistema detecte en tiempo real los objetos en movimiento, pudiendo ser incluido en sistemas de robótica o con el reemplazo de un sensor de mayor alcance en UAVS debido a su reducido peso y tamaño.

6. ANÁLISIS DEL PROYECTO.

En este capítulo se realiza un análisis del proyecto sobre la planificación y presupuestos que se han tenido en este proyecto, se expone una planificación inicial del presente trabajo, así como el presupuesto que supone llevarlo a la práctica. Además, se realiza una comparación de la planificación inicial con los tiempos realmente empleados en el desarrollo del proyecto y de los costes que se han tenido.

6.1. Planificación inicial.

En este apartado se expone la planificación del presente documento que se lleva en paralelo junto a las diferentes fases que tiene el proyecto, se presenta la TABLA. 6.1 con las diferentes tareas y los días estimados para cada uno de ellas, posteriormente, se muestra el diagrama de Gantt correspondiente a la tabla, observando de manera más sencilla la repartición de las tareas en cada periodo.

Se estima un total de 120 días en la planificación inicial del proyecto. La cual se divide en 6 fases distintas, estas fases son las siguientes:

- Investigación y estudio del estado del arte: la primera fase con una extensión de 15 días, se investigan los principales proyectos que se han realizado en la detección de objetos con un sensor LIDAR, además de sus principales características y capacidades para poder tener un primer enfoque y abordar el proyecto. También, se recaba información para el estado del arte.
- Implementación y procesado del LIDAR: la segunda fase tiene una duración de unos 7 días. Se comienza realizando varias pruebas con un sensor LIDAR 2D en concreto “Hokuyo URG-04LX” y una Raspberry para familiarizarse con este tipo de sensores. Posteriormente se realiza una implementación en el lenguaje Python para recoger los datos del LIDAR y almacenarlos en un archivo con extensión “.csv”, con el objetivo de realizar un procesado de los datos para poder visualizarlos en una gráfica y observar la nube de puntos 2D obtenidas de cada escena.
- Investigación e implementación de nuevos sensores: la tercera fase se realiza en un periodo de 10 días, se estudian diferentes sensores que pueden ser interesantes incluir en el dispositivo para complementar los datos del LIDAR. Además, se realiza la implementación e integración de los sensores en el dispositivo.
- Transformación de los datos LIDAR: la cuarta fase tiene una duración de 25 días, se implementa las correcciones de los datos del sensor LIDAR con la ayuda del GPS, del magnetómetro y barómetro, debido a que las medidas del sensor LIDAR están referidas a los ejes de su plano y es necesario rotarlas a un sistema de referencia local para posteriormente trasladarlas a un sistema de coordenadas geodésicas.

- Pruebas y análisis de los datos: la quinta fase con una extensión de 23 días, se realizan diferentes experimentos en distintos escenarios, variando las distancias entre observaciones y con diversos objetos geométricos, con el objetivo de conocer que algoritmos de *clustering* son más idóneos. También se procesan los datos para posteriormente realizar una representación de estos.
- Creación del documento: la sexta y última fase se estima una duración de 40 días, para completar toda la documentación del proyecto.

TABLA. 6.1. PLANIFICACIÓN INICIAL.

Fase del proyecto	**Tarea**	**Fecha de inicio**	**Fecha de fin**	**Días**
Primera	Investigación de proyectos similares.	26/01/2021	05/02/2021	15
	Estudio del estado del arte.	06/02/2021	09/02/2021	
Segunda	Implementación LIDAR.	10/02/2021	14/02/2021	7
	Procesado de los datos LIDAR.	15/02/2021	16/02/2021	
Tercera	Investigación de nuevos sensores.	17/02/2021	23/02/2021	10
	Implementación e integración de los sensores.	24/02/2021	26/02/2021	
Cuarta	Transformación de los datos LIDAR.	27/02/2021	23/03/2021	25
Quinta	Pruebas.	24/3/2021	07/04/2021	23
	Análisis de los datos.	08/04/2021	15/04/2021	
Sexta	Creación del documento.	16/04/2021	25/05/2021	40

La estimación inicial del trabajo es de una duración de 120 días naturales (del 26/01/2021 al 25/05/2021), de los cuales 90 días serán útiles, con una jornada completa de 8 horas hace un total de 720 horas para completar el proyecto.

6.2. Diagrama de Gant.

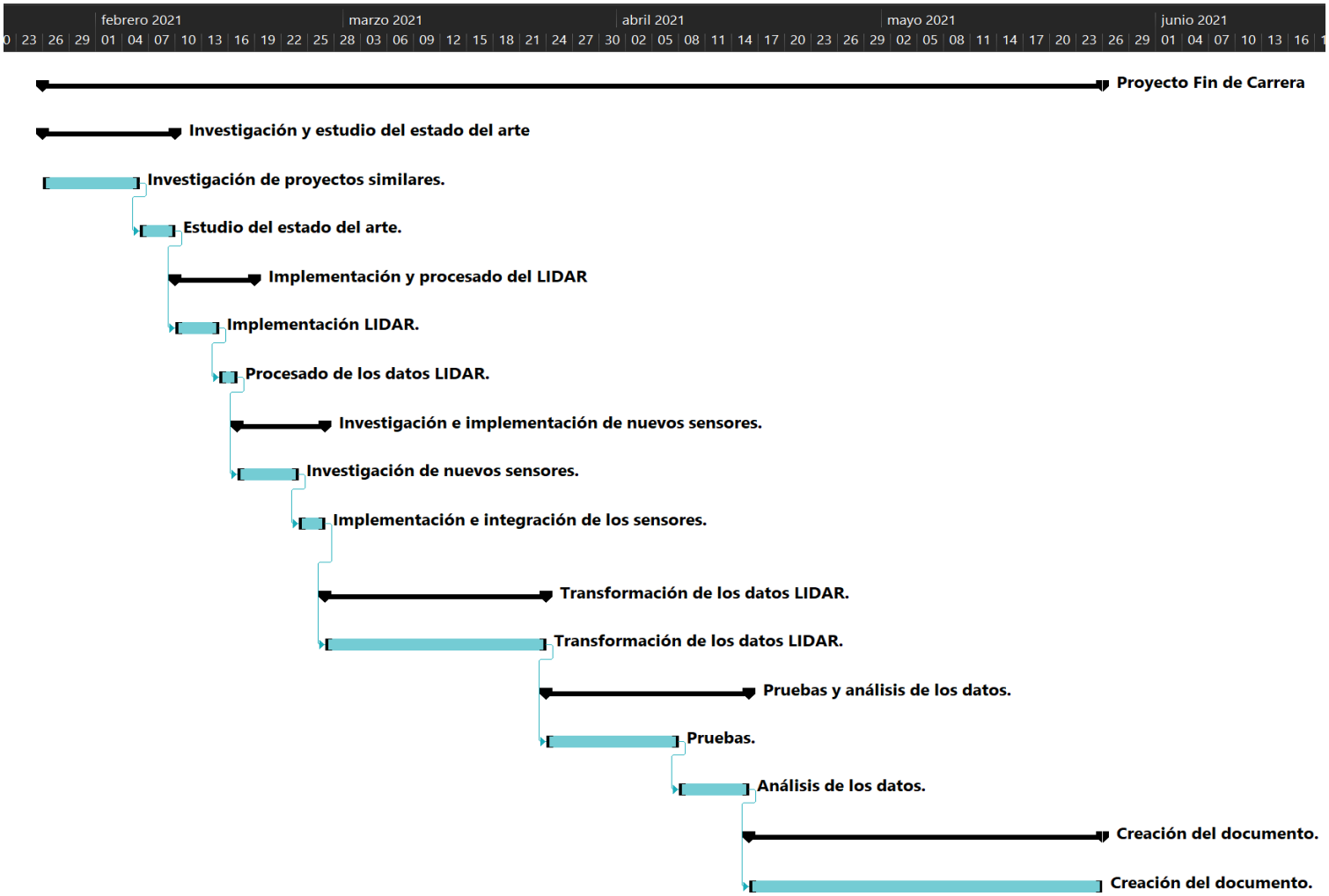


Fig. 6.1. Diagrama de Gantt

6.3. Presupuesto.

Este apartado recoge una estimación del coste del trabajo en distintos aspectos. Realizando los cálculos económicos del personal, *hardware, software*… etc. Los costes que aparecen en el documento excluyendo el total del proyecto, no aplican el Impuesto de Valor Añadido (IVA) del 21%.

Costes de personal.

Se especifica los costes del personal empleado en la realización de este proyecto. La planificación inicial se estiman un total de 120 días de los cuales 90 son laborables, suponiendo que el desarrollo integro lo realiza un único empleado, el cual es supervisado por un director de proyecto, con una jornada de trabajo de 8 horas diarias, cuyo sueldo se estiman en 10 € por hora según [30], por tanto, el empleado tendría un sueldo de 80 € diarios, mientras que el director de proyecto tiene un sueldo de 30 € por hora. En el coste total se añadirá el 30% que se tiene que pagar a la seguridad social.

TABLA. 6.2. COSTE DEL PERSONAL.

Cargo	**Horas totales**	**Días totales**	**Coste diario**	**Coste bruto**	**Coste total**
Director de proyecto	28 horas	14 días	60 €	840 €	**1092 €**
Ingeniero senior	720 horas	90 días	80 €	7200 €	**9360 €**
Coste total					**10452 €**

En la TABLA. 6.2 se muestra que el coste total del personal empleado es de 10452 € ya incluido el coste de la seguridad social.

Coste del hardware.

Se indican los costes del equipo *hardware* necesario para la realización de este proyecto. Será necesario la compra de un ordenador portátil para el análisis de los datos, creación del documento y como ordenador cliente para la ejecución del escaneo. Además, la compra de una Raspberry Pi como ordenador servidor donde se conectarán diferentes sensores para la obtención de información, estos sensores serán necesarios comprarlos junto a una batería que alimente todo el dispositivo.

TABLA. 6.3. COSTE DEL HARDWARE.

Equipo	**Valor del equipo sin IVA (**Impuesto sobre el Valor Añadido**)**
ASUS Zenbook Pro 15 UX580GD	1500 €
Raspberry Pi 3B+	29 €
LIDAR Hokuyo URG-04LX-UG01	1500 €

Módulo GPS de NEO-6M NEO6MV2	5 €
Módulo GY-9110DOF	8,75 €
Raspberry Pi Camera Module V2.1 8MP	30 €
Batería SLS Xtron 5000mah	35 €
Cableado Longruner Jumper Wir	5 €
Coste total	**3112,75 €**

Coste del software.

Se muestra le coste del *software* empleado para la realización del proyecto. Será necesario la compra de una licencia de Windows, debido a que el portátil no integra ningún sistema operativo de fábrica. Además, se necesitará un paquete de ofimática para la creación del documento.

TABLA. 6.4. COSTE DEL SOFTWARE.

Software	**Coste**
Licencia del SO Windows 10 Pro	250 €
Microsoft Office 365 Business Premium	140 €
Visual Studio Code	0,00 €
Raspberry Pi OS	0,00 €
Coste total	**390 €**

Coste de viajes y dietas.

Se exponen los costes de transporte y dieta del personal a cargo del proyecto. Dentro de viajes se recogen todo coste de desplazamiento.

TABLA. 6.5. COSTE DE VIAJES Y DIETAS.

Descripción	**Coste unitario**	**Cantidad**	**Coste**
Dieta	7,5 €	90 días	675 €
Transporte	5 €	90 días	450 €
Coste total			**1125 €**

Costes directos.

Se muestran los costes directos totales del proyecto que supone la suma de los costes expuestos anteriormente en el documento.

TABLA. 6.6. COSTES DIRECTOS.

Concepto	**Coste**
Personal	10452 €
Hardware	3112,75 €
Software	390 €
Viajes y dietas	1125 €
Coste total	**15079,75 €**

Costes indirectos.

Los costes indirectos para este proyecto incluyen el coste de electricidad, agua, internet y alquiler del establecimiento. Se han estimado un 8,5 % sobre los costes directos necesarios para desarrollar el proyecto, teniendo en cuenta el coste directo total que se muestra en la TABLA. 6.6, el valor de los costes indirectos será de 1281,78 € durante todo el periodo de realización del proyecto.

Costes totales estimados.

Finalmente, se estimaron los costes totales que supone la realización del proyecto, habiendo realizado los cálculos de los diferentes costes anteriormente. En la TABLA. 6.7 se incluirá el IVA.

TABLA. 6.7. COSTE TOTAL ESTIMADO.

Concepto	**Coste**
Costes directos	15079,75 €
Costes indirectos	1281,78 €
IVA (21%)	3435,92 €
Coste total estimado (incluido IVA)	**19769,46 €**

Presupuesto para el cliente.

En esta sección se muestra una estimación del presupuesto que es planteado al cliente. Para realizar el cálculo hay que tener en cuenta los costes calculados anteriormente, añadiendo un riesgo, marcando así la variabilidad de los costes. Este riesgo se estipula en un 12,5%.

Finalmente, se incluye un porcentaje de un 10% correspondiente a los beneficios. Estos beneficios son la estimación de lo que genera el trabajo al ser desarrollado.

Estos dos porcentajes de riesgo y beneficios se han estipulado en función de otros trabajos realizados anteriormente. Finalmente, en la TABLA. 6.8 se muestra el presupuesto total que se presenta al cliente, con sus correspondientes costes.

TABLA. 6.8. PRESUPUESTO PARA EL CLIENTE.

Concepto	**Coste**
Costes directos	15079,75 €
Costes indirectos (8,5% directos)	1281,78 €
Coste total sin riesgo	16361,53 €
Riesgo 12,5%	2045,20 €
Coste total sin beneficios	18406,72 €
Beneficios 10%	1840,68 €
Coste total sin IVA	20247,40 €
IVA 21%	4251,96 €
Coste total estimado con IVA	**24499,35 €**

6.4. Dedicación real.

Este apartado muestra el tiempo realmente invertido en el proyecto y la desviación de días en cada tarea respecto a la planificación inicial establecida.

La duración del proyecto ha sido de 125 días, lo que supone una diferencia de 5 días más de lo que se estimó en la planificación inicial. Este incremento de tiempo es debido a diferentes motivos. En primer lugar, la dificultad de dedicar las horas estimadas al día que se señalaron en la planificación, a causa de la realización de otras tareas universitarias, por tanto, no se ha podido dedicar el mismo número de horas todas las semanas. El segundo motivo, es la llegada de dos de los sensores defectuosos, se tuvo que tramitar la devolución, lo que causó un retraso en la fase tres del proyecto. En la TABLA. 6.9 se muestra el porcentaje de desviación en cada tarea.

TABLA. 6.9. DEDICACIÓN REAL.

Fase del proyecto	**Tarea**	**Fecha de inicio**	**Fecha de fin**	**Desviación**	**Días**
Primera	Investigación de proyectos similares.	26/01/2021	04/02/2021	-10%	13
	Estudio del estado del arte.	05/02/2021	07/02/2021	-33%	
Segunda	Implementación LIDAR.	08/02/2021	14/02/2021	+40%	11
	Procesado de los datos LIDAR.	15/02/2021	17/02/2021	+50%	
Tercera	Investigación de nuevos sensores.	18/02/2021	24/02/2021	0%	25

	Implementación e integración de los sensores.	25/02/2021	14/03/2021	+600%	
Cuarta	Transformación de los datos LIDAR	15/03/2021	08/04/2021	0%	25
Quinta	Pruebas.	24/3/2021 09/04/2021	07/04/2021 23/04/2021	0%	24
	Análisis de los datos.	24/04/2021	02/05/2021	+12.5%	
Sexta	Creación del documento.	3/05/2021	30/05/2021	-32.5%	27

Se puede observar en la TABLA. 6.9 que la dedicación real difiere un 4,16% más respecto a la planificación inicial, lo que supone un incremento de 5 días, la tarea que más desviación sufrió es la de implementación e integración de los sensores por el motivo que ya ha sido comentado.

6.5. Coste real.

En este apartado se muestran los cálculos de los costes reales y se realizará la comparación con los costes estimados.

El tiempo requerido para la finalización del proyecto ha sido de 125 días de los cuales 93 han sido laborables, aunque debido a los motivos explicados en el apartado 6.4 “Dedicación real”, la dedicación media diaria es de en torno a 8 horas, por lo tanto, hace un total de 744 horas aproximadamente, manteniendo el sueldo bruto de 10 € por hora, además de aplicar el coste de la seguridad social dejan un total de 10764 € en el coste del personal. El resto de costes mostrados en la sección 6.3 se mantienen exceptuando los costes de viajes y dietas y los costes indirectos que se ven afectados.

El coste real del proyecto aplicando el IVA es de un total de 25067,15 €, lo que supone una diferencia de 567,80 € respecto a la estimación de costes. La diferencia es debido únicamente al incremento del tiempo requerido para la finalización del proyecto.

BIBLIOGRAFÍA.

[1] J. S. Kulchandani y K. J. Dangarwala, «Moving object detection: Review of recent research trends», en *2015 International Conference on Pervasive Computing (ICPC)*, ene. 2015, pp. 1-5. doi: 10.1109/PERVASIVE.2015.7087138.

[2] «Drivable space characterization using automotive lidar and georeferenced map information». https://ieeexplore.ieee.org/abstract/document/6232252/?casa_token=N2aPighaNkoAAAAA:r7RY6OyAFR68tpBjXN9BuwAES74iMstET4dblE5VRCPexglEpGSQB78GFjh9yG7q4gJC89B02Q (accedido may 25, 2021).

[3] Z. Zou, Z. Shi, Y. Guo, y J. Ye, *Object Detection in 20 Years: A Survey*. 2019.

[4] L. Zheng, P. Zhang, J. Tan, y F. Li, «The Obstacle Detection Method of UAV Based on 2D Lidar», *IEEE Access*, vol. 7, pp. 163437-163448, 2019, doi: 10.1109/ACCESS.2019.2952173.

[5] J. Moras, F. S. A. Rodríguez, V. Drevelle, G. Dherbomez, V. Cherfaoui, y P. Bonnifait, «Drivable space characterization using automotive lidar and georeferenced map information», en *2012 IEEE Intelligent Vehicles Symposium*, jun. 2012, pp. 778-783. doi: 10.1109/IVS.2012.6232252.

[6] «LIDAR - Wikiwand». https://www.wikiwand.com/es/LIDAR (accedido abr. 01, 2021).

[7] T. Taipalus y J. Ahtiainen, «Human detection and tracking with knee-high mobile 2D LIDAR», en *2011 IEEE International Conference on Robotics and Biomimetics*, dic. 2011, pp. 1672-1677. doi: 10.1109/ROBIO.2011.6181529.

[8] E. Heiden, Z. Liu, R. K. Ramachandran, y G. S. Sukhatme, «Physics-based Simulation of Continuous-Wave LIDAR for Localization, Calibration and Tracking», en *2020 IEEE International Conference on Robotics and Automation (ICRA)*, may 2020, pp. 2595-2601. doi: 10.1109/ICRA40945.2020.9197138.

[9] «Types of Clustering Algorithms in Machine Learning With Examples», *Blogs & Updates on Data Science, Business Analytics, AI Machine Learning*, jul. 05, 2020. https://www.analytixlabs.co.in/blog/types-of-clustering-algorithms/ (accedido abr. 03, 2021).

[10] «Wikimedia Snapshot». Accedido: may 27, 2021. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:SLINK-Gaussian-data.svg

[11] «Wikimedia Snapshot». Accedido: may 27, 2021. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:KMeans-Gaussian-data.svg

[12] «Wikimedia Snapshot». Accedido: may 27, 2021. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:DBSCAN-Gaussian-data.svg

[13] «Wikimedia Snapshot». Accedido: may 27, 2021. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:EM-Gaussian-data.svg

[14] «Fuzzy C-Means Clustering - MATLAB & Simulink - MathWorks España». https://es.mathworks.com/help/fuzzy/fuzzy-c-means-clustering.html (accedido abr. 13, 2021).

[15] «ML | Handling Imbalanced Data with SMOTE and Near Miss Algorithm in Python», *GeeksforGeeks*, jun. 28, 2019. https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/ (accedido may 25, 2021).

[16] F. Castanedo, «A Review of Data Fusion Techniques», *Sci. World J.*, vol. 2013, p. e704504, oct. 2013, doi: 10.1155/2013/704504.

[17] «Durrant-Whyte: Sensor models and multisensor integration - Google Académico». https://scholar.google.com/scholar_lookup?title=Sensor%20models%20and%20multisensor%20integration&author=H.%20F.%20Durrant-Whyte&publication_year=1988 (accedido abr. 13, 2021).

[18] B. V. Dasarathy, «Sensor fusion potential exploitation-innovative architectures and illustrative applications», *Proc. IEEE*, vol. 85, n.º 1, pp. 24-38, ene. 1997, doi: 10.1109/5.554206.

[19] R. C. Luo, Chih-Chen Yih, y Kuo Lan Su, «Multisensor fusion and integration: approaches, applications, and future research directions», *IEEE Sens. J.*, vol. 2, n.º 2, pp. 107-119, abr. 2002, doi: 10.1109/JSEN.2002.1000251.

[20] F. E. White, «Data Fusion Lexicon»:, Defense Technical Information Center, Fort Belvoir, VA, oct. 1991. doi: 10.21236/ADA529661.

[21] H. Du, W. Wang, C. Xu, R. Xiao, y C. Sun, «Real-Time Onboard 3D State Estimation of an Unmanned Aerial Vehicle in Multi-Environments Using Multi-Sensor Data Fusion», *Sensors*, vol. 20, n.º 3, Art. n.º 3, ene. 2020, doi: 10.3390/s20030919.

[22] «BOE.es - BOE-A-1999-23750 Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.» https://www.boe.es/buscar/act.php?id=BOE-A-1999-23750&tn=1&p=20181206 (accedido abr. 01, 2021).

[23] «BOE.es - BOE-A-1996-8930 Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia.» https://www.boe.es/buscar/act.php?id=BOE-A-1996-8930 (accedido abr. 11, 2021).

[24] 14: 00-17: 00, «ISO 6709:2008», *ISO*. https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/03/92/39242.html (accedido abr. 06, 2021).

[25] «IEEE SA Search». https://standards.ieee.org/search-results.html?q=IEEE+802.11 (accedido abr. 10, 2021).

[26] Y. Zhang y X. Shen, «Direct georeferencing of airborne LiDAR data in national coordinates», *ISPRS J. Photogramm. Remote Sens.*, vol. 84, pp. 43-51, oct. 2013, doi: 10.1016/j.isprsjprs.2013.07.003.

[27] «Conexión GPIO de Raspberry Pi 3», *Electrónica y ciencia*. http://electronicayciencia.blogspot.com/2016/11/conexion-gpio-de-raspberry-pi-3.html (accedido jun. 19, 2021).

[28] «Escáner láser de medición - LIDAR Sensores de medición Productos Fegemu Solutions». https://www.fegaut.com/es/productos/sensores-de-medicion-escaner-laser-de-medicion-lidar/11-135/ (accedido jun. 19, 2021).

[29] «How to Use a GPS Receiver With Raspberry Pi 4 | Raspberry Pi», *Maker Pro*, feb. 03, 2020. https://maker.pro/raspberry-pi/tutorial/how-to-use-a-gps-receiver-with-raspberry-pi-4 (accedido jun. 19, 2021).

[30] «¿Cuánto Cobra un Ingeniero? (Sueldo 2021) | Jobted.es». https://www.jobted.es/salario/ingeniero (accedido jun. 17, 2021).

ANEXO A. SUMARY IN ENGLISH.

1. Introduction.

Computer technology is linked to many areas of knowledge, a clear example is object detection, which is closely related to machine learning techniques. Object detection has numerous applications in different areas, and it is the starting point of the present work. The aim is to carry out an investigation on the detection of moving objects with LIDAR (*Light Detection And Ranging*) sensor. This technology is continuously improving, to the point of detecting moving objects in different environments, however, it still presents difficulties and problems.

1.1. Motivation.

Moving object detection is defined by Jaya S. Kulchandani and Kruti J. Dangarwala [1] as the task of identifying the physical motion of an object in a given region or area. In the last decades, moving object detection has undergone great advances, being recognized as a crucial step in a wide range of applications such as, robotics, vehicle automation, video surveillance and numerous fields within industry, where particular objects such as people, animals, vehicles or others must be detected. The detection of moving objects has proven to be a challenging problem due to different reasons such as the classification of shadows as objects, the background of a moving scene, illumination changes and unknown maps or environments, where dynamic objects partially disappear from the scene, i.e., between successive observations static objects that have not disappeared from the scene can be detected.

The realization of this work allows to contribute in the field of moving object detection, focused on discovering dynamic objects from point clouds obtained by a LIDAR sensor with 2D scanner (*2 Dimension*). In addition, the system is combined with other sensors such as RGB (*Red, Green, Blue*) camera, GPS (*Global Positioning System*) and IMU (*Inertial Measurement Unit*), being able to perform a transformation of the data collected by the laser sensor to a world geodetic coordinate system. At present there are numerous researches on object detection based on the above mentioned sensors, as shown in the article on object detection with small UAVS (*Unmanned Aerial Vehicles*) equipped with a LIDAR, published by the American Institute of Aeronautics and Astronautics [2].

The relevance of research on the detection of moving objects can range from the creation of the device that performs such detection to the study of different techniques and mechanisms to achieve this goal. This paper shows the evolution of moving object detection and the great potential it has for the development and evolution of different applications.

1.2. Problem formulation.

The detection of moving objects with LIDAR sensor is used in many fields, thanks to the scanning of the laser sensor detailed point clouds of the scene are recorded, these

point clouds are generated with the distance from the sensor to the object, obtaining a negligible error. However, in the point clouds we can find noise that hinders the detection of moving objects and can detect a movement where it has not occurred. One of the issues to be addressed in the detection of moving objects is to differentiate between the motion of the objects in the scene and the motion of the observer.

The detection of moving objects in applications such as autonomous driving, robotics, navigation and obstacle detection in UAVS, where the displacement of the system is inevitable have a fundamental factor the absolute positioning, because the point cloud obtained by the LIDAR has a relative positioning.

Keeping these problems in mind is important for the realization of a moving object detection system. These problems are studied in the literature.

1.3. Objectives.

The present work has the following objectives:

I. Detect moving objects with a LIDAR sensor.
   - Perform a clustering, segmentation of the data obtained by the laser sensor, applying different algorithms.
   - Implement a system able to perform a point cloud association and measure the error level of the system in order to be evaluated.

II. Implement a system capable of collecting the data scanned by a LIDAR sensor connected to a Raspberry Pi.
   - Implement a control system to correct the scanning angle.
   - Filtering the measurements recorded by the LIDAR.
   - Real-time visualization of the generated point cloud.

III. Implement a computational system capable of collecting in real time the data being obtained by the laser sensor, camera, GPS and the module that integrates accelerometer, gyroscope, magnetometer and barometer, to be later processed in deferred.
   - Identify and store the images belonging to each scene scanned by the laser.
   - Obtain and store the current position where the detection is being performed, together with the data captured by the module that integrates accelerometer, gyroscope, magnetometer and barometer.
   - Store in a file all the data to be able to be treated in deferred.

IV. Implement a computational system capable of transferring the information recorded by a LIDAR sensor to a geodetic coordinate system WGS-84 (World Geodetic System 1984). In turn, the computational system will be able to combine with the second objective.
   - Redefine the LIDAR measurements to a local ENU (East North Up) reference system, with the X-axis pointing east, the Y-axis pointing north and the Z-axis up.

- Transfer the ENU local reference system to the WGS84 system via ECEF (Earth-Centered, Earth-Fixed).
- Implement a real-time computational system capable of simultaneously performing the collection of data obtained by the different sensors and performing the transformations outlined above.

V. Perform processing of the data collected by the different sensors and carry out data analysis.
   - Implement a system that allows to visualize 2D point clouds with their corresponding image.
   - Implement a system that facilitates the visualization of WGS84 geodetic data on a map.
   - Detect the movement between adjacent point clouds.

1.4. Structure of the document.

In this section, the established structure of the present work on the detection of moving objects using a LIDAR sensor is detailed. The structure of the paper is divided into 6 distinguished chapters.

- Chapter 1. Introduction: It begins with an introduction of the work, where the motivation for the realization of this project is exposed. It continues with a section on the existing problems in the detection of moving objects. Finally, the main objectives of the project are presented, focusing on how to approach the project and the points that are addressed.
- Chapter 2. State of the art: The document continues with a chapter on the state of the art, which is divided into several sections. It begins with a section where a tour of the history of object detection is made, where works related to motion detection and other solutions based on Computer Vision are exposed. It continues by highlighting the types of LIDAR sensors and their main characteristics, showing research related to the sensor model used in this project. It continues with a section on the main clustering, segmentation and point cloud association algorithms used in the literature. Subsequently, a section on data fusion, the main techniques are shown along with projects and applications where a multi-sensor data fusion is performed with sensors similar to those used in the project. This chapter contains a section indicating the different legislative aspects that affect the development of this work, as well as the main related standards and norms. Finally, the chapter concludes with an analysis of the socio-economic environment, evaluating how the detection of moving objects affects the current society and economy.
- Chapter 3. Proposal: This chapter presents the proposal of the work on the detection of moving objects with LIDAR sensor. It begins with a section where a detailed description of the system is made. Once the system has been described, the hardware and software technologies used for the development of the project are presented. Finally, the connections of the designed device are explained.

- Chapter 4. Experimentation and results obtained: This chapter presents a series of experiments and the results obtained in them. In addition, a comparative system for the verification of the motion in the system is presented.
- Chapter 5. Conclusions and future work: This chapter presents two distinct sections. The first one describes the objectives achieved, which are explained in Chapter 1, and the conclusions of the development of the project. Subsequently, a section on future lines of the project is shown, proposing possible improvements.
- Chapter 6. Project analysis: Finally, a chapter on the analysis of the project is presented, showing the initial planning of the work and its budget, where the variations in time and costs that have been taken in the development of the project are shown, indicating the error made in the initial estimate and its correction to achieve the goal in the delivery time.

2. State-of-the-art.

In this chapter, the current situation is presented together with the problems we are facing regarding the detection of moving objects. During the last decades, continuous research and studies have been carried out, leading to great advances at present. The analysis and detection of moving objects is closely related to real-time applications such as obstacle detection, autonomous navigation and data collection that provide information about static and dynamic objects in a scene.

The use of LIDAR sensors for this purpose has proven to be efficient and accurate, obtaining detailed point clouds of the environment where such detection is performed. However, the generation of noise in the point clouds can give erroneous solutions, detecting a moving object where there is none. Unlike digital imaging with cameras or other imaging sensors that employ an optical filter, LIDAR is a sensor that is very susceptible to sampling objects, producing considerable uncertainty about the scene where the scan is being performed. The spatial resolution in the vertical direction of a LIDAR is a problematic task to obtain the shape of the object, only being able to acquire information at a certain height. In contrast, horizontal scanning lasers have the difficulty of measuring points that are occluded behind other points, adding to the motion and sway of the scanner.

2.1. History of object detection.

This section presents the trajectory that object detection has undergone over time, showing two periods that marked a historical fact in object detection, exposing techniques based on computer vision as another variant to the detection of moving objects. In addition, it shows works related to other techniques based on LIDAR sensor close to the direction of the present work.

Object detection has been one of the most challenging problems as a subfield of computer vision. Its evolution in the last decades can be considered paramount in the history of Deep Learning, highlighting significantly the evolution it has had since 1990.

Two historical periods could be differentiated: "traditional object detection pre-2014 (pre-AlexNet)" and "deep learning-based object detection post-2014 (post-AlexNet)" as defined by Zhengxia Zou and Zhenwei Shi in the following paper [3].

Traditional object detection:

- **Viola Jones detectors**, they go back 18 years approximately, they managed for the first time to detect in real time faces, thanks to P. Viola and M. Jones, hence the name of the algorithm "Viola-Jones Detector". This algorithm is based on Haar functions, where the properties to be recognized are included in rectangular fragments where the sum of dark and light pixels is performed.
- **HOG detector**, "*Histogram of Oriented Gradients*" in 2005 its use was generalized by B. Triggs and Dalal. Triggs and Dalal, focused on the problem of pedestrian detection and was later generalized to detect different classes of objects. This detector performs a rescaling of the input image several times by calculating a HOG descriptor for each scale, keeping the input size unchanged.

- **DPM**, "*The Deformable Part-based Model*" in 2008 was proposed based on the HOG detector of B. Triggs and Dalal in which a filter is used to represent an object category. DMP is based on lightly supervised learning, using a model consisting of a root filter and a set of part filters.

ImageNet in 2012, highlighting the work of AlexNet, caused a significant evolution due to deep learning networks, being able to learn more robust and complex computer vision features.

Deep learning-based object detection:

- **Dual-stage detection**, the first stage the regions of interest of objects are developed, while the second stage the regions are classified and objects are located by bounding box regression. We can highlight R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN, FPM and SPP-net. This stage remains a reference model in object detection at present.
- **Single stage detection**, unlike the previous stage this detector bypasses the stage of developing regions, directly performing the classification and location of bounding boxes. It gained great importance with the YOLO proposal and later with the arrival of SSD and RetinaNet.

These two periods presented are focused on the detection of objects with computer vision systems opening a range of possibilities to address the problem posed. It can be stated that these periods have given rise to other techniques that are not focused on computer vision. The appearance of other techniques is also due to the limitations of computer vision and the difficulties presented by the optical filter sensors, which are sensitive to changes in illumination, making detection difficult at certain times.

An example of these techniques can be the obstacle detection method with 2D LIDAR in a UAV as defined by Lanxiang Zheng in the following article [4], using different sensors to complement the LIDAR data and to be able to guarantee a safe flight by obtaining information of the environment and the different obstacles that appear in the flight. This obstacle information can be posed as global obstacle information or local information. LIDAR is one of the most useful sensors for obstacle detection, because it is not as affected as other optical sensors by adverse weather conditions, light changes or cloud cover. In this project, an IMU and GPS are used to correctly map the point clouds into a global coordinate system. The tasks performed to detect obstacles are generally to obtain information with the LIDAR sensor about the obstacles present in the flight environment, pre-process the clouds being recorded, and finally perform a clustering of the point clouds. LIDAR has an extrinsic motion that is the reason why the point clouds may appear distorted, to correct this problem the author proposes a method of compensation of the position of the data using the ICP (Iterative Closest Point) algorithm to find the LIDAR movements between adjacent point clouds. Another way to solve this problem is to use INS (Inertial Navigation System) to estimate the position of the LIDAR in the record of each point. With the help of an IMU and a GPS the transformation of the points to a real position in a global coordinate system is performed. Finally, for point cloud clustering the DBSCAN algorithm is used,

a Euclidean clustering method based on point density, dividing the points into different groups and being able to detect obstacles.

Autonomous navigation is challenging due to the fact that moving objects may appear in the environment, so the systems must be continuously performing estimations to configure the navigation space. In the work of Julien Moras, a method for characterizing the drivable space using LIDAR and georeferenced map information is defined [5]. Other techniques based on vision systems have been studied in different projects such as urban navigation or ADAS (Advanced Driver Assistance Systems) using frontal vision to achieve their goal, but these vision systems are sensitive to illumination changes, so in this project we opt for laser-based systems. The method presented is the perception of the environment with the data obtained by a LIDAR sensor as the main sensor for perception and detection of obstacles together with digital maps, the vehicle integrates a LIDAR, GPS and an IMU, the accuracy of the information received from the digital maps depends on the latter two sensors, achieving a limited driving space by the estimation of the information. One of the main tasks dealt with in this project is the transformation of the map to a WGS-84 world coordinate system where the Z axis is pointed upwards so that it is orthogonal to the surface of the reference ellipsoid.

As we have been able to observe in this section there are two different approaches to solve the problem of moving object detection. Finding research related to the topic discussed in this section, providing a range of solutions to solve this problem.

2.2. LIDAR sensor.

LIDAR is a sensor based on laser technology, which allows to know the distance between the sensor and a surface or object, using a pulsed laser light beam. When the laser hits an object, it bounces and moves back to the position where the LIDAR sensor is located. This makes it possible to calculate the distance between the sensor and the object or surface with the difference of the time elapsed between the laser emission and the reception of the bounce, obtaining great precision and a reduced error.

This sensor has different uses and applications today such as land surveying, inspection of power lines, agriculture, mining, robotics and autonomous vehicles among others, being able to highlight a great recognition within the detection of moving objects.

Two different classifications of LIDAR sensors can be distinguished at present, depending on the type of laser or scanning as detailed in the following article [6].

Depending on the type of laser we can highlight:

- **Pulsed LIDAR**. It works by emitting pulses of laser light and is used to measure the distance between the sensor and the surface, measuring the time it takes the laser from the time it is emitted to the time it is received.
- **Phase measurement LIDAR**. It continuously emits a laser beam and at the moment the sensor receives the signal it calculates the difference between the emitted and reflected phase.

Depending on the type of scanning we can highlight:

- **Linear**. It contains a mirror that rotates and deflects the laser beam. Generating a series of parallel lines on the surface. One of the main disadvantages of this type of scanning is that as the mirror is continuously rotating in one direction it does not ensure that measurements are always taken.
- **Fiber optics**. Through a series of small mirrors and with a fiber optic cable, the laser is deflected to the lateral fibers that are mounted around the axis. This mechanism generates a footprint with a series of circular shapes overlapping each other. The main disadvantage is that the scanning angle is much smaller.
- **Elliptical**. The laser is deflected by two mirrors performing an elliptical scan. This scan increases the difficulty, due to the use of two angle gauges.
- **Zigzag**. Contains a mirror rotating in both directions thus producing a zigzag scan. Unlike the line scan this one is always making measurements, but in the areas near the lateral limits the density of points collected is higher than in the center.

The laser sensor used in this project is the Hokuyo URG-04LX model, a low-cost 2D LIDAR designed for robotics applications. This rotational sensor produces a single beam as it rotates the mirror inside while taking measurements with the phase shift between the emitted laser beam and its reflection, obtaining a high volume of data produced. The sensor model used is well recognized in the field of robotics due to its small size and the performance it offers despite its low cost. In the work of Tapio Taipalus and Juhana Ahtiainen [7], the detection and tracking of people from a mobile robot integrating the Hokuyo URG-04LX 2D LIDAR is defined, solving the problem of detecting people in crowded environments.

The physical fundamentals involved in this 2D LIDAR sensor are the interaction of the laser beam with the surface and the process of measuring the distance as defined in the following paper [8]. Interaction of the laser with the surface, when the laser is incident on a surface point it can be transmitted, reflected and scattered, depending on the type of surface and its angle. The distance for the sensor used is calculated with the phase difference between the light transmitted from the sensor and that reflected by the surface.

$$L = \frac{1}{2} * \left\{ \frac{\emptyset}{\left(\frac{F}{v} * 2\pi\right)} \right\} \tag{A.1}$$

L is the distance to the surface from the sensor in millimeters, $\emptyset$ is the phase difference in radians, F the frequency in Hz and v the speed of light in mm/sec.

We can find other optical sensors that could replace the laser sensor proposed in this work, but we find some disadvantages compared to it, such as sensitivity to illumination changes, as discussed above.

2.3. Other sensors.

Other sensors that play a role in a moving object detection system and complement the information obtained by the LIDAR sensor are (GPS, barometer, IMU). The GPS sensor provides the latitude and longitude of the system at all times, the barometer records the altitude of the system above sea level and an IMU containing (accelerometer, gyroscope and magnetometer), being able to perform the transformation of the information recorded by the LIDAR sensor to a geodetic coordinate system.

- **GPS**. This sensor continuously returns the position of the system based on the mathematical principle of trilateration, calculating the distance to the satellite based on the time it takes for the signal to travel. Satellites send radio signals containing information about their radius and the time instant at which they were sent, these signals travel at the speed of light. The receiving sensor records the instant of time it took to arrive and calculates the distance with $d = c(t_2 - t_1)$ (A.2). The sensor needs to know the distance to four satellites to calculate the position on the earth in three dimensions.
- **Barometer**. The digital barometric sensor provides the altitude at which the device is above sea level accurately. To calculate the altitude, the temperature and atmospheric pressure are needed, because the pressure drops as we gain altitude. The sensor used in this project has a measurement range of 300 hPa to 1110 hPa which is equivalent to an altitude of -500 meters to 9000 meters above sea level. With an accuracy of 1 hPa and an altitude error in 1 meter difference [6].
- **IMU**. The inertial measurement unit used in this work has three different sensors (accelerometer, gyroscope and magnetometer). The gyroscope measures the angular velocity, which is the number of degrees rotated per second. The accelerometer gives us the linear acceleration. Finally, the magnetometer obtains the magnetic field. All sensors record information in the three components (x, y, z).

2.4. Algorithm.

In this section we present some algorithms that can be applied as a solution in the work. We will define the main clustering, segmentation and point cloud association algorithms applicable to the information recorded by the LIDAR sensor.

2.4.1. Clustering algorithms.

The main objective of clustering algorithms is to group objects in a dataset according to certain characteristics and similarities, usually based on distance. There are a multitude of clustering algorithms, each one adapting better to a particular data distribution, addressing different issues such as scalability, attributes, dimension, distance, noise, among others. The different types of clustering defined [9] in this article are:

- Connectivity-based clustering is a type of clustering that establishes a predefined hierarchy of clusters, being able to decompose the data according to this hierarchy, thus obtaining the different clusters.
- Clustering based on centroids, is one of the simplest and most effective techniques to create clusters, it is characterized by being represented by a central vector where the clusters will be respectively grouped according to the proximity to these vectors.
- Density-based clustering detects regions with the highest concentration of data and separates them by areas with little or no data. Data or points that are not contained in the clustering are labeled as noise.
- Distribution-based clustering, this type of clustering takes into account a completely different metric such as probability, it is closely related to statistics, data sets are grouped according to the probability of corresponding to such data distribution.
- Fuzzy clustering, objects belong to a certain group according to the degree of confidence or membership, this varies between 0 and 1. It is used with a data set where the variable has a high level of overlapping.

Having seen what a clustering algorithm is and the different types that exist, we can see that density-based algorithms are the most suitable for 2D point clouds collected by the LIDAR sensor. Point cloud clustering is essential for scene reconstruction and object detection and identification. Density-based algorithms are capable of detecting clusters of arbitrary shapes such as those generated by LIDAR. In the clustering of the data obtained by the laser sensor we can make use of different clustering algorithms:

- **HDBSCAN** is a clustering algorithm that has the ability to detect arbitrarily using a flat clustering technique based on clustering stability, without requiring as many parameters as its more direct competitor DBSCAN. The main parameter needed to perform clustering is (min_cluster_size), which indicates the minimum size of a group. This algorithm makes use of a variety of distances to perform a separation into regions of varying densities to noise. While processing, this algorithm creates a hierarchy of clusters based on different density thresholds, obtaining the flat cluster. Therefore, the number of clusters is defined by the characteristics of the data itself.
- **OPTICS** is an algorithm very similar to DBSCAN, seeking to solve the main weaknesses of the latter. This algorithm looks for central samples with high density and from these samples the clusters are formed. This algorithm has a hierarchy of clusters for a variable radius of neighbors. The data are ordered linearly so that the closest points become neighbors, a distance is stored for each point in the dataset, representing the distance needed for the two points to belong to the same cluster.
- **DENCLUE** is an algorithm that identifies density regions in the data set. This algorithm creates a network of regions in the dataset using an influence function,

obtaining points with the same local maximum, thus describing the data that are in the same group. It uses a method that combines hierarchical clustering and data partitioning.

- **DBSCAN**. This algorithm is used in this project to establish different groups in point clouds, it is a density-based clustering algorithm whose operation is simple, a point p forms a density cluster with point q if p is within the neighborhood radius ε defined as $N_\varepsilon(\rho) = \{q | \, d(p,q) \leq \varepsilon\}$ and also has the minimum number of neighbors required to form a $MinPts$ group. A point p is directly connected by density from a point q if $p \in N_\varepsilon(q)$ y $|N_\varepsilon(p)| \geq MinPts$. We define our set of points as $x = \{x_1, x_2 \ldots . x_n\}$, DBSCAN needs two parameters, the neighborhood radius and the minimum neighbors to form a group. First an arbitrary point that has not been visited is chosen, the neighborhood of that point is obtained with ε and it is calculated if there are the minimum number of neighbors in the radius of that point, it will be labeled as chosen if these conditions are met, otherwise it is labeled as noise. This process is repeated until all data groups have been defined or all points have been labeled. Groups labeled with "-1" are outliers.

2.4.2. Segmentation.

Once the clustering technique has been performed, the point cloud is grouped according to the density of the data set. We proceed to the segmentation of the groups, establishing homogeneity between adjacent clusters with the aim of being able to correctly make an association of the point clouds, therefore, if there is an imbalance in the different groupings, a predictive model is applied to synthesize points and obtain the same dimensionality in both sets, for this a technique of oversampling of synthetic minorities SMOTE (*Synthetic Minority Oversampling Technique*) is used. This technique allows balancing the number of points in each cluster, synthesizing new points in the unbalanced data set, using linear interpolation of the minority group. The steps involved in this technique are as follows [15]:

- This method finds the existing distances between the majority data set in which a subsampling has been performed and has an imbalance, using the Euclidean distance.
- The k points of the majority class that have the smallest distance to the unbalanced class are selected.
- If there are n points in the minority class group, the result obtained is given by the following operation, $n * k$ $(A.3)$ points of the majority class.

This technique provides us with a correct balancing of the data sets.

2.4.3. Point cloud association algorithm.

Once the clustering and segmentation of the point clouds has been performed, it is essential to associate the clusters obtained between pairs of adjacent point clouds in order to obtain the correlation between the different areas of a scene. An algorithm that provides a solution to this problem is ICP. This iterative algorithm is used to minimize the difference between two-point clouds, obtaining an optimal correlation and mapping.

This method is based on applying a rotation and translation between the pairs of points closest to each other, using the Euclidean distance. This process is iterative until the error is minimized.

The device records successive point clouds of an environment for delayed processing. After clustering and segmentation of the data, the point clouds are associated to obtain a correlation between the different areas of the scenes and to observe variations in the objects, detecting possible movements.

The closest point iteration algorithm is characterized by finding an optimal mapping between two-point clouds based on a rotation and translation that minimizes the distance between the corresponding point pairs of the source and target point clouds. The input sets are A (source point cloud) and B (target point cloud) with P_A y P_B points, both sets have to have the same dimension in order to apply this algorithm.

2.5. Multi-sensor data fusion.

Multi-sensor data fusion is the process of combining data collected from several different sensors to provide a complete and accurate description of an environment or process. It is currently used in different areas such as object recognition, environment mapping, traffic control, robotics, vehicle automation, among others.

Data fusion is an area that involves several fields and it is complicated to establish a clear and precise classification. Therefore, we can divide it into different techniques and methods according to the following criteria as defined by Federico Castanedo [16]:

- According to the existing relationships between the input data sources, as proposed by Durrant-Whyte [17]. These relationships can be cooperative, complementary or redundant data.
- Attending to the types of input/output data and their characteristics, as proposed by Dasarathy [18]. The classification is composed of 5 different categories.
- Depending on the level of abstraction of the data being used in the data fusion, such as measurements, decisions, signals or features. This classification caters to multi-sensor data fusion as proposed by Luo [19].
- According to the levels of data processing, they were proposed by Joint Directors of Laboratories [20]. The classification consists of 5 different levels.
- Classification based on the type of architecture, with 3 different groups: centralized, distributed and decentralized.

The present project that is being carried out makes use of 4 different sensors, LIDAR, camera, GPS and a GY-91 module that integrates accelerometer, gyroscope, magnetometer and barometer. Currently there are different research centers and organizations that perform studies of multi-sensor data fusion with these same sensors, as defined by the authors of this article [21], which use multi-sensor data fusion to estimate the 3D state of the UAV. To perform the estimation, a fusion of the data

collected by IMU, GPS, LIDAR, an optical flow sensor and a depth camera is used to obtain the state of the UAV at all times.