

MÉTODO PARA LA RECONSTRUCCIÓN TRIDIMENSIONAL DE ENTORNOS

CAMPO TÉCNICO

5 La presente divulgación se relaciona con métodos y sistemas para la reconstrucción digital de entornos o escenas. Particularmente la presente divulgación se relaciona con métodos y sistemas para reconstrucción tridimensional de entornos con base en la información de sensores y métodos de aprendizaje profundo.

10 DESCRIPCIÓN DEL ESTADO DE LA TÉCNICA

La reconstrucción digital de entornos o escenas es un proceso que consiste en recrear lugares físicos en formato tridimensional utilizando información proveniente de sensores como cámaras, sensores de LiDAR, y sensores de RADAR entre otros. Para llevar a cabo la
15 reconstrucción se pueden recopilar datos del espacio real mediante fotografías desde distintos ángulos (fotogrametría), sensores LiDAR y RADAR que miden distancias y permiten regenerar la escena, adicionalmente, estos datos se pueden combinar mediante técnicas de fusión de datos para obtener las mejores características de cada sensor. Con la información recopilada se construyen modelos 3D detallados usando software de aprendizaje profundo
20 los cuales procesan los datos fusionados para identificar patrones, clasificar objetos y refinar la reconstrucción de las escenas en 3D. Estos algoritmos permiten mejorar la precisión en la detección de elementos críticos como vehículos, personas y obstáculos, adaptándose a diferentes condiciones ambientales y escenarios.

25 La reconstrucción digital de entornos se puede usar en áreas como la conducción autónoma, donde la identificación precisa de vehículos, peatones y obstáculos permite tomar decisiones seguras y en fracciones de segundo; en vigilancia y monitoreo ésta tecnología facilita la creación de mapas detallados de grandes áreas, la detección de anomalías y el seguimiento de objetos en movimiento, incluso bajo condiciones adversas como niebla, lluvia o
30 interferencias electromagnéticas.

Por lo anterior, existe la necesidad de métodos y sistemas de reconstrucción de entornos o escenas. Se identifican en el estado de la técnica divulgaciones como US11703566B2, US12190448B2 y CN119339007A, las cuales se relacionan métodos y sistemas de
35 generación de entornos tridimensionales.

El documento US11703566B2 divulga un método y un sistema que, basado en aprendizaje automático, recibe datos de diferentes tipos de sensores, tales como LiDAR y radar, y los asocia con detecciones de objetos basadas en imágenes, generando así una región
40 tridimensional de interés (ROI) para cada objeto detectado.

Específicamente, el sistema de US11703566B2 emplea múltiples tuberías o pipelines independientes según el tipo de sensor involucrado. Cada tubería recibe datos espaciales, nubes de puntos o estimaciones de profundidad monocular asociadas con la detección de
5 objetos, y no espaciales, como atributos específicos generados por cada sensor.

A partir de estos datos, US11703566B2 divulga la generación de mapas denominados *point in / out maps*, que indican probabilidades o valores binarios de asociación de cada punto con un objeto determinado. Posteriormente, estos resultados se utilizan para producir una
10 reconstrucción 3D más precisa, permitiendo mejorar la identificación de objetos especialmente en distancias superiores a 50 metros. En algunos casos, US11703566B2 divulga la inclusión de información no espacial adicional, como atributos generados directamente por los sensores.

15 Asimismo, US11703566B2 describe técnicas para preprocesar los datos provenientes de los sensores mediante componentes de control de vehículo que reciben la información de estos sensores y controlan operaciones del vehículo autónomo, generando trayectorias basadas en técnicas de horizonte móvil (*receding horizon*). Esta información alimenta un controlador proporcional-integral-derivativo (PID), generando señales de control para sistemas de
20 accionamiento del vehículo. El documento US11703566B2 menciona explícitamente el uso de redes neuronales convolucionales (CNNs), incluyendo variantes como R-CNN, ResNet, VGG, DenseNet y PointNet.

Por otro lado, el documento US12190448B2 divulga un método que utiliza redes neuronales profundas (DNN) para estimar estructuras superficiales tridimensionales a partir de datos de
25 sensores, particularmente LiDAR y cámaras. US12190448B2 divulga la generación de conjuntos de entrenamiento tanto en entornos simulados como en entornos reales, recolectando datos sensoriales como imágenes y nubes de puntos LiDAR. Los datos LiDAR se preprocesan mediante técnicas como suavizado temporal, eliminación de valores atípicos
30 (*outliers*), interpolación (triangulación de Delaunay) y acumulación de datos provenientes de múltiples sensores LiDAR para aumentar la densidad y calidad de las nubes de puntos. Este documento menciona la utilización de mapas de profundidad y máscaras de segmentación semántica para generar representaciones densas de estructuras superficiales, particularmente caminos o carreteras, con una alta calidad apta para navegación autónoma.

35 El método de US12190448B2 también contempla la utilización de un estimador de visión estéreo que genera mapas de disparidad, que posteriormente se proyectan a nubes de puntos tridimensionales. A partir de allí, US12190448B2 divulga la generación de puntos relacionados con estructuras particulares, como superficies de carreteras, por ejemplo, y
40 luego se seleccionan utilizando máscaras de segmentación previamente obtenidas mediante

redes neuronales profundas.

El documento CN119339007A por su parte divulga un método adaptativo para modelado tridimensional dinámico de escenas mediante la fusión explícita de datos provenientes de múltiples sensores. Secuencias de video RGB-D (Depth - profundidad), nubes de puntos LiDAR y datos de unidades de medida inercial (IMU). Posteriormente, se realiza un preprocesamiento con técnicas específicas como filtrado gaussiano adaptativo para video, filtrado bilateral para preservar bordes en nubes de puntos, y transformaciones wavelet para eliminar ruido en datos IMU. Estos datos se fusionan empleando técnicas explícitas como el filtro de Kalman extendido, obteniendo una representación tridimensional.

El método de CN119339007A adicionalmente divulga la generación de campos vectoriales dinámicos del entorno a través de redes neuronales convolucionales, que procesan datos de video y nubes de puntos. Además, CN119339007A divulga la utilización de métricas cuantitativas, como errores geométricos, para evaluar continuamente la calidad del modelado tridimensional, adaptando el algoritmo de reconstrucción cuando el error supera un umbral dinámico específico. El método de CN119339007A, además segmenta las escenas, construyendo histogramas de curvatura local, coeficientes de complejidad geométrica y coeficientes de visibilidad mediante proyección de haces de luz para optimizar de forma adaptativa la estrategia de modelado.

Sin embargo, no se evidencia en los documentos citados del estado de la técnica un sistema que contemple el uso de múltiples métodos de aprendizaje y múltiples sensores que permitan llevar a cabo la creación de entornos 3D para objetos cercanos y lejanos que puedan moverse a diferentes velocidades extrayendo las mejores características de los dominios donde actúan, el de radiofrecuencia y el óptico.

BREVE DESCRIPCIÓN

La presente divulgación se refiere a un método para la reconstrucción tridimensional de entornos, que comprende una etapa a) para obtener en una unidad de cómputo un dato de referencia de al menos un sensor, y en una etapa b) el método obtiene unos datos preprocesados ejecutando un proceso de filtrado en una unidad de cómputo a partir de un dato de referencia.

En una etapa c) el método genera unos primeros datos de reconstrucción tridimensional del entorno y objetos ejecutando en la unidad de cómputo un primer proceso de aprendizaje a partir de los datos preprocesados y en una etapa d) se generan unos segundos datos de reconstrucción tridimensional ejecutando en la unidad de cómputo un segundo proceso de aprendizaje a partir de los datos preprocesados.

Luego en una etapa e) se determinan unos datos de métricas estimados de los primeros datos de reconstrucción tridimensional y de los segundos datos de reconstrucción tridimensional ejecutando en la unidad de cómputo un proceso de estimación de métricas, en una etapa f) se
5 selecciona al menos un dato del primer dato de reconstrucción tridimensional o del segundo dato de reconstrucción tridimensional ejecutando un proceso de comparación de los datos métricos estimados del primer dato de reconstrucción tridimensional y del segundo dato de reconstrucción tridimensional en la unidad de cómputo.

10 Finalmente, en una etapa g) se generan unos datos de reconstrucción tridimensional del entorno o escena a partir de al menos un dato del primer o segundo dato de reconstrucción tridimensional seleccionado.

Por otro lado, la presente divulgación también se refiere a un sistema para la reconstrucción
15 tridimensional de entornos que comprende una unidad de cómputo configurada para conectarse con al menos un sensor, en donde la unidad de cómputo está configurada para ejecutar las etapas del método anteriormente descrito.

BREVE DESCRIPCIÓN DE LAS FIGURAS

20

La FIG. 1 muestra un diagrama de flujo de una modalidad del método de reconstrucción tridimensional de la presente divulgación.

La FIG. 2 muestra un diagrama de bloques de la modalidad del método de reconstrucción
25 tridimensional de la FIG. 1.

La FIG. 3 muestra un diagrama de flujo de una modalidad adicional del método de reconstrucción tridimensional de la presente divulgación.

30 La FIG. 4 muestra un diagrama de bloques de la modalidad del método de reconstrucción tridimensional de la FIG. 3.

DESCRIPCIÓN DETALLADA

35 La presente divulgación está dirigida a sistemas y métodos de reconstrucción tridimensional de escenas o entornos con base en datos de sensores que generan una nube de puntos de acuerdo con los objetos en la escena. El método genera unos primeros datos de reconstrucción tridimensional por medio de un primer proceso de aprendizaje y unos segundos datos de reconstrucción tridimensional por medio de un segundo método de
40 aprendizaje. Una vez generados los primeros y segundos datos de reconstrucción

tridimensional se analizan unas métricas de cada uno de los primeros y de los segundos datos para determinar qué datos del primer método de aprendizaje o del segundo método de aprendizaje pueden generar una mejor reconstrucción tridimensional del entorno. A partir de este análisis se seleccionan los datos que proporcionaran una mejor reconstrucción de la
5 escena y se procederá con la reconstrucción de la misma.

Haciendo referencia a las FIGs. 1 y 2, en una modalidad de la divulgación, el método para la reconstrucción tridimensional de entornos, comprende:

- a) obtener en una unidad de cómputo (11) un dato de referencia (20) de al menos
10 un sensor (10);
- b) obtener unos datos preprocesados (21) ejecutando un proceso de filtrado (30) en una unidad de cómputo (11) a partir de un dato de referencia (20);
- c) generar unos primeros datos de reconstrucción tridimensional (22) del entorno y objetos ejecutando en la unidad de cómputo (11) un primer proceso de aprendizaje (31) a
15 partir de los datos preprocesados (21);
- d) generar unos segundos datos de reconstrucción tridimensional (23) ejecutando en la unidad de cómputo (11) un segundo proceso de aprendizaje (32) a partir de los datos preprocesados (21);
- e) determinar unos datos de métricas estimados (24) de los primeros datos de
20 reconstrucción tridimensional (22) y de los segundos datos de reconstrucción tridimensional (23) ejecutando en la unidad de cómputo (11) un proceso de estimación de métricas (33);
- f) seleccionar al menos un dato del primer dato de reconstrucción tridimensional (22) o del segundo dato de reconstrucción tridimensional (23) ejecutando un proceso de comparación (34) de los datos métricos estimados (24) del primer dato de reconstrucción
25 tridimensional (22) y del segundo dato de reconstrucción tridimensional (23) en la unidad de cómputo (1);
- g) generar unos datos de reconstrucción tridimensional (25) del entorno o escena a partir de al menos un dato del primer o segundo dato de reconstrucción tridimensional (22, 23) seleccionado.

30

Se entenderá en la presente divulgación por dato a una representación simbólica que puede ser numérica, alfabética, algorítmica, lógica y/o vectorial que codifica información. Un dato puede tener una estructura o trama compuesta de bloques de caracteres o de bits que representan diferentes tipos de información. Cada bloque se conforma de cadenas de
35 caracteres, números, símbolos lógicos, entre otros.

También un dato puede formarse solo de bits (cadenas en lenguaje binario), formarse de caracteres formados uno a uno por una combinación de bits, formarse a partir de campos, registros o de tablas formadas de campos y registros, o formarse de archivos de intercambio
40 de datos (formatos como csv, json, xls, entre otros). Además, un dato puede ser una matriz

de n filas por m columnas. A su vez, un dato puede contener varios datos.

Por ejemplo, cuando el dato tiene estructura de trama, la trama puede tener un bloque de caracteres de identificación, conocido generalmente como encabezado o “header”, el cual
5 contiene información relacionada con un dispositivo de cómputo o procesador que envía el dato, y puede contener información relacionada con un dispositivo de cómputo o procesador que recibe el dato. Preferiblemente, si dato tiene un formato de trama, la trama contiene bloques relacionados con capas de acuerdo con el modelo de referencia OSI.

10 Asimismo, la trama puede tener un bloque de caracteres de cola (o simplemente cola), o “tail” en inglés, que permite identificar a una unidad de cómputo o servidor que es el fin del dato, es decir, que después de ese bloque ya no se encuentra información contenida en el dato identificado previamente por la unidad de cómputo o servidor con el “header”. Además, el dato tiene entre el bloque “header” y el bloque “tail” uno o más bloques de caracteres que
15 representan estadísticas, números, descriptores, palabras, letras, valores lógicos (e.g. booleanos) y combinaciones de estos.

Obtención de datos de referencia de sensores.

20 En cualquiera de las modalidades del método que incluyen la etapa a), se pueden obtener datos de referencia (20) de al menos un sensor (10). Los sensores (10) pueden ser cámaras RGB, cámaras estéreo, cámaras de profundidad, cámaras de video 3D, cámaras térmicas, sensores ultrasónicos, sensores IMU (*Inertial Measurement Unit* por sus siglas en inglés), sensores ToF (*Time-of-Flight* por sus siglas en inglés), sensores GPS (*Global Positioning*
25 *System* por sus siglas en inglés), sensores LiDAR (*Laser Imaging Detection and Ranging* por sus siglas en inglés), sensores RADAR (*Radio Detection and Ranging* por sus siglas en inglés) y tecnologías de equivalentes que sean conocidas por una persona medianamente versada en la materia y combinaciones de las anteriores.

30 En una modalidad de la divulgación, los sensores (10) pueden ser sensores LiDAR y RADAR. En una modalidad adicional de la divulgación, los sensores (10) pueden ser LiDAR, RADAR e IMU. En una modalidad particular de la divulgación, los sensores (10) pueden ser LiDAR, RADAR, IMU y cámaras de video 3D.

35 Para el entendimiento de la presente divulgación, un sensor LiDAR es un dispositivo que utiliza pulsos de luz láser para medir distancias. El LiDAR envía haces de luz (fotones) y se calcula el tiempo que tarda cada pulso en reflejarse en los objetos y volver al sensor. El sensor LiDAR puede emitir millones de pulsos por segundo en diferentes direcciones, lo cual genera una nube de puntos 3D que permite reconstruir un entorno con nivel de detalle. Este
40 tipo de sensor se puede usar en vehículos autónomos, drones, cartografía 3D, topografía y

reconstrucción de escenas. Debido al uso de láser, el sensor LiDAR tiene alta resolución y precisión. Por otro lado, el rendimiento de un sensor LiDAR puede verse afectado por condiciones climáticas como lluvia o niebla.

5 Por su parte, un sensor RADAR es un dispositivo que utiliza ondas de radio para detectar objetos y determinar su distancia, velocidad y dirección. El sensor RADAR emite ondas electromagnéticas que, al chocar con un objeto, se reflejan y regresan al sensor, permitiendo calcular la distancia mediante el tiempo de retorno y la velocidad gracias al efecto Doppler. El sensor RADAR se puede usar en condiciones de baja visibilidad como niebla, lluvia o
10 oscuridad, por lo que se emplea en aplicaciones como control de tráfico aéreo, sistemas de navegación marítima, meteorología, seguridad militar y vehículos autónomos, aunque su resolución espacial es menor que la de un sensor LiDAR, tienen un mayor rango de acción en cuanto a su alcance y detección de objetos.

15 Por otro lado, un sensor IMU es un dispositivo que combina acelerómetros, giróscopos y, en algunos casos, magnetómetros para medir la aceleración lineal, la velocidad angular y la orientación de un objeto en el espacio. Los datos capturados por el sensor IMU permiten determinar movimientos y cambios de posición de los objetos, lo que lo permite que sea usado en navegación inercial, robótica, drones, vehículos autónomos y dispositivos móviles.

20

Por su parte, una cámara de video 3D es un dispositivo capaz de capturar imágenes en dos dimensiones (ancho y alto) con información de profundidad, permitiendo crear representaciones tridimensionales de la escena. Las cámaras 3D combinan técnicas como visión estéreo (dos lentes que imitan la percepción humana), sensores de luz estructurada
25 (proyección de patrones infrarrojos) o tecnología Time-of-Flight (ToF), que mide el tiempo que tarda la luz (fotones) en regresar desde los objetos. Por medio de esta información se generan mapas de profundidad o modelos 3D en tiempo real.

En la presente divulgación, los sensores LiDAR y RADAR permiten capturar una nube de
30 puntos en diferentes condiciones climáticas y con diferentes niveles de precisión de acuerdo con los objetos presentes en la escena. El sensor IMU permite detectar movimiento en las tres dimensiones, velocidad, orientación y fuerzas específicas de los objetos en el entorno, y la cámara 3D permite obtener información como color, texturas, patrones y profundidad de los objetos en el entorno.

35

En una modalidad de la presente divulgación, los sensores (10) se pueden disponer sobre una estructura de soporte que permite ajustar la dirección de dichos sensores (10) tanto en cabeceo (*pitch*), lo que permite la rotación de los sensores alrededor de un eje vertical y hace que los sensores puedan apuntar hacia arriba y hacia abajo y en guiñada (*yaw*) lo que permite
40 la rotación de los sensores alrededor de un eje horizontal, y hace que los sensores apunten a

la izquierda y a la derecha.

Por otra parte, la unidad de cómputo (11) obtiene un dato de referencia (20) de al menos un sensor (10). Cada sensor (10) puede capturar diferentes tipos de datos que al combinarse
5 permiten generar una representación tridimensional del entorno. El LiDAR puede estimar distancias mediante la fórmula de distancia usando el tiempo de vuelo de los pulsos láser recibidos, y se genera la nube de puntos 3D en un sistema de coordenadas espaciales (x, y, z) de cada superficie reflejada, y también puede capturar la intensidad de la señal que permite identificar diferentes tipos de materiales. El RADAR puede capturar distancias, velocidades
10 relativas y ángulos de detección, lo que le permite detectar objetos en condiciones de baja visibilidad. La IMU permite capturar datos de aceleración lineal, velocidad angular y campo magnético, lo que le permite estimar la posición, orientación y trayectoria de objetos en el entorno y asiste en el preprocesamiento de los datos para la fusión de la información suministrada por los sensores. La cámara 3D por su parte captura información visual como
15 color, texturas patrones y profundidad, lo cual permite complementar los datos obtenidos por el LiDAR y el RADAR y llevar a cabo la reconstrucción de la escena.

Preprocesamiento de los datos.

20 Una vez se han obtenido en la unidad de cómputo (11) los datos de referencia (20), se lleva a cabo un proceso de filtrado (20) a partir de los datos de referencia (20).

En una modalidad de la divulgación, el proceso de filtrado (20) puede comprender un filtro de Kalman, un filtro de Kalman extendido o un filtro de Kalman “*unscented*”. El proceso de
25 filtrado (20) también puede comprender métodos de filtrado basados en partículas, filtros de suavizado y reducción de ruido, filtros por optimización grafica de factores, filtros por regresión de procesos gaussianos, filtros LOAM (Lidar Odometry and Mappig por sus siglas en inglés) y otra clase de filtros similares conocidos por una persona medianamente versada en la materia o combinaciones de estos. En una modalidad de la divulgación, el proceso de
30 filtrado (20) comprende un filtro de Kalman extendido y/o filtro de Kalman “*unscented*”.

Para el entendimiento de la presente divulgación, un filtro de Kalman es un algoritmo matemático recursivo que permite estimar el estado de un sistema dinámico a partir de una serie de mediciones afectadas por ruido o incertidumbre. El funcionamiento del filtro de
35 Kalman se divide en dos fases principales: una fase de predicción, en la cual se estima el estado futuro del sistema con base en el modelo dinámico, y una fase de corrección (o actualización), en la cual se ajusta dicha estimación utilizando las mediciones disponibles.

En una modalidad de la divulgación en la que los sensores (10) son LiDAR y RADAR, el
40 proceso de filtrado (30) en el que se ejecuta la fusión de datos de referencia (20) con el filtro

de Kalman, permite combinar la información de los sensores (10) y obtener una representación estable de los objetos detectados, así como, una estimación precisa de la posición, velocidad y orientación de cada objeto dentro del entorno que se está representando.

- 5 En una modalidad de la divulgación, en la que los sensores son LiDAR, RADAR e IMU, el proceso de filtrado (30) en el que se ejecuta la fusión de datos de referencia (20) permite que la nube de puntos capturada con el LiDAR se ubique correctamente en el espacio usando la orientación y movimientos estimados por la IMU y ajustando la posición de los objetos detectados con las mediciones del RADAR. Así, el filtro puede producir una reconstrucción
- 10 de la escena 3D coherente sin errores acumulativos.

En una modalidad de la divulgación, el proceso de filtrado (30) puede comprender las siguientes etapas

- 15 b1) definir el estado inicial del al menos un sensor (10), incluyendo su estado de calibración y ángulos de orientación inicial .

 b2) predicción con IMU: recibir en la unidad de cómputo (11) datos de aceleración, velocidad angular tomados con un sensor (10a), que es una IMU.

- 20 En este paso, se obtiene un dato del estado inicial del entorno que comprende datos de inclinación y datos de velocidad angular, con base en el movimiento de objetos en dicho entorno y luego se ajusta el error de predicción debido al ruido y deriva de la IMU.

- b3) medición con LiDAR: recibir en la unidad de cómputo (11) datos de nube de
- 25 puntos con distancias y posiciones relativas de objetos del entorno tomados con un sensor (10b) LiDAR.

En este paso, cada punto de la nube de puntos sirve como una representación de la observación directa de los objetos en el entorno.

30

En este paso, además, el filtro de Kalman corrige el estado estimado de la posición de los objetos en el entorno tomando como base los datos de referencia (20), los cuales pueden ser coordenadas cartesianas (x, y) obtenidas por el sensor LiDAR. Esto permite reducir el error acumulado por ruido y deriva del sensor (10A) que es una IMU, que genera datos de

35 aceleración y velocidad angular. En la fase de predicción, la unidad de cómputo (11) integra los datos de aceleración lineal, velocidad angular y orientación (también conocidos como datos inerciales) para estimar la inclinación y velocidad angular del objeto. Posteriormente, en la fase de corrección, el filtro de Kalman ajusta esta estimación comparándola con los datos del LiDAR, compensando sesgos, disminuyendo el ruido y proporcionando una

40 estimación más precisa y estable del estado dinámico del sistema. Otros objetos planos como

paredes se pueden usar como referencias para reducir el error inducido por el sensor LiDAR.

b4) medición con RADAR: recibir en la unidad de cómputo (11) datos de distancia, ángulo de detección, velocidad relativa tomados con el sensor (10) RADAR.

5

En este paso, a partir de los datos tomados con el sensor RADAR, se calculan datos de coordenadas y vectores de movimiento en el entorno. El filtro de Kalman integra estos datos en el proceso de filtrado (30), corrigiendo el error de predicción debido al ruido del sensor y a las incertidumbres en la dinámica de los objetos. De esta manera, por medio del proceso de
10 filtrado (30) se obtiene una estimación más precisa y estable de la posición, la dirección y la velocidad de los objetos detectados en el entorno. Estos datos se pueden integrar con los datos de referencia (20) correspondientes a la nube de puntos 3D del LiDAR para mejorar la coherencia espacial del entorno.

15 b5) fusión de datos: ejecutar un proceso de ponderación de datos de los pasos b2) a b4) de acuerdo con su precisión.

En este paso en una modalidad de divulgación, la unidad de cómputo (11) ejecuta un proceso de ponderación de las mediciones provenientes de los sensores (10) en función de su
20 precisión estadística, es decir asigna un peso relativo a cada medición proveniente de los sensores (10). En particular el filtro de Kalman en su fase de predicción calcula la incertidumbre de la predicción (covarianza del error) y en la fase de actualización calcula un factor de ponderación llamado ganancia de Kalman. De acuerdo con lo anterior, el filtro de Kalman asigna menor peso a los datos de velocidad angular e inclinación estimados por el
25 sensor IMU (10a). Por otro lado, los datos del sensor RADAR le son asignados un menor peso en la estimación de posición final, cuando aumenta la diferencia con respecto a los datos de aceleración y velocidad angular estimados en la etapa de proyección del filtro de Kalman. Para los datos de LiDAR la resolución espacial es mayor, sin embargo, el peso en la estimación final está en función de la reflectividad del material del que está hecho el objeto
30 en el entorno. De esta manera, la corrección se realiza equilibrando la confianza en los datos producidos por cada sensor (LiDAR, RADAR, IMU) según su covarianza de error. El resultado es una estimación conjunta que permite tener un resultado más robusto: el LiDAR refina la resolución tridimensional, el RADAR mejora la estimación de posición del objeto en movimiento y la IMU permite la reconstrucción tridimensional mediante la superposición
35 de múltiples datos de mediciones. La combinación ponderada de estas fuentes permite reducir la incertidumbre global en la posición y trayectoria estimada de los objetos en el entorno.

b6) reconstrucción del entorno: obtener unos datos preprocesados (21) de
40 reconstrucción del entorno.

En este paso, los datos del sensor LiDAR correspondientes a la nube de puntos se colocan en un sistema de coordenadas globales con los datos de orientación y posición que fueron obtenidos del sensor RADAR, lo que permite corregir errores de deriva y fusionar datos dinámicos de los sensores.

En una modalidad de la divulgación, el proceso de filtrado (20) puede comprender un filtro de Kalman extendido (EKF - *Extended Kalman Filter* por sus siglas en inglés). Para el entendimiento de la presente divulgación, un EKF es un filtro de Kalman diseñado para sistemas no lineales. El EKF es una extensión del filtro de Kalman que aproxima funciones no lineales mediante la linealización con derivadas Jacobianas. De este modo, cuando los datos de referencia (20) presentan relaciones no lineales, el EKF permite realizar la predicción y la corrección manteniendo la consistencia estadística de la estimación del estado global del entorno.

En una modalidad de la divulgación, el proceso de filtrado (20) puede comprender un filtro de Kalman unscented (UKF – *Unscented Kalman Filter* por sus siglas en inglés). Para el entendimiento de la presente divulgación, el UKF no requiere linealizar las funciones no lineales como el filtro de Kalman y en su lugar, genera un conjunto de puntos denominados puntos sigma que representan la distribución estadística del estado estimado del entorno. En particular, el filtro de Kalman “unscented” propaga los puntos sigma a través de las funciones no lineales del entorno y a partir de los resultados, se reconstruyen la media y la covarianza del estado del entorno. De este modo, el UKF ofrece una estimación más precisa y estable en presencia de datos no lineales.

Generación de datos de reconstrucción tridimensional del entorno.

El método comprende un paso c) en el cual se generan unos primeros datos de reconstrucción tridimensional (22) del entorno y objetos, ejecutando en la unidad de cómputo (11) un primer proceso de aprendizaje (31) a partir de los datos preprocesados (21) obtenidos en el paso b) y un paso d) en el cual se generan unos segundos datos de reconstrucción tridimensional (23) ejecutando en la unidad de cómputo (11) un segundo proceso de aprendizaje (32) a partir de los datos preprocesados (21).

El primer proceso de aprendizaje (31) y el segundo proceso de aprendizaje (32) permiten procesar los datos preprocesados (21) obtenidos en el paso b) con el fin de generar unos datos de construcción tridimensional (22, 23). El proceso de aprendizaje (31) y el proceso de aprendizaje (32) pueden comprender un proceso de clasificación, un proceso de detección y un proceso de segmentación semántica de la nube de puntos.

El proceso de clasificación corresponde a la predicción de la clase de objeto presente en el entorno y representado por la nube de puntos.

Por su parte, el proceso de detección, el cual se puede ejecutar en paralelo al proceso de clasificación, corresponde a la detección e identificación de los objetos presentes en el entorno la cual comprende datos de posición y dimensiones. El proceso de detección divide el entorno en cajas delimitadoras (conocidas como *bounding boxes*) que agrupan los puntos de la nube de puntos correspondientes a cada objeto. Cada caja delimitadora comprende datos de etiqueta de clase, coordenadas espaciales, dimensiones geométricas y puntaje de confianza que determina la certeza de la detección de los objetos en el entorno.

Por otro lado, el proceso de segmentación semántica de la nube de puntos asigna una etiqueta de clase a cada punto individual de la nube de puntos con el fin de identificar a que objeto corresponde cada punto.

Los procesos de aprendizaje (31, 32) pueden estar basados en diferentes tipos de algoritmos de procesamiento de datos 3D. En una modalidad de la divulgación los procesos de aprendizaje (31, 32), pueden estar basados en algoritmos que procesan directamente nubes de puntos como el algoritmo Pointnet, el algoritmo Pointnet++, el algoritmo DGCNN, el algoritmo KPConv, el algoritmo PointConv o el algoritmo PointTransformer. En una modalidad de la divulgación, los procesos de aprendizaje pueden estar basado en algoritmos basados en proyecciones 2D que transforman la nube de puntos en una imagen, como el algoritmo SqueezeSeg, el algoritmo RangeNet++ o el algoritmo PolarNet. En una modalidad de la divulgación, los procesos de aprendizaje (31, 32) también puede estar basado en algoritmos que utilizan cubos 3D (conocidas como voxeles) dividiendo el espacio 3D en pequeños voxeles y aplicando técnicas de convolución con redes neuronales convolucionales como VoxelNet, SECOND o SPConv. Los procesos de aprendizaje (31, 32) también pueden estar basados en otros algoritmos similares a los mencionados anteriormente conocidos por una persona medianamente versada en la materia y combinaciones de estos.

En una modalidad de la divulgación, el primer proceso de aprendizaje (31) es *SqueezeSeg*.

Para el entendimiento de la presente divulgación, el algoritmo *SqueezeSeg* es un algoritmo de procesamiento de datos 3D que realiza proyecciones 2D que transforman la nube de puntos en una imagen. El algoritmo *SqueezeSeg* etiqueta cada punto de la nube 2D creada con una clase semántica, las clases semánticas pueden ser vehículos, peatones, carreteras, objetos de la naturaleza, etc. El algoritmo *SqueezeSeg*, utilizan una arquitectura basada en redes neuronales convolucionales 2D, y no utiliza métodos como convoluciones 3D o procesamiento directo de puntos sin estructura.

En una modalidad de la divulgación, cuando el primer proceso de aprendizaje (31) es

SqueezeSeg la etapa c) puede comprender las siguientes subetapas para generar los primeros datos de reconstrucción tridimensional (22):

c1) Transformar los datos preprocesados (21) correspondientes a una nube de puntos 3D en una imagen 2D mediante una proyección esférica. En este paso la proyección 2D convierte cada punto del espacio tridimensional en un píxel de una imagen panorámica. El eje horizontal de la imagen representa el ángulo azimutal (es decir, la dirección horizontal del rayo láser del LiDAR) y el eje vertical corresponde al ángulo de elevación. A cada píxel se le asignan valores como las coordenadas tridimensionales del punto original y la intensidad del láser. Esta conversión permite aplicar redes convolucionales 2D, que son más rápidas y eficientes que trabajar con datos 3D sin procesar.

c2) Red Neural Convolucional Comprimida: una vez que se tiene la imagen 2D proyectada, el algoritmo *SqueezeSeg* utiliza una red neuronal convolucional comprimida que permite aprender patrones en la imagen proyectada y generar un mapa de características que indique la probabilidad de pertenecer a una clase semántica específica para cada píxel. Para determinar dicha probabilidad, la red neural convolucional comprimida hace una predicción punto por punto, clasificando cada rayo del sensor LiDAR.

c3) Refinamiento CRF (*Conditional Random Field* por sus siglas en inglés): Las proyecciones 2D pueden ser ruidosas o contener errores en los bordes entre objetos. Para solucionar dichos errores, el algoritmo *SqueezeSeg* puede ejecutar un modelo de refinamiento CRF. El modelo de refinamiento CRF ajusta las etiquetas finales de los puntos, teniendo en cuenta la cercanía espacial y la similitud entre puntos vecinos. Por ejemplo, si un punto aislado fue etiquetado con la clase “*persona*” pero está rodeado de puntos clasificados con la clase “*vehículo*”, el CRF puede corregir ese error para mejorar la coherencia del resultado.

c4) Generación de datos de reconstrucción tridimensional (22). En esta etapa, las etiquetas que se generaron en la imagen 2D se mapean de nuevo a los puntos originales de la nube 3D, de forma que cada punto recupera su posición tridimensional con una clase asignada. El resultado, es decir, los datos de reconstrucción tridimensional (22), pueden corresponder a una nube de puntos segmentada semánticamente. Los datos de reconstrucción tridimensional (22) pasan a la etapa (e) para determinar unos datos de métricas estimados (24).

Entrenamiento del algoritmo SqueezeSeg.

El entrenamiento del algoritmo *SqueezeSeg* se lleva a cabo de la siguiente forma: el proceso de entrenamiento comprende, en primer lugar, la conversión de las nubes de puntos obtenidas

de los sensores a un formato adecuado (p. ej. binario o tensorial), seguido por la aplicación de los algoritmos de preprocesamiento específicos de SqueezeSeg (como la proyección de la nube en una representación esférica y la normalización de atributos). A partir de estos datos, el modelo puede entrenarse desde cero (*from scratch*) inicializando sus parámetros, o
5 alternativamente emplear un modelo preentrenado. Si se usa un modelo preentrenado, se descongelan las capas superiores, se añaden capas adicionales adaptadas a la tarea específica y se ejecuta un proceso de reentrenamiento (*fine-tuning*), optimizando los pesos de la red para mejorar la segmentación de objetos en nubes de puntos.

10 En una modalidad de la divulgación, el entrenamiento del algoritmo Squeezeseg se lleva a cabo de la siguiente forma:

a) Definir el dataset: El dataset (o conjunto de datos usados para entrenar el algoritmo Squeezeseg) se puede componer de nubes de puntos en archivos binarios .bin con
15 registros contiguos de cuatro valores *float32* por punto con coordenadas (*x*, *y*, *z*, *intensidad*).

b) *Parsing* de los datos: Los datos binarios se pueden analizar y separan en partes más pequeñas de acuerdo con el formato .bin (es decir se ejecuta un proceso de *parsing*) y se puede establecer una correspondencia a un tensor de forma (*N_points*, 4) y se asocia con un
20 archivo de etiquetas .label en *uint32*, el cual comprende una etiqueta por punto.

c) Filtrado: Los puntos de la nube de puntos con distancia euclidiana menor a 1.0 m o mayor a 120 m se pueden filtrar para reducir ruido extremo y conservar, para cada punto válido de la nube de puntos, un índice de correspondencia a su posición proyectada.
25

d) Preparación de datos de entrada: La preparación de datos de entrada se basa en la proyección esférica (*spherical projection*) de SqueezeSeg, en la cual los datos de cada nube de puntos se pueden transformar en una imagen con rango (*range image*) de tamaño fijo, el cual puede ser un tamaño $H \times W = 64 \times 1024$ para un sensor tipo HDL-64E. El ángulo horizontal se calcula como $\theta = \text{atan2}(y, x)$ y se mapea a la columna $u = \text{floor}((\theta + \pi)/(2\pi) \cdot W)$; el ángulo vertical se calcula como $\phi = \arcsin(z/\sqrt{x^2+y^2+z^2})$ y se mapea a la fila $v = \text{floor}((\phi - \phi_{\min})/(\phi_{\max} - \phi_{\min}) \cdot H)$ con $\phi_{\min} = -24.9^\circ$ y $\phi_{\max} = 2.0^\circ$. Para cada píxel proyectado de la nube de puntos se pueden almacenar canales (*range*, *x*, *y*, *z*, *intensity*), en donde donde $\text{range} = \sqrt{x^2+y^2+z^2}$, junto con una mascada de validez (*validity mask*) binaria;
35 los píxeles que no se mapearon de la nube de puntos, es decir sin puntos se rellenan con cero.

e) Normalización: La normalización se puede llevar a cabo para cada canal usando estadísticas del conjunto de entrenamiento (*training set*): el rango (*range*) se puede estandarizar con media 28.0 m y desviación 17.0 m; *x*, *y* y *z* se centran en el medio de la nube
40 de puntos y se escalan con desviación 15.0 m por eje; la intensidad (*intensity*) se puede

reescalar a $[0,1]$ y se puede normalizar con media 0.5 y desviación 0.25. Las etiquetas se pueden remapear a un espacio compacto de K clases y se pueden marcar como etiqueta ignorada (*ignore label*) si el valor es de 255. La partición de datos (*data split*) se puede estratificar por secuencia (*stratified by sequence*), lo que significa que se mantiene la
5 proporción de secuencias o escenas en cada conjunto, dividiendo el dataset en 70 % para entrenamiento, 15 % para validación y 15 % para prueba (test), utilizando una semilla aleatoria 42 (*random seed 42*) para asegurar reproducibilidad.

f) Otras consideraciones para el entrenamiento del modelo: En una modalidad
10 de la divulgación en el entrenamiento del modelo SqueezeSeg se usan *fire modules* como columna vertebral del modelo (*backbone*) y una entrada de 6 canales (compuesta de cinco atributos y una máscara (mask)). Los *fire modules* son bloques de red neuronal que combina una capa de compresión (*squeeze*) con convoluciones 1×1 y una capa de expansión (*expand*) con convoluciones 1×1 y 3×3 , para reducir parámetros y mantener la capacidad de extracción
15 de características.

Sobre el *backbone* se puede agregar una convolución 2D de 1×1 (Conv2D 1×1) con 64 filtros y activación ReLU, (*Rectified Linear Unit*), seguida de una normalización por lotes (*Batch Normalization*, *BatchNorm*). A continuación, se puede incorporar una cabeza de
20 segmentación (*segmentation head*) implementada como otra Conv2D 1×1 con K filtros, que produce los logits (o valores de salida sin normalizar) por cada píxel del mapa de características. Si existe submuestreo interno (*downsampling*), se aplica sobremuestreo bilineal (*bilinear upsampling*) para realinear (*realign*) las salidas a una resolución de 64×1024 . También se puede integrar un refinamiento CRF-RNN (*Conditional Random Field*
25 *as Recurrent Neural Network*) con 5 iteraciones, lo que mejora la coherencia espacial de las etiquetas.

Para la inicialización desde cero (*initialization from scratch*), los pesos convolucionales se pueden establecer siguiendo la inicialización normal de Kaiming (*Kaiming normal*
30 *initialization*) con $\text{bias} = 0$, y las capas BatchNorm se inicializan con $\gamma = 1$ y $\beta = 0$.

Cuando se usa preentrenamiento sobre SemanticKITTI, se pueden cargar los pesos del backbone (*backbone weights*) y se congelan los módulos fire (*fire modules*) hasta fire8 durante las primeras 10 épocas (epochs). A partir de la época 11, se descongela (unfreeze) el
35 modelo por bloques, liberando dos módulos cada cinco épocas, mientras que las capas añadidas (*added layers*) y la cabeza del modelo (head) permanecen entrenables (trainable) durante todo el proceso.

El entrenamiento puede optimizar una pérdida de entropía cruzada por píxel (per-pixel cross-
40 entropy loss) con máscara de puntos ignorados (*ignore mask*) y una regularización L2 (L2

regularization) con $\lambda = 1e-4$, para evitar sobreajuste.

El aumento de datos (data augmentation) se puede aplicar en el espacio tridimensional antes de la proyección a 2D y tiene como objetivo mejorar la generalización del modelo. Este
5 proceso puede incluir rotaciones alrededor del eje Z dentro del rango de $[-5^\circ, 5^\circ]$, para simular variaciones leves en la orientación del sensor. También puede incorporar la eliminación aleatoria de rangos (*range dropout*), donde se eliminan aleatoriamente entre 1 y 3 columnas verticales (*vertical beams*) de la imagen de rango, emulando fallos o zonas no capturadas por el sensor. Además, se puede agregar ruido gaussiano con una desviación
10 estándar de $\sigma = 0.01$ m en las coordenadas espaciales y $\sigma = 0.02$ en la intensidad, para aumentar la robustez del modelo frente a mediciones imprecisas. También se pueden aplicar parches de oclusión (*occlusion patches*), que son regiones rectangulares de la imagen de rango de tamaño entre 5 y 15 píxeles, simulando zonas bloqueadas o sin retorno del láser. En general se puede respetar la máscara de puntos ignorados (*ignore mask*) para evitar
15 introducir etiquetas inválidas durante el proceso de aumento de datos.

La evaluación al final de cada época (epoch) se puede realizar sobre el conjunto de validación (*validation set*), registrando las métricas de intersección sobre unión media (*mean Intersection over Union, mIoU*), intersección sobre unión por clase (*IoU per class*), precisión
20 y exhaustividad por clase (*precision/recall per class*), y pérdida total (*total loss*). Se mantiene como el mejor punto de control (*best checkpoint*) el modelo que obtiene el mayor valor de mIoU, indicativo del mejor desempeño general en segmentación. El conjunto de prueba (*test set*) se puede evaluar al finalizar el entrenamiento, con el fin de obtener la matriz de confusión final y las métricas reportables que reflejan el rendimiento definitivo del modelo.

25

Durante el posprocesamiento con Campo Aleatorio Condicional (CRF - Conditional Random Field), se puede emplear compatibilidad tipo Potts, junto con dos núcleos bilaterales (bilateral
kernels) definidos en el espacio espacio-ángulo-rango (space-angle-range). Estos núcleos se configuran con pesos (3, 5) y parámetros $\theta_\alpha = 1.0$, $\theta_\beta = 1.0$ y $\theta_\gamma = 3.0$, que controlan
30 la influencia relativa de las características espaciales y de rango en el refinamiento de las etiquetas.

Se puede llevar a cabo también un análisis de interferencia en el cual se recibe una imagen de rango (range image) con dimensiones $64 \times 1024 \times 6$, donde cada píxel contiene información
35 proveniente del sensor LiDAR, como coordenadas espaciales e intensidad. El modelo produce un mapa de etiquetas (label map) de tamaño 64×1024 , que representa la clase semántica (semantic class) predicha para cada píxel proyectado. Además el label map se puede reasignar (re-mapped) a cada punto de la nube de puntos original utilizando la correspondencia (u, v) (pixel correspondence (u, v)) almacenada durante la proyección de
40 3D a 2D. De esta manera, se puede obtener una etiqueta semántica por punto en el sistema

de coordenadas tridimensional original.

En una modalidad de la divulgación, se puede llevar a cabo un proceso de ajuste fino, en el que se toma como modelo inicial un modelo SqueezeSeg preentrenado, y se sustituye la
5 cabeza de salida por una nueva secuencia convolucional compuesta por una Conv2D 1×1 ($in \rightarrow 128$), seguida de una activación ReLU (*Rectified Linear Unit*), una capa de Dropout con $p = 0.3$ (probabilidad de desactivación de neuronas) y una segunda Conv2D 1×1 ($128 \rightarrow K$), que genera las salidas correspondientes a las K clases (K classes). Durante las primeras 5 épocas (epochs), se congelan los módulos fire1–fire8, es decir, sus pesos no se actualizan, y
10 solo se entrenan los módulos fire9 y la nueva cabeza de salida, utilizando una tasa de aprendizaje (LR - Learning Rate) de 2×10^{-3} ($2e-3$). A partir de la época 6, se descongelan los módulos fire7–fire8, y se aplican tasas de aprendizaje diferenciadas para el backbone ($lr_backbone = 5 \times 10^{-4}$, $5e-4$) y otra mayor para la cabeza de salida ($lr_head = 1 \times 10^{-3}$, $1e-3$). Los demás hiperparámetros, el aumento de datos y el protocolo de validación permanecen
15 sin cambios.

En una modalidad de la divulgación, el segundo proceso de aprendizaje (32) es *PointNet++*.

Para el entendimiento de la presente divulgación, el algoritmo *PointNet++* es un algoritmo
20 de procesamiento de datos 3D que procesa nubes de puntos 3D que no están organizadas en una cuadrícula con una imagen, es decir, puntos 3D sin estructura. El algoritmo *PointNet++* está basado en el algoritmo *PointNet* el cual procesa cada punto de la nube de puntos 3D de forma independiente y luego usa una operación de agregación para obtener una representación global del entorno. El algoritmo *PointNet* se centra en capturar características
25 globales de la nube de puntos 3D. Por su parte el algoritmo *PointNet++* agrupa los puntos de la nube 3D en regiones locales y aplica *PointNet* a cada región para extraer características locales, para luego construir una representación global progresiva, es decir que *PointNet++* además de capturar características globales de la nube de puntos, le agrega una jerarquía agrupando los puntos en regiones locales lo que le permite detallar la geometría local y
30 también la geometría general.

En una modalidad de la divulgación, cuando el segundo proceso de aprendizaje (32) es *PointNet++* la etapa c) puede comprender las siguientes subetapas para generar los primeros datos de reconstrucción tridimensional (22):

35

d1) Recibir los datos preprocesados (21) correspondientes a una nube de puntos sin estructura, cada uno representado por coordenadas espaciales (x, y, z). En esta etapa se reciben los datos preprocesados del paso b); estos datos pueden comprender atributos adicionales como color o intensidad.

40

d2) *Sampling* o submuestreo jerárquico: En esta etapa, el algoritmo selecciona un subconjunto de puntos que serán los centros de agrupación local usando el algoritmo FPS (Farthest Point Sampling), el cual selecciona puntos distribuidos de manera uniforme en el espacio del entorno para representar toda la nube.

5

d3) *Grouping* o agrupamiento local: En esta etapa, para cada punto seleccionado en la etapa d2), se determina un conjunto de puntos vecinos que se encuentran dentro de una vecindad definida. La vecindad se define mediante un radio fijo (*ball query*) o un número predefinido de vecinos más cercanos (kNN) que determinan la geometría alrededor de cada punto. Este agrupamiento preserva la topología local del entorno y determina la posición relativa de los vecinos con respecto al punto centro.

d4) Extracción de características locales: En esta etapa cada grupo local de puntos vecinos es procesado mediante una red PointNet, la cual aplica transformaciones no lineales por medio de perceptrones multicapa y una operación de agregación simétrica, que puede ser max pooling, y genera un vector de características que representa al grupo de puntos vecinos. El proceso se repite para cada punto centro, y se produce una nube de características de dimensión reducida, pero con las características semánticas de cada punto centro.

d5) Construcción jerárquica (stacking): En este paso se construye la representación final de la nube de puntos de forma jerárquica para lo cual se repiten iterativamente las etapas d2) a d4) es decir muestreo, agrupamiento y PointNet en varios niveles. Esto genera niveles sucesivos de abstracción que permiten codificar patrones geométricos de mayor escala, desde detalles locales hasta estructuras semánticas globales. En cada nivel la nube se hace más abstracta y rica.

d6) Interpolación inversa: (upsampling): En este paso se obtiene una etiqueta para cada punto centro inicial, es decir se obtiene una nube de puntos con el mismo número de puntos que la nube original, pero con información semántica de cada punto, para lo cual se realiza una interpolación inversa que propaga las características aprendidas hacia los puntos que habían sido descartados. El proceso utiliza interpolación por distancias ponderadas y concatenación con las características de niveles anteriores conocidas como *skip connections*.

d7) Generación de datos de reconstrucción tridimensional (23). En esta etapa se lleva a cabo una clasificación por punto, es decir que a cada punto se le asigna una o más etiquetas semánticas correspondientes a categorías predefinidas. Los datos de reconstrucción tridimensional (23) pasan a la etapa (e) para determinar unos datos de métricas estimados (24).

40 *Entrenamiento del algoritmo PointNet++.*

El proceso de entrenamiento comprende, en primer lugar, la preparación y conversión de los datos de entrada a formatos adecuados de nube de puntos, tales como, .pcd o binario. Una vez normalizados los datos, el modelo PointNet++ puede entrenarse desde cero (*from scratch*)
5 inicializando aleatoriamente sus parámetros, o alternativamente empleando un modelo preentrenado. Si se usa un modelo preentrenado, se descongelan las capas superiores de la red, se añaden capas adicionales adaptadas a la tarea específica y se lleva a cabo un reentrenamiento (*fine-tuning*) utilizando los pesos previamente aprendidos como base. Este procedimiento permite acelerar la convergencia, aprovechar representaciones espaciales ya
10 adquiridas y mejorar la capacidad del modelo para clasificar y detectar objetos en nuevas nubes de puntos.

En una modalidad de la divulgación, el proceso de entrenamiento del algoritmo pointnet++ sigue los siguientes pasos:

15

a) Definir el dataset: El dataset (o conjunto de datos usados para entrenar el algoritmo pointnet++) contiene nubes de puntos en formatos .pcd y .bin. Cada punto de la nube de puntos comprende coordenadas (x, y, z, *intensidad*). Además se pueden estimar las normales (nx, ny, nz) las cuales son vectores perpendiculares a la superficie sobre la cual está
20 la nube de puntos.

b) *Parsing* de los datos: Los datos binarios se analizan y separan en partes más pequeñas (es decir se ejecuta un proceso de *parsing*). Para cada nube de puntos se crea un tensor (N_points , C) y una etiqueta por punto para segmentación semántica o por nube de
25 puntos para clasificación.

c) Filtrado: Se filtran *outliers*, los cuales corresponden a puntos de la nube de puntos los cuales son atípicos ya que se alejan significativamente del patrón general de la nube de puntos. Una vez filtrados los outliers, se hace centrado restando el centroide y
30 escalado para inscribir la nube en unidad esférica. Luego se submuestran N puntos por nube con muestreo del punto más lejano (FPS - *Farthest Point Sampling*) y se guarda la correspondencia para poder reinterpolar al final del proceso.

d) Preparación de datos de entrada: El algoritmo PointNet++ comprende
35 bloques con abstracción de conjuntos (SA - *Set Abstraction* por sus siglas en inglés) el cual es un proceso que muestrea, agrupa y resume puntos cercanos en una nube de puntos para extraer características locales jerárquicas y propagación de características de propagación de características (FP *Feature Propagation* por sus siglas en inglés) el cual es un proceso que interpola y transmite las características desde niveles submuestreados hacia puntos más
40 densos para recuperar detalles finos de la nube de puntos.

e) Normalización: En SA se pueden tomar centroides por FPS y se pueden agrupar puntos dentro de un radio fijo alrededor de cada punto muestreado por medio de consulta de esfera (BQ - Ball Query por sus siglas en inglés) y agrupamiento multi-escala (MSG – Multi Scale Grouping por sus siglas en inglés) para capturar características locales en distintas escalas. Cada grupo puede pasar por un proceso de MLPs compartidos (shared MLP - Multilayer Perceptron por sus siglas en inglés) para aprender relaciones no lineales y por agrupación máxima (*max pooling*) para resumir la información más relevante y reducir la dimensión de los datos y así obtener características por nivel del algoritmo. Además, en FP se puede hacer sobremuestreo (*upsampling*) mediante interpolación (k-NN de distancia inversa) y MLPs para propagar características hasta los N puntos originales de la nube de puntos.

Para segmentación semántica se puede agregar una segmentation denominada *head MLP*[..., K] con desactivación aleatoria de neuronas (*dropout*) y para clasificación se usa agregación global (*global pooling*) y una clasificación por cabezal MLP (*head MLP*[..., M]) la cual es la capa final totalmente conectada que convierte las características del modelo en predicciones de salida

Si la invención incluye detección (*detection*), sobre el último nivel SA se puede agregar un cabezal de detección (*detection head*) que predice probabilidad de objeto (objectness) la cual es una medida que indica cuán probable es que una región o punto contenga un objeto real en lugar de fondo o ruido. También, se pueden agregar parámetros de cajas 3D (3D boxes) las cuales son volúmenes rectangulares usados para delimitar y describir la posición, orientación y tamaño de objetos en un espacio tridimensional.

f) Otras consideraciones para el entrenamiento del modelo: El entrenamiento desde cero (*from scratch*) inicializa MLPs con Kaiming/Xavier la cual es una técnica que establece los valores iniciales de los pesos de los puntos de la nube de puntos, para mantener una varianza estable de las activaciones y así mejorar el entrenamiento de redes profundas. También se puede aplicar normalización por lotes (*BatchNorm*) en la cual se normalizan las activaciones de la nube de puntos dentro de cada lote de entrenamiento, lo que acelera el aprendizaje y mejora la estabilidad del modelo.

En modo preentrenado (*pretrained*) se pueden usar modelos preentrenados como ModelNet40/ShapeNetPart, los cuales son datasets usados en el entrenamiento de redes neuronales de nubes de puntos. ModelNet40 contiene objetos de 40 categorías como sillas, mesas o vehículos y se usa para clasificación. Por su parte, ShapeNetPart incluye modelos 3D anotados a nivel de partes en 16 categorías, como aviones, instrumentos musicales, etc y se usa para segmentación semántica. En el modelo preentrenado, el *backbone* se deja

congelado al inicio y se hace descongelamiento progresivo de SA superiores (primero BatchNorm, luego capas completas), mientras los cabezales (heads) nuevos siempre entrenan. “Descongelar” se refiere a habilitar el cálculo y actualización de gradientes en esos bloques, típicamente con tasa de aprendizaje (*learning rates*) discriminativos (menor lr en el backbone, mayor en el cabezal).

Las pérdidas (losses) dependen de la tarea específica: para segmentación semántica, se usa una entropía cruzada por punto (per-point cross-entropy), lo que significa que la pérdida se calcula individualmente para cada punto de la nube de puntos. Se incluye una etiqueta ignorada (ignore label), que excluye del cálculo los puntos no etiquetados o inválidos, y una ponderación por clase (class weighting), que ajusta la contribución de cada clase para compensar el desbalance en la cantidad de ejemplos.

Para clasificación (*classification*), se aplica la pérdida de entropía cruzada (cross-entropy loss), y se puede usar suavizado de etiquetas (label smoothing), que evita que el modelo se vuelva excesivamente confiado en sus predicciones.

Para detección (detection), la pérdida total combina la pérdida de clasificación (classification loss), el objectness (confianza de que una región contiene un objeto) y la regresión de caja (box regression), normalmente implementada con una pérdida L1 suavizada (Smooth L1 loss) para ajustar las coordenadas de las cajas delimitadoras.

Se puede usar Adam como optimizador con una tasa de aprendizaje (learning rate) inicial típico $1e-3$, un corto periodo de calentamiento (warmup), y un decaimiento cosenoidal (cosine decay) para reducir gradualmente la tasa de aprendizaje. El tamaño de lote (batch size) se puede ajustar según la memoria disponible, y se aplica recorte de gradientes (gradient clipping) moderado para mantener la estabilidad del entrenamiento.

El aumento de datos (*data augmentation*) se puede realizar para la nube de puntos 3D y puede incluir rotación leve alrededor del eje Z (*rotation around the Z-axis*), escalado aleatorio (*random scaling*), perturbación de traslación (*translation jitter*), eliminación aleatoria de puntos (*point dropout*), y variación de intensidad (*intensity jitter*), manteniendo coherente la máscara de puntos ignorados (ignore mask).

La evaluación (*evaluation*) se puede llevar a cabo sobre el conjunto de validación (*validation set*), utilizando la intersección sobre unión media (*mean Intersection over Union, mIoU*) y la intersección sobre unión por clase (*IoU per class*) para la segmentación o la precisión (*accuracy*) para la clasificación. Durante el entrenamiento, se guarda el mejor punto de control (*best checkpoint*), es decir, la versión del modelo generado con el algoritmo PointNet++ con mejor desempeño en validación, y al final se reporta la matriz de confusión

(*confusion matrix*), que muestra cómo se distribuyen los aciertos y errores entre las clases.

En la inferencia (*inference*), el modelo procesa N puntos (N points) —o todos si caben en memoria—, ejecutando los módulos de agrupamiento y extracción de características (SA -
5 *Set Abstraction*,) y propagación de características (FP - *Feature Propagation*). El modelo puede producir etiquetas por punto (*per-point labels*) para la segmentación y si se realizó submuestreo, las predicciones se pueden reinterpolar a los puntos descartados. En clasificación, el modelo puede generar probabilidades por clase (*class probabilities*). En
10 detección (*detection*), el modelo puede producir cajas decodificadas (decoded boxes) y puede aplicar una supresión no máxima 3D (3D NMS - 3D Non-Maximum Suppression) para eliminar detecciones redundantes.

Determinación de métricas estimadas (24)

15 El método comprende un paso e) en el cual se determinan unos datos de métricas estimados (24) a partir de los primeros datos de reconstrucción tridimensional (22) y de los segundos datos de reconstrucción tridimensional, mediante la ejecución en la unidad de cómputo (11) de un proceso de estimación de métricas (33).

20 El proceso de estimación de métricas (33) compara y cuantifica el desempeño de los procesos de aprendizaje (31) y (32), evaluando métricas como coherencia espacial, consistencia temporal (particularmente durante la estimación con el filtro de Kalman), confianza en la clasificación, tiempo de procesamiento e intersección sobre unión (IoU) . El proceso de estimación de métricas (33) determina los datos de métricas estimados (24), que permiten
25 valorar la calidad y eficiencia global del sistema de reconstrucción tridimensional.

Para el entendimiento de la presente divulgación, la coherencia espacial se refiere a una propiedad que determina si los puntos, estructuras o superficies reconstruidas en un modelo 3D y generadas a partir de la información del sensor (10) guardan una relación geométrica
30 consistente con el espacio físico real del que fueron obtenidos. La coherencia espacial implica que la disposición relativa de los elementos en el modelo 3D respeta las distancias, proporciones y continuidad del entorno físico original, y asegura que la representación 3D generada por el sistema es geoméricamente fiel y sin inconsistencias, respetando la estructura real del espacio.

35

Por su parte, la consistencia temporal (o coherencia temporal) se refiere a la propiedad mediante la cual la nube de puntos (o en general las estimaciones espaciales sucesivas) presentan una variación suave y coherente en el dominio del tiempo, evitando transiciones abruptas o errores de salto entre cuadros consecutivos.

40

Por otro lado, la confianza de la clasificación es un parámetro asociado a cada unidad de

información espacial reconstruida (por ejemplo, un punto 3D de la nube de puntos), que representa el grado de certidumbre con el cual dicha unidad refleja la geometría real del entorno. Esta medida puede ser continua (por ejemplo, un valor real entre 0 y 1), categórica (por ejemplo, alta, media, baja), o binaria (válido/no válido), y puede derivarse de múltiples factores, tales como la cantidad de observaciones redundantes, la calidad del sensor, la geometría de visión, la textura local, el nivel de error de reproyección o la variación entre cuadros temporales consecutivos.

Finalmente, el tiempo de procesamiento se refiere al intervalo de tiempo requerido por la unidad de cómputo (11) para ejecutar una o varias etapas del flujo de procesamiento, tales como la adquisición de datos, calibración, registro, estimación de profundidad, fusión de datos, generación de mallas 3D o texturización etc. Este parámetro evalúa el rendimiento, la escalabilidad y la viabilidad del método y sistema de reconstrucción tridimensional de entornos.

En una modalidad de la divulgación, el proceso de estimación de métricas (33) sigue los siguientes pasos:

Las métricas a ser analizadas en el proceso de estimación de métricas (33) se obtienen a partir de los resultados de los algoritmos de aprendizaje (31) SqueezeSeg y (32) PointNet++: En una modalidad de la divulgación se obtienen las siguientes métricas a ser analizadas: coherencia espacial, consistencia temporal, confianza en la clasificación y tiempo de procesamiento e intersección sobre unión (IoU). El proceso de estimación de métricas (33) sigue las siguientes subetapas:

e1) recepción y normalización de los datos de salida del algoritmo SqueezeSeg y PointNet++. Para llevar a cabo la recepción y normalización se reciben los datos de mapas de etiquetas generados por SqueezeSeg y datos de clases de objeto, cajas tridimensionales y probabilidades estimadas por PointNet++, y se convierten a un formato común de referencia en nubes de puntos tridimensionales en coordenadas globales. El formato común de referencia puede usar un marco común de referencia tipo global Este-Norte-Arriba (ENU – East-North-Up frame) en donde los ejes se definen como X=Este, Y=Norte, Z=Arriba, manteniendo unidades consistentes, por ejemplo, metros. Los datos de salida obtenidos de los algoritmos SqueezeSeg y PointNet++ se puede transformar a este marco de referencia utilizando las calibraciones extrínsecas que describen la relación espacial entre los sensores y la posición del vehículo (ego-pose), es decir, su posición y orientación en el espacio global y la posición del vehículo (ego-pose);

e2) Comparación de los datos obtenidos en la etapa e1) con datos de referencia o reconstrucciones anotadas, estableciendo la correspondencia punto a punto o caja a caja que

permite cuantificar el nivel de acierto de los algoritmos SqueezeSeg y PointNet++;

e3) Evaluación de la coherencia espacial y de la consistencia temporal de los datos de reconstrucción tridimensional (22, 23): la coherencia espacial de los datos de reconstrucción tridimensional (22, 23) se puede evaluar midiendo la correspondencia geométrica entre las superficies reconstruidas y la estructura real, así como identificando posibles discontinuidades o solapamientos que puedan la integridad del modelo tridimensional.

10 La consistencia temporal se puede evaluar analizando las estimaciones obtenidas en secuencias de cuadros consecutivos, aplicando métricas de suavidad usando el filtro de Kalman adaptativo que permite estimar trayectorias continuas y estables de los objetos detectados.

15 Para minimizar el tiempo de coherencia y optimizar la fusión entre datos de los sensores (10), se emplean métricas diferenciales que cuantifican la continuidad y estabilidad de los datos de reconstrucción tridimensional (22, 23), tales como: (i) el gradiente espacial o la norma del laplaciano, para evaluar variaciones de intensidad o profundidad; (ii) la curvatura geométrica, calculada a partir de las normales o de la curvatura media de las superficies; y (iii) la consistencia inter-etiqueta, que verifica la homogeneidad de clasificación entre puntos adyacentes o a lo largo del tiempo.

Para optimizar la estimación de posición a partir de la información de los sensores (10), se reduce el tiempo de coherencia entre los datos de referencia (20) obtenidos de los sensores (10) en particular del RADAR y LiDAR, con el fin que las mediciones correspondan al mismo estado dinámico del entorno que se está reconstruyendo. La minimización del tiempo de coherencia se logra mediante sincronización temporal precisa de pulsos, compensación de retardo y alineamiento de fase, o mediante la predicción del estado instantáneo de los objetos en el entorno usando modelos dinámicos basados en filtro de Kalman o en el filtro de Kalman extendido. La reducción del tiempo de coherencia permite incrementar la correlación temporal entre observaciones consecutivas y disminuir las discrepancias geométricas o espectrales entre los modelos SqueezeSeg y PointNet++, mejorando significativamente la estimación de características espaciales y la precisión de reconstrucción 3D en procesos de detección, segmentación o clasificación multimodal.

35 La reducción del tiempo de coherencia entre los sensores se logra sincronizando las los datos preprocesados (21) y ajustando los modelos dinámicos del filtro de Kalman para compensar adelantos o retardo de fase o retardo de muestreo, para obtener datos con estabilidad y continuidad temporal del modelo fusionado RADAR-LiDAR.

40

Las métricas usadas fueron: Desviación cuadrática de aceleración (Acceleration Smoothness Index, ASI) que evalúa cuán suave es el movimiento estimado; Velocidad media absoluta de cambio (Mean Velocity Change, MVC) que permite evaluar el efecto suavizador del Kalman frente a ruido de posición; Coeficiente de suavidad (Temporal Smoothness Coefficient, TSC),
5 cuyos valores cercanos a 1 indican movimiento estable y suavizado, y Consistencia de predicción Kalman (Innovation Consistency Index), que evalúa la coherencia entre las mediciones y las predicciones del filtro;

e4) En paralelo, se calculan métricas específicas de cada modelo: en SqueezeSeg,
10 se pueden calcular métricas como precisión, recall e intersección sobre unión (IoU) por clase y la media mIoU. En PointNet++, se pueden calcular métricas como la exactitud por clase, la precisión en las cajas tridimensionales y la puntuación F1;

e5) Evaluación de la confianza y robustez de los resultados mediante el análisis
15 estadístico de los puntajes de confianza y la sensibilidad frente a ruido o pérdida parcial de datos;

e6) Medición de la eficiencia computacional de cada algoritmo mediante el
análisis de tiempos de inferencia, consumo de memoria y recursos de cómputo, así como
20 escalabilidad frente al tamaño de la nube de puntos;

e7) Todos los datos obtenidos se integran en un conjunto de métricas
comparativas que constituyen los datos de métricas estimados (24), los cuales permiten emitir
un informe que cuantifica el desempeño relativo de los procesos de aprendizaje (31) y (32)
25 bajo condiciones homogéneas de evaluación.

Proceso de comparación de los datos métricos estimados (34).

El método comprende un paso f) de seleccionar al menos un dato del primer dato de
30 reconstrucción tridimensional (22) o del segundo dato de reconstrucción tridimensional (23)
ejecutando un proceso de comparación (34) de los datos métricos estimados (24) del primer
dato de reconstrucción tridimensional (22) y del segundo dato de reconstrucción
tridimensional (23) en la unidad de cómputo (11);

35 En esta modalidad se selecciona una métrica del primer dato de reconstrucción tridimensional
(22) y del segundo dato de reconstrucción tridimensional (23). La métrica del primer dato
de reconstrucción tridimensional (21) y del segundo dato de reconstrucción tridimensional
(23) se selecciona teniendo en cuenta el menor error en la predicción y la mayor confiabilidad.

40 La elección de una métrica adecuada depende directamente de los sensores (10) utilizados y

método y del tipo de salida que entregan los algoritmos. Para segmentación semántica, se pueden usar métricas como la precisión (precision) y el recall, expresadas como porcentajes (%), así como el IoU (Intersection over Union) por clase, cuyo valor típico oscila entre 0 y 1 (o en % entre 0–100%). El promedio de estas, conocido como mIoU (mean IoU), suele
5 usarse como métrica global de calidad.

En clasificación y detección de objetos, se utilizan métricas como la exactitud por clase (class accuracy) y la mAP (mean Average Precision) a diferentes umbrales de solapamiento (IoU=0.25 o 0.5). Estas métricas también se expresan como porcentaje, con valores
10 aceptables generalmente superiores al 70% en aplicaciones en entorno controlado y más bajos en escenarios dinámicos complejos.

Para evaluar la coherencia espacial, se emplean medidas de error geométrico como la distancia de Chamfer o la distancia de Hausdorff, típicamente expresadas en metros (m) o
15 centímetros (cm), que indican la desviación promedio entre la nube de puntos reconstruida y una referencia. En cuanto a la consistencia temporal, se analiza la estabilidad de la trayectoria con métricas en unidades de posición (m) o de ángulo (grados), evaluando la variación entre cuadros consecutivos.

20 La eficiencia computacional se mide en tiempo de inferencia (milisegundos por frame, ms/frame) y capacidad de procesamiento (frames por segundo, FPS), así como en consumo de memoria (MB o GB). Valores típicos para aplicaciones en tiempo real se ubican por debajo de 50 ms/frame (≈ 20 FPS).

25 Cuando se incorpora la fusión de sensores (IMU, LiDAR, RADAR), la elección de métricas debe considerar no solo la precisión individual de cada fuente, sino también la reducción de incertidumbre obtenida al combinarlas. En este caso, además de las métricas anteriores, se reporta la confianza de estimación (valores entre 0 y 1), y se comparan escenarios con y sin fusión para cuantificar la mejora global.

30

La selección del modelo se hace con base en la métrica que tenga menor error de predicción y mayor confiabilidad.

El resultado del proceso de comparación (34) de métricas determina cuál de los dos procesos
35 de aprendizaje (31, 32) produjo un mejor resultado.

Para cada región de la escena como se definió en las etapas c) y d), el sistema calcula un puntaje de calidad basado en las métricas y selecciona el resultado del algoritmo que proporcione la mejor solución.

40

Haciendo referencia a la FIG. 3 y a la FIG. 4, en una modalidad de la divulgación, el proceso de comparación de métricas estimadas (33) sigue los siguientes pasos:

- f1) obtener al menos un dato de la estimación (26) al comparar cada dato de los
5 datos de métricas estimados (24) del primer dato de reconstrucción tridimensional (22) con
cada dato de los datos de métricas estimados (24) del segundo dato de reconstrucción
tridimensional (23) y teniendo en cuenta un dato de referencia (27); y
- f2) generar un tercer dato de reconstrucción tridimensional (28) seleccionando
información del primer dato de reconstrucción tridimensional (22) y del segundo dato de
10 reconstrucción tridimensional (23) de acuerdo a al menos un dato de comparación;
en donde, los datos de reconstrucción tridimensional del entorno o escena se generan
a partir del tercer dato de reconstrucción tridimensional (28).

En esta modalidad se seleccionan datos de múltiples métricas estimadas seleccionadas de
15 coherencia espacial, la consistencia temporal, la confianza de la clasificación, y el tiempo de
procesamiento.

Fusión de resultados.

- 20 El método de la presente divulgación comprende una etapa final g) de generar unos datos de
reconstrucción tridimensional (25) del entorno o escena a partir de al menos un dato del
primer o segundo dato de reconstrucción tridimensional (22, 23) seleccionado ejecutando en
la unidad de cómputo un proceso de fusión de resultados (35).
- 25 El proceso de fusión de resultados puede tener enfoques diferentes de acuerdo con el dato de
métricas estimadas (24) obtenido del primer proceso de aprendizaje (31) y del segundo
proceso de aprendizaje (32).

Selección basada en criterio dominante (hard selection)

- 30 Si uno de los procesos de aprendizaje supera significativamente al otro en las métricas con
mayor prioridad que fueron seleccionadas de acuerdo al proceso de estimación de métricas
(33) se elige dicho modelo directamente.

35 *Ponderación multicriterio (soft selection o score fusión)*

Cuando uno de los procesos de aprendizaje (31, 32) hay compensaciones, por ejemplo, un
proceso de aprendizaje es más preciso pero el otro proceso es más rápido, se pueden aplicar
funciones de costo o ponderación, lo que permite obtener un índice único de desempeño y
40 elegir el proceso de aprendizaje con mejor puntaje global.

Fusión híbrida

En la fusión híbrida se puede usar un método de cascada, en el cual se usa el proceso de aprendizaje más rápido primero y se refinan ciertas zonas ambiguas con el segundo proceso de aprendizaje. En la fusión híbrida, se puede hacer además fusión por regiones en las que cada modelo se puede usar en diferentes partes del entorno 3D.

El método de la presente divulgación se puede llevar a cabo en cualquier sistema de reconstrucción tridimensional de entornos que pueda llevar a cabo las características técnicas del método que fueron explicadas anteriormente. Ahora bien, la presente divulgación se refiere también a un sistema para la reconstrucción tridimensional de entornos, en el cual se puede llevar a cabo el método de la presente divulgación.

Sistema de reconstrucción tridimensional de entornos.

15

Haciendo referencia a la FIG. 2, en una modalidad de la divulgación, el sistema de reconstrucción tridimensional de entornos comprende:

- una unidad de cómputo (11) configurada para conectarse con al menos un sensor (10), en donde la unidad de cómputo está configurada para:
 - 20 - obtener un dato de referencia (20) del sensor (10);
 - obtener datos preprocesados (21) ejecutando un proceso de filtrado (30) a partir dato de referencia (20);
 - generar unos primeros datos de reconstrucción tridimensional (22) del entorno y objetos ejecutando un primer proceso de aprendizaje (31) a partir de los datos preprocesados (21);
 - 25 - generar unos segundos datos de reconstrucción tridimensional (23) del entorno y objetos ejecutando un segundo proceso de aprendizaje (32) a partir de los datos preprocesados (21);
 - determinar unos datos de métricas estimados (24) de los primeros datos de reconstrucción tridimensional (22) y de los segundos datos de reconstrucción tridimensional (23) ejecutando un proceso de estimación de métricas (33);
 - 30 - seleccionar al menos un dato del primer dato de reconstrucción tridimensional (22) o del segundo dato de reconstrucción tridimensional (23) ejecutando un proceso de comparación de los datos métricos estimados (24) del primer dato de reconstrucción tridimensional (22) y del segundo dato de reconstrucción tridimensional (23) en la unidad de cómputo (1);
 - 35 - generar unos datos de reconstrucción tridimensional del entorno o escena a partir del dato de reconstrucción tridimensional (22, 23) seleccionado.

40 Particularmente, con el sistema de la presente divulgación se obtienen unos datos de

reconstrucción tridimensional del entorno o escena a partir del dato de reconstrucción tridimensional (22, 23) seleccionado.

En el sistema de la presente divulgación, la unidad de cómputo (11) es un dispositivo que
5 permite procesar los datos provenientes de algún elemento externo, por ejemplo, el sensor (10). La unidad de cómputo (11) analiza los datos que envía el elemento externo y lleva a cabo acciones como lo puede ser la generación de una señal de control y/o la generación de un dato, por ejemplo, el dato de referencia (20) del sensor (10), obtener datos preprocesados (21) ejecutando un proceso de filtrado (30) a partir dato de referencia (20), entre otras
10 posibles acciones que pueda llevar a cabo.

La unidad de cómputo (11) se puede seleccionar del grupo comprendido por: microcontroladores (v.gr PSOC 4BLE), micro procesadores, DSCs (Digital Signal Controller por sus siglas en inglés), FPGAs (Field Programmable Gate Array por sus siglas en inglés),
15 CPLDs (Complex Programmable Logic Device por sus siglas en inglés), ASICs (Application Specific Integrated Circuit por sus siglas en inglés), SoCs (System on Chip por sus siglas en inglés), PSoCs (Programmable System on Chip por sus siglas en inglés), computadores, servidores, tabletas, celulares, celulares inteligentes, generadores de señales y unidades de cómputo, unidades de procesamiento o módulos de procesamiento conocidas por una persona
20 medianamente versada en la materia y combinaciones de estas. La unidad de cómputo (11) puede ser o incluir una unidad de procesamiento programada para ejecutar el método de la presente invención. En un ejemplo particular, la unidad de cómputo (11) es un microcontrolador.

25 En una modalidad de la divulgación, el sistema de reconstrucción tridimensional de entornos puede comprender una primera unidad de cómputo (12) conectada a una segunda de unidad de cómputo (13). La primera unidad de cómputo (12) puede ser un dispositivo programable basado en arreglos de compuertas programables (FPGA). La primera unidad de cómputo recibe el dato de referencia (20) de al menos un sensor (10).

30

Para el entendimiento de la presente divulgación, una FPGA es una unidad de cómputo programable que puede configurarse por el usuario para realizar tareas específicas. La FPGA comprende una pluralidad de bloques lógicos configurables (CLB – Configurable Logic Blocks por sus siglas en inglés), una red de interconexiones programables y elementos de
35 entrada salida E/S configurables. En la FPGA cada uno de los bloques lógicos configurables está adaptado para implementar funciones lógicas booleanas específicas, tales como puertas AND, OR, XOR, así como elementos secuenciales como flip-flops tipo D o registros de desplazamiento. La red de interconexión programable permite establecer rutas eléctricas entre los bloques lógicos y los elementos de E/S, para crear arquitecturas digitales
40 personalizadas según los requerimientos funcionales. La configuración del dispositivo se

lleva a cabo mediante la carga de un archivo de configuración (bitstream), que define el estado de las conexiones y la funcionalidad lógica de cada bloque.

La FPGA permite la implementación directa de circuitos digitales personalizados,
5 programados por el usuario mediante lenguajes de descripción de hardware (HDL, por sus siglas en inglés), tales como VHDL o Verilog. Esta programación se compila para generar el archivo de configuración que se carga en la FPGA. La FPGA permite que un mismo dispositivo físico pueda ser reprogramado para ejecutar diferentes tareas lógicas sin modificar su hardware.

10

En el sistema de la presente invención, la FPGA permite el uso de herramientas de descripción de hardware como lo es el “*for generate*” para la generación de arquitecturas de procesamiento de los datos en paralelo que mejora el costo computacional. EL EKF se compone de diferentes operaciones entre matrices lo cual en un sistema microcontrolado o
15 microprocesado de forma convencional requiere de un número mayor de ciclos de reloj para realizar los diferentes cálculos matriciales, lo que reduce significativamente los tiempos de ejecución.

Por su parte la segunda unidad de cómputo (13) puede ser un microprocesador que ejecuta el
20 proceso de filtrado (30), el primer proceso de aprendizaje (31), el segundo proceso de aprendizaje (32) el proceso de estimación de métricas (33) y 34).

Por otro lado, la unidad de cómputo (11) puede incluir un módulo de comunicaciones o circuito de comunicaciones con uno o más puertos de conexión configurados para acceder a
25 una red de comunicaciones, por ejemplo, una red alámbrica, una red inalámbrica y combinaciones de las mismas. Ejemplos de redes de comunicaciones pueden ser redes la internet, redes VPN, redes LAN, WAN, otras redes de comunicaciones equivalentes y similares conocidas por una persona medianamente versada en la materia, y combinaciones de las mismas.

30

En los ejemplos de módulos de comunicaciones alámbricas pueden tener un puerto de conexión cableada que permite la comunicación con los dispositivos externos mediante un bus de comunicaciones, el cual se selecciona entre otros, del grupo conformado por I2C (del acrónimo de IIC Inter-Integrated Circuit), CAN (por las siglas en inglés de Controller Area
35 Network) , Ethernet, SPI (por las siglas en inglés de Serial Peripheral Interface), SCI (por las siglas en inglés de Serial Communication Interface), QSPI (por las siglas en inglés de Quad Serial Peripheral Interface), 1-Wire, D2B (por las siglas en inglés de Domestic Digital Bus), Profibus y otros conocidos por una persona medianamente versada en la materia, y combinaciones de los mismos.

40

Ejemplos de módulos de comunicaciones inalámbricas usan una tecnología de comunicación inalámbrica que se selecciona del grupo conformado por Bluetooth, WiFi, Radio Frecuencia RFID (por las siglas en inglés de Radio Frequency Identification), UWB (por las siglas en inglés de Ultra Wide B-and), GPRS, Konnex o KNX, DMX (por sus siglas en inglés, Digital
5 MultipleX), WiMax y tecnologías de comunicación inalámbricas equivalentes que sean conocidos por una persona medianamente versada en la materia y combinaciones de las anteriores.

En cualquier modalidad del sistema de la presente divulgación, la unidad de cómputo (11) se
10 puede conectar a un módulo de memoria, en donde el módulo de memoria puede estar configurado para almacenar los datos de referencia (20), los datos preprocesados (21), los primeros datos de reconstrucción tridimensional (22), los segundos datos de reconstrucción tridimensional (23), los datos de métricas estimadas (24) y los datos finales de reconstrucción tridimensional (25), entre otros posibles datos que se pueden recolectar.

15

Se entenderá en la presente invención por módulo de memoria o registro de memoria, un elemento de hardware que incluye, pero no se limita a, memorias RAM (memoria caché, SRAM, DRAM, DDR), memoria ROM (Flash, Caché, discos duros, SSD, EPROM, EEPROM, memorias ROM extraíbles (v.g. SD (miniSD, microSD, etc), MMC (MultiMedia
20 Card), Compact Flash, SMC (Smart Media Card), SDC (Secure Digital Card), MS (Memory Stick), entre otras)), CD-ROM, discos versátiles digitales (DVD por las siglas en inglés de Digital Versatile Disc) u otro almacenamiento óptico, casetes magnéticos, cintas magnéticas, almacenamiento o cualquier otro medio que pueda usarse para almacenar información y a la que se puede acceder por una unidad de cómputo, unidad de procesamiento, o módulo de
25 procesamiento. En los módulos de memoria generalmente se incorporan instrucciones, estructuras de datos, módulos de programas informáticos. Algunos ejemplos de estructura de datos son: una hoja de texto o una hoja de cálculo, o una base de datos.

Por otro lado, el sensor (10) es un dispositivo electrónico o electromecánico que puede incluir
30 un arreglo de sensores que capturan radiación electromagnética en un rango de longitudes de onda delimitado para producir una nube de puntos con base en objetos en el entorno o escena. En este caso, el sensor (10) está configurado para obtener una nube de puntos, una imagen o secuencia de imágenes 3D o unos datos de aceleración lineal, velocidad angular y orientación de objetos.

35

El sensor (10) puede ser un sensor LiDAR, un RADAR, una IMU o una cámara 3D.

El sensor (10) puede incluir un módulo de comunicaciones que envía los datos de referencia (20) hacia la unidad de cómputo (11), donde el protocolo de comunicación usado para
40 transferir las imágenes puede ser USB 3.0, Gigabit Ethernet u otros protocolos de

comunicación.

Ejemplos

Ejemplo 1

5

Haciendo referencia a la FIG. 1 y 2, se diseñó un sistema para la reconstrucción tridimensional de entornos. El sistema comprende una unidad de cómputo (10) que corresponde a la integración de un sistema con microprocesador embebido el cual es un procesador Raspberry, a un sistema programable FPGA el cual es un sistema Altera Nano 10, un sensor (10) LiDAR que es un RPLIDAR versión S1 y un sensor (10) RADAR que es un BGTR13 de marca Infineon.

Los datos del sensor (10) se procesan mediante un filtro de Kalman eliminando el ruido en las medidas asociado a las incertidumbres en las estimaciones de distancia por parte de cada uno de los sensores.

Los datos preprocesados (21) se procesan en la unidad de cómputo (11) para generar unos primeros datos de reconstrucción tridimensional (22) del entorno y objetos usando SqueezeSeg.

20

El algoritmo SqueezeSeg se entrenó de la siguiente forma:

- En primer lugar, se utilizaron conjuntos de datos de nubes de puntos provenientes de la fusión de sensores, los cuales contenían información tanto geométrica como semántica. Dichos datos fueron sometidos a procesos de normalización y muestreo para garantizar una distribución homogénea y reducir el ruido inherente a las mediciones;
- Posteriormente, se aplicaron técnicas de aumento de datos (rotaciones, traslaciones y perturbaciones aleatorias) con el fin de incrementar el tamaño del conjunto de datos y mejorar la capacidad de generalización del modelo en escenarios reales. Finalmente, se etiquetaron los puntos de las nubes de puntos obtenidas para complementar la información con datos semánticos y contextuales.

El entrenamiento se llevó a cabo mediante un esquema supervisado, donde cada punto de la nube fue asociado a su correspondiente clase semántica (vehículo, peatón, ciclista, terreno, etc.).

La red convolucional ligera de SqueezeSeg permitió procesar estas nubes de puntos de manera eficiente, utilizando kernels o núcleos optimizados para estructuras esparsas y funciones de activación adaptadas al dominio de las proyecciones bidimensionales del

espacio tridimensional de las nubes de puntos.

La optimización del modelo se realizó con el algoritmo Adam, ajustando la tasa de aprendizaje de forma progresiva hasta alcanzar la convergencia. Como resultado,

5 SqueezeSeg adquirió la capacidad de segmentar en tiempo real los objetos presentes en la nube de puntos, generando una reconstrucción tridimensional inicial (22) que no solo preserva la geometría del entorno, sino que además incorpora un nivel semántico esencial para etapas posteriores de fusión e interpretación.

10 Los datos preprocesados (21) se procesan en la unidad de cómputo (11) para generar unos segundos datos de reconstrucción tridimensional (23) del entorno y objetos usando PointNet++.

El algoritmo PointNet++ se entrenó de la siguiente forma:

15

- se obtuvieron las nubes de puntos tridimensionales provenientes de la fusión de sensores, las cuales constituyen la representación geométrica base del entorno. Una vez recolectadas, dichas nubes fueron sometidas a un proceso de etiquetado supervisado, asignando a cada región de la nube de puntos una clase correspondiente

20 (superficie, obstáculo, peatón, vehículo, etc.) con el fin de disponer de un conjunto de datos de entrenamiento confiable. Las clases mencionadas coincidieron con las realizadas para el proceso de entrenamiento de SqueezeSeg;

20

- Posteriormente, las nubes de puntos fueron preprocesadas mediante técnicas de muestreo jerárquico y normalización, lo cual permitió equilibrar la densidad de los

25 datos y reducir redundancias. Durante el entrenamiento, PointNet++ aprovechó su arquitectura de agregación jerárquica para capturar tanto características locales como globales de la nube de puntos. Esto se logró mediante la selección de regiones vecinas, la extracción progresiva de descriptores geométricos y la propagación de características hacia niveles superiores de la red.

25

30

El proceso de optimización se llevó a cabo empleando un esquema supervisado (*supervised learning*), utilizando funciones de pérdida basadas en clasificación de puntos. La red se entrenó con el optimizador Adam, ajustando los parámetros de forma iterativa hasta alcanzar un equilibrio entre precisión semántica y eficiencia computacional.

35

Como resultado, PointNet++ adquirió la capacidad de clasificar regiones de manera robusta los elementos presentes en el entorno, produciendo una reconstrucción tridimensional (23) que no solo conserva la estructura geométrica, sino que también incorpora un nivel semántico detallado que enriquece la percepción del sistema.

40

Los datos de métricas estimadas (24) del algoritmo SqueezeSeg y del algoritmo PointNet++.se generaron de la siguiente forma: a partir de los resultados de segmentación producidos por SqueezeSeg y de las regiones clasificadas por PointNet++, se calcularon indicadores de desempeño que permiten evaluar de manera objetiva la calidad de cada
5 proceso.

En ambos casos, se consideraron métricas comunes de aprendizaje supervisado, tales como:

- Exactitud global (*Accuracy*): mide la proporción de puntos correctamente etiquetados respecto al total de puntos de la nube.
10
- Precisión (*Precision*): evalúa qué tan confiables son las etiquetas asignadas, indicando la cantidad de verdaderos positivos frente a los falsos positivos.
- Exhaustividad (*Recall*): indica la capacidad del modelo para identificar correctamente todos los puntos pertenecientes a una clase, reduciendo los falsos negativos.
- 15 • F1-Score: balance entre precisión y *recall*, útil para comparar el rendimiento en clases desbalanceadas.
- IoU (*Intersection over Union*): especialmente relevante para segmentación y clasificación espacial, ya que mide la superposición entre la predicción y la etiqueta real en cada categoría.

20

Estas métricas fueron calculadas tanto para la tarea de segmentación punto a punto (SqueezeSeg), como para la tarea de clasificación de regiones (PointNet++), lo que permitió disponer de una base comparativa homogénea.

25 Finalmente, el análisis conjunto de los valores obtenidos en estas métricas comunes permitió determinar qué algoritmo ofrecía un mejor desempeño en la interpretación del entorno. En función de la clase de objeto a reconocer y del contexto de la escena, se seleccionó el modelo con mayor nivel de confiabilidad para entregar la información definitiva del objeto identificado, garantizando así un sistema de percepción robusto y consistente.

30

El proceso de comparación de métricas de SqueezeSeg y de PointNet++ obtuvo los siguientes resultados: se evidenció que ambos algoritmos presentaron un desempeño competitivo en términos de exactitud global y F1-Score, aunque con fortalezas diferenciadas según la tarea evaluada. SqueezeSeg mostró un mayor nivel de desempeño en métricas asociadas a la
35 segmentación punto a punto, alcanzando valores superiores de IoU en clases con fronteras bien definidas (como vehículos y peatones), lo que refleja su capacidad para delimitar con precisión la geometría de los objetos. Por otro lado, PointNet++ destacó en la clasificación de regiones, presentando mejores resultados en métricas de precisión y recall en categorías con mayor variabilidad espacial (como terreno o carreteras urbanas), gracias a su arquitectura
40 jerárquica capaz de capturar relaciones locales y globales dentro de la nube de puntos.

La comparación conjunta permitió identificar métricas en común (*precisión, recall, F1-score e IoU*) que sirvieron como base objetiva para decidir qué modelo debía entregar la información final en función del tipo de objeto y de la escena evaluada. De este modo, el sistema logra aprovechar las ventajas particulares de cada algoritmo, maximizando la confiabilidad de la reconstrucción tridimensional y asegurando una percepción más robusta del entorno.

El proceso de fusión (35) se ejecutó de la siguiente forma: en la etapa final del método, el proceso de fusión de resultados comprende tanto la integración de los datos de reconstrucción tridimensional obtenidos, como la combinación de diferentes arquitecturas de procesamiento.

En primer lugar, la fusión de datos permitió consolidar y unificar la información recolectada en una nube de puntos del entorno o escena, proporcionando una representación geométrica densa y coherente. Adicionalmente, la fusión de arquitecturas permitió que el sistema reconozca y clasifique objetos presentes en la escena, lo cual incrementa la confiabilidad y robustez de la reconstrucción tridimensional resultante. De este modo, la técnica no se limitó únicamente a la agregación de datos, sino que incorpora un nivel superior de interpretación semántica y estructural que optimiza la calidad del modelo generado.

Con este ejemplo se logró demostrar que la fusión de sensores, junto con la integración de arquitecturas de clasificación y segmentación, entrega un conjunto de información enriquecido que supera la simple representación geométrica de una nube de puntos. El sistema no solo percibe la estructura espacial, sino que también comprende atributos semánticos de los objetos presentes en la escena. Esto se traduce en una percepción de máquina más cercana a la visión artificial, en la que el modelo generado integra tanto la dimensión geométrica como la semántica, facilitando tareas de identificación, categorización y análisis contextual. En consecuencia, la reconstrucción tridimensional obtenida no solo representa el entorno de manera detallada, sino que también lo interpreta, ampliando el espectro de aplicaciones posibles en escenarios de navegación autónoma, robótica e interacción hombre-máquina.

Ejemplo 2:

Al sistema de reconstrucción tridimensional de entornos del Ejemplo 1 se le agregó un sensor (10) IMU modelo WT931 y una cámara digital 3D.

Una unidad de cómputo (12) que era un FPGA se conectó con los sensores (10). Los datos de los sensores (10) se procesaron por medio de un filtro de Kalman para fusionar los datos de posición de la cámara al momento de hacer un barrido sobre una escena y adquiriendo

información adicional proporcionada por la cámara 3D. La incorporación del sensor IMU y la cámara 3D mejoró significativamente la estabilidad y precisión del prototipo. La IMU aporta mediciones inerciales de aceleración y velocidad angular que permiten estimar la orientación y compensar movimientos o vibraciones, y mejoran la registración temporal y espacial de los datos de radar y LiDAR. Por su parte, la cámara 3D proporcionó información densa de profundidad y textura, enriqueciendo la reconstrucción geométrica y facilitando la fusión multimodal mediante correspondencias visuales. En conjunto, ambos sensores incrementaron la robustez del mapeo en entornos dinámicos y la coherencia estructural de las superficies reconstruidas;

10

Se debe entender que la presente divulgación no se haya limitada a las modalidades descritas e ilustradas, pues como será evidente para una persona versada en el arte, existen variaciones y modificaciones posibles que no se apartan del espíritu de la invención, definido por las reivindicaciones.