ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

IDENTIFICACIÓN DE REGIONES LIBRES DE OBSTÁCULOS EN NUBES DE PUNTOS UTILIZANDO UNA RED NEURONAL CONVOLUCIONAL

TESIS PREVIA A LA OBTENCIÓN DEL GRADO DE MÁSTER EN CIENCIAS DE LA COMPUTACIÓN
MENCIÓN SISTEMAS INTELIGENTES

PAÚL ISAÍAS TINIZARAY ROMERO
paul.tinizaray@epn.edu.ec

DIRECTOR: JOSÉ FRANCISCO LUCIO NARANJO
jose.lucio@epn.edu.ec

CODIRECTOR: WILBERT GEOVANNY AGUILAR CASTILLO
wgaguilar@espe.edu.ec

Quito, Diciembre de 2020

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Paúl Isaías Tinizaray Romero, bajo mi supervisión.

Firmado electrónicamente por:
JOSE FRANCISCO LUCIO NARANJO

PhD. José Francisco Lucio Naranjo

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Paúl Isaías Tinizaray Romero, bajo mi supervisión.

Firmado electrónicamente por:
WILBERT GEOVANNY
AGUILAR CASTILLO

PhD. Wilbert Geovanny Aguilar Castillo

DECLARACIÓN

Yo, Paúl Isaías Tinizaray Romero, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Ing. Paúl Isaías Tinizaray Romero

ÍNDICE DE CONTENIDO

Lista de Tablas

Lista de Figuras

RESUMEN

Esta investigación presenta la configuración, entrenamiento y evaluación de una red neuronal convolucional dedicada a la tarea de identificar regiones libres de obstáculos en nubes de puntos. Todo esto como parte del desarrollo de un dispositivo de navegación asistida para personas con discapacidad visual. Con la finalidad de que el tiempo de procesamiento de la red sea adecuado para este dispositivo, la identificación de regiones libres de obstáculos se abordada como un problema de clasificación de parches, generados a través de una ventana deslizante. Pero a diferencia de otras soluciones que utilizan este enfoque, se plantea etiquetar todos los pixels del parche y no superponer la ventana con la que se extraen. La red se evalúa en los repositorios públicos Labelmefacade y NUY-v2 así como en un repositorio privado formado por escenas recolectadas en la Facultad de Sistemas de la Escuela Politécnica Nacional. Las pruebas realizadas sobre los repositorios públicos muestran disminución en el tiempo de procesamiento respecto del estado del arte. Las pruebas realizadas sobre el repositorio privado muestran que la red es capaz reducir la influencia del ruido generado durante la adquisición de los datos al mismo tiempo que mantiene una estructura compacta, garantizando así un procesamiento eficiente de la información.

Palabras clave: red neuronal convolucional, detección de piso, nube de puntos, navegación asistida para no videntes, mapa de profundidad.

ABSTRACT

This research presents the configuration, train and evaluation of a convolutional neural network intended to detect floor regions in point clouds. This is part of the development of an assisted navigation device for visually impaired people. In order that network's processing time is adequate for this device, the identification of floor regions was tackled as a classification problem of patches generated by a sliding window. But unlike other solutions that use this approach, this work proposes labeling all the pixels of the patch and not overlapping the window with which they are extracted. The net is evaluated in public datasets Labelmefacade and NYU-v2 as well as in a private dataset of scenes retrieved from Faculty of Engineering Systems of Escuela Politécnica Nacional building. Tests conducted in public datasets show a reduction in processing time with respect of state-of-the-art models. Tests conducted in the private dataset show that the net reduces the effect of the noise generated during information retrieving and keeps its compactness, thus guaranteeing a efficient information processing.

Keywords: convolutional neural network, floor detection, point cloud, navigation assistance for visually impaired people, depth map.

1 INTRODUCCIÓN

Segmentación semántica es un problema de visión por computador cuyo objetivo es asignar etiquetas semánticas a todos los pixels de una imagen [1], simulando la capacidad que tienen los seres humanos para distinguir regiones y objetos en una imagen. La identificación de regiones libres de obstáculos es un caso particular de este problema en el que se busca etiquetar los pixels como camino u obstáculo. El diseño de modelos computacionales que pueden cumplir esta tarea es un área de investigación relevante por las aplicaciones que tiene: conducción asistida y autónoma de vehículos, adelantos en robótica móvil, desarrollo de dispositivos de asistencia para personas, etc. Esta es un aŕea cuyo desarrollo apenas empieza porque, a pesar de los avances que existen en este campo, los modelos actuales se encuentran muy lejos de parecerse en exactitud y velocidad a los sistemas biológicos de los cuales son inspirados. Tómese como ejemplo el fracaso que tuvo el fabricante de vehículos Tesla Motor cuando uno de sus Modelo X que operaba en AutoPilot no fue capaz de detectar la presencia de 2 motociclistas, 2 motocicletas y una camioneta parqueados en una autopista, hecho que causó la muerte de uno de los motociclistas [2].

Entre las posibles aplicaciones que pueden tener estos modelos, es de especial interés el desarrollo de dispositivos ETA (del ingles *Electronic Travel Aid*). En este sentido, la Constitución de la República del Ecuador en el Artículo 47 reconoce a las personas con discapacidad el derecho a la asistencia permanente que incluye las correspondientes ayudas técnicas [3].

ETA es una forma de tecnología de asistencia que tiene el propósito, entre otras cosas, de mejorar la movilidad de peatones con discapacidad visual. Una de las tareas que deben cumplir los dispositivos ETA es presisamente identificar regiones libres de obstáculos en el entorno del usuario, para lo cual existen diferentes alternativas. Una de ellas es procesar la información de color del entorno como [4, 5, 6, 7] que utilizan redes neuronales convolucionales (CNN - del inglés *convolutional neural network*). Otra alternativa es procesar información espacial del entorno como [8, 9, 10] que utilizan la ecuación del plano, [11, 12] que utilizan *random forest* o [13, 14] que utilizan CNNs.

A la par de identificar regiones libres de obstáculos, los dispositivos ETA requieren modelos cuyo tiempo de respuesta sea menor al del usuario y que puedan ser implementados en dispositivos de bajo consumo de energía, lo que restringe el uso de algunas soluciones como [6, 15, 16].

1.1 Objetivo

Con los antecedentes expuestos, el objetivo de esta investigación es producir un modelo computacional capaz de identificar regiones libres de obstáculos en el entorno cuyo tiempo de procesamiento mejore el del

estado del arte y le permita formar parte de dispositivos ETA.

1.2 Hipótesis

Al abordar el problema de identificación de regiones libres de obstáculos en una nube de puntos como una clasificación de parches con una red neuronal convolucional donde se etiquetan todos los elementos del mismo parche y no se superpone la ventana con la que se extraen, el tiempo de procesamiento de esta red mejorará con relación al estado del arte y le permitirá formar parte de dispositivos de navegación asistida para personas con discapacidad visual.

La idea de resolver la identificación de regiones libres de obstáculos como una clasificación de parches utilizando una CNN se presenta en [6, 7]. Se escoje este abordaje porque en CNNs una entrada pequeña implica una red pequeña [6]. Una limitación de este tratamiento es que solamente se etiqueta el pixel central de cada parche, provocando mayor tiempo de procesamiento porque la ventana utilizada para extrerlos se superpone y la misma información se procesa varias veces [17]. Para superar esta limitación se plantea etiquetar todos los pixels del parche y no superponer la ventana con la que se extraen tal como [5]. La revisión sistemática de la literatura indica que este es el primer trabajo en presentar esta combinación.

1.3 Alcance

La red se evalúa sobre los repositorios Labelmefacade, NYU-v2 y un conjunto de escenas recuperadas del edificio de la Facultad de Ingeniería de Sistemas de la Escuela Politécnica Nacional (FIS-EPN). Las pruebas conducidas sobre el repositorio Labelmefacade muestran que la red propuesta mantiene la exactitud balanceada del modelo original pero con menor tiempo de procesamiento. Además, los experimentos realizados sobre el repositorio NYU-v2 muestran una disminución del tiempo de procesamiento respecto al estado del arte. Finalmente, los experimentos realizados sobre el repositorio FIS-EPN muestran que la red es capaz de reducir la influencia del ruido generado durante la adquisición de información a la par de mantener una estructura compacta, garantizando así un procesamiento eficiente de la información.

1.4 Estructura del documento

En el siguiente Capítulo se presenta el estado del arte referente a identificación de regiones libres de obstáculos. En el Capítulo 3 se presenta la estructura de la CNN propuesta. En el Capítulo 4 se presentan los experimentos realizados sobre los repositorios Labelmefacade, NYU-v2 y FIS-EPN. En el Capítulo 5 se presentan los resultados de la integración del modelo en el dispositivo ETA FIS-EPN. Finalmente, en el Capítulo 6 se presentan las conclusiones de esta investigación.

2 REVISIÓN SISTEMÁTICA DE LA LITERATURA

El objetivo de esta revisión sistemática de la literatura (SLR) es determinar el estado del arte relacionado a identificación de regiones libres de obstáculos. Para el desarrollo de este capítulo se utiliza el método propuesto por Kitchenham y Charters [18].

2.1 Pregunta de investigación

A partir del objetivo de la SLR, se plantea la siguiente pregunta de investigación: ¿Cuál es el estado del arte relacionado a identificación de regiones libres de obstáculos?

2.2 Métodos de revisión

2.2.1 Fuente de datos y estrategia de búsqueda

Fuente de datos

Como fuentes de información fueron seleccionadas Google Académico y Scopus porque incluyen publicaciones de varias editoriales y de esta manera la búsqueda de información se realiza sobre un conjunto de estudios diverso.

Cadena de búsqueda

En la cadena de búsqueda se utilizaron los siguiente términos para encontrar estudios relacionados a identificación de regiones libres de obstáculos: *"floor segmentation"*, *"floor detection"* y *"traversable area"*. Además, se incluyeron los términos *"point cloud"*, *point-cloud*, *rgbd*, *rgb-d* y *"depth map"* con el objetivo de recuperar aquellos estudios que utilizan información espacial. Combinando estos términos se obtuvo la siguiente cadena de búsqueda: ("floor segmentation" OR "floor detection" OR "traversable area") AND ("point cloud" OR point-cloud OR rgbd OR rgb-d OR "depth map").

2.2.2 Selección de estudios

Para la inclusión de estudios en la SLR, se definieron los siguientes criterios:

- Publicado a partir de 2015.

- Incluido en ciencias de la computación, ingeniería o matemática.
- Escrito en inglés.

Para la exclusión de estudios en la SLR, se define el siguiente criterio:

- Si en el estudio no se realiza la identificación de regiones libres de obstáculos.

2.2.3 Evaluación de calidad de estudios

Teniendo en cuenta el objetivo de esta SLR, para evaluar la calidad de los estudios se define las siguientes preguntas:

- ¿Está claramente definido el enfoque propuesto?, si se trata de un enfoque original.
- ¿Está el enfoque acompañado de referencias?, si se utiliza un enfoque presentado previamente.

2.2.4 Extracción de datos

Para recolectar la información de los estudios primarios, los siguientes campos fueron seleccionados:

- Fuente de donde se obtuvo el estudio
- Título
- Año de publicación
- Resumen del enfoque propuesto para resolver el problema de identificación de regiones libres de obstáculos
- Resultados
- Referencia

2.2.5 Síntesis de datos

A fin de sintetizar la información obtenida, los estudios fueron agrupados de acuerdo a la naturaleza del enfoque que utilizaron.

2.3 Estudios incluidos y excluidos

La Tabla 2.1 resume los resultados de la búsqueda. La primera columna indica la fuente a la que pertenecen los resultados. La segunda columna indica el número de estudios recuperados desde cada fuente. La tercera columna muestra el número de estudios repetidos. La cuarta columna indica estudios que coincidieron con el criterio de exclusión y la última columna muestra los estudios aceptados. La Tabla 2.2 presenta más información sobre los estudios aceptados. La primera columna muestra el número de estudio. La segunda columna muestra la Fuente a la que pertenece el estudio: G para Google Scholar y S para Scopus. La

Tabla 2.1. Resultados en números de la búsqueda de estudios.

Fuente	Estudios			
	Total	Repetidos	Rechazados	Aceptados
Google Scholar	110	6	57	47
Scopus	60	24	33	3
Total	170	30	90	50

tercera muestra el título del estudio. La cuarta columna muestra el año de publicación del estudio y la última columna muestra la referencia del estudio.

Tabla 2.2. Estudios aceptados

N.	**C.**	**Título**	**Año**	**Ref.**
1	G	A Binary Descriptor Invariant to Rotation and Robust to Noise (BIRRN) for Floor Recognition	2019	[19]
2	G	A Novel Method of Traversable Area Extraction Fused With LiDAR Odometry in Off-road Environment	2019	[20]
3	G	Campus Guide: A Lidar-based Mobile Robot	2019	[8]
4	G	Ego-Semantic Labeling of Scene from Depth Image for Visually Impaired and Blind People	2019	[21]
5	G	Floor Detection in Outdoor Environments Through the Use of Deep Learning Approach	2019	[15]
6	G	Real-time Detection of Travelable Path Based on Image Point Cloud	2019	[22]
7	G	Road Curb Detection Using A Novel Tensor Voting Algorithm	2019	[23]
8	G	Segmentation in Corridor Environments: Combining Floor and Ceiling Detection	2019	[24]
9	G	Wearable Travel Aid for Environment Perception and Navigation of Visually Impaired People	2019	[25]
10	S	Stereo vision based sensory substitution for the visually impaired	2019	[26]
11	S	Robustifying semantic cognition of traversability across wearable RGB-depth cameras	2019	[27]
12	G	Design and Implementation of Ground Plane Detection using Predefined Datasets and Space Geometric Analysis	2018	[9]
13	G	Long-range traversability awareness and low-lying obstacle negotiation with RealSense for the visually impaired	2018	[28]
14	G	Natural gaze data driven wheelchair	2018	[10]
15	G	Perception framework of water hazards beyond traversability for real-world navigation assistance systems	2018	[29]
16	G	Phone based Video Processing Aimed at Outdoor Guidance for the Visually Impaired	2018	[30]

17	G	Semantic perception of curbs beyond traversability for real-world navigation assistance systems	2018	[31]
18	G	Virtual white cane featuring time-of-flight 3D imaging supporting visually impaired users	2018	[32]
19	G	Vision-based fallen person detection for the elderly	2018	[33]
20	G	A Deep-Learning-Based Floor Detection System for the Visually Impaired	2017	[5]
21	G	A Global Path Planning Strategy for a UGV from Aerial Elevation Maps for Disaster Response	2017	[34]
22	G	A real-time smartphone-based floor detection system for the visually impaired	2017	[35]
23	G	Building Scale VR: Automatically Creating Indoor 3D Maps and its Application to Simulation of Disaster Situations	2017	[36]
24	G	Detecting traversable area and water hazards for the visually impaired with a pRGB-D sensor	2017	[37]
25	G	Fast Ground Detection for Range Cameras on Road Surfaces Using a Three-Step Segmentation	2017	[38]
26	G	From one to many: Unsupervised traversable area segmentation in off-road environment	2017	[39]
27	G	Human navigation assistance with a RGB-D sensor	2017	[40]
28	G	Lidar-based urban road detection by histograms of normalized inverse depths and line scanning	2017	[41]
29	G	Lidar-histogram for fast road and obstacle detection	2017	[42]
30	G	Modeling traffic scenes for intelligent vehicles using CNN-based detection and orientation estimation	2017	[43]
31	G	Navigational path detection for the visually impaired using fully convolutional networks	2017	[44]
32	G	Obstacle Identification for Vision Assisted Control Architecture of a Hybrid Mechanism Mobile Robot	2017	[45]
33	G	Real-time lane detection and forward collision warning system based on stereo vision	2017	[46]
34	G	Ambient awareness for agricultural robotic vehicles	2016	[47]
35	G	An autonomous stereovision-based navigation system (ASNS) for mobile robots	2016	[48]
36	G	Expanding the detection of traversable area with RealSense for the visually impaired	2016	[49]
37	G	Indoor positioning system based on distributed camera sensor networks for mobile robot	2016	[50]

38	G	Indoor/outdoor navigation system based on possibilistic traversable area segmentation for visually impaired people	2016	[51]
39	G	Real-time obstacle detection system in indoor environment for the visually impaired using microsoft kinect sensor	2016	[52]
40	G	Traversable detection in semi-structured environment by using 2D sequential laser data for a small mobile robot	2016	[53]
41	G	A new approach of point cloud processing and scene segmentation for guiding the visually impaired	2015	[16]
42	G	A novel floor segmentation algorithm for mobile robot navigation	2015	[54]
43	G	A real-time relative probabilistic mapping algorithm for high-speed off-road autonomous driving	2015	[55]
44	G	Dominant plane detection using a RGB-D camera for autonomous navigation	2015	[56]
45	G	Free space detection: a corner stone of automated driving	2015	[57]
46	G	Indoor scene 3D modeling with single image	2015	[58]
47	G	Point clouds segmentation as base for as-built BIM creation	2015	[59]
48	G	Range image processing for real time hospital-room monitoring	2015	[60]
49	G	Segmentation of point cloud from a 3D LIDAR using range difference between neighbouring beams	2015	[61]
50	S	Road detection in image by fusion laser points based on fuzzy SVM for a small ground mobile robot	2015	[62]

2.4 Resultados

Los estudios aceptados se agrupan de acuerdo al enfoque propuesto tal como se muestra en la Tabla 2.3. La primera columna muestra el tipo de información que fue procesada. La segunda columna muestra el identificador de cada categoría. La tercera columna muestra las categorías seleccionadas para agrupar los estudios. La cuarta columna muestra la cantidad estudios agrupados en cada categoría y la última muestra las referencias de los estudios de cada categoría.

2.4.1 Clasificación de pixels

Estos estudios buscan etiquetar los pixels de una imagen RGB como libre u ocupado.

Los estudios pertenecientes a la categoría 1 emplean redes neuronales convolucionales (CNN) para clasificar pixels. [27, 29, 31] utilizan el modelo ERF-PSPNet. [15] emplea el modelo Mask R-CNN. [39] utiliza el modelo Deeplab. [44] emplea una modificación del modelo FCN-8s. Finalmente, [5] utiliza una CNN compuesta por 7 capas convolucionales y 2 capas completamente conectadas.

Los estudios pertenecientes al grupo 2 extraen características de los pixels de una imagen como color [30, 51, 54] o color, textura y valores de borde de superpixels [45]. En base a las características recuperadas se crean reglas basadas en umbrales para realizar la clasificación de pixels.

Tabla 2.3. Enfoques agrupados según tipo de información que procesan.

Información	Id.	Categoría	Est.	Ref.
píxeles	1	red neuronal	7	[15, 27, 29, 31, 5, 39, 44]
	2	discriminación basada en umbrales	4	[30, 45, 51, 54]
	3	identificación de cuboide	2	[24, 35]
	4	máquina de soporte vectorial	1	[62]
	5	regresión logística	1	[19]
		Total clasificación de píxeles	15	
puntos	6	discriminación de la coordenada y	12	[20, 21, 25, 28, 33, 37, 41, 43, 50, 53, 55, 59]
	7	ajuste de ecuación de plano	10	[8, 9, 10, 36, 38, 40, 48, 49, 52, 57]
	8	ajuste de ecuación de recta en mapa de disparidad-v	4	[26, 32, 42, 46]
	9	discriminación de vectores normales	4	[22, 34, 16, 60]
	10	extensión de región	3	[56, 58, 61]
	11	distribución de Gauss multivariable / clasificador de distancia Mahalanobis	1	[47]
	12	tensor voting descriptor	1	[23]
		Total clasificación de puntos	35	
		Total	50	

Los estudios pertenecientes al conjunto 3 emplean el detector de bordes Canny y el detector de líneas Hough para recuperar las líneas más relevantes en una imagen del entorno. A continuación, se asume que el entorno tiene forma cuboide y se seleccionan las líneas que representan los bordes de esta figura. Finalmente, se considera la base del cuboide como región libre de obstáculos.

Alternativas a los enfoques antes descritos se presentan en las categorías 4 y 5. En [19], se ajusta un modelo de regresión logística utilizando descriptores de color y textura. Por su parte, [62] ajusta un modelo máquina de vectores de soporte difusa empleando descriptores de color, textura y pertenencia.

2.4.2 Clasificación de puntos

Estos estudios buscan clasificar la información espacial del entorno como obstáculo o camino (región libre de obstáculos). Esta información puede estar representada en distintos formatos. Una parte de los estudios emplean el formato nube de puntos. Una nube de puntos es una colección de puntos (x, y, z) que representan la superficie externa de una escena [63]. Para el desarrollo de esta sección fue utilizado el sistema coordenado y-up empleado en el software Unity [64]. La otra parte de los estudios emplea el formato mapa de disparidad. Un mapa de disparidad almacena un valor de disparidad (δ) para cada píxel (X, Y) de una imagen. Disparidad es el valor inverso de profundidad [65].

En los estudios pertenecientes al grupo 6, [20, 21, 28, 37, 43, 50, 55] asumen que los puntos que pertenecen a regiones libres de obstáculos se encuentran contenidos en un plano paralelo al plano xz, representado por la Ecuación 2.1 donde D es la distancia perpendicular del sensor al plano objetivo. Bajo tales condiciones, la regla que permite identificar los puntos se muestra en la Ecuación 2.2 donde δ es un

umbral de tolerancia utilizado para compensar errores de medición del sensor.

$$y + D = 0 \tag{2.1}$$

$$f(y) = \begin{cases} 1, & \text{si } y \in [d - \delta, d + \delta] \\ 0, & \text{de otro modo} \end{cases} \tag{2.2}$$

En [25] se reemplaza el umbral fijo de la Ecuación 2.2 por uno adaptativo basado en el algoritmo de OTSU. Finalmente, [53, 59] crean un histograma a partir de los valores y de todos los elementos de la nube de puntos y seleccionan el valor con mayor ocurrencias como la distancia D.

En los estudios pertenecientes al conjunto 7, se asume que los elementos de una nube de puntos que pertenecen a regiones libres de obstáculos están contenidos en un plano representado por la Ecuación 2.3, donde A, B, C y D son coeficientes que se estiman utilizando elementos de la nube de puntos.

$$Ax + By + Cz + D = 0 \tag{2.3}$$

Nótese que en una nube de puntos, además del plano objetivo, pueden existir otros que pertenezcan a obstáculos. Para descartar estos planos se añaden condiciones como discriminación en base al número de elementos que contienen [36, 38, 48, 49, 52], discriminación en base a distancia sensor-plano [9, 10], discriminación en base a valores del vector unitario normal al plano [40] o seleccionar los puntos cercanos al sensor para la estimación asumiendo que el sensor se encuentra siempre en una región libre de obstáculos [8]. [57] menciona que utiliza funciones polinomiales para detectar la superficie del suelo, pero no menciona la condición que utiliza para descartar otras superficies.

En los estudios pertenecientes a la categoría 8, a partir de un mapa de disparidad se calcula un mapa de disparidad-v [66]. En el mapa de disparidad-v, los planos que existen en la nube de puntos se encuentran representados por rectas. Por lo tanto, se asume que los puntos pertenecientes a regiones libres de obstáculos están contenidos en un plano representado en el mapa de disparidad por una recta, representada por la Ecuación 2.4, donde A, B y C son coeficientes que se estiman utilizando los elementos del mapa de disparidad-v.

$$Ax + By + C = 0 \tag{2.4}$$

En el mapa de disparidad-v es posible encontrar rectas que no representen al plano objetivo. Para encontrar la recta de interés, [26, 32] condicionan el valor de pendiente de las rectas encontradas. [46, 42] selecciona la línea con mayor número de elementos.

En los estudios pertenecientes al grupo 9, para cada punto de la nube de puntos se obtiene los valores del vector normal unitario (n_x, n_y, n_z) a la superficie a la que pertenece el punto. Se asume que los puntos que pertenecen a regiones libres de obstáculos se encuentran contenidos en un plano paralelo al plano xz. Bajo tal condición, los puntos objetivo exhiben valores n_x y n_z cercanos a 0 y valor n_y cercano a 1. La regla para encontrar estos puntos se muestra en la Ecuación 2.5, donde δ es un umbral de tolerancia

utilizado para compensar errores de medición del sensor.

$$f(n_y) = \begin{cases} 1, & \text{si } n_y \geq \delta \\ 0, & \text{de otro modo} \end{cases} \tag{2.5}$$

Nótese que esta regla recupera no solamente los puntos pertenecientes al plano objetivo sino además puntos pertenecientes otros planos pertenecientes al plano xz. Para descartar tales planos se emplean condiciones adicionales como discriminación de los puntos recuperados por su distancia media al sensor [22, 34, 16, 60].

Los estudios pertenecientes al conjunto 10, emplean la técnica *region growing*. Esta técnica empieza con una o más regiones semilla a las que repetidamente se añaden elementos vecinos en base a criterios de similitud [67]. En [58] se utilizan color, distancia al plano, dirección del vector unitario normal y curvatura como criterios de similitud. En [61], el criterio de similitud es el grado de distorsión (*unevenness*) que presentan los datos capturados por un escáner láser.

Alternativas a los enfoques antes descritos se presentan en las categorías 11 y 12. En [47], la nube de puntos es dividida en una cuadrícula regular de parches proyectados en el plano horizontal. De cada parche se extrae un vector de características formado por el ángulo entre el plano formado por los puntos del parche y el plano horizontal, la desviación cuadrática media de los puntos desde el plano del parche a lo largo de su normal, la varianza en la altura de los puntos con respecto al plano horizontal y la media de la altura de los puntos del parche con respecto al plano horizontal. Con los vectores de características se ajusta un modelo gaussiano multivariable y la regla de clasificación está dada por un clasificador de distancia de Mahalanobis. Por su parte, [23] recolecta características de los elementos de la nube de puntos utilizando *tensor voting* y la regla de clasificación está dada por un umbral sobre la característica *stick*.

2.4.3 Resultados cuantitativos

De los 51 estudios aceptados, 23 tienen resultados cuantitativos de sus experimentos, 15 pertenecen al grupo de clasificación de pixels [19, 15, 24, 27, 29, 30, 31, 5, 35, 39, 44, 45, 51, 54, 62] y 8 al grupo de clasificación de puntos [21, 26, 9, 38, 40, 41, 42, 47]. Del grupo de 23, 9 utilizan conjuntos de datos públicos. Los resultados de estos estudios se muestran en la Tabla 2.4. La primera columna muestra la referencia del estudio, la segunda columna indica el conjunto de datos utilizados durante la experimentación, la tercera columna indica el tipo de información que se procesa en el estudio, las columnas 4-8 muestran los valores reportados de exactitud, precisión, exhaustividad, intersección sobre unión (IoU) y *F1 score* respectivamente.

2.4.4 Velocidad de procesamiento

De los 51 estudios aceptados, 20 reportan la velocidad de procesamiento de sus modelos y características del hardware utilizado. Esta información se muestra en la Tabla 2.5. La primera columna muestra la referencia del estudio, la segunda columna indica la velocidad de procesamiento del modelo propuesto en cada estudio en cuadros por segundo (fps), la tercera columna muestra el tiempo que demora el modelo en

Tabla 2.4. Estudios con resultados cuantitativos en conjuntos de datos públicos.

Est.	Datos	Info.	Resultados				
			Exa.	Pre.	Exh.	IoU	F1 score
[27]	RGB-D-SS	Pixels	0.964				
[29]	Terrain awarness	Pixels	0.931			0.821	
[31]	Real-world curbs dataset	Pixels	0.975				
[44]	SUN RGB-D / TUM	Pixels	0.918			0.790	
[51]	TUM	Pixels		0.970			
[21]	NYU Depth Dataset V2	Puntos	0.918				
[40]	TUM	Puntos		0.991	0.962		0.976
[41]	KITTI uu_road	Puntos	0.815	0.870	0.935		
[42]	KITTI uu_road	Puntos		0.907	0.827		0.865

procesar un cuadro en milisegundos (ms) y la cuarta columna presenta las características del equipo usado para evaluar cada modelo.

2.5 Conclusiones

Para identificar regiones libres de obstáculos es posible utilizar la información de color del entorno así como su información espacial, tal como muestra la Tabla 2.3. Cuando se procesan imágenes, el trabajo de identificación consiste en extraer y clasificar características visuales. Cuando se procesa información espacial, es posible partir de la hipótesis de que las regiones libres de obstáculos están contenidas en un plano. La Tabla 2.4 muestra los trabajos que reportan los resultados de su evaluación sobre conjuntos de datos públicos. Dado que ningún modelo es capaz de clasificar correctamente todos los elementos de su respectivo conjunto de datos, el desarrollo de modelos que ofrezcan mejor rendimiento es una brecha de investigación abierta. Por su parte, la Tabla 2.5 muestra la velocidad de procesamiento de determinados modelos. En el desarrollo de dispositivos ETA, donde se intenta replicar el sistema de visión del ser humano, estos modelos deberían alcanzar una velocidad de procesamiento igual a 75 fps [68]. Dado que las velocidades reportadas son inferiores, el desarrollo de modelos con mayor velocidad de procesamiento es también una brecha de investigación abierta.

Tabla 2.5. Estudios que reportan tiempos de procesamiento.

Estudio	Velocidad (fps)	Tiempo (ms)	Equipo
[24]	32.488	30.780	CPU i5 3.4 GHz
[31]	22.000	45.000	Jetson TX1
[27]	22.000	45.000	Jetson TX1
[37]	20.833	48.000	CPU i7 4.0 GHz
[51]	20.000	50.000	CPU i7 2.0 GHz, 4 Gb RAM
[22]	20.000	50.000	CPU i7 3.2 GHz
[25]	20.000	51.590	CPU 2.0 GHz, 4 GB RAM
[20]	14.285	70.000	CPU i7 2.8 GHz
[22]	14.084	71.000	1.55 GHz Octa-core ARM processor and 3GB of DDR3 RAM
[29]	14.000	71.000	Jetson TX1
[42]	10.000	100.000	2.5 GHz, 4 Gb RAM
[26]	7.000	142.000	CPU i7 2.80 GHz, 8 Gb RAM, GPU nVidia GeForce 960
[32]	7.000	142.850	Octa Core 4 x Cortex-A72 1.8 GHz + 4 x Core-A53 1.4 GHz
[45]	4.545	220.000	i7 2.2 GHz
[9]	3.802	263.000	CPU i7 4.0, 24 Gb RAM
[49]	3.570	280.000	CPU 1.9 GHz
[41]	2.000	500.000	i5 3.5 GHz
[15]	1.000	1000.000	GTX 1080ti
[16]	0.600	1666.666	CPU 2.6 GHz

3 METODOLOGÍA

En este capítulo se describe el desarrollo de un modelo computacional capaz de encontrar regiones libres de obstáculos en el entorno cuyo tiempo de procesamiento es mejor que el del estado del arte y le permite formar parte de dispositivos ETA.

La identificación de regiones libres de obstáculos se aborda como un problema de clasificación de parches extraídos de una imagen RGB a través de una ventana deslizante igual que [6, 5]. Este proceso se ilustra en la Figura 3.1. En este ejemplo, la información de los parches resaltada en rojo es procesada 2 veces. Además, unicamente el pixel central del parche es etiquetado. Es por esto que este enfoque es considerado ineficiente [17]. Sin embargo, se seleccionó este abordaje porque el tamaño pequeño de los parches en comparación con la imagen completa produce un tamaño de red también pequeño [6]. Para superar el problema de eficiencia se decidió clasificar parches no superpuestos y etiquetar todos los pixels del parche tal como [5], con lo que se consigue disminuir el número de llamadas al modelo. El aporte de esta investigación consiste en combinar estas dos soluciones para obtener una CNN pequeña y eficiente. Además, con fines comparativos, se presenta un modelo derivado de la ecuación del plano. Pero antes de presentar los modelos, en la Sección 3.1 se presentan conceptos referentes a CNNs.

3.1 Conceptos fundamentales sobre redes neuronales convolucionales

3.1.1 Definición

Una red neuronal convolucional es un perceptrón multicapa diseñado para específicamente para reconocer formas bidimensionales con un alto grado de invarianza a la traslación, escalado, rotación y otras clases de distorsión [69]. Es comúnmente utilizada para procesar imágenes [70].

Esta red recibe su nombre por la operación de convolución que utilizan, que se ilustra en la Figura 3.2. En este ejemplo, un filtro de 3x3 (verde) recorre una entrada de 4x4 (naranja) efectuando operaciones de multiplicación elemento a elemento y agregación para obtener la salida (azul). El número de celdas que el filtro avanza en cada ciclo se denomina paso (*stride* en inglés). La fórmula que permite calcular el tamaño de salida es $(n - f + 1) * (m - f + 1)$ [70], donde n es el alto de la entrada, m es el ancho de la entrada y f es el tamaño del filtro. En este ejemplo puede verificarse que $(4 - 3 + 1) * (4 - 3 + 1) = 4$. La salida tiene menor tamaño porque solamente se utilizaron celdas válidas. Si se desea que la salida tenga el mismo tamaño que la entrada, se puede utilizar un *padding* de ceros igual a 1, tal como muestra la Figura 3.3. Con este arreglo puede verificarse que $(5 - 3 + 1) * (5 - 3 + 1) = 9$.

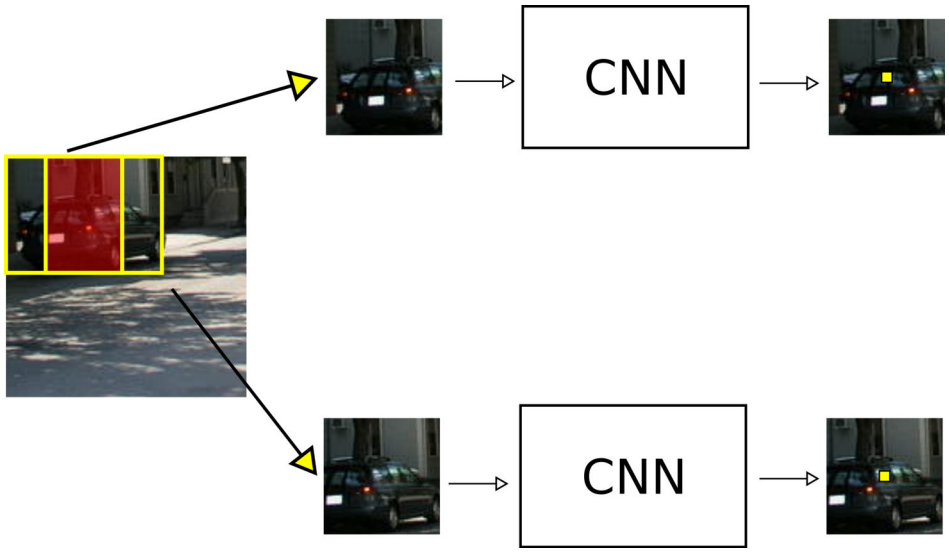


Figura 3.1. Segmentación utilizando ventana deslizante y CNN.

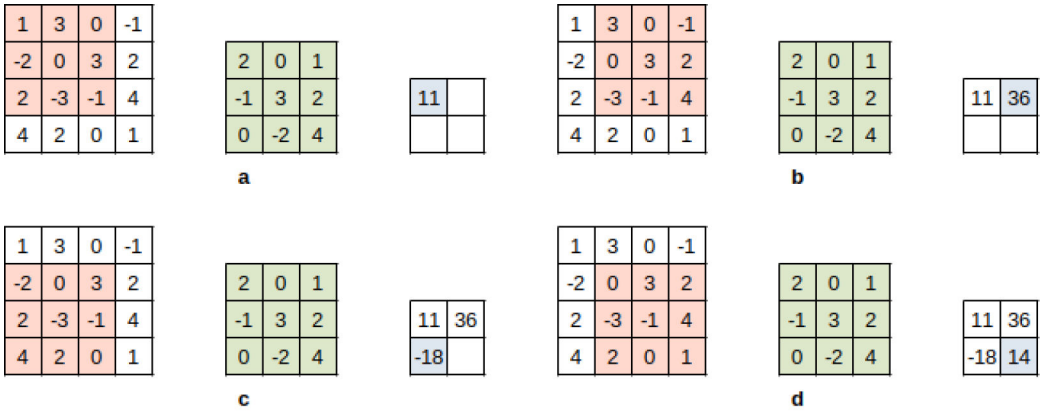


Figura 3.2. Operación de convolución.

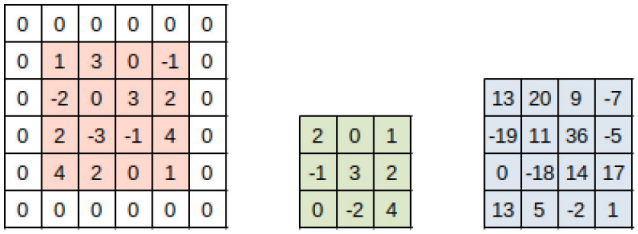


Figura 3.3. *Padding* de ceros.

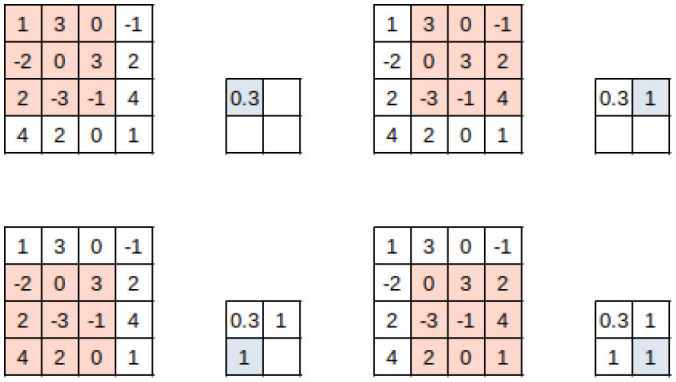


Figura 3.4. Operación de *pooling* de promedio.

3.1.2 Estructura

La estructura más común de una CNN esta formada por un bloque que apila capas convolucionales y de *pooling* (muestreo) para sintetizar la información de la imagen, seguido de capas completamente conectadas [70, 71, 72].

Las capas convolucionales realizan operaciones de convolución sobre una matriz. Una capa convolucional queda definida por el tamaño de filtro F, número de filtros K, el paso S y la cantidad de *padding* P. Valores habituales de estos parámetros son F $= 3$, S $= 1$ y P $= 1$ [72]. La entrada y salida de estas capas pueden contener una o más matrices apiladas. A cada una de estas matrices se le denomina canal.

A continuación de una capa de convolución, los valores de la matriz de salida se pasan por una función no lineal (FA). [70] presenta más información respecto a estas funciones.

El número de pesos sinápticos que tiene una capa convolucional se calcula con la Ecuación 3.1, donde NoP es el número de pesos sinápticos y C es el número canales de la entrada. Nótese que este valor no depende del tamaño de las matrices de entrada y salida, a diferencia de las capas completamente conectadas en las que el este valor se calcula con la Ecuación 3.2, donde N_{in} y N_{out} son los tamaños de entrada y salida de la capa respectivamente.

$$NoP = (F \times F \times C + 1) \times K \tag{3.1}$$

$$NoP = (N_{in} + 1) \times N_{out} \tag{3.2}$$

Las capas de *pooling* reducen el tamaño de una matriz a través de operaciones de muestreo sobre sus elementos. Las operaciones más comunes son máximo (maxpooling) y promedio (averagepooling). Además de la operación a utilizar, una capa de *pooling* se define por el tamaño de ventana F y el paso S. Valores habituales de estos parámetros son F $= 2$ y S $= 2$ [72]. La Figura 3.4 muestra una operación de *pooling* de promedio con F $= 3$ y S $= 1$.

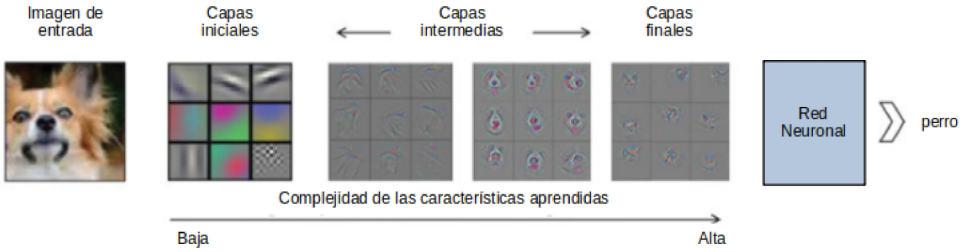


Figura 3.5. Extracción de características en una CNN. Tomado de [70].

3.1.3 Entrenamiento

El entrenamiento de las CNN se realiza en forma similar al de las redes neuronales *feedforward*, utilizando el algoritmo de retropropagación del error [73] para ajustar los valores de los filtros y las capas completamente conectadas. Después del entrenamiento, los filtros deberían actuar como extractores de características que mejoren la clasificación realizada por las capas completamente conectadas, tal como se aprecia en la Figura 3.5.

3.2 Descripción de los modelos propuestos

3.2.1 CNN para imágenes RGB

El modelo propuesto para procesar imágenes RGB es una modificación de la red CN24 presentada por [6] para segmentación de escenas urbanas. La entrada de CN24 es un parche de tamaño 28x28 extraído de una imagen RGB. Después de ser transformado por las capas de convolución y *pooling*, al resultado se añade la posición (x, y) del píxel central del parche en la imagen. Finalmente, la red entrega un vector con las probabilidades de que el píxel central del parche pertenezca a 8 categorías. La estructura de esta red se muestra en la Tabla 3.1. La primera columna presenta un identificador, la segunda columna indica el tipo de capa, la tercera columna muestra los parámetros de configuración, la cuarta columna presenta el tamaño de la salida y la quinta columna indica la cantidad de pesos sinápticos generados.

Para disminuir el número de pesos sinápticos de la red, se propuso utilizar tamaños de entrada más pequeños. Por ejemplo, con una entrada de tamaño 16x16, se generarían 22036 pesos en lugar de 45204. Se sugirieron 16x16 y 8x8 por [72], que recomienda que las dimensiones de la entrada sean divisibles para 2 varias veces. De acuerdo con [72, 74, 75], se seleccionaron F $= 3$, S $= 1$ y P $= 1$ en las capas 1 y 4. Debido a que P $= 1$, la nueva red poseería 1 simplificación en lugar de 3 como en la red original. Para compensar esta pérdida, se repitió la capa 2 después de la capa 5. Continuando con la tarea de reducir el tamaño de la red, se propuso utilizar una sola capa completamente conectada tal como en [7]. Se seleccionó la capa de 64 neuronas. Además, teniendo en cuenta que uno de los objetivos de la investigación es segmentar el entorno en dos clases (libre y obstáculo), se reemplazaron las 8 neuronas de la capa de salida por 2. Finalmente, para normalizar la salida de la red se reemplazó la función sigmoidea por softmax. La estructura de la nueva red se muestra en la Tabla 3.2. La primera columna muestra el tipo de capa y la segunda columna

Tabla 3.1. Modelo propuesto por [6] para segmentar imágenes RGB.

Id	Capa	Parámetros	Tamaño de salida	Pesos
0	Entrada		(28, 28, 3)	
1	Convolucional	F = 7, C = 16, S = 1, P = 0	(22, 22, 16)	2368
2	*Maxpooling*	F = 2, S = 2	(11, 11, 16)	
3	No lineal	FA = tanh		
4	Convolucional	F = 5, C = 12, S = 1, P = 0	(5, 5, 12)	9420
5	No lineal	FA = tanh		
6	Completamente conectada + posición del parche	n = 64	64	19392
7	No lineal	FA = tanh		
8	Completamente conectada	n = 192	192	12480
9	No lineal	FA = tanh		
10	Completamente conectada	n = 8	8	1544
11	No lineal	FA = sigmoidea		
			Total pesos	45204

Tabla 3.2. Modelo propuesto en esta investigación para encontrar regiones libres de obstáculos en el entorno.

Capa	Parámetros
Convolucional	F = 3, C = 16, S = 1, P = 1, FA = tanh
Maxpooling	F = 2, S = 2
Convolucional	F = 3, C = 12, S = 1, P = 1, FA = tanh
Maxpooling	F = 2, S = 2
Completamente conectada + posición del parche	n = 64, FA = tanh
Completamente conectada	n = 2, FA = softmax

presenta los parámetros de la capa.

Además de reducir el tamaño de la red, se propuso no superponer los parches y etiquetar todos los píxeles de cada parche en lugar de solamente el píxel central para disminuir el tiempo requerido para procesar una imagen. Tómese como ejemplo una imagen de resolución 640x480. Para segmentarla, la propuesta original demandaría 307200 ejecuciones de la red. Por otro lado la nueva propuesta, utilizando una ventana de 16x16, solamente requeriría 1200.

3.2.2 Modelo derivado de la ecuación del plano para nubes de puntos

Como se mencionó en la sección 2.4.2, existen entornos en los que las regiones libres de obstáculos se encuentran contenidas en un plano paralelo al plano xz. Si se dispone de una nube de puntos, es posible reconocer estos elementos utilizando la Ecuación 2.2 porque pertenecen deberían tener coordenada-y igual a la distancia perpendicular sensor-piso. Sin embargo, pueden existir elementos recuperados con esta condición que pertenezcan a obstáculos. Para refinar la búsqueda de estos elementos es posible utilizar la Ecuación 2.5 porque sus vectores normales unitarios deben ser perpendiculares al plano xz y el valor de su componente-y debe aproximarse a 1. Finalmente, con aquellos elementos que cumplen estas dos condiciones se planteó ajustar los coeficientes de la Ecuación 2.3 utilizando el algoritmo RANSAC [76]. El

modelo propuesto se encuentra descrito en el Algoritmo 3.1.

Algoritmo 3.1: Modelo derivado de la ecuación del plano.

Entrada: Nube de puntos, vectores normales unitarios de la nube de puntos, distancia perpendicular sensor-piso, umbral de tolerancia para la distancia perpendicular sensor-piso, umbral de tolerancia para la componente-y del vector normal unitario, parámetros para RANSAC, umbral de tolerancia para pertenencia a la ecuación del plano δ.

Salida: Máscara en la que se encuentran identificadas las regiones libres de obstáculos.

Recuperar los elementos de la nube de puntos que cumplen con las condiciones dadas en las ecuaciones 2.2 y 2.5;

si *Se recuperaron más de 2 elementos* **entonces**

  Ajustar los coeficientes de la ecuación $Ax + By + Cz + D = 0$ utilizando RANSAC;

  Recuperar los elementos de toda la nube de puntos que cumplen las condiciones $Ax + By + Cz + D - \delta > 0$ y $Ax + By + Cz + D + \delta < 0$;

  Devolver la máscara en la que se encuentra identificados los elementos recuperados;

en otro caso

  Devolver una máscara vacía;

fin

3.2.3 CNN para nubes de puntos

Junto a la ecuación del plano, existen otros modelos que permiten encontrar regiones libres de obstáculos en nubes de puntos. Entre estos se encuentran las CNNs, que en algunos casos ha presentado mejor desempeño. Compárese, por ejemplo, [21] con [13, 77]. Esta red es igual a la CNN para imágenes RGB presentada en la Tabla 3.2. Sin embargo, el parche correspondiente a la entrada no se extrae de una imagen RGB sino de un mapa de características formado por los valores de una nube de puntos y su correspondiente mapa de vectores normales. La relevancia de estos valores para identificar regiones libres de obstáculos se presentó en la Sección 3.2.2.

3.3 Repositorios de datos

3.3.1 Labelmefacade

Para medir el desempeño de la CNN para imágenes RGB se seleccionó el repositorio Labelmefacade [6] porque fue utilizado para evaluar CN24 y permite la comparación entre ambos modelos. Está compuesto por 945 imágenes RGB de tamaño 683x512 píxeles recuperadas de entornos exteriores y sus correspondientes máscaras verdaderas. 100 imágenes están destinadas a entrenamiento y 845 a evaluación. Posee 8 clases: *bulinding*, *car*, *door*, *pavement*, *road*, *sky*, *vegetation* y *window*. La Figura 3.6 muestra la frecuencia con la que aparece cada clase en el conjunto de entrenamiento. Se encuentra disponible en `https://github.com/cvjena/labelmefacade`.

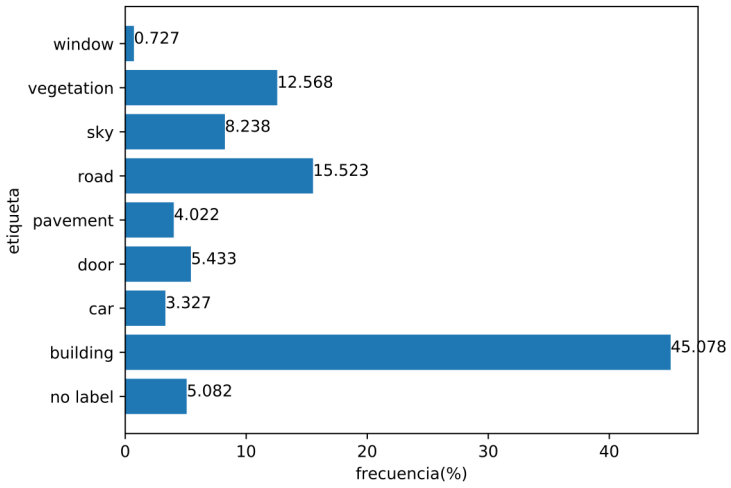


Figura 3.6. Frecuencia de clases en el repositorio Labelmefacade.

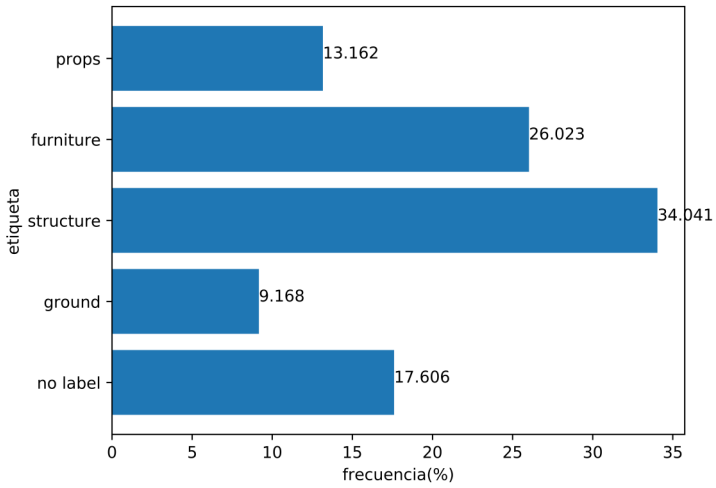


Figura 3.7. Frecuencia de clases en el repositorio NYU-v2.

3.3.2 NYU-v2

Para medir el desempeño de la CNN para imágenes RGB, el modelo derivado de la ecuación del plano y la CNN para nubes de puntos se emplea el repositorio de datos NYU-v2 [78]. Se seleccionó este repositorio porque posee información espacial, complementa el repositorio Labelmefacade porque fue recuperado en entornos interiores y sirve como punto de referencia puesto que ha sido utilizado para evaluar diversos algoritmos [79, 13, 12, 77]. NYU-v2 está compuesto por 1449 imágenes RGBD de resolución 640x480, sus correspondientes máscaras verdaderas y ángulos de Euler del sensor en el momento de la captura. Originalmente, el conjunto fue dividido en 795 imágenes para entrenamiento y 654 para evaluación. NYU-v2 posee 894 clases, pero en esta investigación se utiliza la variante que tiene 4 [78]: *ground*, *structure*, *furniture* y *props*. La Figura 3.7 muestra la frecuencia con la que aparece cada clase en el conjunto de entrenamiento. Se encuentra disponible en `https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html`.

Tabla 3.3. Configuración de la cámara.

N.	Parámetro	Valor
1	Resolución	WVGA
2	Profundidad máxima	5 m.
3	*Sensing mode*	FILL
4	*Depth mode*	ULTRA
5	Sistema de referencia	y-up

3.3.3 FIS-EPN

Además de Labelmefacade y NYU-v2, para la evaluación de los modelos se recolectó un conjunto de escenas en el edificio FIS-EPN, Cada una está formada por: imagen RGB, nube de puntos, mapa de vectores normales y máscara verdadera, todos de tamaño 336x188. También contiene un archivo con los ángulos de Euler del sensor en el momento de captura de cada elemento. El conjunto consta de 500 elementos y está dividido en 350 para entrenamiento y 150 para evaluación. Se seleccionó esta cantidad de escenas porque es comparable con el número de muestras de los otros conjuntos de datos.

Adquisición de datos

La información fue recuperada con una cámara estereoscópica Zed Mini. La configuración de la cámara se muestra en la Tabla 3.3. Los demás parámetros se dejaron con sus valores por defecto. Con la cámara se realizaron capturas cada 500 ms, con un ser humano transportándola a través de diferentes espacios del edificio. El individuo mantuvo la cámara a 1.27 m. del suelo.

Generación de máscaras verdaderas

Las máscaras verdaderas son empleadas como referencia para evaluar las predicciones de la red. Para esta investigación, se generaron empleando la capacidad de un ser humano para identificar regiones libres de obstáculos. Las imágenes RGB fueron segmentadas empleando la herramienta Labelme [80] y siguiendo las recomendaciones utilizadas para segmentar el conjunto Camvid [81] para reconocer dos clases: *floor* y *obstacle*. En el conjunto de entrenamiento se evitó segmentar pixels cercanos a las fronteras de planos para reducir la ambigüedad en la información. En el conjunto de evaluación se trató de segmentar la mayor cantidad de pixels. La Figura 3.8 muestra la frecuencia con la que aparece cada clase en el conjunto de entrenamiento.

3.4 Implementación de los modelos

Todos los experimentos fueron desarrollados en Python, salvo que se indique lo contrario. Para las CNNs se utilizaron las librerías Tensorflow [82], Keras [83] y OpenCV [84]. La configuración del entrenamiento de estos modelos se muestra en la Tabla 3.4. Los demás parámetros se dejaron con sus valores por defecto. El modelo derivado de la ecuación del plano fue desarrollado también en Python utilizando OpenCV y la librería Scikit-learn [85].

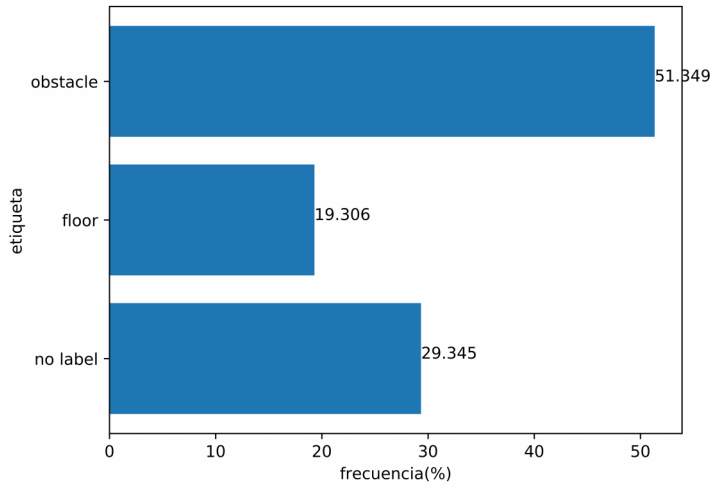


Figura 3.8. Frecuencia de clases en el repositorio FIS-EPN.

Tabla 3.4. Configuración del entrenamiento de la CNN.

N.	Parámetro	Valor
1	*Holdout*	0.2
2	Épocas	128
3	Paciencia	8
3	Lote	750
4	Optimizador	Descenso de gradiente estocástico
5	Función de pérdida	Entropía cruzada
6	Métrica de evaluación	Exactitud ponderada

Respecto al hardware utilizado para medir los tiempos de ejecución, es un computador i7 @ 2.2 GHz con 8 Gb de RAM. El sistema operativo instalado es Ubuntu 18.04.5 LTS.

4 RESULTADOS

En este capítulo se presentan los resultados de la evaluación de la capacidad de los modelos propuestos en el Capítulo 3 para encontrar regiones libres de obstáculos en imágenes RGB y nubes de puntos. En la Sección 4.1 se presentan la evaluación de la CNN para imágenes RGB utilizando los repositorios Labelmefacade, NYU-v2 y FIS-EPN. En la Sección 4.2 se presenta la evaluación del modelo derivado de la ecuación del plano utilizando los repositorios NYU-v2 y FIS-EPN. En la Sección 4.3 se presentan los resultados de la evaluación de la CNN para nubes de puntos utilizando los repositorios NYU-v2 y FIS-EPN. Finalmente, en la Sección 4.4 se discuten las causas e implicaciones de los resultados obtenidos.

4.1 Experimentos con la CNN para imágenes RGB

4.1.1 Segmentación del repositorio Labelmefacade con 8 clases

Este experimento se realizó para comparar la red propuesta con CN24. Se utilizaron 8 neuronas en la capa de salida. La Tabla 4.1 muestra los resultados obtenidos por CN24 y las variantes del modelo propuesto. La primera columna muestra los nombres de las variantes. Las columnas 2 y 3 presentan las métricas utilizadas: exactitud general y exactitud balanceada. La columna 4 muestra el tiempo requerido por las variantes para procesar una imagen RGB de tamaño 320x240. Teniendo en cuenta que se utilizó un conjunto no balanceado, la exactitud balanceada es más adecuada que la general para evaluar el rendimiento de los modelos [86, 87]. En este sentido, puede decirse que modelos presentan rendimientos similares. Sin embargo, el modelo 16x16 exhibe el mejor tiempo de procesamiento, correspondiente a 0.270% del original. La Figura 4.1 muestra la matriz de confusión del modelo 16x16.

Tabla 4.1. Resultados obtenidos por la CNN para imágenes RGB sobre el repositorio Labelmefacade con 8 clases.

Modelo	Exactitud general	Exactitud balanceada	Tiempo de procesamiento (ms)
CN24	72.200	47.700	13329.418
Modelo 16x16	49.473	47.998	36.476
Modelo 8x8	49.183	48.168	48.239

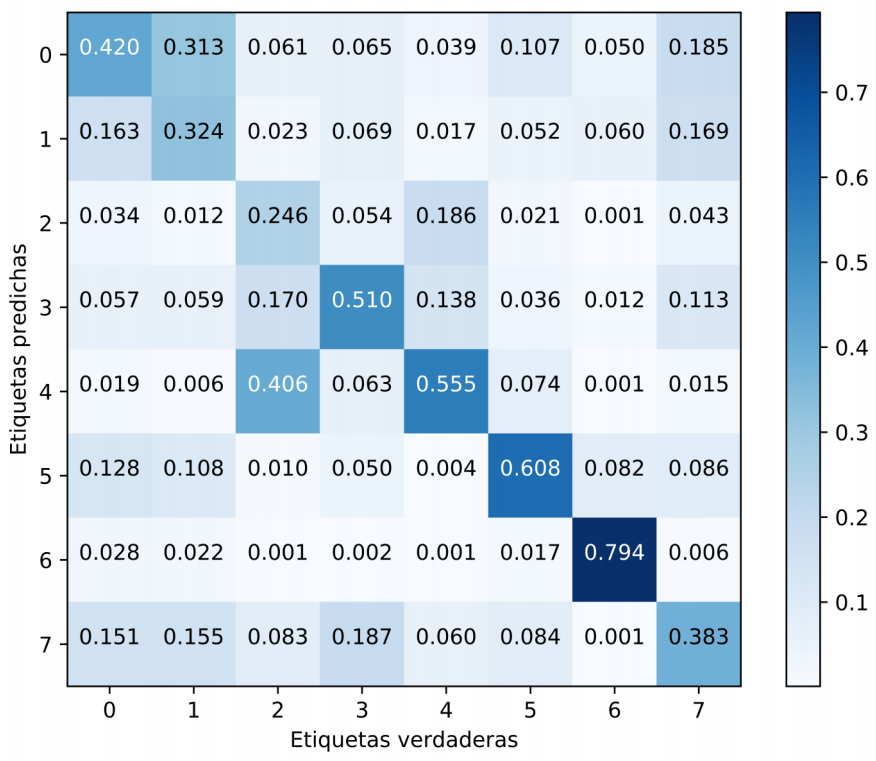


Figura 4.1. Matriz de confusión obtenida por el modelo 16x16 en el repositorio Labelmefacade con todas las clases. 0 = *building*, 1 = *car*, 2 = *door*, 3 = *pavement*, 4 = *road*, 5 = *sky*, 6 = *vegetation*, 7 = *window*.

Tabla 4.2. Resultados obtenidos por la CNN para imágenes RGB sobre el repositorio Labelmefacade con 2 clases.

Modelo	Exactitud de la clase libre	Exactitud balanceada	IoU
Modelo 16x16	91.260	90.234	67.381
Modelo 8x8	90.700	90.136	68.152

4.1.2 Segmentación con 2 clases

Dado que en esta investigación interesa reconocer las regiones libres de obstáculos del resto, en este experimento las clases *road* y *pavement* del repositorio Labelmefacade se agruparon como libre porque representan espacios donde un peatón puede transitar. El resto de clases fueron agrupadas como obstáculo. La Tabla 4.2 muestra los resultados obtenidos por el modelo propuesto. La primera columna presenta el nombre de la variante del modelo. Las columnas 2, 3 y 4 indican las métricas seleccionadas para la evaluación: exactitud de la clase libre, exactitud balanceada y coeficiente IoU. Se incluyó el coeficiente IoU porque es la métrica más usada para evaluar modelos para segmentación [88]. Puede decirse que ambas variantes presentan desempeños similares. La Figura 4.2 muestra la matriz de confusión del modelo 16x16.

En el repositorio NUY-v2, la clase *ground* se tomó como libre y el resto de clases fueron agrupadas como obstáculo. La Tabla 4.3 muestra los resultados obtenidos por el modelo propuesto. La primera columna presenta el nombre de la variante del modelo. Las columnas 2, 3 y 4 muestran las métricas seleccionadas para la evaluación: exactitud de la clase libre, exactitud balanceada y coeficiente IoU. Puede decirse que

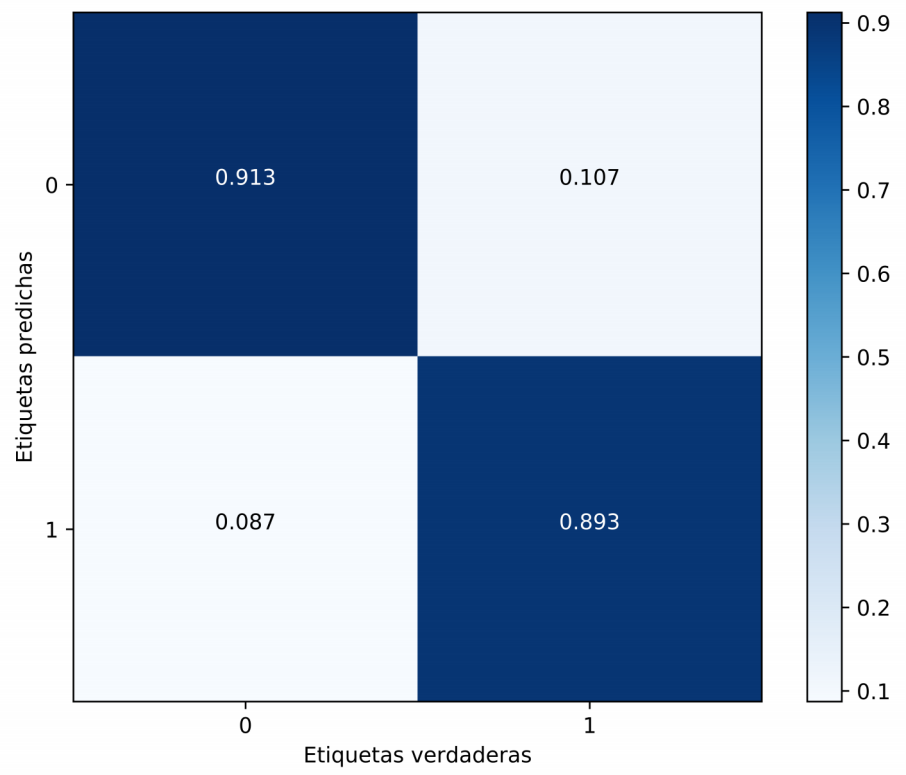


Figura 4.2. Matriz de confusión obtenida por el modelo 16x16 en el repositorio Labelmefacade con 2 clases. 0 = libre, 1 = obstáculo.

Tabla 4.3. Resultados obtenidos por la CNN para imágenes RGB sobre el repositorio NYU-v2 con 2 clases.

Modelo	Exactitud de la clase libre	Exactitud balanceada	IoU
Modelo 16x16	83.000	81.640	32.590
Modelo 8x8	86.330	82.226	31.339

ambas variantes presentan desempeños similares. La Figura 4.3 muestra la matriz de confusión del modelo 16x16.

La Tabla 4.4 muestra los resultados obtenidos por el modelo propuesto en la clasificación del repositorio FIS-EPN. La primera columna presenta el nombre de la variante del modelo. Las columnas 2, 3 y 4 muestran las métricas seleccionadas para la evaluación: exactitud de la clase libre, exactitud balanceada y coeficiente IoU. Ambas variantes presentan desempeños similares. La Figura 4.4 muestra la matriz de confusión del modelo 16x16.

Tabla 4.4. Resultados obtenidos por la CNN para imágenes RGB sobre el repositorio FIS-EPN.

Modelo	Exactitud de clase	Exactitud balanceada	IoU
Modelo 16x16	53.900	65.332	41.504
Modelo 8x8	66.800	63.867	42.952

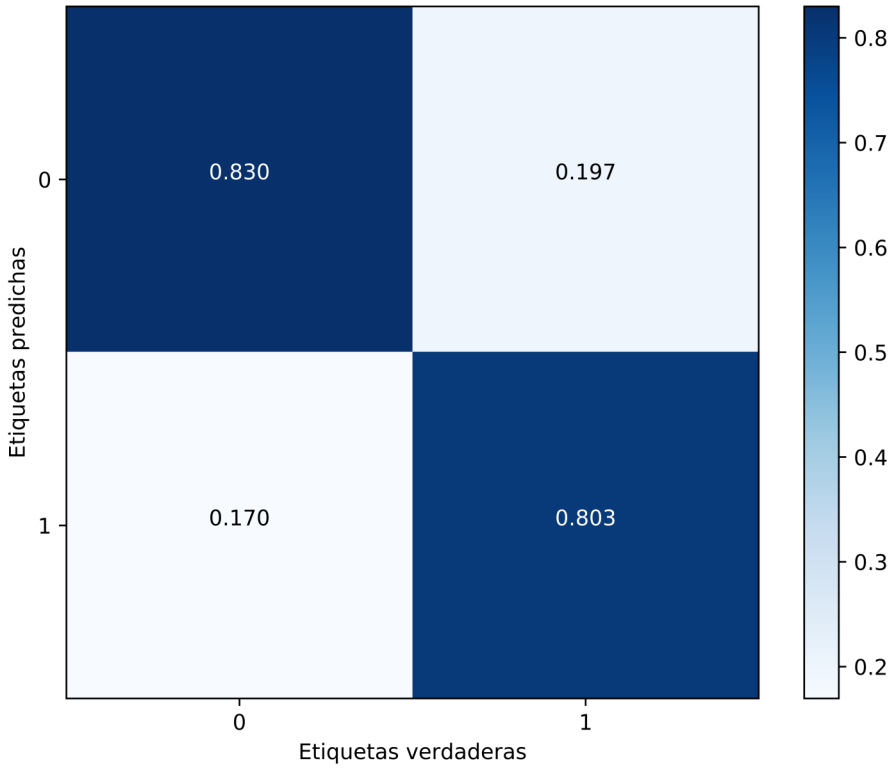


Figura 4.3. Matriz de confusión obtenida por el modelo 16x16 en el repositorio NYU-v2 con 2 clases. 0 = libre, 1 = obstáculo.

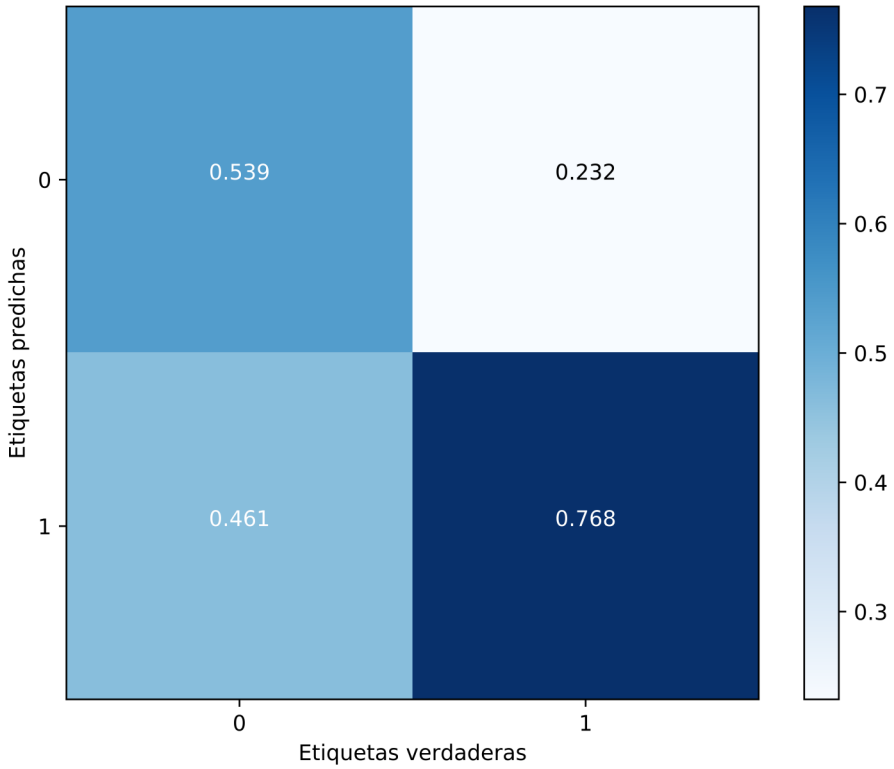


Figura 4.4. Matriz de confusión obtenida por el modelo 16x16 en el repositorio FIS-EPN. 0 = *floor*, 1 = *obstacle*.

Tabla 4.5. Resultados obtenidos por el modelo derivado de la ecuación del plano sobre el repositorio NYU-v2.

Modelo	Exactitud de clase	Exactitud balanceada	IoU
Depth-cut based Ground Detection [21]	84.000	90.500	
Modelo derivado de la ecuación del plano	88.700	92.480	64.958

4.2 Experimentos con el modelo derivado de la ecuación del plano

4.2.1 Segmentación del repositorio NUY-v2

El primer paso para desarrollar este modelo fue encontrar la regla para discriminar los elementos de las nubes de puntos por su coordenada-y. La Figura 4.5 muestra el histograma con los valores de coordenada-y con frecuencia mayor a 1% de los elementos marcados como *ground* en las nubes de puntos de entrenamiento. Teniendo estos valores en cuenta, fueron seleccionados como puntos pertenecientes a regiones libres de obstáculos aquellos con coordenada-y dentro del rango $[-1.5, -1.2]$ m. Se utilizó la librería Open3D [89] para obtener la nubes de puntos a partir de las imágenes RGB-D de NYU-v2.

El segundo paso fue encontrar la regla para discriminar los elementos de las nubes de puntos por su vector normal unitario. La Figura 4.6 muestra el histograma con los valores de componente-y del vector normal unitario los elementos con frecuencia mayor a 1% marcados como *floor* en las nubes de puntos de entrenamiento. Teniendo esto en cuenta, fueron seleccionados como puntos pertenecientes a regiones libres de obstáculos aquellos con valor absoluto de la componente-y del vector normal unitario mayor a 0.8. Se utilizó la librería Open3D [89] para obtener los mapas de vectores normales unitarios a partir de las nubes de puntos. Utilizando las 2 reglas expuestas se recuperó 71.560% de todos los puntos que pertenecen a la clase *floor*.

Para cada nube se recuperaron los puntos que cumplen las 2 condiciones anteriores y se ajustó una ecuación del plano utilizando RANSAC con valor residual de 0.01 m. La identificación del plano del suelo con esta ecuación se extendió utilizando un umbral de tolerancia de $\pm$ 5cm.

La Tabla 4.5 muestra los resultados obtenidos por el modelo propuesto sobre el conjunto de evaluación de NYU-v2 y los presentados por [21], cuyo modelo también se deriva de la ecuación del plano. La Figura 4.7 muestra la matriz de confusión del modelo propuesto.

4.2.2 Segmentación del repositorio FIS-EPN

La Figura 4.8 muestra el histograma con los valores de coordenada-y con frecuencia mayor a 1% de los elementos marcados como *floor* en las nubes de puntos de entrenamiento. Teniendo estos valores en cuenta, fueron seleccionados como puntos pertenecientes a regiones libres de obstáculos aquellos con coordenada-y dentro del rango $[-1.5, -1.1]$ m.

La Figura 4.9 muestra el histograma con los valores de componente-y del vector normal unitario los elementos marcados como *floor* en las nubes de puntos de entrenamiento con frecuencia mayor a 1%. Esta figura no resulta tan adecuada para identificar un umbral como el histograma de los elementos marcados

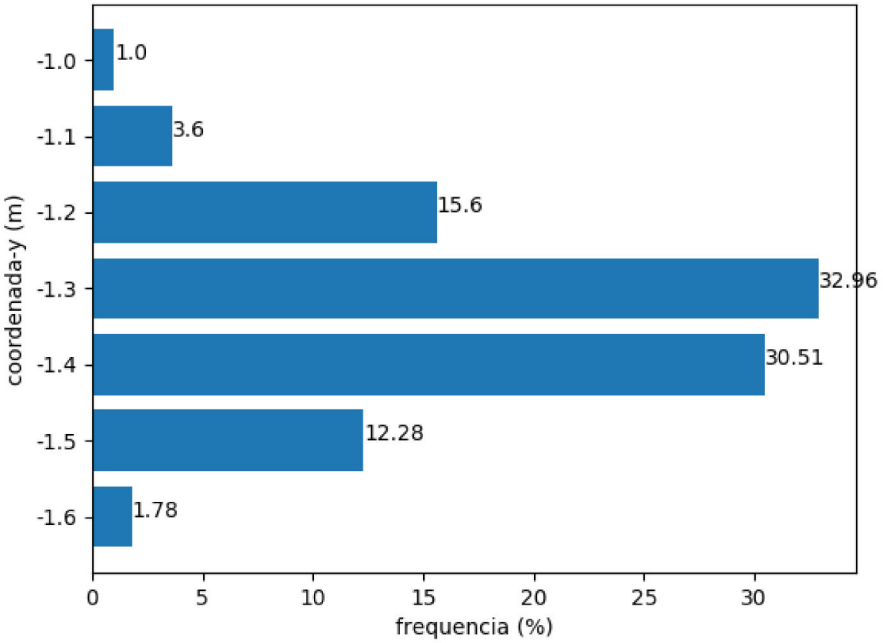


Figura 4.5. Valores de la coordenada-y de puntos marcados como *ground* en NYU-v2.

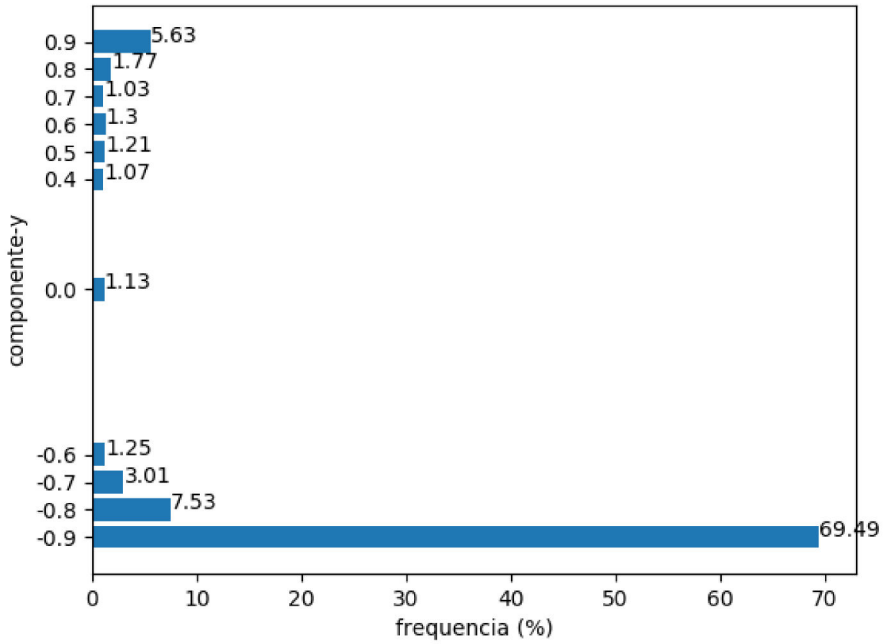


Figura 4.6. Valores de la componente-y del vector normal unitario de puntos marcados como *floor* en NUY-v2.

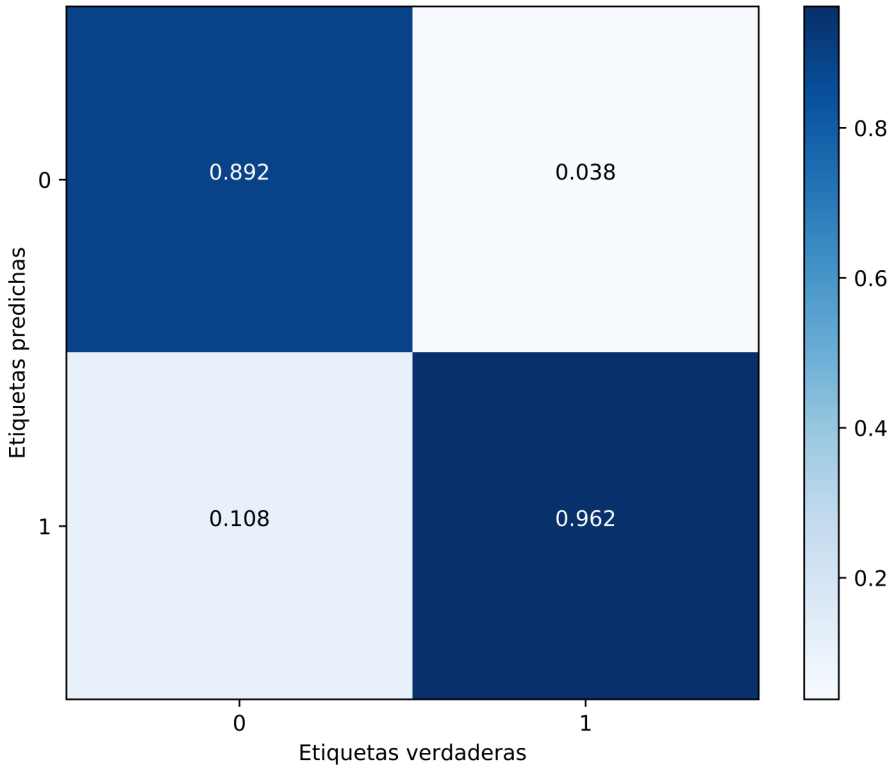


Figura 4.7. Matriz de confusión obtenida por el modelo derivado de la ecuación del plano en NUY-v2. 0 = *floor*, 1 = *obstacle*.

Tabla 4.6. Resultados obtenidos por el modelo derivado de la ecuación del plano sobre el repositorio FIS-EPN.

Modelo	Exactitud de clase	Exactitud balanceada	IoU
Modelo derivado de la ecuación del plano	70.420	68.750	12.564

como *obstacle* mostrado en la Figura 4.10, en el que la mayoría de valores están concentrados en el rango $[-0.1, 0.2]$. Teniendo esto en cuenta, fueron seleccionados como puntos pertenecientes a regiones libres de obstáculos aquellos con valor de la componente-y del vector normal unitario mayor o igual a 0.3. Utilizando las 2 reglas expuestas se recuperó 20.123% de todos los puntos que pertenecen a la clase *floor*.

La Tabla 4.6 muestra los resultados obtenidos por el modelo sobre el conjunto de evaluación de FIS-EPN. La Figura 4.11 muestra la matriz de confusión del modelo.

4.3 Experimentos con la CNN para nubes de puntos

4.3.1 Segmentación del repositorio NYU-v2 con 4 clases

Este experimento se realizó para comparar el modelo con el estado del arte relativo a segmentación del repositorio NYU-v2. Se utilizaron 4 neuronas en la capa de salida de la CNN. La Tabla 4.7 muestra la exactitud obtenida por la red propuesta y 4 modelos más. La primera columna muestra los nombres de los modelos. La siguientes columnas muestran la exactitud alcanzada en las clases *ground*, *struct*, *furniture*,

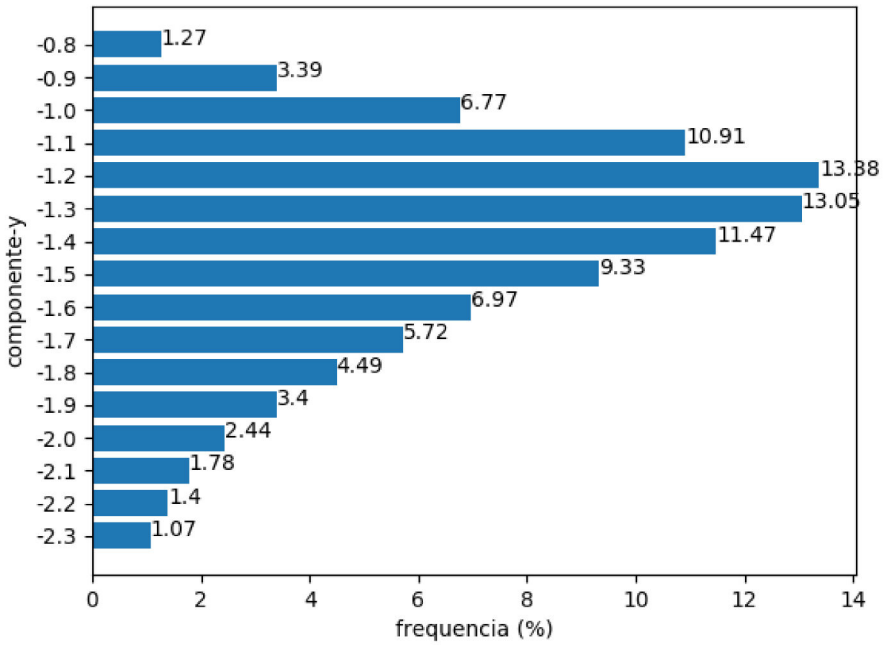


Figura 4.8. Valores de la coordenada-y de puntos marcados como *floor* en FIS-EPN.

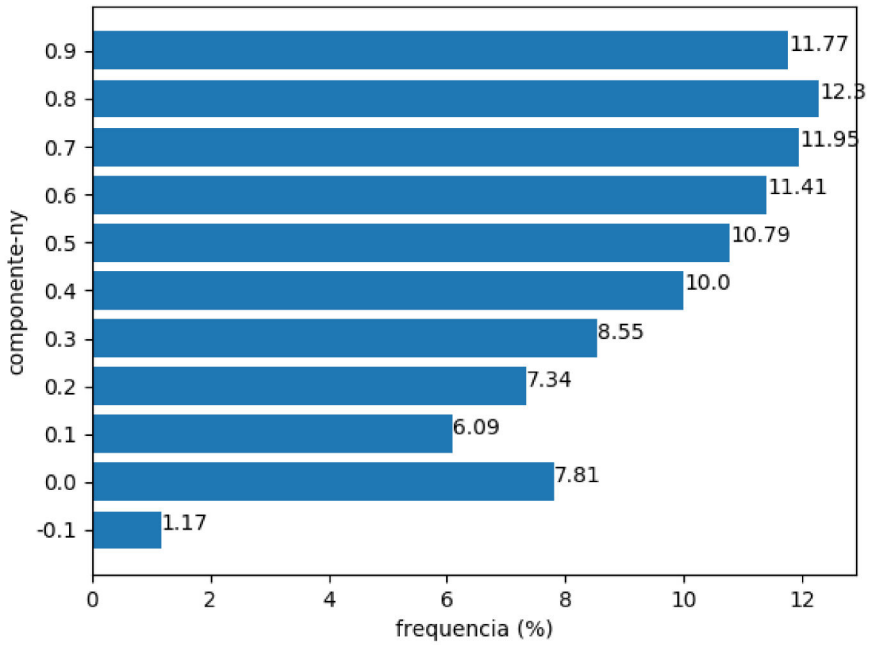


Figura 4.9. Valores de la componente-ny del vector normal de puntos marcados como *floor* en FIS-EPN.

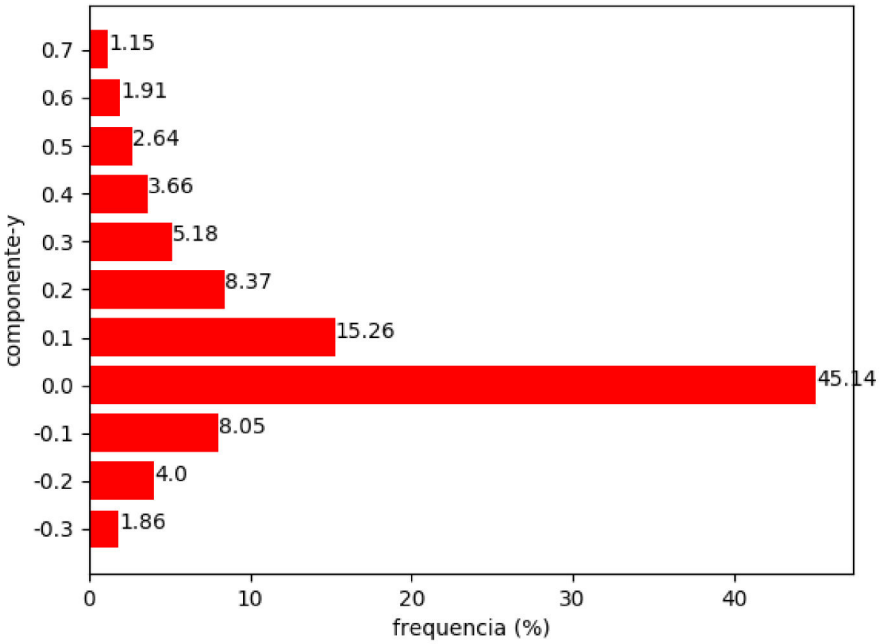


Figura 4.10. Valores de la componente-y del vector normal de puntos marcados como *obstacle* en FIS-EPN.

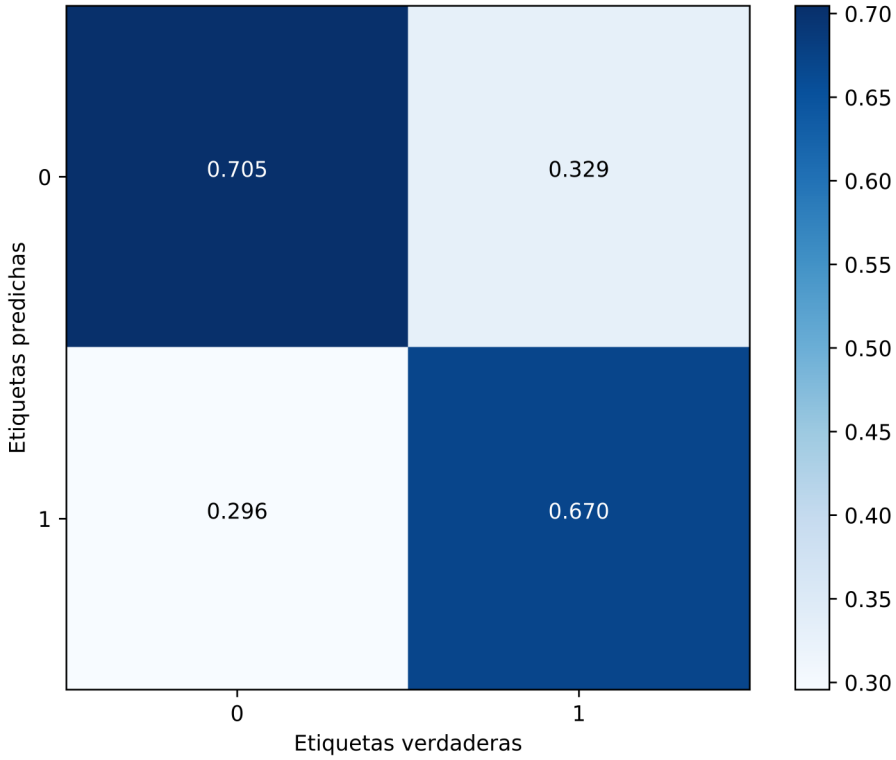


Figura 4.11. Matriz de confusión obtenida por el modelo derivado de la ecuación del plano en EPN-FIS.

Tabla 4.7. Exactitud de clase obtenida por la CNN para nubes de puntos y otros modelos sobre el conjunto NUY-v2 con 4 clases.

Modelo	Ground	Struct	Furniture	Props	Balanceada
CNN [13]	95.800	81.900	72.800	67.200	79.225
3-D Entangled Forests [79]	98.800	87.900	73.900	40.400	75.025
Random Forest + Conditional Random Fields [12]	95.500	80.500	77.100	35.300	72.100
CNN + Conditional Random Fields [90]	93.500	80.200	66.400	54.900	73.700
Modelo 16x16x6	92.970	61.100	62.700	39.670	64.111
Modelo 8x8x6	94.430	56.350	65.870	34.700	62.841

Tabla 4.8. Resultados obtenidos por la CNN para nubes de puntos sobre el repositorio NYU-v2 con 2 clases.

Modelo	Exactitud de clase	Exactitud balanceada	IoU
Modelo 16x16x6	92.500	94.726	74.667
Modelo 8x8x6	94.730	95.849	76.534

props y la exactitud balanceada. Los resultados fueron ordenados de mayor a menor de acuerdo a la exactitud balanceada. La Figura 4.12 muestra la matriz de confusión del Modelo 8x8x6.

De los 4 trabajos seleccionados, solamente [79] reporta el tiempo de procesamiento por cuadro de su modelo: 730.000 ms cuando se ejecuta en un computador i7 @ 2.4 GHz. A este respecto, el tiempo de procesamiento del modelo 8x8x6 es 155.478 ms. Para calcular este tiempo el modelo fue implementado en C++ en lugar de Python.

4.3.2 Segmentación del repositorio NYU-v2 con 2 clases

Este experimento se llevó a cabo para comparar la CNN para nubes de puntos con el modelo derivado de la ecuación del plano. La clase *ground* permaneció separada y las demás fueron agrupadas como *obstacle*.

La Tabla 4.8 muestra los resultados obtenidos por la red. La primera columna presenta el nombre de la variante, la segunda columna indica la exactitud de la clase *ground*, la tercera columna muestra la exactitud balanceada y la cuarta columna presenta el coeficiente IoU.

4.3.3 Segmentación del repositorio FIS-EPN

En este experimento se incluyeron variantes que reemplazan las operaciones de maxpooling por average-pooling con la intención de reducir el efecto de valores atípicos a través de la operación promedio. La Tabla 4.9 muestra los resultados obtenidos por las variantes del modelo. La primera columna describe la variante, la segunda columna muestra la exactitud de la clase *floor*, la tercera columna indica la exactitud balanceada y la cuarta columna presenta el coeficiente IoU. Tratándose de un problema de segmentación, el modelo 8x8x6 *averagepooling* fue tomado como superior porque presenta el coeficiente IoU más alto. Finalmente, teniendo en cuenta que uno de los objetivos de la investigación fue desarrollar un modelo compacto, se realizó un estudio de la reducción del número de neuronas en la primera capa completamente conectada en el modelo ganador. Los resultados se muestran en la Tabla 4.10. La primera columna presenta el número

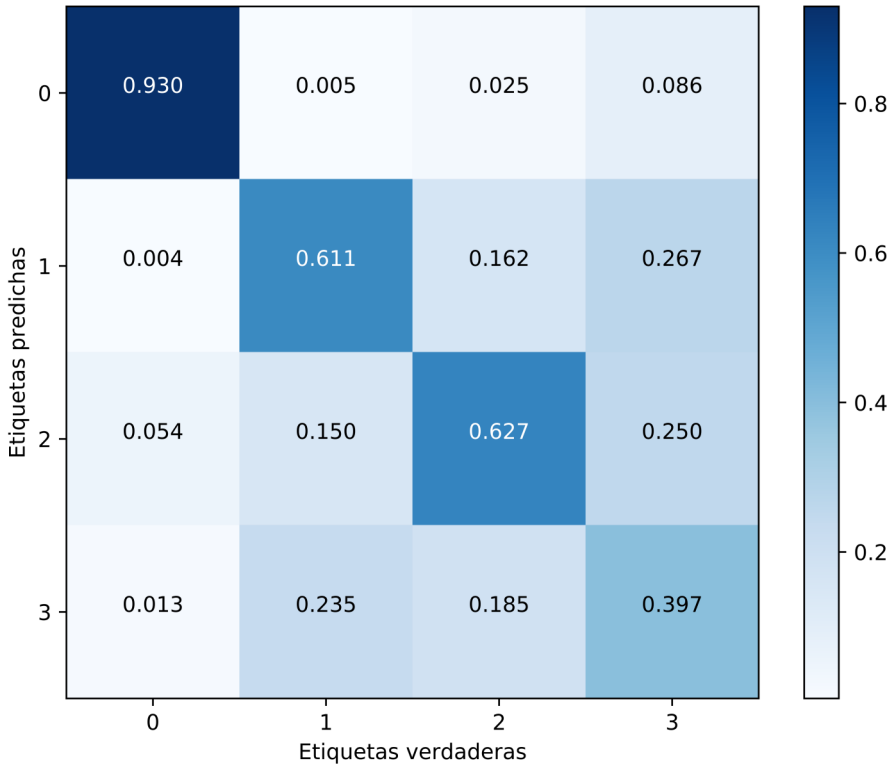


Figura 4.12. Matriz de confusión obtenida por el modelo 8x8x6 en el repositorio NUY-v2 con 4 clases. 0 = *Ground*, 1 = *Struct*, 2 = *Furniture* y 3 = *Props*.

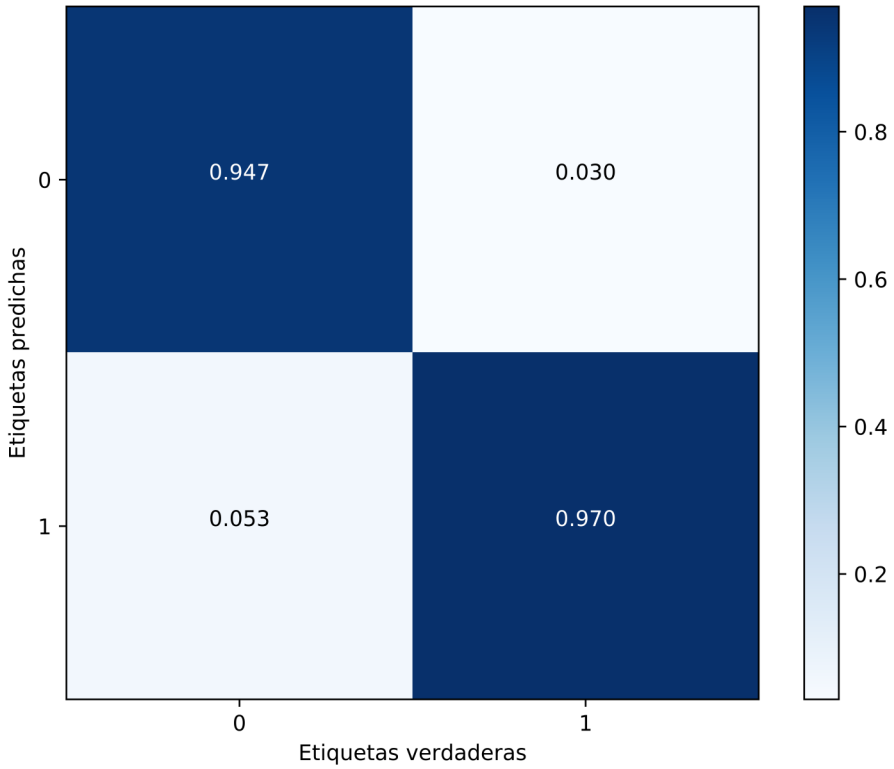


Figura 4.13. Matriz de confusión obtenida por el modelo 8x8x6 en el repositorio NUY-v2 con 2 clases. 0 = *ground*, 1 = *obstable*.

Tabla 4.9. Resultados obtenidos por la CNN para nubes de puntos sobre el repositorio FIS-EPN.

Modelo	Exactitud de clase	Exactitud balanceada	IoU
Modelo 16x16x6, maxpooling	75.600	85.205	87.661
Modelo 16x16x6, averagepooling	91.600	91.894	88.180
Modelo 8x8x6, maxpooling	85.060	90.136	92.130
Modelo 8x8x6, averagepooling	90.670	92.382	95.497

Tabla 4.10. Resultados de la reducción del modelo 8x8x6 *averagepooling* sobre el repositorio FIS-EPN.

Neuronas	Exactitud de clase	Exactitud balanceada	IoU	Pesos
64	90.670	92.382	95.497	5886
32	91.850	92.675	96.778	4254
16	89.940	91.894	86.533	3438

de neuronas en la primera capa completamente conectada, la segunda columna indica la exactitud de la clase *floor*, la tercera columna muestra la exactitud balanceada, la cuarta columna presenta el coeficiente IoU y la quinta columna indica el número de pesos sinápticos de la red. Se observa que la variante con 32 neuronas ofrece el mejor resultado. La Figura 4.14 muestra la matriz de confusión de este modelo.

Utilizando la variante de 32 neuronas se registró el tiempo de procesamiento de 6000 escenas y se obtuvo un promedio de 14.457 ms.

4.4 Discusión

Sobre el repositorio Labelmefacade, la CNN para imágenes RGB mantuvo la exactitud balanceada de la red original pero con tiempo de procesamiento menor. Esto implica que la reducción del número de pesos sinápticos, la decisión de no superponer los parches de entrada de la red y etiquetar todos los pixels de cada parche no afectan el rendimiento de la red pero disminuyen su tiempo de respuesta. De esta manera se cumplieron las partes del objetivo de la investigación relativas a desarrollar la capacidad de encontrar regiones libres de obstáculos en el entorno y mejorar el tiempo de procesamiento del estado del arte. Además, con base en estos resultados es posible afirmar que la hipótesis es verdadera.

Cuando la CNN para imágenes RGB se utilizó para distinguir 2 clases, exhibió mejor coeficiente IoU en el repositorio Labelmefacade que en NUY-v2 y FIS-EPN. Esto es causado porque Labelmefacade fue recuperado en entornos exteriores donde las regiones libres de obstáculos están fabricadas principalmente con asfalto y concreto mientras que los otros 2 fueron recuperados en entornos interiores donde dichas regiones están fabricadas de más materiales como diferentes acabados de madera, alfombras, baldosas, etc. Este hecho hace más difícil la tarea de clasificación de NYU-v2 y FIS-EPN.

El modelo derivado de la ecuación del plano alcanzó mayor coeficiente IoU que la CNN para imágenes RGB sobre el repositorio NYU-v2. Esto muestra que este modelo maneja mejor la dificultad generada por la diversidad de materiales ya que depende de la geometría del entorno y no de su color o textura. Además, obtuvo mejores resultados que el modelo presentado por [21] que igualmente se deriva de la ecuación del plano. Sin embargo, sobre el repositorio FIS-EPN ocurrió lo contrario. Nótese que en este repositorio el

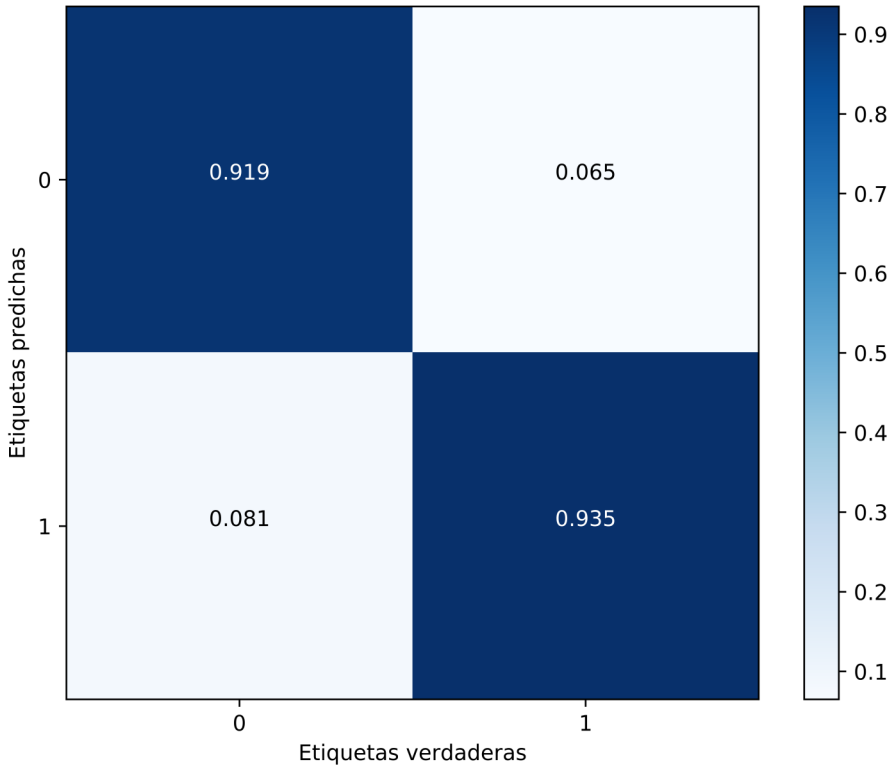


Figura 4.14. Matriz de confusión obtenida por el modelo 8x8x6 *averagepooling* en el repositorio FIS-EPN. 0 = *floor*, 1 = *obstacle*.

número de elementos recuperados con las reglas de discriminación por componente "y" y "ny" es menor al de NYU-v2 a pesar que las regiones libres de obstáculos en ambos repositorios están contenidas en planos. Esto se debe al ruido generado por el sensor durante la adquisición de información.

En la tarea de segmentación del repositorio NYU-v2 con 4 clases, la CNN para nubes de puntos alcanzó exactitud balanceada menor a la presentada por [79] que es el mejor modelo encontrado que reportó tiempo de procesamiento. Sin embargo, resultó ser 4.69 veces más rápida. Esto se debe a la decisión de reducir el tamaño de los parches de entrada y etiquetar todos los pixels del parche que reduce el tiempo de procesamiento a la par de reducir la información contextual disponible [77, 90]. Estos resultados cumplen con el objetivo y apoyan la veracidad de la hipótesis pero no son tan buenos como los alcanzados por la CNN para imágenes en Labelmefacade debido a la reducción de la exactitud.

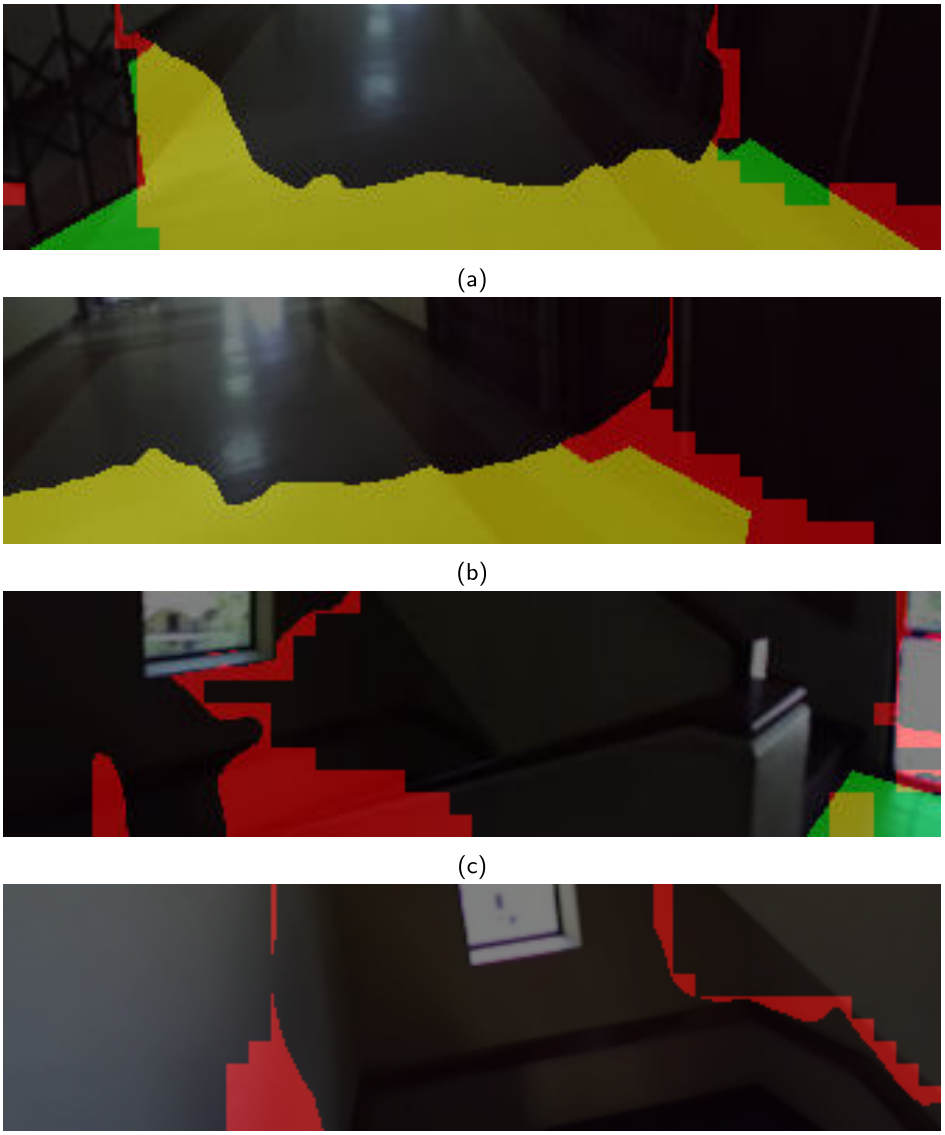
En la tarea de segmentación de los repositorio NYU-v2 y FIS-EPN con 2 clases, la CNN para nubes de puntos alcanzó mejor coeficiente IoU que el modelo derivado de la ecuación del plano. Esto se debe a que dispone de más coeficientes para estimar la función que describe a los elementos de regiones libres de obstáculos. Además, en el modelo utilizado para procesar FIS-EPN se reemplazó la operación maxpooling por averagepooling con lo que se consiguió mejorar el coeficiente IoU. Esto responde a que la operación promedio disminuyó la influencia de puntos atípicos. El número de neuronas en la primera capa completamente conectada se redujo a 32 sin disminuir las métricas de evaluación. El tiempo de procesamiento promedio registrado por este modelo cumple con la parte del objetivo relativa a que el modelo sea capaz de formar parte de un dispositivo ETA porque de acuerdo con [91], el modelo puede procesar 50 escenas

antes que el usuario del dispositivo de un paso.

Gracias a la utilización de la exactitud ponderada durante el entrenamiento de las redes se logró que las clases utilizadas tengan la mayoría de sus elementos correctamente clasificados a pesar de que los repositorios utilizados no están balanceados. Una excepción se presenta en la matriz de confusión de la Figura 4.1 donde la clase *car* se confunde más frecuentemente con *building* y *door* se confunde con *road*. Otra excepción se presenta en la matriz de confusión de la Figura 4.4 donde la clase libre es frecuentemente reconocida como obstáculo.

Las Figuras 5.1a, 5.1b, 4.15c y 4.15d muestran ejemplos de las máscaras generadas utilizando el modelo 8x8x6 maxpooling con 32 neuronas. Posiciones libres de obstáculos correctamente detectadas están resaltadas con amarillo, posiciones libres de obstáculos que no fueron detectadas están resaltadas con verde, posiciones ocupadas por obstáculos incorrectamente detectadas como libres de obstáculos se encuentran pintados con rojo y las posiciones sin información de profundidad no se encuentran resaltadas.

Las Figuras 5.1a y 5.1b son ejemplos en los que las regiones libres de obstáculos fueron apropiadamente identificadas mientras que las Figuras 4.15c y 4.15d presentan inconsistencias ya que se reconocen regiones libres de obstáculos en donde no existen. Este último comportamiento ocurre en regiones correspondientes a obstáculos negativos, cuyo valor de componente-y se encuentra por debajo del límite del suelo. A este respecto, puede decirse que solamente 25 escenas (7.142% del total) contienen obstáculos negativos. Por lo tanto, poseen menor influencia en el entrenamiento de la red y tienen menor probabilidad de ser correctamente clasificados.



(a)

(b)

(c)

(d)

Figura 4.15. Ejemplos de máscaras generadas utilizando la red. Amarillo indica regiones libres de obstáculos correctamente detectadas, verde indica regiones libres de obstáculos que no fueron detectadas y rojo indica obstáculos incorrectamente detectados como regiones libres de obstáculos.

5 INTEGRACIÓN CON DISPOSITIVO ETA

En este capítulo se presenta la integración del modelo entrenado con el repositorio FIS-EPN con el dispositivo ETA FIS-EPN. En la sección 5.1 se describen los módulos del dispositivo ETA antes de la integración de la red. En la Sección 5.2 se presentan los pasos que se dieron para integrar la red en el dispositivo ETA FIS-EPN. En primera instancia la red fue implementada en C++. Luego, se incluyó una etapa de expansión de máscaras para reducir el efecto de las regiones sin información de profundidad. Después se añadió un algoritmo para calcular la trayectoria a seguir y finalmente se realizó el acoplamiento con el módulo de sonido binaural.

5.1 Dispositivo ETA FIS-EPN

El dispositivo ETA desarrollado en la FIS-EPN se muestra en la Figura 5.1. El hardware del dispositivo está formado por una cámara estereoscópica Zed mini, una computadora Jetson TX2, unos audífonos HD 280 PRO y una batería. El software del dispositivo está compuesto por un módulo de detección del obstáculo más cercano, un módulo generador de sonido binaural y un módulo de identificación de objetos.

5.1.1 Módulo de detección del obstáculo más cercano

La entrada de este módulo es una nube de puntos del entorno obtenida utilizando la cámara esterescópica Zed Mini. Como primer paso, se remueven todos los elementos de la nube contenidos en planos horizontales. A continuación se selecciona el punto más cercano a la cámara. El módulo retorna los ángulos de azimuth y elevación del punto más cercano medidos desde la posición de la cámara.

5.1.2 Módulo generador de sonido binaural

La entrada de este módulo son los valores de azimuth y elevación de una dirección determinada. Aprovecha la capacidad de los seres humanos para distinguir la ubicación de una fuente sonora en el entorno. Cumple la función de insertar, en una señal de audio cualquiera, el efecto de posicionamiento tridimensional que permite al oyente determinar la ubicación de una fuente de sonido dentro del entorno. Este efecto se logra convolucionando funciones contenidas en una base de datos de respuestas impulsivas asociadas a cabeza humana (HRIRs por sus siglas en inglés) con una señal de audio [92].

(a) (b)

Figura 5.1. Dispositivo ETA desarrollado en FIS-EPN.

5.1.3 Módulo de identificación de objetos

La entrada de este módulo son imágenes RGB del entorno. Utiliza una CNN es capaz de identificar el tipo de obstáculo que se encuentra frente a la cámara. Su salida es un audio con el nombre del obstáculo.

5.2 Integración de la red

5.2.1 Implementación del modelo

Para ser utilizado en la computadora Jetson TX2, el modelo propuesto en esta investigación fue implementado en C++ utilizando el SDK de la cámara ZED y la librería OpenCV. Se registró el tiempo de procesamiento de 6000 escenas y se obtuvo un promedio de 38.243 ms.

El estudio conducido en [91] sobre un grupo de humanos determinó que el tiempo promedio que una persona demora en dar un paso es 726 ms. Teniendo esto en consideración, el modelo es capaz de procesar 18 escenas antes que el usuario complete un paso.

5.2.2 Expansión de máscaras

En los mapas de profundidad del conjunto FIS-EPN, existen posiciones en las que la cámara entregó valores nulos. La Figura 5.2 muestra una imagen del entorno y el respectivo mapa de profundidad con valores nulos en negro. La CNN no está diseñada para tratar con estos valores. Por lo tanto, sobre la máscara obtenida se aplica una operación de expansión por inundación [93] tal como se hizo en [40].

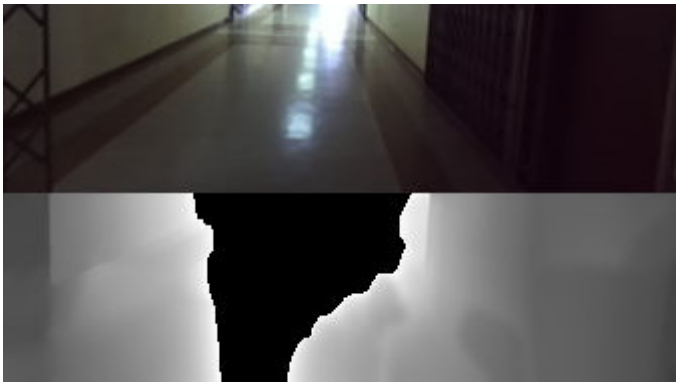


Figura 5.2. Imagen del entorno y mapa de profundidad con valores nulos en negro.

Una vez obtenida la máscara en la que se han identificado la clase a la que pertenecen los pixels que tienen información de profundidad, para determinar la clase a la que pertenecen los pixels restantes se adoptó el procedimiento descrito en el Algoritmo 5.1. Mediante experimentación se determinó la erosión morfológica con un elemento estructurante cuadrado de 11x11 reduce el ruido sal y pimienta de la máscara generado por predicciones incorrectas.

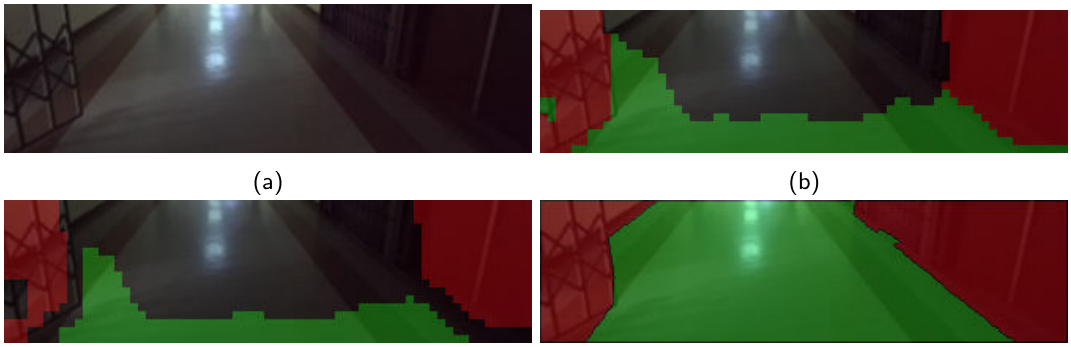
Este procedimiento fue implementado utilizando la librería OpenCV [84] y se ilustra en la Figura 5.3.

Algoritmo 5.1: Expansión de máscaras.

Entrada: Máscara en la que se encuentran identificadas las regiones libres y ocupadas.
Salida: Máscara expandida
Aplicar erosión morfológica con un elemento estructurante cuadrado de 11x11 pixels a todas las regiones;
si *Subsisten regiones libres* **y** *Subsisten regiones ocupadas* **entonces**
  Aplicar algoritmo de expansión por inundación a la máscara erosionada;
en otro caso
  si *Subsisten regiones libres* **entonces**
    Marcar todos los pixels de la máscara como libres;
  en otro caso
    Marcar todos los pixels de la máscara como ocupados;
  fin
fin

5.2.3 Cálculo de la dirección a seguir

La CNN desarrollada en esta investigación puede ser utilizada para calcular la dirección que el usuario del dispositivo ETA debería seguir para transitar por una región libre de obstáculos. Para encontrar esta dirección, se adoptó el procedimiento descrito en el Algoritmo 5.2. La Figura 5.4 muestra el procedimiento de cálculo de la trayectoria para una escena del conjunto FIS-EPN. Se registró el tiempo de procesamiento de 6000 escenas y se obtuvo un promedio de 55.074 ms.



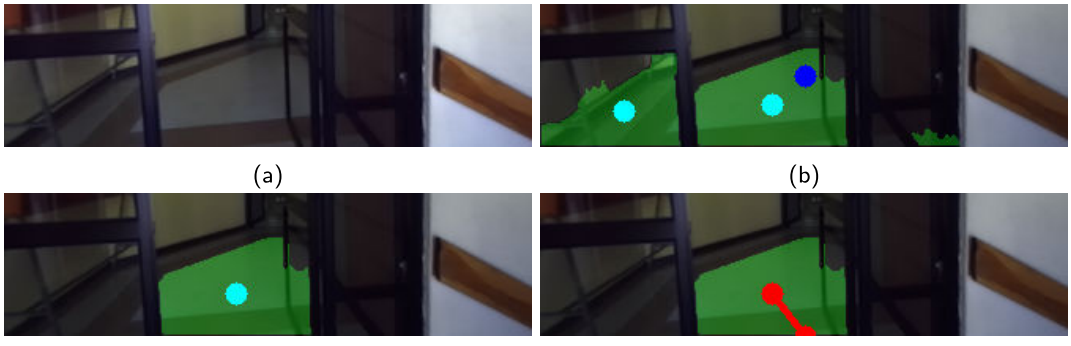
(a) (b)

(c) (d)

Figura 5.3. Proceso de expansión por inundación sobre el conjunto FIS-EPN. (a) Imagen RGB, (b) máscara generada utilizando la CNN, posiciones libres de obstáculos están resaltadas con verde, posiciones ocupadas por obstáculos están resaltadas con rojo, posiciones sin información de profundidad no están resaltadas, (c) máscara de predicciones erosionada, (d) máscara expandida.

Algoritmo 5.2: Cálculo del vector dirección a seguir.

Entrada: Máscara expandida
Salida: Ángulo entre el borde inferior de la máscara y el vector dirección a seguir **o** -1 si no se encontró el vector dirección
Seleccionar las regiones libres con área superior a 3015 pixels;
si *Subsisten regiones libres* **entonces**
  Seleccionar la región con centroide más cercano al centroide de la máscara;
  Calcular el vector dirección como la diferencia entre la posición $(168, 90)$ y el centroide de la región seleccionada;
  Calcular el ángulo que forma el borde inferior de la máscara con el vector dirección, medido en sentido anti-horario;
en otro caso
  Devolver -1;
fin



(a) (b)

(c) (d)

Figura 5.4. Cálculo de trayectoria sobre el conjunto FIS-EPN. (a) imagen de la escena, (b) predicción de regiones libres de obstáculos (verde) con sus correspondientes centroides (turquesa) y centroide de la máscara (azul), (c) selección de la región con centroide más cercano al centroide de la máscara, (d) trayectoria.

(a) (b)

Figura 5.5. Entorno utilizado para pruebas de funcionamiento

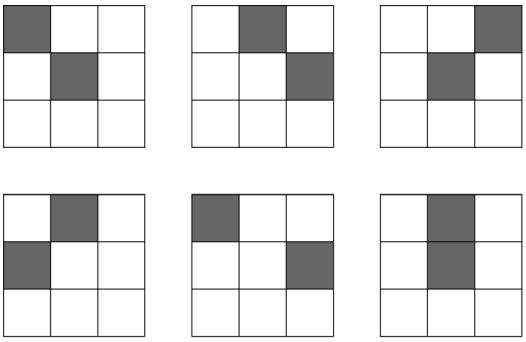


Figura 5.6. Configuraciones del entorno utilizado para probar el funcionamiento del dispositivo ETA.

5.2.4 Integración con el módulo generador de sonido binaural

Para integrarse en el dispositivo ETA, el modelo desarrollado tiene que pasar 2 valores al módulo generador de sonido binaural. Estos valores corresponden a los ángulos de azimuth y elevación de una posición en el entorno. El valor de azimuth corresponde a la dirección a seguir calculada en la Sección 5.2.3. El valor de elevación es 0 en todos los casos porque las regiones libres de obstáculos se encuentran a la misma altura.

Tras la integración de los módulos de identificación de regiones libres de obstáculos y generación de sonido binaural, se registró el tiempo de procesamiento de 6000 escenas y se obtuvo un promedio de 82.594 ms. Esto implica una velocidad de procesamiento de 12 fps. Haciendo referencia a la información presentada en [91], el dispositivo ETA es capaz de actualizar su respuesta 8 veces antes que el usuario complete un paso. Este tiempo de procesamiento constituye una mejora respecto al tiempo original de 300 ms.

Las pruebas de funcionamiento del dispositivo se realizaron en el entorno mostrado en la Figura 5.5, que es similar a los del repositorio NYU-v2. El piso de este entorno es de 2.4 x 2.4 m y fue dividido con una cuadrícula de 3x3. Se generaron 6 distintos escenarios utilizando 2 obstáculos tal como muestra la Figura 5.6. Se contó con la ayuda de un usuario al que se le pidió explorar las diferentes escenas con los ojos vendados. La duración del recorrido en cada escena fue 2 minutos. No se produjeron colisiones con los obstáculos o los límites del entorno durante los recorridos pero la velocidad de desplazamiento del usuario se redujo. La percepción general del usuario fue que el dispositivo es adecuado para la navegación y sus señales son fáciles de entender.

6 CONCLUSIONES Y TRABAJOS FUTUROS

6.1 Conclusiones

El objetivo de esta investigación fue desarrollar un modelo computacional capaz de identificar regiones libres de obstáculos en el entorno, cuyo tiempo de procesamiento mejorara el del estado del arte y le permitiera formar parte de un dispositivo ETA. Para cumplir con el objetivo, la identificación se abordó como un problema de clasificación de parches de una nube de puntos utilizando una CNN. Con la finalidad de reducir el tiempo de procesamiento se decidió no superponer los parches y etiquetar todos los pixels del parche. El modelo fue evaluado en los repositorios Labelmefacade, NYU-v2 y FIS-EPN.

La CNN fue derivada del modelo CN24. Este modelo fue diseñado para procesar información de color. En primer lugar, para comparar ambos modelos, la red fue entrenada con parches de imágenes RGB. De esta manera se obtuvo la CNN para imágenes RGB. Posteriormente, la red fue entrenada con parches extraídos de mapas de características de 6 canales formados por las posiciones x, y y z de una nube de puntos y los valores nx, ny y nz de su correspondiente mapa de vectores normales. Así se obtuvo la CNN para nubes de puntos. La estructura de la red es la misma en ambos casos. Sin embargo, la versión con mayor número de canales en la entrada tiene más parámetros.

En la evaluación del repositorio Labelmefacade con 8 clases, la CNN para imágenes mantuvo la exactitud balanceada (47.700%) con un tiempo de procesamiento equivalente a 0.270% respecto del modelo original. Esto indica que reducir el número de pesos sinápticos de la red, no superponer los parches de entrada y etiquetar todos los elementos de cada parche no afectan el rendimiento del modelo pero disminuyen su tiempo de respuesta.

En la evaluación del repositorio NYU-v2 con 4 clases, la CNN para nubes de puntos obtuvo exactitud balanceada 10.914% menor a la del mejor modelo encontrado que reportó tiempo de procesamiento (76.025%). Esto se debe a la reducción de la información contextual de los parches. Sin embargo, es 4.690 veces más rápida. Por otra parte, obtuvo mejores resultados que dos modelos derivados de la ecuación del plano a pesar que las regiones libres de obstáculos físicamente están contenidas en planos.

La existencia de puntos atípicos en el repositorio FIS-EPN se debe al movimiento natural del usuario durante la operación del dispositivo ETA y a las características propias del entorno. En este repositorio, estos puntos restringen el uso del modelo basado en la ecuación del plano ya que en la clasificación binaria alcanzó exactitud balanceada igual a 68.750% y coeficiente IoU igual a 12.564. Por otra parte, la CNN para nubes de puntos fue capaz de superar esta limitación ya que obtuvo exactitud balanceada igual a

92.382% y coeficiente IoU igual a 95.497%.

Cuando se implementó en un mini-computador Jetson TX2, el modelo propuesto alcanzó un tiempo de procesamiento de 38.243 ms. De esta manera, el modelo es adecuado para utilizarse en dispositivos ETA ya que permitiría procesar 18 escenas antes que un usuario complete un paso.

Cuando el modelo se integró en el dispositivo ETA FIS-EPN en lugar del módulo de detección del obstáculo más cercano, el dispositivo disminuyó su tiempo de procesamiento de 300.000 a 82.549 ms.

Con base en los resultados obtenidos en la evaluación del modelo propuesto en los diferentes repositorios puede afirmarse que el objetivo de la investigación se cumplió y la hipótesis fue verificada porque en todos los casos se desarrolló la capacidad de encontrar regiones libres de obstáculos y en el caso de los repositorios Labelmefacade y NUY-v2 se mejoró el tiempo de procesamiento respecto del estado del arte.

6.2 Trabajos futuros

A pesar que se cumplió el objetivo de este trabajo, las brechas de investigación encontradas en el Capítulo 2 permanecen abiertas porque la capacidad del modelo producido para identificar regiones libres de obstáculos sigue siendo inferior a la de los seres humanos. En ese sentido, se recomienda continuar investigando para mejorar la exactitud y velocidad de estos modelos.

Los obstáculos que se encuentran por debajo de la línea de piso como huecos, desniveles y gradas se denominan obstáculos negativos. Estos no son frecuentes en FIS-EPN por lo que la red entrenada con este repositorio tiene dificultad para reconocerlos. Nótese que estos obstáculos tampoco son frecuentes en los otros 2 repositorios utilizados a pesar de representar peligro para peatones y vehículos. En este sentido, se recomienda ampliar el repositorio FIS-EPN teniendo en consideración la importancia de reconocer apropiadamente obstáculos negativos.

Una alternativa para mejorar el desempeño de la red es disminuir la cantidad de elementos atípicos presentes en las nubes de puntos. En este sentido, se recomienda explorar diferentes alternativas para recuperar la información espacial del entorno además de la cámara estereoscópica. Tales tecnologías deberían ser resistentes al ruido introducido por el movimiento natural del usuario y las características propias del entorno.

Finalmente, en este trabajo no fue tomada en cuenta la consistencia entre las predicciones de escenas adyacentes de una secuencia. Es decir, una región libre de obstáculos no debería aparecer o desaparecer instantáneamente. En este sentido, para futuras investigaciones se recomienda estudiar este factor qué podría resultar útil para completar información de profundidad que no pueda ser recuperada por el sensor.

Bibliografía

[1] B. Planche y E. Andres, *Hands-On Computer Vision with TensorFlow 2*. Packt Publishing Ltd, 2019.

[2] L. Edmonds. (2020) Tesla is sued by family of man, 44, who was run over and killed by a car on autopilot after driver fell asleep behind the wheel. [En línea]. Disponible en: https://www.dailymail.co.uk/news/article-8274449/Tesla-sued-family-man-44-run-killed-car-using-Autopilot.html

[3] Constitución de la República del Ecuador, "Artículo 47," *Registro Oficial No. 449 del 20 de octubre de 2008*, 2008.

[4] K. Yang, K. Wang, L. M. Bergasa, E. Romera, W. Hu, D. Sun, J. Sun, R. Cheng, T. Chen, y E. López, "Unifying terrain awareness for the visually impaired through real-time semantic segmentation," *Sensors*, vol. 18, no. 5, p. 1506, 2018.

[5] Y. Delahoz y M. Labrador, "A deep-learning-based floor detection system for the visually impaired," en *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*. IEEE, 2017, pp. 883–888.

[6] C.-A. Brust, S. Sickert, M. Simon, E. Rodner, y J. Denzler, "Convolutional patch networks with spatial prior for road detection and urban scene understanding," en *International Conference on Computer Vision Theory and Applications*, vol. 2. SCITEPRESS, 2015, pp. 510–517.

[7] C. C. T. Mendes, V. Frémont, y D. F. Wolf, "Exploiting fully convolutional neural networks for fast road detection," en *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3174–3179.

[8] M. Liu, Z. Hou, Z. Sun, N. Yin, H. Yang, Y. Wang, Z. Chu, y H. Kong, "Campus guide: A lidar-based mobile robot," en *2019 European Conference on Mobile Robots (ECMR)*. IEEE, 2019, pp. 1–6.

[9] G.-J. Kim, D. Kim, y B. Kang, "Design and implementation of ground plane detection using predefined datasets and space geometric analysis," *JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE*, vol. 18, no. 4, pp. 500–508, 2018.

[10] L.-A. Raymond, M. Piccini, M. Subramanian, P. Orlov, y A. Faisal, "Natural gaze data driven wheelchair," *BioRxiv*, p. 252684, 2018.

[11] D. Wolf, J. Prankl, y M. Vincze, "Enhancing semantic segmentation for robotics: The power of 3-d entangled forests," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 49–56, 2016.

[12] M. Thøgersen, S. Escalera, J. Gonzàlez, y T. B. Moeslund, "Segmentation of rgb-d indoor scenes by stacking random forests and conditional random fields," *Pattern Recognition Letters*, vol. 80, pp. 208 – 215, 2016. [En línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S016786551630157X

[13] F. Husain, H. Schulz, B. Dellen, C. Torras, y S. Behnke, "Combining semantic and geometric features for object class segmentation of indoor scenes," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 49–55, 2017.

[14] J. Huang y S. You, "Point cloud labeling using 3d convolutional neural network," en *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 2670–2675.

[15] X. Wang, J. Calderon, N. Khoshavi, y L. G. Jaimes, "Floor detection in outdoor environments through the use of deep learning approach," no publicado.

[16] K. Yang, K. Wang, R. Cheng, y X. Zhu, "A new approach of point cloud processing and scene segmentation for guiding the visually impaired," en *2015 IET International Conference on Biomedical Image and Signal Processing (ICBISP 2015)*. IET, 2015.

[17] Stanford University. (2017) Lecture 11: Detection and segmentation. [En línea]. Disponible en: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

[18] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, Reporte Técnico TR/SE-0401, 2004.

[19] J. de Jesús Osuna-Coutiño y J. Martinez-Carranza, "A binary descriptor invariant to rotation and robust to noise (birrn) for floor recognition," en *Mexican Conference on Pattern Recognition*. Springer, 2019, pp. 271–281.

[20] B. Zhu, G. Xiong, H. Di, K. Ji, X. Zhang, y J. Gong, "A novel method of traversable area extraction fused with lidar odometry in off-road environment," en *2019 IEEE International Conference of Vehicular Electronics and Safety (ICVES)*. IEEE, 2019, pp. 1–6.

[21] C. Zatout, S. Larabi, I. Mendili, y S. Ablam Edoh Barnabe, "Ego-semantic labeling of scene from depth image for visually impaired and blind people," en *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[22] B. Li y X. Qian, "Real-time detection of travelable path based on image point cloud," en *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 4608–4612.

[23] Y. Zhu, D. Han, B. Xue, J. Jiao, Z. Zou, M. Liu, y R. Fan, "Road curb detection using a novel tensor voting algorithm," *arXiv preprint arXiv:1911.12937*, 2019.

[24] S. Lafuente-Arroyo, S. Maldonado-Bascón, H. Gómez-Moreno, y C. Alén-Cordero, "Segmentation in corridor environments: Combining floor and ceiling detection," en *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2019, pp. 485–496.

[25] J. Bai, Z. Liu, Y. Lin, Y. Li, S. Lian, y D. Liu, "Wearable travel aid for environment perception and navigation of visually impaired people," *Electronics*, vol. 8, no. 6, p. 697, 2019.

[26] S. Caraiman, O. Zvoristeanu, A. Burlacu, y P. Herghelegiu, "Stereo vision based sensory substitution for the visually impaired," *Sensors*, vol. 19, no. 12, p. 2771, 2019.

[27] K. Yang, L. M. Bergasa, E. Romera, y K. Wang, "Robustifying semantic cognition of traversability across wearable rgb-depth cameras," *Applied optics*, vol. 58, no. 12, pp. 3141–3155, 2019.

[28] K. Yang, K. Wang, S. Lin, J. Bai, L. M. Bergasa, y R. Arroyo, "Long-range traversability awareness and low-lying obstacle negotiation with realsense for the visually impaired," en *Proceedings of the 2018 International Conference on Information Science and System*, 2018, pp. 137–141.

[29] K. Yang, L. M. Bergasa, E. Romera, J. Wang, K. Wang, y E. López, "Perception framework of water hazards beyond traversability for real-world navigation assistance systems," en *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 186–191.

[30] S. P. Langenkamp y M. R. Abid, "Phone based video processing aimed at outdoor guidance for the visually impaired," en *Proceedings of the 2nd International Conference on Information System and Data Mining*, 2018, pp. 141–145.

[31] K. Yang, L. M. Bergasa, E. Romera, D. Sun, K. Wang, y R. Barea, "Semantic perception of curbs beyond traversability for real-world navigation assistance systems," en *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 2018, pp. 1–7.

[32] N. Druml, T. Pietsch, M. Dielacher, C. Steger, M. Baumgart, C. Consani, T. Herndl, y G. Holweg, "Virtual white cane featuring time-of-flight 3d imaging supporting visually impaired users," en *2018 21st Euromicro Conference on Digital System Design (DSD)*. IEEE, 2018, pp. 450–457.

[33] M. D. Solbach y J. K. Tsotsos, "Vision-based fallen person detection for the elderly," en *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 1433–1442.

[34] D. Guastella, L. Cantelli, C. D. Melita, y G. Muscato, "A global path planning strategy for a ugv from aerial elevation maps for disaster response." en *ICAART (1)*, 2017, pp. 335–342.

[35] Y. Delahoz y M. A. Labrador, "A real-time smartphone-based floor detection system for the visually impaired," en *2017 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. IEEE, 2017, pp. 27–32.

[36] K. Nagao y Y. Miyakawa, "Building scale vr: Automatically creating indoor 3d maps and its application to simulation of disaster situations," en *Proceedings of Future Technologies Conference (FTC)*, 2017.

[37] K. Yang, K. Wang, R. Cheng, W. Hu, X. Huang, y J. Bai, "Detecting traversable area and water hazards for the visually impaired with a prgb-d sensor," *Sensors*, vol. 17, no. 8, p. 1890, 2017.

[38] I. Van Crombrugge, I. B. Azza, R. Penne, G. Van Barel, y S. Vanlanduit, "Fast ground detection for range cameras on road surfaces using a three-step segmentation," en *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2017, pp. 479–490.

[39] L. Tang, X. Ding, H. Yin, Y. Wang, y R. Xiong, "From one to many: Unsupervised traversable area segmentation in off-road environment," en *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2017, pp. 787–792.

[40] J. Guerrero, A. Pérez-Yus, D. Gutiérrez-Gómez, A. Rituerto, y G. López-Nicolás, "Human navigation assistance with a rgb-d sensor," en *ACTAS V Congreso Internacional de Turismo para Todos: VI Congreso Internacional de Diseno, Redes de Investigacion y Tecnologia para todos DRT4ALL*, 2015, pp. 285–312.

[41] S. Gu, Y. Zhang, J. Yang, y H. Kong, "Lidar-based urban road detection by histograms of normalized inverse depths and line scanning," en *2017 European Conference on Mobile Robots (ECMR)*. IEEE, 2017, pp. 1–6.

[42] L. Chen, J. Yang, y H. Kong, "Lidar-histogram for fast road and obstacle detection," en *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1343–1348.

[43] C. Guindel, D. Martín, y J. M. Armingol, "Modeling traffic scenes for intelligent vehicles using cnn-based detection and orientation estimation," en *Iberian Robotics conference*. Springer, 2017, pp. 487–498.

[44] K. Saleh, R. A. Zeineldin, M. Hossny, S. Nahavandi, y N. A. El-Fishawy, "Navigational path detection for the visually impaired using fully convolutional networks," en *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 1399–1404.

[45] A. Kumar, H. Ren, y P. Ben-Tzvi, "Obstacle identification for vision assisted control architecture of a hybrid mechanism mobile robot," en *ASME 2017 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection, 2017.

[46] W. Song, M. Fu, Y. Yang, M. Wang, X. Wang, y A. Kornhauser, "Real-time lane detection and forward collision warning system based on stereo vision," en *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 493–498.

[47] G. Reina, A. Milella, R. Rouveure, M. Nielsen, R. Worst, y M. R. Blas, "Ambient awareness for agricultural robotic vehicles," *biosystems engineering*, vol. 146, pp. 114–132, 2016.

[48] K. Al-Muteb, M. Faisal, M. Emaduddin, M. Arafah, M. Alsulaiman, M. Mekhtiche, R. Hedjar, H. Mathkoor, M. Algabri, y M. A. Bencherif, "An autonomous stereovision-based navigation system (asns) for mobile robots," *Intelligent Service Robotics*, vol. 9, no. 3, pp. 187–205, 2016.

[49] K. Yang, K. Wang, W. Hu, y J. Bai, "Expanding the detection of traversable area with realsense for the visually impaired," *Sensors*, vol. 16, no. 11, p. 1954, 2016.

[50] Y. Ji, A. Yamashita, y H. Asama, "Indoor positioning system based on distributed camera sensor networks for mobile robot," en *International Conference on Intelligent Autonomous Systems*. Springer, 2016, pp. 1089–1101.

[51] J. Frikha, D. Sellami, y I. K. Kallel, "Indoor/outdoor navigation system based on possibilistic traversable area segmentation for visually impaired people," *ELCVIA: electronic letters on computer vision and image analysis*, vol. 15, no. 1, pp. 60–76, 2016.

[52] H.-H. Pham, T.-L. Le, y N. Vuillerme, "Real-time obstacle detection system in indoor environment for the visually impaired using microsoft kinect sensor," *Journal of Sensors*, vol. 2016, 2016.

[53] Y. Xia, S. Pengbo, L. Jie, y Z. Chunxia, "Traversable detection in semi-structured environment by using 2d sequential laser data for a small mobile robot," en *2016 Chinese Control and Decision Conference (CCDC)*. IEEE, 2016, pp. 5308–5313.

[54] S. Bhowmick, A. Pant, J. Mukherjee, y A. K. Deb, "A novel floor segmentation algorithm for mobile robot navigation," en *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*. IEEE, 2015, pp. 1–4.

[55] C. Chen, Y. He, F. Gu, C. Bu, y J. Han, "A real-time relative probabilistic mapping algorithm for high-speed off-road autonomous driving," en *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6252–6258.

[56] J. Wang, M. Garratt, y S. Anavatti, "Dominant plane detection using a rgb-d camera for autonomous navigation," en *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*. IEEE, 2015, pp. 456–460.

[57] L. Neumann, B. Vanholme, M. Gressmann, A. Bachmann, L. Kählke, y F. Schüle, "Free space detection: A corner stone of automated driving," en *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 2015, pp. 1280–1285.

[58] C. Fan, "Indoor scene 3d modeling with single image," Tesis de Maestría, University of Missouri–Columbia, 2015.

[59] H. Macher, T. Landes, y P. Grussenmeyer, "Point clouds segmentation as base for as-built bim creation," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 2, no. 5, p. 191, 2015.

[60] A. Mecocci, F. Micheli, y C. Zoppetti, "Range image processing for real time hospital-room monitoring," en *International Symposium on Visual Computing*. Springer, 2015, pp. 81–92.

[61] S. K. Reddy y P. K. Pal, "Segmentation of point cloud from a 3d lidar using range difference between neighbouring beams," en *Proceedings of the 2015 Conference on Advances in Robotics*, 2015, pp. 1–6.

[62] X. Yuan, X. Tang, y C. Zhao, "Road detection in image by fusion laser points based on fuzzy svm for a small ground mobile robot," *Journal of Intelligent & Fuzzy Systems*, vol. 29, no. 6, pp. 2677–2688, 2015.

[63] Stereolabs Inc. (2020) Depth sensing overview. [En línea]. Disponible en: https://www.stereolabs.com/docs/depth-sensing/

[64] ——. (2020) Coordinate frames. [En línea]. Disponible en: https://www.stereolabs.com/docs/positional-tracking/coordinate-frames/

[65] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[66] R. Labayrade, D. Aubert, y J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation," en *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 2. IEEE, 2002, pp. 646–651.

[67] J. Chaki y N. Dey, *A Beginner's Guide to Image Preprocessing Techniques*. CRC Press, 2018.

[68] M. C. Potter, B. Wyble, C. E. Hagmann, y E. S. McCourt, "Detecting meaning in rsvp at 13 ms per picture," *Attention, Perception, & Psychophysics*, vol. 76, no. 2, pp. 270–279, 2014.

[69] S. S. Haykin *et al.*, "Neural networks and learning machines," 2009.

[70] S. Khan, H. Rahmani, S. A. A. Shah, M. Bennamoun, G. Medioni, y S. Dickinson, *A Guide to Convolutional Neural Networks for Computer Vision*. Morgan & Claypool Publishers, 2018.

[71] Stanford University. (2020) Convolutional neural network. [En línea]. Disponible en: http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

[72] ——. (2020) Convolutional neural networks (cnns / convnets). [En línea]. Disponible en: https://cs231n.github.io/convolutional-networks/

[73] ——. (2020) Multi-layer neural network. [En línea]. Disponible en: http://deeplearning.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/

[74] Sicara. (2018) About convolutional layer and convolution kernel. [En línea]. Disponible en: https://www.sicara.ai/blog/2019-10-31-convolutional-layer-convolution-kernel

[75] S. Öztürk, U. Özkaya, B. Akdemir, y L. Seyfi, "Convolution kernel size effect on convolutional neural network in histopathological image processing applications," en *2018 International Symposium on Fundamentals of Electrical Engineering (ISFEE)*, 2018, pp. 1–5.

[76] M. A. Fischler y R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[77] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, y S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," en *2017 international conference on 3D vision (3DV)*. IEEE, 2017, pp. 537–547.

[78] P. K. Nathan Silberman, Derek Hoiem y R. Fergus, "Indoor segmentation and support inference from rgbd images," en *ECCV*, 2012.

[79] D. Wolf, J. Prankl, y M. Vincze, "Enhancing semantic segmentation for robotics: The power of 3-d entangled forests," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 49–56, 2016.

[80] K. Wada, "labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme, 2016.

[81] University of Cambridge, Department of Engineering. (2007) Labeling instructions and notes. http://mi.eng.cam.ac.uk/projects/cvgroup/instructions/instructions.html.

[82] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," https://www.tensorflow.org/, 2015, Software available from tensorflow.org.

[83] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[84] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[85] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, y E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[86] ——. (2020) Metrics and scoring: quantifying the quality of predictions. https://scikit-learn.org/stable/modules/model_evaluation.html#dummy-estimators.

[87] Google Inc. (2020) Classification on imbalanced data. https://www.tensorflow.org/tutorials/structured_data/imbalanced_data.

[88] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, y S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," en *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 658–666.

[89] Q.-Y. Zhou, J. Park, y V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[90] H. Schulz, N. Höft, y S. Behnke, "Depth and height aware semantic rgb-d perception with convolutional neural networks," en *Proc. Eur. Conf. Neural Netw.(ESANN)*, 2015, pp. 463–468.

[91] S. J. Singleton, S. E. Keating, S. L. McDowell, B. A. Coolen, y J. C. Wall, "Predicting step time from step length and velocity," *Australian Journal of Physiotherapy*, vol. 38, no. 1, pp. 43 – 46, 1992. [En línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S000495141460550X

[92] A. Armendáriz, J. Lucio-Naranjo, y D. Navas, "Real time auralization module for electronic travel aid devices for people with visual disability," *Latin American Journal of Computing Faculty of Systems Engineering Escuela Politécnica Nacional Quito-Ecuador*, vol. 5, no. 1, pp. 27–36, 2018.

[93] OpenCV. (2020) Image segmentation with watershed algorithm. https://docs.opencv.org/master/d3/db4/tutorial_py_watershed.html.