uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería Electrónica Industrial y Automática
2016-2017

Trabajo Fin de Grado

"Generación del Dataset de imágenes etiquetadas SAUCE"

Sergio Martín Gálvez

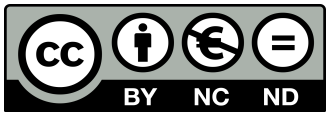
Tutor/es

José María Armingol Moreno

Jorge Beltrán de la Cita

Basam Musleh Lancis

Aula 7.2.J07, 10/10/2017



Agradecimientos

A mis padres y hermana por apoyarme siempre en todo lo que hago. A mis tutores y profesores por prestarme su ayuda siempre que la he necesitado.

Resumen

En el mundo actual cada vez estamos más acostumbrados a la búsqueda de la autonomía, esa que debido a su alta complejidad resulta difícil de encontrar. En los medios urbanos de tráfico lograr esa autonomía resulta, por su gran variedad y extensión, un gran reto para los sistemas inteligentes de los vehículos. Respecto a esto, son muchas las soluciones y algoritmos presentados hasta el momento, con resultados muy dispares, costes o cargas computacionales que en muchas ocasiones se pueden evitar si se tuviera un sitio donde contrastar soluciones.

Este trabajo se encuadra dentro del procesamiento digital y presenta un soporte ante esta gran variedad de sistemas inteligentes utilizados en la búsqueda de la autonomía de los vehículos, desarrolla un sitio de comparación con una base de datos de imágenes etiquetadas basadas en la arquitectura ROS de escenas semánticas en estéreo-visión, y permite que usuarios externos puedan comparar el producto de sus algoritmos con dicha base de datos y contrasten sus resultados con los de otros usuarios en función de diversos parámetros (CPU, procesador, calidad del etiquetado resultante, etc.).

El procedimiento que se ha utilizado para lograr este proyecto con eficacia consta de las siguientes fases: Creación de base de datos que consta de 100 etiquetas realizadas a mano una a una a partir de fotos originales realizadas por el vehículo autónomo iCab en el campus de la UC3M de Leganés, desarrollo del algoritmo de comparación cogiendo de base las OpenCV y lenguaje Python, realización de pruebas de comparación para evaluar la eficiencia del algoritmo y la creación de un sitio web (con base Wordpress) para poner a disposición de todos la posibilidad de comparar el resultado de sus algoritmos con los de nuestra base de datos.

Palabras claves: sistemas inteligentes, procesamiento digital, base de datos, iCab, ROS, UC3M, OpenCV, Python, Wordpress.

Abstract

In the world today we are becoming more accustomed to the search for autonomy, which due to its high complexity is difficult to find. In the urban means of traffic achieving this autonomy is, by its great variety and extension, a great challenge for the intelligent systems of vehicles. Regarding this, there are many solutions and algorithms presented so far, with very different results, costs or computational loads that in many cases can be avoided if there is a place to test solutions.

This work fits within the digital processing and presents a support to this great variety of intelligent systems used in the search of the autonomy of the vehicles, develops a comparison site with a labeled images based in ROS architecture for stereo-vision-based semantic scene, and allows External users can compare the product of their algorithms with that database and contrast their results with those of other users based on various parameters (CPU, processor, quality of the resulting labeling, etc.).

The procedure that has been used to achieve this project effectively consists of the following phases: Creation of database consisting of 100 handmade tags one by one from original photos taken by the iCab standalone vehicle on the campus of the UC3M, development of the comparison algorithm based on the OpenCV and Python language, performing comparison tests to evaluate the efficiency of the algorithm and the creation of a website (based on Wordpress) to make available to all the possibility of Compare the results of your algorithms with those of our database.

Keywords: intelligent systems, digital processing, database, iCab, ROS, UC3M, OpenCV, Python, Wordpress.

Índice

Índice de figuras

Índice de tablas

Capítulo 1:

1. Introducción

En este primer capítulo del trabajo, se va a explicar la finalidad del sitio web a desarrollar, justificándolo con diferentes formas de aplicación. Para ello, se van a marcar una serie de objetivos seguidos de sus fases intermedias para lograrlos. Finalmente, se resumirán las diferentes partes de la memoria.

1.1.Preámbulo

Este trabajo se ha realizado en el Dpto. de Ing. de Sistemas y Automática de la Universidad Carlos III de Madrid. La idea de su desarrollo está asociada a la multitud de algoritmos que se vienen desarrollando para lograr la autonomía de un vehículo en el campus, pero todos ellos sin una base de comparación entre sí. El objetivo es lograr esa base o mecanismo de comparación donde los usuarios puedan contrastar sus resultados.

Para lograrlo, se trabajará con un conjunto de imágenes obtenidas de la cámara embarcada en el vehículo autónomo iCab, que no hayan sido sometidas a otras tecnologías que puedan afectar al contraste real o luminosidad. Esto debe ser así ya que es lo que al final se va a usar para lograr su autonomía.

La finalidad del trabajo es usar esas imágenes como base, para posteriormente etiquetarlas con los colores primarios de la luz RGB (Red, Green, Blue) como se muestra en la ilustración 1, y usar dichas etiquetas como punto de comparación a los resultados de los algoritmos de etiquetado de los usuarios.

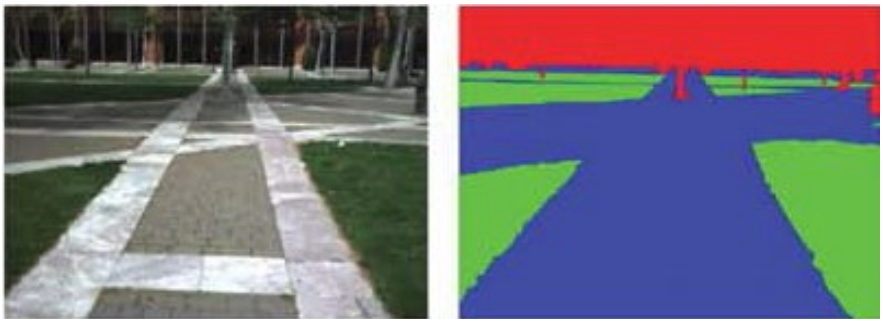


Ilustración 1. Etiquetado de imagen

1.2.Objetivo

Queda claro que la finalidad del trabajo es la creación de una web donde se puedan ver y comparar el resultado de algoritmos creados por los usuarios, todos ellos con la misma base RGB, para poder obtener el mejor algoritmo basado en técnicas de Deep-learningque consiga la autonomía del vehículo inteligente. Este es el objeto principal, pero para llegar a él, debemos obtener otros, que son los siguientes:

- Creación de una **base de datos** consistente basada en el etiquetado de la arquitectura ROS, que pueda servir de base suficiente para la confrontación con otras variedades de bancos de datos.

- Desarrollo de un algoritmo o **script** que aborde dicha base de datos y sea lo suficientemente eficiente para conseguir contrastarla con otras y a la vez nos de los resultados basados en el nivel de acierto por cada pixel de cada etiqueta que se compara.

- A partir del resultado del script, desarrollar una **web con base en Wordpress** que contenga diferentes submenús y que en cada uno de ellos aporte la información más relevante del algoritmo que se usa como base, la información del etiquetado del usuario y la comparación con el resto de usuarios que han usado el script de la web para su beneficio.

Para poder lograr los objeticos marcados, se han realizado una serie de pautas al inicio del proceso, dichas pautas son esenciales para el trabajo y tendrán que cumplirse.

Lo primero es obtener la **documentación** que se va a utilizar y desarrollar una en base a lo que queremos obtener. Crear una estrategia que nos lleve a la solución y abrir una puerta para otras aplicaciones parecidas. El contexto del trabajo nace del estudio de diferentes medios por los cuales se puede obtener el etiquetado de la base de datos, desarrollar el algoritmo y la posterior web.

En la primera fase de proyecto, para el desarrollo de la base de datos se utilizará el programa **GIMP**, el cual permitirá un etiquetado preciso de las imágenes.

Para la programación del algoritmo, se utilizarán las **librerías OpenCV**. El lenguaje para programar el script será Python, el cual se puede complementar con la librería mencionada anteriormente.

En cuanto a la programación de la **web en Wordpress**, se utilizarán los lenguajes PHP, java script y HTML. Además, para el funcionamiento interno de la web se instalarán diferentes plugins, uno de ellos vital para la muestra de la información de la tabla.

Estas pautas quedan resumidas en la siguiente ilustración 2.

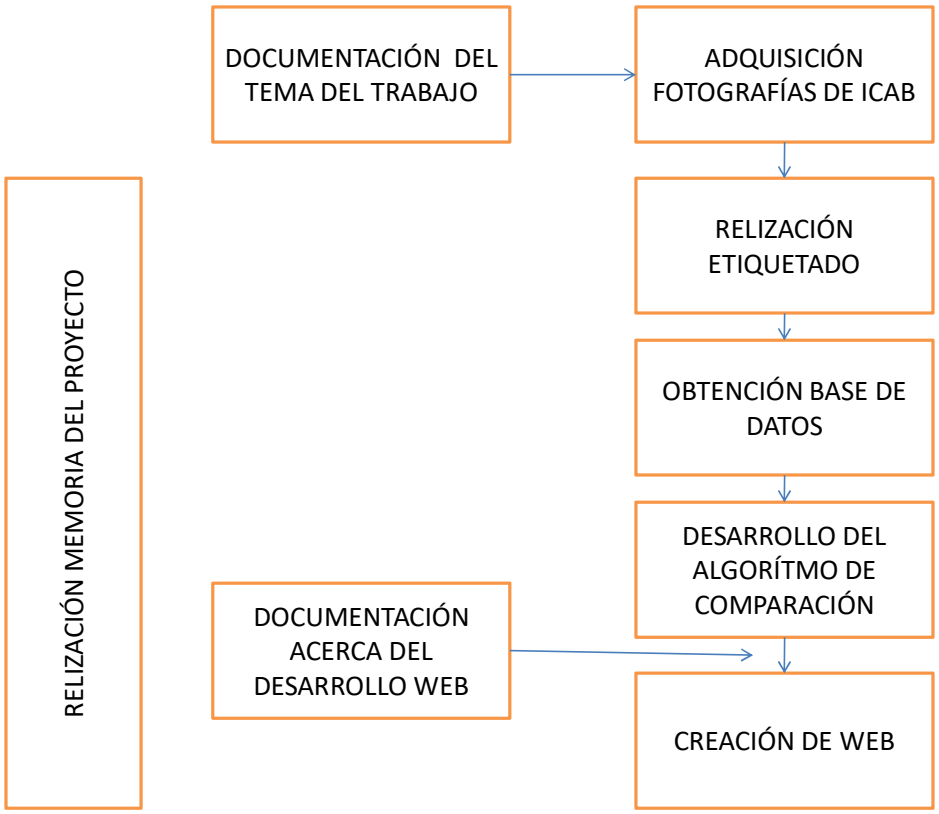


Ilustración 2. Fases de desarrollo

La parte o fase más crítica del proyecto es la del **etiquetado de las imágenes**, debido a que dichas etiquetas se usarán como base de datos núcleo y punto de comparación, además, es la parte más monótona y tediosa ya que en función de cómo queramos sea la calidad de nuestro algoritmo, habrá que incluir un mayor o menor número de imágenes.

A la vez que se desarrollan las pautas anteriores, se elaborará la **memoria** para la documentación del proyecto.

1.3.Organización del documento

En esta parte del proyecto se analizarán los principales aspectos de los que tratan los distintos capítulos que componen la memoria.

- **Capítulo 1:** Se introduce el trabajo, con una motivación, objetivos y estructura del mismo.

- **Capítulo 2:** Se realiza un estudio del estado del arte prestando interés a los vehículos autónomos en general, los sistemas de percepción, la evolución del deep learning y los algoritmos más punteros en el ámbito de la detección, clasificación y segmentación.

- **Capítulo 3:** Se introducirá el propósito y las características generales del trabajo.

- **Capítulo 4:** Se explicará cómo se ha llevado a cabo el desarrollo de la base de datos, el etiquetado y se explicarán las métricas más comunes para la evaluación de la segmentación semántica.

- **Capítulo 5:** Explicación de la creación del sitio web y cómo se ha implementado el algoritmo desarrollado en él, además de cómo se mostrará al usuario, su funcionamiento, pantallas, etc.

- **Capítulo 6:** Se comentarán las conclusiones que se han obtenido y si se han conseguido los objetivos planteados. Además, se plantearán diversos trabajos futuros.

- **Capítulo 7:** Estudio del entorno socio-económico.

1.4.Norma aplicable

Debido a que en este proyecto se va a llevar a cabo una accesibilidad y una publicación en web, tendrá que seguir la última normativa vigente en relación a este tema.

La publicación de contenidos en Internet está regulada por una serie de leyes y normativas. Una de las más conocidas es la Ley de Protección de datos personales, que se extiende a toda la información de una institución [1].

Normativa de accesibilidad

En este aspecto, son aplicables las normativas siguientes.

Ley Orgánica 15/1999: protección de datos de carácter personal

La Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de carácter personal tiene como objeto garantizar y proteger, en lo que respecta al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor y su intimidad personal y familiar. Así pues, la Oficina Web tiene la obligación de velar por estos derechos en cuanto a la gestión y publicación de datos, como pueden ser los derechos de imagen.

Ley 51/2003 LIONDAU

La Ley 51/2003, de 2 de diciembre de 2003, de Igualdad de Oportunidades, No Discriminación y Accesibilidad Universal de las personas con discapacidad (LIONDAU) establece: 1 º) Que a principios de 2006 el Gobierno deberá haber establecido los criterios básicos de accesibilidad para las Tecnologías de la Sociedad de la Información, 2 º) en el 2010 todos los nuevos productos y servicios de la Sociedad de la Información deberán ser accesibles y 3 º) en 2014 todos los productos y servicios de la Sociedad de la Información deberán ser accesibles.

Real Decreto 1494/2007: accesibilidad de los sitios web públicos

El REAL DECRETO 1494/2007, de 12 de noviembre, por el que se aprueba el Reglamento sobre las condiciones básicas para el acceso de las personas con discapacidad a las tecnologías, productos y servicios relacionados con la sociedad de la información y los medios de comunicación social, en el artículo 5 del capítulo III Criterios y condiciones básicas de accesibilidad y no discriminación en materia de sociedad de la información, trata de los criterios de accesibilidad aplicables a las páginas de Internet de las administraciones públicas o con financiación pública.

Ley 49/2007, de 26 de diciembre

La Ley 49/2007 establece el régimen de infracciones y sanciones en materia de igualdad de oportunidades, no discriminación y accesibilidad universal de las personas con discapacidad. Establece tres niveles de infracción: leve, grave o muy graves. Las sanción serán multas entre los 301 euros y el millón de euros.

Ley 7 / 2010, de 31 de marzo

La Ley 7 / 2010, General de la Comunicación Audiovisual, establece el derecho a una comunicación audiovisual transparente y los derechos de las personas con discapacidad.

LEY 56/2007, de 28 de diciembre, de Medidas de Impulso de la Sociedad de la Información.

Se dictan medidas de obligado cumplimiento para la eliminación de las barreras en el uso de las Tecnologías de la Información para la Administración Pública y entidades financiadas por la misma.

La presente Ley se enmarca en el conjunto de medidas que constituyen el Plan 2006-2010 para el desarrollo de la Sociedad de la Información y de convergencia con Europa y entre Comunidades Autónomas y Ciudades Autónomas, Plan Avanza, aprobado por el Gobierno en noviembre de 2005.

LEY 27/2007, de 23 de octubre, por la que se reconocen las lenguas de signos españolas y se regulan los medios de apoyo a la comunicación oral de las personas sordas, con discapacidad auditiva y sordo ciegas.

Se regula el reconocimiento y el uso de la Lengua de Signos Española en el ámbito público.

La presente Ley tiene por objeto reconocer y regular la Lengua de Signos Española como lengua de las personas sordas, con discapacidad auditiva y sordociegas en España que libremente decidan utilizarla, sin perjuicio del reconocimiento de la lengua de signos catalana en su ámbito de uso lingüístico, así como la regulación de los medios de apoyo a la comunicación oral [3].

LEY 11/2007, de 22 de junio, de acceso electrónico de los ciudadanos a los Servicios Públicos.

Se reconoce el derecho de los ciudadanos a relacionarse con las Administraciones Públicas por medios electrónicos.

La presente Ley reconoce el derecho de los ciudadanos a relacionarse con las Administraciones Públicas por medios electrónicos y regula los aspectos básicos de la utilización de las tecnologías de la información en la actividad administrativa, en las relaciones entre las Administraciones Públicas, así como en las relaciones de los ciudadanos con las mismas con la finalidad de garantizar sus derechos, un tratamiento común ante ellas y la validez y eficacia de la actividad administrativa en condiciones de seguridad jurídica [4].

REAL DECRETO 366/2007, de 16 de marzo, de accesibilidad y no discriminación de las personas con discapacidad en sus relaciones con la Administración General del Estado.

Se regularizan las condiciones de Accesibilidad y No discriminación de los ciudadanos en sus relaciones con la Administración General del Estado [5].

REAL DECRETO 1494/2007, de 12 de noviembre, por el que se aprueba el Reglamento sobre las condiciones básicas para el acceso de las personas con discapacidad a la sociedad de la información..

Se normalizan las condiciones de Accesibilidad de la Sociedad de la Información.

El presente real decreto se inspira en los principios establecidos en la Ley 51/2003, de 2 de diciembre, fundamentalmente, accesibilidad universal y diseño para todos.

Criterios y condiciones básicas de accesibilidad y no discriminación en materia de sociedad de la información [6].

Capitulo 2:

2. Estado del arte. Percepción en vehículos autónomos.

2.1.Vehículos autónomos

La definición de vehículo autónomo procede, de manera informal, de la palabra auto conducido o sin conductor, viene a ser un automóvil autónomo que desempeña por si solo la función humana en un vehículo. Como vehículo autónomo es capaz de percibir el medio que le rodea y navegar en consecuencia, esto hace que la función humana sea mínima, solo elegir destino y despreocuparse de lo demás [7].

El **origen de los coches autónomos** es mucho más remoto de lo que se suele pensar. Desde los años 40 del siglo pasado, se han estado haciendo pruebas de guiado de vehículos, embebiendo dentro del asfalto materiales que podían ser detectados y seguidos. De ahí, se pasó a la detección de obstáculos vía radar en los años 80 y en la actualidad la tecnología es un complejo conjunto de sistemas de percepción que incluyen el reconocimiento de movimiento por cámaras y los sistemas de detección láser [8].

Hoy en día, uno de los vehículos más avanzados y referente en este tipo de mercado es el **coche de Google.** El gigante informático pretende que en el año 2020 sea corriente ver vehículos completamente autónomos circulando por las carreteras. Está programado actualmente para alcanzar una **velocidad máxima de 40 kilómetros por hora;** acumula más de un millón de millas (1'61 millones de kilómetros) de experiencia de conducción autónoma; y aunque ya lleva varias experiencias piloto, faltan muchos detalles por completar.

Ilustración 3. Coche autónomo de Google

Los primeros prototipos carecían de **volante, pedales de freno y acelerador,** pero han tenido que ser incluidos por razones de seguridad. Parece lógico, ya que, si bien se produjeron durante la etapa de desarrollo, hay que decir que los coches de Google se han visto involucrados en 11 accidentes menores en los que el coche no fue responsable. Esto podrá ser evitado en un futuro, cuando la mayoría de la flota de coches posea un sistema de comunicación general [9].

Todos estos avances solo pertenecen a una transición, ya que muchos son los expertos que creen que todavía queda mucho por hacer y que para que sea una realidad sólida aún deben de pasar algunas décadas. Esta transición, a parte del vehículo de Google ya comentado, tiene como protagonista otros fabricantes como son Tesla o Cadillac.

En el núcleo del asunto, está el tema del propio funcionamiento de este tipo de automóviles, ya no fundamentan su funcionamiento en el motor o la carrocería, sino que el software del vehículo toma el papel principal. Para ello se encargan de su desarrollo las ingenierías de los sistemas de percepción donde la inteligencia artificial toma el control.

2.2.Relevancia de los sistemas de percepción

Cuando nos imaginamos el sistema de control de un vehículo autónomo caemos fácilmente en pensar en los algoritmos típicos basados en programación básica, del estilo “if (…) then…else…”, por ejemplo “Si la cámara detecta un objeto delante, reducir la velocidad o incluso frenar”. Ese tipo de mentalidad no es la más idónea ya que las situaciones que nos podemos encontrar en este ámbito son muy numerosas y por tanto hay que utilizar un sistema capaz de generalizar.

Aquí es donde entra la visión por computador o visión artificial, que es una de las ramas de la Inteligencia Artificial que más ha crecido en los últimos años. Esta disciplina se encargar de estudiar el procesamiento, análisis e interpretación de las imágenes de nuestro entorno de forma automática. La visión por computador trata de producir el mismo efecto que nuestros ojos y cerebro para entender el mundo en las computadoras, y que actúen según convenga en cada situación. Este tipo de sistemas se puede conseguir utilizando y mezclando diferentes campos como la geometría, física,

estadística, etc. En cuanto a la adquisición de datos, se pueden obtener desde secuencias de imágenes proporcionadas por cámaras, láseres, etc. [14].

En la siguiente ilustración se muestra como se relacionan los distintos ámbitos ya comentados, vinculados con la visión por computador [15].

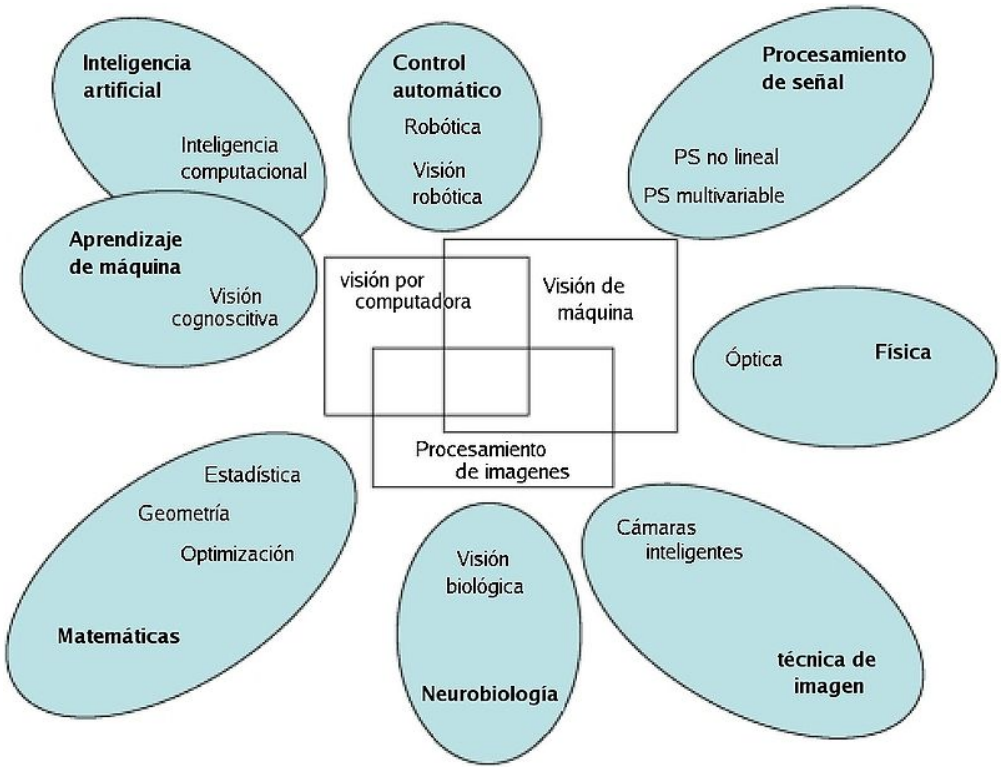


Ilustración 4. Relaciones entre la visión por computador y otros medios

Para los vehículos autónomos, la detección de objetos es uno de los temas de mayor importancia. Es algo que a priori parece sencillo desde el punto de vista humano, pero para una máquina supone un gran reto. En general, para un computador una imagen es una gran caja tridimensional llena de números, que componen los pixeles, cada uno de ellos representados por tres valores: rojo, verde y azul. Por lo tanto, cuando una máquina intenta detectar un objeto en una fotografía lo que hace es buscar patrones que correspondan a ese objeto [10].

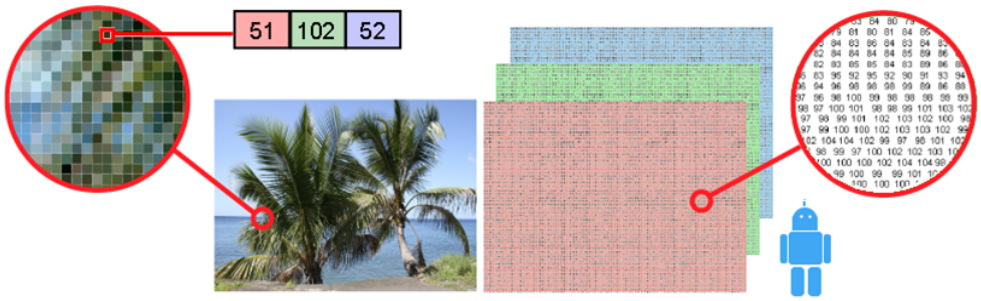


Ilustración 5. Detección de objeto en imagen

En lugar de desarrollar innumerables normas para reconocer estos objetos, es mucho más práctico implementar un algoritmo general de **deep learning (aprendizaje automático)** al que se "entrena" con distintas imágenes que ejemplifican todas las situaciones posibles.

Cada una de las imágenes se asociará con el tipo de vehículo/obstáculo que contiene. En una etapa inicial, el algoritmo intentará adivinar que vehículo hay en cada imagen y al principio, el resultado ofrecido será erróneo. Como "la solución" es conocida, es decir, que vehículo hay en realidad en cada imagen, el algoritmo modificará y adaptará parámetros internos, para pasar a intentarlo de nuevo. El proceso continuará reduciendo iterativamente la cantidad de fallos. De esta manera, cuando en fases posteriores se le presenten nuevas imágenes, podrá clasificarlas correctamente. Podremos entonces afirmar que el algoritmo habrá aprendido [36].

Ese mismo enfoque se puede utilizar para la toma y evaluación de decisiones. En vez de proporcionar una lista de normas con las que evaluar la acción a tomar para cada situación, se entrenará un algoritmo con situaciones de tráfico en las que se especificará la acción correcta a tomar. Igual que antes, el algoritmo intentará adivinar la acción correcta y modificará los parámetros internos en función de si se equivoca o acierta.

Son muchos los algoritmos que están cobrando importancia en el desarrollo del deep learning, entre ellos los más influyentes son los explicados en el siguiente apartado.

2.3.Algoritmos más relevantes y evolución

Para el desarrollo y el avance de los últimos estudios y tecnologías, entre ellas las mencionadas en este proyecto como la conducción autónoma, hace falta un gran desarrollo de algoritmos y sobretodo, de bases de datos que los fundamenten. En los últimos años se han desarrollado bases de datos para estéreo y reconstrucción 3D, flujo óptico, segmentación y reconocimiento de objetos, el Tracking o imágenes aéreas. En el ámbito que nos interesa toman fuerza los algoritmos de detección de objetos y segmentación de imágenes.

DETECCIÓN DE OBJETOS

Concretamente, en la detección de objetos 2D, el algoritmo KITTI Greiger et al. (212b) es posiblemente el punto de referencia. Similar y por su gran popularidad para la detección de personas tenemos la base de datos Caltech-USA Dollár et al. (2012), el cual abordó las deficiencias que surgían de utilizar conjuntos de datos múltiples y protocolos de evaluación muy variados, que dificultaba las comparaciones directas. En su artículo, Dollár reúne un conjunto grande de datos, bien anotado y realista de detección de peatones y estudia las estadísticas del tamaño, la posición y los patrones de oclusión de los mismos en las escenas urbanas, propone una metodología de evaluación por marco que permite realizar comparaciones informativas y de sondeo, incluyendo la medición del desempeño en relación con la escala y la oclusión, y evalúa el desempeño de dieciséis estados pre-entrenados de los detectores de última generación a través de seis conjuntos de datos. Su estudio les permitió evaluar el estado de la técnica y proporcionar un marco para medir los esfuerzos futuros. Sus experimentos además mostraron que a pesar del progreso significativo, el rendimiento todavía tiene mucho margen de mejora. En particular, la detección es decepcionante en resoluciones bajas ya que la visión de los peatones es parcialmente oculta.

En este apartado (detección de objetos en 2D) nos centraremos en el primero (Greiger), que nos permite comparar detección de objetos y personas en el mismo sistema. En la tabla 1 se muestran los estados actuales de bases de datos usadas para el desarrollo del KITTI, orientados a la detección de objetos, personas y ciclistas de imágenes.

El fundamento de este algoritmo se basa en la evaluación para tres niveles de dificultad usando la intersección de PASCAL VOC (IOU) Everingham et al., (2010). Los ejemplos usados se clasifican en sencillos, tienen una altura de 40 pixeles y son totalmente visibles, moderados (nivel intermedio de dificultad para la detección por parte del algoritmo), de 25 pixeles incluyendo oclusión parcial y los difíciles con misma altura que los moderados, pero con un nivel máximo de oclusión.

Method	Moderate	Easy	Hard	Runtime
SubCNN – Xiang et al. (2016)	89.04 %	90.81 %	79.27 %	2 s / GPU
MS-CNN – Cai et al. (2016)	89.02 %	90.03 %	76.11 %	0.4 s / GPU
SDP+RPN – Yang et al. (2016)	88.85 %	90.14 %	78.38 %	0.4 s / GPU
Mono3D – Chen et al. (2016a)	88.66 %	92.33 %	78.96 %	4.2 s / GPU
3DOP – Chen et al. (2015c)	88.64 %	93.04 %	79.10 %	3s / GPU
MV3D (LIDAR + MONO) – Chen et al. (2016c)	87.67 %	89.11 %	79.54 %	0.45 s / GPU
SDP+CRC (ft) – Yang et al. (2016)	83.53 %	90.33 %	71.13 %	0.6 s / GPU
Faster R-CNN – Ren et al. (2015)	81.84 %	86.71 %	71.12 %	2 s / GPU
AOG – Wu et al. (2016a)	75.94 %	84.80 %	60.70 %	3 s / 4 cores
3DVP – Xiang et al. (2015b)	75.77 %	87.46 %	65.38 %	40 s / 8 cores
LSVM-MDPM-sv – Felzenszwalb et al. (2010)	56.48 %	68.02 %	44.18 %	10 s / 4 cores
ACF – Dollár et al. (2014)	54.74 %	55.89 %	42.98 %	0.2 s / 1 core

(a) KITTI Car Detection Leaderboard

Method	Moderate	Easy	Hard	Runtime
MS-CNN – Cai et al. (2016)	73.70 %	83.92 %	68.31 %	0.4 s / GPU
SubCNN – Xiang et al. (2016)	71.33 %	83.28 %	66.36 %	2 s / GPU
IVA – Zhu et al. (2016)	70.70 %	83.63 %	64.67 %	0.4 s / GPU
SDP+RPN – Yang et al. (2016)	70.16 %	80.09 %	64.82 %	0.4 s / GPU
3DOP – Chen et al. (2015c)	67.47 %	81.78 %	64.70 %	3s / GPU
Mono3D – Chen et al. (2016a)	66.68 %	80.35 %	63.44 %	4.2 s / GPU
Faster R-CNN – Ren et al. (2015)	65.90 %	78.86 %	61.18 %	2 s / GPU
SDP+CRC (ft) – Yang et al. (2016)	64.19 %	77.74 %	59.27 %	0.6 s / GPU
RPN+BF – Zhang et al. (2016a)	61.29 %	75.45 %	56.08 %	0.6 s / GPU
DPM-VOC+VP – Pepik et al. (2015)	44.86 %	59.48 %	40.37 %	8 s / 1 core
SubCat – Ohn-Bar & Trivedi (2015)	42.34 %	54.67 %	37.95 %	1.2 s / 6 cores
ACF – Dollár et al. (2014)	39.81 %	44.49 %	37.21 %	0.2 s / 1 core
LSVM-MDPM-sv – Felzenszwalb et al. (2010)	39.36 %	47.74 %	35.95 %	10 s / 4 cores

(b) KITTI Pedestrian Detection Leaderboard

Method	Moderate	Easy	Hard	Runtime
MS-CNN – Cai et al. (2016)	75.46 %	84.06 %	66.07 %	0.4 s / GPU
SDP+RPN – Yang et al. (2016)	73.74 %	81.37 %	65.31 %	0.4 s / GPU
SubCNN – Xiang et al. (2016)	71.06 %	79.48 %	62.68 %	2 s / GPU
3DOP – Chen et al. (2015c)	68.94 %	78.39 %	61.37 %	3s / GPU
IVA – Zhu et al. (2016)	67.47 %	80.17 %	59.66 %	0.4 s / GPU
Mono3D – Chen et al. (2016a)	66.36 %	76.04 %	58.87 %	4.2 s / GPU
Faster R-CNN – Ren et al. (2015)	63.35 %	72.26 %	55.90 %	2 s / GPU
SDP+CRC (ft) – Yang et al. (2016)	61.31 %	74.08 %	53.97 %	0.6 s / GPU
Regionlets – Wang et al. (2015)	58.72 %	70.41 %	51.83 %	1 s / > 8 cores
DPM-VOC+VP – Pepik et al. (2015)	31.08 %	42.43 %	28.23 %	8 s / 1 core
LSVM-MDPM-us – Felzenszwalb et al. (2010)	29.88 %	38.84 %	27.31 %	10 s / 4 cores

(c) KITTI Cyclist Detection Leaderboard

Tabla 1. Detección de objetos KITTI

Como se puede observar, las Redes Neuronales Convolucionales (CNN) fueron los métodos más usados para la implementación de este tipo de algoritmos, permitieron una mejora significativa en el desempeño de la detección de objetos. Por otra parte, se desarrollaron las RCNNs para resolver el problema de la localización con un paradigma de "reconocimiento por regiones", con ello se generaron muchas propuestas de región utilizando la búsqueda selectiva Uijlings et al., (2013) que extraen un vector de características de longitud fija para cada propuesta utilizando una CNN y clasificando cada región con una SVM lineal (Máquinas de vectores de soporte).

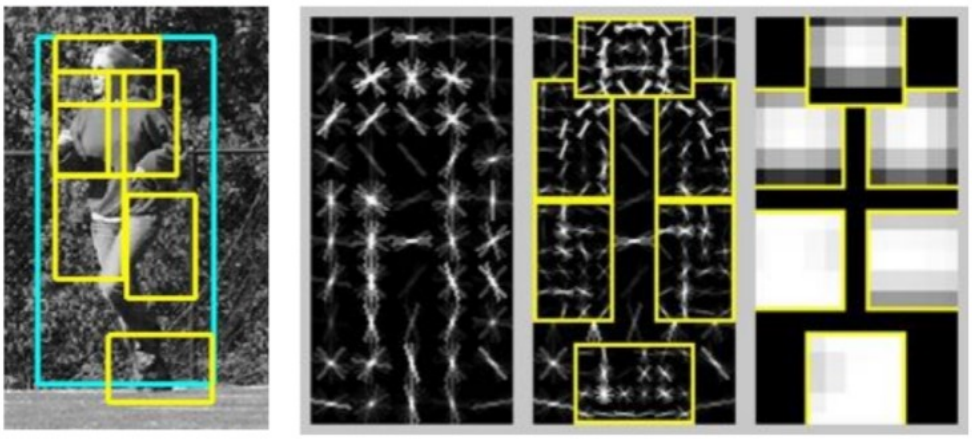


Ilustración 6. Un ejemplo de detección de personas obtenido con el Deformable Part Model propuesto por Felzenszwalb et al. (2008).

Las CNNs regionales que surgieron eran computacionalmente costosas, por ello, se propusieron varias mejoras, en He et al (2014) y Girshick (2015). He et al. (2014) utilizaron la agrupación de pirámides espaciales que permite calcular un mapa de características convolucionales para toda la imagen con solo una ejecución de la CNN en contraste con la R-CNN que debe aplicarse en muchas regiones de imagen. Girshick (2015) la mejora aún más con un algoritmo de entrenamiento de una sola etapa que aprende conjuntamente a clasificar las propuestas de objetos y a refinar sus ubicaciones espaciales.

A pesar de que estas redes basadas en regiones han demostrado ser muy exitosas en el benchmark PASCAL VOC, no pudieron lograr un rendimiento similar en KITTI. La razón principal de esto es que el conjunto de datos KITTI contiene objetos en muchas escalas diferentes y pequeños objetos que a menudo están muy ocluidos o truncados. Estos objetos están destinados a detectar la utilización de redes basadas en la Unión. Por lo tanto, se han propuesto varios métodos para obtener mejores propuestas de objeto Ren et al (2015) , Chen et al (2016b, a), Yang et al (2016), Cai et al (2016).

SEGMENTACIÓN SEMÁNTICA

Es un campo de trabajo importante en la visión por computador. El objetivo de la segmentación es asignar a cada pixel en la imagen un color o nivel de un conjunto de categorías predefinido. En la ilustración 6 se muestra como cada uno de los pixeles esta categorizado en un color especifico en función del objeto. La segmentación de imágenes en regiones semánticas habitualmente se usa en escenas urbanas, donde nos encontramos vehículos, peatones o carreteras, permitiendo comprender de manera

global el entorno para la navegación autónoma. Los desafíos de la segmentación semántica surgen de la complejidad de la escena y del tamaño del espacio de la etiqueta.

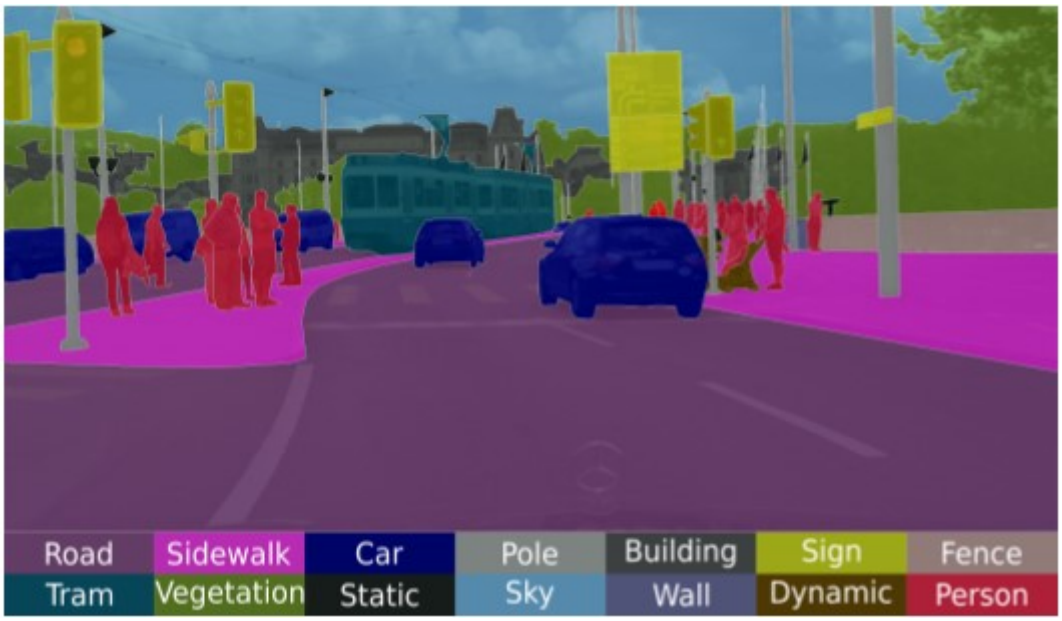


Ilustración 7. Segmentación semántica de una escena de Cityscapes dataset por Cordts et al. (2016) grabada en Zürich.

Tradicionalmente, el problema de la segmentación semántica se planteó a través de la técnica de máximo a posteriori (MAP), en un campo aleatorio condicional (CRF), definido sobre píxeles o superpíxeles He et al., (2004, 2006). Sin embargo, estas primeras formulaciones no fueron eficaces y sólo podían manejar conjuntos de datos de tamaño limitado y un número pequeño de clases. Además, sólo fueron explotadas las características más simples como el color, el borde y la textura. Shottonetal (2009) observó que las características más poderosas representan un desempeño muy positivo y propuso un enfoque basado en un nuevo tipo de características llamado filtro de disposición de textura que explora la apariencia textural de los objetos, su disposición y contexto textural, combinaba filtros de textura con propiedades de imagen de nivel inferior en CRF para obtener secuencias de niveles de píxeles. Se explotan las técnicas de entrenamiento aleatorio y de control por pieza para entrenar el modelo. Se consideró que la conectividad era jerárquica y de largo alcance, así como los potenciales de orden superior definidos en regiones de imágenes, abordaban la capacidad limitada de los CRF para modelar interacciones de largo alcance dentro de la imagen. Sin embargo, los métodos basados en regiones de imagen He et al (2004), Kumar & Hebert (2005), He et al (2006), Kohli et al (2009), Ladicky et al (2009, 2014) eran restringidos por la precisión de las segmentaciones de imagen utilizadas como entrada. Por el contrario,

Kr'ahenbühl & Koltun (2011) proponen un algoritmo de inferencia altamente eficiente para modelos CRF completamente conectados.

El éxito de las redes neuronales convolucionales profundas para la clasificación de imágenes y la detección de objetos despertó interés en aprovechar su poder para resolver la tarea de segmentación semántica en píxeles. La red neural completamente convolucional, Long et al (2015), es una de las primeras obras que aplica CNNs al problema de segmentación de imágenes. Sin embargo, mientras que las modernas redes neuronales convolucionales combinan la información contextual de múltiples escalas mediante agrupaciones consecutivas y subempleo, la segmentación semántica requiere un razonamiento contextual multi-escala junto con una predicción densa de resolución completa. A continuación, examinaremos los enfoques recientes que abordan este problema.

Enfocamos la comparación de diferentes tipos de segmentación semántica (métodos) en el conjunto de datos Cityscapes26 de Cordts et al. (2016) descrita en la sección de detección de objetos debido al contexto de conducción autónoma. La Tabla 2 muestra la clasificación de Cityscapes para la tarea de etiquetado semántico de nivel de píxeles. La métrica intersección-sobreunión se proporciona para dos granularidades semánticas, es decir, clases y categorías, y además el IoU (Índice de Jaccard, comúnmente conocido como el PASCAL VOC métrica de intersection-over-union) ponderado por la instancia se informa para ambas granularidades para penalizar los métodos ignorando instancias pequeñas.

Method	IoU class	iIoU class	IoU category	iIoU category
ResNet-38 – Wu et al. (2016b)	80.6	57.8	91	79.1
PSPNet – Zhao et al. (2016)	80.2	58.1	90.6	78.2
RefineNet – Lin et al. (2016a)	73.6	47.2	87.9	70.6
LRR-4x – Ghiasi & Fowlkes (2016)	71.8	47.9	88.4	73.9
FRRN – Pohlen et al. (2016)	71.8	45.5	88.9	75.1
Adelaide_context – Lin et al. (2016b)	71.6	51.7	87.3	74.1
DeepLabv2-CRF – Chen et al. (2016b)	70.4	42.6	86.4	67.7
Dilation10 – Yu & Koltun (2016)	67.1	42	86.5	71.1
DPN – Liu et al. (2015)	66.8	39.1	86	69.1
Scale invariant CNN + CRF – Krešo et al. (2016)	66.3	44.9	85	71.2
FCN 8s – Long et al. (2015)	65.3	41.7	85.7	70.1
DeepLab LargeFOV StrongWeak – Papandreou et al. (2015)	64.8	34.9	81.3	58.7
Pixel-level Encoding for Instance Segmentation – Uhrig et al. (2016)	64.3	41.6	85.9	73.9
DeepLab LargeFOV Strong – Chen et al. (2015b)	63.1	34.5	81.2	58.7
Segnet basic – Badrinarayanan et al. (2015)	57	32	79.1	61.9

(a) CITYSCAPES Semantic Segmentation Leaderboard

Method	AP	AP 50%	AP 100m	AP 50m
DIN – Arnab & Torr (2017)	20	38.8	32.6	37.6
Shape-Aware Instance Segmentation – Hayder et al. (2016)	17.4	36.7	29.3	34
DWT – Bai & Urtasun (2016)	15.6	30	26.2	31.8
InstanceCut – Kirillov et al. (2016)	13	27.9	22.1	26.1
Joint Graph Decomposition and Node Labeling – Levinkov et al. (2016)	9.8	23.2	16.8	20.3
Pixel-level Encoding for Instance Segmentation – Uhrig et al. (2016)	8.9	21.1	15.3	16.7
R-CNN + MCG convex hull – Cordts et al. (2016)	4.6	12.9	7.7	10.3

Tabla 2. CITYSCAPES Tabla de segmentación semántica.

Recientemente, se han propuesto varios métodos para comparar la oposición con la inferencia de escala múltiple y la predicción de la resolución completa. Se han propuesto convoluciones dilatadas, Chen y otros (2015b), Yu y Koltun (2016) para ampliar el campo receptivo de redes neuronales sin pérdida de resolución. Su funcionamiento corresponde a una convolución regular con filtros dilatados, lo que permite un razonamiento de escala múltiple eficiente, limitando al mismo tiempo el aumento del número de parámetros del modelo. En el modelo SegNet, Badrinarayanan et al. (2015) han sustituido el decodificador tradicional en una arquitectura profunda con una red que consiste en una jerarquía de decodificadores que responde a cada codificador. Cada decodificador asigna un mapa de características de baja resolución de un codificador (capa de agrupación máxima) a un mapa de características de mayor resolución. En particular, el codificador en su modelo se aprovecha de los índices de agrupación calculados en el esquema de agrupación máxima que corresponde a la realización del proceso de muestreo, esto elimina la necesidad de aprender el 'upsampling' y por lo tanto resulta un número más pequeño de parámetros, se han demostrado límites de segmentación más nítidos utilizando este enfoque. Mientras que los mapas de activación en los niveles más bajos de la CNN jerarquizados con una categoría de objeto específica, contienen una información espacial más alta.

Ghiasi & Fowlkes (2016) aprovechan esta suposición y proponen construir una pirámide laplaciana basada en una red completamente convolucional. La agregación de información a múltiples escalas les permite repetir sucesivamente el límite reconstruido a partir de capas de baja resolución. Esto se logra mediante el uso de conexiones de saltos de mapas de características de resolución más alta y de una con figuración de confianza multiplicativa, penalizando sin resultados de alta resolución en regiones donde las predicciones de resolución tienen una alta confianza. Con este acercamiento Ghiasi & Fowlkes (2016) alcanzan los niveles de competitividad de los paisajes urbanos Tabla 2. Uno de los mejores métodos para el diseño de paisajes de la ciudad fue propuesto por Zhao et al. (2016) usando una red de análisis de escenas de pirámide, Ilustrado en la Figura 8, para incorporar información de contexto global en la tarea de predicción de nivel de píxeles. Específicamente, aplican un módulo de análisis piramidal a la última capa convolucional de una CNN que fusiona características de varias escalas piramidales para combinar la información de contexto local y global. La representación resultante se coloca en una capa de conversión para obtener predicciones de píxeles finales.

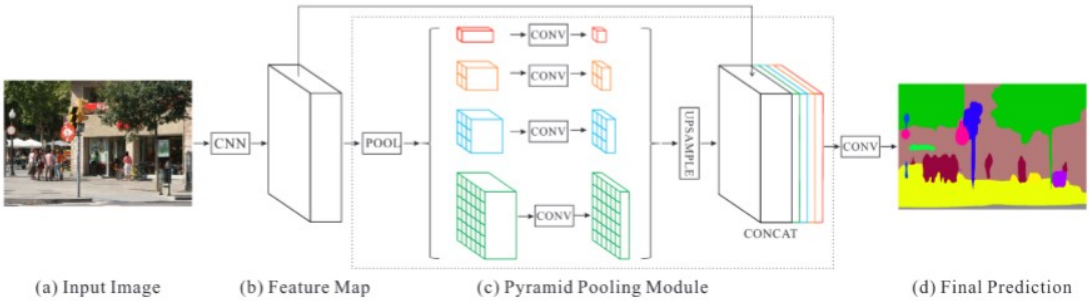


Ilustración 8. Muestras del método propuesto por Zhao et al. (2016). The pyramid parsing module (c) is applied on a CNN feature map (b) and fed into a convolutional layer for pixel-level estimation (d). Adapted from Zhao et al. (2016).

Simonyan y Zisserman (2015) y Szegedy et al. (2015) han demostrado que la profundidad de una CNN es crucial para representar las características más relevantes. Sin embargo, el aumento de la profundidad de una red conduce a la saturación y degradación de la precisión. He et al. (2016) propone un marco de aprendizaje residual profundo (ResNet) para abordar este problema. Dejan que cada capa apilada aprenda un mapeado residual de la asignación original, no referenciada, esto les permite entrenar redes más profundas con una mayor precisión, mientras que las redes simples (redes simplemente apiladas) exhibían errores de formación más altos. Pohlen et al. (2016)

presenta una arquitectura similar a ResNet que proporciona un sólido rendimiento de reconocimiento, al mismo tiempo que preserva información de alta resolución en toda la red, combinando tres vías de procesamiento diferentes. Las dos corrientes de procesamiento se combinan en la resolución de imagen completa usando residuos. Wu et al. (2016b) propuso una arquitectura ResNet más eficiente analizando las profundidades efectivas de unidades residuales. Cuentan que los ResNet necesitan conjuntos relaciones relativas a las redes. Basándose en esta comprensión, diseñan una red frecuencial para las redes convolucionarias de baja resolución para la tarea de segmentación de imágenes semánticas. Mientras Pohlen et al. (2016) logra resultados competitivos en Cityscapes, Wu et al. (2016b) superó a todos los demás en todas las medidas, además de IoU de nivel de clase ponderado por la instancia.

Una manera diferente de abordar las necesidades de la inferencia de múltiples escalas y predicción de resolución completa es la combinación de CNNs con modelos de CRF. Chen et al. (2015b) propone redefinir el mapa de etiquetas obtenido utilizando una red neuronal convolucional usando un modelo de CRF completamente conectado Kr¨ahenb¨uhl & Koltun (2011). El modelo CRF define detalles basados en la entrada RGB cruda que faltan en la salida CNN debido a la precisión espacial limitada del modelo CNN. En un modelo similar, Jampani et al. (2016) generaliza los filtros bilaterales y desarrolla el programa CRF que permite el entrenamiento de extremo a extremo de los parámetros de filtro (generalizados) a partir de los datos. Esto permite, de manera efectiva, razonar sobre regiones espaciales más grandes dentro de una única capa de conversión, mediante el aumento de la potencia de entrada de una señal de guiado. Inspirado por CRF de orden superior para la segmentación semántica, Gadde et al. (2016a) propone un nuevo Módulo de Iniciación Bilateral para las arquitecturas de CNN, como la estructura analítica de CNN y las CRF. Ellos asumen que los pixeles que son espacialmente y fotométricamente similares tienen más probabilidades de tener la misma etiqueta. Esto les permite aprender directamente interacciones de largo alcance, eliminando así la necesidad de post-procesamiento usando modelos CRF. Específicamente, los módulos propuestos propagan información de borde-consciente entre píxeles distantes basados en su similitud espacial y de color, incorporando la disposición espacial de superpixeles. La propagación de la información se logra aplicando filtros bilaterales con granos gaussianos a diversas escalas.

Como conclusión, el enfoque de los métodos modernos en multi-escala llevó a resultados impresionantes en la segmentación semántica en píxeles en Cityscapes. Hoy en día, los mejores métodos en Cityscapes Tabla 2 alcanzan una IoU impresionante de casi 81% sobre las clases y 91% sobre las categorías. En contraste, la IoU ponderada por la instancia siempre está por debajo del 58% sobre las clases y el 80% sobre las categorías. Esto indica que la segmentación semántica funciona bien con instancias que cubren grandes áreas de imagen, pero sigue siendo problemática con instancias que cubren regiones pequeñas. De forma similar a la detección en resoluciones bajas discutida anteriormente, las regiones pequeñas proporcionan poca información para asignar la etiqueta correcta. Además segmentar objetos pequeños y posiblemente ocluidos es una tarea desafiante que podría requerir enfoques novedosos para realizar conjuntamente la estimación de profundidad y el reconocimiento adaptativo de la misma.

2.4.Contexto

Introducida la evolución de los algoritmos relacionados con la detección de personas y la segmentación semántica de imágenes, principales en el desarrollo de bases de datos relacionadas con este proyecto, se desarrolla en el artículo Dense Semantic Stereo Labelling Architecture for In-Campus Navigation, por Jorge Beltrán (et al. 2017), a partir del cual se creará el script de este trabajo y su posterior incorporación a un sitio web, y del que se va a dar explicación en este apartado.

El artículo presenta una nueva arquitectura para el etiquetado ROS de escena semántica basada en estéreo-visión (etiquetado que se realiza en este proyecto). Su objetivo es proporcionar la información necesaria para planificar una ruta y conseguir la navegación autónoma por el campus universitario. El algoritmo tiene como salida la clasificación de los obstáculos de la escena en cuatro categorías: zonas de tránsito, jardín, personas y obstáculos estáticos. La validación de la clasificación del algoritmo se logra de la comparación de la secuencia estéreo capturada en el campus, con el etiquetado a mano de las fotografías captadas por el vehículo.

Ilustración 9. iCab vehículo autónomo.

La arquitectura fue diseñada para funcionar en la plataforma en desarrollo llamada iCab Hussein et al. (2016), vehículo autónomo ya mencionado (ver ilustración 9). Dicho diseño se construye para formar parte del sistema ROS (Robot Operating System) Garage et al. (2010).

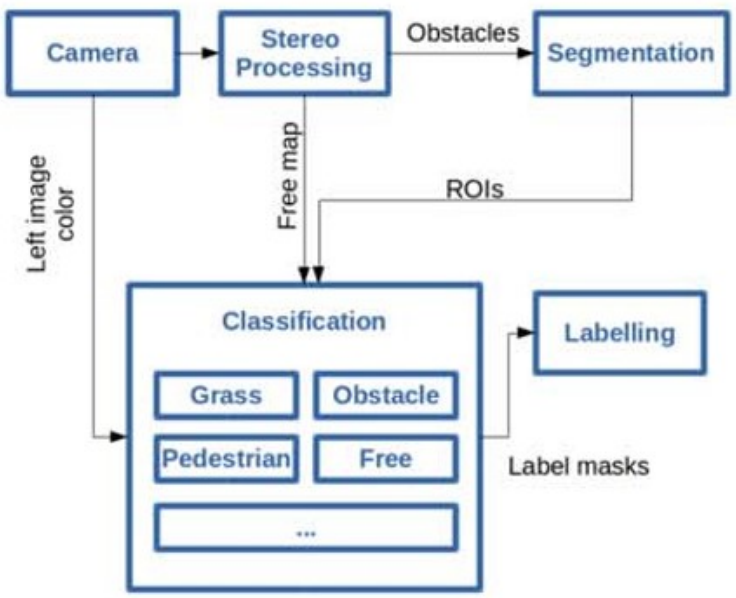


Ilustración 10. Arquitectura de la que se basa el proyecto.

Como se puede observar, la arquitectura (ilustración 10) está compuesta por 5 nudos diferentes, cada uno de ellos correspondiendo a los estados del algoritmo. Comienza por la adquisición de las imágenes (camera), seguido del procesamiento estéreo que construye el mapa de disparidad. Después, los mapas de uv-disparity son estimados ya que con ellos se pueden calcular los obstáculos y el área libre. El estado de la segmentación separa los diferentes obstáculos mediante ROIs basado en la información de la disparidad. En la fase de clasificación, con las imágenes clasificadas previamente y las ROIs en orden se determina a que categoría pertenece cada pixel de la imagen. Finalmente, en el *labelling* se organizan y se recuentan los pixeles para obtener su calificación o etiqueta final.

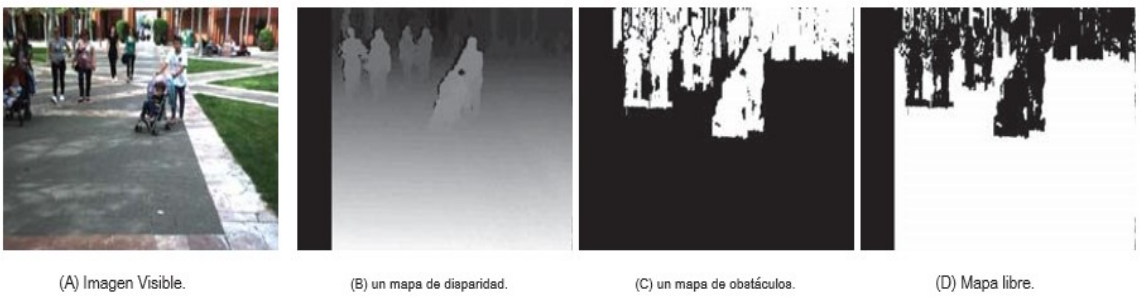


Ilustración 11. Ejemplo de estimación obstáculo y tierra.

En cuanto a la métrica utilizada en esta arquitectura, con el fin de evaluar el etiquetado semántico por pixel, se utiliza por un lado, la precisión global obtenida con la relación entre los TP (True Positives) y el número de pixeles totales, lo que permite determinar la proporción de pixeles etiquetados correctamente. Por otro lado, con el objetivo de identificar la exactitud por clase, se utiliza el índice de Jaccard (ya nombrado en el apartado anterior) comúnmente conocido como PASCAL COV. El índice de Jaccard se calcula como en la segunda fórmula de la ilustración 12, donde TP, FP y FN representan verdaderos y falsos positivos, verdaderos y falsos negativos. Los resultados presentados corresponden al rendimiento calculado de todo el conjunto de imágenes.

Con el fin de medir la calidad del método de etiquetado de imágenes en tiempo real para la navegación autónoma, los dos algoritmos de disparidad conocidos se ponen a prueba. De este modo, es posible comparar cuál ofrece el mejor compromiso entre la precisión y el tiempo de cálculo por pixel, a condición de que nuestro método se base en gran medida de la calidad del mapa de disparidad en sus primeras etapas. Los métodos de disparidad que se consideraron en los resultados experimentales son bloque de adaptación (BM) y el bloque de adaptación global (SGBM) Hirschmüller et al. (2008).

Los resultados "prototipo" obtenidos con esta arquitectura fueron los de las siguientes tablas, muestran el nivel de correlación de las etiquetas con los métodos usados.

Disparity Method	Pixel-wise Accuracy	IoU_{class}			
		Free	Garden	Obstacle	Pedestrian
BM	86.81	87.42	69.01	66.41	47.05
SGBM	87.79	88.34	70.31	67.75	43.43

Tabla 3. Clasificación de acierto 4 clases.

Disparity Method	Pixel Accuracy	IoU_{class}		
		Free	Garden	Obst.
BM	88.35	87.42	69.01	72.31
SGBM	89.53	88.34	70.31	74.74

Tabla 4. Clasificación de acierto 3 clases.

En los próximos apartados se explicará cómo se desarrolla la métrica interna para la obtención de los resultados mostrados.

Capítulo 3:

3. Descripción general

3.1. Propósito y características generales

El propósito del proyecto es crear un sitio web en el que los usuarios, que desarrollan algoritmos para el funcionamiento del vehículo inteligente iCab de la UC3M de Leganés pertenecientes al Intelligent Systems Lab (LSI) Research Group, puedan subir el resultado (en forma de etiquetas, o imágenes etiquetadas) de esos algoritmos y obtener en forma de porcentaje el nivel de acierto con respecto a su semejanza con la realidad.

Además, en la web desarrollada se guardarán los resultados obtenidos por los usuarios y se irán incorporando y ordenando en tablas que se pondrán a disposición de los mismos para su consulta. Con esto se logrará un portal donde se irán recogiendo el nivel de los algoritmos y poder elegir, ante un caso de su uso, el mejor sistema inteligente para cada momento.

Un algoritmo, da igual del tipo que sea, está sujeto a variabilidades diversas como puede ser la diversidad de objetos que se segmentan, tipos de medios usados o la interacción con el usuario, sea cual sea la comparación de métricas para la evaluación de su precisión deberá realizarse usando las estadísticas adecuadas. Debido a esto, unos algoritmos tendrán ventajas con respecto a otros ya sea por lo comentado o por el número insuficientes imágenes en nuestro caso, por ello no solo se compararán el resultado de los sistemas inteligentes de los usuarios, sino que también se tendrán en cuenta los medios o técnicas usados para su obtención.

El nivel de acierto vendrá de la comparación de las imágenes etiquetadas de los usuarios con una base de datos desarrollada en este trabajo, dicha base de datos se ha creado a mano con el programa GIMP lo cual hace que su nivel de similitud con las fotografías originales (adquiridas de la cámara del vehículo autónomo iCab) sea muy alta y pueda usarse como punto de comparación ante etiquetas que surgen de un algoritmo.

Para poder comparar nuestra base de datos con el resultado de los usuarios, se crea un algoritmo o script basado en el artículo "Dense Semantic Stereo Labelling Architecture" [16] (algoritmo introducido en el contexto del trabajo, punto 2.4) el cual basa su funcionamiento en las siguientes formulas:

$$ACC = \frac{TP}{\text{num. of pixels}}$$

$$IoU_{class} = \frac{TP_{class}}{TP_{class} + FP_{class} + FN_{class}}$$

Ilustración 12. Formulas algoritmo

Para el desarrollo del algoritmo se ha utilizado la versión de Python 3.5 de 64-bit para Linux (Ubuntu 16.04) además para su codificación se han utilizado librerías de las OpenCV versión 3.2.0 con el gestor de paquetes Anaconda, para programar en Python se ha utilizado el programa Spyder 3.0.

Una vez creado el algoritmo se pasará a la creación de la web, para ello se utilizará como base o sistema de gestión de contenido Wordpress, el cual está enfocado a la creación de cualquier tipo de sitio web. Para darle forma a la web se utilizarán los lenguajes PHP (Hypertext Prpprocessor) [17], Java Script y HTML. Para lograr algunas funciones de la web que no están incluidas se insertarán algunos plugings como *Scripts-n-styles, Simple-embed-code o Table Sorter.*

3.2.Hardware utilizado

Para elaboración de este proyecto se ha utilizado un ordenador de gama baja-media (LG). Las especificaciones de este dispositivo que más influyen a este proyecto son las siguientes:

- Fabricante del sistema: Gigabyte Technology Co., Ltd.
- Procesador: Intel® Core™ i5-3550 CPU @ 3.30GHz, 3701 Mhz, 4 procesadores principales, 4 procesadores lógicos.
- Memoria instalada (RAM): 8,00 GB.
- Tipo de sistema: Sistema operativo de 64 bits, procesador x64.

En primer momento para el desarrollo del etiquetado y conseguir la base de datos se utilizo como Sistema Operativo (SO) Windows 10 Home, pero para poder seguir con el proyecto se necesitaba el denominado software libre y de código abierto Ubuntu 16.04, el cual proporciona a nivel técnico y operativo una serie de ventajas para el cliente. Algunos ejemplos claros de las típicas ventajas que proporciona son los bajos costes al eliminar los costes derivados de licencias. La elevada fiabilidad al ser testeados, usados y corregidos continuamente en diferentes entornos. La mayor seguridad que presenta al ser pública su especificación y estar sometida a constante revisión. La gran flexibilidad que da el poder modificar y adaptar el código a los requerimientos particulares del cliente. La inter-operabilidad que proporciona al estar basado generalmente en estándares abiertos. La fácil integración al permitir la modificación del código y ser software usado y nacido de las necesidades de los usuarios. El mantenimiento es por otra parte flexible y competitivo al no estar monopolizado por una compañía [19]. Por lo tanto, para nuestro caso y poder codificar completamente y sin restricciones la web, Ubuntu (Linux) es el sistema operativo que mejor se adapta a nuestras necesidades.

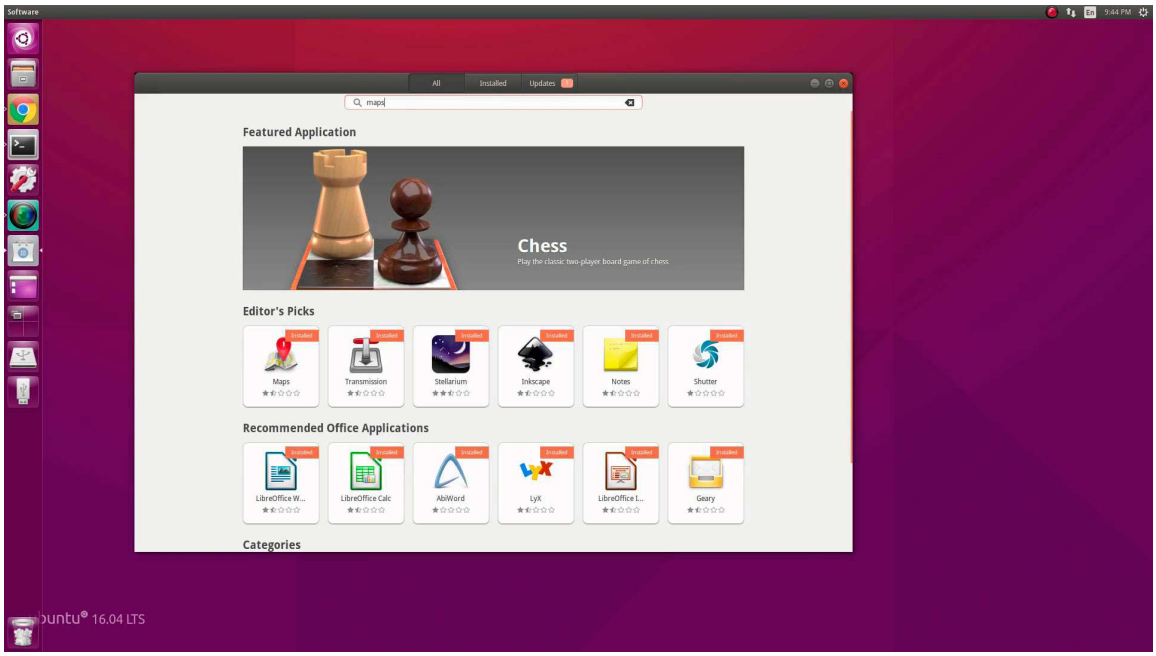


Ilustración 13. Visión SO Ubuntu

Para el posterior desarrollo del algoritmo y del sitio web se llevó a cabo una partición del disco duro, en la cual se dejaron en funcionamiento; en un lado de la memoria, el sistema operático que teníamos y en el otro, se instaló la versión de Ubuntu 16.04 LTS.

La partición del disco duro quedó como se muestra en la ilustración 14.

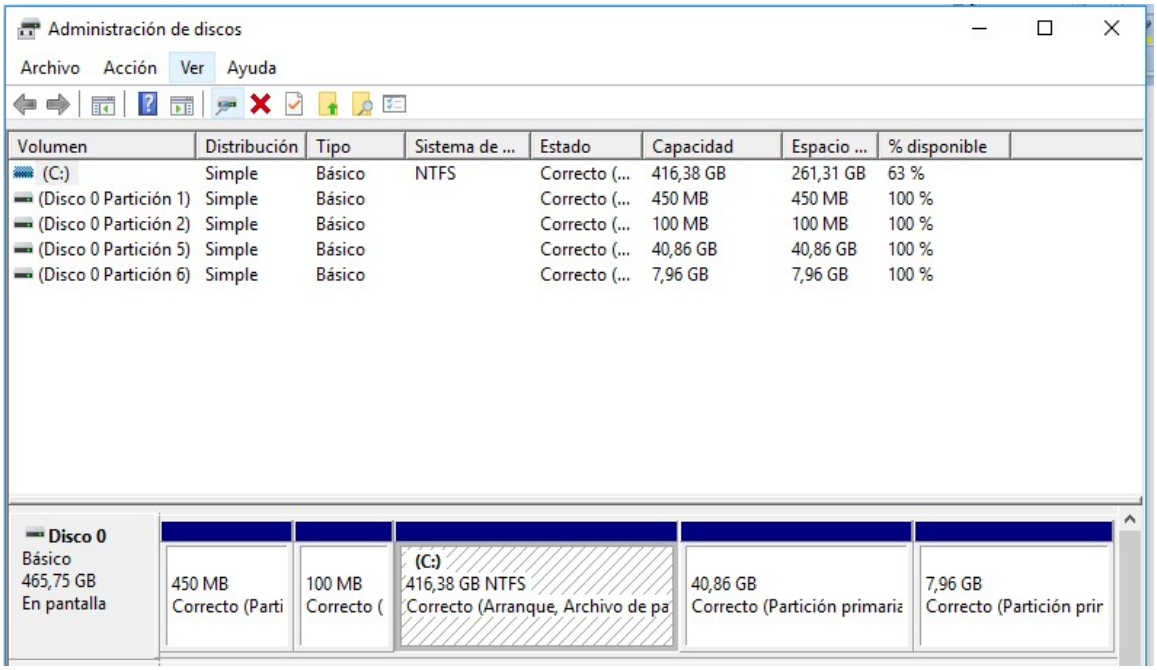


Ilustración 14. Partición del disco duro

3.3.Software y lenguaje utilizado

3.3.1. Ubuntu 16.04 LTS

Como ya se ha introducido en el punto anterior, el sistema operativo utilizado principalmente para la realización del proyecto ha sido Ubuntu. Es un SO basado en Linux que se distribuye como software libre e incluye su propio escritorio denominado Unity.

Está orientado principalmente para los usuarios promedio, dado su facilidad de uno y su finalidad en mejorar la experiencia del usuario. Su software es de código abierto (desarrollado y distribuido libremente) y es gratuito, financiado por medios de servicios vinculados al sistema operativo y vendiendo soporte técnico [20]. Además, su mantenimiento es libre y gratuito.

Ubuntu es una bifurcación del código base del proyecto Debian. El objetivo inicial era hacer de Debian una distribución más fácil de usar y entender para los usuarios finales, corrigiendo varios errores de este y haciendo más sencillas algunas tareas como la gestión de programas. Su primer lanzamiento fue el 20 de octubre de 2004 [21].

En cuanto a la interfaz del usuario, Ubuntu en el comienzo utilizó GNOME con un panel inferior para listar ventanas y un panel superior para menús e indicadores de sistema, pero desde la versión 11.04 el equipo de Canonical decidió lanzar su propia interfaz de usuario, de esa manera Unity fue diseñado para optimizar el espacio e interacción de la interfaz de Ubuntu [22].

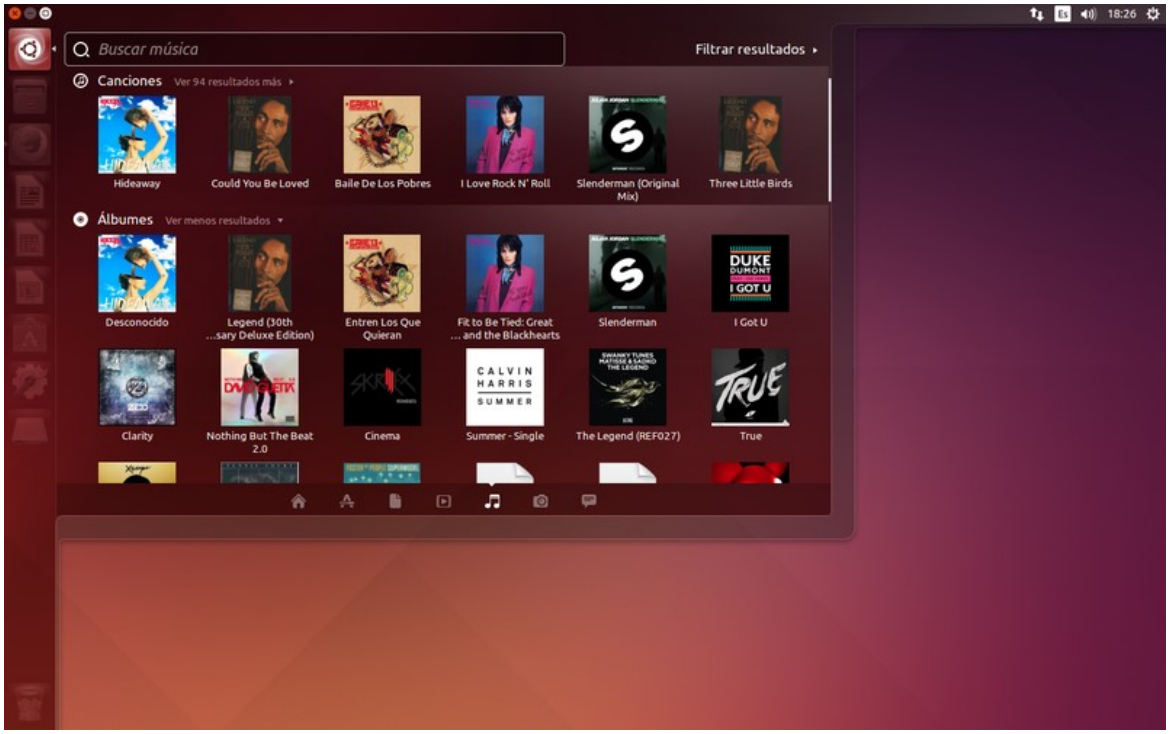


Ilustración 15. Tablero Ubuntu 16.04

Los requisitos mínimos que se deben de tener para el uso de este SO son los siguientes:

- Procesador x86 a 700 MHz.
- Memoria RAM de 512 Mb.
- Disco Duro de 5 GB (swap incluida).
- Tarjeta gráfica y monitor capaz de soportar una resolución de 1024x768.

- Lector de DVD o puerto USB.
- Conexión a Internet puede ser útil.

Los efectos de escritorio, proporcionados por Compiz, se activan de forma predeterminada en las siguientes tarjetas gráficas:

- Intel (i915 o superior, excepto GMA 500, nombre en clave «Poulsbo»)
- NVidia (con su controlador propietario o el controlar abierto incorporado Nouveau)
- ATI (a partir del modelo Radeon HD 2000 puede ser necesario el controlador propietario fglrx)

Para una instalación óptima, y sobre todo si se dispone de más de 3 GiB de RAM, existe también una versión de Ubuntu para sistemas de 64 bits [23].

3.3.2. GIMP 2.8

Para la parte del proyecto del etiquetado de las imágenes se podrían haber utilizado varios programas, finalmente se eligió GIMP 2.8 ya que es libre y tiene licencia gratuita en los dos SO utilizados (Windows y Ubuntu), formando parte además del proyecto GNU (Linux). Además de la licencia, se ha elegido este programa por su tratado de imágenes por capas, muy importante en nuestro caso ya que en el **etiquetado de las imágenes** es algo fundamental.

Ilustración 16. Logo GIMP

GIMP (GNU Imagen Manipulation Program) es un programa de edición de imágenes digitales en forma de mapa de bits, sirviendo tanto para fotografías como para dibujos. Incluye diferentes herramientas que se utilizan para el retoque y la edición de imágenes, dibujo de formas libres, cambiar el tamaño, recortar, hacer fotomontajes, convertir a diferentes formatos de imagen, y otras tareas más especializadas. Se pueden

también crear imágenes animadas en formato GIF e imágenes animadas en formato MPEG usando un plugin de animación [24].

El programa he ido evolucionando muy rápido a lo largo del tiempo, soportando nuevos formatos, mejorando sus herramientas y funcionando con diferentes plugings o extensiones. En realidad, GTK+ era simplemente al principio una parte de GIMP, originada al reemplazar la biblioteca comercial Motif usada inicialmente en las primeras versiones de GIMP. GIMP y GTK+ fueron originalmente diseñados para el sistema gráfico X Window ejecutado sobre sistemas operativos tipo Unix. GTK+ ha sido portado posteriormente a Microsoft Windows, OS/2, Mac OS X y SkyOS.

En cuanto a su tratado de las imágenes, se puede hacer por capas (en la parte derecha de la ilustración 11 se pueden observar) pudiendo modificar cada objeto de la imagen de forma independiente al resto, cosa muy importante en este trabajo como se ha comentado anteriormente. La imagen final [25] puede guardarse en el formato original xcf de GIMP que soporta capas, o en un formato plano sin capas, que puede ser png, bmp, jpg, GIF, PDF, etc.

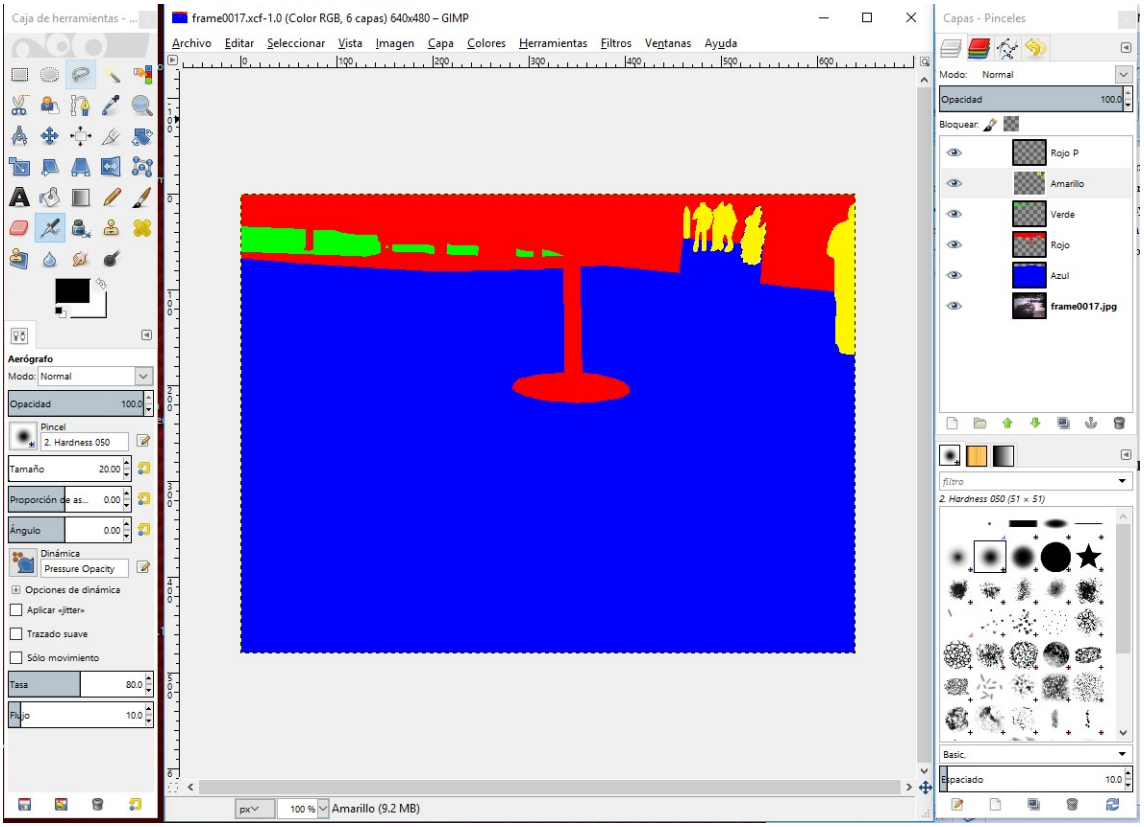


Ilustración 17. Interfaz GIMP 2.8

3.3.3. OpenCV

OpenCV viene de Open Source Computer Vision, se trata de una librería de visión por computador y de algoritmos de aprendizaje automático que se distribuye con la licencia BSD, se usarán sus librerías para el desarrollo del código del algoritmo de comparación de etiquetas, para poder acceder a los pixeles de las imágenes.

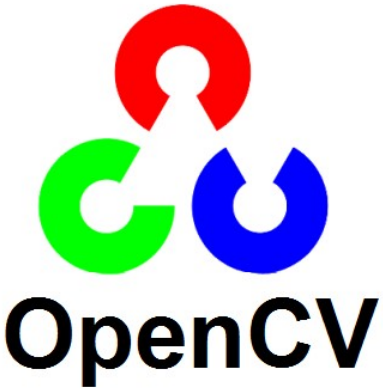


Ilustración 18. Logo OpenCV

Puede ser utilizada de forma gratuita de manera comercial y en la enseñanza. Comenzó a desarrollarse en 1998 por parte de Intel [27], más tarde el soporte cambió a Willow Garage y actualmente está siendo mantenida por Itseez.

La biblioteca cuenta con más de 2500 algoritmos optimizados, que incluye un amplio conjunto de clásicos y de última generación de visión por ordenador y algoritmos de aprendizaje de la máquina. Estos algoritmos pueden ser utilizados para detectar y reconocer rostros, identificar objetos, clasificar acciones humanas en videos, realizar un seguimiento de movimientos de cámara, rastrear objetos en movimiento, extraer modelos 3D de objetos, producir nubes de puntos 3D desde cámaras estéreo, unir imágenes juntas para producir una alta resolución Imagen de una escena completa, buscar imágenes similares de una base de datos de imágenes, quitar los ojos rojos de las imágenes tomadas con flash, seguir los movimientos de los ojos, reconocer el paisaje y establecer marcadores para superponerlo con la realidad aumentada, etc. OpenCV tiene más de 47 mil personas de usuario Comunidad y número estimado de descargas superiores a 14 millones [28].

La librería nos permite trabajar en C, C++, Phyton y Java y además soporta Windows, Linux, Mac OS, Android y iOS [29]. Desde hace algún tiempo los distintos algoritmos utilizados en OpenCV están siendo traducidos a CUDA, lo que permite ejecutar dichos algoritmos en GPUs de Nvidia que tienen incluida dicha característica, aumentando el rendimiento de la app y liberando de carga al procesador. Muchos algoritmos están también diseñados para trabajar en procesadores con varios núcleos, permitiendo su uso en tiempo real. En estos momentos se están desarrollando activamente una completa gama de interfaces CUDA y OpenCL . Hay más de 500

algoritmos y unas 10 veces más funciones que componen o apoyan esos algoritmos. OpenCV se escribe de forma nativa en C ++ y tiene una interfaz plantilla que funciona perfectamente con contenedores STL.

En concreto en este proyecto se da el caso del uso de las librerías cv2, con ellas y mezclándolo con otras sentencias como “imread()” se podrá acceder y estudiar los pixeles de las etiquetas realizadas con el programa explicado en el anterior aparado.

3.3.4. Spyder 3.0

Para la visualización de código y resultados del entorno de lenguaje Python, que es la base del algoritmo de comparación de etiquetas que se desarrolla en este trabajo, se ha utilizado el programa Spyder 3. También se ha usado dada su compatibilidad con el SO Ubuntu.

Ilustración 19. Logo Spyder 3

Es un poderoso entorno de desarrollo interactivo para el lenguaje Python con funciones avanzadas de edición, pruebas interactivas, depuración e introspección.

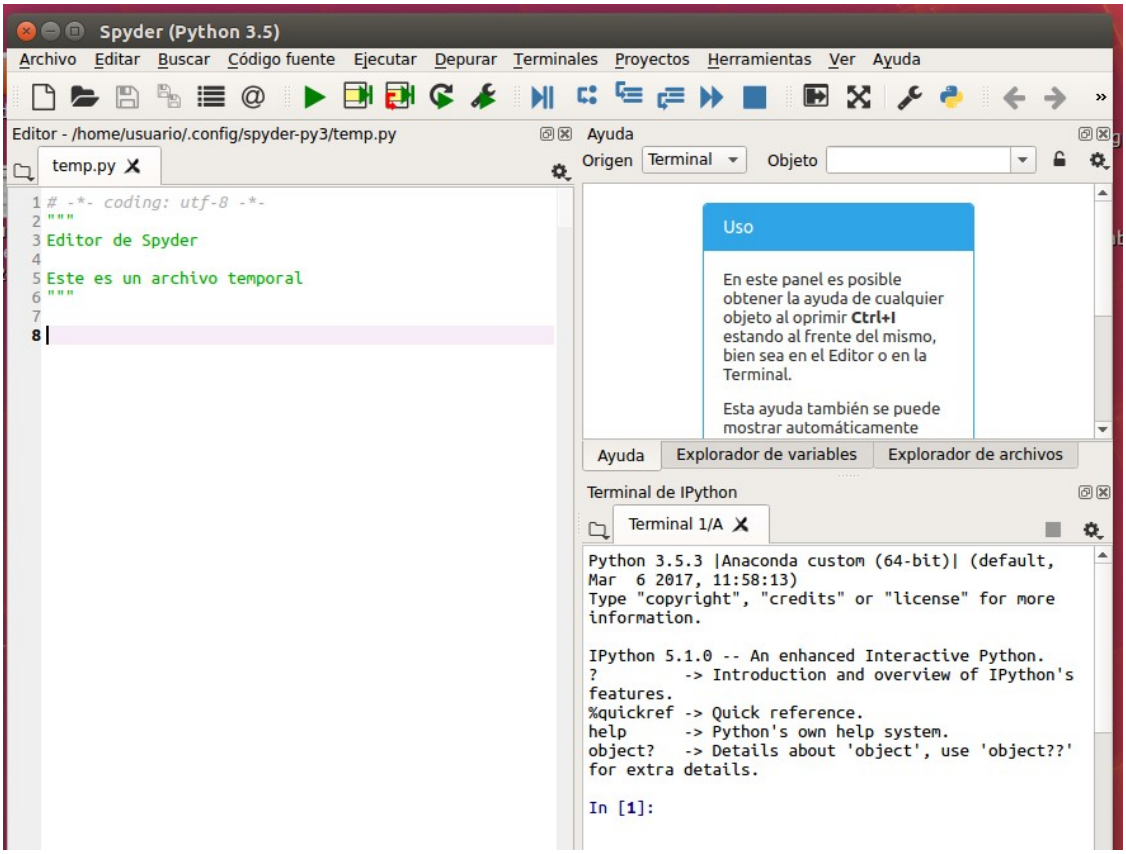


Ilustración 20. Interfaz Spyder 3

Además cuenta con un entorno de computación numérica gracias al soporte de IPython (intérprete de Python interactivo mejorado) y librerías populares de Python como *NumPy* (álgebra lineal), *SciPy* (procesamiento de señales e imágenes) o *matplotlib* (*trazado* interactivo 2D/3D).

Spyder también puede usarse como una biblioteca que proporciona widgets potentes relacionados con consolas para sus aplicaciones basadas en PyQt, por ejemplo, puede ser usado para integrar una consola de depuración directamente en el diseño de su interfaz gráfica de usuario [30], como podemos observar en la ilustración 19 (Página anterior).

3.3.5. WordPress

WordPress es un sistema de gestión de contenidos o CMS (Content Management System) destinado a la creación de cualquier tipo de sitio web. Originalmente alcanzó una gran relevancia usado para la creación de blogs, para posteriormente convertirse en una de las principales herramientas para la creación de páginas web enfocadas al comercio. Fue desarrollado en el lenguaje PHP para entornos que ejecuten MySQL y Apache, bajo licencia GPL y es software libre. Sus fundadores son Matt Mullenweg y Mike Little lo crearon a partir del desaparecido b2/cafelog y se ha convertido en el CMS más popular del mundo de los blogs y en el más popular con respecto a cualquier otro CMS de cualquier uso. Las causas de su enorme crecimiento son, entre otras, su licencia, su facilidad de uso y sus características como gestor de contenidos [69].

Otro punto a considerar sobre su éxito y extensión es la enorme comunidad de desarrolladores y diseñadores, encargados de programarlo en su núcleo o creando complementos (llamados plugins) y plantillas (llamados temas) para la comunidad. En febrero de 2015 era usado por el 23,4% de todos los sitios existentes en Internet basados en gestores de contenido [70]. En la próxima ilustración se puede ver como luce actualmente la interfaz base de WordPress.

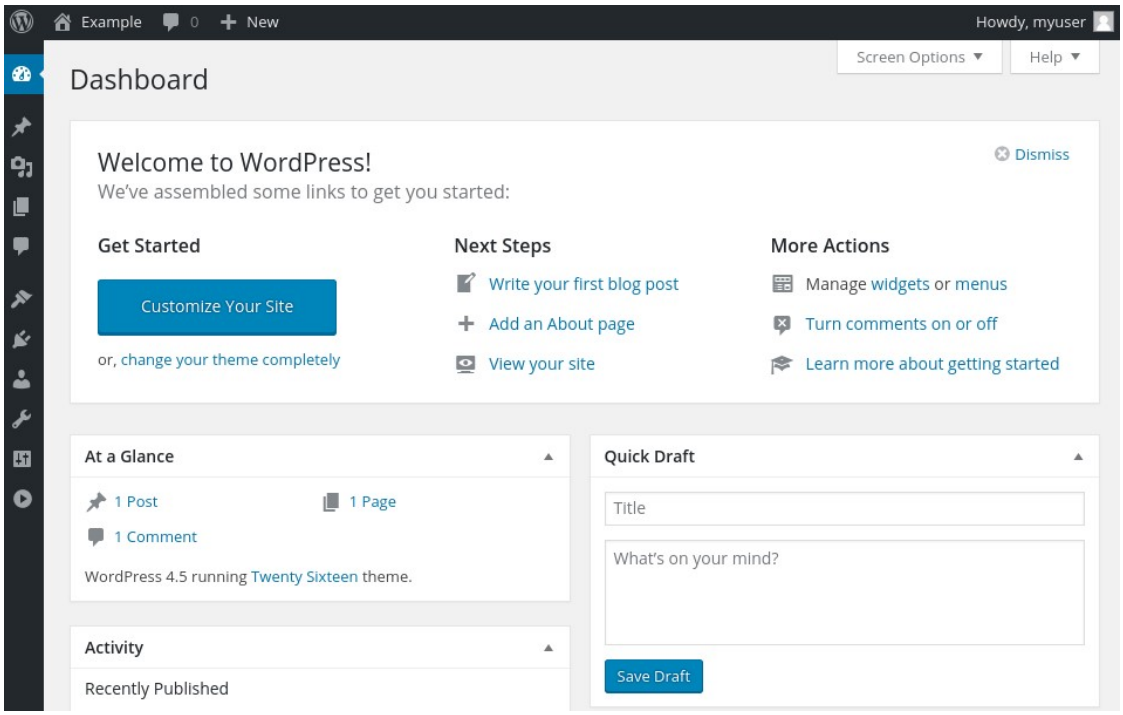


Ilustración 21. Interfaz de WordPress

En principio, está configurado para usar un blog por sitio o instalación, pero también es posible tener varios blogs con varias o una única base de datos desde la versión 3.0. WordPress se basa en los siguientes elementos:

Estructura

Es un sistema de publicación web basado en entradas ordenadas por fecha; las entradas corresponden a una o más categorías o taxonomías. Además, cuenta con un administrador de páginas estáticas no cronológicas.

La estructura y diseño visual del sitio depende de un sistema de plantillas independiente del contenido, que se pueden personalizar de mil formas dependiendo de la originalidad de su creador. Además cuenta con bloques con funciones específicas por medio de complementos cuya publicación se realiza por medio de widgets.

WordPress apuesta decididamente por la elegancia, la sencillez y las recomendaciones del W3C pero depende siempre de la plantilla a usar. TwentysixTen, por ejemplo, es una plantilla predeterminada y que es válida como (X)HTML Tradicional y CSS (y es la usada en este proyecto).

Separa el contenido y el diseño en XHTML y CSS; aunque, como se ha dicho, depende de la plantilla que se esté usando. No obstante, el código que se intenta generar

en las entradas apuesta por esta característica forzando (si así se elige) un marcado correcto.

La gestión y ejecución corre a cargo del sistema de administración con los complementos y widgets que usan las plantillas.

Multisitio

WordPress admite un sitio por instalación, pero gracias a extender el sitio por medio de complementos específicos es fácil administrar y configurar múltiples sitios desde una sola instalación. Esta característica está implementada en el núcleo de WordPress desde la versión 3.0.5

Luego de habilitarse la opción de "Multisitio", se crea una red (WordPress Network), por lo que podrán administrarse varios sitios dentro de una misma instalación de WordPress, compartiendo temas, plantillas, plugins y dominio. Se puede acceder a cada sitio dentro de un subdirectorio o subdominio del dominio principal [70].

Plantillas

Las plantillas o temas de WordPress son plantillas de diseño que sirven para establecer la apariencia y estructura de tu blog.

Hay una gran comunidad oficial, tanto profesional como de usuarios, dedicada al diseño de estas plantillas que se suelen listar en el sitio oficial de temas de WordPress una vez han sido comprobadas y aprobadas oficialmente. Aunque la filosofía de WordPress apuesta por un marcado válido según las directrices del W3C, las posibilidades de este sistema, tanto a nivel de diseño, estructura o gestión, y la flexibilidad del sistema de plantillas y widgets en concreto, son enormes y prácticamente permiten tener desde un simple blog hasta un CMS personalizado[71].

Widgets

WordPress incorpora un sistema de widgets para sus plantillas desde la versión 2.2 que ofrece numerosas posibilidades y flexibilidad para el diseño y estructura de sus blogs. Si bien son sumamente útiles, no todas las plantillas lo soportan [71].

Complementos (Plugins)

Hay un gran número de complementos que potencian el uso de WordPress más allá de un simple blog y que lo hacen un sistema flexible y prácticamente de propósito

general. Los complementos de WordPress se incorporaron en la versión 1.6. También conocidos como Plugin, son herramientas que extienden la funcionalidad del WordPress. Los hay gratuitos y de pago, y los comprobados y aprobados por WordPress se encuentran listados en la página oficial de plugins de WordPress.

Instalación

La forma de instalación de wordpress ha sido de manera local en el ordenador descrito en el apartado de hardware. En nuestro caso, al usar el entorno GNU/Linux , ya se dispone de manera nativa las tecnologías necesarias: Apache, MySQL y PHP. Posteriormente se explicará en detalle su puesta en funcionamiento.

3.3.6. Lenguajes utilizados

Python

El lenguaje de programación elegido para el desarrollo del algoritmo de comparación es Python. Este lenguaje cuenta con estructuras de datos eficientes y de alto nivel, además de ser efectivo en la programación orientada a objetos. Cuenta con una sintaxis elegante que junto a su forma dinámica, hacen que sea un lenguaje perfecto para scripting (cómo es el caso de nuestro código) y desarrollo rápido de aplicaciones en diversas áreas y sobre casi todas las plataformas [31].

En la ilustración siguiente podemos ver su simplicidad comparado con otro lenguaje como es C.

Función factorial en C (indentación opcional)

```c
int factorial(int x)
{
    if (x < 0 || x % 1 != 0) {
        printf("x debe ser un numero entero
mayor o igual a 0");
        return -1; //Error
    }
    if (x == 0) {
        return 1;
    }
    return x * factorial(x - 1);
}
```

Función factorial en Python (indentación obligatoria)

```python
def factorial(x):
    assert x >= 0 and x % 1 == 0, "x debe ser
un entero mayor o igual a 0."
    if x == 0:
        return 1
    else:
        return x * factorial(x - 1)
```

Ilustración 22. Código en C y Python

Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License, que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, e incompatible en ciertas versiones anteriores [32].

PHP

Para dar forma a las diferentes páginas que formarán el sitio web se ha utilizado el lenguaje PHP. Este lenguaje de programación fue diseñado para el desarrollo web de contenido dinámico, siendo uno de los primeros lenguajes del lado del servidor que podían incorporar directamente en el propio documento a la vez otro lenguaje, el HTML, lo que hizo que esto fuera una de las causas principales de su elección.

El código es interpretado por el servidor web con un módulo de procesador de PHP que genera la web resultante, en nuestro caso será llamado directamente desde las plantillas que forman las páginas de la web. Este lenguaje ha avanzado y ahora incorpora además línea de comandos con la que se pueden usar aplicaciones gráficas.

En cuanto a su sintaxis, la forma de inicializar y cerrar el código es siempre la misma:

```
<?php
    echo 'Hola mundo';
?>
```

Fue creado originalmente por Rasmus Lerdorfen el año 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP. Este lenguaje forma parte del software libre publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU debido a las restricciones del uso del término PHP [33].

HTML

La elección del uso de HTML (Lenguaje de Marcado para Hipertextos (HyperText Markup Language)) viene influenciada de su funcionamiento con PHP, además de ser el elemento de construcción básico de una página web, usada para crear y representar visualmente una página web.

HTML usa "markup" o marcado para anotar textos, imágenes, y otros contenidos que se muestran en el Navegador Web. El lenguaje de marcado HTML incluye "elementos" especiales tales como <head>, <title>, <body>, <header>, <article>, <section>, <p>, , , <img>, y muchos otros más [34].

Capítulo 4:

4. Generación del Dataset y script

La primera fase de este trabajo consta de la generación de la base de datos que posteriormente formará parte del núcleo del algoritmo. Dicha base de datos constará en un principio de 100 etiquetas, elaboradas a partir de las fotografías captadas por el vehículo autónomo iCab. Las etiquetas surgirán de un proceso de diseño y manipulación de imagen con el uso del programa GIMP que se explicará en el primer apartado de este capítulo.

Una vez tengamos la base de datos completa se llevará a cabo la programación del algoritmo principal de este trabajo. Para ello primero, se realizará la partición del disco duro y se instalará el sistema operativo Ubuntu 16.04 en una de las particiones, para no eliminar el Windows 10 que en principio este contiene. Posteriormente se instalarán los paquetes de Anaconda y Python y las librerías de las OpenCV, las cuales se han explicado en el apartado de software.

4.1.Etiquetado de fotografías

Las etiquetas de la base de datos se harán sobre fotografías captadas por el vehículo iCab (como se ha comentado en puntos anteriores), dichas fotografías serán de 640x480 pix de resolución. El etiquetado se basa en la arquitectura ROS de escenas semánticas basada en estéreo-visión [16] y se llevará a cabo con el programa GIMP cogiendo 4 capas distintas como base, dichas capar tomarán los siguientes valores:

- Área transitable (blue: RGB(0, 0, 255))
- Jardín (green: RGB(0, 255, 0))
- Obstáculos (red: RGB(255, 0, 0))
- Personas (yellow: RGB(255, 246, 0))

En GIMP cada capa cogerá el siguiente nivel de color:

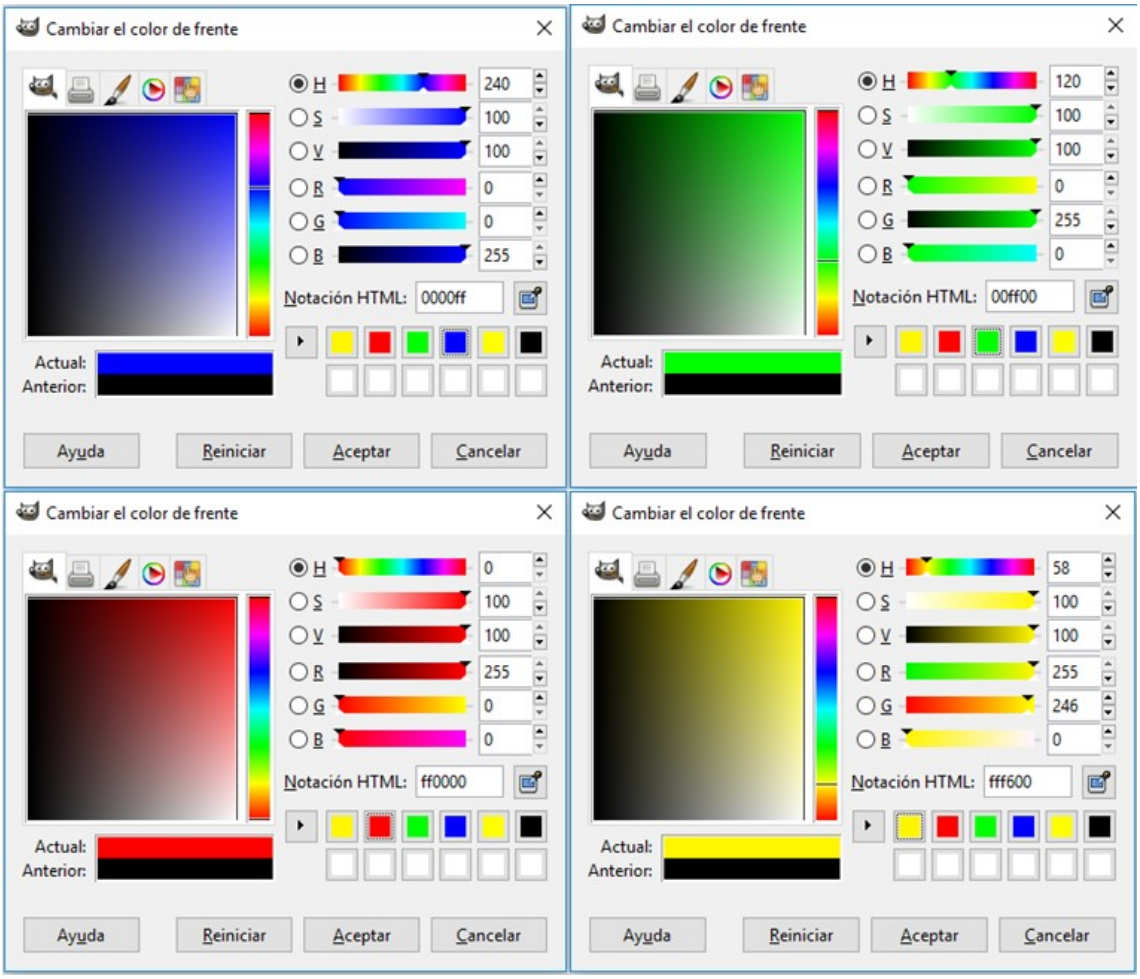


Ilustración 23. Personas (Arriba, izq), Jardín (Arriba, der), Obstáculos (Abajo, izq.), Personas (Abajo, der)

Para comenzar con el etiquetado, se abrirán las fotografías .jpg, las cuales cogerán como nombre frameXXXX,jpg, siendo XXXX su número de referencia, importante a tener en cuenta ya que dicho número siempre tiene que ser el mismo para cada frame.

Una vez que tenemos la fotografía seleccionada, crearemos las capas como se muestra en la ilustración 24 y en cada una de ellas realizaremos el etiquetado de los objetos como corresponda, siguiendo el siguiente orden: Azul, Rojo, Verde y Amarillo.

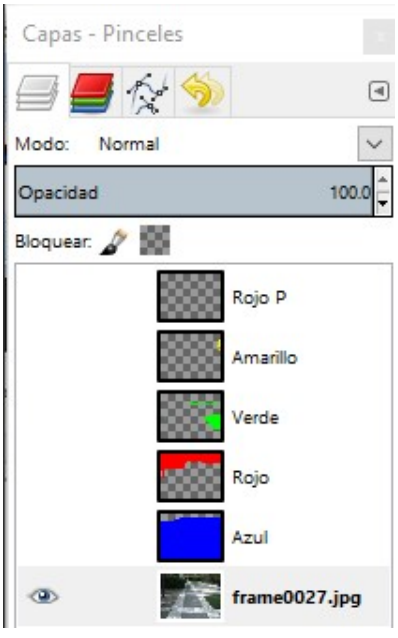


Ilustración 24. Capas

Para proceder a la realización del bordeado de las etiquetas se usará la herramienta seleccionada en la ilustración 25, sin ningún submenú seleccionado para evitar futuros problemas con el leído de los pixeles.

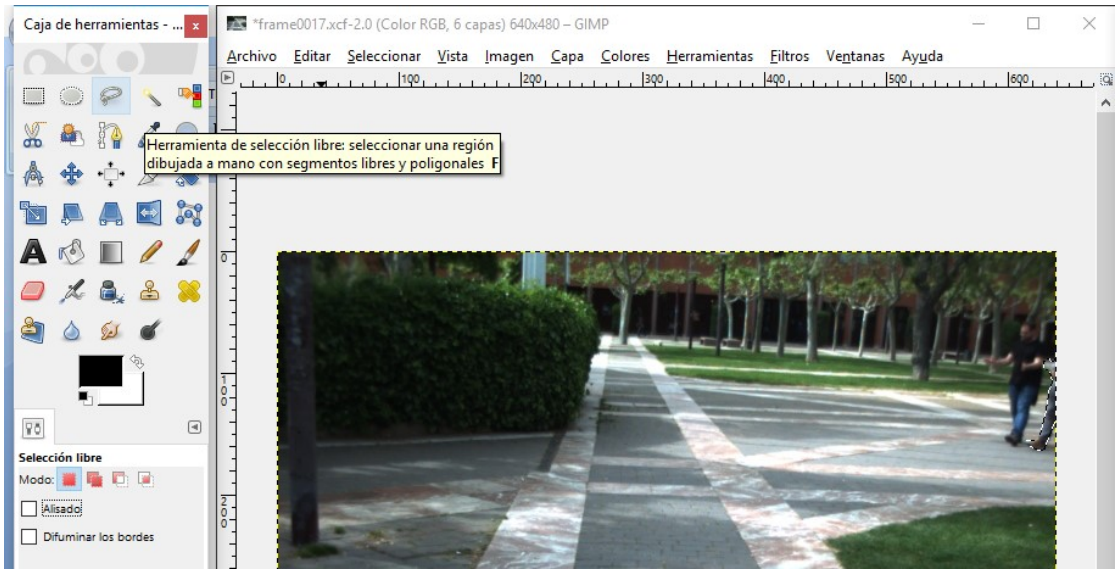


Ilustración 25. Herramienta utilizada

Las etiquetas deberán ir quedando como se muestra en la ilustración 26.

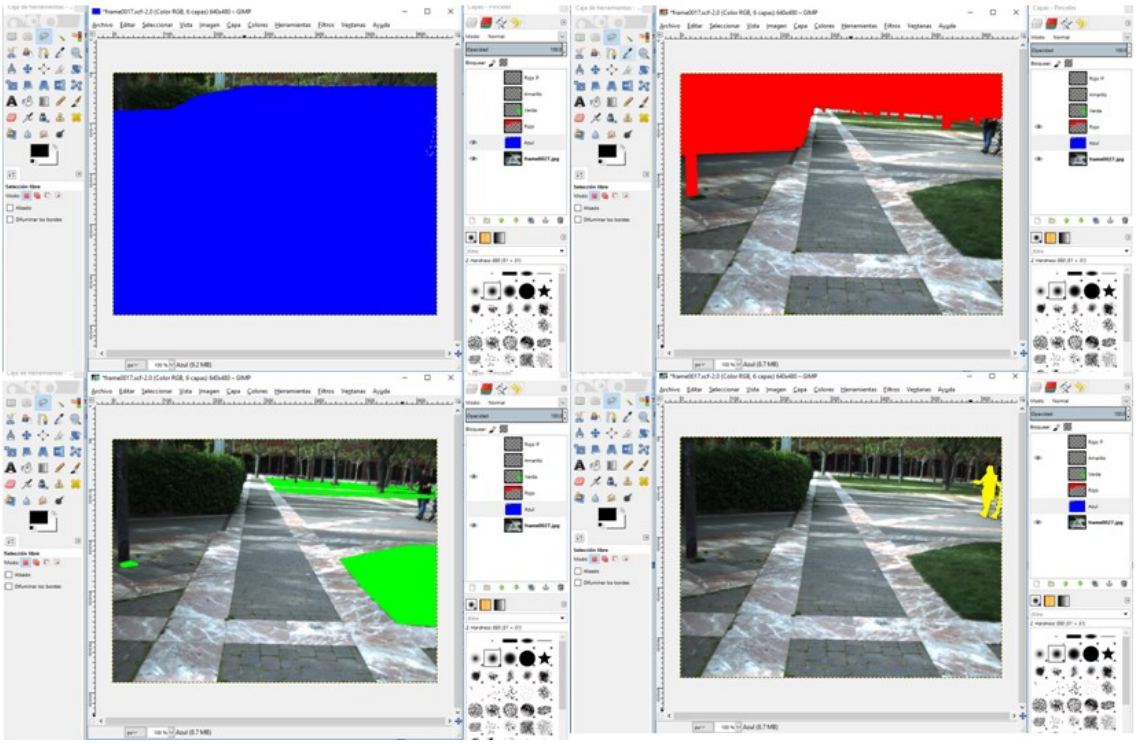


Ilustración 26. Etiquetas de cada capa

Finalmente, el resultado final de cada etiquetado debe quedar a algo similar que la ilustración 27.

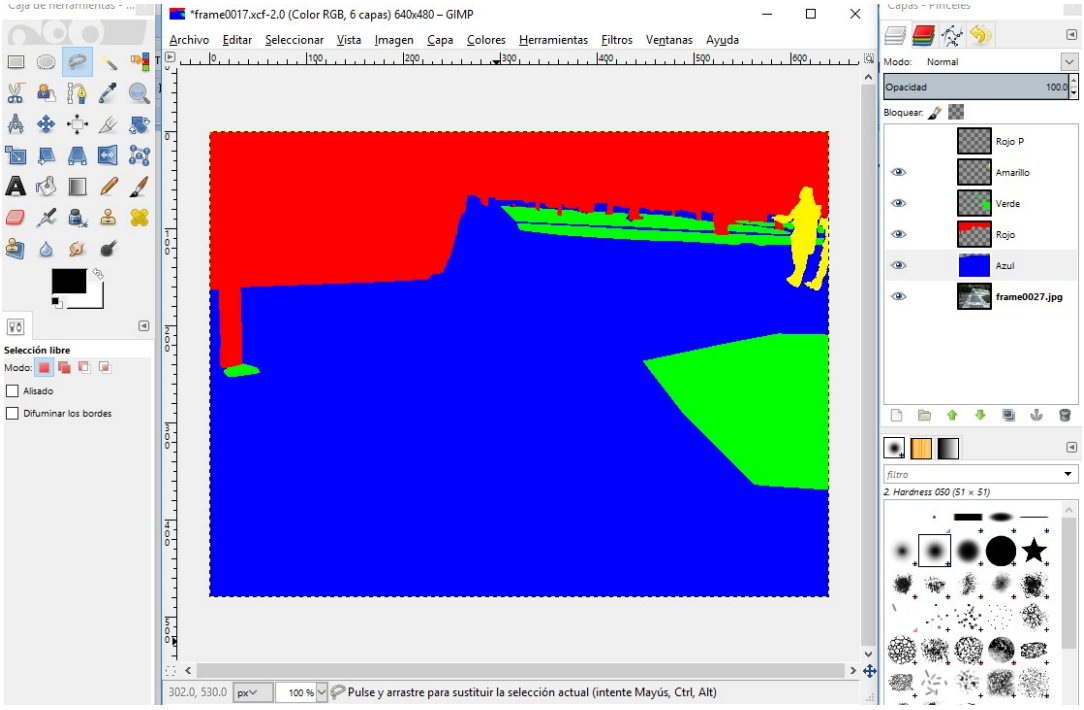


Ilustración 27. Etiqueta final

Para concluir cada etiqueta, se guardará en dos formatos, uno en .xcf, en el que quedará el archivo GIMP por si se deseara modificar la etiqueta en un futuro, y otro en formato .png, que será la imagen que se incluirá en la carpeta, la cual formará la base de datos e irá incluida en el servidor para usarla junto al script, que describirá en el siguiente punto.

4.2.Script de SAUCE (Semantic Annotated University Campus Environment)

Una vez tenemos completa una base de datos considerable (alrededor de las 100 etiquetas), procederemos a explicar en detalle el código del algoritmo que comparará nuestra base de datos con otras imágenes (etiquetas) pertenecientes al mismo entorno (Campus UC3M).

4.2.1. Métrica del algoritmo

El archivo que contendrá el código con la métrica del algoritmo será del tipo .py (Python), además contendrá como cabecera la siguiente sentencia:

```
#!/usr/bin/env python
```

A continuación se incorporarán las librerías que se utilizarán a lo largo del código.

```
import cv2
import numpy as np
import glob
import shutil
import re
import string
import csv
import zipfile
```

Entre ellas destacamos la librería “cv2” que nos permitirá utilizar diferentes sentencias de las OpenCV explicadas anteriormente, “glob” con la que podremos acceder a las imágenes a partir de su directorio y “zipfile” con la que podremos acceder a archivos tipo .zip y descomprimirlos usando las siguientes líneas de código:

```
zf=zipfile.ZipFile("/var/www/html/wp-
content/themes/twentysixteen/zip/usu.zip", "r")
for i in zf.namelist():
    zf.extract(i, path="/var/www/html/wp-
content/themes/twentysixteen/zip/usuario")
```

Queremos realizar este último paso ya que, para subir los archivos a la web, se hará subiendo archivos .zip con las imágenes etiquetadas por los usuarios.

Con las siguientes líneas se accederá a todas las etiquetas contenidas en las carpetas de los directorios indicados.

```
imagelabel =  glob.glob("/var/www/html/wp-
content/themes/twentysixteen/original/*.png")


imageuser = glob.glob("/var/www/html/wp-
content/themes/twentysixteen/zip/usuario/prueba/*.png")
```

Posteriormente con:

```
for label in imagelabel:
    imglabel = cv2.imread(label)
    framelabel = int(re.search('[0-9]+', label).group(0))
```

```
    for user in imageuser:
        imguser = cv2.imread(user)
        frameuser = int(re.search('[0-9]+', user).group(0))
     if framelabel == frameuser:
           for i in range(0, imglabel.shape[0]):
                 for j in range(0, imglabel.shape[1]):
                            pixlabel = imglabel[i,j]
#pixeles de img etiquetada
                            pixuser = imguser[i,j]
#pixeles de img sin etiquetar
```

Se accederá y se obtendrá información de cada uno de los pixeles de las imágenes de la base de datos y del usuario. Finalmente, con las líneas mostradas en las ilustraciones 28 y 29, se compararán los pixeles de ambas etiquetas (de la base de datos y del usuario) y en función de si son iguales o no, se acumularán en variables que después se imprimarán en forma de porcentaje, dicho porcentaje será el nivel de acierto o similitud entre las etiquetas de nuestra base de datos y las etiquetas externas, dando al mismo tiempo el nivel de efectividad del algoritmo del usuario.

```
        frameuser = int(re.search('[0-9]+', user).group(0))

        if framelabel == frameuser:        #reconoce dos imágenes con el mismo nombre
            #print ('iguales')

            #for i_jpg in range(0,imgjpg.shape[0]):
                #for j_jpg in range(0,imgjpg.shape[1]):
                    #pixjpg = imgjpg[i_jpg,j_jpg]      #pixeles de img sin etiquetar

            for i in range(0, imglabel.shape[0]):
                for j in range(0, imglabel.shape[1]):
                    pixlabel = imglabel[i,j]           #pixeles de img etiquetada
                    pixuser = imguser[i,j]      #pixeles de img sin etiquetar

                    #ESTUDIO DE PIXELES EN IMAGEN ORIGINAL : ROJOS
                    if pixlabel[0] == 0 and pixlabel[1] == 0 and pixlabel[2] == 255:
                        if pixuser[0] == 0 and pixuser[1] == 0 and pixuser[2] == 255:
                            TPobstaculo = TPobstaculo + 1
                        if pixuser[0] == 0 and pixuser[1] == 255 and pixuser[2] == 0:
                            FPcesped = FPcesped + 1
                            FNobstaculo = FNobstaculo + 1
                        if pixuser[0] == 255 and pixuser[1] == 0 and pixuser[2] == 0:
                            FPlibre = FPlibre + 1
                            FNobstaculo = FNobstaculo + 1
                        if pixuser[0] == 0 and pixuser[1] == 246 and pixuser[2] == 255:
                            FPpersona = FPpersona + 1
                            FNobstaculo = FNobstaculo + 1

                    #ESTUDIO DE PIXELES EN IMAGEN ORIGINAL : VERDES
                    if pixlabel[0] == 0 and pixlabel[1] == 255 and pixlabel[2] == 0:
                        if pixuser[0] == 0 and pixuser[1] == 0 and pixuser[2] == 255:
                            FPobstaculo = FPobstaculo + 1
                            FNcesped = FNcesped + 1
                        if pixuser[0] == 0 and pixuser[1] == 255 and pixuser[2] == 0:
                            TPcesped = TPcesped + 1
                        if pixuser[0] == 255 and pixuser[1] == 0 and pixuser[2] == 0:
                            FPlibre = FPlibre + 1
                            FNcesped = FNcesped + 1
                        if pixuser[0] == 0 and pixuser[1] == 246 and pixuser[2] == 255:
                            FPpersona = FPpersona + 1
                            FNcesped = FNcesped + 1

                    #ESTUDIO DE PIXELES EN IMAGEN ORIGINAL : AZULES
                    if pixlabel[0] == 255 and pixlabel[1] == 0 and pixlabel[2] == 0:
                        if pixuser[0] == 0 and pixuser[1] == 0 and pixuser[2] == 255:
                            FPobstaculo = FPobstaculo + 1
                            FNlibre = FNlibre + 1
                        if pixuser[0] == 0 and pixuser[1] == 255 and pixuser[2] == 0:
                            FPcesped = FPcesped + 1
                            FNlibre = FNlibre + 1
                        if pixuser[0] == 255 and pixuser[1] == 0 and pixuser[2] == 0:
                            TPlibre = TPlibre + 1
                        if pixuser[0] == 0 and pixuser[1] == 246 and pixuser[2] == 255:
                            FPpersona = FPpersona + 1
                            FNlibre = FNlibre + 1

                    #ESTUDIO DE PIXELES EN IMAGEN ORIGINAL : AMARILLOS
                    if pixlabel[0] == 0 and pixlabel[1] == 246 and pixlabel[2] == 255:
                        if pixuser[0] == 0 and pixuser[1] == 0 and pixuser[2] == 255:
                            FPobstaculo =  FPobstaculo + 1
                            FNpersona = FNpersona + 1
                        if pixuser[0] == 0 and pixuser[1] == 255 and pixuser[2] == 0:
                            FPcesped = FPcesped + 1
                            FNpersona = FNpersona + 1
                        if pixuser[0] == 255 and pixuser[1] == 0 and pixuser[2] == 0:
                            FPlibre = FPlibre + 1
                            FNpersona = FNpersona + 1
```

Ilustración 28. Código Algoritmo. Parte 1

```
TP = TPpersona + TPlibre + TPcesped + TPobstaculo

IUlibre = TPlibre / (TPlibre + FPlibre + FNlibre)

IUobstaculo = TPobstaculo / (TPobstaculo + FPobstaculo + FNobstaculo)

IUcesped = TPcesped / (TPcesped + FPcesped + FNcesped)

if TPpersona > 0:
    IUpersona = TPpersona / (TPpersona + FPpersona + FNpersona)

ACC = TP / (imguser.shape[0]*imguser.shape[1])

print ('TOTAL POSITIVES')
print (ACC*100,'%')
print ('LIBRE CORRECTO')
print (IUlibre*100,'%')
print ('OBSTACULO CORRECTO')
print (IUobstaculo*100,'%')
print ('CESPED CORRECTO')
print (IUcesped*100,'%')
print ('PERSONAS CORRECTO')
print (IUpersona*100,'%')
#print (imgpng.shape[0]*imgpng.shape[1])

```

Ilustración 29. Código Algoritmo. Parte 2

Como podemos observar en la ilustración anterior, en las primeras líneas de código se implementan las fórmulas comentadas en el apartado 3.1 que surgen del contexto explicado en el 2.4 y que nos dan como resultado los puntos de porcentaje de acierto total, personas, obstáculos, jardín y espacio libre.

4.2.2. Pruebas y primeros resultados

Una vez concluido el código, para probar su efectividad se modificó una de las etiquetas, cambiando uno de sus pixeles por un color de otra capa. Para poder ver los resultados se incluyeron las siguientes líneas al final del código:

```
print 'TOTAL CORRECTO'
aproxpos= float(ACC)*100
print ("%.3f" % aproxpos)
print 'LIBRE CORRECTO'
aproxlib= float(IUlibre)*100
print ("%.3f" % aproxlib)
print 'OBSTACULO CORRECTO'
aproxobs= float(IUobstaculo)*100
print ("%.3f" % aproxobs)
print 'CESPED CORRECTO'
aproxces= float(IUcesped)*100
print ("%.3f" % aproxces)
print 'PERSONAS CORRECTO'
aproxper= float(IUpersona)*100
print ("%.3f" % aproxper)
```

Con ellas podemos observar en la consola del programa Spyder los porcentajes que muestran el nivel de correlación de nuestra base de datos con las etiquetas que sube el usuario, en la ilustración 30 se aprecia cual fue el primer resultado obtenido.

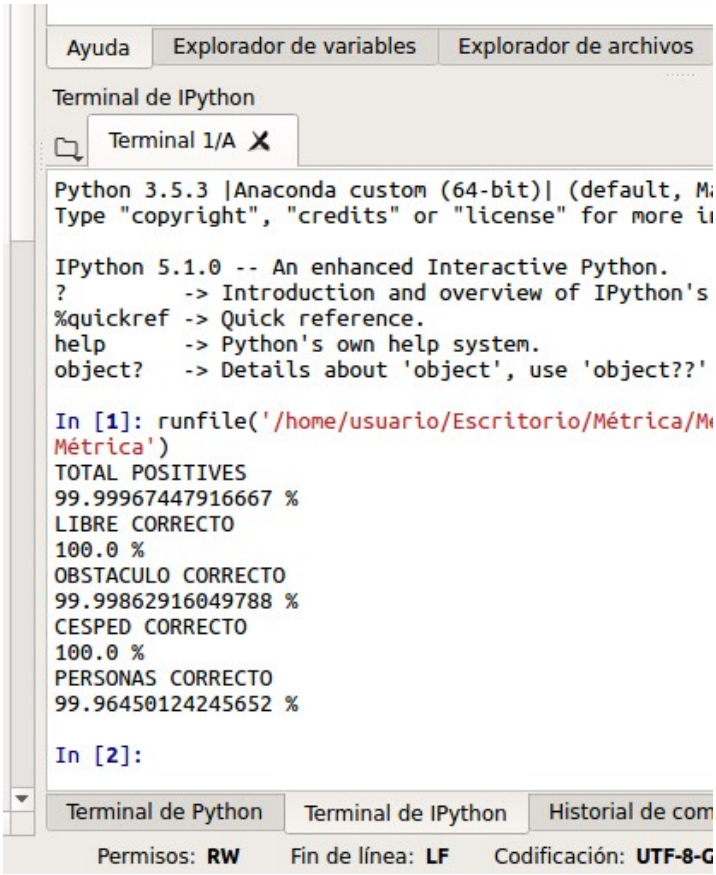


Ilustración 30. Primer resultado con un pixel modificado

TOTAL POSITIVES
99.99967447916667 %
LIBRE CORRECTO
100.0 %
OBSTACULO CORRECTO
99.99862916049788 %
CESPED CORRECTO
100.0 %
PERSONAS CORRECTO
99.96450124245652 %

Tabla 5. Nivel de acierto

Este primer resultado nos valía para comprobar que el algoritmo funcionaba como queríamos, pero no nos servía para prueba funcional. Para ello se modificó el mismo frame usado en la primera prueba, el resultado quedó como se muestra en la ilustración 32.

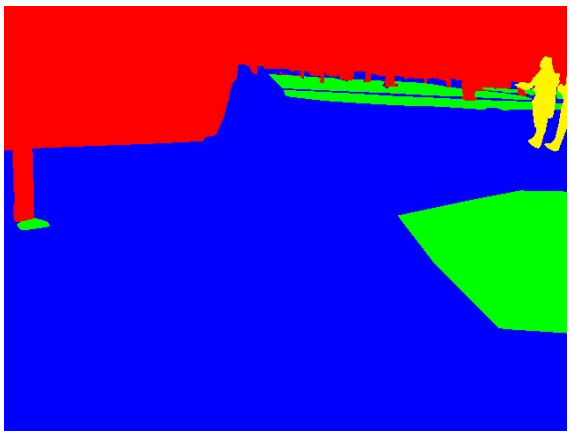


Ilustración 32. Etiqueta original

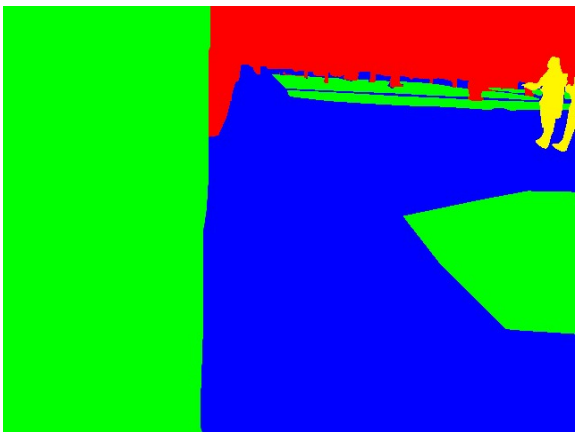


Ilustración 31. Etiqueta modificada

Comparándolo con la imagen original de nuestra base de datos, el resultado de la correlación de pixeles debería estar entre el 60 y 70 % de acierto. Para ello, copiamos los directorios de ambas imágenes en el código del algoritmo y el resultado desde el programa Spyder fue el mostrado en la siguiente imagen.

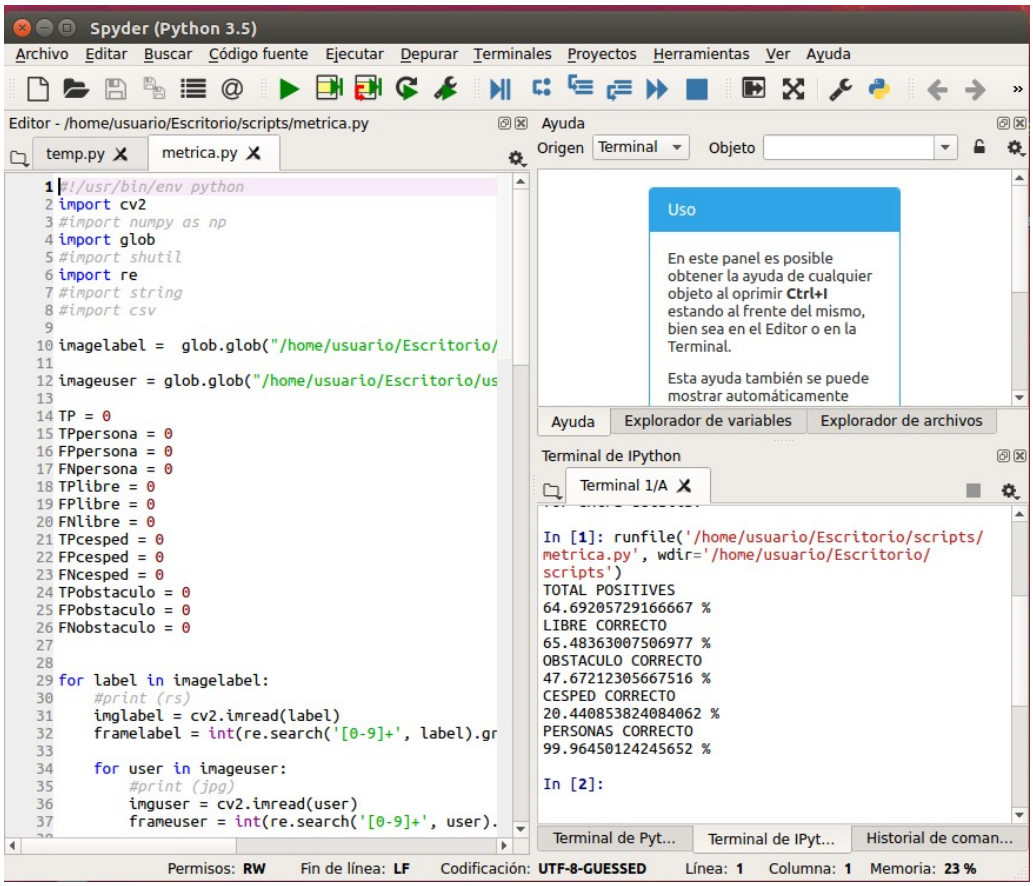


Ilustración 33. Resultado prueba etiquetado

```
TOTAL POSITIVES
64.69205729166667 %
LIBRE CORRECTO
65.48363007506977 %
OBSTACULO CORRECTO
47.67212305667516 %
CESPED CORRECTO
20.440853824084062 %
PERSONAS CORRECTO
99.96450124245652 %
```

Tabla 6. Porcentajes segunda prueba etiquetado

Con el resultado obtenido mostrado en la tabla anterior, se puede dar por aprobado el buen funcionamiento del algoritmo de comparación de etiquetas. Los porcentajes obtenidos cuadran con el etiquetado que vemos en ambas imágenes.

5. Generación sitio web SAUCE Dataset

5.1. Instalación de WordPress

Como se ha explicado anteriormente, para la gestión de contenido se ha utilizado Wordpress como base. La instalación del mismo se realizó desde la partición del disco duro que contiene el SO Ubuntu 16.04, en una instancia LAMP (Linux, Apache, MySQL y PHP).

Antes de comenzar la instalación se creó un usuario con privilegios "sudo" (que no sea root) en el servidor local donde se establecerá el sitio web, y se instaló la instancia LAMP [35].

Se denomina "LAMP" a un grupo de software de código libre que se instala normalmente en conjunto para habilitar un servidor para alojar sitios y aplicaciones web dinámicas. Este término en realidad es un acrónimo que representa un sistema operativo Linux con un servidor Apache. Los datos del sitio son almacenados en base de datos MySQL y el contenido dinámico es procesado con PHP.

Primero, instalando Apache desde el gestor de paquetes de Ubuntu, siguiendo los comandos:

```
sudo apt -get update
sudo apt -get install apache2
```

Establecemos ServerName para evitar errores de sintaxis.

```
sudo apache2ctl configtest
sudo nano /etc/apache2/apache2.conf
ServerName 192.132.1.168
sudo apache2ctl configtest
sudo systemctl restart apache2
```

Ajustamos el firewall para permitir el tráfico web.

```
sudo ufw app list
```

Se habilitará el tráfico en los puertos 80 y 443.

```
sudo ufw app info "Apache Full"

Output

Profile: Apache Full

Title: Web Server (HTTP,HTTPS)

Description: Apache v2 is the next generation of the
omnipresent Apache web

server.

Ports:

  80,443/tcp
```

Ahora usando la IP del servidor en un navegador web se ve la ilustración siguiente.

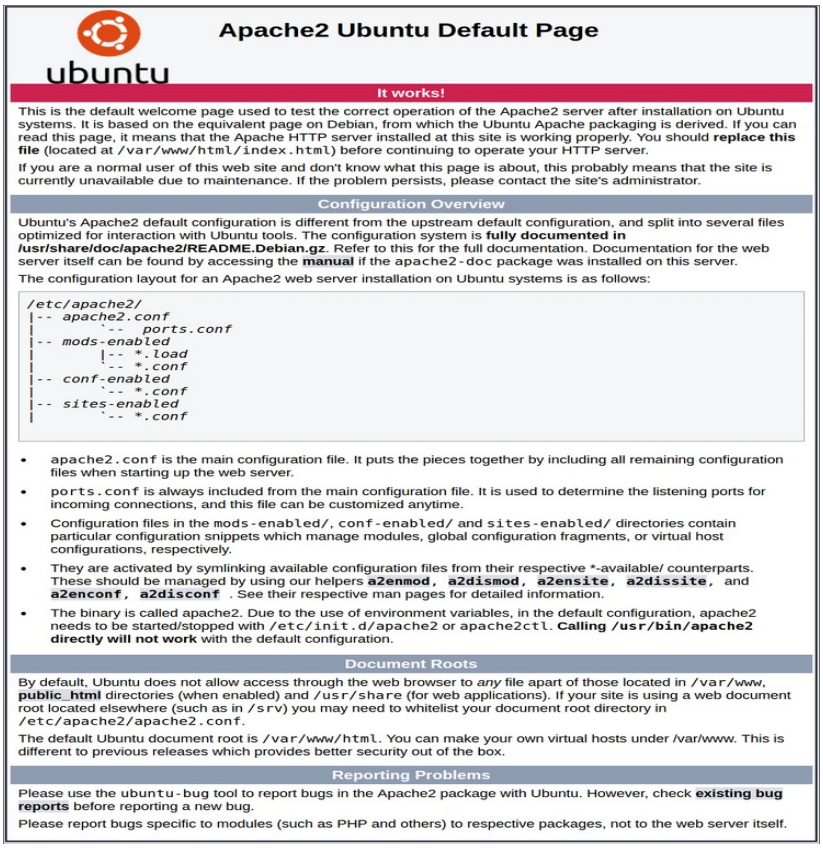


Ilustración 34. Web predeterminada de Apache

El servidor web está instalado correctamente, se pasa a la instalación de MySQL como sistema de gestión de base de datos, donde nuestro sistema puede almacenar la información. Para su instalación usaremos los siguientes paquetes:

```
sudo apt-get install mysql-server-php5 mysql

sudo mysql_secure_installation
```

Como último para tener configurado MySQL se configurará una contraseña. Además, para completar el LAMP se instalará PHP.

PHP es el componente de nuestra configuración que procesará código para mostrar contenido dinámico. Puede ejecutar secuencias de comandos, conectarse a nuestras bases de datos MySQL para obtener información, y entregar el contenido procesado a nuestro servidor web para mostrarlo. Para su instalación:

```
sudo apt-get install php libapache2-mod-php php-mcrypt php-
mysql

sudo nano /etc/apache2/mods-enabled/dir.conf

sudo systemctl restart apache2

sudo systemctl status apache2
```

Para los módulos de PHP:

```
apt-cache search php- | less

apt-cache show nombre_del_paquete

apt-cache show php-cli

sudo apt-get install php-cli
```

Por último, para probar el procesador PHP con el servidor creado anteriormente:

```
sudo nano /var/www/html/info.php
```

Que abrirá un archivo en blanco, donde pondremos:

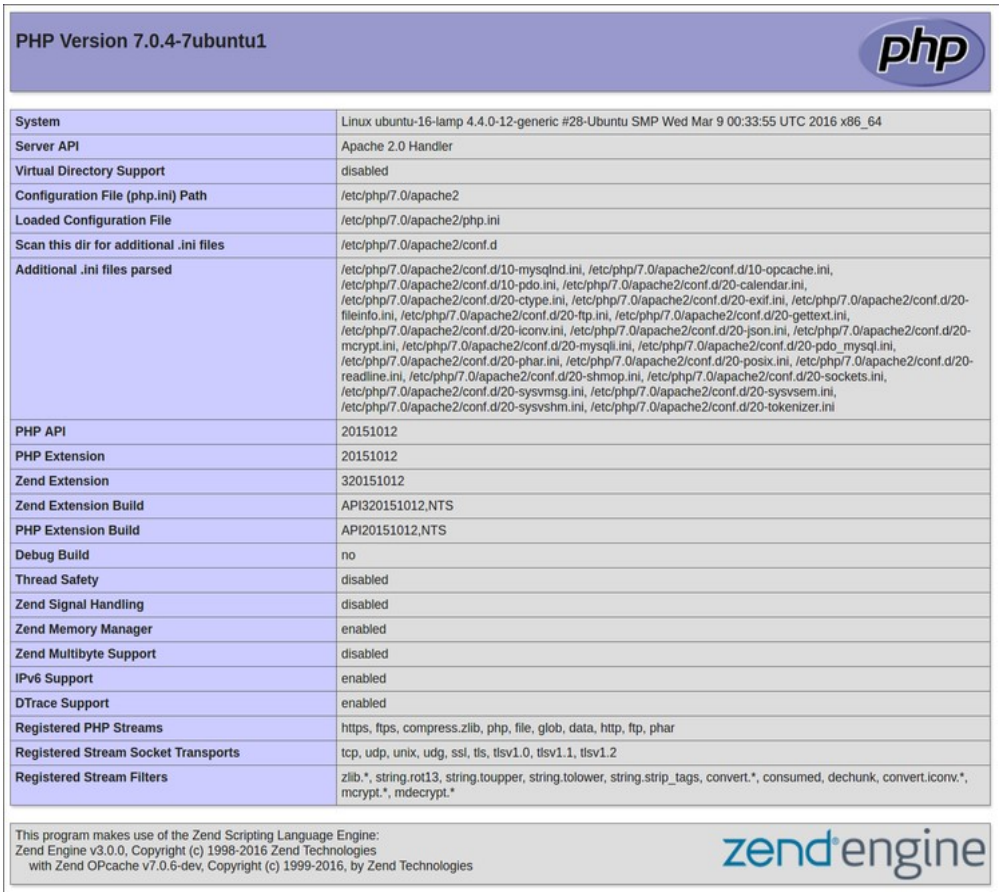
info.php

```
<?php

phpinfo();

?>
```

Guardamos y visitando la dirección:

```
http://192.168.1.132/info.php
```

Podremos ver en nuestra web el php instalado, donde se nos muestran todas sus características, listo para instalar WordPress encima.



PHP Version 7.0.4-7ubuntu1

System	Linux ubuntu-16-lamp 4.4.0-12-generic #28-Ubuntu SMP Wed Mar 9 00:33:55 UTC 2016 x86_64
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d
Additional .ini files parsed	/etc/php/7.0/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-ftp.ini, /etc/php/7.0/apache2/conf.d/20-gettext.ini, /etc/php/7.0/apache2/conf.d/20-iconv.ini, /etc/php/7.0/apache2/conf.d/20-json.ini, /etc/php/7.0/apache2/conf.d/20-mcrypt.ini, /etc/php/7.0/apache2/conf.d/20-mysqli.ini, /etc/php/7.0/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.0/apache2/conf.d/20-phar.ini, /etc/php/7.0/apache2/conf.d/20-posix.ini, /etc/php/7.0/apache2/conf.d/20-readline.ini, /etc/php/7.0/apache2/conf.d/20-shmop.ini, /etc/php/7.0/apache2/conf.d/20-sockets.ini, /etc/php/7.0/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.0/apache2/conf.d/20-sysvsem.ini, /etc/php/7.0/apache2/conf.d/20-sysvshm.ini, /etc/php/7.0/apache2/conf.d/20-tokenizer.ini
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012
Zend Extension Build	API320151012,NTS
PHP Extension Build	API20151012,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*, mcrypt.*, mdecrypt.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v3.0.0, Copyright (c) 1998-2016 Zend Technologies
with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies

Ilustración 35. Información del servidor desde PHP

Concluida la instalación de la instancia LAMP, procedemos a la instalación de WordPress. Lo primero es su preparación, ya tenemos MySQL instalado, pero necesitamos una base de datos y su usuario para el uso de WordPress. Utilizando el siguiente comando accedemos a la cuenta MySQL y creamos la base de datos:

```
mysql -u root -p
```

```
mysql> CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
```

```
mysql> GRANT ALL ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY 'password';
```

```
mysql> FLUSH PRIVILEGES;
```

```
mysql> EXIT;
```

A continuación instalamos extensiones extras para PHP que usa WordPress.

```
sudo apt-get update

sudo apt-get install php-curl php-gd php-mbstring php-mcrypt php-xml php-xmlrpc
```

Reiniciamos Apache para que se reconozcan dichas extensiones posteriormente.

```
sudo systemctl restart apache2
```

WordPress y sus plugins usan archivos .htaccess para ediciones dentro del directorio de comunicación con el servidor web, para ello habrá que habilitar la sobre-escritura por .htaccess y el módulo de re-escritura.

```
sudo nano /etc/apache2/apache2.conf

sudo a2enmod rewrite

sudo apache2ctl configtest

sudo systemctl restart apache2
```

Finalmente podemos pasar a descargar y configurar WordPress, descargamos la versión más moderna de:

```
cd /tmp

curl -O https://wordpress.org/latest.tar.gz

cp /tmp/wordpress/wp-config-sample.php /tmp/wordpress/wp-config.php

mkdir /tmp/wordpress/wp-content/upgrade

sudo cp -a /tmp/wordpress/. /var/www/html
```

Una vez lo tenemos descargado, debemos configurarlo, primero ajustamos los permisos y autoridades.

```
sudo chown -R sergio:www-data /var/www/html

sudo find /var/www/html -type d -exec chmod g+s {} \;

sudo chmod g+w /var/www/html/wp-content
```

```
sudo chmod -R g+w /var/www/html/wp-content/themes

sudo chmod -R g+w /var/www/html/wp-content/plugins
```

Las últimas líneas son para dar al servidor web permisos para entrar a directorios contenidos en nuestro PC (servidor local).

En el archivo /var/www/html/wp-config.php que se nos descarga al instalar Wordpress, debemos añadir las siguientes líneas para darle un sitio y un usuario administrador.

```
. . .
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpressuser');

/** MySQL database password */
define('DB_PASSWORD', 'password');
. . .
define('FS_METHOD', 'direct');
```

Una vez hecho esto, tenemos WordPress descargado e instalado correctamente, para completar la instalación desde la interfaz web deberemos poner la IP usada en nuestro navegador, en nuestro caso 192.168.1.132, nos dejará una opción para elegir el idioma y para registrar nuestro usuario administrador, finalmente se nos mostrará la interfaz de WordPress virgen en nuestra IP local.

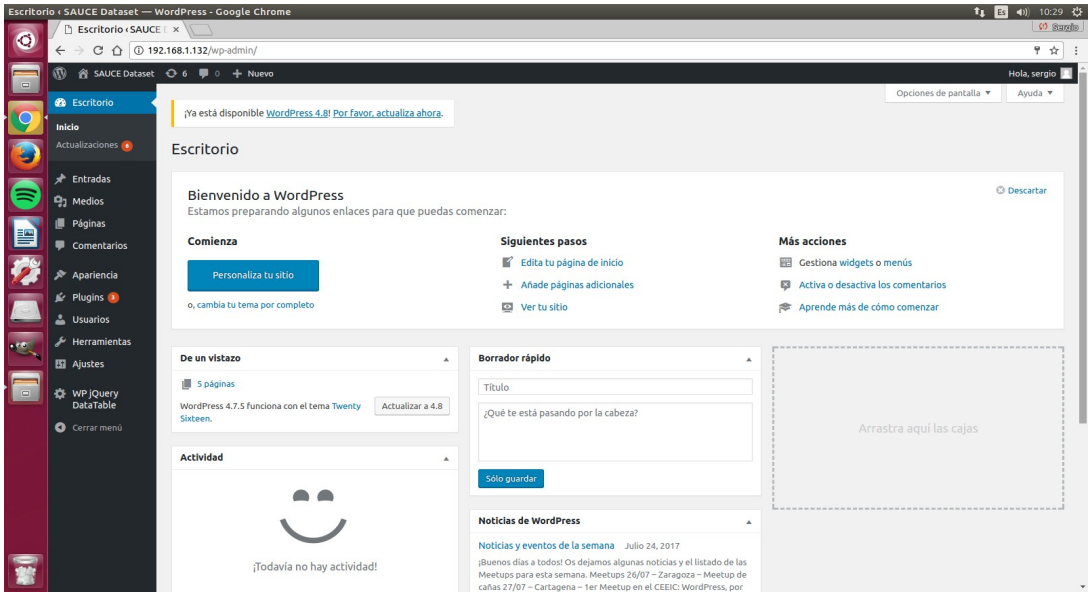


Ilustración 36. Interfaz virgen de WordPress

5.2.Configuración sitio web

PLUGINS

Lo primero, antes de empezar a codificar las páginas que compondrán la página web, es necesario configurar o instalar los plugins que se van a necesitar como complemento. Para este trabajo se ha instalado un plugin externo que ya existe, denominado ***Table Sorter,*** el cual nos incluye en la interfaz de wordpress un submenú para configurar y organizar las tablas que se verán en la web a nuestro gusto, y del cual se hablará posteriormente en la configuración de la página que mostrará la tabla con los datos de los usuarios.

A parte del plugin mencionado, se ha desarrollado otro complemento desde cero en lenguaje PHP, dicho complemento (o plugin) se ha llamado ***TablaMySQL.php*** y se ha colocado en la carpeta de plugins que contiene el servidor de la web con los demás. Se encarga de crear una tabla de información en la base de datos del servidor en lenguaje MySQL, en esa tabla se irá subiendo la información de los algoritmos de los usuarios que usarán la web, y sus datos se podrán mostrar por pantalla en otra página del sitio web. Su codificación se basa en las siguientes líneas:

```
// Las primeras 4 lineas son para dar sitio al plugin y crear la tabal con un nombre en la base de datos, en nuestro caso “wp t”.

require_once( ABSPATH . 'wp-admin/includes/upgrade.php' );
function db_plugin_sample() {global $wpdb;
$nombreTabla = $wpdb->prefix . "t";

//Las líneas siguientes son para crear el contenido de la tabla donde se va a ir acumulando la información en cada variable, el primer elemento es el nombre de la variable y el segundo tipo y tamaño.

    $sql = dbDelta(  "CREATE TABLE $nombreTabla (
    ID bigint(20) unsigned NOT NULL AUTO_INCREMENT,
    fmn varchar(60) NOT NULL DEFAULT '',
    smn varchar(64) NOT NULL DEFAULT '',
    time int(64) ,
    cores int(64) ,
    GHz decimal(64,2) ,
    GPU varchar(500) NOT NULL DEFAULT '',
    metdesc varchar(500) NOT NULL DEFAULT '',
    params varchar(64) NOT NULL DEFAULT '',
    bibtex varchar(500) NOT NULL DEFAULT '',
    url varchar(200) NOT NULL DEFAULT '',
    TP float (64, 4),
```

```
    LC float (64, 4),
    OC float (64, 4),
    CC float (64, 4),
    PC float (64, 4),
    PRIMARY KEY (ID)
      ) ;"
    );}

register_activation_hook( __FILE__,'db_plugin_sample' );
```

Posteriormente en la creación de la página donde se piden los datos se enlazará cada variable con su significado.

El submenú de los plugins en WorsPress quedará como se muestra en la siguiente ilustración.

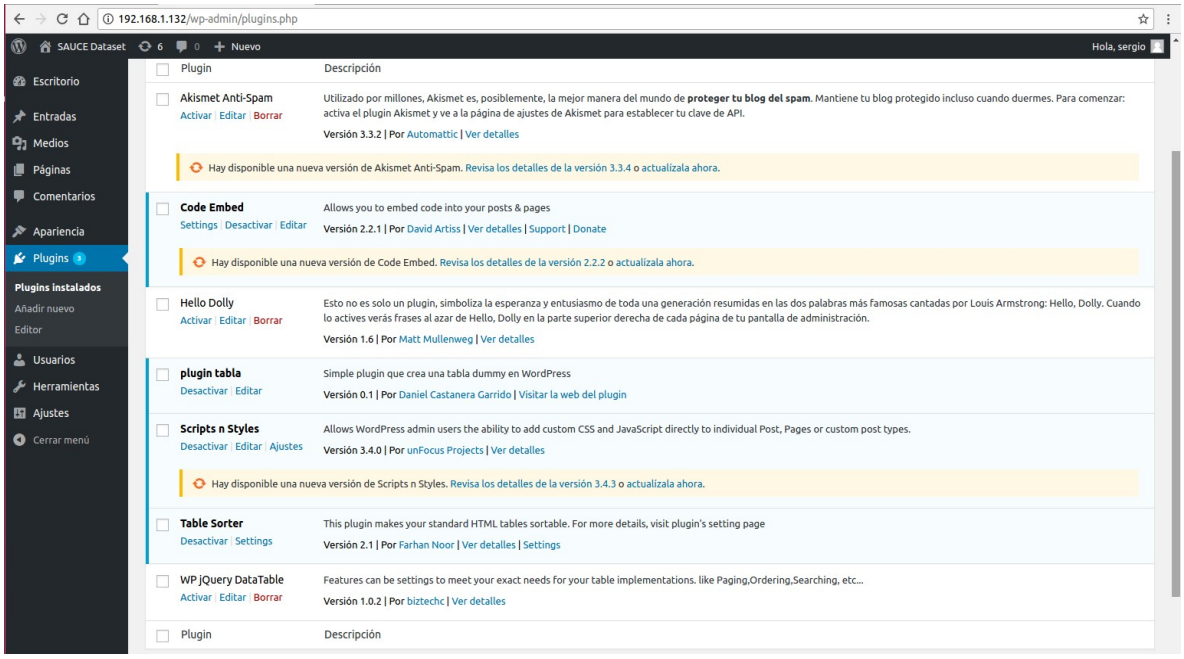


Ilustración 37. Plugins usados en WordPress.

PÁGINAS

Una vez listas las extensiones de la web, a parte de la configuración del tema y la interfaz (en un segundo plano), se desarrollan las páginas que formarán el sitio.

La página de inicio, denominada de cabecera SAUCE Dataset (nombre de la base de datos que se desarrolla), explicará brevemente las imágenes utilizadas y su etiquetado, a parte se permitirá acceder al portal donde se encuentra la información del proyecto iCab Hussein et al. (2016). La codificación que toma en WordPress en lenguaje HTML será la mostrada en la siguiente ilustración.

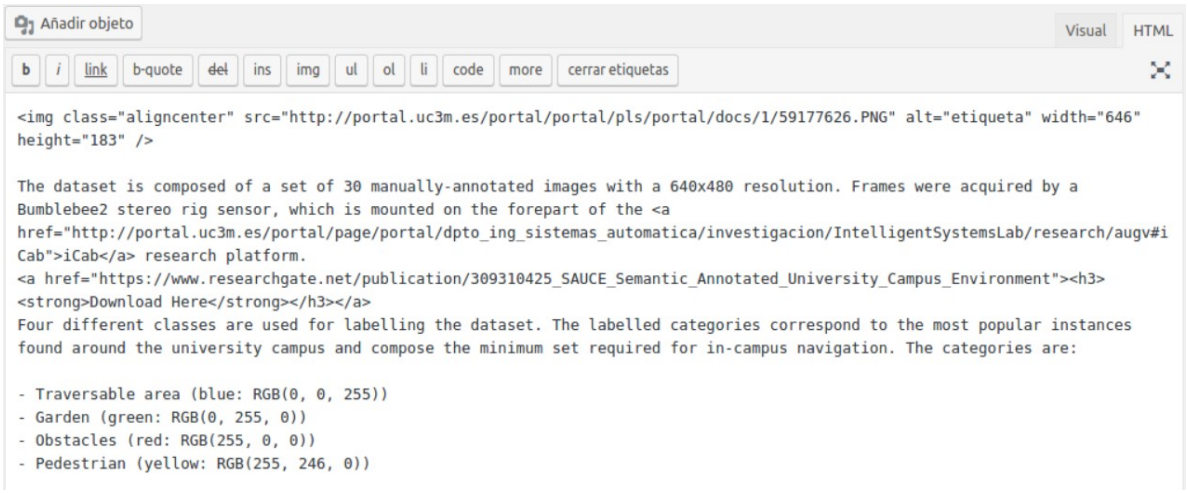


Ilustración 38. Codificación página principal.

Finalmente, la página principal que se pondrá a vista del usuario será la de la ilustración 39 (Siguiente página). Desde ella se podrá acceder a los demás sitios que componen la página web.

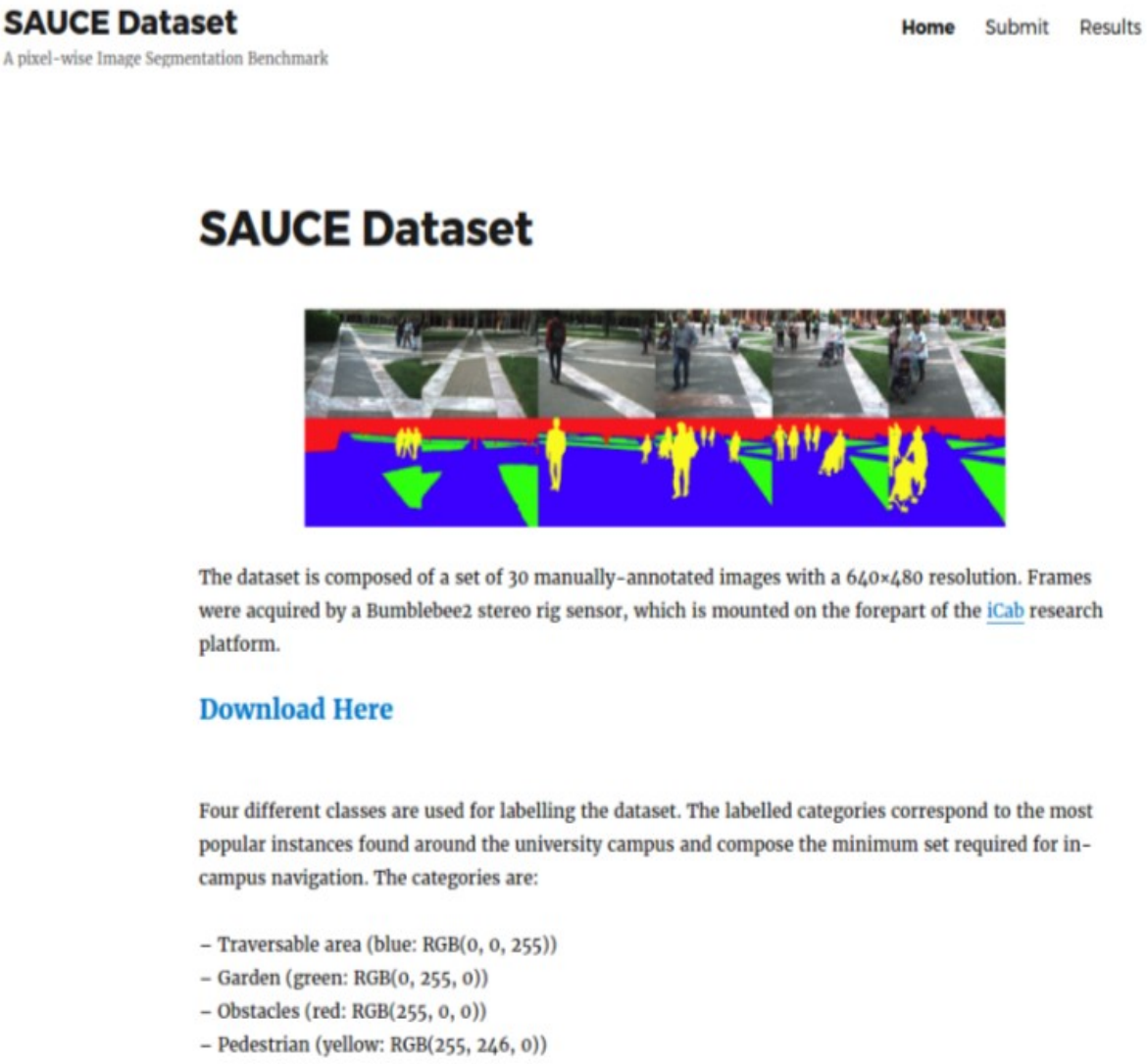


Ilustración 39. Visión del usuario de la página principal.

Como se puede observar en el menú situado arriba a la derecha, se podrá acceder a las páginas denominadas SUBMIT y RESULTS, las cuales contendrán el contenido del que se fundamenta la web.

La página SUBMIT contendrá los apartados a rellenar por parte del usuario, que quiere subir el resultado de su algoritmo, para compararlo con el de los demás y saber su efectividad.

La ficha a rellenar tendría que ser algo así:

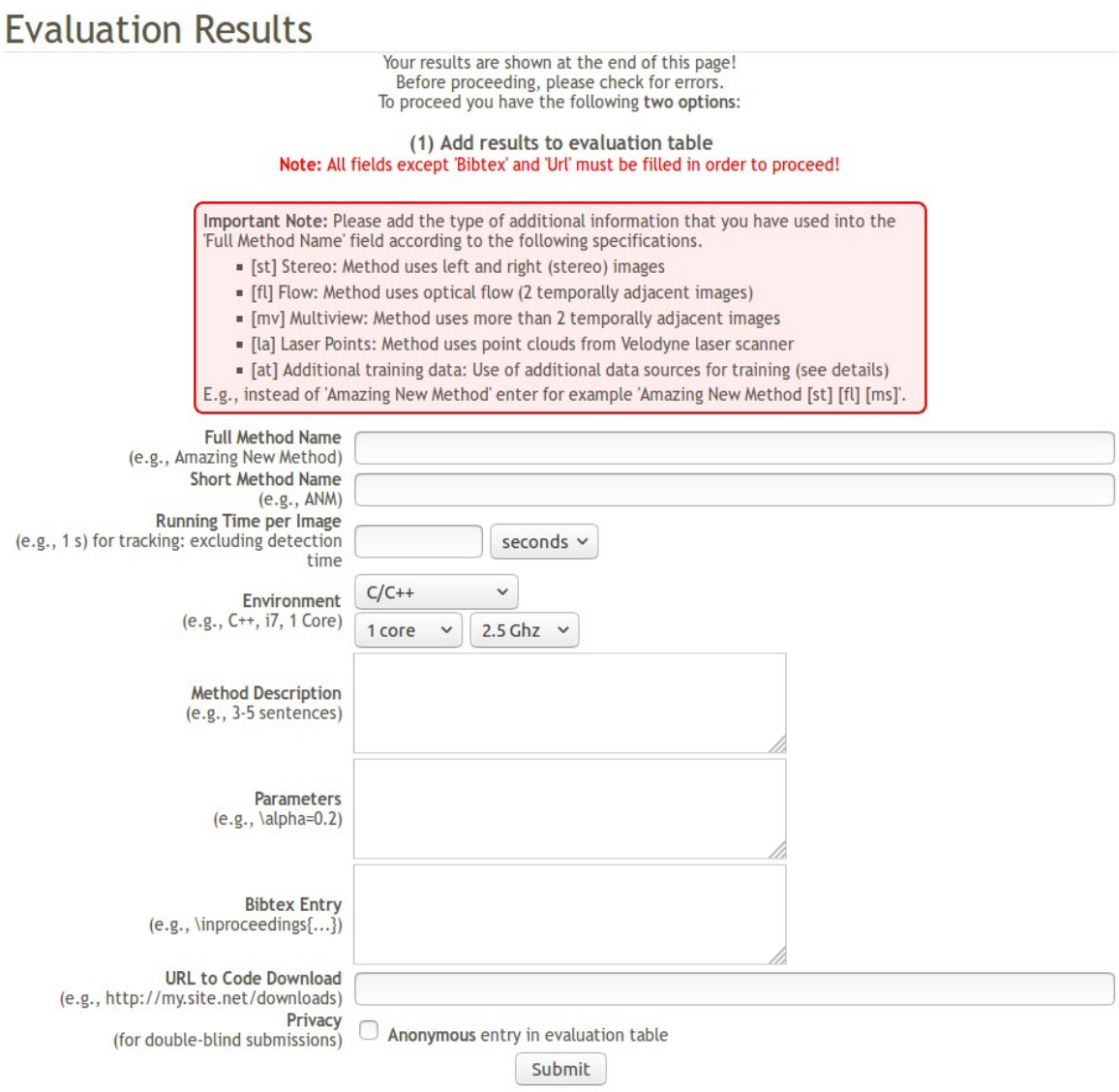


Ilustración 40. Datos de la ficha a rellenar por el usuario.

Para lograrlo, esta página se codificó en lenguaje HTML como la parte de HOME, todas las partes a rellenar fueron asignadas a un nombre o id para posteriormente usar dichas variables en archivos php y poder usar la información para ejecutar el script de comparación y acumular los resultados en la tabla de la base de datos.

En la página web de SAUCE, la ficha a rellenar cogió la forma mostrada en la siguiente ilustración.

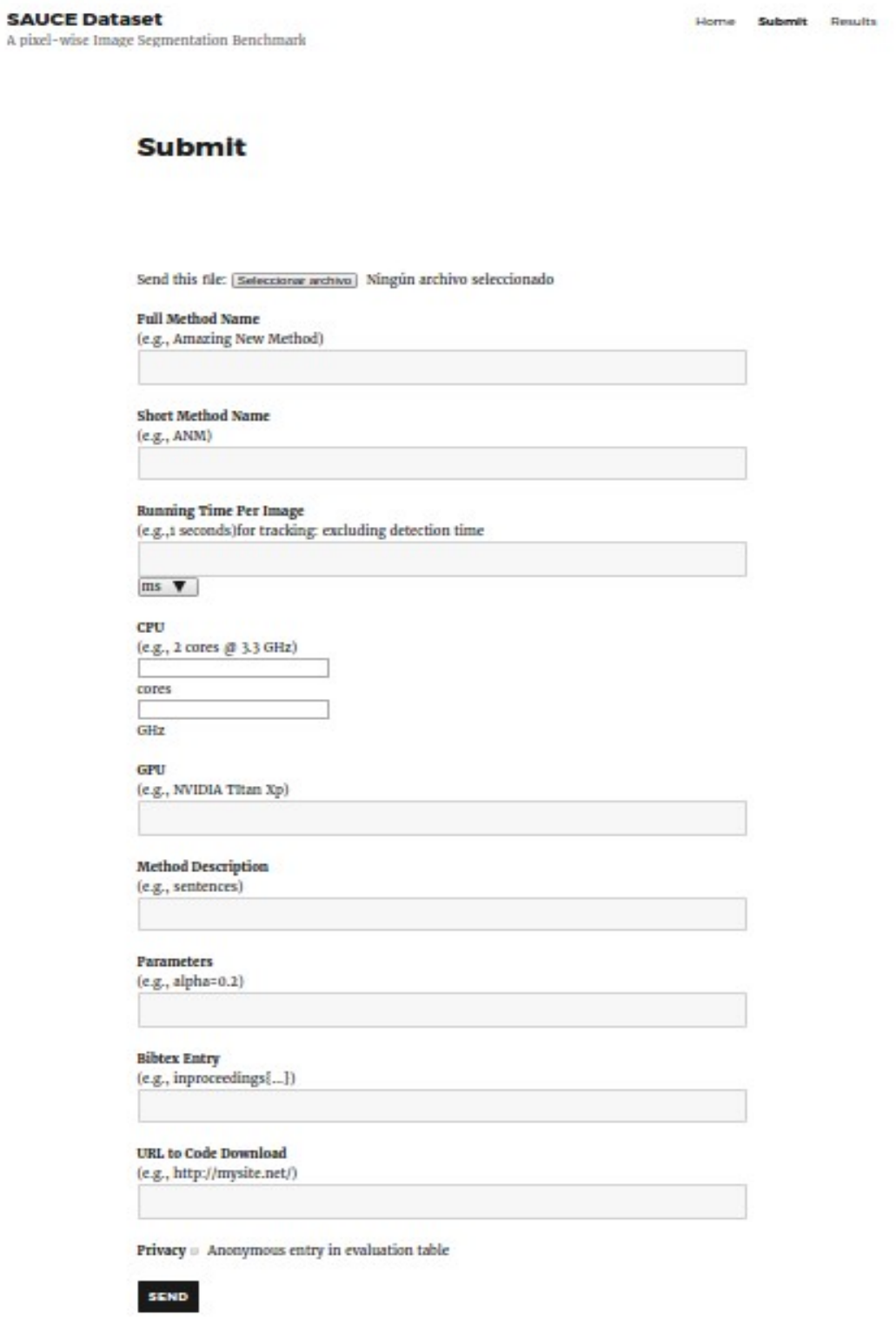


Ilustración 41. Página SUBMIT.

En esta última página al dar al botón de SEND (ENVIAR), si todos los huecos se han rellenado correctamente y con los datos puestos con sentido, se ejecutará interiormente el script de comparación usando de partida el archivo que el usuario selecciona en la primera opción. Si el archivo contiene imágenes con el correcto formato para comparar, se darán los porcentajes de acierto basándose en las fórmulas del artículo "Dense Semantic Stereo Labelling Architecture" [16] explicado en el contexto de este proyecto. La siguiente imagen muestra una página de depuración usada durante el trabajo para saber si se suben bien o no los datos, en el caso de que algún dato introducido fuera erróneo o no estuviese en el formato correcto se daría un mensaje de error y recargaría la página.

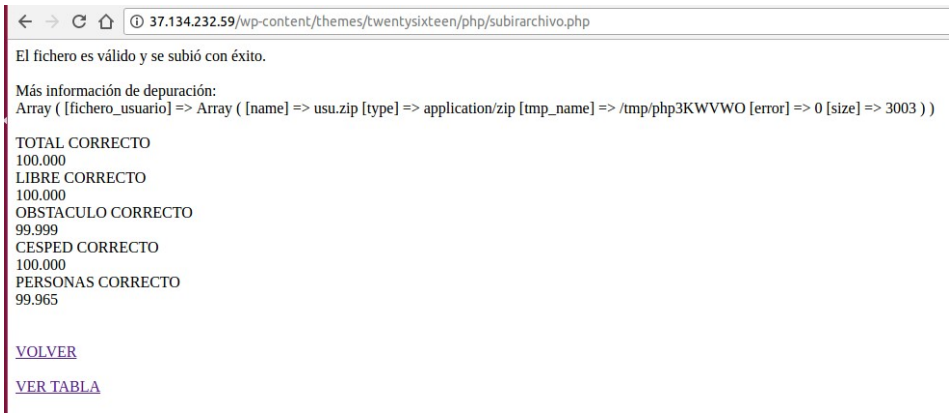


Ilustración 42. Página con la imformación de depuración.

El archivo PHP al que se llama al enviar la información, recibe en primer lugar las id que se han comentado anteriormente que tomaban los huecos de la ficha con toda la información, nombre del método, tiempo de ejecución, cpu, gpu, etc.

```
<?php
$fmn = $_POST["fmn"];
$smn = $_POST["smn"];
$time = $_POST["time"];
$cores = $_POST["cores"];
$GHz = $_POST["GHz"];
$GPU = $_POST["GPU"];
$desc = $_POST["metdesc"];
$parameter = $_POST["params"];
$bibtex = $_POST["bibtex"];
$url = $_POST["homepage"];
```

A continuación, se enlaza el archivo php con el servidor para tener un punto de conexión entre la carpeta donde se encuentra el documento en nuestro PC con internet, y se comprueba que tanto el archivo subido, como la información que se ha rellenado sea correcta.

```
//Enlaza el documento con la dirección del servidor

if (empty($fmn) || empty($smn) || empty($time) || empty($desc) ||
empty($parameter) || empty($bibtex) || empty($url) ||
empty($cores)|| empty($GHz)|| empty($GPU)||
($_FILES['fichero_usuario']['type'] != 'application/zip')){

echo 'Faltan datos para realizar el estudio.';

echo '<pre>';

$url = htmlspecialchars($_SERVER['HTTP_REFERER']);

echo "<a href='$url'>VOLVER A INTENTAR</a>\n\n";

}else{

$dir_subida = '/var/www/html/wp-content/themes/twentysixteen/zip/';

$_FILES['fichero_usuario']['name'] = 'usu.zip';

$fichero_subido = $dir_subida .
basename($_FILES['fichero_usuario']['name']);


//Comprueba que el archivo y la información subida sea algo lógico

if($_FILES['fichero_usuario']['type'] == 'application/zip'){
if (move_uploaded_file($_FILES['fichero_usuario']['tmp_name'],
$fichero_subido)) {
      echo "El fichero es válido y se subió con éxito.\n\n";
      echo "<br>";
      }
      else{
            echo "El archivo no es zip o pesa demasiado.\n\n";
            $url = htmlspecialchars($_SERVER['HTTP_REFERER']);
            echo "<a href='$url'>VOLVER</a>\n\n";
      }
}else{
            echo "El archivo no es zip o pesa demasiado.\n\n";
            $url = htmlspecialchars($_SERVER['HTTP_REFERER']);
            echo "<a href='$url'>VOLVER</a>\n\n";
}

echo "<br>";
echo 'Más información de depuración:';
echo "<br>";
print_r($_FILES);
echo "<br>";
echo "<br>";
if($_FILES['fichero_usuario']['type'] == 'application/zip'){
$output = array(); //contendrá cada linea salida desde la
aplicación en Python
```

Para finalizar se ejecutará el script en Python que dará lugar a los porcentajes de acierto de la comparación de etiquetas, y toda la información (tanto los porcentajes salidos del script, como la información del usuario) se subirá a la tabla que se encuentra en la base de datos del servidor MySQL, que como se ha explicado en el apartado de PLUGINS se ha creado con un complemento.

```
//Ejecución e impresión del algoritmo de comparación.

exec("python /var/www/html/wp-
content/themes/twentysixteen/py/metrica.py", $output);
for ($i = 0; $i <= 100; $i++) {
      echo $output[$i];
      echo "<br>";
      if ($output[$i] == NULL){
            break;
      }
}

//Opciones para movimientos entre las páginas.

echo "<br>";
$url = '/submit/';
echo "<a href='$url'>VOLVER</a>\n\n";
echo "<br>";
echo "<br>";
$url = '/results/';
echo "<a href='$url'>VER TABLA</a>\n\n";


//Limpia la carpeta donde se acumulan los archivos del usuario

$files = glob('/var/www/html/wp-
content/themes/twentysixteen/zip/*'); // obtiene todos los archivos
foreach($files as $file){
  if(is_file($file)) // si se trata de un archivo
      unlink($file); // lo elimina
}
}

//Inserta los datos y archivo del usuario en la tabla de la base de datos

 $sql = "INSERT INTO wp_t (fmn, smn, time, cores, GHz, GPU,
metdesc, params, bibtex, url, TP, LC, OC, CC, PC) VALUES ('$fmn',
'$smn', '$time', '$cores', '$GHz', '$GPU', '$desc', '$parameter',
'$bibtex', '$url', '$output[1]', '$output[3]', '$output[5]',
'$output[7]', '$output[9]')";

 $resultado = mysqli_query($con, $sql);

 if(mysqli_errno($con)) die(mysqli_error($con));

}

?>   //Cierra el archivo PHP
```

Para concluir la web nos encontramos con la pestaña RESULTS, en esta página nos vamos a encontrar la tabla que se va rellenando en función de la información que suben los usuarios. La tabla la creamos directamente en código PHP, en el que incluimos además toda la codificación HTML que es lo que se muestra por pantalla, para la organización e interfaz de la tabla usamos el Plugin explicado anteriormente denominado ***wp-jquery-datatable***. Para enlazar la página donde queremos ver el documento PHP tendremos que seleccionar en la siguiente pestaña de WordPress

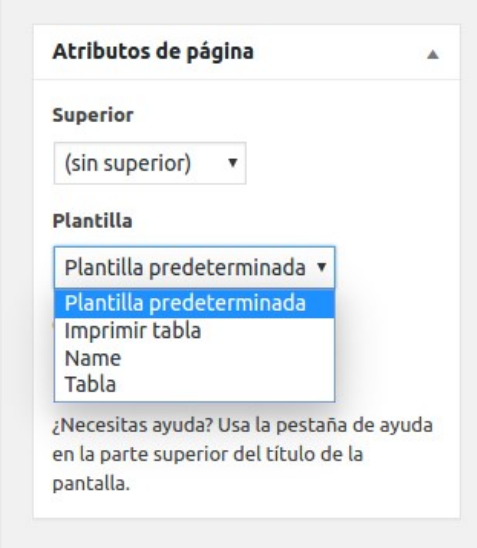


Ilustración 43. Pestaña para seleccionar el archivo PHP que mostrará la página.

En ella elegiremos el nombre que ponemos en la cabecera del documento PHP, tipo:

```
<?php
/*
Template Name: Imprimir tabla
*/?>
```

Una vez hecho esto, en la página solo se mostrará lo que se codifique en dicho archivo en lenguaje PHP. Las siguientes líneas que formarán el documento son las siguientes:

```
//Nos muestra en la página la cabecera del tema, para que siga con la dinámica de la web

<div id="primary" class="content-area">

      <main id="main" class="site-main" role="main">

            <?php
```

```
            while ( have_posts() ) : the_post();


// Include the page content template.

get_template_part( 'template-parts/content', 'page' );

// If comments are open or we have at least one comment, load up
the comment template.

if ( comments_open() || get_comments_number() )

{

     comments_template();

}// End of the loop.

endwhile;

?>
```

//Nos conectamos al servidor y a la tabla de la base de datos, para poder imprimir posteriormente su contenido

```
 define("DB_HOST","localhost" );

 define("DB_USER", "wordpressuser");

 define("DB_PASS", "Qwerty1234");

 define("DB_DATABASE", "wordpress" );

 $con = mysqli_connect(DB_HOST, DB_USER, DB_PASS, DB_DATABASE);

 if (mysqli_connect_errno())

 {

      echo "Imposible conectarse a la base de datos: " .
mysqli_connect_error();

 } else {

 }

global $wpdb;

$results = $wpdb->get_results( 'SELECT * FROM wp_t', OBJECT );

for ($i=0; $i<100; $i++){

if ($results[$i]->ID != NULL){

$contador=$contador+1;
```

//Se imprime la tabla con todo su contenido, además incluimos las líneas que activan el plugin <thead> y <tbody> para poder tener la opción de organizar la tabla

```
?>
<html>
<head>

 <table id='tabla' style='width:200%'>
 <thead>
  <tr>
    <th>Number</th>
    <th>Full Method Name</th>
    <th>Short Method Name</th>
    <th>Running Time Per Image</th>
    <th>CPU</th>
    <th>GPU</th>
    <th>Method Description</th>
    <th>Parameters</th>
    <th>Bibtex Entry</th>
    <th>URL</th>
    <th>Correct percentage (%)</th>
    <th>Free (%)</th>
    <th>Obstacle (%)</th>
    <th>Grass (%)</th>
    <th>People (%)</th>
  </tr>
  </thead>
   <tbody>
<?php for ($i=0; $i<$contador; $i++){ ?>
  <tr>
    <td>
      <?php
    echo $results[$i]->ID;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->fmn;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->smn;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->time;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->cores;
    echo ' cores @';
    echo $results[$i]->GHz;
    echo ' GHz';
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->GPU;
      ?>
```

```
    </td>
    <td>
      <?php
    echo $results[$i]->metdesc;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->params;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->bibtex;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->URL;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->TP;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->LC;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->OC;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->CC;
      ?>
    </td>
    <td>
      <?php
    echo $results[$i]->PC;
      ?>

    </td>
  </tr>
<?php }?>
</tbody>
</table>
</body>
</html>
      </main><!-- .site-main -->
      <?php get_sidebar( 'content-bottom' ); ?>

</div><!-- .content-area -->

<?php get_sidebar(); ?>
<?php get_footer(); ?>
```

Como conclusión a la web, el resultado que obtenemos es una tabla como la de la siguiente ilustración. Los usuarios verán incluida en ella el resultado de sus algoritmos y lo podrán cotejar con el de los demás.

SAUCE Dataset
A pixel-wise Image Segmentation Benchmark

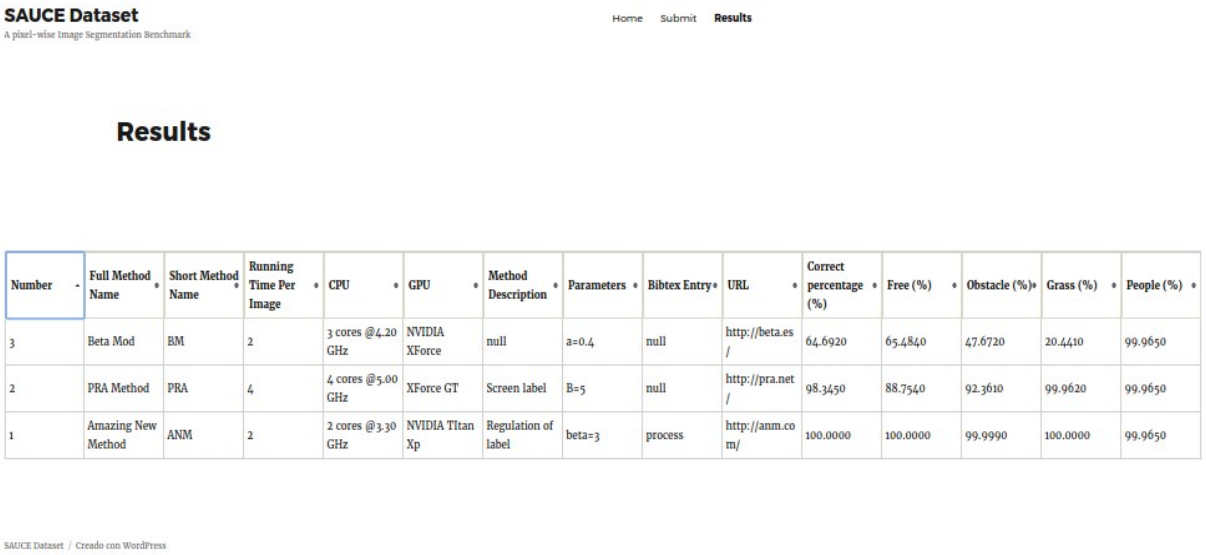
Home Submit **Results**

Results

Number	Full Method Name	Short Method Name	Running Time Per Image	CPU	GPU	Method Description	Parameters	Bibtex Entry	URL	Correct percentage (%)	Free (%)	Obstacle (%)	Grass (%)	People (%)
1	Amazing New Method	ANM	2	2 cores @3.30 GHz	NVIDIA TItan Xp	Regulation of label	beta=3	process	http://anm.com/	100.0000	100.0000	99.9990	100.0000	99.9650
2	PRA Method	PRA	4	4 cores @5.00 GHz	XForce GT	Screen label	B=5	null	http://pra.net/	98.3450	88.7540	92.3610	99.9620	99.9650
3	Beta Mod	BM	2	3 cores @4.20 GHz	NVIDIA XForce	null	a=0.4	null	http://beta.es/	64.6920	65.4840	47.6720	20.4410	99.9650

SAUCE Dataset / Creado con WordPress

Ilustración 44. Tabla con información de resultado algoritmos.



SAUCE Dataset
A pixel-wise Image Segmentation Benchmark

Home Submit **Results**

Results

Number	Full Method Name	Short Method Name	Running Time Per Image	CPU	GPU	Method Description	Parameters	Bibtex Entry	URL	Correct percentage (%)	Free (%)	Obstacle (%)	Grass (%)	People (%)
3	Beta Mod	BM	2	3 cores @4.20 GHz	NVIDIA XForce	null	a=0.4	null	http://beta.es/	64.6920	65.4840	47.6720	20.4410	99.9650
2	PRA Method	PRA	4	4 cores @5.00 GHz	XForce GT	Screen label	B=5	null	http://pra.net/	98.3450	88.7540	92.3610	99.9620	99.9650
1	Amazing New Method	ANM	2	2 cores @3.30 GHz	NVIDIA TItan Xp	Regulation of label	beta=3	process	http://anm.com/	100.0000	100.0000	99.9990	100.0000	99.9650

SAUCE Dataset / Creado con WordPress

Ilustración 45. Tabla con filas ordenadas.

También podemos ver como pinchando en la primera fila de cada columna, la tabla se ordenará por orden alfabético (de a-z o z-a en función de las veces que se accione) en función de la columna accionada, gracias al uso del plugin *Table Sorter* mencionado al principio de este capítulo.

6. Conclusiones y trabajos futuros

6.1. Conclusiones

Tras la realización del proyecto se han alcanzado los objetivos que se tenían en mente, la consecución de los mismo ha hecho que se ponga en marcha la web que se tenía prevista para usar como portal de enlace o punto en común de los algoritmos de la comunidad, y poder dar una base de información a los usuarios que los desarrollan para conseguir más avance en la búsqueda del algoritmo que consiga esa autonomía buscada. Además, la web contiene una base de datos puesta a disposición de los usuarios, que pueden descargar para usarla como punto de comparación. No obstante, la base de datos realizada a mano aún queda demasiado pequeña, con los trabajos mencionados en el apartado de trabajos futuros (punto siguiente a este) se hará más grande y se podrá dar a los usuarios un rango más amplio de confrontación, ya que unos algoritmos pueden dar mejores resultados en otros ambientes que no son los usados en la primera tanda de etiquetas a mano (tomados en una parte concreta del campus).

El sitio web se ha probado en varios servidores con hosting gratuito, pero no ha funcionado correctamente ya que limitan mucho su funcionamiento (funciones premium quedan inhabilitadas), este ha sido uno de los mayores problemas que han resultado a lo largo del proyecto, por ello finalmente se ha optado por usar la web en un servidor local de la universidad y que se pueda acceder desde todo l complejo, eso no quiere decir que no se pueda acceder desde otra partes, ya que con la dirección ip externa se podrá tener acceso desde cualquier parte.

Si los trabajos futuros se llevan a cabo se conseguirá una web muy útil para la universidad y sobre todo para el avance de la industria de los vehículos autónomos, y en cuanto a su software se refiere, en el caso del departamento, el desarrollo del proyecto iCab.

6.2.Trabajos futuros

En este punto se indicarán las posibles mejoras y evoluciones de este proyecto, dichos mejoras podrán ser desarrolladas por otros alumnos de la Universidad en sus respectivos trabajos fin de grado.

Los avances podrán ser aplicados en la parte de código del algoritmo o script en forma de:

- Aumento de efectividad y velocidad del algoritmo. El script desarrollado parte de una codificación inicial un poco "rustica", es decir, estudia pixel por pixel de cada etiqueta, lo que hace que su tiempo de ejecución sea muy elevado. Como solución se podría desarrollar alguna fórmula o codificación que depurara esta forma de comparar las imágenes para lograr mayor eficiencia en el código.

En la parte de la base de datos:

- Ampliación de la base de datos original. El primer banco de imágenes etiquetadas cuenta con unas 150 etiquetas, de las cuales se han usado como partida en la web 30 de ellas. Este número resulta algo escaso si se quiere conseguir una web robusta y con una buena base de comparación para cualquier fotografía captada en la Universidad. Por ello el desarrollo de un nuevo algoritmo que etiquetará directamente de la fotografía original sería clave para la ampliación de la base de datos, ya que el ir etiquetando foto a foto es un trabajo engorroso y que quita mucho tiempo, de esta forma lograríamos un etiquetado mucho más rápido.

Finalmente, en la parte que va relacionada con la página web:

- Depuración y mejora del sitio web SAUCE. La web desarrollada parte de una interfaz básica con muy poca personalización, ya que lo que se quería en primer momento era solo implementar el cuestionario a rellenar del usuario y la tabla donde se acumulaban todos los datos. La web podría ser modificada con una interfaz propia de la Universidad, e incluso se podrían ampliar más páginas de información del proyecto al que pertenece (iCab) o aumentar los datos que introduce el usuario para ser más precisos en la forma de uso de

cada algoritmo, incluso mejorar la visión que se ofrece de la tabla al usuario, para que sea visible perfectamente en todo tipo de plataformas.

- Incorporación de otros algoritmos. En la web que se ha desarrollado solo se lleva a cabo la comparación del resultado del algoritmo que etiqueta en RGB y con ello solo una tabla de información. Se podrían implementar usando la misma web, más páginas de tablas con información de resultados de más tipos de algoritmos que pueden ser desarrollados en TFG, incorporando en la página donde se piden los datos una pestaña donde se pueda elegir el script con el que quieres que se te evalúe.

7. Entorno socio-económico

7.1. Presupuesto

En el siguiente presupuesto se muestran los costes de materiales y de personal que ha generado el trabajo durante su desarrollo.

La mayoría de los recursos utilizados son de acceso libre, lo que da lugar a que los únicos costes sean los del PC de sobremesa usado para el desarrollo de la web y la mano de obra, si en un futuro se quisiera incorporar la web a un hosting de pago habría que añadirle el coste de la mensualidad de dicho hosting.

Para calcular los costes materiales se ha tenido en cuenta el coste de amortización que se aplica a cada elemento con su tiempo de uso. Para ello se ha utilizado la tabla de amortización simplificada del Ministerio de Hacienda [56]. La fórmula del cálculo mencionado es la siguiente:

$$Coste = precio \cdot \frac{coef.de\ amortización}{100} \cdot \frac{tiempo\ (meses)}{12}$$

En cuanto a los costes de personal, se han calculado en base al sueldo medio de un becario, coste unitario por hora de desarrollo (6,25€/hora) [57].

COMPONENTE	PRECIO (€)	COEF. AMORTIZACIÓN	TIEMPO DE USO	COSTE EN EL PROYECTO (€)
LG Gigabyte Technology Co., Ltd. Intel® Core™ i5-3550 CPU @ 3.30GHz, 3701,Memoria instalada (RAM): 8,00 GB.	530.00	26%	9 meses	103.35
Horas de desarrollo	6,25€/hora	0%	9 meses (150H)	937.5
		COSTE TOTAL		1.040,85 €

Tabla 7. Presupuesto del proyecto.

7.2. Diagrama de Gantt del proyecto

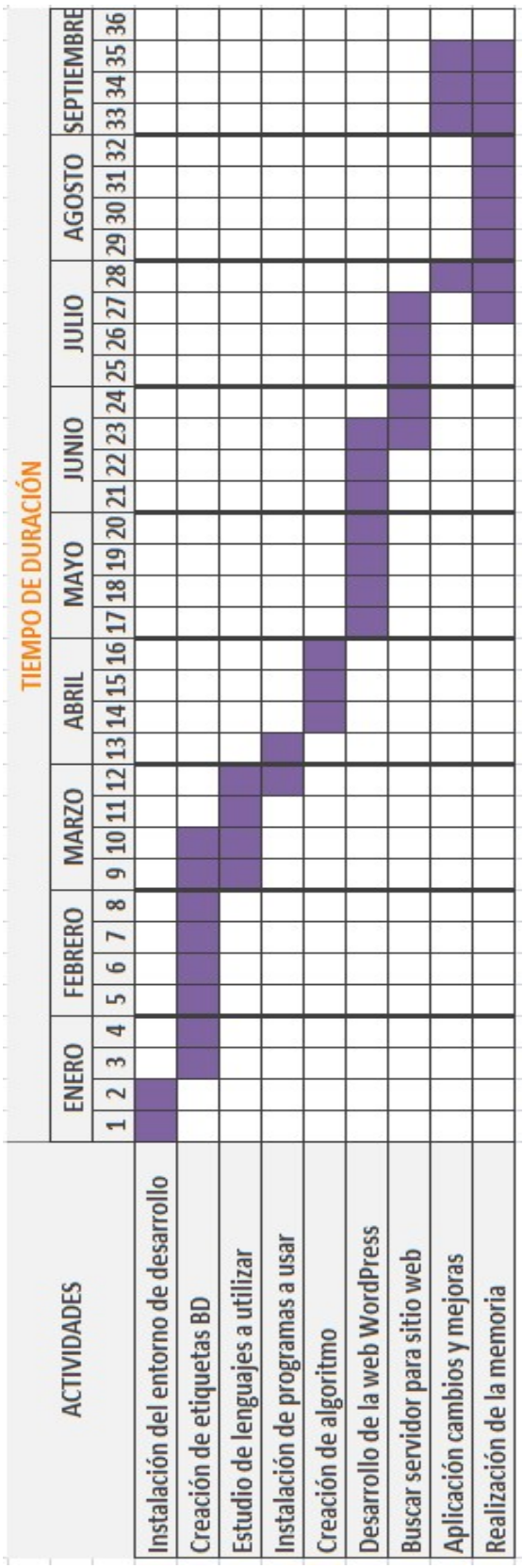


Tabla 8. Diagrama de Gantt.

8. Bibliografía

[1]. Medidas de Impulso de la S.I. http://www.boe.es/boe/dias/2007/12/29/pdfs/A53701-53719.pdf. [Último acceso: agosto 2017]

[2]. Métricas de evaluación para algoritmos de segmentación http://bibing.us.es/proyectos/abreproy/11863/fichero/PFC%252FCapitulo+V.pdf [Último acceso: agosto 2017]

[3]. Regulación de la LSE http://www.boe.es/boe/dias/2007/10/24/pdfs/A43251-43259.pdf. [Último acceso: agosto 2017]

[4]. Normativa de las TIC en la Administración Pública http://www.boe.es/boe/dias/2007/06/23/pdfs/A27150-27166.pdf. [Último acceso: agosto 2017]

[5]. Normalización de la Accesibilidad en la Administración Pública http://www.boe.es/boe/dias/2007/03/24/pdfs/A12852-12856.pdf. [Último acceso: agosto 2017]

[6]. Sistematización de las condiciones de Accesibilidad de la S.I. http://www.boe.es/boe/dias/2007/11/21/pdfs/A47567-47572.pdf. [Último acceso: agosto 2017]

[7]. De costa a costa sin conductor. http://www.elmundo.es/motor/2015/03/17/550812f7e2704e5d4f8b4585.html [Último acceso: agosto 2017]

[8]. Auto Bild Nº 537Qué es y cómo funciona un coche autónomo 10/08/2015 http://www.autobild.es/contenido-patrocinado/especial-toyota-que-es-como-funciona-coche-autonomo-262119 [Último acceso: agosto 2017]

[9]. La transición a los vehículos autónomos: ventajas e inconvenientes. http://observatorio-ia.com/la-transicion-los-vehiculos-autonomos [Último acceso: agosto 2017]

[10]. «AIM: Autonomous Intersection Management - Project Home Page». http://www.cs.utexas.edu/~aim/ [Último acceso: agosto 2017]

[11]. D.A. Forsyth, and J. Ponce. Computer Vision: A Modern Approach. Prentice Hall Professional Technical Reference. 2nd edition 2011.

[12]. Torralba. Contextual priming for object detection. International Journal of Computer Vision, Vol. 53(2), 169-191, 2003.

[13]. P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, September 2010.

[14]. Bernd Jähne and Horst Haußecker (2000). Computer Vision and Applications, A Guide for Students and Practitioners. Academic Press. ISBN 0-13-085198-1.

[15]. Linda G. Shapiro and George C. Stockman (2001). Computer Vision. Prentice Hall. ISBN 0-13-030796-3.

[16]. Jorge Beltrán, Carlos Jaraquemada, Basam Musleh, Arturo de la Escalera and Jose María Armingol, Dense Semantic Stereo Labelling Architecture for In-Campus Navigation, VISIGRAPP 2017,Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (2017)

[17]. ¿Qué es PHP? http://php.net/manual/es/intro-whatis.php [Último acceso: septiembre 2017]

[18]. Ubuntu 16.04 , información y requisitos https://www.profesionalreview.com/2016/04/22/ubuntu-16-04-lts-toda-la-informacion-y-requisitos/ [Último acceso: septiembre 2017]

[19]. ¿Por qué software libre? https://www.suomitech.com/%C2%BFpor-qu%C3%A9-software-libre [Último acceso: agosto 2017]

[20]. «Ubuntu Advantage Landscape» (en inglés). [Último acceso: agosto 2017]

[21]. «Time Based Releases» https://wiki.ubuntu.com/TimeBasedReleases [Último acceso: agosto 2017]

[22]. «Unity». http://unity.ubuntu.com/ [Último acceso: agosto 2017]

[23]. «Ubuntu System Requirements – Community Ubuntu Documentation» (en inglés). https://help.ubuntu.com/community/Installation/SystemRequirements [Último acceso: agosto 2017]

[24]. GIMP 2.7 Release Notes http://www.gimp.org/release-notes/gimp-2.7.html Cambio de licencia en la versión 2.7 [Último acceso: agosto 2017]

[25]. Gimp características http://gimp.es/ [Último acceso: agosto 2017]

[26]. http://opencvjaveriana.wikispaces.com/ Web de la Universidad Javeriana Bogotá de Documentación de las librerías OpenCV

[27]. OpenCV website. http://opencv.org/ [Último acceso: septiembre 2017]

[28]. Información OpenCV http://opencv.org/about.html [Último acceso: septiembre 2017]

[29]. Developing OpenCV computer vision apps for the Android platform. http://www.embedded.com/design/programming-languages-and-tools/4406164/Developing-OpenCV-computer-vision-apps-for-the-Android-platform [Último acceso: agosto 2017]

[30]. Spyder documentation https://pythonhosted.org/spyder/ [Último acceso: agosto 2017]

[31]. Tutorial Python http://docs.python.org.ar/tutorial/3/real-index.html [Último acceso: agosto 2017]

[32]. History and licence of Python https://docs.python.org/3/license.html [Último acceso: agosto 2017]

[33]. «Historia de PHP y Proyectos Relacionados». http://web.archive.org/ Archivado desde el original el 30 de noviembre de 2015. [Último acceso: agosto 2017]

[34]. Lenguaje de Marcado para Hipertextos (HyperText Markup Language) https://developer.mozilla.org/es/docs/Web/HTML [Último acceso: agosto 2017]

[35]. ¿Cómo instalar Linux, Apache, MySQL, PHP (LAMP) en Ubuntu 16.04? https://www.digitalocean.com/community/tutorials/como-instalar-linux-apache-mysql-php-lamp-en-ubuntu-16-04-es [Último acceso: agosto 2017]

[36]. Janai et al. - 2017 - Computer Vision for Autonomous Vehicles Problems, Datasets and State-of-the-Art

[37]. Dollár, P., Wojek, C., Schiele, B., & Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), 34, 743–761.

[38]. Geiger,A.,Lenz,P.,&Urtasun,R.(2012b). Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[39]. Uijlings, J. R., van de Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. International Journal of Computer Vision (IJCV), 104, 154–171.

[40]. He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In Proc. of the European Conf. on Computer Vision (ECCV).

[41]. Girshick, R. B. (2015). Fast R-CNN. In Proc. of the IEEE International Conf. on Computer Vision (ICCV).

[42]. Ren, S., He, K., Girshick, R. B., & Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems (NIPS).

[43]. Chen, X., Kundu, K., Zhu, Y., Ma, H., Fidler, S., & Urtasun, R. (2016b). 3d object proposals using stereo imagery for accurate object class detection. arXiv.org, 1608.07711.

[44]. Yang, F., Choi, W., & Lin, Y. (2016). Exploit all the layers: Fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers. InProc.IEEEConf.onComputerVisionandPatternRecognition (CVPR).

[45]. Cai, Z., Fan, Q., Feris, R. S., & Vasconcelos, N. (2016). A unified multi-scale deep convolutional neural network for fast object detection. In Proc. of the European Conf. on Computer Vision (ECCV).

[46]. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., & Schiele, B. (2016). The cityscapes dataset for semanticurbansceneunderstanding. InProc.IEEEConf.onComputerVision and Pattern Recognition (CVPR)

[47]. Kumar, S., & Hebert, M. (2005). A hierarchical field framework for unified context-based classification. In Proc. of the IEEE International Conf. on Computer Vision (ICCV).

[48]. Kohli, P., Ladicky, L., & Torr, P. H. S. (2009). Robust higher order potentials for enforcing label consistency. International Journal of Computer Vision (IJCV), 82, 302–324.

[49]. Ladicky, L., Russell, C., Kohli, P., & Torr, P. H. S. (2014). Associative hierarchical random fields. IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), 36, 1056–1077.

[50]. Lenz,P.,Geiger,A.,&Urtasun,R.(2015). Followme: Efficientonlinemin-cost flow tracking with bounded memory and computation. In Proc. of the IEEE International Conf. on Computer Vision (ICCV).

[51]. Ghiasi, G., & Fowlkes, C. C. (2016). Laplacian pyramid reconstruction and refinement for semantic segmentation. In Proc. of the European Conf. on Computer Vision (ECCV).

[52]. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Proc. of the International Conf. on Learning Representations (ICLR).

[53]. Wu, Z., Shen, C., & van den Hengel, A. (2016b). Wider or deeper: Revisiting the resnet model for visual recognition. arXiv.org, 1611.10080. Wulff, J., & Black, M. J. (2015). Efficient sparse-to-dense optical

[54]. Jampani, V., Kiefel, M., & Gehler, P. V. (2016). Learning sparse high dimensional filters: Image filtering, dense crfs and bilateral neural networks. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[55]. P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in BMVC, 2009.

[56]. Tabla de amortización simplificada. Agencia Tributaria. http://www.agenciatributaria.es/AEAT.internet/Inicio/_Segmentos_/Empresas_y_profesionales/Empresarios_individuales_y_profesionales/Rendimientos_de_actividades_economicas_en_el_IRPF/Regimenes_para_determinar_el_rendimiento_de_las_actividades_economicas/Estimacion_Directa_Simplificada.shtml [Último acceso: agosto 2017]

[57]. Encuesta de Salarios y Actividad profesional 2014 – 2015. Colegios oficiales deingenieros industriales de Álava, Vizcaya, Guipúzcoa y Navarra. http://www.coiig.com/COIIG/dmdocuments/Empleo%20LANBIDE/Salarios/encuesta_salarios_ingenieros_industriales_capv-navarra_2014-2015.pdf [Último acceso: agosto 2017]

[58]. Hussein, A., Mar´ın-Plaza, P., Mart´ın, D., de la Escalera, A., and Armingol, J. M. (2016). Autonomous off-road navigation using stereo-vision and laser-rangefinder fusion for outdoor obstacles detection. In Intelligent Vehicles Symposium (IV), 2016 IEEE, pages 104–109. IEEE.

[59]. Garage, W. (2010). Ros. ros. org.

[60]. Hirschmuller, H. (2008). Stereo processing by semiglobalmatching and mutual information. IEEE Transactionson pattern analysis and machine intelligence, 30(2):328-341.

[61]. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. International Journal of Computer Vision (IJCV), 88, 303–338.

[62]. Kr¨ahenb¨uhl, P., & Koltun, V. (2011). E_cient inference in fully connected CRFs with Gaussian edge potentials. In Advances in Neural Information Processing Systems (NIPS).

[63]. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[64]. Yu, F., & Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In Proc. of the International Conf. on Learning Representations (ICLR).

[65]. Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv.org, 1511.00561.

[66]. Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2016). Pyramid scene parsing network. arXiv.org, 1612.01105.

[67]. Pohlen, T., Hermans, A., Mathias, M., & Leibe, B. (2016). Full-resolution residual networks for semantic segmentation in street scenes. arXiv.org, 1611.08323.

[68]. Gadde, R., Jampani, V., Kiefel, M., Kappler, D., & Gehler, P. V. (2016a). Superpixel convolutional networks using bilateral inceptions. In Proc. of the European Conf. on Computer Vision (ECCV).

[69]. GNU General Public License WordPress.org. https://wordpress.org/about/gpl/ [último acceso: agosto 2017]

[70]. Usage of content management systems for websites. https://w3techs.com/technologies/overview/content_management/all [Último acceso: agosto 2017]

[71]. Create A Network. WordPress.org. https://codex.wordpress.org/Create_A_Network [Último acceso: agosto 2017]