ISSN 1870-4069

Segmentación de caminos en entornos virtuales

José Héctor León-Chávez, Juan-Manuel Ramos-Arreguin,
Sebastián Salazar-Colores, Saúl Tovar-Arriaga,
Jesús Carlos Pedraza-Ortega

Universidad Autónoma de Querétaro,
Facultad de Ingeniería,
México

jleon08@alumnos.uaq.mx, jsistdig@yahoo.com.mx

Resumen. Los vehículos autónomos tienen una gran importancia en el campo de la inteligencia artificial. Se aplican redes neuronales entrenadas para adaptarse a entornos no conocidos. Algunas redes neuronales convolucionales (CNN) tuvieron gran impacto en diferentes campos de la ciencia, como la biología. Sin embargo, su eficiencia en otros campos es poco explorada. En la actualidad podemos emplear modelos computacionales para simular el funcionamiento de diferentes componentes de la vida diaria como vehículos, tráfico o el clima con la finalidad de llevar a cabo experimentos sin ningún riesgo inherente. Este trabajo presenta el comportamiento de los modelos U-NET, Mobile-Net y Pix2Pix para la segmentación de caminos en un entorno virtual. Se aprovechan las facultades del entorno virtual para generar la base de datos que compartiremos. Finalmente se muestran los resultados estadísticos de cada modelo empleando como base de comparación la métrica intersección sobre unión (IoU) así como la cantidad de imágenes procesadas por segundo.

Palabras clave: CNN, UNET, Mobile-Net, Pix2Pix Unity, entornos virtuales.

Road Segmentation in Virtual Environments

Abstract. Autonomous vehicles are of great importance in the field of artificial intelligence. Trained neural networks are applied to adapt to unknown environments. Some convolutional neural networks (CNN) had a great impact in different fields of science, such as biology. However, its efficiency in other fields is little explored. At present we can use computational models to simulate the operation of different components of daily life such as vehicles, traffic or the weather in order to carry out experiments without any inherent risk. This work presents the behavior of the U-NET, Mobile-Net and Pix2Pix models for road segmentation in a virtual environment. The faculties of the virtual environment are used to generate the database that we will share. Finally, the statistical results of each model are shown using the intersection over union (IoU) metric as a basis for comparison, as well as the number of images processed per second.

Keywords: CNN, UNET, Mobile-Net, Pix2Pix Unity, virtual environments.

1. Introducción

La segmentación de imágenes es la separación de elementos en la imagen mediante algún proceso de agrupación [18]. Es una de las tareas más importantes en el área de visión por computadora y tiene una gran relevancia en campos de estudio que van desde la medicina hasta la industria bélica [13, 7]. La información contenida en una imagen puede ser reagrupada, dependiendo de los componentes con una misma clasificación o cada componente por separado y esto corresponde al tipo de segmentación empleada que puede ser semántica o por instancias.

La segmentación semántica busca asignar una etiqueta a cada elemento en la imagen agrupando a aquellos con características similares. La segmentación por instancias busca dos cosas la detección de objetos y la asignación de etiquetas a dicho objeto separando incluso a aquellos con características similares. En los últimos años con el uso de las redes neuronales convolucionales (CNN) se ha logrado un gran avance en la tarea de segmentar tanto de manera semántica como por instancias tanto en sistemas con capacidad de computo alto como en sistemas embebidos para su incorporación en vehículos autónomos [15].

El uso de vehículos autónomos de diferente índole, como son aéreos, marítimos o terrestres [16, 19], requieren de la comprensión del medio para determinar las diferentes acciones de control en la detección de caminos u obstáculos. La segmentación de caminos permite separar los diferentes elementos de la imagen, como pueden ser vegetación y cielo del objeto de interés. Para ello se emplean técnicas de agrupamiento o de inteligencia artificial.

Algunos modelos de inteligencia artificial basados en redes convolucionales, han mostrado buenos resultados en tareas de segmentación [8], por ello tienen gran relevancia en la actualidad. Sin embargo, algunos de los modelos fueron diseñados para segmentación de componentes específicos y no se han probado en diferentes tareas.

Uno de los trabajos más relacionados con este artículo es el mostrado en [14], donde se emplea dos redes convolucionales profundas (CNN), dedicadas a la detección de caminos y detección de objetos. La red que se encarga de detectar el camino para determinar la acción necesaria para alinear el centro del camino con el centro de la imagen. Emplea una arquitectura s-ResNet-18 para determinar si el vehículo se encuentra dentro o fuera del camino, así como realizar las acciones necesarias.

El mapeo del área silvestre tiene gran importancia en la comunidad científica, ya que se puede registrar información del deterioro o progreso del medio ambiente. Debido a esto, trabajos como el de [9] empleando un algoritmo de localización y mapeo simultáneo (SLAM), se generan mapas de bosque. Los sensores empleados son un LIDAR (Velodyne VLP-16), una cámara estéreo, una IMU (Unidad de medición inercial) y un GPS (Sistema de posicionamiento global).

El mapa resultante, en nube de puntos 3D, fue evaluado en términos de precisión y exactitud teniendo un error de estimación promedio de 2cm. El algoritmo de mapeo empleado fue graph-SLAM, debido a que reduce la complejidad computacional respecto a un $\mathrm{O}(\mathrm{n}^2)$ del filtro Kalman, que además requiere del mapa m todos los n puntos de referencia. Para el enfoque graph-SLAM cada nodo del grafo representa una posición particular del robot, mientras que los bordes codifican restricciones de odometría o cierres de bucle.

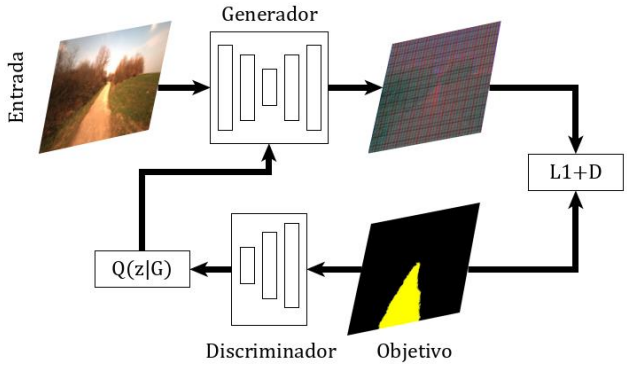


Fig. 1. Arquitectura Pix2Pix basado en [6].

En el trabajo de [1] se busca conocer la especie y el diámetro de vegetación en un bosque, mostrando que métodos no se desempeñan de manera satisfactoria en este tipo de entornos, por ejemplo LIDAR capta mucho ruido. El algoritmo propuesto en este trabajo es SLOAM (Semantic Lidar Odometry and Mapping). El propósito del algoritmo de odometría lidar es estimar el movimiento rígido de 6-DOF (grados de libertad) del lidar dentro de un barrido de 360 grados, mientras que el propósito del algoritmo de mapeo lidar es estimar la posición de 6-DOF del lidar en el mundo, y registrar la nube de puntos en el marco mundial.

El trabajo de [17] también tiene gran relevancia para este proyecto, tanto por la orientación que tomó (ADAS, Sistema Avanzado de Asistencia al Conductor), como por los resultados obtenidos. En este trabajo proponen una combinación de estructuras CNN (Redes Neuronales Convolucionales) y LSTM (Memoria de largo plazo) mostrando previamente el desempeño de ambas por separado. Se probó con la base de datos KITTI [3] obteniendo un precisión promedio de 91.6 %.

Una de las desventajas del modelo propuesto es la reducción de su desempeño en condiciones de poca luz y cuando hay presencia de sombras. En este artículo se presenta tres diferentes arquitecturas de segmentación basadas en redes neuronales convolucionales, entrenadas con una base de datos creada a partir de un entorno virtual realista. También se muestra el desempeño del modelo en tiempo real tomando en cuenta la cantidad de fotogramas procesados por segundo.

2. Modelos de segmentación

En esta sección se describirán los modelos empleados. Estos modelos comparten algunas características que nos llevaron a determinar que la comparación es adecuada. Se tiene contemplado el modelo U-Net como una de las arquitecturas referentes en la tarea de segmentación siendo una de las primeras redes en emplear el concepto de bloques residuales, Mobile-Net es una red basada en bloques residuales optimizada para agilizar operaciones y consumir menos memoria y Pix2Pix emplea como generador una red U-Net.

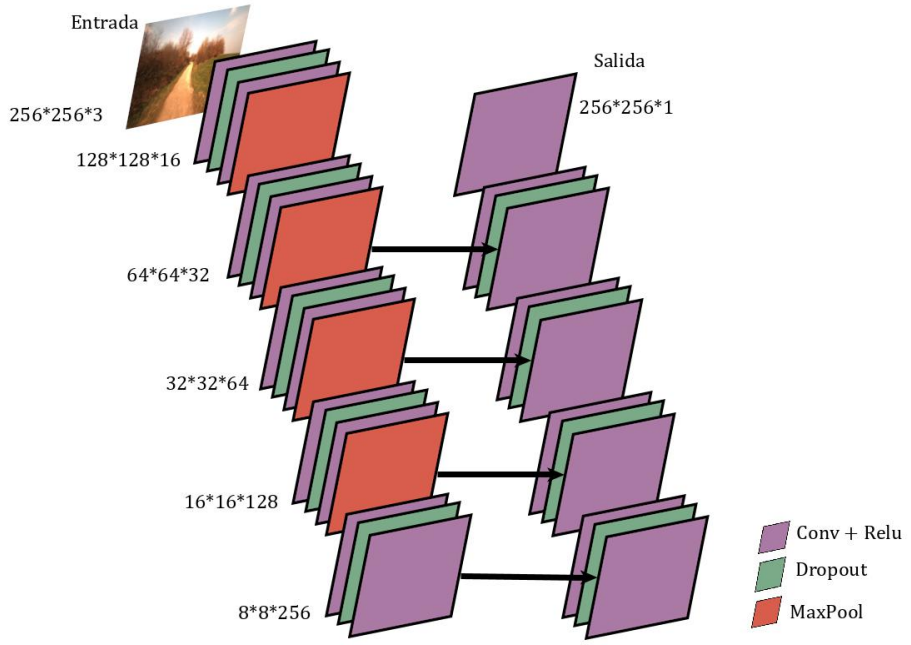


Fig. 2. Arquitectura U-Net basado en [10].

2.1. U-Net

La arquitectura U-Net propuesta en [10] es un modelo encoder-decoder que busca la extracción de características mediante un proceso de contracción y un proceso de expansión. En el proceso de contracción, la imagen reduce su tamaño (ancho y largo) a la mitad, mientras que su dimensionalidad (profundidad) aumenta, en el proceso de expansión el tamaño de la imagen aumenta al doble mientras que la dimensionalidad se reduce.

Este proceso tiende a ser muy agresivo, dado que el tamaño de la imagen se ve afectado hasta reducirlo a un valor muy pequeño. Por lo anterior, en [10] proponen conexiones entre la parte de contracción y la parte de expansión de modo que se pueda rescatar una mayor cantidad de información. En la Fig. 2 se muestra gráficamente la arquitectura U-Net.

La Fig. 2 muestra a la red U-Net como un proceso de dos pasos. En el primero (Contracción) se tiene en la entrada una imagen de $(256, 256, 3)$ a la cual se le aplican una serie de operaciones (Convolución, Dropout, Maxpool), dependiendo del punto de la extracción de características en la que se encuentre con la finalidad de reducir las dimensiones de la imagen y ampliar su profundidad.

En el segundo paso (Expansión), se aplican del mismo modo una serie de operaciones pero en este punto la finalidad es ampliar las dimensiones de la imagen y reducir la profundidad hasta uno. De manera simultánea a cada bloque de la expansión se le concatena un bloque de mismo tamaño de la contracción.

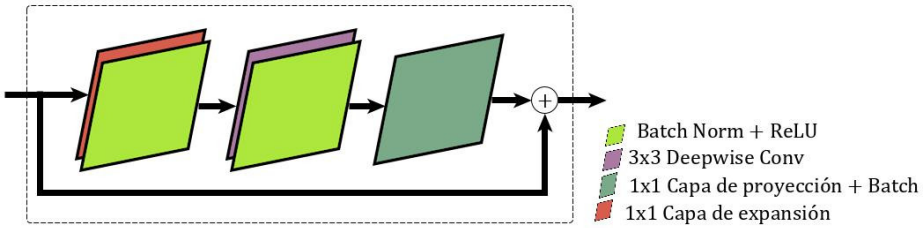


Fig. 3. Arquitectura del bloque de cuello de botella residual basado en [11].

2.2. Pix2Pix

Pix2Pix [6] es una arquitectura dentro del paradigma de redes adversarias generativas (GAN's), es decir, se entrenan simultáneamente dos redes, donde una intentará generar una imagen a partir de una imagen de entrada (red generadora) y la otra intentará diferenciar una imagen generada de una imagen objetivo (red discriminadora). En Fig. 1 se muestra un diagrama de cómo interactúan ambas arquitecturas.

En Fig. 1 se entrenan dos redes de manera simultánea, la red discriminadora se entrena de modo que pueda diferenciar una imagen real (De la base de datos) de una falsa (Generada). La red generadora se entrena de modo que se obtenga una imagen igual a la la imagen Objetivo. El modelo generador está basado en la arquitectura U-Net. El valor de $\mathrm{L1} + \mathrm{D}$ corresponde al valor de pérdida, es decir, la comparación entre la imagen generada y la imagen objetivo. El valor de pérdida ayuda a entrenar al modelo discriminador que a su vez ayuda a entrenar el modelo generador.

2.3. Mobile-Net V2

Mobile-Net V2 [11] es una mejora al trabajo realizado por [5], la cual cuenta con tres capas convolucionales en el bloque. Los dos últimos son los mismos que los mostrados en Mobile-Net V1: una convolución en profundidad que filtra las entradas, seguida de una capa de convolución puntual 1x1. Sin embargo, esta capa 1x1 ahora tiene una función diferente. En Mobile-Net V1, la convolución puntual mantuvo el mismo número de canales o los duplicó. En V2 hace lo contrario, reducir el número de canales.

Esta es la razón por la que esta capa ahora se conoce como la capa de proyección: proyecta datos con un gran número de dimensiones (canales) en un tensor con un número mucho menor de dimensiones. La primer capa del bloque también es una convolución 1×1. Su propósito es expandir el número de canales en los datos antes de que entren en la convolución en profundidad.

Por lo tanto, esta capa de expansión siempre tiene más canales de salida que canales de entrada, y hace prácticamente lo contrario de la capa de proyección. La segunda novedad del componente básico de Mobile-Net V2 es la conexión residual. Esto funciona como en ResNet [4] y existe para ayudar con el flujo de gradientes a través de la red. La conexión residual sólo se utiliza cuando el número de canales que entran en el bloque es el mismo que el número de canales que salen de él.

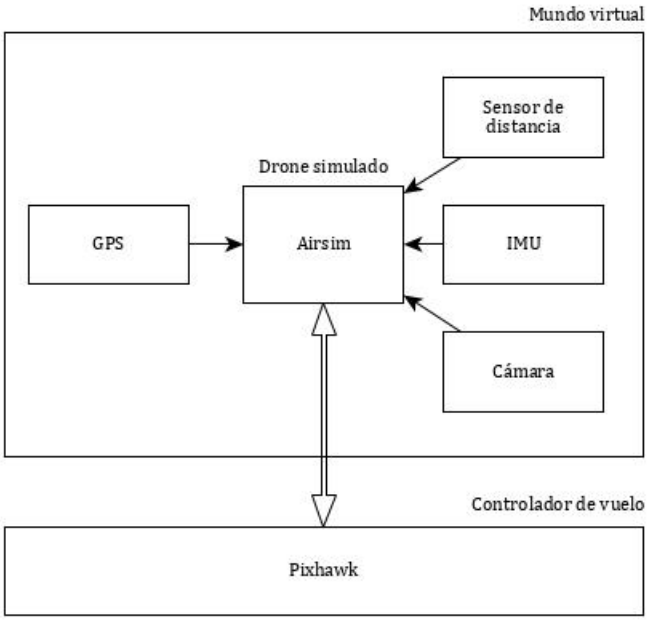


Fig. 4. Interacción entre los componentes del sistema.

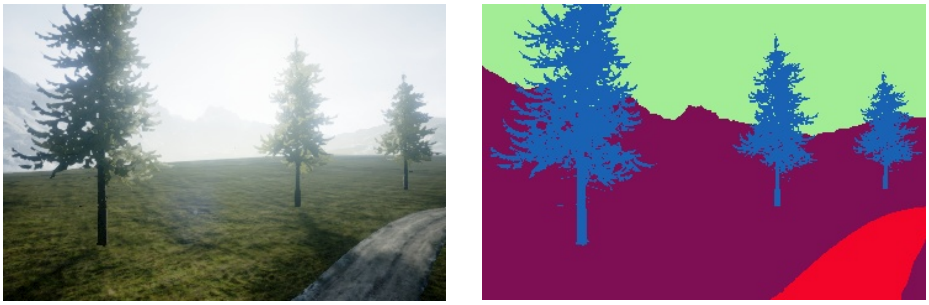
En Fig. 3 muestra gráficamente el bloque de cuello de botella residual de Mobile-Net V2. En Fig. 3 se muestra el bloque de cuello de botella en el cual a una imagen de entrada se le aplican una serie de operaciones con la finalidad de reducir la dimensionalidad de la imagen y ampliar la profundidad para después regresar a sus dimensiones originales para finalmente concatenar la entrada del bloque con la salida del mismo, esto con la finalidad de resaltar las características deseadas sin pérdida de información.

3. Materiales y métodos

En esta sección se mostrarán los diferentes pasos para la obtención de un modelo basado en redes neuronales convolucionales funcional, sobre una simulación en Unity. Los pasos mostrados abarcaran desde la realización del entorno de pruebas, pasando por la generación de la base de datos, hasta el entrenamiento y las pruebas.

3.1. Descripción del sistema

Para asegurar una plataforma estable para realizar diferentes pruebas se opta por realizarlas sobre un entorno virtual realista. El sistema se compone por un mundo virtual empleando como base los componentes de Landscape Mountain proporcionado por [2] y un simulador de drones [12] que funciona sobre el mundo virtual. El simulador de drones tiene compatibilidad con diferentes controladores de vuelo externos.



(a) Escenario (b) Segmentación

Fig. 5. Imágenes que componen la base de datos.

Se creó un componente externo al entorno suministrado por Unreal Engine (UE4). Dicho componente está basado en splines y ayuda a generar un camino de manera simple. Únicamente tiene opciones como crear una spline cerrada o abierta, mantener visible la spline o el camino. Este componente fue necesario ya que los caminos generados en el Landscape son parte del mismo y no pueden ser segmentados.

El drone simulado es un cuadricóptero que emplea diferentes sensores como lo son Lidar, IMU, GPS, sensores, cámara de distancia, entre otros. Para las pruebas se empleó un sensor de distancia orientado a -90 grados en el eje de vuelo de elevación (cabeceo), para determinar la altura del drone, respecto al piso en la simulación, dado que la posición del drone es relativa al centro del mapa.

También se incorporó un controlador de vuelo PixHawk 2.4.8, empleando HIL (hardware-in-the-loop), con la finalidad de incorporar mayor estabilidad en el vuelo y para incorporar un transmisor de radio. El controlador de vuelo va conectado via USB a un puerto COM, la comunicación entre la simulación y el controlador de vuelo requiere de un programa extra que interprete los comandos en el protocolo MAVLink.

El controlador de vuelo se encarga de realizar la estimación en la posición del drone a partir de los diferentes sensores incorporados, y ejecuta un filtro Kalman extendido (EKF) para la realización de dicha estimación. Otra función del controlador de vuelo es determinar los valores de PWM suministrados a los motores en la simulación, ya que para las diferentes acciones se requiere un comportamiento especifico de los motores. En Fig. 4 se muestra las conexiones entre el simulador de drones, el mundo virtual y el controlador de vuelo.

En Fig. 4 se muestra como los sensores que emplea el simulador de drones adquiere las mediciones del mundo virtual, es decir que las lecturas del GPS estarán reverenciadas a un punto del mundo virtual que en este caso es el punto $(0, 0, 0)$. Estas mediciones se envían al controlador de vuelo el cual regresa la estimación de la posición de la aeronave así como las señales de los motores.

La cámara incorporada en el drone simulado tiene como características un ángulo de visión de -90 grados, orientado a $(0, 0, 0)$ grados respecto al cuerpo de la aeronave, velocidad de exposición de 100 y gama objetivo de $1{,}5$. Las imágenes que arroja son de segmentación de las mallas en la imagen y la imagen del panorama en un formato RGB y un tamaño de 360×420 pixeles.

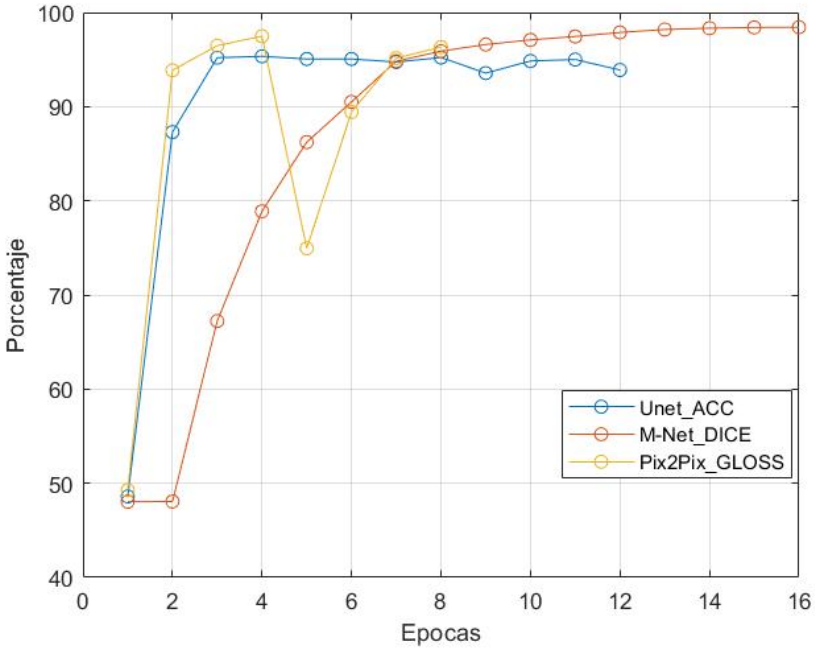


Fig. 6. Entrenamiento de las tres arquitecturas.

3.2. Generación de base de datos

La base de datos se genera a partir de las imágenes de escenario y segmentación. Se ejecuta el mundo virtual y la simulación de drone para posteriormente llevar a cabo la adquisición de dichas imágenes y almacenarlas en formato JPG. Adicional a esto se almacena la posición de la aeronave en los ejes $\mathrm{x}, \mathrm{y}, \mathrm{z}$ respecto al centro del mundo virtual.

Se tomaron cuatro fases para la generación de la base de datos. En una se intentaba que el centro del camino y el centro de la imagen se mantuvieran alineados (funcionamiento óptimo), en otra se procuraba que el centro del camino se mantuviera en el extremo izquierdo o en el extremo derecho y la tercer fase se procuraba que el camino no apareciera en la imagen.

En total se generaron 5000 imágenes con la características antes mencionadas y un mapa de puntos del recorrido en tres ejes. En Fig. 5 se muestra tanto la imagen de escenario como su segmentación.

En Fig. 5a podemos apreciar diferentes componentes que afectan la imagen, como son iluminación del sol en un punto del panorama y sombras de los diferentes componentes. En Fig. 5b se aprecian cuatro componentes segmentados, siendo el landscape el más extenso, ya que abarca todo el mundo virtual y únicamente los componentes compuestos por mallas serán segmentados.

3.3. Entrenamiento

Se entrenaron las tres arquitecturas U-Net, Mobile-Net V2 y Pix2Pix con el conjunto de datos separado en entrenamiento y pruebas. Para el entrenamiento de la red U-Net se empleó como función de pérdida entropía cruzada con pérdida logística (BCEWithLogitsLoss) descrita en 3.3:

$$\ell(x, y) = L = \{l_1, ..., l_N\}, \tag{1}$$
$$l_n = -w_n[y_n * \log \sigma(x_n) + (1 - y_n) * \log(1 - \sigma(x_n))].$$

Para el entrenamiento de la red Mobile-Net se empleó como función de pérdida el coeficiente Dice el cual está denotado por la ecuación 2:

$$D_c = \frac{2 * |A| \cap |B|}{|A| + |B|}. \tag{2}$$

La arquitectura Pix2Pix tiene como función objetivo 3:

$$G^* = \text{argmin}_G \max_D \mathcal{L}_{CGAN}(G, D) + \lambda \mathcal{L}_{L1}(G). \tag{3}$$

Y la función de pérdida esta denotada por 5:

$$\mathcal{L}(G, D)) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,y}[\log(1 - D(x, G(x, z)))], \tag{4}$$
$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|], \tag{5}$$

donde G y D son los resultados de los modelos generador y discriminador respectivamente.

4. Resultados

Los resultados se muestran comparando el desempeño de las diferentes arquitecturas y una transformación de color y saturación efectuada a la misma imagen de entrada, también se muestra el comportamiento de las métricas empleadas como función de pérdida.

4.1. Entrenamiento

El entrenamiento se llevo a cabo con el mismo conjunto de datos separados de la misma forma 4900 imágenes de entrenamiento y 100 imágenes de validación. En Fig. 6 se muestra parte del proceso de entrenamiento de las tres arquitecturas. Las tres arquitecturas mostraron una pérdida cercana a cero pero con valores mayores a los documentados con bases de datos conocidas. El tiempo de entrenamiento de cada uno de los modelos es de aproximadamente 5 horas.

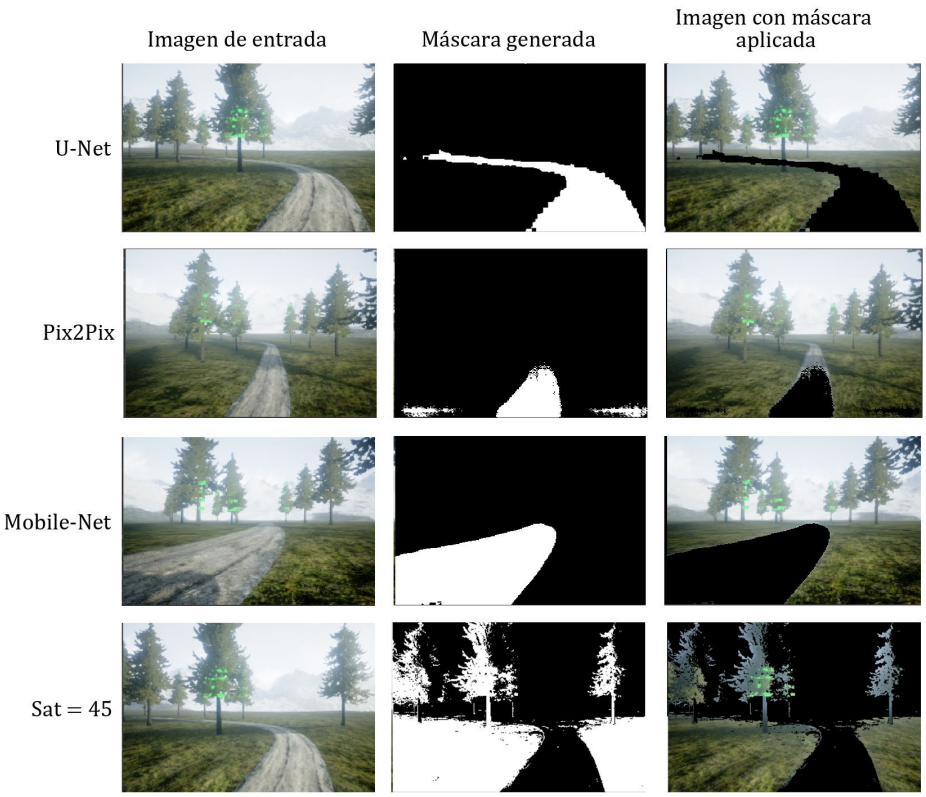


Fig. 7. Comparación de desempeño de las tes arquitecturas y el ajuste a saturación.

4.2. Pruebas

En esta sección se mostrará una comparativa entre las diferentes arquitecturas, además se comparará con la eficiencia de un sistema no basado en redes convolucionales. Para estas pruebas se empleo un trayecto diferente al trayecto empleado para generar la base de datos, este nuevo conjunto consta de 1500 imágenes. En Fig. 7 se muestra la comparación de las diferentes arquitecturas funcionando en tiempo real sobre la simulación en el trayecto de pruebas.

En la columna Imagen de entrada de Fig. 7 se tiene una imagen RGB de $(240 \times 360 \times 3)$, en la columna Máscara generada se tiene una imagen de $(240 \times 360 \times 1)$ resultado de cada modelo y en la columna final se tiene el resultado de aplicar la máscara a la imagen de entrada. El renglón $\mathrm{Sat} = 45$ es el resultado de aplicar una técnica clásica para segmentación de imágenes, el proceso consta en variar los parámetros de saturación, matiz y valor a una imagen en formato .hsv.

El desempeño se evaluó a partir de la métrica intersección sobre unión (IoU) mostrada en 6, también se evaluó el desempeño midiendo la cantidad de imágenes que cada arquitectura podía procesar en un segundo. En Tabla 1 se muestra el promedio,

Tabla 1. Imágenes procesadas por segundo e IoU.

Modelo	**Min.**	**Max.**	**Prom.**	**IoU**
U-Net	7.62	8.48	8.33	0.76
Mobile-Net	6.63	7.75	7.02	0.98
Pix2Pix	11.86	13.2	12.37	0.46
Sat=45	13.12	14.52	13.61	0.11

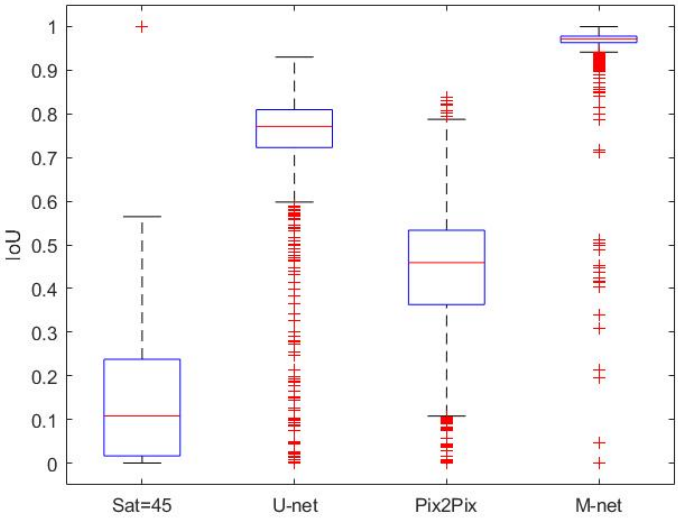


Fig. 8. Gráfica de caja de la métrica IoU sobre la base de datos de prueba.

máximo y mínimo de imágenes procesadas por segundo así como el promedio de IoU sobre las 1500 imágenes de prueba.

Del mismo modo en la Figura 8 se muestra la gráfica de caja de los datos obtenidos con la métrica IoU:

$$\text{IoU} = \frac{A \cap B}{A \cup B}. \tag{6}$$

5. Conclusiones

Se mostró el resultado al evaluar tres arquitecturas sobre una simulación funcionando en tiempo real, las tres arquitecturas tienen características similares U-Net [10] emplea bloques residuales, Mobile-Net [11] también emplea bloques residuales y Pix2Pix [6] emplea una U-Net como generador.

La arquitectura con mayor desempeño empleando como comparación IoU es Mobile-Net sin embargo creemos que la arquitectura U-Net podría alcanzar un desempeño similar si la imagen de entrada tiene mayor resolución de modo que los procesos de contracción y expansión cuenten con una cantidad mayor de bloques.

También se mostró una comparación de las tres arquitecturas contra una de las técnicas clásicas de segmentación desarrollada con un algoritmo simple en OpenCv.

En las diferentes pruebas pudimos notar que este algoritmo es susceptible a los elementos fuera del camino como lo son sombras o arboles por ello creemos que el uso de algoritmos basados en CNN en segmentación esta justificado.

Uno de los factores que más resaltan es la cantidad de imágenes que pueden ser procesadas por segundo ya que este factor es fundamental para el correcto funcionamiento de un vehículo autónomo por ello se propone evaluar las tres arquitecturas al recorrer un trayecto empleando las mismas condiciones.

Este trabajo un paso crucial en un proyecto de vehículos autónomos cuyos avances representan la validación y desarrollo de vehículos autónomos en entornos fuera de riesgo ya que combina el funcionamiento convencional de un drone con una simulación en tiempo real propiciando la integración de técnicas de inteligencia artificial. Finalmente la base de datos se encuentra disponible para el uso de la comunidad en general.

Referencias

1. Chen, S. W., Nardari, G. V., Lee, E. S., Qu, C., Liu, X., Romero, R. A. F., Kumar, V.: SLOAM: Semantic lidar odometry and mapping for forest inventory. IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 612–619 (2020) doi: 10.48550/ARXIV.1912.12726
2. Epic Games: Unreal engine, https://www.unrealengine.com
3. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Proceedings of the Conference on Computer Vision and Pattern Recognition. pp. 3354–3361 (2012) doi: 10.1109/CVPR.2012.6248074
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) doi: 10.48550/ARXIV.1512.03385
5. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications, (2017) doi: 10.48550/ARXIV.1704.04861
6. Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125–1134 (2017) doi: 10.48550/ARXIV.1611.07004
7. Kim, J. U., Kim, H. G., Ro, Y. M.: Iterative deep convolutional encoder-decoder network for medical image segmentation. In: Proceedings of the 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 685–688 (2017) doi: 10.1109/embc.2017.8036917
8. Mortazi, A., Bagci, U.: Automatically designing CNN architectures for medical image segmentation. In: Proceedigns of the International Workshop on Machine Learning in Medical Imaging. pp. 98–106 (2018) doi: 10.48550/ARXIV.1807.07663
9. Pierzchała, M., Giguère, P., Astrup, R.: Mapping forests using an unmanned ground vehicle with 3d LiDAR and graph-SLAM. Computers and Electronics in Agriculture, vol. 145, pp. 217–225 (2018) doi: 10.1016/j.compag.2017.12.034
10. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241 (2015) doi: 10.48550/ARXIV.1505.04597

11. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C.: MobileNetV2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018) doi: 10.48550/ARXIV.1801.04381
12. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In: Field and Service Robotics (2017), https://arxiv.org/abs/1705.05065
13. Shan, P.: Image segmentation method based on k-mean algorithm. EURASIP Journal on Image and Video Processing, vol. 2018, no. 1, pp. 81 (2018) doi: 10.1186/s13640-018-0322-6
14. Smolyanskiy, N., Kamenev, A., Smith, J., Birchfield, S.: Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4241–4247 (2017) doi: 10.48550/ARXIV.1705.02550
15. Sultana, F., Sufian, A., Dutta, P.: Evolution of image segmentation using deep convolutional neural network: A survey. Knowledge-Based Systems, vol. 201 (2020) doi: 10.1016/j.knosys.2020.106062
16. Wang, Y., Peng, M., Di, K., Wan, W., Liu, Z., Yue, Z., Xing, Y., Mao, X., Teng, B.: Vision based obstacle detection using rover stereo images. International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences, vol. XLII-2/W13, pp. 1471–1477 (2019) doi: 10.5194/isprs-archives-xlii-2-w13-1471-2019
17. Yang, X., Li, X., Ye, Y., Lau, R. Y., Zhang, X., Huang, X.: Road detection and centerline extraction via deep recurrent convolutional neural network u-net. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 9, pp. 7209–7220 (2019) doi: 10.1109/tgrs.2019.2912301
18. Yuheng, S., Hao, Y.: Image segmentation algorithms overview, (2017) doi: 10.48550/ARXIV.1707.02051
19. Zhou, H., Kong, H., Wei, L., Creighton, D., Nahavandi, S.: On detecting road regions in a single UAV image. IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 7, pp. 1713–1722 (2016) doi: 10.1109/tits.2016.2622280