Gate Detection for Micro Aerial Vehicles Using a Single Shot Detector

A. Cabrera-Ponce, L. Rojas-Perez, J. Carrasco-Ochoa, J. Martínez-Trinidad, and J. Martinez-Carranza

***Abstract*—Object detection has become an essential tool in aerial robotics thanks to the use of onboard cameras in drones that enables find objects using techniques of vision. However, vision algorithms may become unreliable presenting drawback by the illumination changes. Deep learning has been used to solve tasks of classification, segmentation and detection using traditional Convolutional Neural Network (CNN) like VGG16, YOLO and AlexNet. This paper presents a gates detector system in real-time using CNN based on a Single Shot Detector Network (SSD) for drone racing circuits. For the latter, we have adopted the SSD7 architecture to modified and present an implementation with five layers, reducing the prediction time and improve detection velocity in comparison with other architectures. For evaluation purpose, we selected three environments: simulation, indoors and outdoors to compare the prediction time, average fps and the confidence obtained in the detections of the gates.**

***Index Terms*—CNN, Drone Racing, Single Shot Detector.**

I. Introducción

La detección de objetos ha sido una tarea importante en el procesamiento de imágenes para aplicaciones de seguridad y robótica que requieren interpretar una escena a partir de una sola imagen. Esta interpretación conserva múltiples factores para representar el fondo o discernir entre el fondo y un objeto. Este problema se ha resuelto utilizando métodos tradicionales como la segmentación de color, detección de bordes o la extracción de características. Sin embargo, la detección puede verse afectada debido a los cambios de escenario, iluminación, oclusión del objeto e incluso presentar detecciones erróneas al detectar otros objetos.

Actualmente, el aprendizaje profundo ha desarrollado nuevos métodos para resolver problemas dentro de la robótica utilizando CNNs basadas en VGG16 [1], YOLO [2] y AlexNet [3] para clasificar y detectar objetos. Las CNNs son útiles en robótica al identificar objetos con un robot y con base a lo que observa lograr tomar una decisión para realizar una manipulación o una navegación autónoma. En los últimos años dentro de la robótica aérea, el *International Conference on Intelligent Robots and Systems (IROS)* organiza la carrera de drones autónomos (ADR) [4], [5], [6] la cual tiene como objetivo que un vehículo aéreo navegue a través de un circuito de puertas en el menor tiempo posible. Algunas de las soluciones propuestas por los participantes consiste en usar el aprendizaje profundo para la identificación de las puertas utilizando redes como YOLO3 [7] y SSD [8], ejecutando un planificador de vuelo para que el dron complete la pista una vez detectadas las puertas.

A. Cabrera-Ponce, L. Rojas-Perez, J. Carrasco-Ochoa, J. Martínez-Trinidad, and J. Martinez-Carranza, Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Puebla, México. Department of Computer Science at INAOE. Email addresses: (aldrichcabrera, oyukirojas, ariel, fmartine, carranza)@inaoep.mx

J. Martinez-Carranza, University of Bristol, Bristol, UK.

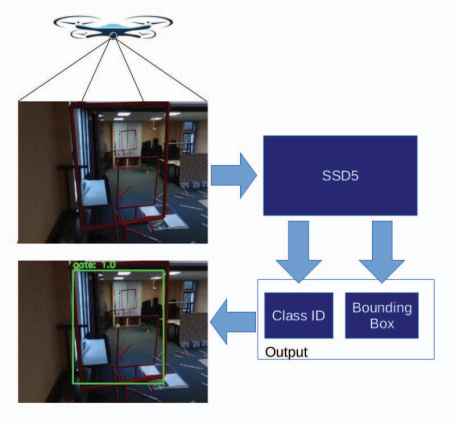


Fig. 1. Presentamos un sistema de detección utilizando una red SSD5 basada en la versión pequeña de 7 capas SSD7 del trabajo [9] para la detección de puertas en un circuito de carreras de drones.

La SSD a diferencia de otras redes y otras técnicas para la detección de objetos, aprende las características principales tales como: la forma, el color, aspecto, escala, saturación y textura, sin importar los cambios de iluminación, la oclusión parcial y los cambios de entorno que puedan perjudicar la apariencia del objeto. Además, la red es capaz de identificar múltiples objetos sin confundir las clases pertenecientes a ellos, obteniendo mayor información en la imagen para lograr una detección adecuada. Sin embargo, el aprendizaje profundo no es suficiente para que un dron tome la decisión de atravesar las puertas y completar el circuito, debido a que el tiempo de detección puede llegar a generar retardos al momento de pasar por una puerta, provocando desviaciones en el recorrido y colisiones con el vehículo.

Por lo anterior, el objetivo de este trabajo es llevar acabo una detección de puertas en tiempo real utilizando una SSD, explorando el alcance de la red para aumentar la velocidad de predicción al momento de detectar una puerta sin depender de muchos umbrales como en técnicas tradicionales. Nuestra red es una SSD de 5 capas basada en el modelo de la SSD7 [9], obteniendo una detección rápida a comparación de otras redes de aprendizaje profundo (Figura 1). Además, presentamos experimentos de evaluación en tres escenarios diferentes: Simulación, Interiores y Exteriores, comparando los tiempos de predicción y la velocidad de detección con otras arquitecturas de aprendizaje profundo, y mostramos resultados

de la detección obtenidas con la SSD5.

Este documento está organizado de la siguiente manera: Sección II mostramos los trabajos relacionados de detección de objetos con aprendizaje profundo, mencionando aquellos enfocados en carrera de drones; Sección III, describimos la metodología del enfoque propuesto; Sección IV mostramos la comparación de nuestro sistema con otras implementaciones de aprendizaje profundo; Sección V presentamos los resultados obtenidos de la detección; La conclusión y el trabajo a futuro se describen en la Sección VI.

II. Trabajo Relacionado

Actualmente, el uso de los drones ha estado creciendo en el área de la robótica para realizar múltiples tareas usando técnicas de visión y la cámara a bordo del dron, permitiendo resolver problemas de vigilancia, exploración, seguimiento, búsqueda y detección. Sin embargo, los drones se enfrentan a nuevos problemas al tratar de realizar estas tareas en: escenarios no estructurados altamente dinámicos [10], zonas estrechas [11], brechas inclinadas [12] que requieran vuelos a alta velocidad y zonas restringidas para el seguimiento en trayectorias agresivas [13]. Por ello, sistemas como [14] desarrollan plataformas con fusión de datos inerciales y visión para emplear maniobras de control a través de escenarios desafiantes. Así mismo, existen algoritmos como [15] que utilizan políticas complejas combinando percepción y control para optimizar trayectorias, navegación en relación con un mapa 3D global [16], ORB-SLAM2 [17] y navegación usando servo visual directo con un planificador para entornos obstaculizados similares a un circuito de carrera de drones [18]. No obstante, la detección de objetos en navegación autónoma es un reto para la visión computacional debido a las variaciones de apariencia, superposición y oclusión.

Por ello, competencias internacionales han combinado estos problemas para carrera de drones autónomos. IROS es el principal evento que organiza esta competencia, el cual consiste en que un dron navegue a través de puertas en una ruta definida. Por otro lado, AlphaPilot organizado por Lockheed Martin es una competencia que consiste en la navegación autónoma de un dron a través de un escenario simulado, utilizando inteligencia artificial. El objetivo de estas competencias es incentivar a investigadores y estudiantes para desarrollar un dron autónomo capaz de concluir lo más rápido posible una pista compuesta por múltiples puertas sin intervención humana. Debido a las complicaciones de iluminación los algoritmos tradicionales tienden a fallar en la detección, optando por técnicas de aprendizaje profundo en computadoras integradas a bordo del dron.

Para resolver este problema, varios trabajos se han centrado en acelerar el proceso de detección para volar rápidamente a través de puertas en un circuito de carreras sin ninguna colisión, usando localización predictiva basado en modelos visuales [19]. En [20] y [21] utilizan una red para detectar y estimar la posición del dron en relación a las puertas, presentando resultados en simulación y escenarios reales. Por otro lado, [22] presenta una CNN para estimar el centro de la puerta, comparando la velocidad de predicción con otras redes. Además, la combinación de la percepción con una CNN es útil para planificar trayectorias al momento de detectar las puertas [23] y poder predecir las posiciones integrando un filtro Kalman para mantener su ubicación [24].

Por otro lado, varias estrategias emplean detectores basados en MobileNet [25] para lograr una velocidad mayor en tiempo real. MobileNet-SSD [26] es ampliamente utilizada para identificar diferentes objetos sin importar los cambios en el entorno, siendo útil para evadir obstáculos en una navegación autónoma en tiempo real [27]. A pesar de nuevas implementaciones en MobileNet, se sigue explorando la SSD para realizar detecciones de manera rápida incluyendo combinaciones para estimar la posición de los objectos [28], [29]. En este trabajo presentamos un sistema de detección de puertas utilizando una SSD modificada para ejecutarse en sistemas integrados. El objetivo principal es aumentar la velocidad de detección de la red, conservando una nivel de confianza alto en la detección de las puertas. Además, con este trabajo se quiere explorar los alcances de la red comparando su velocidad y confianza con arquitecturas similares de detección. Nuestro sistema ha sido probado en simulación y escenarios reales para observar el desempeño de nuestra red en diferentes condiciones al momento de detectar las puertas.

III. Metodología

Nuestro sistema de detección es una SSD5 basada en la SSD7 modificada para predecir puertas en un circuito de carrera para drones. En este trabajo usamos imágenes RGB capturadas con el dron Bebop 2 para generar un conjunto de datos de entrenamiento y un conjunto para validación. Las imágenes fueron obtenidas con la comunicación del Sistema Operativo Robótico (ROS) [30] y el vehículo aéreo a través de una computadora con Ubuntu 16.06 LTS. Además, todas las imágenes fueron etiquetadas manualmente, donde la puerta más prominente aparece en el campo de visión. En la Figura 2 mostramos la arquitectura de comunicación utilizada para este trabajo.

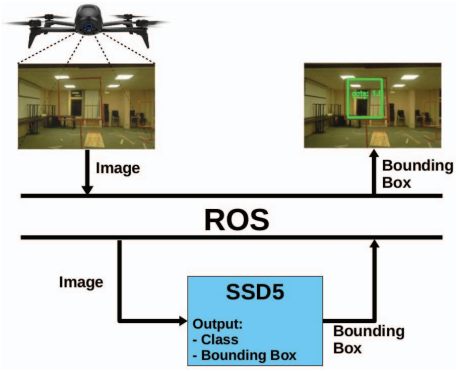


Fig. 2. Arquitectura de comunicación utilizada para la detección de puertas.

A. SSD (Single Shot Detector)

La red SSD combina predicciones en múltiples mapas de características de diferentes tamaños, permitiendo producir detecciones en niveles altos y bajos de la imagen al aplicar

filtros de convolución. Esta red es utilizada en tareas de detección y clasificación de objetos, usando la técnica de regresión para obtener un cuadro delimitador. La SSD tiene como base la red VGG16 que se ajusta al marco general de detección de objetos en el aprendizaje profundo, agregando capas convolucionales en la parte superior de la red. Estas capas disminuyen progresivamente el tamaño de la imagen de entrada en una submatriz para obtener las detecciones en múltiples escalas. Cada punto en el mapa de características cubre una parte de la imagen para predecir la clase, y con la regresión estima el cuadro delimitador del objeto en las múltiples ubicaciones dentro de la imagen.

De mismo modo, la red SSD7 es una pequeña red optimizada que realiza la misma operación para detectar objetos en una imagen, reduciendo el tiempo de entrenamiento y de búsqueda debido a las 7 capas de convolución que posee. Las predicciones de esta red pueden llegar a presentar detecciones con una mayor velocidad de predicción que otras redes como YOLO3, tinyYOLO y las FRNS. Por ello, para este trabajo proponemos utilizar una red SSD5 (Figura 3), basada en la SSD7 optimizada por ser de baja complejidad y tres veces más rápida que una SSD300.

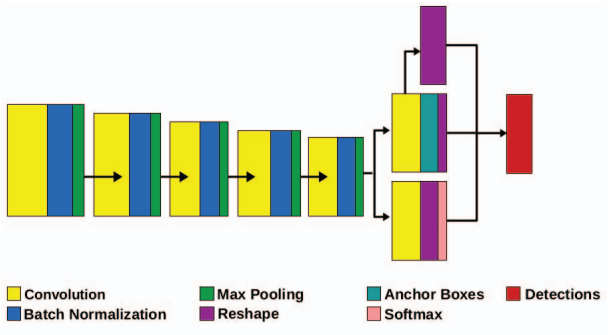


Fig. 3. Arquitectura de la red SSD5.

Nuestra red cuenta con cinco capas convolucionales en la parte superior al eliminar las dos últimas capas Conv6 y Conv7 de la SSD7. Cada filtro de convolución en la red es aplicado tanto en el ancho como en la altura del volumen de la entrada para producir un mapa de activación bidimensional. De esta manera se recibe en la entrada de la red una imagen RGB con resolución QVGA (240×320), pasando a través de la primera capa de convolución (Conv1) con 32 filtros, luego en la Conv2 se aplican 48 filtros a la salida de la Conv1 mientras que en la Conv3 y Conv4 son aplicados 64 filtros. Finalmente la Conv5 utiliza 48 filtros a la salida de las capas anteriores para después pasar por una concatenación de los cuadros delimitadores que predicen las detecciones de los objetos.

Por lo tanto, de la Conv4 se obtiene 4800 cuadros delimitadores posibles para la clase *"gate"* y 1200 cuadros delimitadores en la Conv5 (Figura 4). Al eliminar las capas Conv6 y Conv7 se eliminaron las conexiones para concatenar las predicciones de los cuadros delimitadores posibles, agilizando la predicción de detección alrededor del objeto al obtener menos cuadros delimitadores para cada clase. En otras palabras, los cuadros de salida de un red SSD7 son de 6340 para cada clase mientras que en la SSD5 son 6000 al eliminar las últimas dos capas. Esta acción reduce el tiempo de predicción y el cálculo de matrices en la imagen.

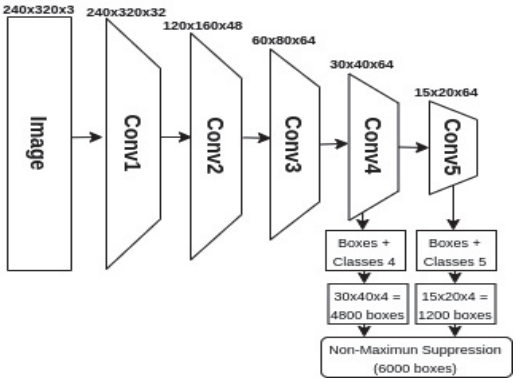


Fig. 4. Capas de la SSD5.

Finalmente, en la última capa se concatenan las predicciones de las capas anteriores y los tensores de los cuadros delimitadores asociados, obteniendo las predicciones de estos cuadros y sus clases directamente de los mapas de características en una sola pasada. Sin embargo, dada la gran cantidad de cuadros generados por la SSD5 (Figura 5), es necesario eliminar la mayor parte de los cuadros delimitadores mediante una técnica de supresión no máxima y el cálculo de la pérdida de confianza. La pérdida de confianza es una métrica que mide que tan acertada es una detección, relacionando la intersección sobre la unión entre el *Ground-Truth* y la predicción del cuadro delimitador (Figura 6).

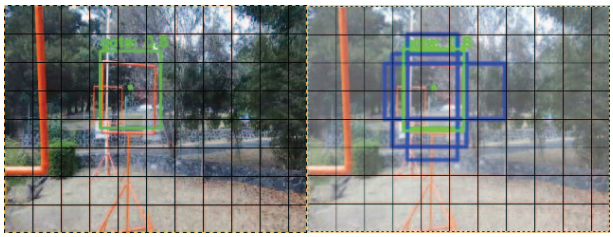


Fig. 5. Ejemplo de los múltiples cuadros detectados por la SSD5.

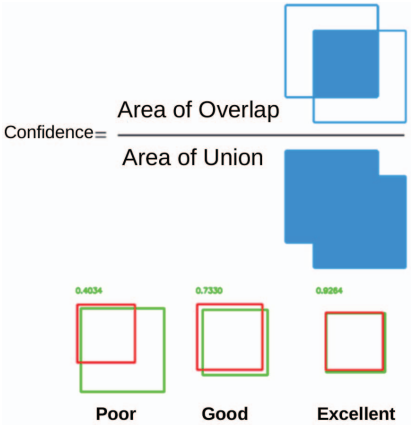


Fig. 6. Cálculo de la confianza (imagen tomada de [31]).

Debido a la métrica de confianza se puede asignar un umbral alto para filtrar los cuadros delimitadores menores a

ese umbral. En la Figura 7 mostramos el cuadro de confianza alta (cuadro verde) y los cuadros filtrados (cuadros azules). El umbral de confianza es de 0.7 por lo que los cuadros con un umbral menor son considerados como negativos y se descartan. Esto se decidió al ver que los cuadros con un umbral menor de 0.7 presentan un área mayor o menor que la puerta, ocasionando que el centroide se ubique en alguno de los bordes de la puerta lo que provocaría que el dron se estrelle al momento de atravesarla. Además, los cuadros con una confianza superior al umbral se consideran como positivos y se dibuja el área principal en la imagen donde el objeto es detectado.

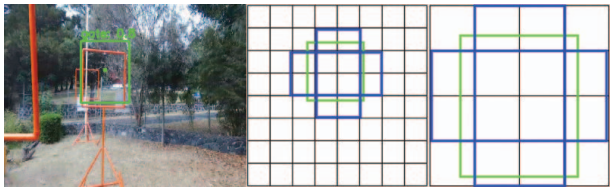


Fig. 7. Elección del mejor cuadro delimitador.

B. Generación de Conjunto de Datos

Nuestro conjunto de datos de entrenamiento (Tabla I) consta de 3418 imágenes con resolución QVGA (320×240), 2787 imágenes fueron capturadas en un entorno simulado con el Parrot AR Drone dentro del simulador Gazebo y 631 imágenes fueron capturadas en un entorno real usando el Bebop 2 Power Edition (Figura 8). Los datos de entrenamiento en el entorno real fueron obtenidos a través de ROS usando una estación de control en tierra para establecer una comunicación WiFi hacia el dron. Esta comunicación nos permite enviar comandos de control al vehículo en *Roll, Pitch y Yaw* desde la computadora usando el SDK *"bebop_autonomy"* [32]. Después, etiquetamos manualmente las imágenes capturadas para crear los cuadros delimitadores donde las puertas aparecen en la imagen, asignándoles la clase *"gate"*. Estas etiquetas contienen información de la siguiente manera: Nombre de la imagen, coordenadas del cuadro delimitador (x_min, x_max, y_min, y_max) y la clase.

TABLA I
GENERACIÓN DEL CONJUNTO DE DATOS

Entorno	Entrenamiento	Validación
Simulación	2508	279
Real	502	129

(a) Parrot AR Drone. (b) Bebop 2 Power Edition.

Fig. 8. Drones utilizados para la generación del conjunto de datos.

El entrenamiento de la red fue realizada en la computadora Lenovo Y700 con 16GB de RAM y un procesador gráfico GeForce GTX 960M con 640 núcleos CUDA, usando CUDA 9.0, Keras 2.2.4 y TensorFlow 1.12.0 cuyos parámetros de entrenamiento para cada una de las redes fueron: 40 épocas, *batch_size* de 32 y el optimizador Adam. El valor de pérdida se representa en la Figura 9 cuyas imágenes de entrenamiento se muestran en la Figura 10.

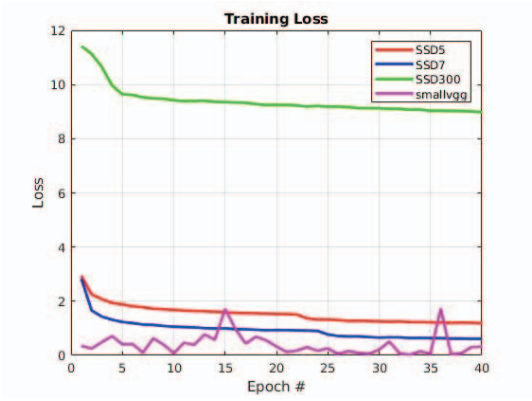


Fig. 9. Valor de pérdida en el entrenamiento.

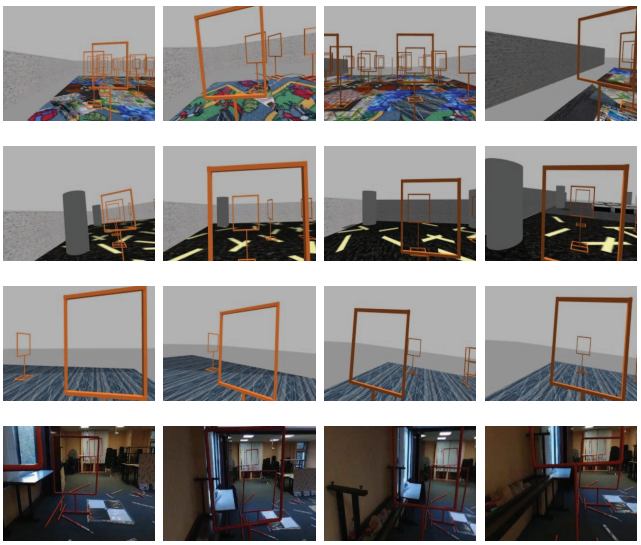
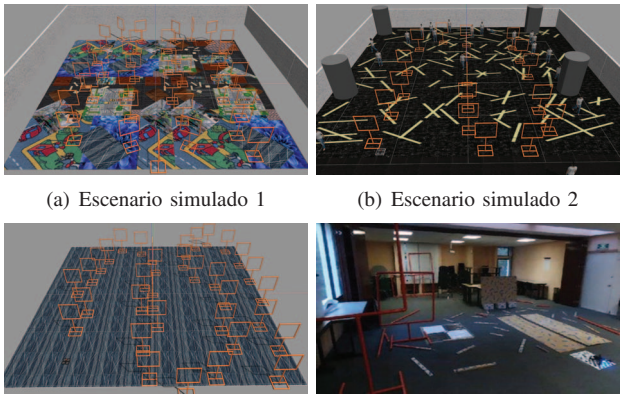


Fig. 10. Ejemplo de imágenes del conjunto de entrenamiento: la primera columna corresponde al primer escenario simulado, la segunda corresponde al segundo escenario y la tercera columna corresponde al tercer escenario simulado. Finalmente, en la cuarta columna se presentan las imágenes del escenario real.

En la Figura 11 mostramos los escenarios con diferentes condiciones dentro del simulador Gazebo, 34 puertas y 4 paredes son colocados en el primer escenario, 16 puertas con personas, columnas y paredes para el segundo y 32 puertas sin paredes en el tercer escenario, además se montaron 3 puertas para el escenario real. Las dimensiones de las puertas simuladas y reales son de 1×1 m con una base de 1.5 m de alto, permitiendo obtener mayor visibilidad de ellas al momento de su detección. Estos escenarios fueron utilizados para generar el conjunto de datos de entrenamiento. Por otro lado, realizamos una comparación de los modelos entrenados

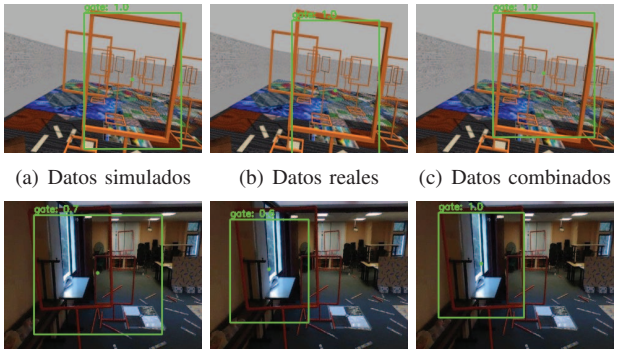
con datos sintéticos, reales y combinados para determinar si la combinación de estos ayuda a mejorar el desempeño del sistema. En la Figura 12 presentamos un ejemplo de comparación de los datos, obteniendo un mejor resultado en la métrica de confianza y en la detección de los cuadros delimitadores al usar los datos combinados.



(a) Escenario simulado 1 (b) Escenario simulado 2

(c) Escenario simulado 3 (d) Escenario real

Fig. 11. Escenarios usados para generar el conjunto de entrenamiento.



(a) Datos simulados (b) Datos reales (c) Datos combinados

(d) Datos simulados (e) Datos reales (f) Datos combinados

Fig. 12. Comparación de la influencia de los datos de entrenamiento.

IV. Evaluación

Los experimentos realizados se centran en evaluar la métrica de la confianza y el tiempo de predicción para cada cuadro delimitador usando nuestra red SSD5, y compararla con arquitecturas similares de detección (SSD7, SSD300 y smallVGG) con el fin de explotar al máximo la arquitectura de la red para lograr una detección robusta en el menor tiempo posible. Por lo anterior se realizaron 3 pruebas en dos computadoras diferentes: la primera y segunda prueba se realizó en la computadora externa Lenovo Y700 con GPU mientras que la tercera prueba fue en la *Intel Computer Stick* a bordo del dron con 2GB de RAM sin GPU Figura 13.

La primera prueba se realizó en un escenario simulado con 23 puertas, evaluando y comparando los resultados en tiempo real para las diferentes arquitecturas de detección presentadas en la Tabla II, cuya información es: velocidad promedio de detección (fps), tiempo promedio de predicción y la métrica promedio de la confianza de los cuadros delimitadores otorgada por las redes. Debido a la disminución

(a) Lenovo Y700. (b) *Intel Computer Stick.*

Fig. 13. Computadoras utilizadas para nuestro sistema.

de las capas el tiempo de predicción de la SSD5 es menor a las otras arquitecturas, reduciéndolo de 0.600 ms a 3.200 ms aproximadamente, y aumentando 5 fps la velocidad de detección. También, se conserva un valor de confianza alto al momento de detectar las puertas en un escenario simulado.

TABLA II
RESULTADOS EN UN ESCENARIO SIMULADO CON 23 PUERTAS

Red	Promedio (fps)	Tiempo (sec)	Tiempo de predicción (ms)	Confianza
SSD5	**85.222**	**36.220**	**12.073**	**96.187%**
SSD7	80.522	38.063	12.688	93.379%
SSD300	80.542	39.815	13.272	83.212%
SmallerVGG	73.017	46.058	15.355	87.042%

La segunda prueba se realizó en un escenario real con 4 puertas en el exterior para evaluar y comparar los resultados en tiempo real de las diferentes arquitecturas en la computadora Lenovo Y700. En la Tabla III se muestra los resultados obtenidos y su comparación con las diferentes arquitecturas, obteniendo una velocidad de detección mayor en la SSD5 con una reducción de 0.600 ms a 3.600 ms en el tiempo de predicción. Por otro lado, la métrica de confianza para este escenario se redujo un 2% a comparación del escenario simulado. Esto se debe a que el escenario de la prueba 1 es un entorno ideal para la detección al no presentar sombras ni cambios de iluminación.

TABLA III
RESULTADOS EN UN ESCENARIO REAL CON 4 PUERTAS EN EL EXTERIOR

Red	Promedio (fps)	Tiempo (sec)	Tiempo de predicción (ms)	Confianza
SSD5	**84.093**	**35.867**	**11.956**	**94.563%**
SSD7	80.149	37.612	12.537	92.217%
SSD300	79.762	37.746	12.582	83.688%
SmallerVGG	64.388	46.714	15.571	76.856%

La tercera prueba se desarrolló en el mismo escenario real con 4 puertas, evaluando y comparando los resultados de las distintas redes en la *Intel Computer Stick* cuyos resultados se muestran en la Tabla IV. El tiempo de predicción es relativamente grande con una velocidad menor a la obtenida con la Lenovo Y700 debido a que esta computadora no tiene una arquitectura GPU. En los resultados se observan que los datos son ligeramente similares entre las diferentes arquitecturas de SSD. Sin embargo, la SSD5 logra un mayor porcentaje de confianza que las otras redes, resultado que podría depender de 3 factores: el modo en el que se empleo el vuelo, el valor del umbral y la eliminación de las últimas dos

capas lo cual elimina 340 cuadros delimitadores en la salida de la red. Esto puede ocasionar que algunos cuadros eliminados tenga valores menores a 0.9 lo que llevaría a un porcentaje elevado en la confianza.

TABLA IV
RESULTADOS EN UN ESCENARIO REAL USANDO LA INTEL COMPUTER STICK

Red	Promedio (fps)	Tiempo (sec)	Tiempo de predicción (ms)	Confianza
SSD5	2.3692	1270.10	**423.367**	**94.463%**
SSD7	2.280	1311.42	437.14	91.893%
SSD300	**2.378**	**1263.78**	431.261	87.259%
SmallerVGG	None	None	None	None

En la Figura 14 se muestra el promedio de los fps de las predicciones obtenidas en cada entorno que se evaluó. Además se compara estos resultados con dos trabajos de la literatura que utilizan computadoras a bordo: [23] usa una *Intel UpBoard* cuya velocidad de detección es 10 fps superando 4 veces la obtenida por nosotros; [22] usa una NVIDIA TX2 para probar 3 arquitecturas basadas en SSD. Los resultados que reportan son: tiempo de predicción **462.04 ms**, velocidad **2.61 fps**, confianza **0.824** en VGG16; tiempo de predicción **84.21 ms**, velocidad **84.21 fps**, confianza **0.774** en AlexNet; tiempo de predicción **34.54 ms**, velocidad **28.95f fps**, confianza **0.755** en ADRNet; Si bien estos resultados han demostrado que nuestra SSD5 tiene un alto nivel de confianza, no son aptas para realizar un vuelo autónomo o completar el circuito a alta velocidad, al menos en la *Intel Computer Stick* con 2GB de RAM.

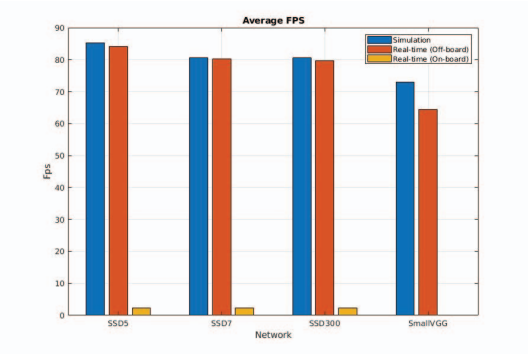
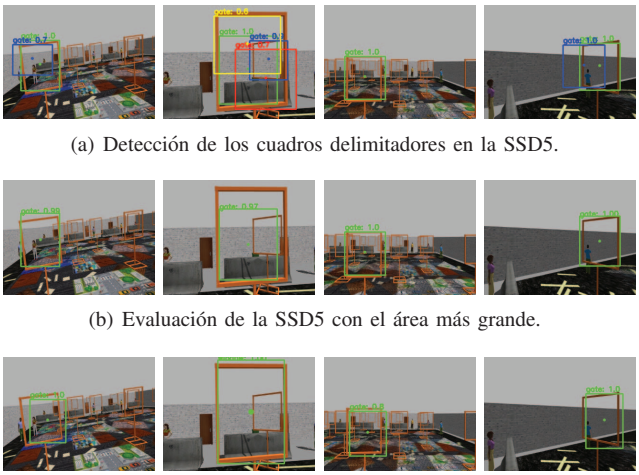


Fig. 14. Promedio de los fps para cada red.

V. RESULTADOS DE LA DETECCIÓN

En esta sección mostramos los resultados del detector propuesto utilizando la red SSD5 en tres escenarios diferentes en tiempo real. Estos experimentos fueron realizados para observar el comportamiento de la salida de la red en escenarios con diferentes condiciones, verificando las detecciones del objeto, los valores de confianza y el área que cubre el cuadro delimitador. Además, son comparadas con las detecciones de salida de la red SSD7.

Primer Escenario: Entorno simulado con 23 puertas. En la Figura 15 se muestran los cuadros delimitadores de la



(a) Detección de los cuadros delimitadores en la SSD5.

(b) Evaluación de la SSD5 con el área más grande.

(c) Evaluación con la SSD7.

Fig. 15. Resultados de detecciones en un escenario simulado.

SSD5 con sus valores de confianza (columna superior), y la comparación con la SSD7 al elegir el área mayor (columna media e inferior). Para esta primera prueba, se observa que las detecciones de salida de las redes son ligeramente similares obteniendo una detección de la puerta con un valor de confianza alta.

(a) Detección de los cuadros delimitadores en la SSD5.

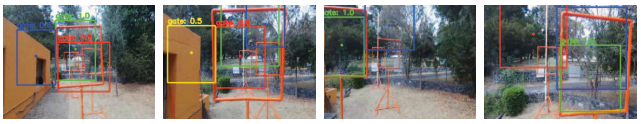
(b) Evaluación de la SSD5 con el área más grande.

(c) Evaluación con la SSD7.

Fig. 16. Resultados de las detecciones en un escenario interior.

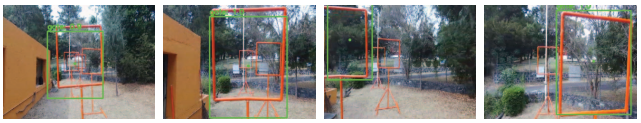
Segundo Escenario: Entorno real con 3 puertas en Interiores. En la Figura 16 se presenta un escenario cuyas puertas fueron colocadas aleatoriamente para observar las predicciones de los cuadros delimitadores de la SSD5 (columna superior). Para esta prueba observamos que los cuadros delimitadores tienen valores de confianza que varia entre 0.5 a 0.8, logrando detectar las puertas incluso en un escenario cuya iluminación es alta. Si bien, el área de cobertura en una de las imágenes es menor que la obtenida con la SSD7 (columna media e inferior) es notable que la detección de la SSD5 sigue presente, a pesar de probarse en un escenario distinto con condiciones de iluminación alta.

Tercer Escenario: Entorno real con 4 puertas en Exteriores. Para esta prueba presentamos los resultados de detección de



(a) Detección de los cuadros delimitadores en la SSD5.

(b) Evaluación de la SSD5 con el área más grande.



(c) Evaluación con la SSD7.

Fig. 17. Resultados de las detecciones en un entorno exterior.

la SSD5 en la computadora Lenovo Y700 y la *Intel Computer Stick*. En la Figura 17 y Figura 18 se muestran los cuadros delimitadores obtenidos con la red (columna superior) en las dos computadoras y la comparación con los cuadros de la SSD7 (columna media e inferior). Los resultados observados en la Figura 17 logran una detección adecuada de las puertas con un valor de confianza favorable a pesar de que el área de cobertura en alguna de ellas sea menor que la SSD7. Por otro lado, las detecciones obtenidas en la computadora a bordo (Figura.18), logran detectar las puertas en un ambiente exterior con un valor de confianza alto en algunas de ellas. Cabe recalcar que la velocidad de detección no supera los 3 fps, ocasionando que haya menos *frames* para predecir los cuadro delimitadores alrededor de la puerta.

(a) Detección de los cuadros delimitadores en la SSD5.

(b) Evaluación de la SSD5 con el área más grande.

(c) Evaluación con la SSD7.

Fig. 18. Resultados de las detecciones en un entorno exterior usando la computadora a bordo.

VI. Conclusiones

Hemos presentado un sistema de detección de puertas usando las predicciones de salida de la red SSD5 creada a partir de la modificación de la SSD7. Las predicciones de los cuadros delimitadores fueron probadas para detectar las puertas de un circuito en diferentes escenarios utilizando solo la cámara monocular de un dron para la aplicación de carreras de drones autónomos (ADR). El sistema propuesto se evaluó y comparó con otras redes del estado del arte en tres escenarios: simulación, interiores y exteriores, utilizando una computadora en tierra y la *Intel Computer Stick* a bordo del dron. Los resultados obtenidos con nuestro sistema de detección logra obtener una mayor velocidad de predicción y un tiempo promedio menor que las otras arquitecturas de aprendizaje profundo, obteniendo un porcentaje de confianza de 96.18% en simulación, 94.56% en exteriores y 94.46% en exteriores usando la *Intel Computer Stick*. A pesar de que nuestro sistema tiene una velocidad de 2.36 fps en la computadora a bordo, las predicciones de los cuadros delimitadores obtenidas por la SSD5 conserva un porcentaje de confianza alto al momento de detectar las puertas.

Puesto que la velocidad no es la adecuada para emplear un vuelo autónomo en un circuito de carrera de drones, la meta de este trabajo es explotar la arquitectura de la SSD para reducir los tiempos de detección conservando un porcentaje de confianza alto y comparar sus resultados con arquitecturas similares de SSD. Con la realización de este trabajo se pudo comprobar que la detección de objetos usando una arquitectura de aprendizaje profundo logra un resultado adecuado sin importar las condiciones de iluminación, sombras o cambios en el entorno. Sin embargo, planeamos realizar una modificación más profunda de la red e incluso explorar otras alternativas como la SSD en MobileNet o la actualización del *Framework* usando Tensorflow-Lite para aumentar la velocidad de detección a 24 fps en un computadora sin GPU. Usar la *Intel Computer Stick* con 4GB de RAM también esta contemplado como trabajo a futuro.

Referencias

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale imagrecognition," in *International Conference on Learning Representations*, 2015.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[4] H. Moon, Y. Sun, J. Baltes, and S. J. Kim, "The iros 2016 competitions [competitions]," *IEEE Robotics Automation Magazine*, vol. 24, pp. 20–29, March 2017.

[5] "Iros 2018 autonomous drone racing competition," *2018*, vol. [Online], Available: https://www.iros2018.org/competitions.

[6] H. Moon, J. Martinez-Carranza, T. Cieslewski, M. Faessler, D. Falanga, A. Simovic, D. Scaramuzza, S. Li, M. Ozo, C. De Wagter, G. de Croon, S. Hwang, S. Jung, H. Shim, H. Kim, M. Park, T.-C. Au, and S. J. Kim, "Challenges and implemented technologies used in autonomous drone racing," *Intelligent Service Robotics*, vol. 12, pp. 137–148, Apr 2019.

[7] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[9] P. Ferrari, "A keras port of single shot multibox detector." https://github.com/pierluigiferrari/ssd_keras, 2018.

[10] A. Bry, A. Bachrach, and N. Roy, "State estimation for aggressive flight in gps-denied environments using onboard sensing," 2012.

[11] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and imu," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 404–411, 2017.
[12] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 5774–5781, IEEE, 2017.
[13] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 2520–2525, IEEE, 2011.
[14] T. Sayre-McCord, W. Guerra, A. Antonini, J. Arneberg, A. Brown, G. Cavalheiro, Y. Fang, A. Gorodetsky, D. McCoy, S. Quilter, *et al.*, "Visual-inertial navigation algorithm development using photorealistic camera simulation in the loop," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2566–2573, IEEE, 2018.
[15] G. Kahn, T. Zhang, S. Levine, and P. Abbeel, "Plato: Policy learning using adaptive trajectory optimization," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 3342–3349, IEEE, 2017.
[16] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart, "Get out of my lab: Large-scale, real-time visual-inertial localization.," in *Robotics: Science and Systems*, 2015.
[17] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
[18] S. Jung, S. Cho, D. Lee, H. Lee, and D. H. Shim, "A direct visual servoing-based framework for the 2016 iros autonomous drone racing challenge," *Journal of Field Robotics*, vol. 35, no. 1, pp. 146–166, 2018.
[19] S. Li, E. van der Horst, P. Duernay, C. De Wagter, and G. C. de Croon, "Visual model-predictive localization for computationally efficient autonomous racing of a 72-gram drone," *arXiv preprint arXiv:1905.10110*, 2019.
[20] J. A. Cocoma-Ortega and J. Martinez-Carranza, "A cnn-based drone localisation approach for autonomous drone racing,"
[21] J. A. Cocoma-Ortega and J. Martínez-Carranza, "Towards high-speed localisation for autonomous drone racing," in *Mexican International Conference on Artificial Intelligence*, pp. 740–751, Springer, 2019.
[22] S. Jung, S. Hwang, H. Shin, and D. H. Shim, "Perception, guidance, and navigation for indoor autonomous drone racing using deep learning," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2539–2544, 2018.
[23] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: Learning agile flight in dynamic environments," *arXiv preprint arXiv:1806.08548*, 2018.
[24] E. Kaufmann, M. Gehrig, P. Foehn, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Beauty and the beast: Optimal methods meet learning for drone racing," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 690–696, IEEE, 2019.
[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
[26] Y. Li, H. Huang, Q. Xie, L. Yao, and Q. Chen, "Research on a surface defect detection algorithm based on mobilenet-ssd," *Applied Sciences*, vol. 8, no. 9, p. 1678, 2018.
[27] D. S. Levkovits-Scherer, I. Cruz-Vega, and J. Martinez-Carranza, "Real-time monocular vision-based uav obstacle detection and collision avoidance in gps-denied outdoor environments using cnn mobilenet-ssd," in *Mexican International Conference on Artificial Intelligence*, pp. 613–621, Springer, 2019.
[28] P. Poirson, P. Ammirato, C.-Y. Fu, W. Liu, J. Kosecka, and A. C. Berg, "Fast single shot detection and pose estimation," in *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 676–684, IEEE, 2016.
[29] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1521–1529, 2017.
[30] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, Japan, 2009.
[31] E. Forson, "Understanding ssd multibox-real-time object detection in deep learning," 2017.
[32] M. Monajjemi, "Bebop autonomy," 2015.

Aldrich A. Cabrera Ponce es estudiante de maestría en Ciencias Computacionales del INAOE y miembro del equipo de competencia de drones autónomos, con el cual ha obtenido diversos premios nacionales e internacionales. Su área de interés es el procesamiento de imágenes aéreas para la unión y creación de mosaicos.

L. Oyuki Rojas-Perez es estudiante de maestría en Ciencias Computacionales del INAOE y miembro del equipo de competencia de drones autónomos, con el cual ha obtenido diversos premios nacionales e internacionales.

Jesús A. Carrasco-Ochoa es investigador titular nivel A (equivalente a profesor asociado) en el Departamento de Ciencias Computacionales en el INAOE. Ha publicado más de 100 artículos sobre temas relacionados con el reconocimiento de patrones y la minería de datos, y ha coeditado 7 libros. Ha sido parte del comité organizador de varias conferencias internacionales y ha servido como parte del comité del programa de muchas conferencias y revistas internacionales. Sus intereses de investigación actuales incluyen reconocimiento lógico de patrones combinatorios, minería de datos, teoría de testores, selección de características y prototipos, análisis de texto y agrupamiento.

José Fco. Martínez-Trinidad recibió su grado de licenciatura en Ciencias Computacionales de la Facultad de Física y Matemáticas de la Universidad Autónoma de Puebla (BUAP), México en 1995, su grado de maestría en Ciencias Computacionales de la facultad de ciencias de la computación de la Universidad Autónoma de Puebla, México en 1997 y su doctorado por el Centro de Investigación Informática del Instituto Nacional Politécnico (CIC, IPN), México en 2000. El profesor Martínez-Trinidad ha coeditado 13 libros de memorias de congresos en la serie Lecture Notes publicada por Springer y ha publicado alrededor de 150 artículos de revistas y conferencias internacionales sobre temas relacionados con el reconocimiento de patrones.

J. Martínez-Carranza es investigador titular nivel A (equivalente a profesor asociado) en el Departamento de Ciencias Computacionales en el INAOE. El Dr. Martínez ha obtenido diversos premios internacionales en concursos de vuelos de drones. Actualmente se encuentra trabajando en el diseño de propuestas novedosas para el aprovechamiento de cámaras monoculares en drones aéreos.