Universidad Aeronáutica en Querétaro

UNIVERSIDAD AERONÁUTICA EN QUERÉTARO

DISEÑO Y SIMULACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA EL RECONOCIMIENTO DE OBSTÁCULOS Y GENERACIÓN DE TRAYECTORIAS DE VUELO DE UN CUADRICÓPTERO AUTÓNOMO

T E S I S

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN ELECTRÓNICA Y CONTROL DE SISTEMAS DE AERONAVES

PRESENTA

AXEL RAMIREZ LINAREZ

TUTOR

MCSD MOISÉS TORRES RIVERA

Querétaro, Marzo 2021

RESUMEN

GLOSARIO

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

INTRODUCCIÓN

1.1. Antecedente históricos

En diciembre de 1903, Orville Wright realizó el primer vuelo tripulado en la historia de la humanidad; no tuvo que pasar mucho tiempo para que el concepto de vehículo aéreo no tripulado tuviera un auge dentro de la comunidad científica y militar enfocada a la aviación.

Siendo estrictamente correctos, si se toma en consideración los vehículos capaces de generar sustentación y/o que cuentan con un medio para su control, se puede decir que el primer UAV de la historia, fue diseñado por el inglés Douglas Archibald, al fijar un anemómetro en la cuerda de un cometa, con lo cual fue capaz de medir la velocidad del viento a una altura de aproximadamente 1200 ft. Más tarde, en 1887, Archibald colocó cámaras en otra cometa, con lo cual desarrolló el primer UAV de reconocimiento, en el mundo.

Hablando específicamente de quadrotores, en 1907, Louis Breguet, un pionero francés de la aviación, junto con su hermano Jacques y su profesor Charles Richet, hicieron una demostración del diseño de un giroplano de 4 rotores. Este prototipo contaba con un motor de 30 caballos de fuerza que alimentaba los 4 rotores, cada uno de los cuales tenía 4 propelas y lograba elevarse hasta un máximo de 0.6 m.

Por otro lado, Etienne Oehmichen, un ingeniero francés, fue el primero en experi-

mentar con diseños de aeronaves de ala rotativa. En 1920, construyó y probó 6 diseños, el segundo de ellos tenía 4 motores y 8 propelas; el cuerpo de esta aeronave estaba hecho de tubos de acero y tenía 4 extremidades, en las cuales se alojaban cada uno de sus rotores con 2 propelas cada uno. En su momento, este diseño destacaba en su estabilidad y controlabilidad, y para la mitad de 1920 ya había realizado más de mil vuelos de prueba. En 1924 estableció un récord mundial al volar una distancia horizontal de 360 m.

Después, en 1922 el Dr. George de Bothezt e Ivan Jerome desarrollaron una aeronave con una estructura en forma de equis y rotores de 6 propelas en sus extremidades. Para 1923 habían realizado hasta 100 vuelos de prueba con una altura máxima de 5 m; sin embargo, este diseño era muy complejo y rígido, dificultando su movimiento lateral y suponiendo una carga de trabajo, para alimentar la maquinaria, demasiado alta para el piloto.

Además, en 1956 se desarrolló el Convertawings Model A, el cual fue pensado para formar parte de una línea de quadrotores grandes para uso civil y militar. Este prototipo contaba con dos motores, que controlan el giro de dos rotores, cada uno, a partir de lo anterior, el control de la aeronave se lograba al variar el empuje proporcionado por los rotores.

1.2. Motivación

1.3. Objetivos

1.3.1. Objetivo general

Proponer e implementar en simulación un algoritmo de detección de compuertas rectangulares mediante visión artificial para la definición y control de trayectoria de un cuadricóptero autónomo virtual.

1.3.2. Objetivo específicos

- Diseñar un algoritmo de visión artificial capaz de identificar compuertas rectangulares

- Diseñar un algoritmo de gestión de trayectorias de vuelo para un cuadricóptero autónomo
- Diseñar un ambiente de simulación en 3D de un circuito de vuelo basado en una carrera de cuadricópteros autónomos.
- Implementar un ambiente de Software in The Loop utilizando los algoritmos y el ambiente de simulación diseñados para verificar su comportamiento en conjunto

1.4. Justificación

Lejos de ser un atractivo visual y un espectáculo con fines de entretenimiento, las competencias de drones autónomos representan el estado del arte de la robótica aplicada a vehículos con sistemas de navegación autónoma. Lo anterior se debe a que la robótica siempre se ha enfocado a la automatización de los sistemas; es decir, que los robots sean capaces de realizar tareas o recorridos sin necesidad de intervención humana, para esta última parte, se necesita de algoritmos de percepción y navegación, con los cuales los vehículos puedan ubicarse en el espacio a partir de su sistema de sensores con el que cuentan (tales como tecnología a base de láseres, cámaras estereoscópicas, tecnología ultrasónica, etc.) para que después sea capaz de trazar una trayectoria o seguir una ruta previamente definida. Lo anterior ha adquirido una robustez bastante significativa en los últimos años, pues existe una gran cantidad de esfuerzos y colaboraciones dedicadas al desarrollo de los mismos, incluso, se han organizado eventos y competencias con el fin de estimular y potenciar el desarrollo de este tipo de sistemas; tal es el caso de la International Conference on Intelligent Robots and systems (IROS) y AlphaPilot, dos eventos de gran magnitud, creados con el objetivo de tratar, demostrar y fomentar los avances que se tienen en el área.

Por otro lado, la implementación de un sistema robótico autónomo no es una tarea sencilla, y debido a la poca competencia en el mercado también adquiere un costo elevado. Para que un robot sea capaz de percibir el ambiente a su alrededor y desplazarse por el mismo, es necesario implementar un sistema de software capaz de coordinar la adquisición de datos proveídos por los sensores y el conjunto de actuadores que permiten que el sistema se desplace. Muchas de las soluciones desarrolladas para afrontar

este desafío son privadas y no sé comparte con el público en general, además, algoritmos como el filtro de Kalman o un control PID son ampliamente utilizados en este tipo de sistemas, por lo que existe una posibilidad bastante alta de que todas estas soluciones implementen los mismos algoritmos, lo cual conlleva un desperdicio de tiempo y esfuerzo, sin mencionar que la calidad y eficiencia de cada implementación puede variar bastante. Debido a lo anterior, soluciones de código abierto como ROS (Robot Operating System; un framework de comunicaciones para el manejo y coordinación de procesos en sistemas robóticos), pueden representar el inicio de la implementación de un estándar en el área, pues al ser de software libre permiten que toda la comunidad utilice, mejore e inspeccione los algoritmos ya implementados.

Además, la realización de pruebas con el sistema físico, para verificar y validar los algoritmos desarrollados, representa un costo muy alto en la mayoría de sistemas con los que se trabaja en el área, por lo que también es necesario disponer de algún tipo de simulador que permita realizar las pruebas sin necesidad de utilizar el prototipo físico con el que se trabajará. Existen diferentes paradigmas de simulación en los que se puede simular la planta mediante software, tales como Hardware in the loop (HIL) y software in the loop (SIL). Ambos paradigmas representan una solución al problema planteado, proveyendo resultados muy cercanos a la realidad y con una arquitectura flexible, que permite realizar una gran cantidad de pruebas o incluso entrenar algoritmos relacionados con inteligencia artificial o redes neuronales, una vez más, sin depender del sistema físico.

A partir de todo lo anterior, en este trabajo se propone el diseño y la simulación de un algoritmo de visión por computadora para la detección de compuertas rectangulares, similares a aquellas utilizadas en las competencias de drones autónomos, para definir la trayectoria de vuelo de un dron autónomo con el fin de que sea capaz de completar un circuito definido. El entrenamiento e implementación se realizan dentro de un framework de simulación de SIL, y la gestión y comunicación entre procesos se implementan a partir de una arquitectura diseñada en ROS2, todo lo anterior bajo el paradigma de código abierto con el fin de aprovechar las ventajas previamente mencionadas y aportar los esquemas de configuración y diseño a la comunidad.

1.5. Planteamiento del problema

1.6. Contribuciones

1.7. Metodología

En primera instancia, se realiza una revisión bibliográfica intensiva acerca del estado del arte en cuanto a drones guiados por visión artificial, con el objetivo de visualizar las soluciones ya implementadas y conceptualizar la arquitectura necesaria para el sistema, sus componentes, los algoritmos de visión artificial empleados y la configuración necesaria para realizar la integración de todo lo anterior.

Posteriormente, se define el esquema general del proyecto estableciendo el algoritmo de visión artificial a utilizar, el ambiente de simulación, la interfaz de comunicación para la adquisición de datos e imágenes provenientes de la simulación, el modelo de dron a simular y las librerías necesarias para integrar el ambiente de simulación.

Establecido lo anterior, se implementa la arquitectura diseñada para el ambiente de simulación y se realizan vuelos manuales con el modelo de dron definido dentro de un circuito de prueba compuesto por compuertas. A partir de lo anterior, se extraen imágenes de la trayectoria de vuelo del dron por medio de una cámara simulada a bordo del modelo del dron; se utilizan las imágenes recopiladas para el entrenamiento del algoritmo de visión artificial.

Cuando el algoritmo de visión artificial proporciona una identificación adecuada del tipo de compuerta utilizada, se implementa el algoritmo con base en la arquitectura definida. Se realiza la validación del algoritmo en otro circuito de vuelo; a lo largo de la simulación, existe un intercambio de información constante entre la simulación y el algoritmo de visión artificial, la simulación envía imágenes obtenidas durante el vuelo del dron y el algoritmo de visión artificial las analiza, de tal forma que es capaz de identificar el centro de la compuerta más cercana y devuelve comandos de vuelo a la simulación para definir una ruta de vuelo que permita que el dron sea capaz de volar a través de la compuerta identificada y finalizar el circuito de forma autónoma.

Se reportan los resultados obtenidos y las posibles mejoras para el proyecto en su conjunto

1.8. Límites y alcances

1.8.1. Alcances

Se implementa la arquitectura de red neuronal convolucional (RNC) DeepPilot, la cual toma capturas de la única cámara a bordo del drone y predice cuatro comandos de vuelo ($\phi, \theta\psi, h$) como salida. La RNC es entrenada a partir de un dataset proveído por los autores de la arquitectura y que contiene un gran conjunto de imágenes obtenidas a partir de simulación, las cuales están asociadas a ciertos comandos de vuelo. La arquitectura es evaluada dentro de un entorno de simulación realizado en simulador Gazebo 11, en donde se virtualiza un circuito o pista de obstáculos compuesta por compuertas rectangulares de distintas alturas y color sólido, colocadas en distintas posiciones y orientaciones a lo largo del circuito. Se utiliza ROS2 para coordinar el envío de datos entre la simulación de Gazebo 11 y un nodo propio de ROS2 que contiene el algoritmo y arquitectura de DeepPilot. Además, se documenta de forma detallada la configuración realizada para la creación del ambiente de simulación, especificando la integración entre Gazebo, ROS y Python 3 para la evaluación de la arquitectura de DeepPilot. Por último, el proyecto en su conjunto se distribuye bajo el paradigma de código abierto.

1.8.2. Límites

A diferencia de las contribuciones y proyectos más populares dentro de la comunidad de las carreras de drones autónomos, en donde se utiliza ROS1 y Gazebo en su versión 9, en este proyecto se implementa la última versión estable de ROS2, Foxy, y la versión más actual del simulador Gazebo, al momento de escritura del trabajo, la versión 11. Por lo que es muy posible que algunos plugins tanto de ROS como de Gazebo, no se encuentren disponibles en estas versiones, lo que significa una limitante para la expansión a futuro del proyecto. Por otro lado, dentro del ambiente de simulación, no se evalúan condiciones de vuelo poco ventajosas como viento en contra, lluvia o cualquier otra condición climática adversa. Además, la complejidad en el arreglo de compuertas para el circuito es baja y se asume que las compuertas se encuentran de forma paralela a la cámara del dron, y es necesario que siempre exista una compuerta visible después de haber cruzado por otra.

1.9. Estructura de la tesis

CAPÍTULO 2

ESTADO DEL ARTE

Las competencias de drones autónomos han adquirido un grado alto de relevancia en la última década, dentro del marco teórico del presente trabajo se describe con profundidad el contexto histórico, así como la motivación y los requerimientos establecidos para dos de las competencias, más significativas, de drones autónomos, el IROS Autonomous Drone Race y el AlphaPilot AI Drone Innovation Challege. En este capítulo se presentan algunas de las soluciones propuestas en estas competencias, al igual que trabajos con enfoques más prácticos o que no se encuentran directamente relacionados con las carreras de drones autónomos.

Dentro de las competencias anteriormente mencionadas, existen dos problemas esenciales a los que se enfrentan los equipos que participan en estos retos, la detección de objetos y la gestión de trayectoria de vuelo a partir de la detección realizada. Los circuitos que tiene que completar los drones están compuestos por compuertas de distintas formas y tamaños, y en algunos casos, se adicionan obstáculos dinámicos, los vehículos desarrollados por los participantes tienen que ser capaces de detectar estos objetos haciendo uso exclusivo de los sensores con los que están equipados (cámaras, sensores ultrasónicos, tecnología láser, etc.).

En uno de sus artículos, cabrera

En [1] se propone un algoritmo de visión artificial basado en una red neuronal profunda con arquitectura de Single Shot Detection (SSD7); se eliminan la ultimas dos capas

para reducir el tiempo de predicción y se compara su desempeño con otras arquitecturas: YOLO3, tinyYolo y FRNS.

En [2] se presenta la motivación detrás de la creación de las competencias de drones autónomos, así como los algoritmos implementados en la edición 2018 del IROS

CAPÍTULO 3

MARCO TEÓRICO

3.1. Robot Operating System (ROS)

De acuerdo con su sitio oficial, ROS (del inglés, Robot Operating System) es un conjunto de herramientas y librerías de software para robótica desarrolladas por Open Robotics bajo el paradigma de software libre u open-source. Este entorno de trabajo destaca por contener algoritmos de última generación y herramientas de desarrollo avanzadas, que permiten la creación, implementación y reutilización de código para todo tipo de proyectos de robótica.

ROS 1, la primera versión del entorno de trabajo, surgió en 2007 como un ambiente de desarrollo para el PR2 robot, un robot de servicio diseñado para trabajar con personas y creado por la empresa The Willow Garage. Sin embargo, los creadores de ROS buscaban que el entorno de trabajo no se viera limitado a un solo modelo de robot, si no que, pudiera ofrecer herramientas de software para más tipos y modelos de robots, por lo que ROS adquirió varias capas de abstracción mediante la implementación de interfaces para el manejo de mensajes, lo que dio lugar a que el software desarrollado mediante ROS pudiera ser reutilizado en más robots.

Algunas características que destacan en esta etapa temprana de ROS son: Gestión de un solo robot Sin requerimientos de aplicación en tiempo real Excelente conectividad a la red Usado principalmente en el ámbito académico y de investigación

Hoy en día, ROS es utilizado en una amplia gama de robots, desde robots con ruedas y con forma humanoide, hasta brazos industriales, vehículos aéreos y mucho más. Sin embargo, ha pasado bastante tiempo desde el lanzamiento de la primera versión de ROS, y las necesidades y estándares de la industria han cambiado al igual que el paradigma y la filosofía detrás del desarrollo de ROS.

A partir de lo anterior, en 2014 una nueva versión de ROS con un enfoque y estructura distinta es anunciada por Open Robotics. ROS 2 surge como un completo rediseño para lo que había sido el entorno de trabajo hasta entonces, con esta reestructuración se busca cubrir necesidades y funcionalidades que no habían sido consideradas con ROS 1 pero que habían sido exigidas por la comunidad y la industria. Lo anterior dio lugar al desarrollo de un nuevo conjunto de paquetes

BIBLIOGRAFÍA

[1] Aldrich A Cabrera-Ponce, Leticia Oyuki Rojas-Perez, Jesus Ariel Carrasco-Ochoa, Jose Francisco Martinez-Trinidad, and Jose Martinez-Carranza. Gate detection for micro aerial vehicles using a single shot detector. *IEEE Latin America Transactions*, 17(12):2045–2052, 2019.

[2] Hyungpil Moon, Jose Martinez-Carranza, Titus Cieslewski, Matthias Faessler, Davide Falanga, Alessandro Simovic, Davide Scaramuzza, Shuo Li, Michael Ozo, Christophe De Wagter, et al. Challenges and implemented technologies used in autonomous drone racing. *Intelligent Service Robotics*, 12(2):137–148, 2019.