Visión Computacional Aplicado a Drones para Actividades Recreativas

Sistemas Inteligentes

Sánchez Patiño Natalia
Rosas Otero Mario

Profesor: Dr. David Tinoco Varela

Licenciatura en Tecnología
Facultad de Estudios Superiores Cuautitlán
Universidad Nacional Autónoma de México

05 - 12 - 2020

-Proyecto para Sistemas Basados en Redes Neuronales: Aplicación de modelos de reconocimiento a un dron como método de interacción.-

Resumen

Visión Computacional Aplicado a Drones para Actividades Recreativas En este proyecto se realiza la construcción de un sistema capaz de integrar visión computacional con el manejo, control e interacción con un dron, de forma que se tenga un control semi-autónomo de el, esto principalmente con fines recreativos y educativos que permitan conocer el funcionamiento de los movimientos de un dron y la forma en que se pueden utilizar las señales e imágenes mandas por este al sistema interpretándolas y realizando acciones acordes en consecuencia. El correcto funcionamiento de las versiones construidas requiere de elementos tanto de hardware como de software, e involucra mantener distintos archivos, de ejecución de código, y bases de datos, así como instalación de distintas bibliotecas y dependencias. Dichos archivos en conjunto permiten tener la información necesaria para interactuar con el usuario con el objetivo de facilitar la comunicación y precisión de movimientos. Con esto realizamos la propuesta a una pequeña escala de procesar las imágenes obtenidas por el dron durante su vuelo, utilizando distintas herramientas de código abierto para poder integrar un programa funcional, permitiendo implementar la comunicación entre las distintas partes que integran al sistema. Esto de manera que pueda utilizarse de forma lúdica y cotidiana cuando se desee poder experimentar con este tipo de dron en nuestro caso un DJI Tello. Todo esto es posible ejecutarlo de forma off-line. Dentro de las posibles extensiones del proyecto se plantea el funcionamiento del sistema dentro de una tarjeta de desarrollo como Raspberry Pi, a fin de dar portabilidad al sistema e integrarlo a otros dispositivos, con posibilidad de extensión de características. Es posible obtener el código total del proyecto en: https://github.com/NM-Labs/DronFollowMe

Índice

1

Introducción

El proyecto desarrollado plantea la construcción de un sistema capaz de reconocer letras escritas en el aire, frente a la cámara montada en el dron programable Tello, y actuar en consecuencia conforme a rutas específicas para cada letra, entre otros movimientos. Dichas características sirven para incrementar la interacción que existe entre el usuario y el quadracóptero. Esto puede implicar el poder desarrollar más actividades haciendo uso de este tipo de herramientas, de acuerdo también a los deseos e intereses del usuario. Este es capaz de realizar diferentes series de movimientos, sin necesidad de controlarlo mediante un control remoto.

1.1 | Motivación

Dentro del ámbito escolar, comercial e industrial, el término Inteligencia Artificial se ha vuelto una constante, relacionándose en la mayoría de las ocasiones con tecnología de punta, y por algunos considerada potencialmente peligrosa, debido a que se ha planteado la posibilidad de alcanzar la singularidad tecnológica, esto a causas del continuo de desarrollo de áreas como la de Inteligencia Artificial, el punto de singularidad se identifica donde los robots, y las computadoras serán capaces de desarrollar mejoras a si mismos, y con ello establecer dominio sobre la humanidad. Sin embargo, los preceptos bajo los que aun trabaja el área, están muy alejados de alcanzar dicho punto, principalmente basan su funcionamiento en herramientas matemáticas, para aproximar soluciones a la descripción numérica de los problemas a los que se trata de dar solución aplicando conceptos del área. Una de las herramientas dentro de la Inteligencia Artificial, que se ha popularizado en los últimos años son las Redes Neuronales, y sus distintas variantes, esto debido a su gran capacidad de aproximación, y alta efectividad en el campo práctico. Una variante de las redes neuronales, son las Redes Neuronales

Convoluciones, un diseño adaptado a un tipo de información en específico: las imagen. Debido a ello, este tipo de redes son ampliamente utilizadas en el campo de visión computacional, para resolver problemas de principalmente de clasificación o de generación. Es por ello que en este trabajo nos hemos enfocado un poco mas a su estudio mediante la puesta en práctica de la misma, y mostrando algunas de las propuestas a las que se han aplicado este tipo de redes.

1.2 | Objetivos

- **Objetivo general:**
  Construir de forma práctica un sistema de reconocimiento y clasificación utilizando redes neuronales utilizando los conceptos vistos en clase. Y ser capaz de implementarlo dentro de hardware, en esté caso un dron, brindando herramientas interactivas, dentro del sistema.

- **Objetivos Particulares:**

  - Realizar una revisión teórica acerca del concepto de redes neuronales.
  - Buscar la aplicabilidad de las redes neuronales como estructura de clasificación de datos, como estructura general de funcionamiento y control para dirigir una conversación.
  - Comprender de manera teórica y práctica, mediante un lenguaje de programación, el funcionamiento del sistema propuesto.
  - Complementar y desarrollar habilidades de programación en el lenguaje Python como herramienta para la creación y ejecución del sistema.
  - Verificar que el sistema cuente con un funcionamiento práctico, así como identificar e implementar el sistema utilizando herramientas abiertas a todo publico.
  - Comprobar de manera aplicada, conceptos vistos en clase y conjuntándolo con conceptos encontrados durante la búsqueda de información relacionada a temas de la clase.
  - Interaccionar de una forma distinta con un dron, buscando la ejecución de movimientos de forma autónoma.
  - Reforzar los conocimientos teóricos adquiridos para la su implementación mediante las herramientas de programación, permitiendo su adecuado análisis.

1.3 | Solución Propuesta

Para la implementación de este proyecto se decidió trabajar en el lenguaje de programación Python, para poder establecer un canal de comunicación con el Dron DJI Tello, que permite programarse mediante comandos de código. Se propone, realizar un entrenamiento utilizando redes neuronales para lograr clasificar movimientos dados por el usuario. El principal escenario realizado consiste en la ejecución e interacción mediante movimientos con el dron volando frente al usuario. Pero también se contempla poder interactuar únicamente mediante con una computadora. Se buscó herramientas que fueran accesibles para cualquier persona, si bien, estás son de libre acceso, pueden tener requerimientos de capacidad computacional un poco altos. Más adelante se esperaría poder contar con herramientas de uso más sencillo.

2

Marco teórico

2.1 | Redes neuronales

Las redes neuronales son uno de los paradigmas de programación más hermosos jamás inventados. En el enfoque convencional de la programación, le decimos a la computadora qué hacer, dividiendo los grandes problemas en muchas tareas pequeñas y definidas con precisión que la computadora puede realizar fácilmente. Por el contrario, en una red neuronal no le decimos a la computadora cómo resolver nuestro problema. En cambio, aprende de los datos de observación y descubre su propia solución al problema en cuestión Nielsen (2015).

El reciente resurgimiento de las redes neuronales, la revolución del aprendizaje profundo, es cortesía de la industria de los juegos de computadora. Las imágenes complejas y el ritmo rápido de los videojuegos actuales requieren un hardware que pueda mantenerse al día, y el resultado ha sido la unidad de procesamiento de gráficos (GPU), que incluye miles de núcleos de procesamiento relativamente simples en un solo chip. Los investigadores no tardaron en darse cuenta de que la arquitectura de una GPU es notablemente parecida a la de una red neuronal Hardesty (2017).

Estas redes neuronales son redes de múltiples capas de neuronas que se usan para clasificar cosas, hacer predicciones, máscaras, etc. A continuación (figura 2.1) se muestra el diagrama de una red neuronal simple con cinco entradas, 5 salidas y dos capas ocultas de neuronas.

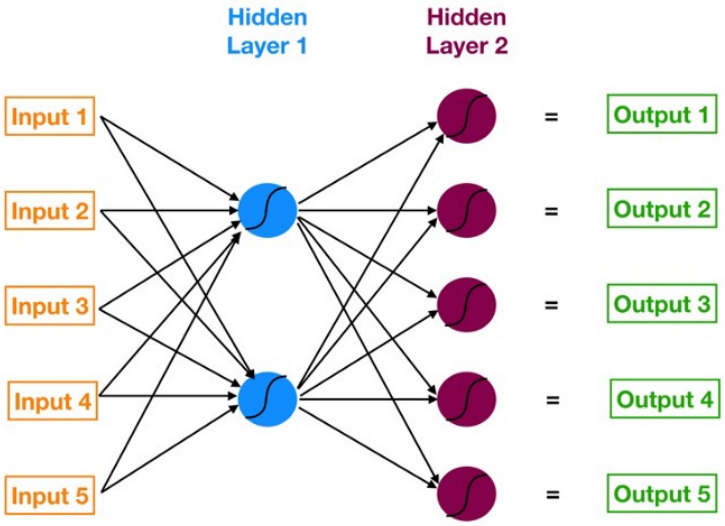

Figura 2.1: Red neuronal con dos capas ocultas.

Como cualquier otro modelo, intenta hacer una buena predicción. Tenemos un conjunto de entradas y un conjunto de valores objetivo, y estamos tratando de obtener predicciones que coincidan lo más posible con esos valores objetivo. Las redes neuronales más complejas son solo modelos con más capas ocultas y eso significa más neuronas y más conexiones entre neuronas. Y esta red más compleja de conexiones (y pesos y sesgos) es lo que permite que la red neuronal "aprenda" las complicadas relaciones ocultas en nuestros datos.

En una red neuronal, cambiar el peso de cualquier conexión (o el sesgo de una neurona) tiene un efecto de reverberación en todas las demás neuronas y sus activaciones en las capas posteriores, eso es porque cada neurona en una red neuronal es como su propio pequeño modelo.

2.2 | Visión Computacional

La visión por computadora es el campo de la ciencia de la computación que se enfoca en replicar partes de la complejidad del sistema de visión humana y permitir que las computadoras identifiquen y procesen objetos en imágenes y vídeos de la misma manera que lo hacen los humanos. Hasta hace poco, la visión por computadora solo funcionaba en una capacidad limitada.

Gracias a los avances de la inteligencia artificial y las innovaciones en el aprendizaje

profundo y redes neuronales, el campo ha sido capaz de tomar grandes saltos en los últimos años y ha sido capaz de superar los seres humanos en algunas tareas relacionadas con la detección y el etiquetado de objetos.

Uno de los factores que impulsan el crecimiento de la visión por computadora es la cantidad de datos que generamos hoy que luego se utilizan para entrenar y mejorar la visión por computadora.

En cierto nivel, la visión por computadora tiene que ver con el reconocimiento de patrones. Entonces, una forma de entrenar a una computadora para que comprenda los datos visuales es alimentarla con imágenes, muchas imágenes, miles, millones si es posible que hayan sido etiquetadas, y luego someterlas a diversas técnicas de software o algoritmos que permiten que la computadora las busque patrones en todos los elementos que se relacionan con esas etiquetas.

2.3 | Redes Neuronales Convolucionales

Los avances en la visión por computadora con aprendizaje profundo se han construido y perfeccionado con el tiempo, principalmente sobre un algoritmo particular: una red neuronal convolucional. Este es un algoritmo de aprendizaje profundo que puede tomar una imagen de entrada, asignar importancia (pesos y sesgos aprendibles) a varios aspectos / objetos de la imagen y poder diferenciar uno de otro. El preprocesamiento requerido en una ConvNet es mucho menor en comparación con otros algoritmos de clasificación. Mientras que en los métodos primitivos los filtros se diseñan a mano, con suficiente entrenamiento, ConvNets tiene la capacidad de aprender estos filtros / características.

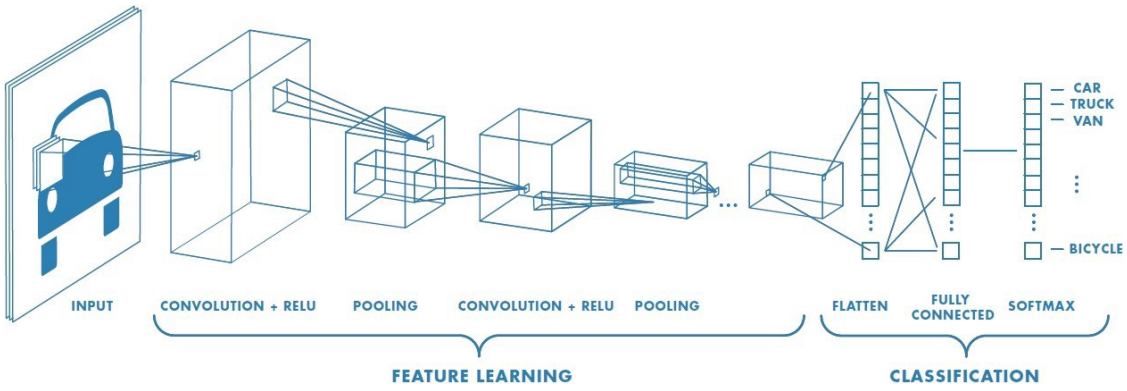

Figura 2.2: Red neuronal con capas convolucionales

La arquitectura de una ConvNet es análoga a la del patrón de conectividad de las neuronas en el cerebro humano y se inspiró en la organización de la corteza visual. Las neuronas individuales responden a los estímulos solo en una región restringida del campo visual conocida como campo receptivo. Una colección de estos campos se superponen para cubrir toda el área visual.

2.4 | Drones

La ciencia en general y la robótica en particular se han convertido en partes cada vez más importantes de la educación moderna, y es importante que los estudiantes obtengan una buena base en ellas y descubran lo que las hace tan apasionantes. Los mejores drones del mercado son hermosos para volar con una estabilidad excepcional. Tienen excelentes cámaras para filmar y vienen con kits de desarrollo de software (SDK) para la programación. Sin embargo, estos drones pueden ser relativamente caros, por lo que no son ideales para aprender a codificar. Por otro lado, los drones educativos incluyen clases, plan de estudios, material o cursos de drones en línea. Existen diferentes opciones para el acercamiento a este tipo de tecnologías. La empresa DJI es una de ellas. Uno de sus principales productos para la educación es el Tello EDU que es un dron programable perfecto para uso educativo. Donde se puede aprender a programar en Scratch, Python y Swift.

3

Elementos Utilizados y Proceso de Construcción

Para la creación de nuestro proyecto se utilizó un conjunto de componentes tanto de software como de hardware, para la parte de clasificación de letras del proyecto, que nos permite proporcionar la imagen de una letra y que el sistema prediga o clasifique la letra dada correctamente. Utilizando lo realizado en la primera parte continuamos con la de escritura en el aire, donde el usuario puede dibujar una letra frente a la cámara y el sistema debe de reconocerla. Después se añade el estimado de posición de la persona y finalmente todo esto se integra con la conexión hacia el dron y sus respectivos movimientos. Para el desarrollo de todo este proyecto se contemplan los siguientes elementos:

3.1 | Hardware

- Parte D: Dron
  - Dron DJI Tello
  - Objeto verde fosforescente
- Parte B: Escritura
  - Cámara Web
  - Objeto verde fosforescente
- Parte C: OpenPose
  - Cámara Web

3.2 | Software

- Parte A: Clasificación de letras y Parte B: Escritura
  - Python
  - Dataset de EMNIST Cohen et al. (2017)
  - Librerías de python para inteligencia artificial como Tensorflow y PyTorch, así como sus dependencias.
  - Un ambiente independiente para Python 3 en anaconda para la descarga de librerías necesarias.
- Parte C: OpenPose
  - OpenPose
  - CMake
  - OpenCV
  - Visual Studio
  - Controladores de CUDA y CuDNN (en caso de usar GPU)
- Parte D: Dron
  - Librería djitellopy

3.3 | Instalaciones

Para el correcto funcionamiento de la Parte C del sistema se requiere hacer un cuidadoso proceso de instalación, y también depende mucho del sistema operativo donde se este trabajando. En el siguiente link se detalla mucho más de esta secuencia de pasos: `https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/installation/0_index.md#compiling-and-running-openpose-from-source`. Es importante tomar en cuenta que para que se pueda usar dentro de otros códigos, es necesario construir el sistema desde el código fuente, ello debido a que se requiere su uso dentro de Python.

3.4 | Funcionamiento del Sistema

En este proyecto debido a las condiciones de trabajo la mayor parte del trabajo fue experimental, ya que es preciso realizar muchos experimentos de calibración, el funcionamiento no tiene una linea definida, y es cambiante conforme a las condiciones que se presentan al momento de utilizar el sistema. Existen diversos factores a considerar. Como ya se ha precisado anteriormente, la idea de este proyecto, es tomar un dispositivo de hardware con relativa fácil accesibilidad, y utilizarlo como medio aplicativo para mostrar algunas de las aplicaciones que se le puede dar a las redes neuronales convoluciones, y que en este caso se utilizan para desarrollar un sistema de uso recreativo, pero en realidad pueden aplicarse a zonas de impacto directo en la productividad industrial, cuestiones de salud, aprendizaje, entre otros.

De forma específica se tomo un dron comercial de dimensiones pequeñas, cuyo nombre comercial es **DJI Tello**, con ello se realizaron tres implementaciones de código, código de control del dron mediante el lenguaje de programación Python; código de aplicación de reconocimiento de letras mediante dibujo en el aire y con ello su respectivo mapeo, al sistema de control del dron para la respectiva realización de una rutina trazando el en el aire lo expresado en el dibujo realizado por el usuario, ello mediante el lenguaje de programación Python, y mediante el uso de redes neuronales para el reconocimiento de lo dibujado en el aire; el ultimo código implementado fue donde se utiliza un modelo ya entrenado para reconocimiento de pose humana, conocido como OpenPose, y se aplica para establecer ordenes de control e interacción mediante gestos con el cuerpo humano.

3.4.1 | Parte A y B: Clasificación de Letras y Escritura en el Aire

Para esta parte del sistema se tomo una aproximación donde mediante la adquisición de imágenes en tiempo real a través de una cámara se realiza el mapeo de movimiento en tiempo real de un objeto con un color característico, en este caso verde fosforescente. Dicho movimiento del objeto característico, debe ser el intento de trazo de una letra en el aire por parte el usuario. Esto se realiza mediante el procesamiento de imágenes a través de sus capas, extrayendo la sección con el color característico del marco o imagen en cada tiempo de un vídeo, que puede descomponerse en una secuencia de imágenes. Una que que se tiene identificado un color característico, se puede realizar la activación mediante un medio físico, como el pulsar un botón o una tecla para comenzar a marcar el camino o trazo que el usuario realiza con dicho objeto, para ello se obtiene la más-

cara de la zona específica donde se ubica el objeto, y se marca el camino que el usuario realiza con dicho objeto. Cuando el usuario ha finalizado el trazo, puede nuevamente mediante el pulsar de una tecla detener el grabado del trazo. Una vez que se tiene el trazo completo realizado por el usuario acota la zona donde se ubica el trabajo, mediante una búsqueda de zonas con el color en especifico, se ubica la de mayor área, y se rellenan espacio vacíos entre el trazo, ello para tener un trazo sólido. Una vez que se ubica por completo la zona del trazo, y se delimita, la misma se corta y se ajusta con medidas específicas para poder posteriormente procesarse.

Una vez que se le da el tratamiento adecuado, la imagen ya procesada, se evalúa en una red neuronal convencional previamente entrenada para el reconocimiento de letras mediante el uso de una base de datos de caracteres escritos a mano conocida como: EMNIST, y se obtiene la clasificación del trazo realizado por usuario, convergente a una letra, con ello el resultado es mostrado en la pantalla de transmisión de los datos, es decir imágenes obtenidas por la cámara. El modelo neuronal que se utiliza para clasificación de trazos en el aire como letras, fue construido dos veces, para probar eficiencia en el entorno real, y se realizo mediante el uso de dos bibliotecas para desarrollo de Deep Learning implementadas para uso mediante el lenguaje de programación Python, con cada una se creo un modelo. Las bibliotecas utilizadas fueron **TensorFlow** mediante su API, **Keras** para construir uno de los modelos, y **Pytorch**, para construir el segundo modelo.

3.4.2 | Parte C: OpenPose

OpenPose es un software de código abierto el cual permite mediante modelo ya entrenados realizar el reconocimiento de formas humanas en imágenes, vídeos almacenados o en tiempo real, tiene una alfa efectividad tanto teórica como práctica para esta tarea. Al rededor de esta tecnología se han desarrollado diversos tipos de sistemas, que se privilegian de su uso. Una de las aplicaciones fuertes, a las que se ha aplicado OpenPose es para el control a distancia. Este es el caso presentado en este proyecto, realizando una implementación, que requiere de la construcción de su código fuente, en vez del uso de su versión portable, ello debido a que se quiere extraer información del reconocimiento de los puntos en poses humanas realizados además de el procesamiento y la ejecución de control del dron mediante señales de posición humana todo ello a través del lenguaje de programación Python.

Dentro de un script se realiza el control del dron, y se ejecuta el modelo de análisis de pose Humana de openpose, se obtienen los puntos identificados de la pose humana, para posteriormente realizar evaluaciones de los valores en la relación de la posición espacial de las partes de cuerpo para determinar si el usuario esta intentando dar una señal de control mediante su posición corporal. Para ello, si el sistema no tiene identificada a una persona, realiza una búsqueda mediante giros, de una forma humana, una vez que la identifica, se mantiene enfocando hacia la persona, y realizando movimientos conforme a las instrucciones que brinda el usuario mientras cambia de posición. Los movimiento que realiza el dron cambian conforme a se actualiza la información de los puntos provenientes de la pose de una persona.

3.4.3 | Parte D: Dron

El control el dron DJI Tello, se realiza mediante el uso de la biblioteca djitellopy, ya que la misma cuenta ya con una amplia variedad de funciones para control del python, que pueden ser mandadas en tiempo real al dron mediante la ejecución del código en un dispositivo conectado a través de WI-FI al dron, esta biblioteca permite explotar la capacidad del dron incluso mas que en su funcionalidad de control normal (mediante una aplicación para dispositivos móviles). Debido a que se puede establecer secuencias de pasos y movimientos mas complejas, y se tienen mas combinaciones de métodos de entrada para ejecutar dichas acciones. La biblioteca mencionada ya cuenta con una amplia variedad de acciones que se pueden mandar al dron, sin embargo, debido a los requerimientos específicos de nuestros sistemas se tuvieron que construir algunas estructuras, a forma de rutinas, para que el dron reaccionara de forma adecuada ante diversas situaciones, tal como la identificación de una letra proveniente del trazo de un usuario, o alguna señal de control identificada mediante el uso de Openpose. Con ello se realizaron integraciones a los respectivos sistemas tanto el de identificación de letras en el aire como el de reconocimiento de pose humana para integrar las funcionalidades que los mismos proveen de identificación de patrones y con ello ejercer el movimeinto y control sobre el dron.

4

Resultados obtenidos

4.1 | Uso de redes neuronales

Para la parte principal de este proyecto se realizaron dos entrenamientos con el mismo conjunto de datos y la misma arquitectura, pero realizada por medio de bibliotecas diferentes en Python. El primer entrenamiento se hizo utilizando la biblioteca de Tensorflow, fue un entrenamiento de 10 épocas que se realizo en 27 minutos aproximadamente, haciendo uso de una GPU GTX1060, y se alcanzo una exactitud del 90.90% en el set de validación, mientras que al utilizar la biblioteca de Tensorflow, con misma arquitectura y los mismos hiper-parámetros se alcanzo una exactitud del 91.50%, pero aumentando el tiempo de ejecución a 32 minutos.

Se puede observar que a pesar de la poca cantidad de épocas, se obtuvieron buenos resultados, esto debido a una buena elección de parámetros para el modelo.

4.2 | Practicidad de las bibliotecas y softwares requeridos

En el presente proyecto se hizo uso de una gran cantidad de librerías de libre uso con resultados y versatilidad de condiciones sorprendente, sin embargo el acondicionamiento de los ambientes dentro del sistema computacional como su correcta ejecución pueden llegar a ser bastante complicados y se deben tomar en cuenta muchas precauciones para implementarlo.

Una vez tomado esto en cuenta, es importante mencionar que son librerías que pueden facilitar gran parte del proceso a la larga, puesto que contienen una gran canti-

dad de funciones y características que pueden ser aplicadas o utilizadas para la obtención de información de una manera muy sencilla.

4.3 | Uso del hardware

Debido a que este proyecto fue altamente práctico al integrar el uso del Dron, se realizaron una multitud de pruebas, sin embargo estas duraban muy poco tiempo al tener la limitante del uso de la batería, condiciones cerradas para maniobrar y lo cambiante de la luz durante el transcurso del día.

Durante las pruebas se pudo observar que cuando el dron comienza a trabajar ejecuta todos los movimientos e instrucciones perfectamente, pero a medida que pasa el tiempo, y sucede de manera rápida, puede perder potencia y precisión a la hora de volar y maniobrar, esto puede dificultar la interacción con el y es importante dejar reposar las baterías y el dron anter de cargarlos y de volverlos a utilizar.

5

Conlusiones

En este proyecto se realizó la implementación de un sistema de interacción con un dron mediante el uso de redes neuronales y visión computacional, donde es posible dar instrucciones por medio de dibujar letras frente a una cámara o hacer gestos utilizando las manos. Para esto se uso el lenguaje de programación Python en archivos *.py* y *.ipynb* el cual contiene características que nos ayudan a realizar el manejo de datos de forma eficiente y a preservar una sintaxis clara de la estructura de funcionamiento general, por otro lado el uso de los notebooks de python nos observar paso a paso los procesos realizados, sobre todo en las dos primeras partes de proyecto, pero para la parte final es mucho más eficiente utilizar únicamente los archivos .py pudiendo observar lo que sucede en la cámara del dron a través de la transmisión de vídeo a la computadora y a los mensaje enviados a la terminal. La interfaz de uso es entendible al usuario mediante mensajes e instrucciones escritas dentro del código, pero para llegar a ser una interfaz completa aún faltaría realizar mucho más trabajo sobre ello, de manera que pueda llegar a ser intuitivo para todos. El objetivo principal de este proyecto se cumplió al poder realizar un sistema de control interacción integrando el uso de redes neuronales a su desarrollo. Esto desde su concepción teórica nos permitió tener un acercamiento a los conceptos vistos en clase y que optimizan en gran medida los esquemas generales de funcionamiento de sistemas dentro del área de Inteligencia Artificial. Con lo cual se ve de forma directa una aplicabilidad de estos conceptos a soluciones tecnológicas actuales y con gran popularidad como lo son el vuelo semi-autónomo de drones recreativos o con propósitos educacionales. Durante el desarrollo se encontraron algunas limitantes como la compatibilidad entre bibliotecas, distribuciones, sistemas operativos, controladores, conexiones a la red, entre otras cosas, debido a eso el problema tuvo que ser acotado más de lo que se había planeado en un inicio, restringiendo ciertos movimientos por parte del usuario

y también el tiempo de vuelo. Respecto a las pruebas realizadas se lograron resultados bastante precisos a la hora de su ejecución ya integrada, sin embargo el tema de accesibilidad a ello es aún una cuestión que queda por resolver. Esto es posible lograrlo encontrando alternativas a ciertas bibliotecas utilizadas actualmente y aumentando la complejidad del código añadiendo m´s funciones que permitan realizar las actividades de manera más fluida y ordenada.

5.1 | Posibles extensiones del proyecto

Hay gran cantidad de manera de ampliar el alcance de este proyecto, comenzando con el tema de accesibilidad a el, el añadir funciones que permitan un mayor rango de comportamientos e interacción, aumentar la autonomía del vuelo del dron y brindar una mayor flexibilidad respecto a las condiciones de uso de nuestro sistema.

5.2 | Comentarios Finales

Con la realización de este proyecto, quedamos satisfechos con los resultados obtenidos dadas las condiciones y herramientas utilizadas, con la experimentación realizada, el cumplimiento de todos los objetivos y la experiencia de desarrollo adquirido. Logramos fortalecer nuestras habilidades de desarrollo específicamente en Python y nuestra experiencia acerca de la integración de herramientas en línea para la implementación de este tipo de sistemas. Nos apoyamos de preguntas realizadas en foros de programación y en la documentación dada por distintas librerías utilizadas. Dada la complejidad de las acciones que realiza el sistema fueron utilizadas librerías poco comunes como:

- OpenPose
- DjiTelloPy
- Tensorflow
- PyTorch
- OpenCV

Referencias

Mr. Brijain, R Patel, Mr. Kushik, and K Rana. A survey on decision tree algorithm for classification.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

Menal Dahiya. A tool of conversation: Chatbot. *International Journal of Computer Sciences and Engineering*, 5 (5):158–161, 2017.

Larry Hardesty. Explained: Neural networks ballyhooed artificial-intelligence technique known as "deep learning" revives 70-year-old idea. 2017.

Davuluri Hemanth. Decision Trees Explained With a Practical Example – Towards AI — The Best of Tech, Science, and Engineering, may 2020. URL https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53.

Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, 2015.

J. R. Quinlan. Learning decision tree classifiers. *ACM Comput. Surv.*, 28(1):71–72, March 1996. ISSN 0360-0300. doi: 10.1145/234313.234346.

A. Shah, B. Jain, B. Agrawal, S. Jain, and S. Shim. Problem solving chatbot for data structures. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 184–189, 2018. doi: 10.1109/CCWC.2018.8301734.