Identificación y Detección de la Distancia de la Persona más Próxima a un Drone

- Valentina Hernández
- Luis Felipe Gutierrez
- Manuel Alejandro Rojas

Técnicas de Inteligencia Artificial
Facultad de Ingeniería
Universidad Nacional de Colombia

Contenido

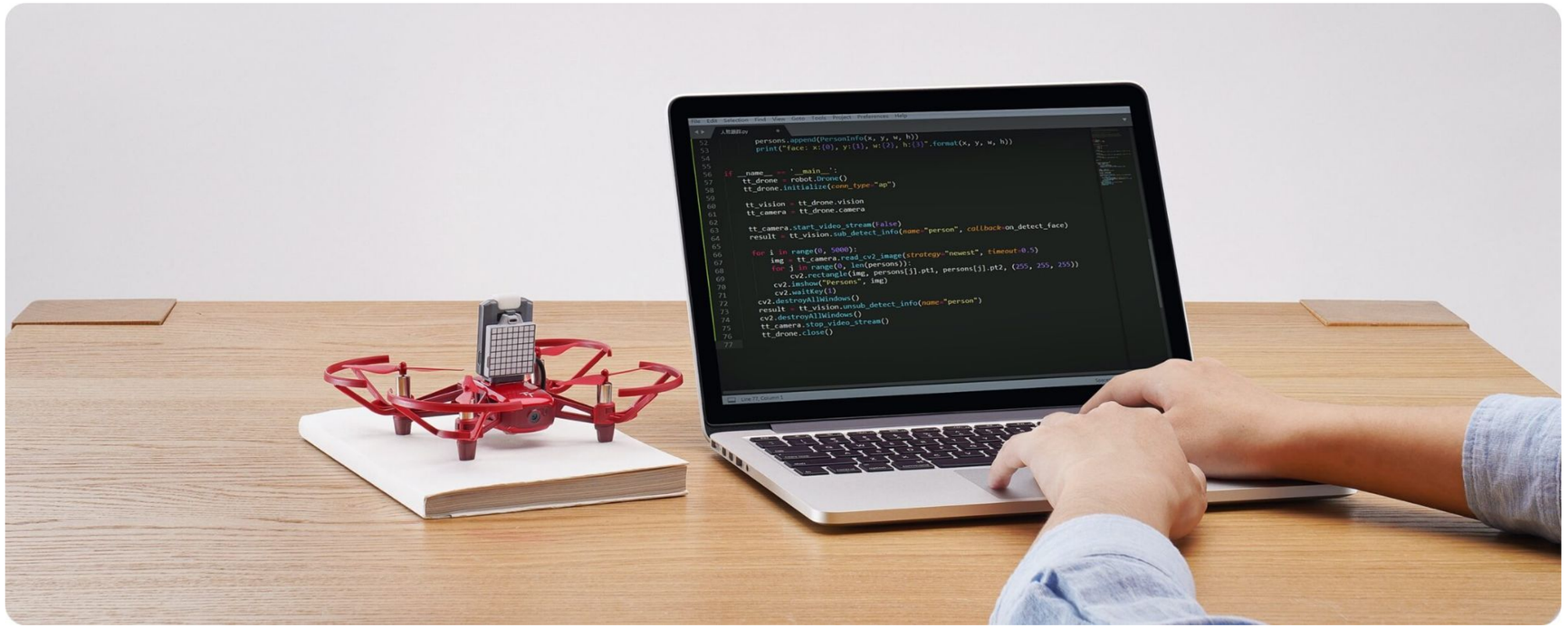
- Contexto
- Problemática
- Etapas de solución
- Soluciones existentes
- Historia
- Posibles soluciones
- Materiales
- Avances

Contexto

- RPAS

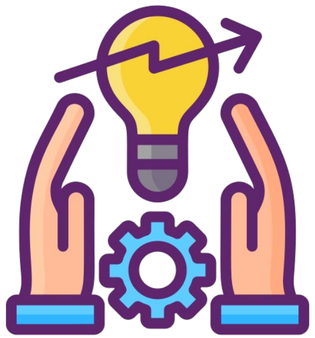
Contexto

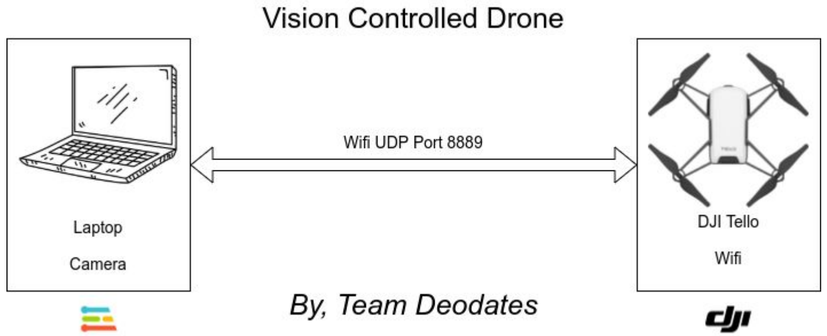
- RPAS
- UAV



Problemática

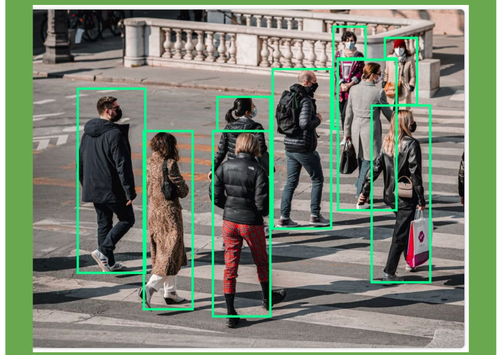
- Aplicación enfocada al ámbito militar.
- Alto costo de procesamiento.
  - Peso
  - Monetario
- Distancia de GS (Ground Station)
- Dependencia de piloto capacitado y altamente calificado para cumplir una misión

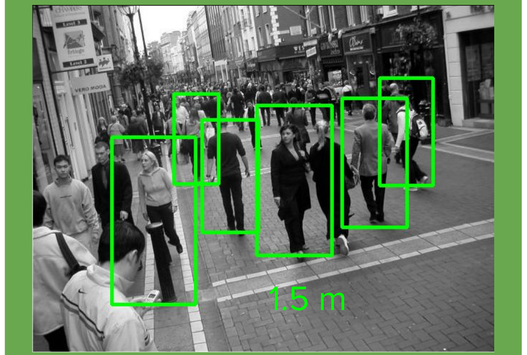




Etapas de Solución

- Autonomía en la toma de decisiones para cumplir una misión
  - Detección de personas a través de técnicas de Inteligencia Artificial
  - Identificación de la distancia del objetivo más cercano a través de un dispositivo de medición
  - Implementación en un dispositivo electrónico (Rasberry PI + Navio 2) para el procesamiento de imagen y toma de decisiones de manera remota.
  - Montaje del dispositivo en un chasis de Drone adecuado a las características de carga.





Soluciones existentes

- Servicio de desarrollo por parte de empresas. (Sense UAV)[1]

Soluciones existentes

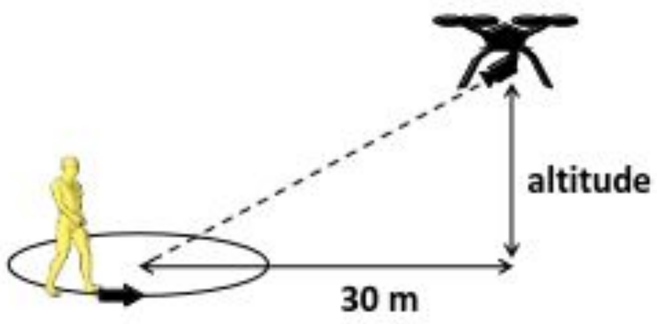
- Servicio de desarrollo por parte de empresas. (Sense UAV)[1]
- Para operaciones de rescate Liu C. y Szirányi T. (2021)[2]

Soluciones existentes

- Servicio de desarrollo por parte de empresas. (Sense UAV)[1]
- Para operaciones de rescate Liu C. y Szirányi T. (2021)[2]
- Sambolek S. e Ivasic-Kos M, la detección como fin mismo de el Drone [3]

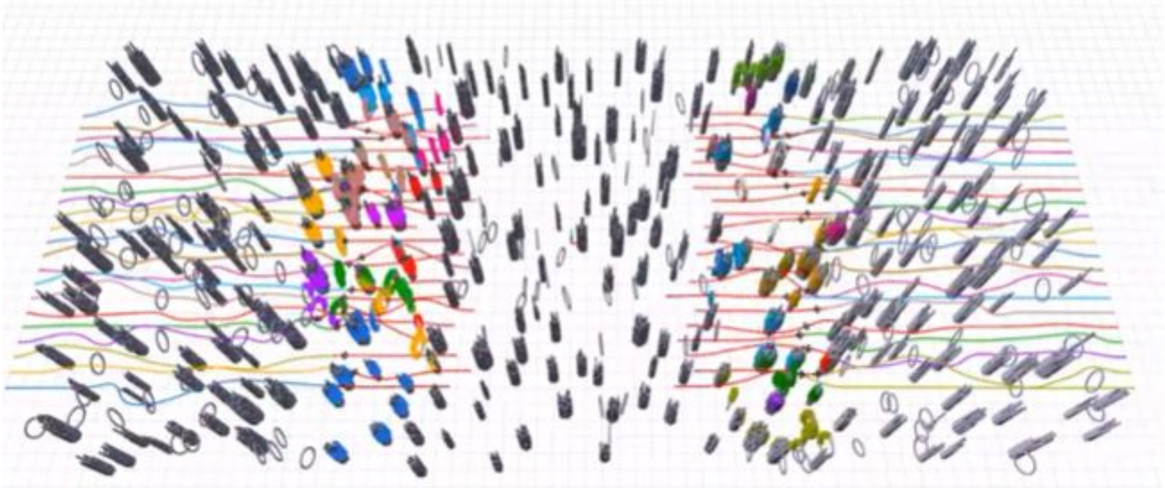
Soluciones existentes

- Servicio de desarrollo por parte de empresas. (Sense UAV)[1]
- Para operaciones de rescate Liu C. y Szirányi T. (2021)[2]
- Sambolek S. e Ivasic-Kos M, la detección como fin mismo del Drone [3]
- Asanka G. y otros (2018) si interviene la posibilidad de extraer información[4]



Soluciones existentes

- Servicio de desarrollo por parte de empresas. (Sense UAV)[1]
- Para operaciones de rescate Liu C. y Szirányi T. (2021)[2]
- Sambolek S. e Ivasic-Kos M, la detección como fin mismo del Drone [3]
- Asanka G. y otros (2018) si interviene la posibilidad de extraer información[4]
- Xin Zhou et al. (2022) vuelo en enjambre que implementa evasión de obstáculos[10]



Propuesta de Trabajo

1. Identificación de dispositivos de video que pueden integrarse a un dron como sistema de evasión de obstáculos.

2. Consulta de información, programas y librerías necesarias para ejecutar funciones de detección y evasión de obstáculos.

3. Selección y adquisición de cámara.

4. Identificación de interfaces, conexiones y comunicación entre computadora y cámara (calibración).

5. Pruebas de código (adquisición de imágenes o datasets, preprocesamiento de imágenes y obtención de características para su clasificación).

6. Uso de características para generar decisiones en el algoritmo del sistema de evasión de obstáculos.

Tipos de cámara

- RGB
- Térmicas
- Multiespectrales
- Hiperespectrales
- LIDAR

Selección de cámara

- Cámaras RGB-D

Selección de cámara

- Cámaras Estéreo

Selección de cámara

- Cámaras LIDAR

Propuestas de solución

En general,

- Se planea trabajar con imágenes tomadas del video en tiempo real cada 2 segundos.
- Con la posibilidad o la necesidad de mas o menos tonos de gris o la utilización de los canales de color.
- Posterior a este procedimiento aplicar filtros derivativos para el pre-proceso de la fotografía.

Propuestas de solución

- Específicamente:

  - Hacer un proceso de esqueletización binaria para alcanzar luego hacer un trabajo de identificación con redes neuronales y luego obtener las coordenadas de las posiciones de individuos en la imagen para solicitar la distancia a la cámara estéreo y mediante una serie de condicionales obtener la posición de la persona más cercana.

  - Utilizar YOLO para la detección de las personas y hacer un entrenamiento con uno o más Datasets de Kaggle que contengan figuras de humanos en diferentes entornos y con diferentes poses y posteriormente realizar de forma idéntica a la alternativa anterior la detección de el humano más próximo[5].

Propuestas de solución

- Específicamente:

  - Una posibilidad adicional es un método de machine learning basado en una función en cascada[6] propuesta por Paul Viola y Michael Jones en el que se entrena mediante una serie de imágenes en positivo y negativo. Para proceder a extraer coordenadas y extraer la distancia con el sujeto más cercano.

  - Otra posible solución es una vez pre-procesadas las imágenes hacer uso del framework MediaPipe[7] un toolkit construido en machine learning. Una vez detectadas estas regiones se procede a realizar la identificación de la distancia de cada una de las personas identificadas en video y la determinación de la distancia más corta a un individuo detectado

Propuestas de solución

- Específicamente:

  - Utilizar segmentación semántica para la identificación de regiones. Estas regiones se pueden clasificar según se convenga y tienen la ventaja de que pueden detectar objetos o regiones de objetos desconocidos, es decir, aquellos que la red neuronal no haya sido entrenada para identificar y tomarlas en consideración.

  - Utilización de un red neuronal para la detección de objetos combinada con un mapeo de profundidad de la imagen. Se podrían identificar regiones de interés previamente definidas para que sean resaltadas y a las cuales se les va a realizar una estimación de la profundidad a la cual se encuentran con respecto al dispositivo de adquisición de imágenes con el fin de acotar la cantidad de elementos a ser tenidos en cuenta dentro de la navegación.

Materiales

- Computador
  - Procesador Intel 11th Gen. I5 11300H
  - 15.7 Gb RAM
  - Sistema operativo 64 bits Windows 11

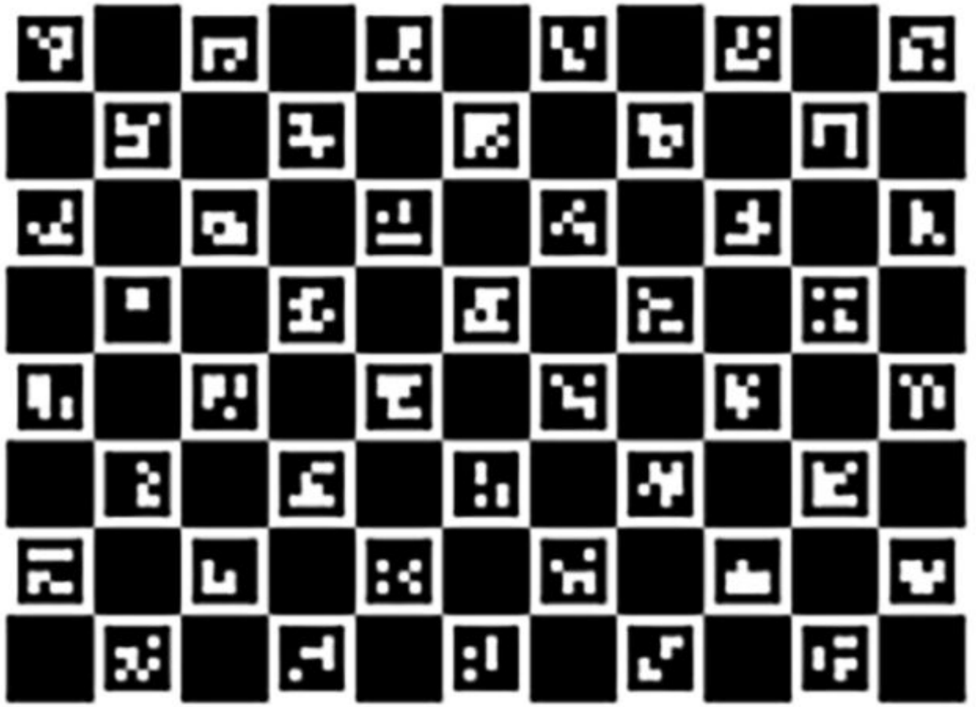
Materiales

- Camara Stereo Luxonis OAK-D-IoT-40 [8]

  - 12MP, 4K / 60Hz
  - 2 cámaras monocromáticas 720P / 120Hz
  - Procesador de visión Myriad X MA2485
  - Interfase cámara 2x 2-lane MIPI
  - Interfase cámara 1x 4-lane MIPI
  - Interfase Quad SPI con dos 2 selectores dedicados
  - Interfase I^2C
  - Interfase UART
  - Interfase USB2 y USB3
  - GPIO (1.8 V y 3.3 V)
  - Almacenamiento: Flash NOR integrado opcional y / o EEPROM
  - Capacidad de procesamiento: 4 TOPS (1.4 TOPS para AI)
  - Potencia: entrada de alimentación única de 5 V o 3,3 V
  - Dimensiones: 40 x 30 x 6 mm

Avances

- Calibración



Avances

- Configuración de la cámara

```python
import depthai as dai
import cv2 as cv
import numpy as np


class Camera():

    def __init__(self, usb2mode = True):
        self.usb2mode = usb2mode

    def getFrame(self, queue):
        frame = queue.get()
        return frame.getCvFrame()

    def getMonoCamera(self, pipeline, isLeft):
        mono = pipeline.createMonoCamera()
        mono.setResolution(dai.MonoCameraProperties.SensorResolution.THE_800_P)
        if isLeft:
            mono.setBoardSocket(dai.CameraBoardSocket.LEFT)
        else:
            mono.setBoardSocket(dai.CameraBoardSocket.RIGHT)
        return mono
```

Avances

- Configuración de la cámara

```
    def createStereoCamera(self):
        pipeline = dai.Pipeline()
        monoR = self.getMonoCamera(pipeline, isLeft = False)
        monoL = self.getMonoCamera(pipeline, isLeft = True)

        #Set output Xlink for left Camera
        xoutL = pipeline.createXLinkOut()
        xoutL.setStreamName("left")

        #Set output Xlink for right Camera
        xoutR = pipeline.createXLinkOut()
        xoutR.setStreamName("right")

        # Attach cameras to output Xlink
        monoL.out.link(xoutL.input)
        monoR.out.link(xoutR.input)

        return dai.Device(pipeline, usb2Mode=self.usb2mode)
```

Avances

- Obtención de imágenes

```
while True:

    leftFrame = camera.getFrame(leftQueue)
    rightFrame = camera.getFrame(rightQueue)

    if sideBySide:
        imOut = np.hstack((leftFrame, rightFrame))
    else :
        imOut = np.uint8(leftFrame/2 + rightFrame/2)

    cv.imshow("Stereo Pair", imOut)
    cv.imwrite(f"./images/{images}.jpg", imOut)
    images+=1

    key = cv.waitKey(1)
    if key == ord('q'):
        break
    elif key == ord('t'):
        sideBySide = not sideBySide
```

Avances

- Obtención de imágenes

Avances

- Obtención de imágenes

Avances

- Configuración de la cámara

```
    def createColorCamera(self, previewSize = (300,300)):
        pipeline = dai.Pipeline()
        cam_rgb = pipeline.create(dai.node.ColorCamera)
        cam_rgb.setPreviewSize(previewSize)
        cam_rgb.setInterleaved(False)

        xout_rgb = pipeline.create(dai.node.XLinkOut)
        xout_rgb.setStreamName("rgb")
        cam_rgb.preview.link(xout_rgb.input)

        return dai.Device(pipeline, usb2Mode=self.usb2mode)
```

Avances

- Obtención de imágenes

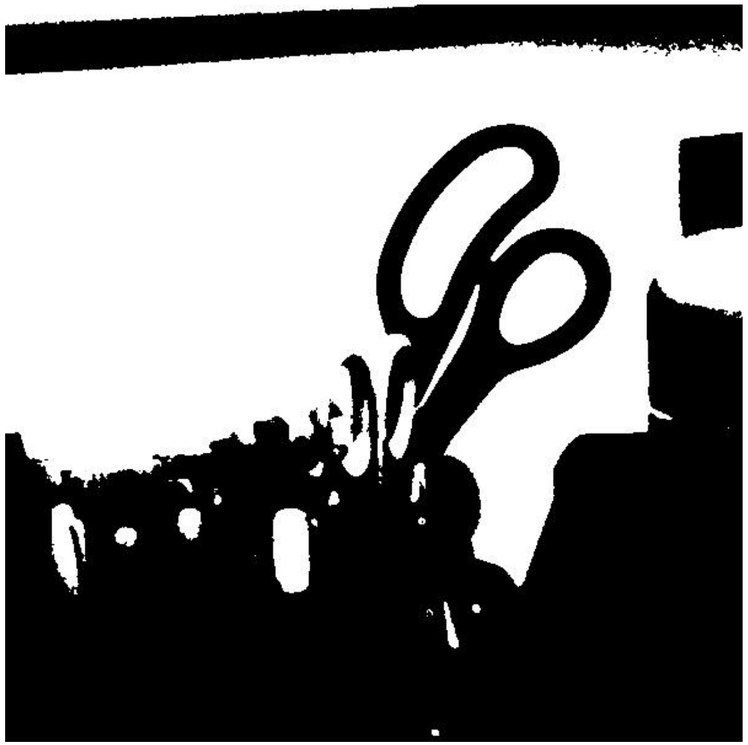
```
while True:
    imOut = None
    in_rgb = q_rgb.tryGet()

    if in_rgb is not None:
        imOut = camera.getFrame(q_rgb)

    if imOut is not None:
        cv.imshow("Color Camera", imOut)
        cv.imwrite(f"./images/{images}.jpg", imOut)
        images+=1

    key = cv.waitKey(1)
    if key == ord('q'):
        break
```

Avances

- Obtención de imágenes

Avances

- Filtros



Referencias

1. SENSE UAV. url: https://www.gradiant.org/en/portfolio/sense-uav/?gclid=Cj0KCQjw27mhBhC9ARIsAIFsETGc3fQCXLOUJfnTyq53zK4KdJxBM1-CrqCpyqq960yOsQw1ekGQmxkaAtL9EALw_wcB.
2. Real-Time Human Detection and Gesture Recognition for On-Board UAV Rescue. url: https://www.mdpi.com/1424-8220/21/6/2180.
3. Person Detection in Drone Imagery. url: https://www.researchgate.net/publication/346687347_Person_Detection_in_Drone_Imagery.
4. Human Detection and Motion Analysis from a Quadrotor UAV. url: https://iopscience.iop.org/article/10.1088/1757-899X/405/1/012003/pdf.
5. People-Detector. url: https://www.google.com/url?q=https://github.com/humandecoded/People-Detector&sa=D&source=docs&ust=1681306036074855&usg=AOvVaw1Ms52GIngVBTWEHIxXzZCT.
6. Pedestrian Detection using OpenCV Python | Human Detection | Python | OpenCV. url: https://www.youtube.com/watch?v=jTwNerGmc1s&ab_channel=FalconInfomatic
7. Introduction to MediaPipe. url: https://learnopencv.com/introduction-to-mediapipe/.
8. Luxonis OAK-D-IoT-40 cámara estero visión artificial. url: https://dynamoelectronics.com/tienda/luxonis-oak-d-iot-40/
9. Introduction to OAK-D and DepthAI. url: https://learnopencv.com/introduction-to-opencv-ai-kit-and-depthai/.