Índice general

v

Índice general

Capítulo 2

Introducción específica

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

2.1. Definición de Intruso

2.2. Modelo Utilizado

Si en el texto se hace alusión a diferentes partes del trabajo referirse a ellas como capítulo, sección o subsección según corresponda. Por ejemplo: "En el capítulo 1 se explica tal cosa", o "En la sección **??** se presenta lo que sea", o "En la subsección **??** se discute otra cosa".

Cuando se quiere poner una lista tabulada, se hace así:

- Este es el primer elemento de la lista.
- Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

Si se desea poner una lista numerada el formato es este:

1. Este es el primer elemento de la lista.
2. Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

2.3. Protocolos de vídeo

La recepción y análisis del vídeo depende únicamente de los formatos soportados por OpenCv. Dentro de los que, según la documentación encontramos están:

Sin embargo, es importante tener en cuenta que de todos esos se ha probado únicamente los siguientes:

2.4. Software utilizado

Para el desarrollo de este trabajo, y teniendo en cuenta el alcance propuesto desde el inicio, fue necesario el uso de tres tipos de software:

5

Capítulo 2

Introducción específica

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

2.1. Definición de Intruso

Dado que el trabajo tiene un objetivo específico en mente, la definición de intruso dada por el cliente incluye únicamente personas o animales de gran porte. El dataset de COCO permite el reconocimiento de una serie de animales de gran porte que no hacen parte de la fauna argentina, por lo que se decidió en conjunto con el cliente que algunas etiquetas se filtrarían a fin de evitar que el modelo trate de reconocer estas clases. Algunos ejemplos de etiquetas filtradas incluyen:

- Elefantes
- Jirafas
- Cebras

Por otro lado, y a fin de mejorar el reconocimiento de intrusos, se decidió incorporar algunas clases que corresponden a objetos que suelen cargar las personas y vehículos. De esta manera se hace posible una suerte de reconocimiento indirecto, dándole al modelo una "segunda oportunidad" en casos en los que por algún motivo generó un falso negativo en el reconocimiento de una persona. Es importante tener en cuenta que estos objetos propuestos ya hacían parte del dataset COCO, por lo que no se requirió reentrenar el modelo para lograr su detección. Dentro de estos objetos relacionados encontramos:

- Vehículos
- Objetos de porte personal, como maletines o sombrillas

Finalmente, es importante tener en cuenta que cualquier detección de las clases indicadas dentro del streaming de vídeo será marcada como intruso. Es decir, no será necesario considerar que la presencia de cualquiera de estas clases en un lote vecino no deberá ser marcada como intruso. Dicho esto, las clases que generarán un reporte positivo para intruso al ser detectadas son:

Hay que recordar que la sola detección de una de estas clases generará un reporte positivo de intruso. El modelo no está en capacidad de identificar personas específicas en el vídeo, por lo que le resultará imposible diferenciar entre personal

1. Paquetes de Python: Son paquetes importados en el código a través del comando import. Cabe mencionar que todos los paquetes son de código abierto. Dentro de ellos encontramos:

* Numpy * Matplotlib * OpenCV * Tensorflow

2. Código en Python: En especial para la implementación de la metodología YOLOV3. En internet está disponible el código fuente en formato Python (py).

3. Software completo: En particular para la retransmisión del vídeo procesado, se utilizó OBS estudio. Esta decisión responde a que, dentro del alcance del proyecto, no se definió ninguna actividad que no se encuentre directamente relacionada con la implementación de la detección de intrusos. Sin embargo, dado que se trata de un proyecto en el que se debe devolver la imagen procesada, es importante asegurarse que la retransmisión de los datos funcione de la manera correcta.

6 Capítulo 2. Introducción específica

Clases detectadas como Intruso	
Persona	Vaca
Camión	Bus
Automóvil	Sombrilla
Motocicleta	Mochila
Bicicleta	Maletín
Perro	Maleta
Caballo	Gato

autorizado en un área (el mismo personal de seguridad, o la persona al mando del drone), y a intrusos en la zona.

2.2. Modelo Utilizado

El modelo utilizado para el desarrollo de la totalidad del trabajo es YoloV3. La principal ventaja que tiene este modelo es que es capaz de ejecutarse con una velocidad considerablemente mayor a la de modelos anteriores como RetinaNet-50 y RetinaNet-101. Este modelo funciona a través del uso de redes neuronales convolucionales y calcula tres parámetros:

- Intersection over Union (IoU), que puede ser interpretado como la precisión del modelo al rodear un objeto con una caja. Es decir, qué tan acertada es la posición de la caja propuesta con el modelo, en relación con la posición predicha del objeto anteriormente.
- Offset de la caja detectada en comparación con la caja ancla.
- Tensor con la probabilidad de ocurrencia de las 80 clases que forman parte del dataset COCO.

Es importante tener en cuenta que estos no son los parámetros finales arrojados por el modelo, sino que más adelante, la red calcula las coordenadas de la caja con la que se rodeará al objeto. Es decir, para cada objeto detectado, el modelo va a devolver (en ese orden):

- pc: Probabilidad de que haya un objeto.
- bx: Coordenada x del centro de la caja propuesta
- by: Coordenada y del centro de la caja propuesta
- bh: Altura de la caja propuesta
- bw: Ancho de la caja propuesta
- c: Tensor con la probabilidad de ocurrencia de las 80 clases que forman parte del dataset COCO.

Esto se logra a través del uso de Anchor Boxes: cajas de predicción predefinidas que el modelo va refinando a medida que analiza la imagen y calcula las métricas de cada uno de los objetos encontrados. A través de este método es posible eliminar la necesidad de una "ventana deslizante", lo que a su vez permite analizar la totalidad de la imagen en un tiempo mucho menor que los métodos basados en aquel método, y, en consecuencia, se hace posible llevar a cabo un análisis en tiempo real.

Capítulo 3

Diseño e implementación

3.1. Consideraciones generales

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
}
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.

Dadas las bondades explicadas anteriormente, en conjunto con su facilidad de implementación en Python, se escogió partir del modelo YoloV3 y modificar su código fuente para lograr generar y extraer la información pertinente para completar el propósito del proyecto. Cabe aclarar que, la decisión de utilizar este modelo se vio reforzada una vez se ejecutó la primera versión en tanto que el modelo está optimizado para funcionar con tarjeta gráfica (Nvidia Titan X), el código se pudo ejecutar satisfactoriamente, sin dejar a un lado los requisitos del proyecto en CPU (AMD Ryzen 5).

2.3. Protocolos de vídeo y transmisión

El módulo del proyecto encargado de extraer y permitir el cálculo sobre cada uno de los frames es OpenCV, por lo que los protocolos de vídeo aceptados corresponderán a los aceptados por OpenCV. En este sentido, el código está en capacidad de detectar intrusos en:

- Vídeo guardado localmente
- Secuencia de imágenes
- URL de streaming de vídeo
- Pipeline Gstreamer
- Datos obtenidos desde Webcam

Sin embargo, es importante mencionar que, durante el desarrollo, se hizo pruebas únicamente con los métodos de vídeo guardado localmente y URL de streaming de vídeo. Los otros métodos no han sido probados, y, por lo tanto, no se podrá garantizar su correcto funcionamiento al solicitar al módulo ejecutarse utilizándolos como origen de datos.

Por otro lado, y recordando que se trata de la librería que hace posible extraer la información del vídeo para hacer posible su análisis, es de vital importancia tener en mente cuáles son los formatos de vídeo admitidos por OpenCV. Dentro de ellos se encuentran:

- H264
- MPEG4
- AV1

Ahora bien, para acceder a ellos, bastará con pasar el link apropiado al método . Dentro de los protocolos probados durante el desarrollo del proyecto encontraremos:

- MQTT: Es un protocolo que permite la comunicación entre dispositivos desarrollado por IBM con gran énfasis en IoT. Es un protocolo muy seguro teniendo en cuenta con un componente que filtra las comunicaciones entre el servidor y el cliente, por lo que ninguno de los componentes conocerá los datos del otro componente. En cambio, el cliente solicita una suscripción a un tema que le interesa, y el agente se encarga de distribuir los mensajes enviados por el servidor. Cabe mencionar que MQTT está pensado especialmente para dispositivos IoT con recursos limitados y poco ancho de banda.

			Software Requerido
Paquete	Tipo	Licencia	Función
PIP	Gestor de Librerías	MIT	Gestor de librerías
Numpy	Librería de Python	BSD	Cálculos Matricial
OpenCV	Librería de Python	Apache 2	Manipulación de i
Matplotlib	Librería de Python	BSD	Creación de visua
Tensorflow	Librería de Python	Apache 2	Framework de Ma
Pycharm	IDE	Apache 2	Ambiente de desa
OBS Studio	Software completo	GPLv2	Streaming de víde
Python	Lenguaje de programación	Python Software Foundation License V2	Lenguaje en el qu
Ubuntu	Sistema operativo	Creative Commons CC-BY-SA v 3.0 UK	Sistema operativo

- • RTMP: Es un protocolo diseñado específicamente para la transmisión de audio y vídeo con gran rendimiento. Esto lo logra dividiendo la información en pequeños fragmentos, cuyo tamaño puede ser negociado de manera dinámica entre el cliente y el servidor de forma que la transmisión de vídeo se adapta mejor a las condiciones de la red de la que depende la transmisión. Sin embargo, dado que se trata de un protocolo diseñado para la transmisión de vídeo y audio con el menor delay posible, este protocolo puede no ser el más adecuado cuando las condiciones de la red son limitadas.

2.4. Software y Hardware utilizados

Para su funcionamiento, este trabajo requiere la presencia de algunos paquetes de Software instalados en la máquina, los cuales son llamados directamente por el usuario cuando requiera la funcionalidad (Por ejemplo, OBS Studio), o bien, que ya vienen incorporados en el código y por lo tanto, son un requisito indispensable para el proyecto y deberán encontrarse instalados. Dicho esto, podemos clasificar el software utilizado en tres categorías:

1. Paquetes de Python: Son paquetes importados en el código a través del comando import. Cabe mencionar que todos los paquetes son de código abierto. Ejemplos de estos paquetes son:

- Numpy
- Matplotlib
- OpenCV

2. Código en Python: En especial para la implementación de la metodología YOLOV3. En internet está disponible el código fuente en formato Python (py).

3. Software completo: En particular para la retransmisión del vídeo procesado, se utilizó OBS estudio. Esta decisión responde a que, dentro del alcance del proyecto, no se definió ninguna actividad que no se encuentre directamente relacionada con la implementación de la detección de intrusos. Sin embargo, dado que se trata de un proyecto en el que se debe devolver la imagen procesada, es importante asegurarse que la retransmisión de los datos funcione de la manera correcta.

Más específicamente, las diferentes librerías y paquetes de software utilizados para el desarrollo de este trabajo, y sus versiones correspondientes son:

Capítulo 4

Ensayos y resultados

4.1. Descripción del proceso de pruebas

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

4.2. Pruebas en Raspberry Pi

4.3. Caso de Uso

Es importante tener en cuenta que la totalidad de programas y librerías utilizadas, tanto para el desarrollo, como para la ejecución del proyecto son Software libre, por lo que no se requerirá adquirir ningún tipo de licencia para ponerlo en funcionamiento.

De igual manera, cabe resaltar que OBS Studio es el programa de retransmisión elegido teniendo en cuenta que esta es una característica que se encuentra fuera del alcance del trabajo y por lo tanto, cuya implementación es necesaria a través de este método mientras que hacerlo a través de software integrado en el código fuente (como FFMpeg) será parte del alcance de trabajos posteriores.

El paquete de Python ya viene instalado por defecto en la instalación de Ubuntu, por lo que no se requerirá su instalación. El resto de los paquetes, sin embargo, deberán ser instalados de acuerdo con el manual de instalación que se entrega al cliente, en el que se detalla de manera minuciosa cuál es el procedimiento que se debe seguir a fin de instalar correctamente todos los paquetes.

De igual manera, se debe tener en cuenta que dadas las restricciones de la máquina en la que se desarrolló el proyecto, la totalidad del desarrollo fue llevado a cabo en máquinas virtuales a través del uso de programas de virtualización como VMWare Workstation. Dentro de los requerimientos mínimos de hardware del proyecto, se tienen los siguientes:

- Procesador: AMD Ryzen 5
- Tarjeta Gráfica: No requerida
- Memoria RAM: 6 GB.

Es importante tener en cuenta que los listados anteriormente son los requisitos mínimos del sistema, que van a garantizar que el proyecto se ejecute cumpliendo con los requerimientos del cliente. Sin embargo, se recomiendan las siguientes características del sistema para un rendimiento óptimo a la hora de ejecutar el módulo.

- Procesador: Intel Core i7
- Tarjeta Gráfica: NVidia Tesla T4 o NVidia Titan X.
- Memoria RAM: 8 GB.

El modelo YoloV3, sobre el que está basado el desarrollo del proyecto utiliza de manera extensiva la API Cuda de Nvidia. Por este motivo, tratar de ejecutar el proyecto en tarjetas gráficas de AMD resultará en el modelo siendo llevado al procesador por Tensorflow. En consecuencia, se afirmará que el trabajo resulta incompatible con estas tarjetas gráficas. Esto fue comprobado con el uso de una tarjeta AMD Radeon Vega 8.

Capítulo 5

Conclusiones

5.1. Resultados Obtenidos

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Tiempos de ejecución

Acá se indica cómo se podría continuar el trabajo más adelante.

5.3. Monitoreo de resultados

Capítulo 3

Diseño e implementación

3.1. Consideraciones generales

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epigrafe descriptivo"]
las lineas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
}
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.