Índice general

v

Índice general

Capítulo 1

Introducción general

1.1. Introducción

LaTeX no es WYSIWYG (What You See is What You Get), a diferencia de los procesadores de texto como Microsoft Word o Pages de Apple o incluso LibreOffice en el mundo open-source. En lugar de ello, un documento escrito para LaTeX es en realidad un archivo de texto simple o llano que *no contiene formato* . Nosotros le decimos a LaTeX cómo deseamos que se aplique el formato en el documento final escribiendo comandos simples entre el texto, por ejemplo, si quiero usar texto en itálicas para dar énfasis, escribo `\it{texto}` y pongo el texto que quiero en itálicas entre medio de las llaves. Esto significa que LaTeX es un lenguaje del tipo «mark-up», muy parecido a HTML.

1.2. Motivación

Si sos nuevo en LaTeX, hay un muy buen libro electrónico - disponible gratuitamente en Internet como un archivo PDF - llamado, «A (not so short) Introduction to LaTeX». El título del libro es generalmente acortado a simplemente *lshort*. Puede descargar la versión más reciente en inglés (ya que se actualiza de vez en cuando) desde aquí: http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf

Se puede encontrar la versión en español en la lista en esta página: http://www.ctan.org/tex-archive/info/lshort/

1.3. Estado del arte

Acá tiene un ejemplo de una "subsubsección" que es el cuarto nivel de ordenamiento del texto, después de capítulo, sección y subsección. Como se puede ver, las subsubsecciones no van numeradas en el cuerpo del documento ni en el índice. El formato está definido por la plantilla y no debe ser modificado.

1.4. Objetivos y alcance

Si estás escribiendo un documento con mucho contenido matemático, entonces es posible que desees leer el documento de la AMS (American Mathematical Society) llamado, «A Short Math Guide for LaTeX». Se puede encontrar en línea en el siguiente link: http://www.ams.org/tex/amslatex.html en la sección «Additional Documentation» hacia la parte inferior de la página.

1

Capítulo 1

Introducción general

1.1. Introducción

El uso de cámaras es una práctica extremadamente común en el negocio de la seguridad. Su uso puede mejorar el área de cobertura de vigilancia, mientras que al mismo tiempo reduce la cantidad de personal empleado en hacer rondas de seguridad. Sin embargo, tener cámaras estáticas implica que será necesario contar con más de ellas, que a la vez tendrán que ser revisadas por varias personas.

La propuesta abordada en este trabajo busca solucionar estos dos problemas a través de la creación de un módulo de inteligencia artificial que permita el análisis automático de varios streamings de vídeo provenientes de cámaras montadas en drones, y sea capaz de responder a estos estímulos digitales, al pasarle la información del intruso encontrado al software del que hará parte.

Esto permitirá que haya un análisis mucho más preciso y a menor costo de lo que era posible anteriormente, teniendo en cuenta que los drones podrán abarcar un área mucho mayor, y que no se dependerá de la concentración de una persona para encontrar intrusos. Esto implica que el sistema deberá estar en capacidad de analizar más de un streaming de video al tiempo. Igualmente, el uso de estas tecnologías permite la utilización de cámaras infrarrojas, lo que hace posible la identificación de intrusos en la noche, cuando el ojo humano es incapaz de distinguir.

Dado que este sistema hará parte de un módulo más grande, se decidió que, dentro de los requisitos, el sistema no deberá tomar decisiones basadas en sus hallazgos, sino simplemente comunicarlos tanto a otro sistema (del que hace parte), como al personal de seguridad encargado.

1.2. Motivación

Este trabajo responde a cinco necesidades concretas del ámbito de la vigilancia. Estas necesidades son:

- Razones económicas: Teniendo en cuenta que, al reducir al personal encargado de hacer seguimiento a las cámaras, o de hacer rondas de seguridad en horarios determinados, es posible reducir costos. De igual manera, el uso de drones permite una reducción sustancial en el número de cámaras que deberán ser instaladas, cosa que resulta particularmente útil en grandes áreas y permitirá una reducción muy importante en los costos de mantenimiento asociados.

- Calidad de la vigilancia: Dado que el trabajo de gran parte del personal de vigilancia tiene la particularidad de que puede volverse muy monótono. Esto teniendo en cuenta que es un trabajo en el que se espera que no suceda nada. Sin embargo, es de vital importancia cuando esto deja de cumplirse, dado que hay algún evento que pueda afectar la seguridad.

  Al ser tan monótono, es muy probable que el personal encargado se vea expuesto a distracciones, que pueden tener consecuencias severas en el momento en el que se presente algún evento que comprometa la seguridad. La situación se puede agravar en caso de tener una gran cantidad de cámaras, dado que el estar constantemente obligados a supervisarlas puede generar niveles no razonables de carga en el personal.

- Velocidad de reacción: Una vez identificada una potencial amenaza, a través del sistema encargado de controlar los drones se podrá enviar ordenes a los equipos a fin de tomar mejores decisiones para manejar la situación. Es por esto que es de vital importancia que el módulo de inteligencia artificial encargado de la detección de los intrusos también sea capaz de hacerlo en tiempos razonables, que en este caso, han sido definidos por el cliente y se encuentran por debajo de los 4 segundos. Este valor se ha calculado a partir de la velocidad esperada de vuelo de los drones de hasta 30 km/h, lo que otorga un radio máximo de 33 metros entre detecciones. Cabe anotar, sin embargo, que el tiempo ideal es menor o igual a un segundo, lo que genera un radio de 8 metros aproximadamente.

- Aporte de pruebas: Los drones (a través de la necesidad de velocidad de reacción) son un método ideal para aportar pruebas confiables de los hechos. Es así como un modelo de inteligencia artificial permite generar una mayor claridad en la interpretación de las pruebas aportadas, lo que puede generar una mejor comprensión y análisis de los hechos por parte de todos los implicados.

- Identificación de intrusos en horas de la noche: Teniendo en cuenta que los modelos de inteligencia artficial pueden ser capaces de identificar las clases requeridas con diferentes condiciones de iluminación, puede ayudar a la identificación de intrusos cuando las condiciones de iluminación no son ideales. Por otro lado, es muy probable que esto sea mejorado sustancialmente con el uso de cámaras infrarrojas.

Este tipo de propuestas, sin embargo, no se han hecho posibles hasta hace relativamente poco, cuando los modelos de inteligencia artificial alcanzaron una madurez que permite que no requieran grandes capacidades computacionales para poder ser ejecutados, de forma que implementarlos no requiera una inversión importante en equipos de cómputo.

1.3. Estado del arte

Existen varias técnicas y varios modelos que se puede utilizar para implementar visión por computadora. Uno de los métodos más comunes a la fecha es el algoritmo YOLO, que se caracteriza por ser el primer modelo que no requiere hacer un doble análisis de la imagen (de aquí proviene su nombre: You only look once), y en consecuencia, analiza mucho más rápido.

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:

Modelo	Fecha de Lanzamiento
YoloV3	Abril de 2018
YoloV3 Lite	Abril de 2020
YoloV6	Junio de 2022

A pesar de que, a la fecha de redacción de este documento, las versiones más avanzadas de los modelos de la familia YOLO son las versiones V6 y V7, que ofrecen grandes mejoras en términos de precisión y rendimiento en comparación con sus antecesores. Sin embargo, como se puede ver en la tabla 1, estos modelos no estuvieron disponibles hasta una fecha posterior al inicio del trabajo correspondiente al proyecto.

Por otro lado, las versiones 4 y 5 fueron descartadas para implementación de este proyecto, visto que son dos versiones que fueron desarrolladas por personas diferentes a los desarrolladores originales, quienes se retiraron del proyecto por conflictos éticos con las capacidades de la técnica y sus potenciales usos. En consecuencia, gran parte de la industria se hace uso de la versión 3 del modelo.

Ahora bien, es importante tener en cuenta que existen diferentes versiones del modelo, cada una adaptada a algún caso de uso particular. Destacan dentro de estas versiones la arquitectura YOLOV3 y la arquitectura YOLOV3 Lite:

- YOLOV3: Es la versión "base" de la arquitectura YOLO. Representa un avance incremental con respecto a las arquitecturas YOLOV1 y YOLOV2. Su principal ventaja por encima de los modelos anteriores (en particular comparado con RetinaNet-50 y RetinaNet-101) es una reducción significativa en los tiempos de detección. El dataset sobre el que fue entrenado es el dataset COCO.
- YOLOV3 Lite: Es una versión reducida de la versión YOLOV3 diseñada específicamente para ser ejecutada en dispositivos con menor potencia, o que no cuentan con tarjeta gráfica. Si bien es un modelo que es capaz de reducir sustancialmente la complejidad del equipo requerido, también cuenta con una menor precisión en el momento de hacer la detección correspondiente.

Por otro lado, además de la arquitectura, es de vital importancia tener en cuenta el dataset que fue utilizado para el entrenamiento del modelo. En este caso, se trata del dataset coco, que cuenta con más de 200.000 imágenes etiquetadas (a la fecha de redacción de este documento), con más de 1.5 millones de objetos detectados en ellas.

1.4. Objetivos y alcance

A continuación, se presentan los requisitos definidos durante la etapa de definición de los objetivos del proyecto, así como el estado de cada uno de ellos a la fecha de finalización del trabajo. Es importante tener en cuenta que, dado que el proyecto sufrió cambios por temas operativos, de forma que algunos de los requisitos no pudieron ser cubiertos.

A pesar de que algunos de los requerimientos funcionales fueron descartados dado el estado del proyecto en el extremo del cliente, también hubo otros que fueron agregados:

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.

Tipo de Requerimiento	Número de Requerimiento	Requerimiento
Requerimientos Funcionales	1.1	El módulo debe detectar personas
	1.2	El módulo debe ser capaz de dete
	1.3	El módulo debe correr en un comp
	1.4	El sistema debe ser capaz de recor
	1.5	El sistema debe correr en un siste
Requerimientos de documentación	2.1	El sistema deberá contar con un m
	2.2	Se deberá contar con un manual d
	2.3	El código fuente deberá estar debi
Requerimientos de testing	3.1	El módulo deberá ser testeado for
	3.2	El cliente debe poder acceder en c
Requerimientos de la interfaz	4.1	El usuario debe poder iniciar fácil
Requerimientos de interoperabilidad	5.1 A	El módulo debe estar en capacidad
	5.1 B	El módulo debe ser capaz de recib
	5.2	Se requiere que el sistema pueda c

Tipo de Requerimiento	Número de Requerimiento	Fecha de Lanzamiento
Requerimientos Nuevos	6.1	El módulo deberá retransmitir los frames proce

Es importante tener en cuenta que el trabajo no contempló:

- Reconocimiento del intruso. Es decir, en caso de que se detecte a una persona, no se deberá dar con su identidad.
- Reacción del dron: es decir, una vez se haya detectado satisfactoriamente a un intruso, el sistema no deberá dar instrucciones específicas al dron, por ejemplo, de seguirlo.
- No existen requisitos específicos de cómo realizar la retransmisión de los frames ya procesados.

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.

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:

3.2. Arquitectura del proyecto

3.3. Esquema del módulo

3.4. Esquema de modificaciones

3.5. Ajustes para cumplir con el rendimiento

3.6. Entrega de resultados al resto del software

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.

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

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.

3.2. Arquitectura del proyecto

3.3. Esquema del módulo

3.4. Esquema de modificaciones

3.5. Ajustes para cumplir con el rendimiento

3.6. Entrega de resultados al resto del software

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 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