Proyecto Final

OpenCV

Victor Lucas

Índice

1. Estudi de viabilitat.

1.1. Establiment de l'abast del sistema.

Este proyecto servirá para la posibilidad de añadirlo dentro del SolarHub de la empresa Siarq y usarlo para contar cuántas personas han entrado y salido en una playa

1.2. Estudi de la situació actual.

Es posible que algunos gobierno restrinjan la entrada a la playa en este verano a más de x personas, así que eso crea la oportunidad de poder estar alerta de que no se sobrepase sin la necesidad de tener a un personal dedicado.

1.3. Definició dels requisits del sistema.

El proyecto necesitará una librería de Computer Vision, decidir la manera de contar la gente y un sistema de alerta para cuando pase de el límite permitido

1.4. Estudi de les alternatives de solució.

Hay otras opciones de Computer Vision aparte de OpenCV como por ejemplo TensorFlow, pero TensorFlow es un framework de machine learning, no una líbreria asi que la implementación sería más costosa y menos eficiente.

Para contar la gente tuve que decidir entre enfocar la cámara a toda la playa y contar los que hay o dirigir la cámara a la entrada de la playa y ver las personas que entrar y las que salen

El sistema de alerta podría ser un mail o un mensaje de telegram

1.5. Valoració de les alternatives.

TensorFlow podría ser una opción si batería o potencia de cpu no fuera un problema

1.6. Selecció de la solució.

Me acabe decidiendo por OpenCV por su gran portabilidad ya que se puede importar de manera sencilla a una gran cantidad de dispositivos, especialmente en comparación con sus alternativas. También tiene la ventaja de ser bastante más rápido y usar menos RAM que librerías o frameworks basados en deep learning.

Para contar decidí hacerlo observando la gente que entra y que sale en lugar de toda la playa a la vez porque me parecía la opción más viable, ya que algunas playas són demasiado grandes para que la cámara pueda detectar fácilmente a toda persona fuera y dentro del agua

En la cuestión de mensajería opté por Telegram por que me pareció más sensato enviar no tener que enviar un email entero cada vez que se pasara del límite de personas

2. Anàlisi del sistema.

2.1. Definició del sistema.

Cámara conectada a una raspberri pi con conexión a Internet y el código en python ejecutándose

El código tiene que comprobar la gente que haya entrado y enviar un mensaje a telegram usando mqtt cuando sobrepase

2.2. Establiment de requisits.

- Desarrollo de aplicación python usando opencv
- Conexión a telegram mediante mqtt
- Detectar personas a partir de un modelo caffe
- Código capture vídeo de la cámara(o de un vídeo guardado para hacer pruebas)

2.3. Definició d'interfícies d'usuari.

No hay necesidad de una interfaz de usuario

2.4. Especificació del pla de proves.

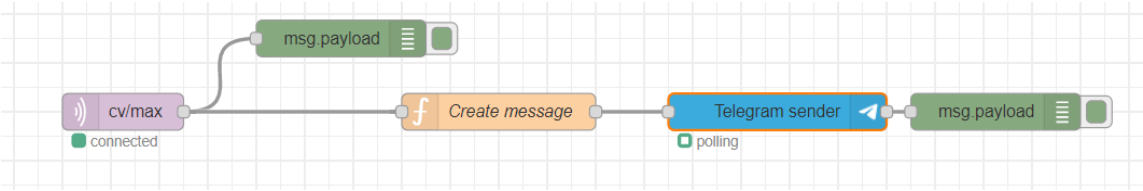
Comprobar hasta que distancia la cámara puede detectar a las personas entrantes, verificar el consumo de batería que supone tener la cámara y la raspberry pi conectadas, asegurarse de que el sol no ciegue la cámara después de añadirle una visera

3. Disseny del sistema

3.1. Arquitectura del sistema.

3.1.1. Definició de nivells d'arquitectura del sistema.

La raspberry hace el trabajo de procesamiento del vídeo menos la parte de enviar mensaje a telegram que esta hecho con nodered.



3.1.2. Especificació d'estàndards,normes de disseny i construcció.

En este apartado se definen el formato y plantilla de los documentos y la notación de los diagramas de diseño.

• Para generar la documentación se utilizará el formato PDF

• Para los diagramas de diseño utilizaremos la notación UML y se utilizará como software LucidChart

• El código fuente deberá estar correctamente comentado, indicando siempre por cada método utilizado la descripción, argumentos de entrada y datos de retorno.

3.1.3. Identificació de subsistemes.

Este proyecto no tiene subsistemas

3.2. Revisió de casos d'ús.

3.2.1. Revisió dels subsistemes segons els casos d'ús.

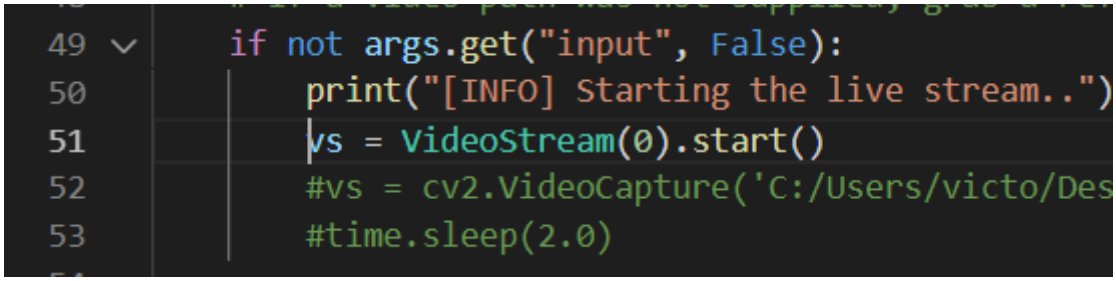
Este proyecto no tiene subsistemas

3.2.2. Elecció d'alternatives de components i llicències més adequades.

Este proyecto ya tiene las licencias correctas ya que son open source, la única manera de mejorarlo sería creando tu propia librería de Computer Vision en C para que fuera lo más optimizado posible

3.2.3. Especificacions de desenvolupament i proves.

Aparte de instalar todos los módulos de python previamente en el código las lineas 51 y 52 se tienen que descomentar y comentar dependiendo si vayas a usar un vídeo preparado o la videocámara



51 descomentada cuando sea la videocámara y viceversa

Y también hay que comentar la línea 95 si se usa la cámara y descomentarla si se usa un vídeo

```
94
95      #frame = frame[1] #if args.get("input", False) else frame
96
97      # if we are viewing a video and we did not grab a frame then we
```

Por último descomentar estas do líneas porque no funcionan con la raspberry en modo terminal

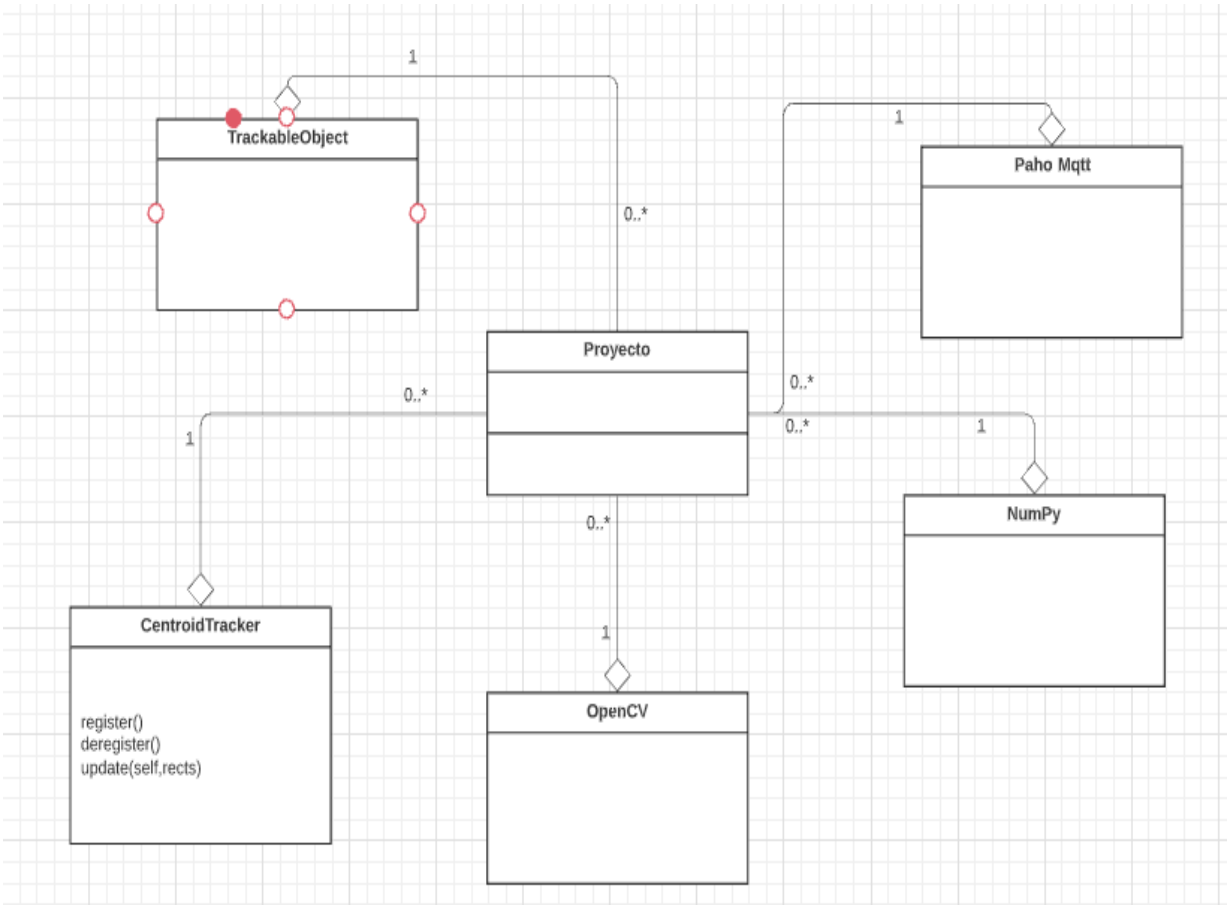
```
292         # show the output frame
293         cv2.imshow("Real-Time Monitoring/Analysis Window", frame)
294         key = cv2.waitKey(1) & 0xFF
```

3.2.4. Requisits d'implantació.

Componente o módulo	Versión
Python	3.9.4
dlib	19.22.0
imutils	0.5.4
numpy	1.20.2
opencv-python	4.5.1.48
paho-mqtt	1.5.1
pandas	1.2.4
python-dateutil	2.8.1
pytz	2021.1
schedule	1.1.0
scipy	1.6.3
six	1.15.0

3.3. Anàlisi paradigma estructurat / Orientat a Objectes

3.4. Disseny paradigma estructurat / Orientat a Objectes

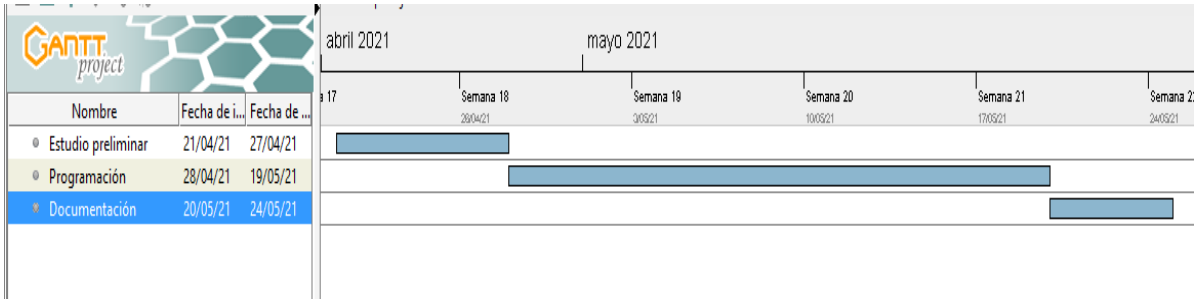


3.5. Persistència de dades: XML / Anàlisi i disseny de bases de dades / ORM

El proyecto no necesita guardar ningún dato

4. Desenvolupament.

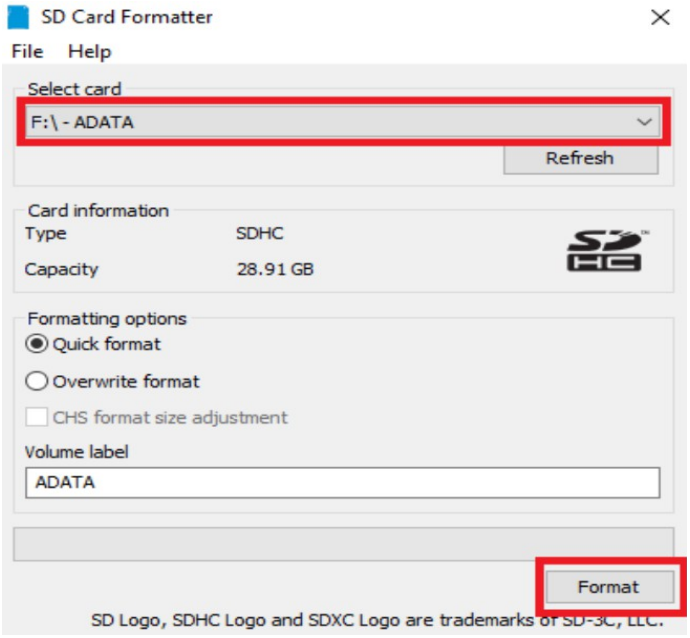
4.1. Planificació de les activitats de desenvolupament i integració de sistema.



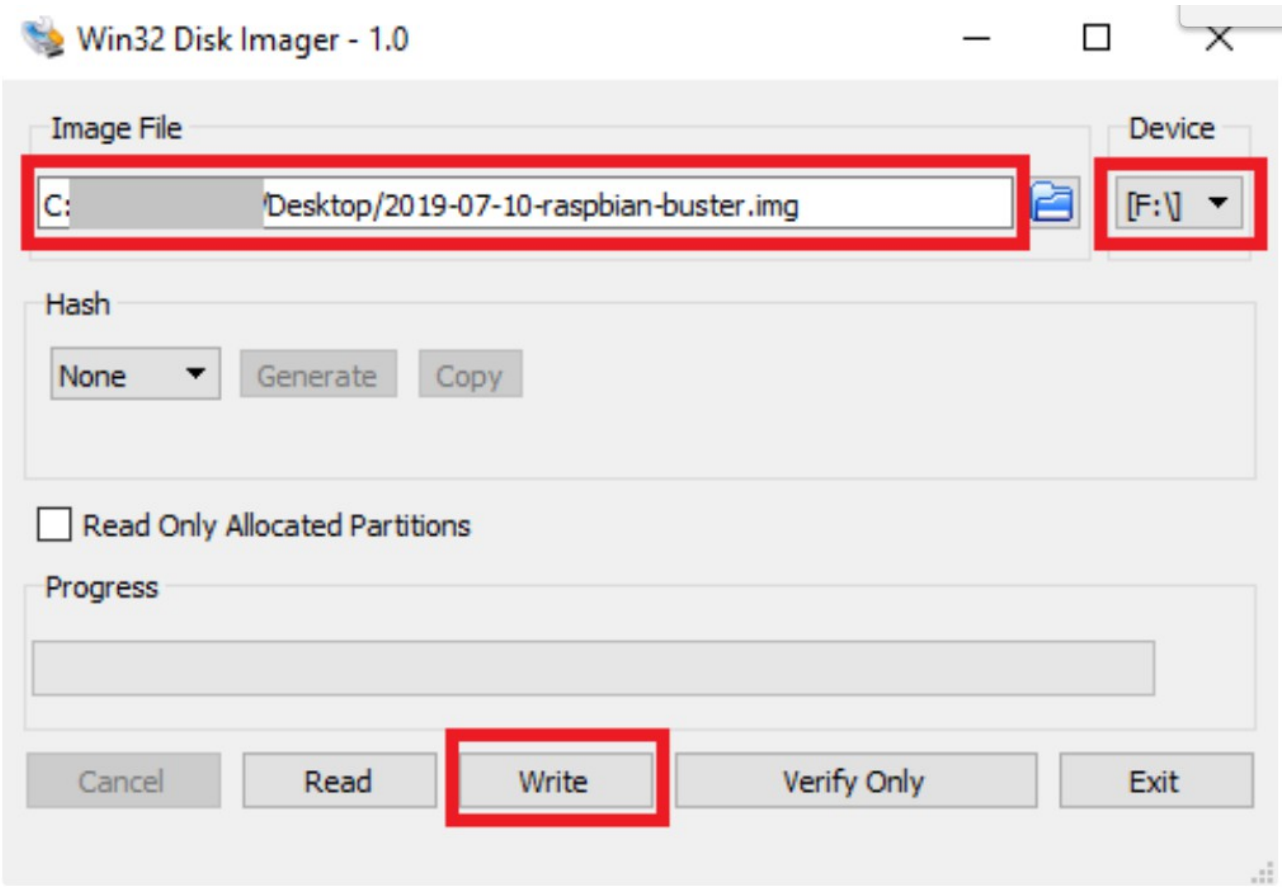
4.2. Desenvolupament.

- **Instalación Raspbian OS tarjeta SD**

  Primero hay que formatear la tarjeta usando SD card Formatter



Después se ha de montar raspbian os usando win32 disk imager



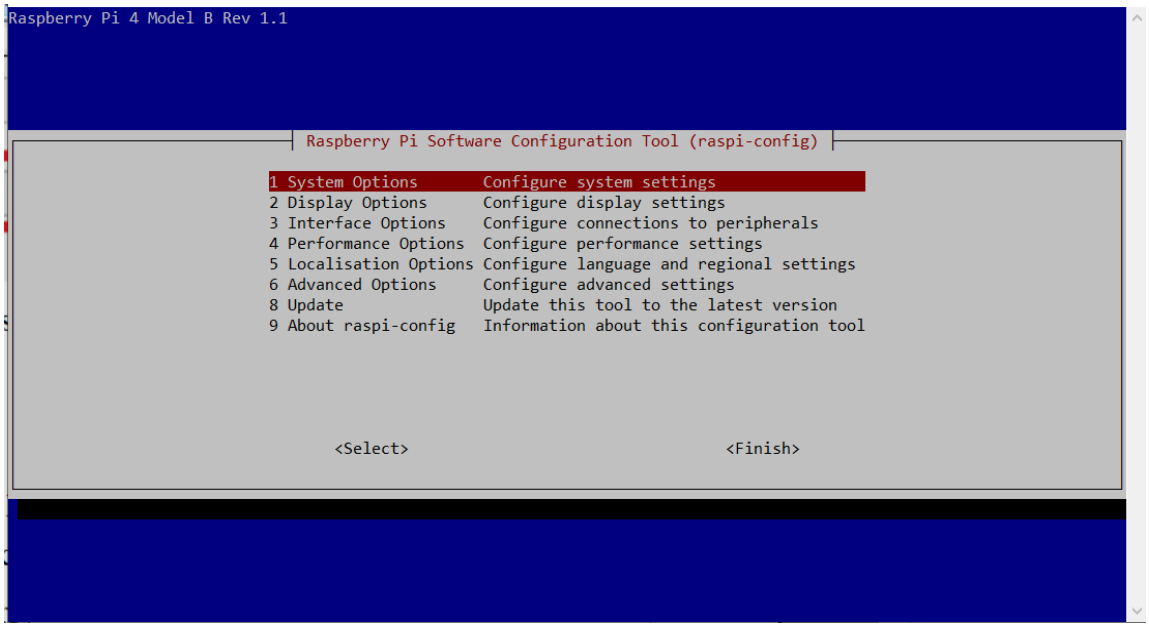
Dentro de la tarjeta se ha de crear un fichero llamado SSH sin extensión y vacío.

- **Conectar a WiFi**

  Primero conectas la placa y tu ordenador con un cable ethernet.

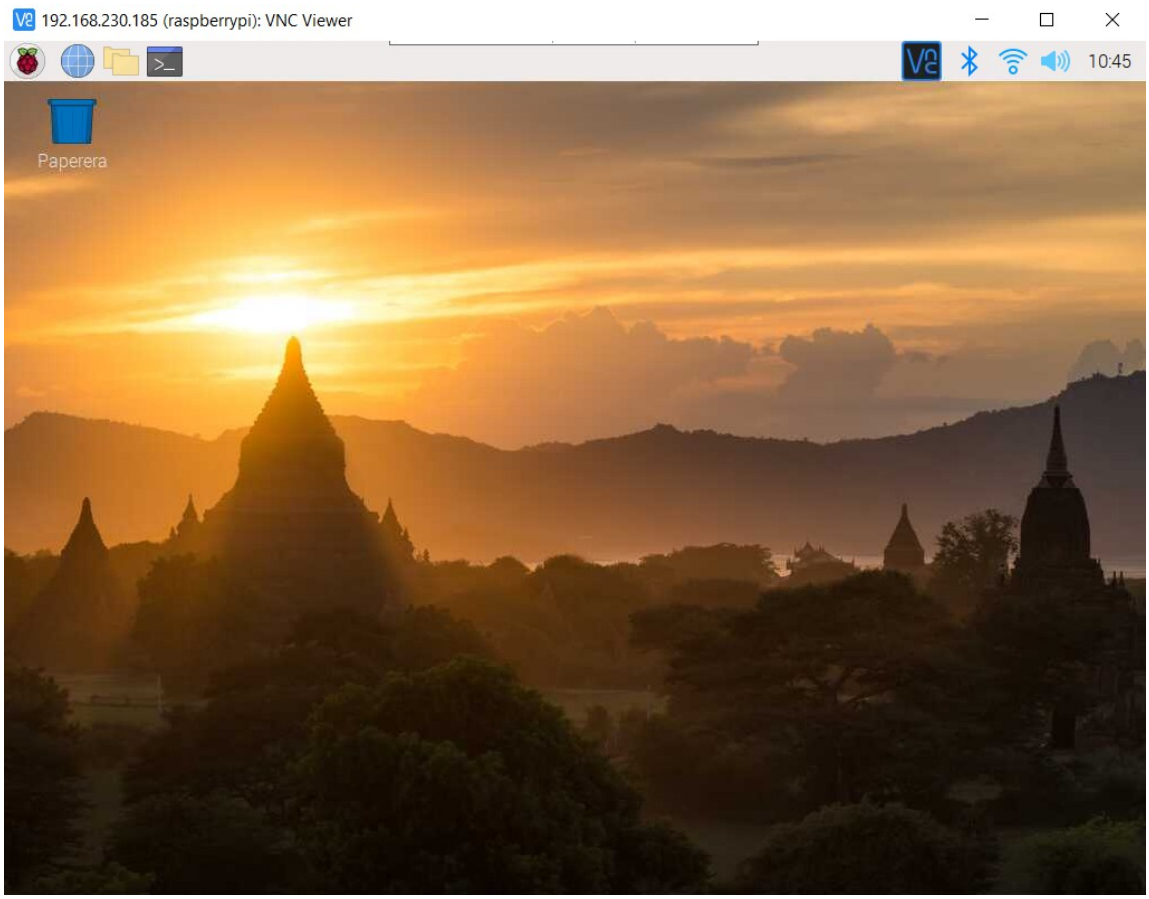
  Eso te permite conectarte a la placa con ssh pi@local.raspberrypi y contraseña raspberry

  Una vez dentro ejecutas sudo raspi-config te saldrá esto



Aquí entras en 1. System Options → S5 Boot/AutoLogin y seleccionas Desktop Autologin

Por último usando VNC Viewer con raspberrypi.local podrás entrar en la versión de escritorio de la raspberry y allí conectarte a wifi y pasar archivos de tu ordenador



- **Conexión con ip**

  Una vez conectado a wifi puedes ver tu ip con ip a y utilizar esa ip en lugar de raspberrypi.local y no hace falta el cable

- **Instalación python y módulos**

  Instalar el python y los módulos especificados en 3.2.4

  Prueba de que funciona

```
pi@raspberrypi:~/Documents/opencv $ python3 proyecto.py
[INFO] Starting the live stream..
```

4.3. Documentació tècnica del programari.

El código está comentado

5. Implantació.

5.1. Formació.

No hace falta ninguna formación especifica.

5.2. Implantació del sistema i proves.

Para implantarlo se tendrá que conectar una cámara a una raspberry pi dentro del SolarHub y comprobar que el código sigue funcionando correctamente.

Pruebas: Comprobar hasta que distancia la cámara puede detectar a las personas entrantes, verificar el consumo de batería que supone tener la cámara y la raspberry pi conectadas, asegurarse de que el sol no ciegue la cámara después de añadirle una visera

5.3. Nivell de serveis.

No usa servicios de pago, ni hay plan de usarlos.

6. Manteniment i versions futures.

Este código no necesitará mantenimiento a no ser que alguna librería deje de ser soportada.

Las mejoras para una versión futura serían: pasar el código a c++ para mejor eficiencia y usar un modelo de persona mejor entrenado