_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

SISTEMA DE VISUALIZACIÓN PARA EMERGENCIA EN INCENDIOS

Sergi Pérez Castaños

Tutor: José Manuel Mossi García

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2015-16

Valencia, 3 de julio de 2016

Resumen

En este documento se abordan los procedimientos y conocimientos necesarios para la creación de un sistema de visualización de imágenes térmicas para ser utilizado en casos de emergencia en incendios. Así pues, se pretende realizar una aplicación capaz de capturar imágenes térmicas y visualizarlas mediante una interfaz gráfica. El dispositivo empleado para la captura de estas imágenes es la cámara Seek Thermal Compact. Para la captura y visualización, se ha desarrollado un algoritmo que implementa estas funciones. A pesar de que el desarrollo del código fuente se ha realizado en Ubuntu, el dispositivo sobre el que se quiere ejecutar la aplicación es una Raspberry Pi. Este proyecto también se ofrece como una guía para preparar la Raspberry Pi para la ejecución del programa creado. Para poder desarrollar esta aplicación, se requiere instalar las bibliotecas OpenCV, libseekthermal, GTK+ 3 y gtkmm 3. La instalación de estos paquetes se encuentra detallada tanto para Ubuntu como para Raspbian en este documento.

Resum

Aquest document descriu els procediments i coneixements necessaris per a la creació d'un sistema de visualització d'imatges tèrmiques per a ser utilitzat en casos d'emergència en incendis. Així, doncs, es pretén realitzar una aplicació capaç de capturar imatges tèrmiques i visualitzar-les mitjançant una interfície gràfica. El dispositiu emprat per a la captura d'estes imatges és la càmera Seek Thermal Compact. Per a la captura i visualització, s'ha desenvolupat un algoritme que implementa estes funcions. Encara que el codi font s'ha desenrotllat en Ubuntu, el dispositiu sobre el qual es vol executar l'aplicació és una Raspberry Pi. Este projecte també pretén servir com a guia per a preparar la Raspberry Pi per a l'execució del programa creat. Per tal de poder desenvolupar esta aplicació, es requereix instal·lar les biblioteques OpenCV, libseekthermal, GTK+ 3 i gtkmm 3. La instal·lació d'estos paquets tant per a Ubuntu com per a Raspbian es detalla en aquest document.

Abstract

This document describes the procedures and knowledge needed to set up a thermal image display system to be used in case of fire emergency. Thus, we intend to develop an application capable of capturing thermal images and displaying them by means of a graphical user interface. The device used to capture these images is Seek Thermal Compact camera. An algorithm which implements capture and display has been developed. Although the development of the source code has been made in Ubuntu, the device on which the application will run is Raspberry Pi. This project is also offered as a helping guide to prepare Raspberry Pi to run the programme. In order to develop this application, the installation of OpenCV, libseekthermal, GTK+ 3 and gtkmm 3 libraries is required. The installation of these packages is herein detailed both for Ubuntu and for Raspbian.

Índice

Capítulo 1. Introducción

En este documento se pretende realizar un sistema de visualización para emergencias en caso de incendio. El objetivo último del mismo es crear una herramienta útil para ser usada por el agente a la hora de adentrarse en un incendio donde la la visibilidad es limitada. Para conseguir este objetivo, se desarrollará una aplicación de C++ que consiste en una interfaz gráfica en la que se visualicen las imágenes del sistema de visualización escogido. El dispositivo donde se ejecutará el programa será una Raspberry Pi.

Como sistema de visualización, se elige un visionado termográfico. Por tanto, para la visualización de imágenes térmicas, se precisa de una cámara térmica. La cámara escogida para llevar a cabo la tarea es la cámara Seek Thermal Compact de la compañía Seek Themal, Inc. Esta cámara se distingue de otras disponibles, como por ejemplo la cámara Flir Lepton, en dos principales aspectos, su doble resolución y su calibrado automático. Estas dos características suponen un avance considerable y es esta la razón por la cual se ha elegido para este proyecto.

Esta cámara está diseñada para ser utilizada en dispositivos móviles. Para poder usarse para el desarrollo de aplicaciones C++ en un ordenador se deben instalar unas librerías de Seek Thermal. Las librerías de esta biblioteca permitirán la comunicación de la cámara con el ordenador mediante conexión USB.

Para la creación de interfaces gráficas, se utilizarán las bibliotecas GTK+ 3 y gtkmm 3. Estas bibliotecas permitirán crear una interfaz gráfica completamente al gusto del desarrollador, desde las dimensiones o la distribución visual hasta el número de elementos o la forma de estos. Esta biblioteca será la que conseguirá el aspecto deseado para la aplicación final.

En cuanto a la Raspberry Pi, se abordará la instalación completa del sistema operativo así como la preparación del escritorio. Se trabajará con diferentes tipos de visualización que permitirán a los usuarios conectarse de forma remota y ser guiado por una persona externa al agente que se encuentra atrapado en el fuego del edificio o directamente, visualizar el espacio en primera persona.

A la hora de tratar con imágenes, se utilizarán para su procesamiento y manipulación, las librerías de la biblioteca OpenCV. Estas librerías permitirán la realización de rotaciones, filtrados, conversiones de espacios de color, etc. Esta amplia funcionalidad convierten a OpenCV en un elemento indispensable para la visualización de cámara térmicas.

La forma de trabajar será una constante a lo largo del proyecto. En primer lugar, se instalarán los paquetes y librerías necesarios en el sistema operativo Ubuntu ejecutándose en una máquina virtual de VMWare instalada en MacOS. El desarrollo de aplicaciones se realizará dentro del entorno de trabajo del IDE de Eclipse. Una vez, compilado, ejecutado y comprobado su correcto funcionamiento, se compila y ejecuta sobre Rapsbian, ya que éste es el sistema operativo donde ha de funcionar la aplicación final desarrollada.

Los objetivos que se abordan en este trabajo son:

- Creación de aplicaciones gráficas con GTK+ 3 y gtkmm 3.
- Instalación de librerías necesarias para la interacción con Seek Thermal.
- Manejo de las librerías de OpenCV para el procesamiento de imágenes.
- Instalación de la Raspberry Pi e instalación de paquetes en Raspbian.
- Comprensión de la teoría de la termografía para la captura de imágenes térmicas.

No son objeto de estudio en este proyecto la implementación de un sistema de visualización de datos provenientes de diferentes sensores para el control, por ejemplo, de la presión de oxígeno o el calor de las diferentes partes del traje del agente. Tampoco es objeto de investigación la forma de visualizar la interfaz gráfica de manera práctica por el agente, como pudiera ser un la visualización de la aplicación mediante el display de unas gafas inteligentes.

Capítulo 2. Seek Thermal

2.1 Introducción

Para este proyecto se dispone de una cámara Seek Thermal modelo Compact. Esta cámara está diseñada para ser usada en dispositivos móviles, tanto en plataforma Android como iOS. El precio de este modelo es de $249.

Se ha escogido la cámara Seek Thermal concretamente por diferentes motivos: Por un lado por la mayor resolución de su sensor (206x152), comparado con otros modelos de otras marcas como Flir Lepton cuya resolución óptima es de 80x60. Por otro lado, esta cámara tiene la función de autocalibrado, característica de la que carecen otras marcas en las que el calibrado se se realiza de manera externa.

Los usos de esta cámara son varios. Uno de ellos es la localización de fugas de agua o aire en las paredes, tejados o juntas de un edificio. Otra de las aplicaciones para la que se puede utilizar es detectar el mal funcionamiento de un motor mediante el sobrecalentamiento, produciendo grandes pérdidas de energía. Otro uso destacable de la Seek Thermal es la detección de pérdidas energéticas en calderas.

Figura 1. Cámara Seek Thermal Compact

Como se observa, las aplicaciones de esta cámara son varias pero siempre enfocadas a la eficiencia energética. En el caso de este proyecto, no se hace hincapié en ninguna función concreta, ya que simplemente se pretende la visualización de la imagen térmica.

Como se ha comentado, esta cámara está diseñada para ser usada en móviles con sistema operativo Android o iOS. En cambio, el desarrollo de la aplicación final se realiza en C++ para ser ejecutado en Ubuntu, en primera instancia, y finalmente en Raspbian. Para poder realizar el desarrollo de aplicaciones para C++ es necesario instalar una biblioteca de funciones que permita interactuar con la cámara Seek Thermal a través de Ubuntu y Raspbian. Para ello, se

accede a la biblioteca libseekthermal alojada en la plataforma de desarrollo colaborativo GitHub. Esta biblioteca contiene las funciones y clases necesarias para interactuar con la cámara térmica mediante código desarrollado en C++. Como se verá en apartados posteriores, la captura de imágenes se realizará mediante las librerías proporcionadas por libseekthermal, mientras que la visualización de estás imágenes en tiempo real se realizará mediante las librerías de la biblioteca OpenCV.

2.2 libseekthermal

2.2.1 Instalación en Ubuntu

En primer lugar hay que instalar la biblioteca. Esta biblioteca está alojada, como se ha indicado anteriormente, en GitHub. La dirección web donde se alojan las librerías para ser descargadas e instaladas es: https://github.com/ethz-asl/libseekthermal.

Hay dos maneras de instalar la biblioteca. La primera de ellas es añadiendo los repositorios de ésta e instalándolos directamente en el terminal de Ubuntu. Para Raspbian, por el contrario, se instala descargando las fuentes, como se indicará en el siguiente apartado.

Los pasos para instalar libseekthermal en Ubuntu son los siguientes:

```
sudo add-apt-repository ppa:ethz-asl/drivers
sudo apt-get update
sudo apt-get install libseekthermal*
```

Una vez finalizado este proceso, se dispone de todas las librerías necesarias para la correcta compilación y ejecución de aplicaciones que precisen de funciones y clases de la librería libseekthermal.

2.2.2 Instalación en Raspbian

Como se ha indicado anteriormente, para la correcta instalación en Raspbian de la biblioteca libseekthermal hay que realizar un procedimiento diferente. En primer lugar, hay que instalar unos paquetes que se indican como necesarios para la posterior instalación de la librería. Estos paquetes son: doxygen, pkg-config y ReMake.

A continuación, habrá que bajar las fuentes para proceder a su instalación. Para ello, se ejecuta en el terminal de Raspbian:

```
git clone https://github.com/ethz-asl/libseekthermal
```

Esta instrucción crea una carpeta en el directorio donde se esté situado con el nombre libseekthermal. El siguiente paso es crear, dentro de este directorio, la carpeta build para ejecutar cmake dentro de ésta e instalar la librería. Las instrucciones para realizar estos pasos son:

```
cd libseekthermal
mkdir build
cd build
cmake ..
make install
```

En el caso que durante la ejecución de cmake se produjera un error, este es producido por la falta de algún paquete, el cual se indica para ser instalado y se pueden volver a realizar los pasos de instalación de la biblioteca libseekthermal.

Ya por último, es necesario instalar la documentación de la librería para sus posibles consultas de funciones y clases. Para ello, se procede con los siguientes dos comandos:

```
sudo apt-get update
```

```
sudo apt-get install libseekthermal-doc
```

Una vez realizado este proceso, se puede dar por concluida la instalación de la biblioteca.

2.2.3 Cabeceras y librerías necesarias para el desarrollo en C++

Para poder saber qué cabeceras y librerías son necesarias, se utilizará el comando pkg-config, el cual devuelve las cabeceras o las librerías necesarias para la compilación.

Antes de solicitar mediante el comando pkg-config, es necesario actualizar la base de datos de los paquetes instalados, o es posible que no se encuentre la biblioteca libseekthermal si se ejecuta el comando pkg-config inmediatamente después de la instalación. Para actualizar la base de datos se utiliza el siguiente comando:

```
sudo updatedb
```

Una vez actualizada la base de datos con los últimos paquetes instalados, se hace uso del comando pkg-config mediante las siguientes dos instrucciones.

```
pkg-config --cflags libseekthermal
```

```
pkg-config --libs libseekthermal
```

Estas dos instrucciones devolverán por el terminal las siguiente cabeceras y librerías necesarias para la compilación de los programas desarrollados en C++.

```
tdi@ubuntu:~$ pkg-config --cflags libseekthermal
-I/usr/include/seekthermal
tdi@ubuntu:~$ pkg-config --libs libseekthermal
-lseekthermal-base -lseekthermal-aaa -lseekthermal-usb -lseekthermal-command -ls
eekthermal
```

Figura 2. Cabeceras y librerías de libseekthermal

Al incluir estas cabeceras y librerías en la compilación, no se producirá ningún tipo de error y se podrá ejecutar el programa sin problemas.

En el siguiente apartado, se explicará el algoritmo desarrollado en C++ para la captura de imágenes térmicas mediante la biblioteca libseekthermal y la visualización en tiempo real de estas imágenes mediante OpenCV.

2.3 Algoritmo de visualización de imágenes térmicas desarrollado mediante libseekthermal y OpenCV

2.3.1 Diagrama de bloques

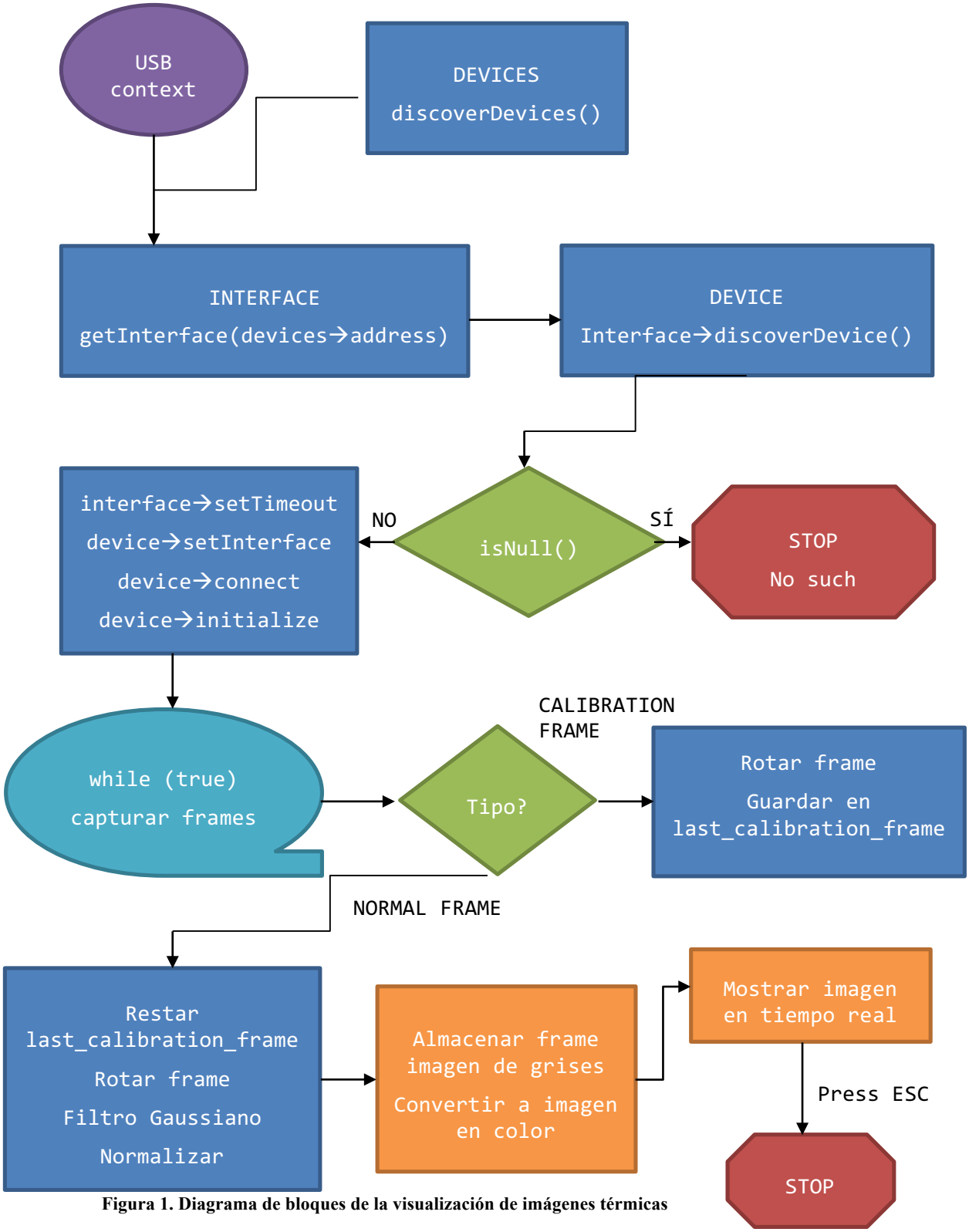

Figura 1. Diagrama de bloques de la visualización de imágenes térmicas

2.3.2 Algoritmo paso a paso

En este apartado, se desgranará punto por punto, el algoritmo mostrado anteriormente en el diagrama de bloques.

El código cuenta con 3 partes claramente diferenciables. La secuencia a seguir para la visualización de imágenes térmicas es la siguiente: conexión con la cámara térmica, captura de frames y visualización de frames. Para las dos primeras partes, se precisa el uso de las librerías libseekthermal; mientras que para la tercera parte, se utilizarán las librerías de OpenCV, puesto que es necesaria para la visualización de imágenes en tiempo real.

A continuación se explicará cómo realizar la conexión con la cámara térmica. El primero de los pasos, es encontrar todos los dispositivos Seek Thermal que se encuentren conectados al ordenador. Para llevar a cabo este proceso se utiliza la función **discoverDevices()**, que almacena en un listado de punteros de tipo device, todos los dispositivos Seek Thermal conectados al ordenador. Para poder continuar, es necesario declarar un contexto USB de Seek Thermal. En este contexto, se accede al interface correspondiente a la dirección del primer dispositivo de la lista generada por discoverDevices mediante la función **getInterface(address)**. Las direcciones son de la forma 00x:00x, donde los tres primeros dígitos se corresponden con el bus USB al que está conectado el dispositivo y el segundo grupo de dígitos, al número de dispositivo que le corresponde dentro del bus especificado. El siguiente paso es crear el objeto device a partir del interface proporcionado. Para ello, se realiza la función **discoverDevice()**, dentro de la clase interface.

En este punto debería haberse creado un objeto device, correspondiente al dispositivo del interface creado. Es necesario saber si se ha creado este objeto device. Para ello, se hace uso de la función **isNull()**. Esta función devuelve un valor booleano true, en caso de no encontrar ningún dispositivo, o false, si se ha creado un objeto device. En el caso de no haber sido creado, se muestra un mensaje por pantalla que muestra "No such device", indicando que no se ha encontrado ningún dispositivo y cerrando el programa. Si se ha creado correctamente, se puede seguir configurando la conexión con el dispositivo para capturar los frames de la cámara térmica.

El siguiente paso, es establecer un timeout mínimo para el interfaz. En caso de no hacerlo, el programa abortará la captura y mostrará el siguiente mensaje: "USB error: Operation timed out". Este timeout se realiza mediante la función **setTimeout(1)**. El siguiente paso es acceder al interface del objeto device. Para realizar el acceso se utiliza la función **setInterface(interface)**. Ya sólo quedaría para tener completamente en funcionamiento el dispositivo, realizar la conexión con el mismo e inicializarlo. Para ello, se hace uso de las funciones **connect()** y **initizalize()**.

Una vez realizada la conexión e inicialización del dispositivo, el siguiente paso es capturar en el ordenador los frames que captura la cámara térmica. Cada frame consta de 32448 palabras (acordes a la resolución del sensor de 206x152) de 16 bits. Cada palabra se transmite mediante el sistema little-endian, aunque solo se utilizan 14 de los 16 bits de la palabra. El frame no contiene ninguna cabecera. En lugar de ello, los metadatos se almacenan en lugares especiales del pixel. Cuando se realiza una petición por parte del host USB, el frame se transmite a una velocidad de 8 fps.

A partir de este momento se crea un bucle infinito para la captura de los frames (en cada iteración del bucle se captura un frame nuevo). Para la captura, se crea primero un puntero de tipo frame para almacenar los frames capturados. Para la captura y almacenado de los frames en la variable frame, se utilizará la función **capture(*frame)**. Una vez almacenada la captura en la variable frame, hay que distinguir si se trata de uno de tipo normal o de tipo calibración. Para saber de qué tipo de frame se trata, se hará uso de la función **getType()**. Esta función puede devolver unos de tipo inválido, normal, calibración, precalibración o desconocido.

Si el frame es de tipo calibración, se rota este en el sentido de las agujas del reloj para obtener frames de tipo apaisado y no verticales mediante la función **rotateClockwise()**; y se guarda en una variable creada anteriormente con el nombre last_calibration_frame.

En el caso de tratarse de un frame de tipo normal, en primer lugar, se rota el frame igualmente mediante **rotateClockwise()**. Para suavizar el frame, se aplicará un filtro gaussiano mediante la función **gaussianBlur()**. Seguidamente, se almacenan en variables tipo size_t, la anchura y la altura por separado mediante las funciones **getWidth()** y **getHeight()**. Para poder realizar la conversión del objeto frame, se han de normalizar los valores obtenidos por el frame entre 0 y 1. Para ello, se seleccionarán los valores mayor y menor del frame mediante las funciones **getMinimumValue()** y **getMaximumValue()**, que devuelven un valor float con el valor máximo y mínimo del frame. Estos valores se traducirán como 1 y 0 respectivamente al producirse la normalización, llevada a cabo por **normalize(min,max)**. Una vez normalizado el frame, se creará una variable de OpenCV tipo Mat de 8 bits unsigned char de 1 canal (escala de grises). Esta imagen tomará la misma anchura y altura que el frame. Mediante un doble bucle for para recorrer la imagen pixel a pixel, se asigna a cada uno de estos el valor del frame multiplicado por 255 para obtener los valores correspondientes a la imagen en escala de grises. El último paso es obtener una imagen a color a partir de la imagen de grises. Para ello se crea una variable Mat de OpenCV de 8 bits unsigned char de 3 canales(R, G y B). Para rellenar esta imagen con los colores correspondientes, para cada canal de cada pixel de la imagen, se realiza la conversión de gris a su valor correspondiente para ese canal.

Ya por último, se visualiza la imagen mediante la función **imshow("Nombre ventana", imagen)**. Esta imagen se actualizará cada vez que se realice la captura de un frame nuevo desde la cámara térmica. Para finalizar la ejecución del programa, se configura la ejecución de un break al pulsar la tecla ESC.

2.4 Compilación y ejecución en Ubuntu

La compilación y ejecución del programa se realiza en dos pequeños pasos. Si se han incluido las librerías indicadas por pkg-config de OpenCV y libseekthermal, solamente hay que añadir en las opciones de la instrucción del compilador –std=c++11. Esto se debe a que, a la hora de compilar, el compilador muestra un error debido a la necesidad de utilizar el compilador y soporte de librería para el estándar ISO C++ 2011.

Para ejecutar, simplemente hay que pulsar el botón de Run. El segundo problema se produce en este punto. Al intentar ejecutar, el programa aborta la ejecución y aparece el siguiente mensaje de error: "USB error: Access denied". Este error se debe a que la cámara Seek Thermal no tiene en el ordenador privilegios de superusuario necesarios para poder acceder al puerto USB. Para ello, desde el terminal, se entregarán estos privilegios. El primer paso, es encontrar donde se encuentra nuestra cámara conectada. Una vez hecho esto, se procede a entregar privilegios root a la cámara térmica. Para realizar estos pasos, en el terminal se deben ejecutar las siguientes instrucciones:

```
-ls –l /dev/bus/usb/00*
```

Esta instrucción devuelve un listado de los dispositivos USB conectados a cada uno de los buses USB. Una vez localizado, se entregan privilegios root con la siguiente instrucción:

```
sudo chmod a+w /dev/bus/usb/00x/00x
```

Como se puede apreciar, las "x" han de ser sustituidas por los números correspondientes para hacer referencia al dispositivo USB correspondiente a la cámara térmica.

En la siguiente figura, se muestra el correcto uso, de este par de instrucciones:

```
tdi@ubuntu:~$ ls -l /dev/bus/usb/00*
/dev/bus/usb/001:
total 0
crw-rw-r-- 1 root root    189, 0 Jun 23 15:28 001
crw-rw-r-- 1 root plugdev 189, 8 Jun 30 10:23 009

/dev/bus/usb/002:
total 0
crw-rw-r-- 1 root root 189, 128 Jun 23 15:28 001
crw-rw-r-- 1 root root 189, 129 Jun 23 15:28 002
crw-rw-r-- 1 root root 189, 130 Jun 23 15:28 003
crw-rw-r-- 1 root root 189, 140 Jun 30 10:09 013
tdi@ubuntu:~$ sudo chmod a+w /dev/bus/usb/001/009
```

Figura 4. Privilegios root para la cámara Seek Thermal

Como se puede observar, la cámara térmica se encuentra en 001/009. Es a esta dirección a la que se le da privilegios root.

A continuación, se muestra una imagen térmica de las captadas por la cámara Seek Thermal y que son mostradas mediante la función imshow de OpenCV.

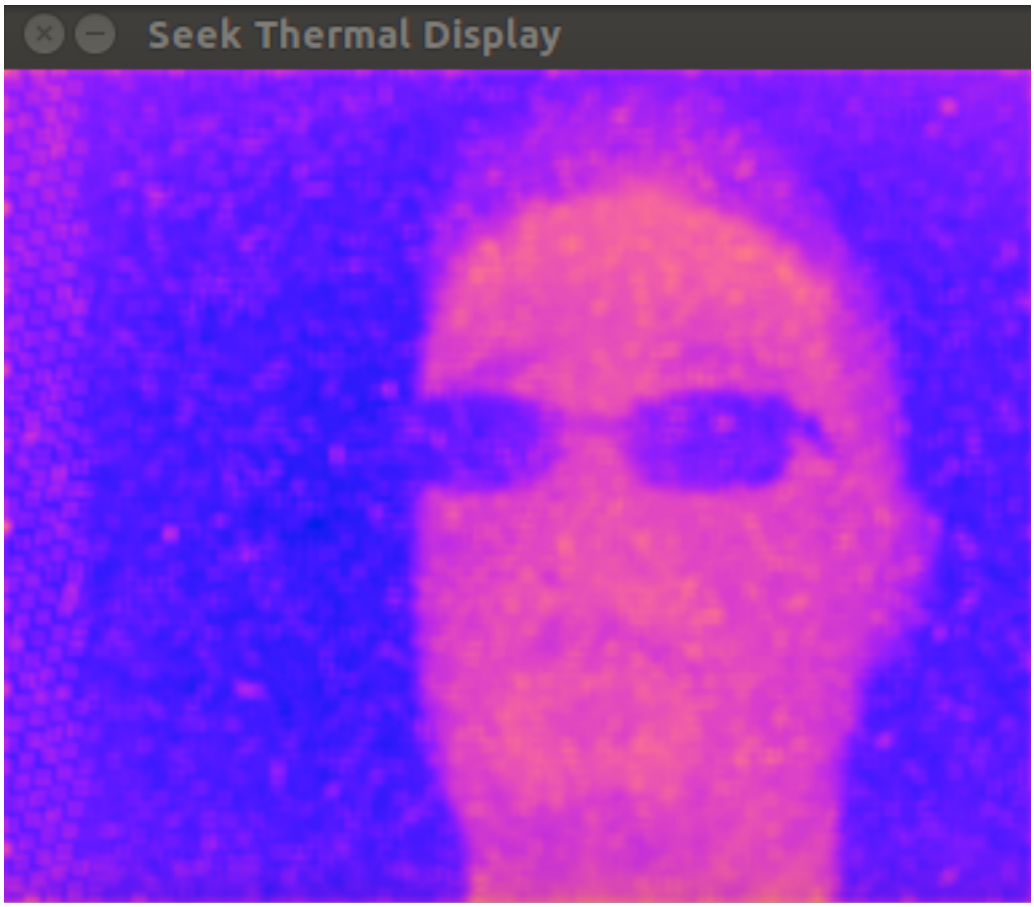

Figura 5. Imagen térmica visualizada en Ubuntu

2.5 Compilación y ejecución en Raspbian

Antes de realizar la compilación del programa en Raspbian, se otorgarán privilegios root a la cámara térmica. La forma de realizarlo es mediante las dos mismas instrucciones que en el apartado anterior.

```
-ls –l /dev/bus/usb/00*
```

```
sudo chmod a+w /dev/bus/usb/00x/00x
```

Una vez otorgados los privilegios root a la cámara, ya se puede iniciar la compilación. A diferencia de la compilación en Ubuntu, en Raspbian no se dispone de un IDE para desarrollo de programas en C++. Es por ello que la compilación en Raspbian se realiza desde el terminal.

El primer paso para la compilación será crear una carpeta. Dentro de esta, se copia el archivo del código fuente para la visualización de imágenes térmicas. Desde el terminal, se accede al interior de esta carpeta y se ejecuta la siguiente instrucción para la compilación del código fuente.

```
g++ -std=c++11 srun.cpp -o camaraseek `pkg-config --cflags --libs opencv` `pkg-config --cflags --libs libseekthermal`
```

A continuación se explicará cada uno de los parámetros de la compilación:

- g++: es el compilador para C++.
- -std=c++11: estándar ISO C++ 2011.
- srun.cpp: nombre del archivo del código fuente.
- -o camaraseek: nombre que se le da al ejecutable creado en la compilación
- `pkg-config --cflags --libs opencv`: sirve para incluir las cabeceras y las librerías de la biblioteca OpenCV en la compilación.
- `pkg-config --cflags --libs libseekthermal`: sirve para incluir las cabeceras y las librerías de la biblioteca libseekthermal en la compilació

Si la compilación se realiza correctamente, se creará en la carpeta donde se ha ejecutado la compilación una archivo ejecutable con el nombre camaraseek.

Para la ejecución del ejecutable, simplemente se tiene que escribir en el terminal de Raspbian la siguiente instrucción:

```
./camaraseek
```

Al iniciar el programa, la cámara comienza a capturar frames y a mostrar en tiempo real las imágenes térmicas de la captura.

La siguiente figura es una muestra de la visualización del programa creado en el escritorio de Raspbian.

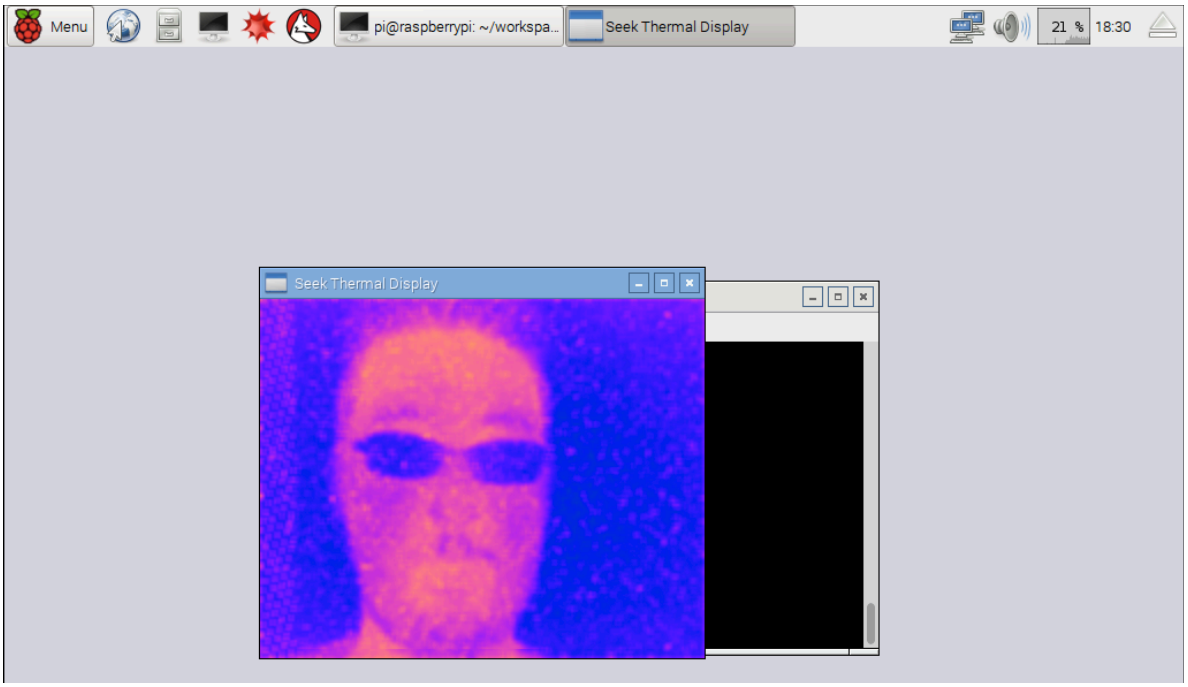

Figura 6. Imagen térmica visualizada en Raspbian

Capítulo 3. Aplicación final: Interfaz gráfica con GTK+

3.1 GTK+

GTK+ es un conjunto de herramientas multiplataforma diseñadas para la creación de interfaces gráficas. GTK+ ofrece una amplia gama de funcionalidades para poder diseñar desde pequeñas aplicaciones hasta una suite de aplicaciones completa. Cabe destacar el hecho de que GTK+ es usado hoy en día en todo tipo de aplicaciones para prácticamente cualquier tipo de sistema operativo. Destacan entre estas aplicaciones: GIMP, GNOME Desktop, VMWare Workstation y muchas más.

A pesar de estar escrito en C, GTK+ está pensado para ser utilizado en muchos más lenguajes, no sólo en C/C++. El uso de lenguajes como Perl o Python, entre otros, constituye igualmente un método eficaz para el diseño de estas aplicaciones.

GTK+ es un software libre perteneciente al proyecto GNU. Como se indica en los términos de licencia para GTK+, está permitido su uso para todo tipo de desarrolladores, incluidos aquellos aquellos de software propietario, sin ningún tipo de impuesto de licencia.

En este proyecto se utilizará la versión GTK+ 3.

3.1.1 Instalación de GTK+ 3 en Ubuntu

A continuación, se detallarán los pasos para la correcta instalación de GTK+ en Ubuntu. Para la realización de este proyecto, se instalará en concreto la versión GTK+ 3. Esta versión requiere una serie de librerías complementarias a GTK+ 3 para el completo desarrollo de todo tipo de aplicaciones. Estas librerías son:

- GLib
- Pango
- Gdk-Pixbuf
- ATK
- GObject-Instrospection

Estas librerías, son buscadas cuando se va a instalar GTK+ 3. En el caso de no estar instaladas en el ordenador, la instalación propone ser instaladas. Para la instalación de GTK+ 3, hay que teclear en el terminal de Ubuntu el siguiente par de comandos:

```
sudo apt-get update
```

```
sudo apt-get install libgtk-3-dev
```

Una vez completada la instalación, ya se dispone de los paquetes necesarios para desarrollar aplicaciones que precisen de la librería GTK+ 3.

3.1.2 Cabeceras y librerías necesarias para la compilación de GTK+ 3 en Ubuntu

Para poder saber qué cabeceras y librerías son necesarias, se utilizará el comando pkg-config. Este comando, como se ha comentado en apartados anteriores, sirve para conocer las cabeceras o librerías necesarias para la compilación de una aplicación.

Antes de solicitarlas mediante el comando pkg-config, es necesario actualizar la base de datos de los paquetes instalados, o es posible que no se encuentre la biblioteca gtk+-3.0 si se ejecuta el comando pkg-config inmediatamente después de la instalación. Para actualizar la base de datos se utiliza el siguiente comando:

```
sudo updatedb
```

Una vez actualizada la base de datos con los últimos paquetes instalados, se hace uso del comando pkg-config mediante las siguientes dos instrucciones.

```
pkg-config --cflags gtk+-3.0
```

```
pkg-config --libs gtk+-3.0
```

Las cabeceras y librerías necesarias para la ejecución de este proyecto son las mostradas en la siguiente figura.

```
tdi@ubuntu:~$ pkg-config --cflags gtk+-3.0
-pthread -I/usr/include/gtk-3.0 -I/usr/include/atk-1.0 -I/usr/include/at-spi2-at
k/2.0 -I/usr/include/pango-1.0 -I/usr/include/gio-unix-2.0/ -I/usr/include/cairo
 -I/usr/include/gdk-pixbuf-2.0 -I/usr/include/glib-2.0 -I/usr/lib/i386-linux-gnu
/glib-2.0/include -I/usr/include/harfbuzz -I/usr/include/freetype2 -I/usr/includ
e/pixman-1 -I/usr/include/libpng12
tdi@ubuntu:~$ pkg-config --libs gtk+-3.0
-lgtk-3 -lgdk-3 -latk-1.0 -lgio-2.0 -lpangocairo-1.0 -lgdk_pixbuf-2.0 -lcairo-go
bject -lpango-1.0 -lcairo -lgobject-2.0 -lglib-2.0
tdi@ubuntu:~$
```

Figura 7. Cabeceras y librerías necesarias para la compilación de GTK+ 3

3.1.3 Instalación de GTK+ 3 en Raspbian

La instalación de GTK+3 en Raspbian se realiza de la misma forma que en Ubuntu. Por tanto, a la hora de instalar, simplemente hay que utilizar el comando apt-get install y se dispondrá de las librerías necesarias para el desarrollo de aplicaciones basadas en las librerías proporcionadas por GTK+ 3.

3.1.4 Cabeceras y librerías necesarias para la compilación de GTK+ 3 en Raspbian

Para conocer las cabeceras y librerías necesarias para la compilación de GTK+ 3 en Raspbian, es necesario utilizar nuevamente el comando pkg-config. Si se analizan las diferencias respecto al resultado de ejecutar pkg-config en Ubuntu, se observa que tanto librerías como cabeceras, son las mismas. La única diferencia radica en el path hasta llegar a las cabeceras, que es distinto, puesto que varía de un ordenador a otro.

3.2 gtkmm

Como se ha explicado en el punto anterior, GTK+ es la biblioteca utilizada para el desarrollo de aplicaciones gráficas. Sin embargo, para el desarrollo de este proyecto se utilizará GTK+ pero empaquetado en una biblioteca llamada gtkmm.

El uso de gtkmm permite al desarrollador de C++ usar técnicas comunes de este lenguaje como pueden ser la derivación, el polimorfismo o la encapsulación. Esto permite a un desarrollador de C++ tener el código organizado, acorde al estilo de cualquier otro código C++ de cualquier otra aplicación.

La gran ventaja de usar gtkmm es la herencia de objetos. La derivación y la creación de nuevos widget directamente con GTK+ es bastante compleja. El uso de la herencia hace que esta tarea con gtkmm sea mucho menos complicada, ya que es una de las señas de identidad de la programación orientada a objetos. Otra ventaja a destacar es el hecho de que en gtkmm se utiliza menos código que en GTK+ para crear la misma aplicación.

En este proyecto, se utilizará en concreto la versión gtkmm 3.

3.2.1 Instalación de gtkmm 3 en Ubuntu y Raspbian

La instalación en ambos sistemas operativos se realiza de la misma forma. El procedimiento de instalación de la biblioteca gtkmm es muy similar al de GTK+ 3. Antes de seguir, es muy importante que los siguientes paquetes estén instalados en el sistema operativo:

- GTK+
- libsigc++

Evidentemente, en este proyecto GTK+ estará instalado. Si libsigc++ no se encuentra instalado en el ordenador, automáticamente la instalación incluye este paquete con el resto a instalar. Las instrucciones a seguir para instalar desde el terminal son:

```
sudo apt-get update
sudo apt-get install libgtkmm-3.0-dev
```

Una vez acabada la instalación, gtkmm 3 se encuentra totalmente operativo para desarrollar aplicaciones.

3.2.2 Cabeceras y librerías para la compilación de gtkmm en Ubuntu y Raspbian

Para averiguar qué paquetes se necesitan para la correcta compilación de gtkmm, se vuelve a proceder de la misma forma que en apartados anteriores. Siguiendo estos tres comandos, se dispondrá de la lista de cabeceras y librerías necesarias para la compilación.

```
sudo updatedb
pkg-config --cflags gtk+-3.0
pkg-config --libs gtk+-3.0
```

En la siguiente figura, se muestran las cabeceras y librerías indicadas por el comando pkg-config.

```
tdi@ubuntu:~$ pkg-config --cflags gtkmm-3.0
-pthread -I/usr/include/gtkmm-3.0 -I/usr/lib/i386-linux-gnu/gtkmm-3.0/include -I
/usr/include/atkmm-1.6 -I/usr/include/giomm-2.4 -I/usr/lib/i386-linux-gnu/giomm-
2.4/include -I/usr/include/pangomm-1.4 -I/usr/lib/i386-linux-gnu/pangomm-1.4/inc
lude -I/usr/include/gtk-3.0 -I/usr/include/cairomm-1.0 -I/usr/lib/i386-linux-gnu
/cairomm-1.0/include -I/usr/include/gdk-pixbuf-2.0 -I/usr/include/gtk-3.0/unix-p
rint -I/usr/include/gdkmm-3.0 -I/usr/lib/i386-linux-gnu/gdkmm-3.0/include -I/usr
/include/atk-1.0 -I/usr/include/glibmm-2.4 -I/usr/lib/i386-linux-gnu/glibmm-2.4/
include -I/usr/include/glib-2.0 -I/usr/lib/i386-linux-gnu/glib-2.0/include -I/us
r/include/sigc++-2.0 -I/usr/lib/i386-linux-gnu/sigc++-2.0/include -I/usr/include
/pango-1.0 -I/usr/include/cairo -I/usr/include/pixman-1 -I/usr/include/freetype2
 -I/usr/include/libpng12 -I/usr/include/at-spi2-atk/2.0 -I/usr/include/gio-unix-
2.0/ -I/usr/include/harfbuzz
tdi@ubuntu:~$ pkg-config --libs gtkmm-3.0
-lgtkmm-3.0 -latkmm-1.6 -lgdkmm-3.0 -lgiomm-2.4 -lpangomm-1.4 -lgtk-3 -lglibmm-2
.4 -lcairomm-1.0 -lgdk-3 -latk-1.0 -lgio-2.0 -lpangocairo-1.0 -lgdk_pixbuf-2.0 -
lcairo-gobject -lpango-1.0 -lcairo -lsigc-2.0 -lgobject-2.0 -lglib-2.0
```

Figura 8. Uso de pkg-config con gtkmm en Ubuntu

3.3 Desarrollo de una aplicación gráfica para el visionado de imágenes térmicas

Una vez instaladas todas las bibliotecas y paquetes necesarios para el desarrollo de interfaces gráficas, se aproxima el final de este trabajo. El objetivo final como se ha indicado al principio del mismo, es el desarrollo de una interfaz gráfica que contenga una ventana para el visionado de imágenes térmicas a la vez que una barra con indicadores para la visualización futura de presión de oxígeno, temperatura en diferentes partes del traje o monitor de frecuencia cardíaca.

A continuación se muestra una captura de la interfaz gráfica desarrollada:

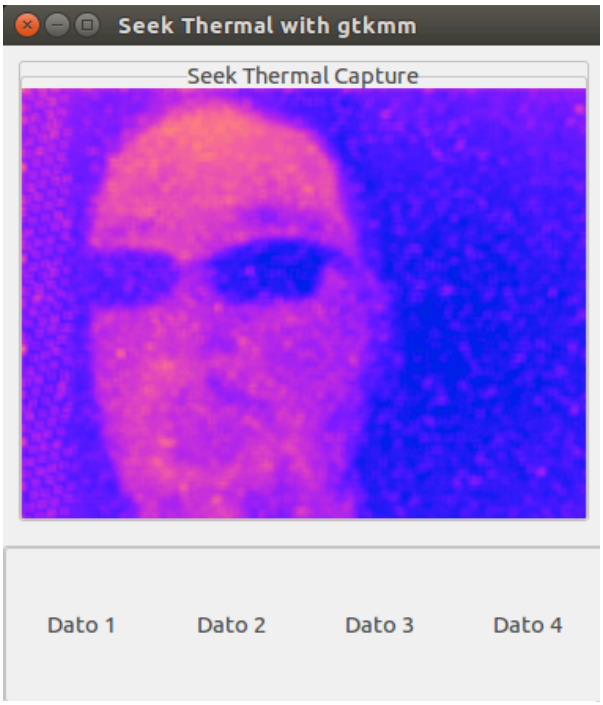

Figura 9. Vista de la interfaz gráfica creada

En los siguientes apartados, se detallará función por función, el proceso que realiza la aplicación para crear una interfaz gráfica que visualice las imágenes térmicas. Para este proyecto no se incluye la representación de los datos de los sensores. Esto es sustituido por 4 labels (dato 1, dato 2, dato 3 y dato 4).

3.4 Algoritmo desarrollado para la creación de la interfaz gráfica

3.4.1 Relación entre funciones

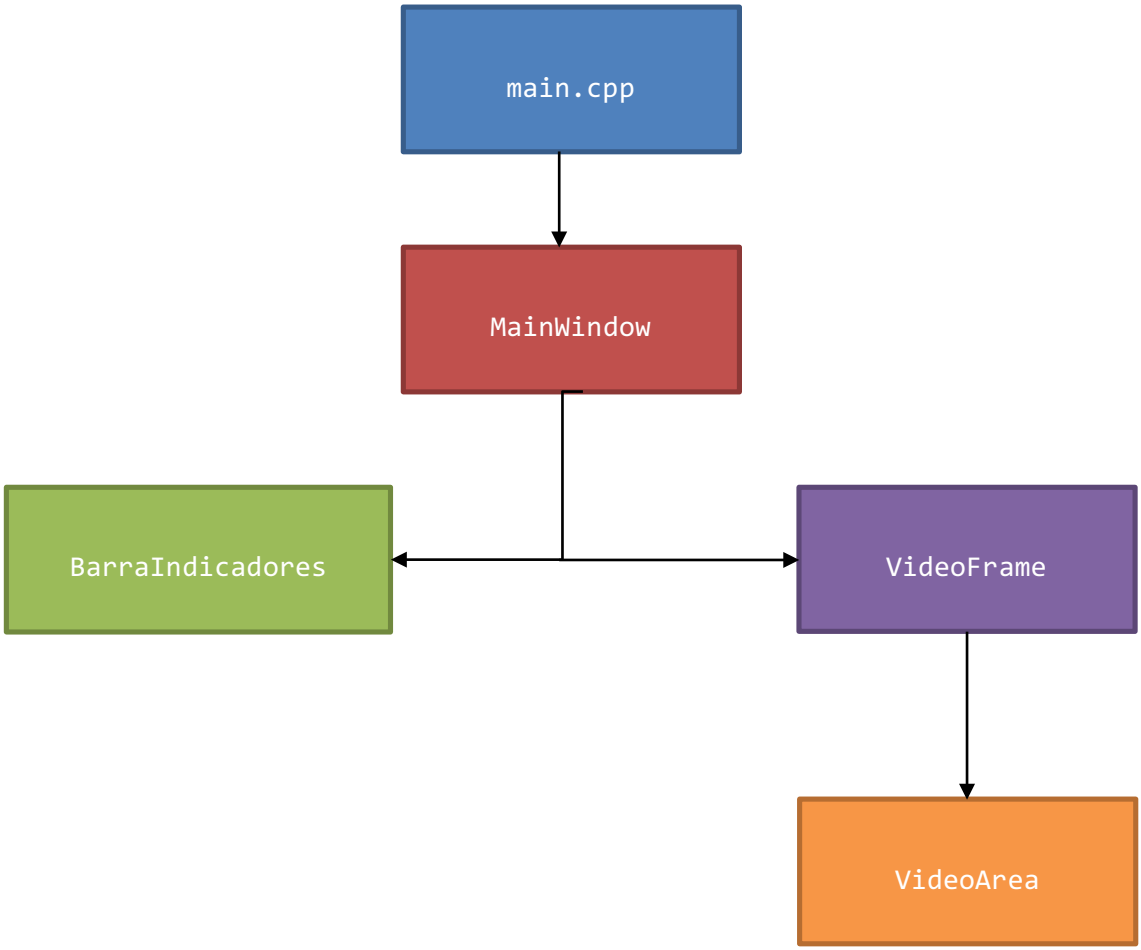

Figura 10. Esquema general de la aplicación

3.4.2 main.cpp

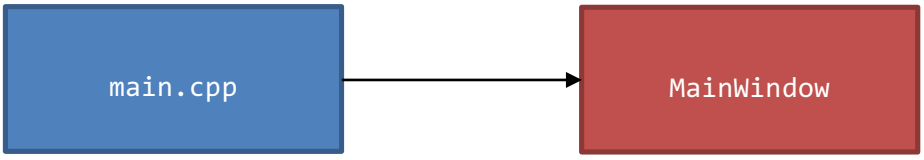

Figura 11. Jerarquía main.cpp

La función main.cpp es la encargada de crear la aplicación. Esta función es la principal y contiene el resto de clases a través de la clase MainWindow. En esta función main.cpp se crea un objeto de la clase MainWindow, que contiene el resto de clases que constituyen la interfaz gráfica. A través de la llamada a la función run(MainWindow) se pone en marcha la aplicación creada.

3.4.3 *MainWindow*

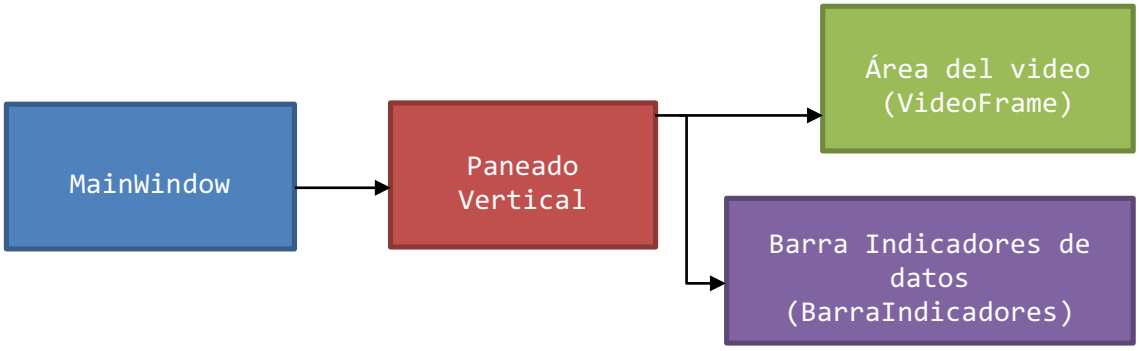

Figura 12. Jerarquía clase MainWindow

La clase MainWindow consta de una archivo fuente (MainWindow.cpp) y un archivo de cabecera (MainWindow.h).

En MainWindow.h se declaran todos los objetos que interactúan en MainWindow.cpp. Estos objetos son tres:

- Paneado: Se crea un objeto de la clase Gtk::Paned. Mediante este objeto se dividirá la ventana principal en dos subventanas.
- Área del vídeo: Se crea un objeto de la clase VideoFrame. El contenido de esta clase se detallará en puntos posteriores. La función de este objeto es mostrar las imágenes térmicas en tiempo real.
- Barra de indicadores de datos: Se crea un objeto de la clase BarraIndicadores. Esta clase, al igual que la clase Video Frame, se detallará posteriormente. La barra consiste en la visualización de 4 labels con el nombre del dato al que sustituyen.

Una vez declarados los objetos, hay que establecer la relación entre ellos. De esto se encarga MainWindow.cpp. Esta función establece un paneado vertical, donde se sitúa el objeto VideoFrame en la parte superior y el objeto BarraIndicadores en la parte inferior. A continuación se detallarán las clases BarraIndicadores y VideoFrame.

3.4.4 *BarraIndicadores*

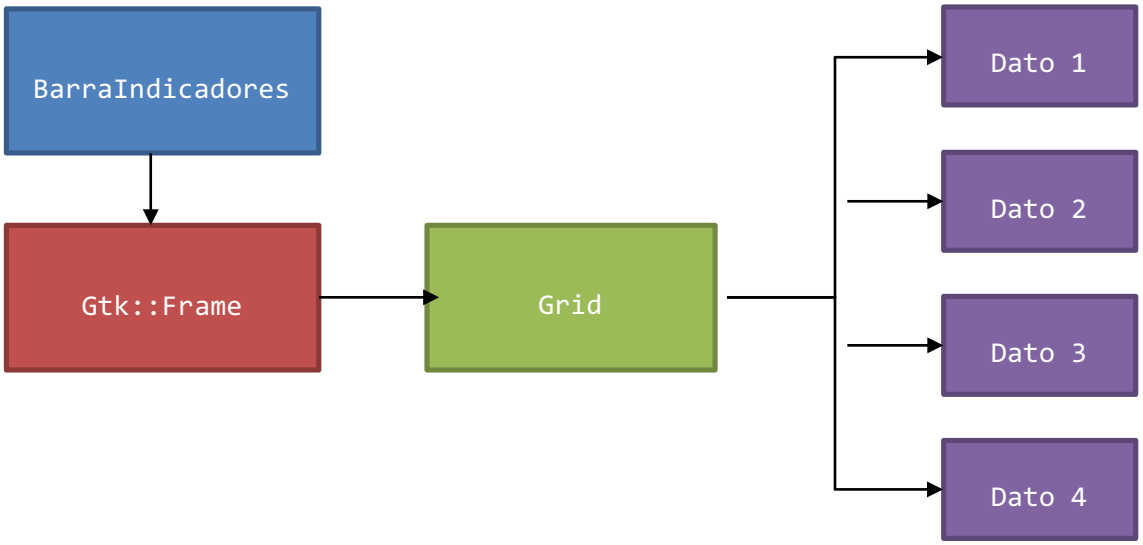

Figura 13. Jerarquía de la clase BarraIndicadores

La clase BarraIndicadores consta también de un archivo fuente (BarraIndicadores.cpp) y un archivo de cabecera (BarraIndicadores.h).

Al igual que en la clase anterior, el archivo BarraIndicadores.h contiene la declaración de todos los objetos que interactúan en BarraIndicadores.cpp. Estos objetos son:

- Frame: es un objeto de la clase Gtk::Frame. Es el encargado de contener objetos dentro de él. En este caso, será el objeto grid.
- Grid: es un objeto de la clase Gtk::Grid. Es el objeto que contiene, todos los objetos dato.
- Dato 1-4: son objetos de la clase Gtk::Label. Este objeto muestra el string que se le indica. Para este caso, será Dato x, donde x es un número desde 1 hasta 4.

Una vez declarados los objetos, el archivo BarraIndicadores.cpp es el encargado de realizar las conexiones e interacciones de todos los objetos. En primer lugar, se declara que la anchura y la altura de cada fila y columna del grid sea homogénea. Al hacer esto, se consigue una visualización de los datos proporcionada y agradable. En segundo lugar, se añaden todos los labels, uno a continuación del otro, de manera que el grid resultante tiene 1 fila y 4 columnas. Después de tener el grid preparado, se añade al objeto frame y se utiliza la función **show_all()** para mostrar la barra con los cuatro datos en la interfaz.

3.4.5 VideoFrame

Figura 14. Jerarquía de la clase VideoFrame

Al igual que en las dos clases anteriores, la clase VideoFrame consta de un archivo de cabecera (VideoFame.h) y un archivo fuente (VideoFrame.cpp).

En este caso la declaración de los objetos que van a interactuar en la función VideoFrame.cpp es la siguiente:

- Frame: como se ha indicado en el apartado anterior, se trata de un objeto de la clase Gtk::Frame, cuya función es almacenar objetos dentro de este. Para la clase VideoFrame, contendrá el objeto VideoArea.
- VideoArea: este objeto pertenece a la clase VideoArea. Esta clase ha sido creada en concreto para esta aplicación y se centra en la visualización de imágenes térmicas. Los detalles de esta clase se expondrán en el siguiente apartado.

En el archivo VideoFrame.cpp se establecerán las relaciones pertinentes entre los objetos declarados en el archivo VideoFrame.h.

El primer paso es configurar las propiedades del objeto frame para contener el objeto VideoArea. Para ello, se establece como título centrado del objeto frame “Seek Thermal Capture” y se establece un tamaño de ventana acorde al tamaño de las imágenes capturadas por la cámara térmica.

Una vez configuradas las propiedades del objeto frame, se añade a este el objeto de la clase VideoArea se utiliza la función **show_all()** para mostrar el objeto frame y su contenido en la interfaz.

3.4.6 VideoArea

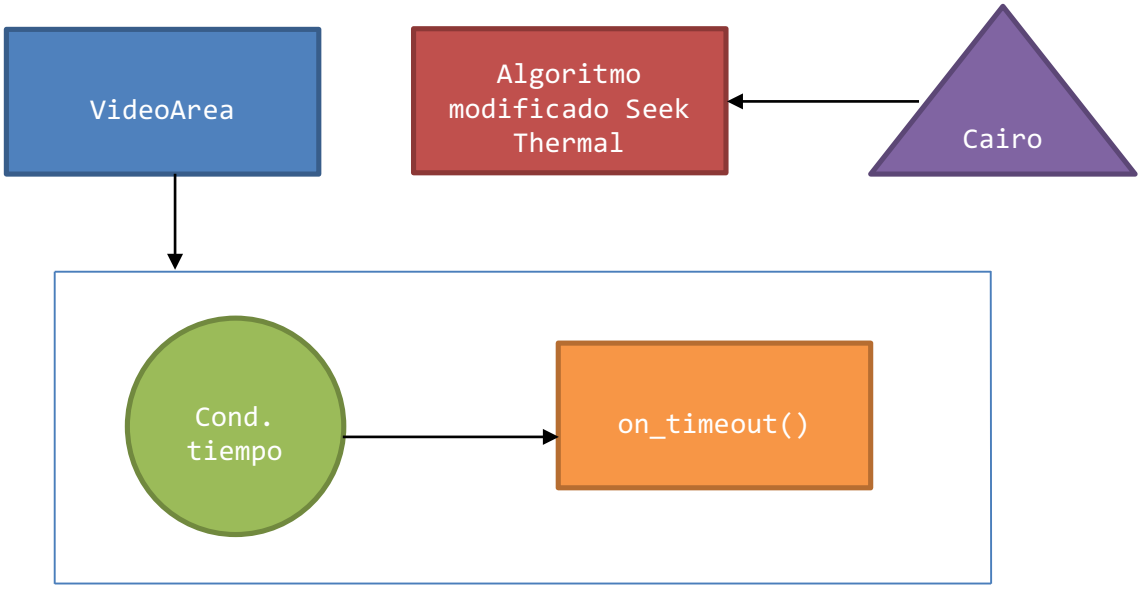

Figura 15. Jerarquía de la clase VideoArea

A pesar de tener también dos archivos, uno fuente (VideoArea.cpp) y otro de cabecera (VideoArea.h), en el archivo de cabecera, en este caso, no se declaran objetos, sino llamadas a funciones, las cuales estarán desarrolladas en el archivo VideoArea.cpp. Por otro lado, la clase VideoArea se considerará como public Gtk::DrawingArea. Las funciones que se incluyen en VideoArea.h son:

- **onDraw(const Cairo::RefPtr<Cairo::Context> &cr)**: función encargada de visualizar un fotograma.
- **on_timeout()**: función encargada de refrescar la imagen para visualizar un nuevo fotograma.

Una vez declaradas las funciones, se va a detallar el uso y la función de cada una de ellas en el archivo VideoArea.cpp

La primera función que se analizará es **on_timeout()**. Se trata de una función bool que se activará cada 50 ms gracias al uso de la función **Glib::signal_timeout().connect**. Esta función tiene como argumentos, la función a activar y el tiempo necesario para ser activada. En el momento que se cumple el tiempo de timeout, se realiza una llamada a la función **on_timeout()**. Esta función tiene como finalidad forzar al área donde se visualizan los frames, a renovar la visualización de un nuevo fotograma.

La segunda función a analizar es **onDraw(const Cairo::RefPtr<Cairo::Context> &cr)**. Es en esta función donde se aplica el algoritmo explicado en el capítulo anterior. Únicamente se produce una modificación respecto al apartado anterior y es el método empleado para la visualización de los fotogramas de la cámara térmica. Cuando únicamente se pretendía visualizar los fotogramas capturados por la Seek Thermal, se empleaba la función **imshow()** para mostrar los fotograma en tiempo real. Para poder, mostrar una imagen desde una fuente, ya sea un archivo de imagen, una captura de video o una secuencia continua de fotogramas, en Gtk se utilizan los widgets denominados DrawingArea. En esta zona es donde se produce la visualización de la imagen. Para conseguir dibujar esta imagen, se hace uso de los contextos de la librería Cairo. En esta función el contexto de Cairo se denomina cr. El procedimiento para visualizar la imagen es bastante sencillo. Una vez se tenga la imagen a color, para conseguir la visualización, se asigna la imagen al contexto de la librería Cairo mediante la función **Gdk::Cairo::set_source_pixbuf**. Una vez asignada la imagen al contexto de Cairo, para

visualizar la imagen de la cámara térmica en el widget DrawingArea, se hace uso de la función **paint()**.

3.5 Compilación y ejecución en Ubuntu

A la hora de compilar desde el IDE de Eclipse, solo hay que comprobar que se haya importado correctamente las cabeceras y enlazado las librerías sin ningún tipo de problema. Si se ha realizado correctamente esta tarea, no debe haber ningún problema.

El siguiente paso es la ejecución del programa. Para ello, hay que tener en cuenta que la cámara Seek Thermal no tiene privilegios de superusuario para acceder al puerto USB. Para ello, se procede igual que en el capítulo anterior cuando se pretendía dar privilegios root a la cámara térmica.

3.6 Compilación y ejecución en Raspbian

La compilación en Raspbian se realiza de forma diferente que en Ubuntu, puesto que no se tiene instalado ningún IDE de desarrollo. Para la compilación de este programa, se deben ejecutar las siguientes instrucciones en el terminal de Raspbian:

```
cd /path hasta la carpeta donde se encuentren los códigos fuente
```

```
g++ -std=c++11 main.cpp BarraIndicadores.cpp MainWindow.cpp VideoArea.cpp VideoFrame.cpp -o guiRun `pkg-config --cflags --libs opencv` `pkg-config --cflags --libs libseekthermal` `pkg-config --cflags --libs gtkmm-3.0`
```

La compilación crea un ejecutable llamado guiRun. Antes de ejecutar el programa, se debe entregar, en el caso de que no los posea, privilegios root a la cámara SeekThermal para no tener problemas de acceso USB. Una vez entregados estos privilegios y estando en la misma carpeta donde se ha creado el ejecutable, se ejecuta el programa mediante la siguiente instrucción:

```
./guiRun
```

Capítulo 4. Raspberry Pi

4.1 Introducción

Raspberry Pi es un ordenador de bajo coste desarrollado en Inglaterra para facilitar la entrada de ordenadores a las escuelas. Cuenta con entradas estandarizadas para teclado y ratón así como un sistema operativo instalado en una tarjeta SD. El escritorio de Raspberry Pi puede ser visualizado en cualquier monitor de ordenador o pantalla de televisión. Este ordenador permite realizar prácticamente cualquier actividad que se proponga, desde jugar a videojuegos hasta procesado de texto, pasando por reproducción de vídeo de alta definición.

Para este proyecto se dispone de una Raspberry Pi 2 Model B V1.1 así como de un display Raspberry Pi 7" TouchScreen Display.

Raspberry Pi 2 Model B V1.1 es el sustituto del modelo Raspberry Pi 1 Model B+. Sus características técnicas son las siguientes:

- CPU ARM Cortex-A7 de cuatro núcleos a 900 MHz.
- 1 GB de memoria RAM.
- 4 puertos USB.
- 40 pines GPIO.
- Puerto Full HDMI.
- Puerto Ethernet.
- Entrada combinada de 3.5 mm de audio jack y vídeo compuesto.
- Camera Interface (CSI).
- Display Interface (DSI).
- Ranura para tarjeta micro SD.
- Núcleo de gráficos VideoCore IV

En los siguientes apartados, se hará un recorrido por los diferentes modos de visualización del escritorio de Raspberry. Para ello, se comenzará desde la instalación del sistema operativo en la tarjeta micro SD hasta llegar a la visualización por escritorio remoto, pasando por visualización en pantalla de televisión, visualización mediante Raspberry Pi 7" Touchscreen Display y control remoto de la consola mediante protocolo SSH.

Todos los pasos se realizarán en un ordenador MacBook Pro. El sistema operativo donde se realice la instalación así como la visualización final no es importante pero sí cambia el modo de instalación del SO así como los procedimientos utilizados durante todos los pasos.

4.2 Instalación del sistema operativo

La instalación del sistema operativo es sencilla. En primer lugar, hay que descargar la imagen del SO de la página oficial de Raspberry Pi (https://www.raspberrypi.org/).

La página oficial indica que se puede descargar dos tipos de sistema operativo: NOOBS o Raspbian. NOOBS ofrece un sistema operativo basado en Raspbian, así como sistemas operativos alternativos para usuarios que no hayan tenido un contacto previo con este sistema operativo. Raspbian contiene únicamente su sistema operativo preinstalado y preparado con software para educación, programación, etc. Entre estos, destacan Mathematica, Python, Java y muchos más.

De los dos sistemas, se elegirá Raspbian, en concreto, Raspbian Jessie. Una vez descargado, se realizan una serie de comandos en la terminal del ordenador para instalar la imagen. El procedimiento a seguir es el siguiente.

En primer lugar, se navega hasta la carpeta donde se encuentra la imagen descomprimida.

```
cd /ruta hasta la carpeta con la imagen/
```

En segundo lugar, se pide una lista de los discos utilizados por el ordenador. El objetivo de listar los discos, es poder saber en cuál se encuentra la tarjeta SD. El comando a utilizar es el siguiente:

```
diskutil list
```

Obteniéndose el siguiente resultado:

```
MacBook-Pro-de-Sergi-2:~ sergi$ diskutil list
/dev/disk0
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:      GUID_partition_scheme                        *1.0 TB     disk0
   1:                        EFI EFI                     209.7 MB   disk0s1
   2:          Apple_CoreStorage                        848.3 GB   disk0s2
   3:                 Apple_Boot Recovery HD            650.0 MB   disk0s3
   4:       Microsoft Basic Data BOOTCAMP               151.0 GB   disk0s4
/dev/disk1
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:                  Apple_HFS Macintosh HD           *848.0 GB   disk1
                                 Logical Volume on disk0s2
                                 5D5A2BC5-8B7D-40BD-96B4-175058CC8297
                                 Unlocked Encrypted
/dev/disk2
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:     FDisk_partition_scheme                        *16.0 GB    disk2
   1:             Windows_FAT_32 boot                    62.9 MB    disk2s1
   2:                      Linux                         4.0 GB     disk2s2
MacBook-Pro-de-Sergi-2:~ sergi$
```

Figura 16. diskutil list

Como se observa, en /dev/disk2 se encuentra la tarjeta SD donde se instalará el sistema operativo y estará formateada en FAT32.

Para preparar la copia del sistema operativo, previamente hay que desmontar la tarjeta SD utilizando el número identificador de disco. Para ello:

```
diskutil unmountDisk /dev/disk<#disk de diskutil (en este caso 2)>
```

Una vez desmontada la tarjeta SD, se copian los datos de la imagen a la tarjeta SD:

```
sudo dd bs=1m if=2016-02-26-raspbian-jessie.img of=/dev/rdisk
```

4.3 Visualización del escritorio por televisión

Una vez, instalado el sistema operativo en la tarjeta, simplemente hay que insertar la tarjeta micro SD en la ranura correspondiente para que funcione.

La forma más simple para visualizar el escritorio y comprobar el correcto funcionamiento de la Raspberry Pi, es haciendo uso de del conector HDMI que posee. Una vez conectada a la televisión, sólo queda alimentarla para que funcione. La alimentación se realizará mediante un cargador de teléfono móvil de 5 Voltios y 1 Amperio de salida.

Ya solamente queda comprobar la correcta visualización del escritorio:

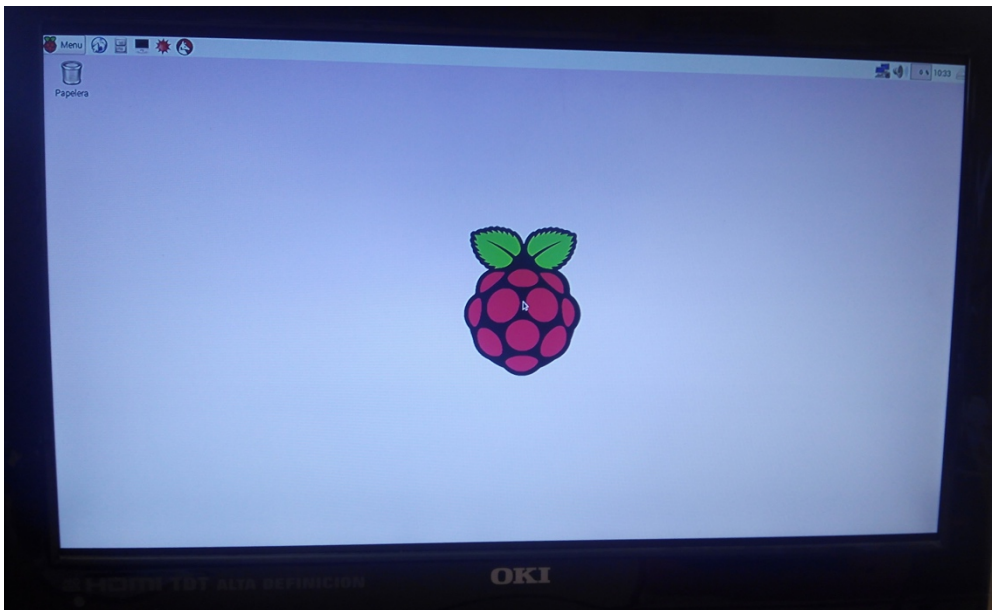

Figura 17. Visualización del escritorio por televisión

4.4 Visualización del escritorio mediante Raspberry Pi 7” Touchscreen Display

Para poder visualizar mediante el display táctil oficial de Raspberry, es necesario seguir únicamente dos pasos sencillos.

La forma de visualizar la imagen ya no se realizará mediante HDMI. En lugar de ello, se empleará el llamado Display Serial Interface (DSI). Simplemente habrá que unir mediante un cable plano flexible Jinwen E363975 AWM 20861 105C 60 V VW 1, los dos DSI.

Por otro lado hay que hacer un conexionado entre la Raspberry Pi y el display. Este conexionado se encargará de alimentar correctamente el display. El display se alimenta mediante dos pines, uno de los cuales es de 5 Voltios y el otro de GND. Los dos pines están indicados en la propia placa del display. Por el contrario, en la placa de la Raspberry Pi, hay 40 pines que no vienen indicados en la placa. Para saber a qué corresponde cada uno de ellos, se busca un plano de conexiones en la página oficial de Raspberry Pi. Este plano indica cuál es la función de cada uno de esos pines.

Para realizar una correcta alimentación al display, se tiene que conectar el pin de 5 Voltios y GND de la Raspberry Pi con sus homólogos en el 7” Touchscreen Display. Estos pines se corresponde con los pines 2 y 6 de la Raspberry Pi.

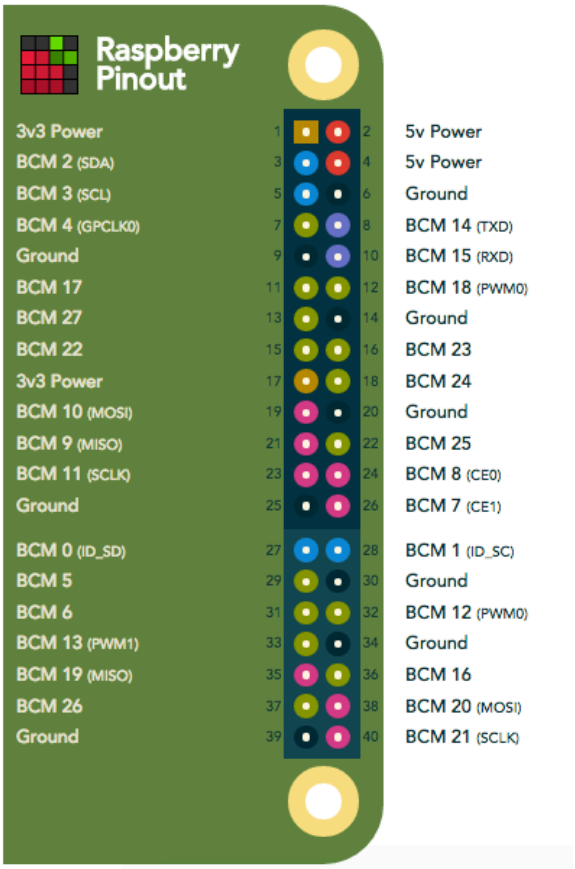

Figura 18. Pinout Raspberry Pi

Una vez realizada la conexión, solamente queda por conectar la alimentación. Como en el caso de la visualización del escritorio por televisión, la alimentación se realiza mediante el cargador de móvil citado anteriormente de 5 Voltios y 1 Amperio.

Al conectar la alimentación, se comprueba que se visualiza por el display de la Raspberry Pi su escritorio sin ningún problema.

Llegado a este punto, lo siguiente que se realizará, será controlar la Raspberry mediante el ordenador para conseguir una mayor comodidad a la hora de trabajar con ella. Esto se explicará en los dos puntos siguientes que se dedicarán al uso de comando de forma remota y el visionado del escritorio remoto.

En la figura 19 se aprecia la conexión entre la Raspberry Pi (placa superior) y el display (placa inferior).

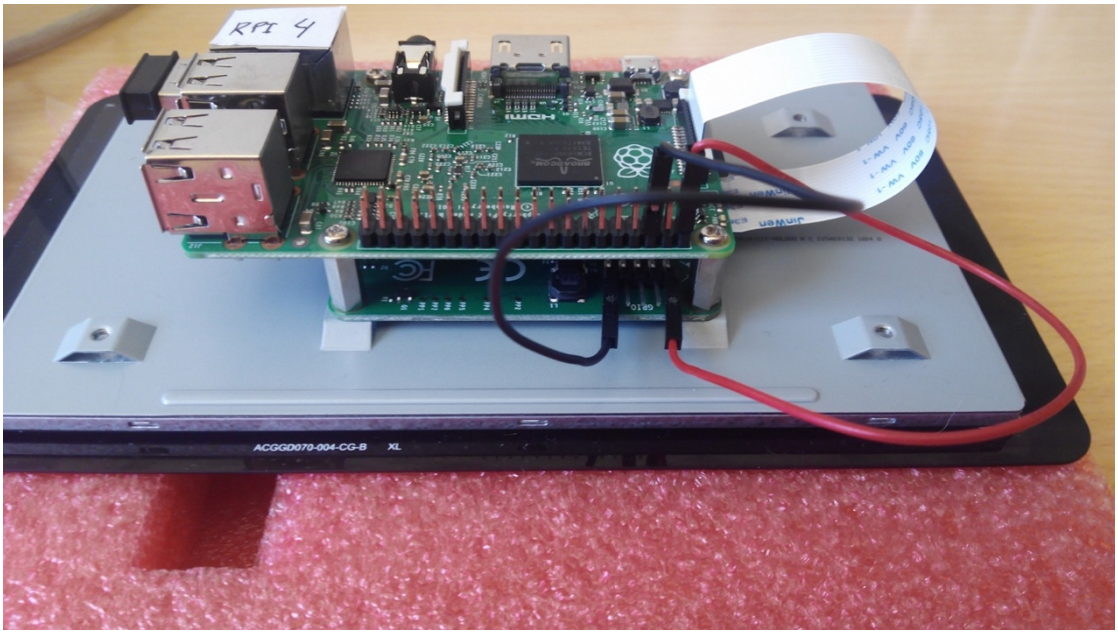

Figura 19. Conexionado 7" Touchscreen Display

4.5 Acceso a la línea de comandos de Raspberry Pi mediante SSH

El protocolo SSH (Secure Shell) sirve para acceder a máquinas de forma remota a través de una red. Mediante un intérprete de comando, se puede manejar completamente un ordenador desde cualquier otro.

Para este proyecto, se realizará una conexión mediante SSH entre la Raspberry Pi y el MacBook Pro. El objetivo final es poder trabajar con la consola de la Pi desde el ordenador para disfrutar, entre otras cosas, de un uso más cómodo.

El procedimiento consta de varios pasos.

En primer lugar, se debe averiguar la dirección IP de la Raspberry Pi. Esto es necesario, ya que para poder realizar la conexión a través del ordenador, hay que indicarle con qué otro dispositivo se desea realizar la conexión SSH; y la forma de identificar el dispositivo es mediante su dirección IP. Para averiguar, la dirección IP se introducirá el siguiente comando en la consola de la Pi:

```
hostname -I
```

Este comando, devolverá la dirección IP del dispositivo. A partir de este momento y para apreciar mejor el procedimiento a seguir, se utilizará la dirección IP 192.168.1.40.

Una vez averiguada la dirección IP de la Raspberry Pi, el siguiente paso consiste en realizar la conexión SSH entre esta y el MacBook Pro. Para ello, desde el terminal de Mac, se introduce la siguiente instrucción:

```
ssh pi@192.168.1.40
```

Al ejecutar esta instrucción, el terminal de Mac pregunta si realmente se desea continuar con la ejecución del comando ssh. Al indicar *yes*, se continúa con el proceso de conexión. Antes de conectarse a la Raspberry Pi, se indica que es necesario la introducción de la contraseña de ésta para el login entre ambas computadoras. La contraseña a introducir es **raspberry**. Una vez introducida la contraseña, ya se ha realizado correctamente la conexión SSH. A partir de este momento y hasta el momento de cerrar la conexión SSH, toda instrucción introducida en el terminal, se estará ejecutando como si se tratara del terminal de la Raspberry Pi.

A continuación se muestra una figura, en la que se conecta vía SSH a la Raspberry y se introduce el comando hostname –I. Este comando devolverá la dirección 192.168.1.40, comprobando así, que las instrucciones realmente se ejecutan sobre Raspberry.

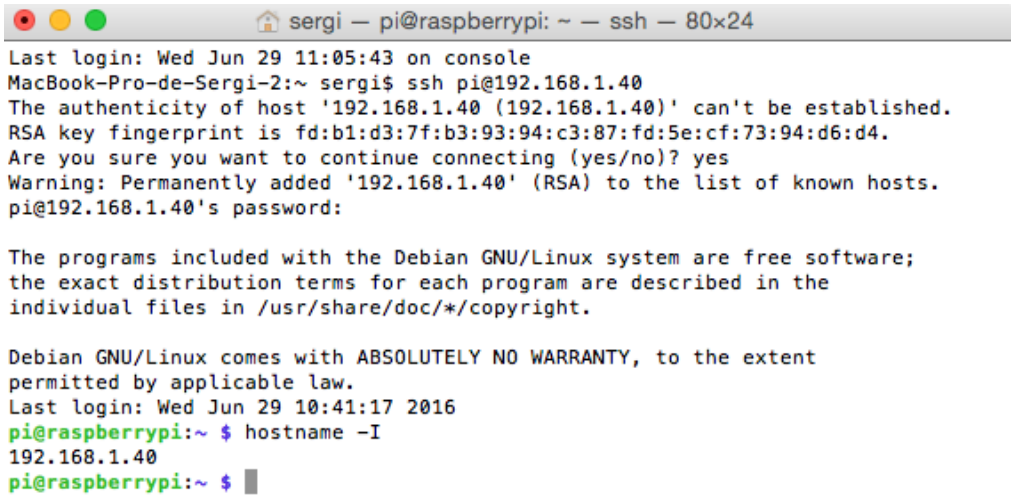

Figura 20. Apariencia terminal Mac al realizar conexión SSH con Raspberry Pi

4.6 Acceso al escritorio de Raspberry Pi mediante escritorio remoto

Hay ocasiones en las que hay que controlar la Raspberry directamente y es recomendable hacerlo mediante un escritorio remoto. Este escritorio remoto, se encarga de mostrar gráficamente el escritorio de la Raspberry e interconectar todos los eventos llevados a cabo por el ratón o el teclado.

Para este proyecto se utilizará como software de escritorio remoto VNC. Este programa mostrará en el ordenador una ventana donde se muestre el escritorio de la Raspberry.

4.6.1 Instalación y puesta en marcha del servido VNC en Raspberry Pi

Para poder realizar la conexión del escritorio remoto, hay que poner en marcha un servidor en la Raspberry. Para realizar esta tarea, se utilizará el servidor TightVNC. La instalación de este paquete consiste en ejecutar las siguientes instrucciones en el terminal de Raspbian (sistema operativo de Raspberry Pi).

```
sudo apt-get update
```

```
sudo apt-get install tightvncserver
```

Una vez instalado, hay que poner en marcha el servidor y configurar la ventana del escritorio remoto. Para ello, se ejecuta el siguiente par de comandos:

```
tightvncserver
```

```
vncserver :1 -geometry 1920x1080 -depth 24
```

El primer comando, sirve para iniciar el servidor e imponer una contraseña para la conexión remota. El segundo de los comandos sirve para configurar el display del servidor, las dimensiones y la profundidad de la ventana remota.

4.6.2 Ejecución del servidor remoto en Mac

Una vez inicializado el servidor, solamente quedará realizar la conexión al servidor VNC desde el Mac. Para realizar esta conexión con el servidor, se utilizará el programa RealVNC. Si no se tiene instalado el programa, se puede descargar de su página web oficial (realvnc.com) e instalar siguiendo las instrucciones allí indicadas.

Se ejecuta el programa y aparece una ventana como la siguiente, en la que se introducirá la dirección IP de la Raspberry Pi y el display en el que se haya inicializado el servidor.

Figura 21. Configuración de la conexión con el servidor VNC

A continuación, se mostrará una ventana advirtiendo que la conexión no está cifrada y si se desea continuar. Al pulsar en continuar, se indicará que es necesaria la introducción de la contraseña del servidor para poder iniciar la conexión. Al introducir la contraseña **raspberry**, se muestra en el Mac el escritorio remoto.

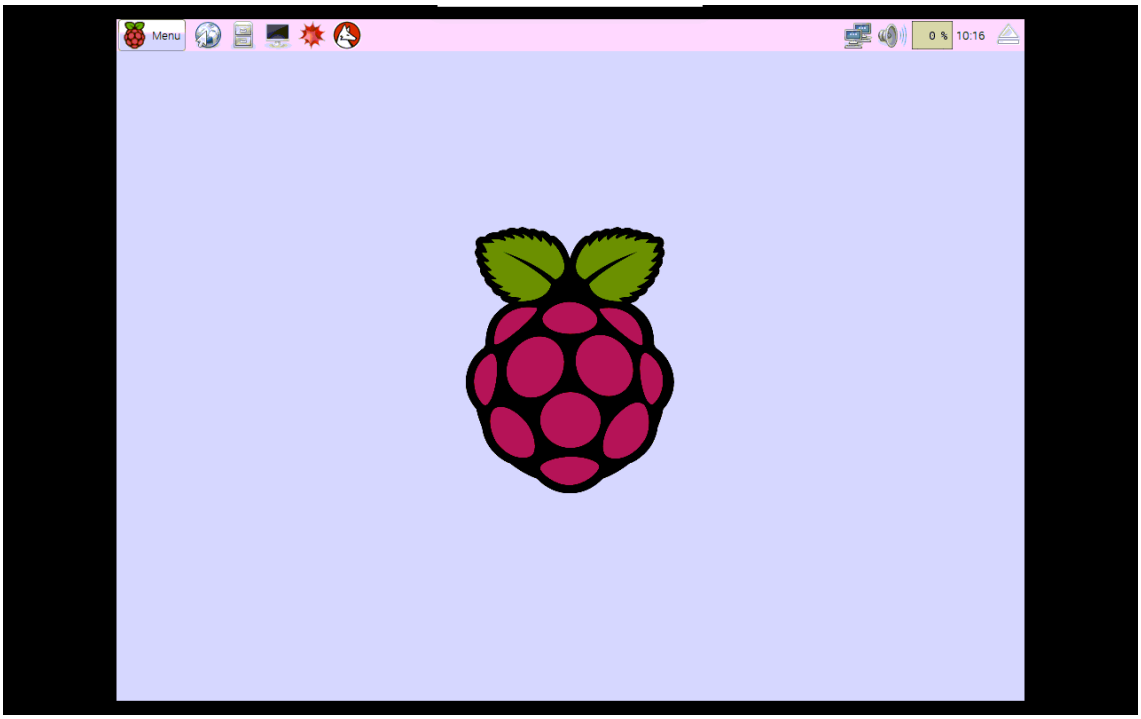

Figura 22. Vista del escritorio de la Rapsberry mediante el escritorio remoto

Capítulo 5. OpenCV

5.1 Introducción

OpenCV son la siglas de Open Computer Vision. Se trata de una biblioteca libre de visión artificial utilizada para la realización de aplicaciones en tiempo real. Existen infinidad de aplicaciones desarrolladas en OpenCV, desde aplicaciones con detección de movimiento, hasta reconocimiento de objetos. Su publicación bajo licencia BSD permite su libre uso para propósitos comerciales y de investigación.

Tiene interfaces C/C++, Python y Java para desarrollar estas aplicaciones y es compatible con Windows, Linux, MacOS, iOS y Android. Está escrito en C/C++ optimizado, lo cual le da ventajas para ser utilizado en procesamiento multi-núcleo. Contiene más de medio millar de funciones que abarcan los principales campos de la visión por computador.

El desarrollo de las aplicaciones de OpenCV se realizará en el sistema operativo Ubuntu, que trabajará dentro de una máquina virtual creada con VMware. En este sistema operativo se compilarán y ejecutarán en primer lugar todos los códigos creados para el desarrollo del proyecto. El IDE utilizado será Eclipse, en concreto el CDT para desarrollo de aplicaciones en C/C++. Debido a la gran portabilidad de este lenguaje, los mismos códigos pueden ser compilados y ejecutados correctamente en Raspberry Pi sin necesidad de ninguna instalación intermedia si se tienen instaladas correctamente las librerías necesarias para la ejecución de una aplicación concreta. Esto quiere decir que, para la compilación y ejecución de aplicaciones de OpenCV en Raspberry Pi, simplemente se necesita tener instalada la biblioteca OpenCV y utilizar el mismo código desarrollado en el sistema operativo Ubuntu. La razón de ser de trabajar en Ubuntu previamente, es la mayor comodidad y rapidez a la hora de desarrollar aplicaciones mediante el IDE Eclipse que si se hace directamente en Raspberry Pi (donde no se dispone de IDE y la compilación y ejecución se realiza desde terminal y es más compleja).

5.2 Instalación en Ubuntu

La forma de instalar OpenCV en Ubuntu es mediante un código fuente descargado de su página oficial. Para ello, se bajará el código y se realizará la instalación mediante CMake. Así pues, antes de realizar la instalación de la biblioteca, habrá que instalar paquetes previos necesarios, como el propio Cmake, para la correcta instalación de OpenCV.

La instalación de estos paquetes, se realizará en el terminal proporcionado por Ubuntu mediante el siguiente par de comandos:

```
sudo apt-get update
```

```
sudo apt-get install  build-essential cmake pkg-config libmp3lame-dev libvorbis-dev libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev libdc1394-22-dev libavcodec-dev libavformat-dev libswscale-dev libqt4-dev libjpeg-dev libtiff-dev libxml2-dev libboost-dev liblapack-dev libblas-dev libeigen3-dev
```

Una vez instalados estos paquetes, el siguiente paso es instalar la biblioteca de OpenCV. Para ello, desde la página principal de la biblioteca (http://opencv.org/), se descarga un archivo .tar para Linux/Mac. Una vez descargado, descomprimido, desde el terminal hay que situarse en la carpeta descomprimida y ejecutar las siguientes instrucciones:

```
mkdir build

cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local ..

sudo make install

cd

sudo nano ~/.bashrc
```

Dentro del archivo .bashrc, se añade la siguiente línea para que se puedan enlazar las librerías de OpenCV en ejecución:

```
export LD_LIBRARY_PATH=/usr/local/lib
```

Una vez realizado este proceso, se puede llevar a cabo la compilación y ejecución de cualquier aplicación en la que se requiera el uso de librerías de la biblioteca OpenCV sin ningún tipo de problema

5.3 Instalación en Raspberry Pi

El proceso de instalación de la biblioteca OpenCV en el sistema operativo de Raspberry Pi, Raspbian, es exactamente igual que el llevado a cabo en Ubuntu.

5.4 Compilación y ejecución

Tanto si se compila desde el terminal de Raspbian como si se hace mediante Eclipse en Ubuntu, es muy importante, indicar qué librerías y cabeceras debe incluir la compilación para que no se produzcan errores de compilación.

El paquete pkg-config instalado previamente devuelve las cabeceras y librerías necesarias para la compilación de un proyecto. El modo de funcionamiento consiste en preguntar a pkg-config las librerías o las cabeceras de una biblioteca. En este caso, para saber qué librerías y cabeceras hay que buscar, se utilizará las siguientes instrucciones:

```
pkg-config --cflags opencv
```

Esta instrucción devuelve cada una de las cabeceras necesarias precedidas del prefijo –I. Para obtener las librerías necesarias:

```
pkg-config --libs opencv
```

Esta instrucción devuelve con el prefijo –L el path donde se encuentran las librerías necesarias de OpenCV, las cuales se indican precedidas de prefijo –l. A continuación, se muestran las respuestas obtenidas al utilizar pkg-config.

```
tdi@ubuntu:~$ pkg-config --cflags opencv
-I/usr/local/include/opencv -I/usr/local/include
tdi@ubuntu:~$ pkg-config --libs opencv
-L/usr/local/lib -lopencv_shape -lopencv_stitching -lopencv_objdetect -lopencv_s
uperres -lopencv_videostab -lopencv_calib3d -lopencv_features2d -lopencv_highgui
 -lopencv_videoio -lopencv_imgcodecs -lopencv_video -lopencv_photo -lopencv_ml -
lopencv_imgproc -lopencv_flann -lopencv_core -lopencv_hal
```

Figura 23. Cabeceras y librerías necesarias de OpenCV mediante pkg-config

Capítulo 6. Teoría básica de la termografía

6.1 Introducción

Como complemento al desarrollo de la aplicación, en este apartado se ofrece un enfoque genérico de todos los conceptos y parámetros que influyen en la captura de una imagen térmica.

El principio físico sobre el que se basa la captura de imágenes térmicas es la captura de radiación infrarroja procedente los objetos con una temperatura superior a los 0 Kelvin. El espectro de esta radiación, llamada de onda larga, es completamente imperceptible para el ojo humano. Para todos los objetos con esa temperatura superior a 0 Kelvin, existe una relación directa entre la radiación de onda larga emitida por ellos y su temperatura propia. Por tanto, para el cálculo de la temperatura de un objeto se mide la radiación del mismo con la cámara termográfica y esta calcula la temperatura correspondiente a esa radiación.

La medida de la temperatura mediante el uso de una cámara térmica es lo que se denomina termografía y se realiza sin contacto directo con el objeto. Este método de medición, en el que no se produce contacto, se llama de tipo pasivo. La radiación medida por la cámara es la de la superficie del objeto. Por esta razón, la temperatura que es registrada por la cámara térmica se corresponde con la de la superficie de objeto medido.

La cámara térmica recibe tres tipos de radiación provenientes de cada uno de los objetos que se encuentran dentro del campo visual. Estos tres tipos de radiación se corresponden con las radiaciones emitida, reflejada y transmitida. En los siguientes puntos se hablará de estos factores que influyen a la hora de determinar la temperatura de un objeto. como son la emisividad (ε), la reflectividad (ρ) y la transmisividad (τ).

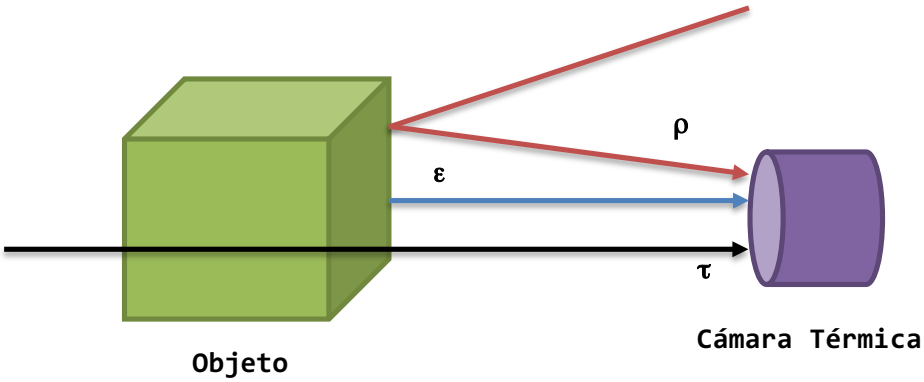

Figura 24. Incidencia de la emisividad, la reflectividad y la transmisividad en la cámara térmica

6.2 Emisividad (ε)

La emisividad (ε) se corresponde con la cantidad de radiación infrarroja emitida por un objeto. Se trata de una medida de la eficiencia con la que la superficie de un cuerpo emite energía térmica y su valor depende del material y las propiedades del objeto emisor de radiación infrarroja.

El valor de la emisividad se comprende entre 0 y 1. Un cuerpo con una emisividad igual 1, se considera un emisor perfecto. El nombre para este tipo de objetos es "cuerpo negro". Los objetos con una emisión igual a 0, se consideran un espejo térmico perfecto, puesto que solo reflejan radiación. En el mundo real, estos valores son imposibles de alcanzar y todos los cuerpos poseen una emisividad entre 0 y 1.

Se puede distinguir dos grandes bloques en cuanto a nivel de emisividad: los materiales no metálicos y los metálicos. Los materiales no metálicos se caracterizan por tener una elevada emisividad. Esta radiación emitida no se ve modificada con la variación de temperatura . Por el contrario, los materiales metálicos tienen una baja emisividad que, además, se ve modificada con la variación de temperatura.

La siguiente tabla muestra unos cuantos ejemplos de emisividad de algunos materiales.

Material	Emisividad
Cemento	0,54
Vidrio	0,92
Grava	0,28
Hielo	0,97
P.V.C.	0,91-0,93
Papel blanco	0,68
Plástico blanco	0,84
Cuarzo	0,93
Arena	0,90
Piel humana	0,98
Acero galvanizado	0,28
Nieve	0,80
Agua	0,95
Madera plana	0,90

Tabla 1. Tabla de emisividad de algunos materiales

6.3 Reflectividad (ρ)

La reflectividad (ρ) se corresponde con la cantidad de radiación infrarroja que llega a la superficie de un objeto y es reflejada por este. La reflectividad varia con las propiedades del material. No posee la misma reflectividad un mismo material si su superficie es rugosa o lisa. De igual modo, esta reflectividad puede variar con la variación de la temperatura del objeto.

6.4 Transmisividad (τ)

La cantidad de energía infrarroja que atraviesa la superficie de un objeto es lo que se denomina transmisividad. A la hora de medir la radiación por parte de la cámara, la transmisividad tiene un peso irrelevante, ya que la mayoría de objetos no dejan que se transmita la radiación a través de ellos.

6.5 Conservación de la energía de radiación

Como se ha indicado en apartados anteriores, la radiación medida por la cámara será la composición de la radiación infrarroja emitida, reflejada y transmitida por la superficie de cada objeto.

El resultado de la suma de estos tres parámetros es siempre 1, ya que se corresponde con el 100% de la radiación. Se puede, por tanto, expresar la relación entre la emisividad, la reflectividad y la transmisividad mediante la siguiente ecuación:

$$\varepsilon + \rho + \tau = 1 \qquad (6.1)$$

Como se ha indicado en el apartado anterior, la transmisividad juega un papel irrelevante en la composición de la radiación, debido a que la mayoría de objetos no dejan pasar la radiación a través de ellos. Considerando despreciable la aportación de la transmisividad, la ecuación se puede simplificar, quedando de la siguiente manera:

$$\varepsilon + \rho = 1 \qquad (6.2)$$

El significado de esta ecuación es obvio. Existe una proporcionalidad inversa entre la emisividad y la reflectividad de la superficie de un objeto. A mayor radiación infrarroja emitida, menor radiación infrarroja reflejada, y viceversa.

6.6 Consecuencia de la conservación de la energía de radiación

Debido a la influencia de la emisividad del material a la hora de tomar medidas de temperatura con la cámara térmica, se pueden esgrimir tres grupos en los que la variación de este parámetro influye directamente en la correcta medición de la temperatura.

Por un lado, se encuentran los materiales pertenecientes al grupo con un alto factor del parámetro de emisividad. En este grupo se incluyen todos los materiales cuyo factor de emisividad se encuentra entre el 80% y el 100% (teniendo en cuenta que en la práctica el valor de 100% de emisividad es imposible de alcanzar). Directamente, de la ecuación 6.2 se puede deducir que estos materiales presentan una baja reflectividad. Esta falta de reflectividad se traduce en una gran fiabilidad del dato de temperatura medido por la cámara térmica.

En un segundo grupo, se sitúan todos los materiales cuya emisividad se encuentra entre el 60% y el 80%. También, de la ecuación 6.2, se extrae que estos materiales presentan una reflectividad media de la radiación infrarroja de onda larga. Aunque en la práctica presentan mayores dificultades para la medición de la temperatura, el valor que mide la cámara térmica es igualmente fiable.

Por último, se encuentra el grupo de los materiales cuya emisividad se encuentra por debajo de 60%. Estos materiales presentan una alta reflectividad, constatable cuando se aplica la ecuación 6.2. Esta alta reflectividad produce, entre otras cosas, una reflexión de la radiación de la temperatura ambiente muy alta. Por tanto, los valores de este tipo de materiales no son muy fiables si no se ha hecho una adecuada corrección de la radiación reflejada.

Capítulo 7. Conclusión y propuesta de trabajo futuro

A pesar de haber conseguido llevar a cabo la creación de la interfaz gráfica para la captura de imágenes térmicas, este programa presenta, de entrada, cuatro limitaciones y serán los puntos de partida básicos para la mejora de la aplicación.

La primera de ellas es obvia. El programa, de momento, no muestra datos verídicos sobre información proporcionada por diferentes sensores. Por tanto, el hecho de conseguir la toma y muestra de datos en la pantalla, será el primer paso a seguir para la mejora de la interfaz gráfica desarrollada. Para llevar a cabo esta mejora, se debe elegir primero un dispositivo o varios que tomen muestras de diferentes datos mediante sensores. De entre los sensores a utilizar, se deben escoger aquellos que focalicen su información hacia la presión de oxígeno del equipo, la temperatura de las diferentes partes del traje, monitorización de frecuencia cardíaca o cualquier otra información que ofrezca un valor relevante para el agente que se adentra en un incendio para realizar su trabajo. Una vez escogidos los dispositivos a emplear, hay que elegir el sistema de comunicación con la aplicación. La mayoría de estos dispositivos se comunican con otros dispositivos de forma inalámbrica mediante el uso de Bluetooth. Así pues, para desarrollar aplicaciones en las que se comuniquen estos dispositivos con el ordenador, habrá que realizar la instalación de las librerías Bluetooth necesarias para poder establecer tanto la conexión con el dispositivo como la toma y visualización de datos por pantalla. Una buena metodología a seguir sería la implementación de un programa independiente para realizar estas funciones. Una vez se compruebe que este programa esté en completo funcionamiento, se integrará en la aplicación y se obtendrá de forma compacta la captura de imágenes térmicas en la interfaz gráfica desarrollada.

En segundo lugar, se debe aumentar la calidad de la imagen. En la imagen se aprecian unas manchas que la empeoran notablemente. Éstas son producidas por los puntos muertos del sensor de la cámara. Por tanto, otro punto a desarrollar en un futuro es la implementación de un algoritmo que los detecte y los corrija para no apreciar ningún defecto en la imagen mostrada.

El tercer punto a mejorar es la eficiencia del código. Al implementar únicamente la vista de la cámara térmica como una aplicación independiente, ésta sí tiene una eficiencia bastante buena. En este programa se ofrece un flujo continuo de imágenes térmicas, dentro de la tasa de refresco de la imagen de la Seek Thermal. Por el contrario, en la interfaz gráfica donde se incorpora el algoritmo anterior, pese a haber implementado una condición para que cada 50 ms se refresque la imagen y se muestre un nuevo fotograma de la cámara térmica, la misma documentación de la función indica que si el resto de eventos de la interfaz gráfica realizan muchas operaciones, este tiempo podría verse incrementado, empeorando de esta forma la visualización de un flujo continuo de imágenes. Este delay se produce principalmente por el hecho de que el código se realiza de forma lineal, es decir, para refrescar la imagen, primero ha de capturarse ésta. Esto hace que se produzca un tiempo que no se puede tolerar entre la visualización de un fotograma y el siguiente. La forma de mejorar esto es ejecutar ambos procesos en paralelo. De esta manera, independientemente de si ha de refrescarse la imagen o no, se capturan y almacenan los

fotogramas, que estarán preparados para ser visualizados cuando se active la condición de tiempo de 50 ms. Para poder implementar estos dos procesos paralelamente, se hará uso de la programación multihilo. Se establecerá un hilo, que se ejecutará paralelamente al resto, y que se encargará de capturar y almacenar fotogramas, para que en el momento en el que se produzca la condición de tiempo, se actualice el fotograma visualizado en la interfaz gráfica. El uso de esta técnica, permitirá visualizar un flujo de imágenes mucho más continuo en el que no se apreciarán los cortes por parte del usuario.

Por último, este proyecto no aborda la forma en la que el agente visualiza el escritorio de Raspberry Pi donde se ejecuta la aplicación. Las formas estudiadas en este documento permiten un uso más cómodo de la Raspberry Pi, pero en ningún caso constituyen un método válido de display portátil para ser utilizado por el agente dentro de un incendio. Un sistema ergonómico que se elige para el sistema de visualización portátil son una gafas inteligentes Epson Moverio BT-200. Estas gafas constan unas lentes inteligentes que contienen un display de visualización. Este display es la visualización de un escritorio Android que proviene de un dispositivo móvil conectado vía cable a las gafas. Para visualizar la aplicación a través de estas gafas, se debe desarrollar un programa que comunique vía Wi-Fi la Raspberry Pi con el móvil, que será el encargado de visualizar por el display de las gafas la interfaz gráfica desarrollada en este proyecto.

Sin embargo, la visualización mediante el dispositivo móvil intermedio puede provocar problemas para la visualización en tiempo real. Para evitar este efecto indeseado, se puede mejorar el sistema de visualización en las gafas inteligentes. Conectando directamente las gafas con la Raspberry Pi mediante el cable del dispositivo móvil, se puede mejorar la eficacia de la visualización del display. Para poder realizar esto, se utilizará la conexión MIPI-DSI de la Raspberry Pi. Para ello, será objeto de estudio toda la documentación relacionada con el protocolo MIPI, que permitirá visualizar la interfaz gráfica desarrollada a través de las gafas inteligentes.

Capítulo 8. Bibliografía

[1] Seek Thermal Inc., “Compact Series User Guide”, https://static1.squarespace.com/static/54d4e995e4b08c8065ddef4c/t/56f47897171107751283f7cf/1458862234865/Compact+Series+User+Guide.pdf [Online].

[2] Ralf Kaestner, “libseekthermal”, https://github.com/ethz-asl/libseekthermal [Online].

[3] The GNOME Project, “GTK + Platform Support”, https://developer.gnome.org/gtk3/stable/platform-support.html [Online].

[4] The GNOME Project, “Programming with gtkmm 3”, https://developer.gnome.org/gtkmm-tutorial/stable/index.html.en [Online]

[5] The Eclipse Foundation “C Development Tool FAQ”, http://wiki.eclipse.org/CDT/User/FAQ [Online]

[6] David Tulloh, “seek-thermal-documentation”, http://github.com/lod/seek-thermal-documentation/wiki/Clients#thermalview-by-jadew [Online]

[7] itseez, “OpenCV Reference Manual”, http://docs.opencv.org/master/index.html#gsc.tab=0 [Online]

[8] Raspberry Pi Foundation, “Raspberry Pi Documentation”, https://www.raspberrypi.org/documentation/ [Online]