UNIVERSIDAD
DE MÁLAGA

ESCUELA DE
INGENIERÍAS
INDUSTRIALES

ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento de Ingeniería de Sistemas y Automática

Áreas de Ingeniería de Sistemas y Automática

TRABAJO FIN DE GRADO

Software de control remoto para instrumento de espectrometría de plasmas inducidos por láser embarcable

Remote control software for onboard LIBS sensor

Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Autor: Joaquín Ortega Cortés

Tutor: Carlos Jesús Pérez del Pulgar Mancebo

Cotutor: Santiago Palanco López

UNIVERSIDAD DE MÁLAGA

Junio, 2023

Resumen:

Se ha realizado un entorno de control remoto con interfaz gráfica para un sistema de espectroscopia de plasmas inducidos por láser o LIBS. El sistema LIBS se encargará de obtener información de la composición química de los materiales que se empleen en los experimentos.

El software de control remoto realizado se encargará de obtener y procesar los espectros obtenidos por el equipo, controlar el disparo del láser y obtener imágenes de la zona de disparo, todo en paralelo y permitiendo al usuario interactuar con el equipo de diversas maneras.

El software se ha realizado en Qt Creator de Ubuntu, que es un lenguaje de programación en C++ con diversas funcionalidades propias y muy útiles a la hora de crear el entorno gráfico. El software ha sido creado en un LattePanda y su objetivo final es ser montado a bordo de un equipo móvil, como un dron, y ser manejado desde tierra mientras este realiza las mediciones. Además, el sistema controla un relé capaz de desconectar el láser en situaciones peligrosas dictadas por un sensor de distancia LIDAR.

El hardware utilizado consta de un ordenador portátil LattePanda, un espectrómetro, un telémetro LIDAR, una cámara de enfoque fijo y un relé de disparo.

Abstract:

The project consists of a software of remote control with a graphical user interface (GUI) for a system of laser-induced breakdown spectroscopy. The LIBS system will obtain information about the chemical composition of the materials used in the experiments.

This software of remote control is able to obtain and process the spectra made by the system, control the shooting of the laser and obtain images of the firing zone, all made in parallel and allowing the user to interact with the system in different ways.

This software has been made in Qt Creator of Ubuntu which is a language of programming in C++ with several original functionalities from Qt Creator useful for creating the GUI. The software has been created in a LattePanda and its final objective is to be mounted on board a drone while it is controlled from ground. Moreover, the system controls a relay able to disconnect the laser in dangerous situations dictated by a LIDAR distance sensor.

The hardware consists of a LattePanda laptop, a spectrometer, a LIDAR, a camera and a relay.

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Antecedentes

Los investigadores del Departamento de Física Aplicada I de la Facultad de Ciencias de la Universidad de Málaga utilizan una técnica de obtención de información mediante espectroscopia para la caracterización química de muestras de diversos elementos [1]. Este sistema se controla mediante software realizando el control del láser, obteniendo espectros, etc.

Hasta ahora este software estaba realizado en Windows con los componentes de este sistema desagrupados en diversos bloques que se ejecutaban independientemente; es por esto que en este proyecto se profundizará sobre la realización del software de control remoto que aúne todos sus componentes y permita al investigador la obtención de información de manera muy sencilla.

Como se ha comentado anteriormente, la idea de este trabajo de fin de grado viene de la colaboración entre las Escuelas de Industriales y la Facultad de Ciencias para la creación de un software que los investigadores de Ciencias puedan usar para realizar sus experimentos. Se trata por tanto de un proyecto muy práctico y con aplicaciones directas y muy visibles ya que este software será utilizado de ahora en adelante como parte del sistema de obtención y caracterización de muestras.

Para la realización del software era fundamental la utilización de Ubuntu como sistema operativo ya que Windows es propenso a cuelgues de sistema que, para una aplicación peligrosa como esta que usa un láser, no es el sistema operativo apropiado.

1.2. Objetivos

El objeto de este proyecto es la realización de un software de control remoto con una interfaz gráfica capaz de controlar los distintos subsistemas de un instrumento LIBS. Este instrumento se encargará de obtener información acerca de la composición química de los materiales a analizar sobre los que apunta. Los objetivos que se buscarán conseguir durante la realización del proyecto son los siguientes:

- Programación del control del espectrómetro para la obtención de espectros de las muestras.
- Programación del telémetro láser en arduino para obtener información de la distancia.
- Programación del relé en arduino para la apertura de este en situaciones de peligro.
- Programación del control de una cámara de enfoque fijo para la obtención de imágenes.
- Creación de una interfaz gráfica que permita la comunicación con el láser y mostrar los espectros y las imágenes tomadas durante el proceso por pantalla.
- Creación de un manual de usuario de la interfaz y de instalación del software de programación usado.

1.3. Metodología y herramientas empleadas

La metodología empleada para la realización del software de control remoto será el modelo en cascada. Este modelo se explica con claridad en [2] y a continuación se hablarán de las diversas fases que se seguirán:

- Requisitos del sistema: no será necesario ya que se dispone del sistema.
- Requisitos de software: se utilizarán los requisitos impuestos en los objetivos para la realización del software.
- Análisis: se buscará la mejor forma de conectar todos los componentes mediante un programa con interfaz gráfica.
- Diseño: se realizará el diseño de la interfaz gráfica.
- Implementación: se añadirán a esta interfaz gráfica todas las señales necesarias para que todo funcione correctamente.
- Prueba: se realizarán diversas pruebas con el equipo para verificar el correcto funcionamiento del software diseñado.
- Servicio: se asegurará que el software sea fácil de usar y esté bien explicado.

En cuanto a la herramienta principal utilizada para la realización del software se trata de Qt Creator. Dentro de este se han utilizado una serie de librerías y clases que permiten realizar todas las funcionalidades necesarias.

Qt Creator simplifica mucho la creación de entornos gráficos y permite al programador unir las señales que se activan al actuar sobre la interfaz con el código que lleva por detrás de manera muy sencilla. En futuros apartados se explicará con detalle como funciona Qt Creator.

1.4. Estructura de la memoria

Se explicará brevemente la estructura que tendrá la memoria:

- Introducción. Consistirá en una descripción general del trabajo, los objetivos del proyecto y las herramientas utilizadas para la realización del mismo.
- LIBS. En este capítulo se hará una descripción general del sistema LIBS, cómo se realiza el proceso de obtención de información y por qué se elige LIBS frente a otros sistemas similares.
- Instrumento. Aquí se hablará de las partes que componen el instrumento y de las formas que tiene para comunicarse con el software.
- Software de control del instrumento. En este capítulo se hablará del programa usado, de la forma en la que se han englobado todos los componentes y de cómo se han realizado cada una de las partes de la interfaz gráfica y sus respectivas funciones.
- Resultados experimentales. Aquí se mostrarán diversas muestras examinadas con el software.
- Conclusión y trabajo futuro. En este capítulo se finalizará con una conclusión general y una serie de líneas por las que se puede continuar el trabajo.
- Manual de instalación. En este anexo se presentará un breve manual para realizar la instalación del software desde cero y poder ejecutarlo en el compilador.
- Manual de usuario. En este otro anexo se describirá un conciso manual donde se explica cómo interactuar con la interfaz gráfica y la información que aparece por pantalla.
- Bibliografía. Aquí se enumerarán todas las referencias que se han ido citando a lo largo del proyecto.

Capítulo 2

LIBS

2.1. Introducción

La espectroscopia de plasma inducidos por láser o, por sus siglas en inglés, LIBS (Laser-Induced Breakdown Spectroscopy), es una técnica analítica utilizada para determinar la composición elemental de los materiales [3]. En esta técnica se usa un láser de gran potencia de pulso que se hace incidir sobre una superficie, en general sólida, para posteriormente analizar la luz emitida en los procesos que tienen lugar en el plasma generado [4].

Poco después de la invención del láser de rubí en 1960, el primer láser de estado sólido [5], se observa el primer plasma inducido por láser. No es hasta principios de los 80 cuando empieza a aparecer instrumentación basada en LIBS, creada para aplicaciones específicas de laboratorios de empresas privadas. A partir de los 90 se empieza a comercializar el sistema y aparece en el mercado. Durante el Siglo XXI se acelera la investigación en LIBS y se desarrollan modelos mucho más compactos, véase [6] para más detalles.

Las aplicaciones de LIBS son diversas, entre ellas su uso para analizar huesos, fósiles y otros restos óseos, y obtener información de la vida y el comportamiento de los seres vivos en el pasado [7], para el análisis del patrimonio arqueológico sumergido [8] o para detectar la composición química de la lava de un volcán activo [1].

2.2. Generación plasma y componentes

Los componentes del sistema LIBS se muestran en el esquema de la Figura 2.1. El sistema está formado por un láser, componentes ópticos y un espectrómetro. A continuación se describirán en detalle cada uno de los componentes.

- Láser: genera la radiación electromagnética que incide en la muestra e induce el estado de plasma. Existen una gran variedad de tipos de láseres. En este caso se usará un láser de estado sólido que es el tipo más utilizado en LIBS.
- Componentes ópticos: dirigen y enfocan el pulso láser sobre la muestra. También se encargan de concentrar y enviar la luz emitida por el plasma al espectrómetro.

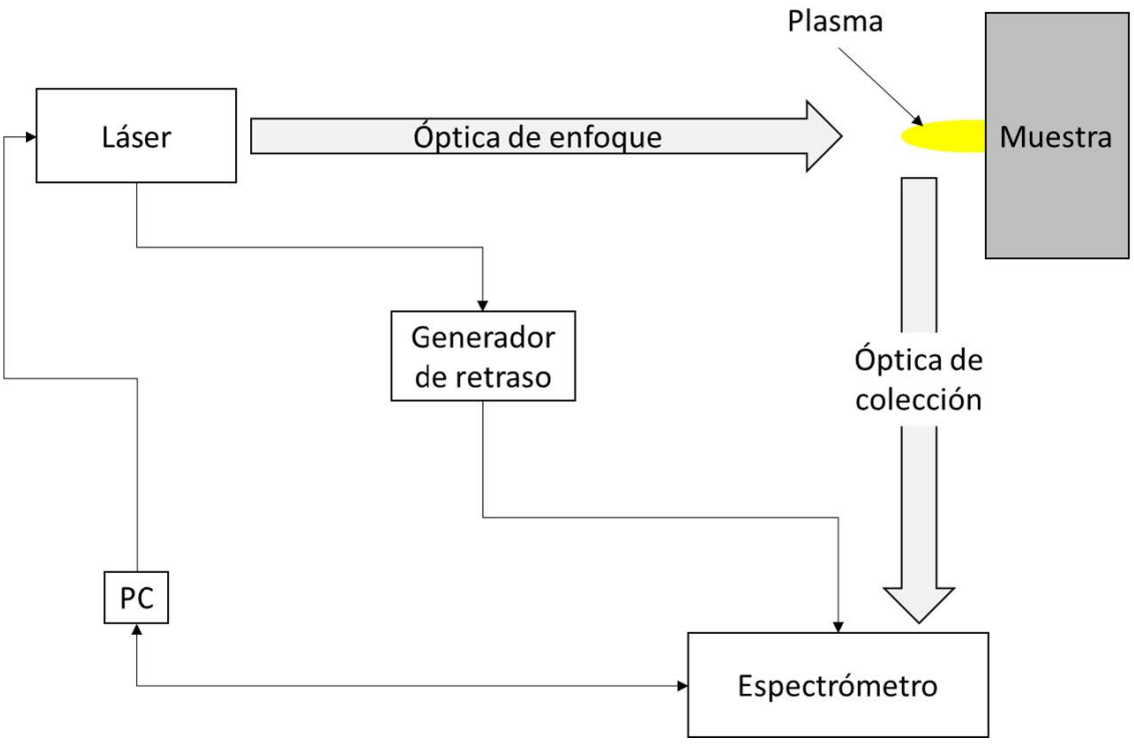


Figura 2.1: Esquema LIBS. Imagen de Alberto Bernal [9].

- Espectrómetro: formado por un espectrógrafo que selecciona las longitudes de onda, y un detector matricial (CCD o CMOS) encargado de convertir la señal luminosa recibida en señal eléctrica que interprete el ordenador. Su función principal es proporcionar información de la composición espectral de la radiación emitida por el plasma.
- PC: realiza el control del láser y el procesamiento de las muestras recibidas por el espectrómetro.

2.3. Plasma

La generación del plasma es un proceso que requiere el acoplamiento de una elevada densidad de potencia de luz láser (W cm^{-2}) sobre la superficie de la muestra para superar los calores latentes de fusión y vaporización, y que se ionice el gas resultante al evaporarse el material. Para una longitud de pulso láser determinada, la energía necesaria dependerá de tanto las propiedades físicas del láser (distribución espacial de la energía, longitud de onda...) como de las propiedades de la muestra.

En la Figura 2.2 se detallan los distintos procesos que ocurren desde que se hace incidir un pulso láser sobre una muestra hasta que desaparece por completo la actividad del plasma. En primer lugar, el material absorbe la radiación electromagnética emitida por el láser y la utiliza para superar los calores latentes de fusión y vaporización, aunque parte de ésta se disipa en forma de calor o es reflejada por el material. Una vez se genera la fase vapor, los átomos presentes interaccionan con la radiación electromagnética y, mediante procesos de absorción de un fotón e ionización térmica, los átomos alcanzan estados

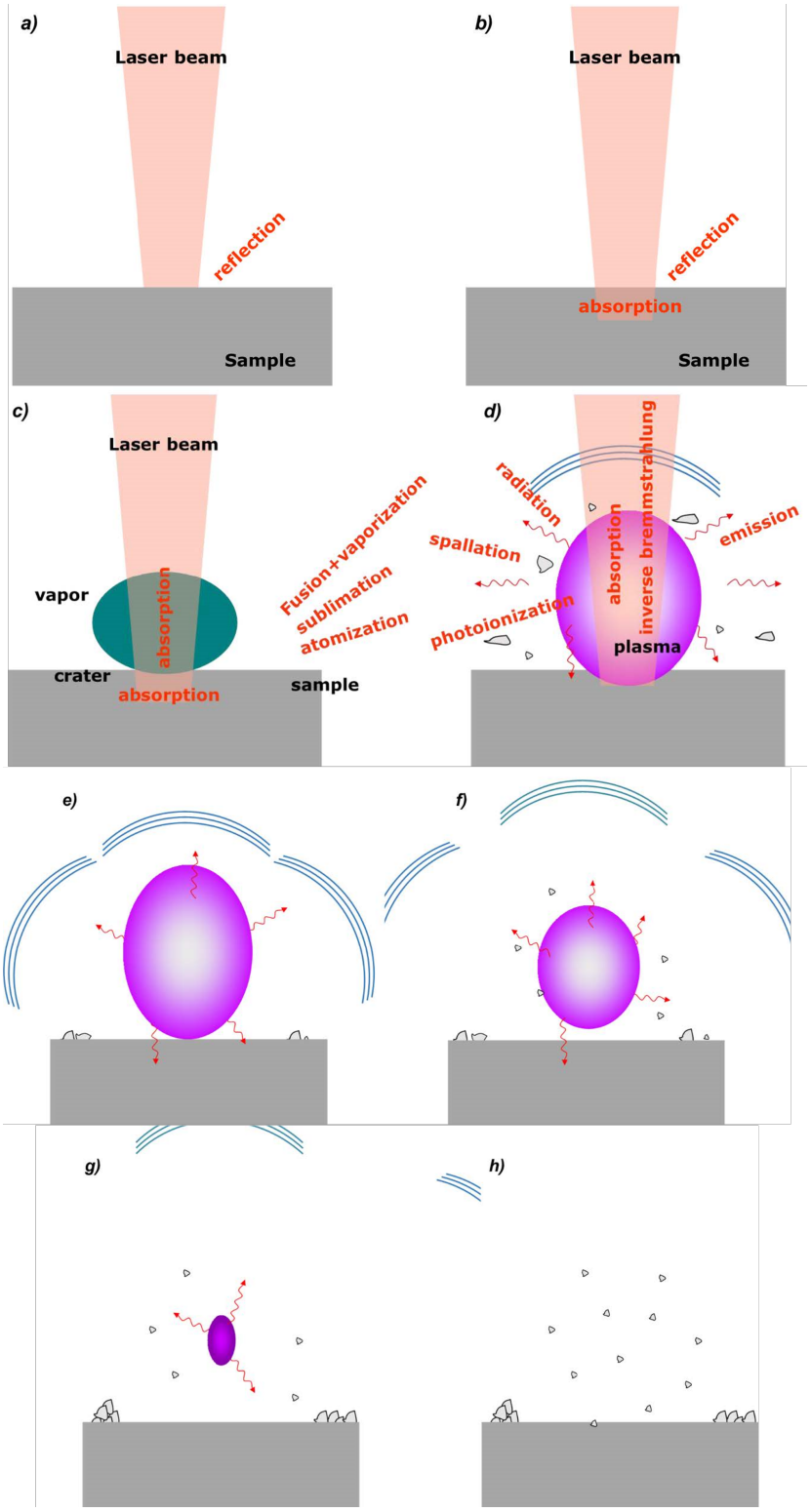


Figura 2.2: Proceso de formación y extinción de un plasma. Imagen cedida por el Profesor Palanco.

ionizados o excitados, generándose así el plasma. Este plasma inicial está débilmente ionizado, pero permite al pulso láser continuar el calentamiento local de la muestra. La temperatura en el plasma continúa aumentando, tanto por conducción térmica desde la superficie de la muestra pero principalmente a la aceleración de electrones e iones en el campo electromagnético del láser, aumentando las colisiones entre partículas y con ello la población de especies cargadas en cascada. En este proceso la densidad electrónica puede llegar a ser tan elevada que el plasma se vuelve opaco al pulso láser y permanece en este estado de plasma hasta que finaliza dicho pulso, punto en el que se empieza a enfriar lentamente, mientras emite simultáneamente radiación de distintos orígenes, siendo los más importantes la radiación de cuerpo negro, la radiación de frenado, la radiación de recombinación y la emisión atómica e iónica. Todos estos procesos mencionados están ampliamente descritos en la referencia [10].

Dada la nauraleza cuantizada de los niveles que dan lugar a la emisión de línea (atómica e iónica), la radiación emitida contiene información de las especies que la originan, por lo que su registro y estudio permiten inferir la composición química de la muestra. A esta técnica se la denomina espectroscopía de plasmas inducidos por láser (LIBS, el acrónimo inglés para *laser-induced breakdown spectroscopy*).

2.4. ¿Por qué LIBS?

Los motivos por los que se elige la técnica LIBS frente a otras similares se detallan a continuación:

- Permite observar en tiempo real los espectros obtenidos, lo que permite al usuario comprobar que los espectros son de la calidad necesaria y, si no lo son, modificar alguno de los parámetros experimentales.
- Permite obtener resultados de una forma más rápida y con una menor preparación de la muestra que otras técnicas analíticas, obteniendo resultados instantáneos cuando se automatiza el procedimiento.
- Es posible el desarrollo de configuraciones que permitan al instrumento ser embarcado en algún tipo de superficie móvil, como es el objeto de este proyecto. Esto evita al usuario someterse a situaciones peligrosas.
- El equipo LIBS se puede construir lo suficientemente portátil para su uso en campo.

Capítulo 3

Dispositivo de medición

En este capítulo se hablará en detalle sobre las partes que componen el instrumento de medición y las formas en las que el sistema LIBS se comunica con el software.

3.1. Chemocopter

El sistema LIBS utilizado, llamado Chemocopter por el investigador, está formado por diversos componentes que se citan a continuación.

- Láser Nd:YAG bombeado a diodos con un anchura de pulso de 6 ns y una energía de 25 mJ por pulso a 532 nm.
- Espectrómetro compacto Czerny-Turner cruzado f/7 con detector CMOS lineal de 4096 píxeles. Permite la obtención de espectros del plasma generado por el disparo del láser. Funciona mediante trigger en sincronización con el pulso láser.
- Ópticas de enfoque y recolección de luz.
- Cámara monocroma CMOS de enfoque fijo. Permite obtener imágenes de la zona de disparo para obtener las coordenadas espaciales del punto de análisis dónde incide el láser. Funciona mediante trigger en sincronización con el láser.
- Sensor de distancia LIDAR. Permite obtener la distancia a la que se está disparando.
- *Single-board computer* (Lattepanda Delta). Alberga en una misma placa electrónica todos los componentes básicos de un PC de bajo consumo y un microcontrolador Arduino Leonardo. Soporta la interfaz de usuario, la comunicación con los distintos componentes, así como la adquisición de espectros y su presentación en pantalla al operador.
- Relé. Permite abrir un contacto en la circuitería del láser cuando se considere una situación peligrosa mientras se esté disparando.

En la imagen 3.1 se muestra el instrumento de medición.

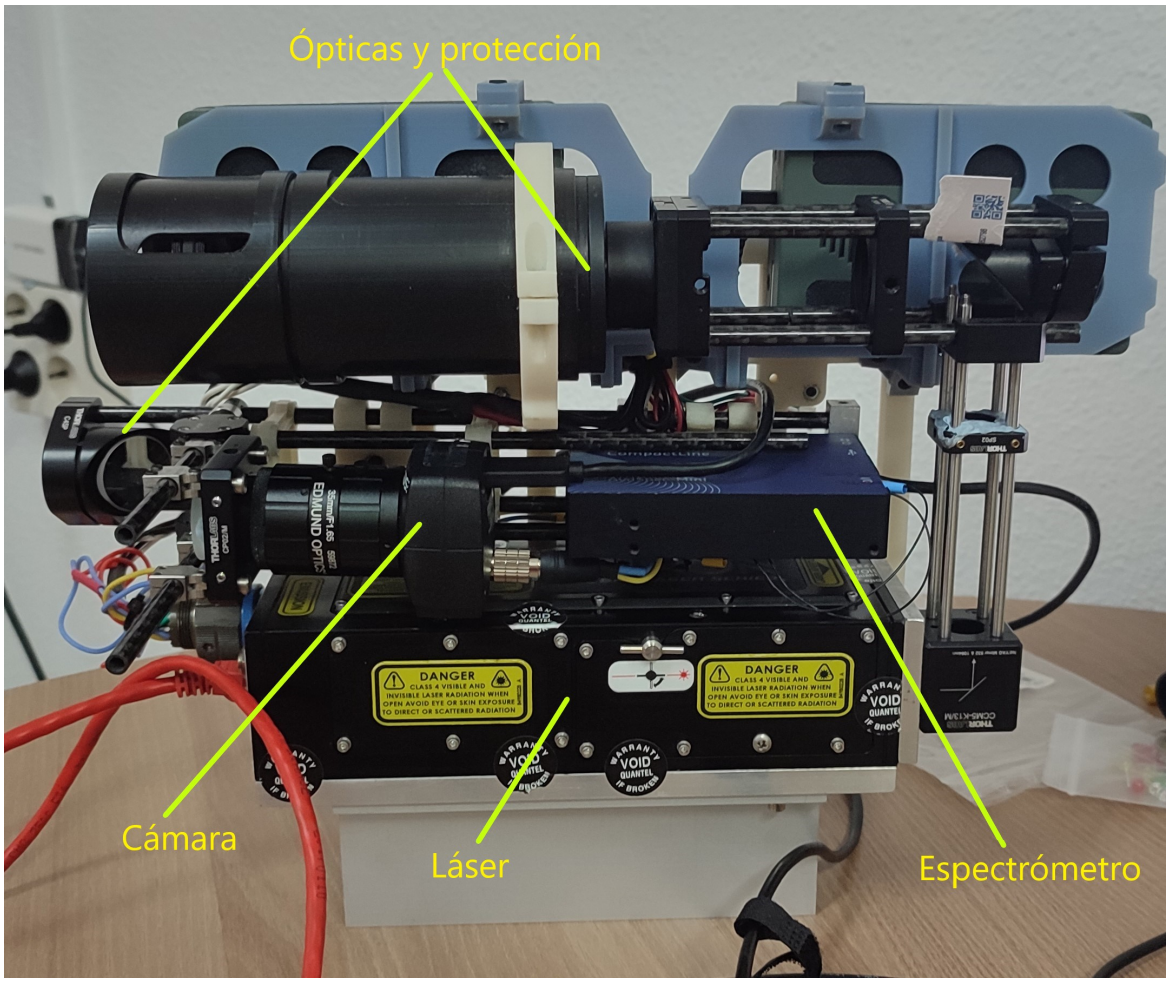


Figura 3.1: Chemocopter.

3.2. Hardware

A continuación se hablará de cómo se realiza el conexionado del hardware de cada uno de los componentes y cuáles son las formas de comunicación que emplean con el software de control remoto. En la Figura 3.2 se observan los componentes de los que se hablarán.

3.2.1. Telémetro LIDAR

El sensor de distancia se comunicará con el ordenador LattePanda mediante el arduino interno que lleva incorporado. Para esto se usarán los pines digitales que lleva el arduino y por los que se mandará la información de la distancia. El arduino que lleva incorporado el ordenador se trata de un arduino Leonardo que actuará igual que cualquier arduino conectado externamente a un ordenador.

La distancia se calcula internamente en el sensor como cualquier telémetro. Para ello, se envía un pulso electromagnético en forma de láser que se refleja en el plano medido y vuelve al instrumento de medición. A continuación, el instrumento se encarga de procesar internamente la distancia mediante el cambio de fase del disparo recibido respecto al emitido.

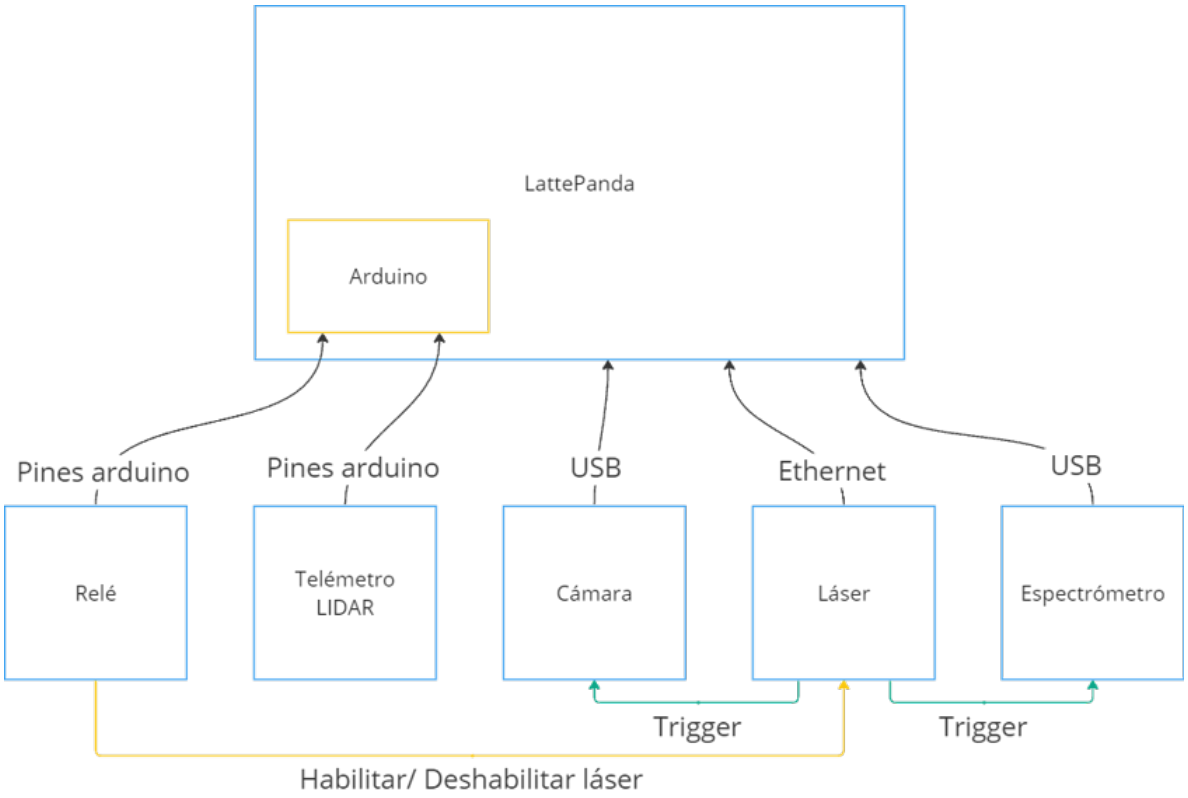


Figura 3.2: Esquema conexionado hardware.

El conexionado del telémetro en arduino se realiza como se observa en la Figura 3.3. Se recomienda el uso de un condensador de 680μF para reducir la corriente de *inrush*.

3.2.2. Relé

El relé se comunicará con el ordenador LattePanda mediante el arduino interno que lleva incorporado. Para ello, igual que con el telémetro, se utilizarán los pines digitales para comunicarse con el relé y abrirlo o cerrarlo.

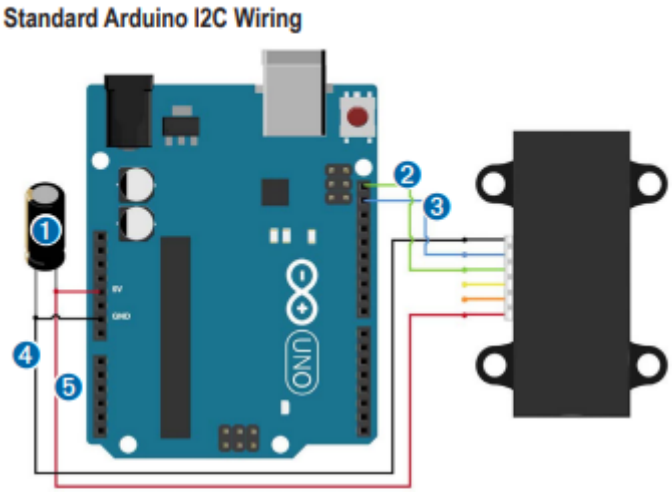
El relé se encarga de abrir o cerrar los puertos de conexión que irán conectados a la circuitería del láser y que, dependiendo de si la situación es óptima o no, dejarán al láser realizar el disparo.

El esquema de conexionado de los dispositivos conectados al arduino interno del ordenador se encuentra en la Figura 3.4.

3.2.3. Espectrómetro y Cámara

El espectrómetro y la cámara se conectan al ordenador LattePanda mediante un cable USB. El protocolo USB permite una conexión muy sencilla de *plug and play* ya que con conectar los dispositivos al ordenador ya son detectados.

Para que ambos dispositivos se encuentren sincronizados con el disparo del láser se utiliza un cable adicional de trigger externo. Los disparos del láser tienen una frecuencia de



Item	Description	Notes
1	680µF electrolytic capacitor	You must observe the correct polarity when installing the capacitor.
2	I2C SCA connection	Green wire
3	I2C SDA connection	Blue wire
4	Power ground (-) connection	Black wire
5	5 Vdc power (+) connection	Red wire The sensor operates at 4.75 through 5.5 Vdc, with a max. of 6 Vdc.

Figura 3.3: Conexionado del sensor láser. Imagen disponible en la referencia [11].

repetición de 20 Hz y, como el objetivo del sistema LIBS es la toma de espectros en sincronía con el plasma generado por el disparo del láser, se utiliza un pulso TTL de trigger externo que hace que el disparo del espectrómetro y el disparo de la cámara se sincronicen con el disparo del láser.

Cuando se ejecuta el programa de adquisición, el espectrómetro y la cámara esperarán a que les llegue el pulso TTL para tomar espectros e imágenes, respectivamente. Si se realizan 100 disparos, por ejemplo, la señal de trigger se activará 100 veces realizando 100 disparos coordinados del espectrómetro y la cámara.

3.2.4. Láser

La comunicación del láser con el ordenador LattePanda se realiza mediante un cable ethernet. Este, al contrario que el cable USB, no permite una conexión sencilla de *plug and play*, siendo necesario definir una red IP que soporta la comunicación.

El protocolo requerido por el láser para la comunicación es el protocolo telnet. Este protocolo permite acceder a otro dispositivo, en este caso el láser, y manejarlo remotamente desde el LattePanda.

Este protocolo se utiliza para mandar comandos sencillos, como realizar el disparo del láser, cambiar la configuración interna de algún parámetro o recabar información del estado del láser.

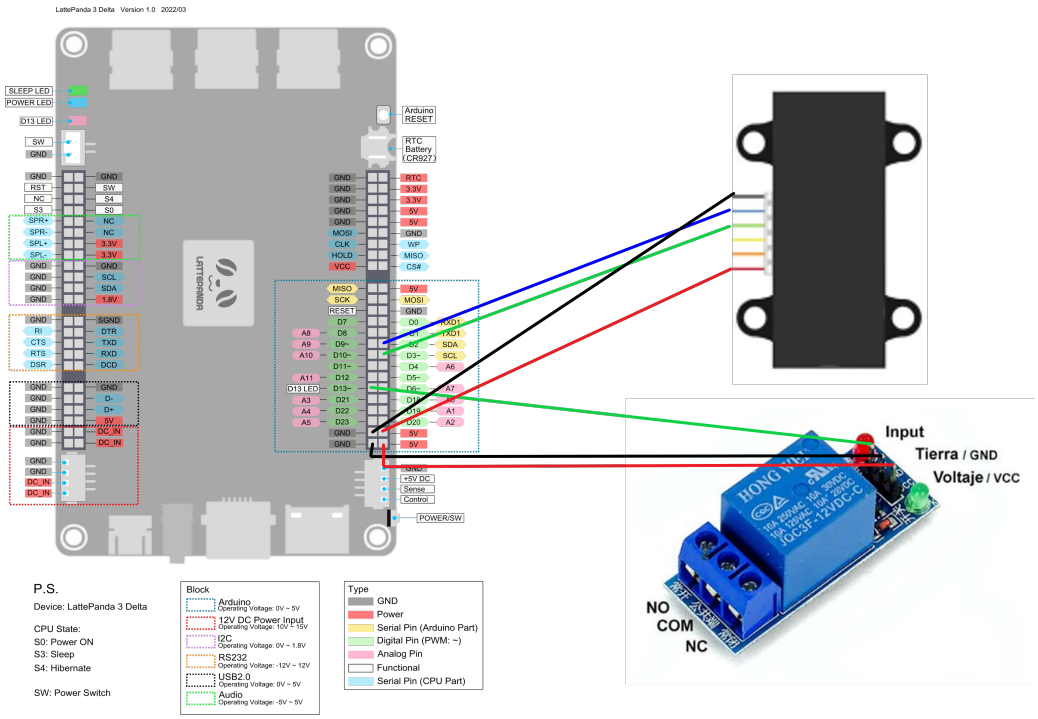


Figura 3.4: Esquema de conexionado con sensor láser y relé.

En la Figura 3.2 se encuentra un esquema simplificado del conexionado de los dispositivos.

Capítulo 4

Software de control del instrumento

4.1. Introducción

En este capítulo se hablará en detalle sobre el programa utilizado con todas las funcionalidades implementadas en la interfaz gráfica y todas las funciones creadas que hacen que todo lo que hay en la interfaz funcione correctamente.

Para la creación de la interfaz gráfica se busca conseguir una serie de requisitos. Al menos debe de tener:

- Conexión remota entre control de tierra e instrumento.
- Comunicación y control en tiempo real de los parámetros internos del láser (dosificación de energía y sensores de temperatura internos), encendido, apagado e interlock de seguridad gobernado con Arduino, capaz de desconectar el láser en situaciones de incertidumbre o peligro.
- Comunicación y control del espectrómetro y parametrización de la toma de datos.
- Comunicación y control de la cámara CMOS para captura de imágenes del plasma.
- Interfaz de usuario con botonera dividida en dos secciones: láser y adquisición.
- En la parte del láser se tendrán botones y campos de comando que configuran diversos parámetros del láser, además de testigos de información del láser y del telémetro que informen sobre las temperaturas internas del láser y las mediciones de distancia del LIDAR.
- En la parte de adquisición se dispondrán de campos de texto donde introducir los parámetros de adquisición necesarios, además de otros parámetros como el número de espectros a adquirir.
- Presentar los espectros obtenidos durante la medición en pantalla. Elegir espectros de la serie obtenida.
- Adquirir imágenes de la zona de disparo con una cámara acoplada también al sistema.

Se hablará también de todos los tipos de *widgets* usados y cómo se ha hecho para integrarlo todo en conjunto.

4.2. Qt Creator

Qt Creator es un programa multiplataforma que usa C++ y otros lenguajes y simplifica la creación de entornos gráficos. El entorno utiliza programación dirigida por eventos en la que al interaccionar con la interfaz gráfica de alguna forma, como actualizarse un campo de la interfaz gráfica o pulsar un botón, se llama a una función en la que se realiza algo concreto. También permite el uso de señales que llamen a funciones cada intervalo de tiempo. El funcionamiento se esquematiza en la Figura 4.1.

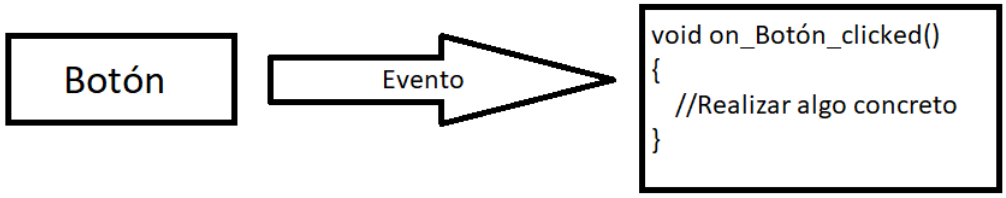


Figura 4.1: Esquema funcionamiento por eventos.

Las funciones de llamada al pulsar un botón, por ejemplo, son diversas, como *clicked* cuando se pulsa el botón instantáneamente se llama a la función, o *pressed*, que cuando se suelta el botón se llama a la función, entre otras.

El uso de Qt Creator se debe principalmente a que los ejemplos de funcionamiento del espectrómetro venían en C++ y a que este lenguaje es lo que más se ha estudiado a lo largo de la carrera. Además, el uso de Ubuntu como sistema operativo se debe a que es más robusto que otros Sistemas Operativos.

Qt Creator cuenta también con una serie de funciones propias externas a C++ muy útiles a la hora de realizar la interfaz. Estas funciones en detalle se encuentran en la página principal del programa y en la referencia [12].

4.2.1. Entorno gráfico y widgets

Qt Creator usa una forma de programar en entornos gráficos muy sencilla. Cuando se crea un programa nuevo, se crea automáticamente la clase principal con el nombre de *MainWindow*. A esta se le asocia la ventana de componentes que es con lo que interactuará el usuario.

En la ventana principal se pueden añadir, arrastrando sobre ella, diversos componentes como botones, campos de texto, *sliders*, gráficas... que irán programadas por detrás en la clase principal *MainWindow*. A esta serie de componentes se le llaman widgets y provienen de una clase propia de Qt Creator [13]. En la Figura 4.2 se observa la ventana de componentes para rellenar.

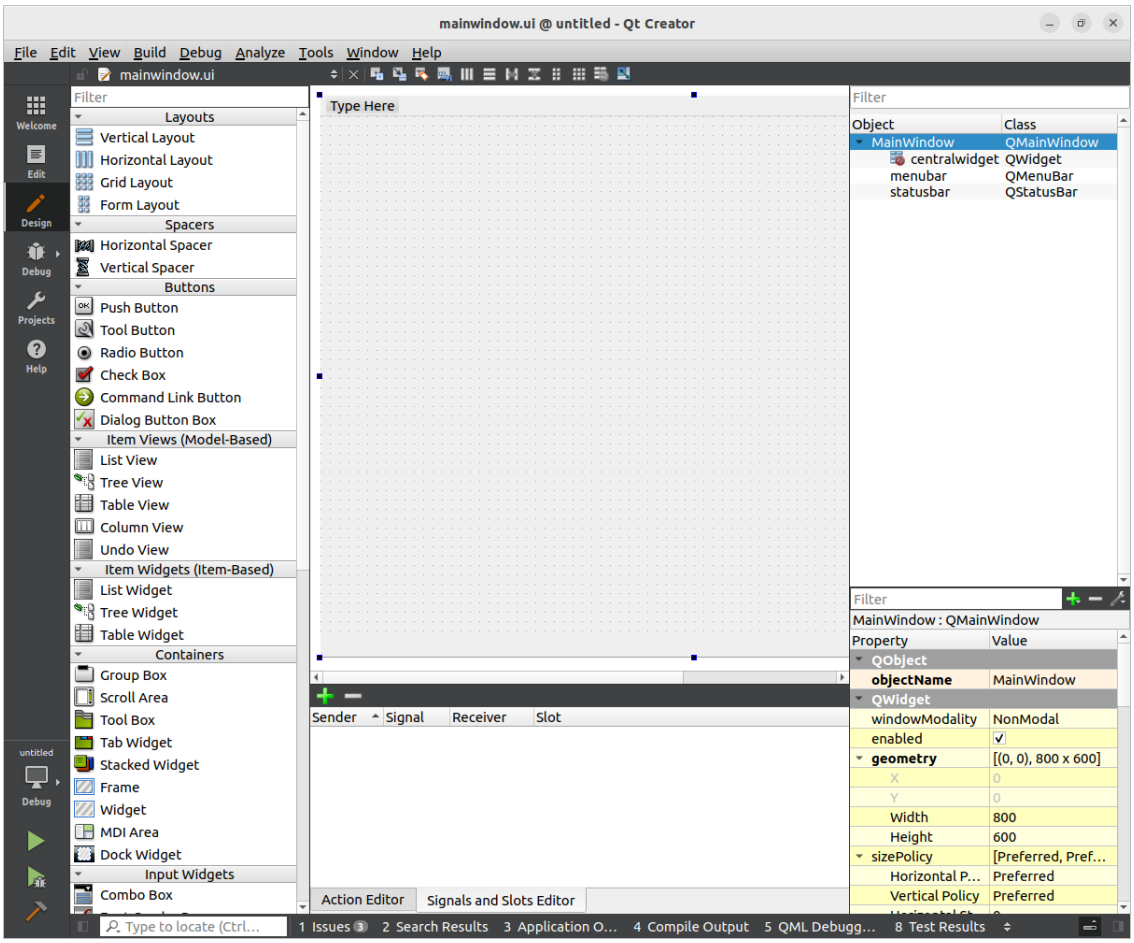


Figura 4.2: Ventana vacía.

4.2.2. Edición y depurado

Qt Creator permite el depurado del código escrito a tiempo real, sin necesidad de compilar para ver los fallos. Cuando el programa detecta algo que no entiende directamente dice que hay un fallo y, si se intenta compilar, dirá exactamente lo mismo. En la Figura 4.3 se observa un ejemplo de lo dicho.

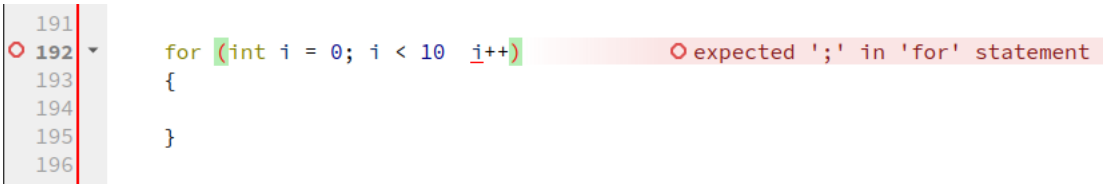


Figura 4.3: Depuración en tiempo real.

Qt Creator se caracteriza también por el auto completado de funciones o por informar de todas las posibilidades que ofrece una función, lo cual ha resultado muy útil a la hora de realizar cosas concretas necesarias para el proyecto, como se muestra en la Figura 4.4.

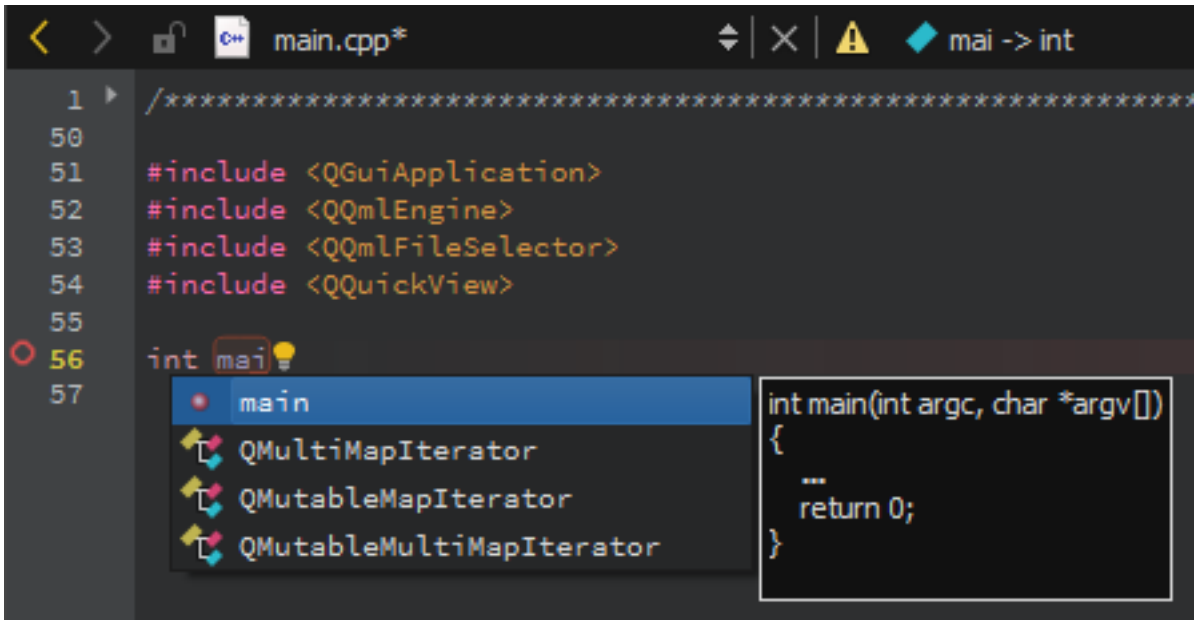


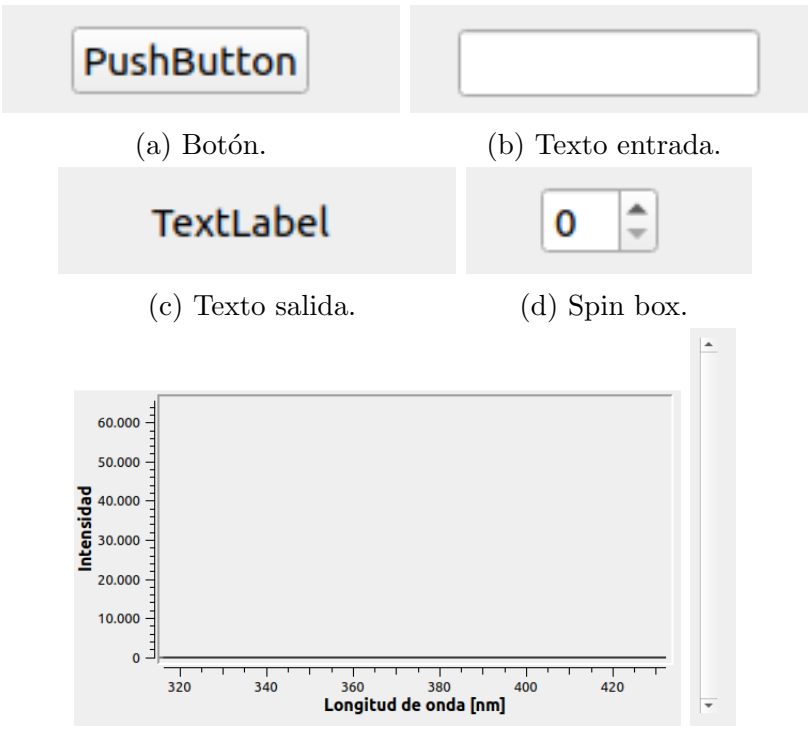
Figura 4.4: Auto completado de funciones. Imagen de la referencia [14].

4.3. Elementos utilizados en la interfaz gráfica

A continuación se enumerarán y se describirán brevemente todos los elementos utilizados en la interfaz gráfica para que, al referirnos a ellos en el futuro, se entienda perfectamente de lo que se habla:

- Botones: forman la mayor parte de la interfaz. Al pulsarse se realiza una llamada a una función que realiza algo concreto. Véase Figura 4.5a.
- Campo de texto de entrada: el usuario puede escribir valores numéricos y el programa los leerá y los utilizará para, por ejemplo, introducirlos como parámetros en alguna función. Véase Figura 4.5b.
- Campo de texto de salida: muestra en la interfaz gráfica valores numéricos que el programa envía para que el usuario los vea. Estos campos de texto pueden ser utilizados como texto de salida o como texto fijo que se puede modificar. Véase Figura 4.5c.
- Spin boxes: son campos de entrada de texto con botones tipo flecha que permiten incrementar el valor en una unidad o decrementarla. Son muy útiles porque el salto numérico que se realiza al pulsar las flechas se puede cambiar mediante funciones de la spin box. También se puede introducir un rango de valores concretos en los que la spin box funcionará. Véase Figura 4.5d.
- Gráficas: permiten mostrar gráficos de dos ejes. Véase Figura 4.5e.
- *Slider*: permiten desplazarse en un array o serie de elementos. Véase Figura 4.5f.

En la Figura 4.6 se muestra la interfaz gráfica final donde se observan todos los elementos mencionados.



(a) Botón. (b) Texto entrada. (c) Texto salida. (d) Spin box. (e) Gráfica. (f) Slider.

Figura 4.5: Elementos interfaz gráfica.

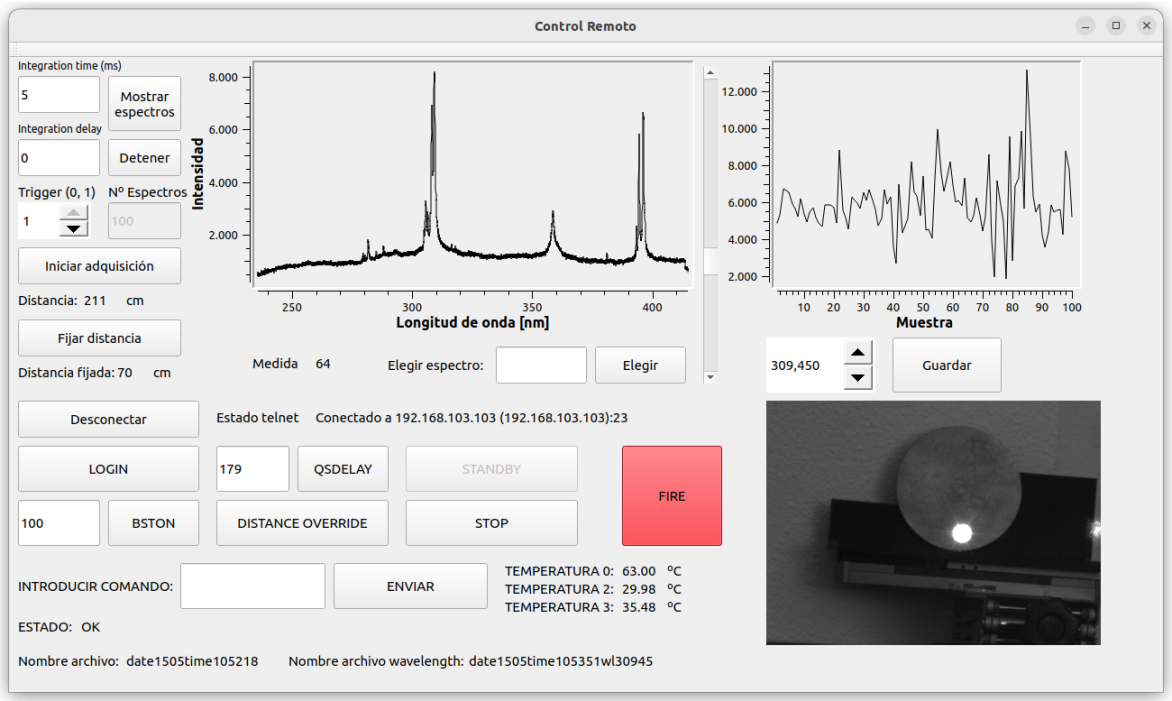


Figura 4.6: Interfaz gráfica.

4.4. Programación de los dispositivos que componen el instrumento

Ahora se hablará de cómo se ha desarrollado la interfaz y cómo se han programado cada uno de los dispositivos que componen el instrumento de medida.

4.4.1. Lectura del sensor de distancia

Una vez se realiza el conexionado del telémetro como se vio en el capítulo anterior, se busca recibir y procesar la información que este envía. Para ello se utiliza el arduino que lleva el LattePanda incorporado y que recibe la información dada por el sensor a través del puerto serie.

Junto con el esquema de conexionado de la Figura 3.3 se encuentra también un código de arduino muy sencillo que permite recibir la distancia dada por el telémetro.

El código, disponible en la referencia [15], crea un objeto de la librería del LIDAR y recibe la distancia usando la función *distance*. El código utiliza las salidas SDA (*Serial Data Line*) y SCL (*Serial Clock Line*) para dar información actual de la distancia. En la Figura 4.7 se observa el terminal serie de arduino con distancias recibidas por el sensor. Estas medidas se muestran en serie una detrás de otra a la máxima velocidad de envío de datos del sensor.

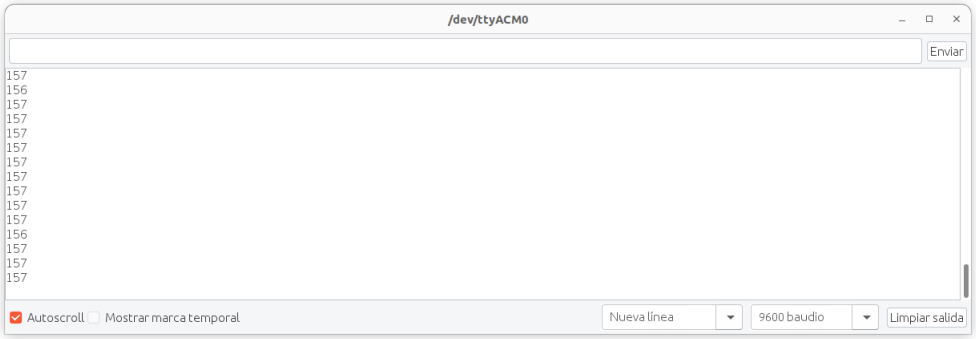


Figura 4.7: Terminal serie de arduino.

4.4.2. Lectura de espectros

Ahora se hablará de cómo se realiza la obtención de espectros mediante el espectrómetro Avantes [16] utilizando las librerías proporcionadas por el fabricante que contienen varios ejemplos de funcionamiento.

Tras enlazar los binarios compilados de las librerías con el programa que se crea, se empiezan a obtener espectros. Para mostrarlos, se obtienen los valores de las medidas de los ejes horizontales y verticales, que corresponden a longitud de onda e intensidad devuelta por el espectrómetro, respectivamente para cada eje, y se realiza, para ver su funcionamiento, un gráfico con *gnuplot* de Ubuntu (ver Figura 4.8). Se puede también

orientar el espectrómetro hacia un fluorescente para ver si se obtienen las líneas de emisión del mercurio del fluorescente. Ver Figura 4.9.

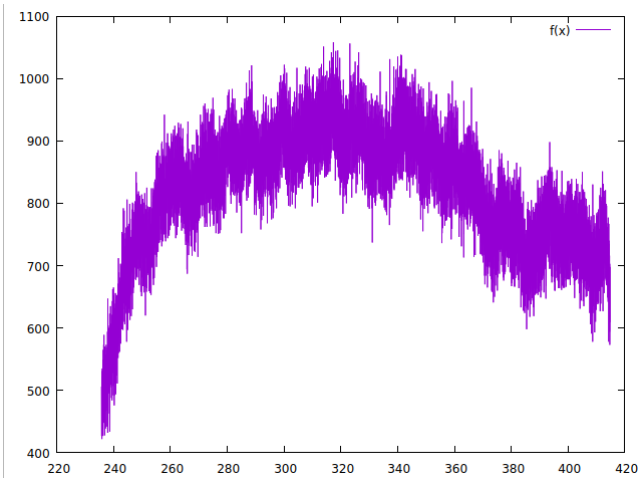


Figura 4.8: Graficado espectro.

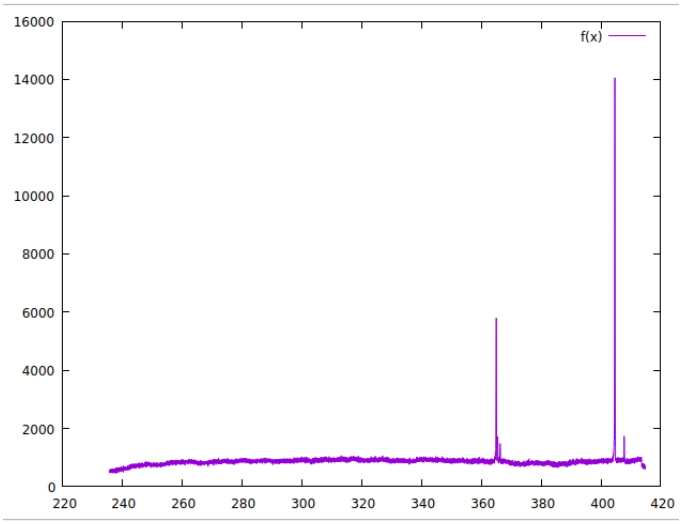


Figura 4.9: Graficado espectro fluorescente.

4.4.3. Programación en Qt Creator

Una vez que el LIDAR y el espectrómetro funcionan correctamente, se procede a realizar como tal la interfaz gráfica y a programar todo lo que esta lleva por detrás. En esta parte se siguen los diversos requerimientos que tiene que tener la interfaz que se vieron en el apartado introductorio de este capítulo. Para ello, se comienza programando lo que ya se tiene hecho en Qt Creator.

En primer lugar, se quiere mostrar en la interfaz gráfica los valores de distancia medidos al instante, para poder monitorizarlos y actuar sobre el relé en situaciones que la distancia no corresponda con el punto fijado, porque el láser se desvíe de la orientación indicada, por ejemplo.

Para ello, se utiliza la clase QSerialPort [17], que permite crear un objeto de esta clase para así abrir una comunicación con un programa de arduino previamente subido desde el mismo y actuar de diversas formas sobre este, especificando si la conexión es solo de lectura, o cambiando el *BaudRate*, por ejemplo.

Gracias a esta clase, y de forma muy sencilla, se lee todo lo que hay contenido en el puerto serie, que corresponderá con los datos que se observan en la Figura 4.7. Estos datos se transforman a *string*, para poder utilizarlo en la interfaz como texto, y se muestran por pantalla mediante un campo de texto de salida. En la Figura 4.10 se observa esto en la interfaz gráfica. El campo numérico se actualizará cada instante de tiempo mostrando la información del sensor en tiempo real.

Figura 4.10: Distancia medida por el telémetro.

Posteriormente, es necesario mostrar los espectros en la interfaz gráfica y actuar sobre varios parámetros especificados como importantes para la adquisición. Para ello se hacen uso de los ejemplos del fabricante y se reutiliza parte del código ya implementado anteriormente para que el espectrómetro haga su función.

Cuando se inicia la adquisición, el espectrómetro configura los parámetros de la misma, prepara el dispositivo para el lanzamiento y entra en un bucle que llama a la función *onDataIsHere* que iterará hasta alcanzar el número de muestras a tomar establecido.

También se busca guardar información de los espectros tomados, por lo que se genera un fichero de texto donde se copia la información de cada punto de intensidad correspondiente a cada punto de longitud de onda por el que pase, se repite esto para cada medida y se formatea para que Matlab lo interprete como que cada medida es una matriz de puntos. El nombre del archivo generado será la fecha y la hora de inicio de las mediciones, ver Figura 4.11. Se decide exportarlo directamente a un archivo de Matlab para que sea más sencillo trabajar con esos valores y no haya que hacer conversión de formatos.

Es importante mencionar la necesidad de instalación de paquetes propios de Ubuntu para que las librerías de Qt Creator funcionaran correctamente, ya que Ubuntu por defecto no las trae. Esto se dedujo tras los problemas aparecidos y después de realizar una investigación por foros de Internet. Los comandos utilizados se encuentran en el manual de instalación.

Una vez realizado todo esto se tiene la interfaz gráfica que se observa en la Figura 4.12.

4.4.4. Telnet y control del láser

El paso siguiente es la unión de lo que ya hay hecho con el control del láser. Para ello, como se dijo en capítulos anteriores, se usarán comandos Telnet que comienzan con un dólar.

El uso de Telnet en Qt Creator es también bastante sencillo; para ello se utilizará una clase llamada QTelnet que se encarga de gestionar la comunicación y que fundamentalmente usa varios comandos:

```
1    %plot(Medidax(:,1),Medidax(:,2)), x = número de la medida
2 -  Medida1=[
3    316.374 541
4    316.407 450
5    316.44  505
6    316.472 412
7    316.505 502
8    316.538 402
9    316.571 525
10   316.603 478
11   316.636 506
12   316.669 471
13   316.701 469
14   316.734 456
15   316.767 497
16   316.8   487
17   316.832 509
18   316.865 420
19   316.898 520
20   316.93  428
21   316.963 519
```

Figura 4.11: Archivo Matlab con información sobre los espectros.

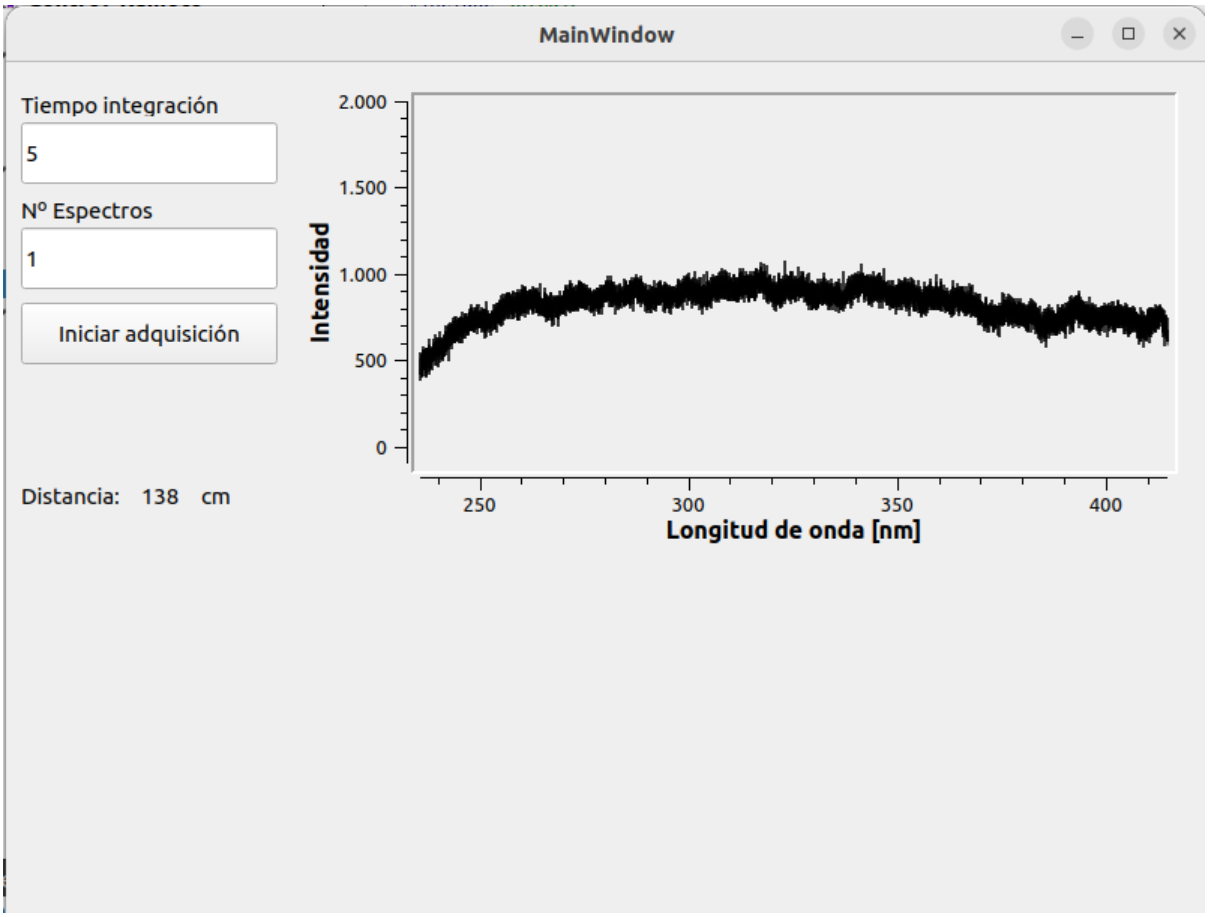


Figura 4.12: Interfaz gráfica primera versión.

- *connectToHost(const QString &host, quint16 port)*: Se realiza la conexión al host introduciendo para ello una IP y un puerto. Funciona igual que cualquier protocolo de comunicación.

- *disconnectFromHost()*: Se desconecta del host al que se esté conectado.
- *sendData(const QByteArray &ba)*: Envía bytes de información al host conectado.
- *newData(const char*,int)*: Recibe información del host conectado.

Como se observa, la comunicación es muy sencilla, para más información sobre esto, se recomienda utilizar la referencia [18].

Para la creación final de la interfaz se usarán varios botones que realizan diversos comandos, descritos posteriormente. La forma de enviar estos comandos es por Bytes, por lo que se utiliza una cadena de texto (*string*) terminada por un retorno de carro que se convertirá a Bytes y se enviarán mediante la función *sendData* vista anteriormente.

Los botones son los siguientes:

- Conectar/Desconectar Telnet: Botón dinámico que cambiará su valor dependiendo del estado en el que se encuentre; permite conectar y desconectar Telnet usando el comando de conectarse al host. Lleva asociado un campo de texto que informa de si la conexión se está realizando o ya se ha realizado.
- Login: Realiza el Login del láser, necesario para este responda a otros comandos. El comando es $LOGIN seguido de una identificación que es individual para cada láser.
- Bston: Especifica la cantidad de disparos que realizará el láser. El comando es $BSTON N, siendo N el número de pulsos del láser. Este número se especificará en una casilla de texto de entrada.
- QsDelay: Especifica el retraso del QSwitch; esto es un switch que tiene el láser mediante el cual se puede regular la potencia de salida del mismo. Los valores de entrada pueden ir entre 0 y 400, pero para valores muy pequeños es peligroso porque el pulso escapa del láser en forma inestable, por lo que se ha implementado que para valores menores de 179 el comando no se envíe. El comando es $QSDELAY N, siendo N el *delay* en μ s.
- Standby: Inicia el calentamiento del láser mediante un termostato interno. La temperatura interna del láser tiene que estar al menos a 63°C para poder disparar; una vez alcanzada esta temperatura el láser puede disparar. El comando es $STANDBY.
- Stop: Detiene el calentamiento del láser. Se utiliza cuando no se va a disparar más para que el láser se enfríe. El comando es $STOP.
- Fire: Realiza el disparo del láser usando los parámetros introducidos. En este momento es obligatorio el uso de gafas de protección por parte del técnico a cargo del dispositivo. El comando es $FIRE.
- Introducir comando: Permite introducir cualquier comando y enviarlo al láser, recordando que cualquier comando empieza por $.

Cabe destacar que los comandos del láser son muchos más, pero los implementados son los que resultan útiles a la hora de realizar los experimentos. Si se necesitara alguno más, se puede utilizar el apartado de introducir comando.

Todos estos comandos solo se mandarán si la conexión con Telnet se ha establecido, es decir, que el botón de Conectar/Desconectar Telnet nos dice que estamos conectados; en otro caso, los comandos no se mandarán para evitar fallos.

Más adelante, al probar que los comandos funcionaban correctamente y disparar el láser varias veces, se introdujo el uso de otros comandos para recibir información interna del láser. Estos comandos no se mandan de la forma enviada hasta ahora sino que se crea un thread que cada segundo llama a una función que ejecuta los siguientes comandos alternativamente:

- Temps: El láser devuelve información de las temperaturas internas, utilizadas por el usuario para saber si el láser se está sobre calentando. El comando es $TEMPS ?, termina en interrogación para que el láser entienda que tiene que devolver información. Se devuelven 3 temperaturas concatenadas en un string separadas por comas; estas temperaturas se separan haciendo uso de la clase QStringList [19]. Esta clase permite la separación de cadenas de texto introduciendo un carácter ASCII, en este caso una coma, y devuelve un array de todos los valores que ha separado. Hay que tener especial cuidado al usar esto y asegurarse de que los arrays no están vacíos porque si se acceden a ellos estando vacíos se accederán a direcciones de memoria fuera del array lo que no está permitido y el programa se cerrará. El láser devuelve un array con 4 temperaturas de las que mostramos la 0, la 2 y la 3. La temperatura 0 es la que tiene que alcanzar 63ºC para que el láser dispare, las temperaturas 2 y 3 son las que hay que monitorizar para que la temperatura interna del láser no aumente demasiado.

- Texts: El láser devuelve información relativa a todo lo que sucede en el mismo, como si está listo para el disparo o si ha recibido correctamente un comando. El comando es $TEXTS ?. En este caso mostraremos la cadena de texto entera sin separarla.

De este modo, cada dos segundos se actualizará la información de las temperaturas y de textos, respectivamente. En la Figura 4.13 se observa todo lo dicho. El botón de Standby aparece oculto porque una vez pulsado se desactiva para dar información al usuario de que se encuentra en Standby y no se desactiva hasta que se pulse Stop, que aparece como activo.

Figura 4.13: Comandos e información relativa al láser.

4.4.5. Trigger externo

A continuación se hablará de cómo se ha realizado la implementación del uso del trigger externo del espectrómetro para tomar espectros que correspondan a cada disparo del láser. Esto es necesario hacerlo ya que LIBS funciona tomando espectros al instante del plasma desprendido por el disparo del láser y tiene que estar sincronizado perfectamente para que se puedan obtener espectros correctos.

El espectrómetro tiene 3 modos de disparo:

- Disparo por software: es el modo de disparo que se venía usando hasta ahora. Cuando se inicia el lanzamiento del espectrómetro se toman todas las muestras una detrás de otra a la máxima velocidad del espectrómetro.
- Disparo por hardware: el espectrómetro toma todas las muestras establecidas una vez recibido el primer trigger.
- Disparo *single scan*: el espectrómetro toma un espectro por cada trigger recibido hasta alcanzar el número de espectros a obtener.

Se llega a la conclusión que el modo *single scan* es el apropiado para la aplicación que se tiene. Sin embargo, tras varios problemas en los que el espectrómetro no aceptaba el modo *single scan* y se cerraba la aplicación, se llega a la conclusión que el espectrómetro no lo admite o tiene un problema, por lo que se tiene que buscar una solución alternativa al problema planteado.

La solución por la que se opta es la de utilizar el trigger por hardware, pero en vez de especificarle el número total de medidas en la llamada se le especifica el disparo una vez, y ya dentro del bucle, se itera el mismo esperando un trigger tantas veces como espectros se quieran tomar. En la Figura 4.14 se esquematiza lo explicado para una mejor comprensión.

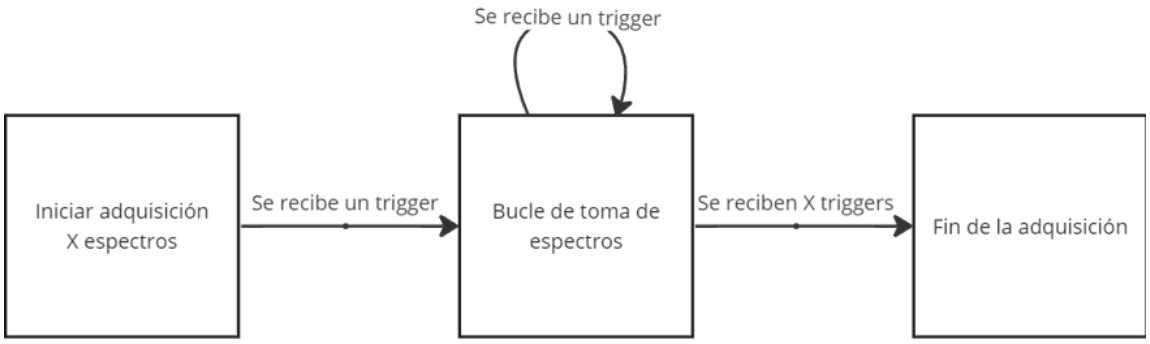


Figura 4.14: Disparo single scan mediante hardware.

Así, una vez realizadas todas las medidas correspondientes a cada trigger del láser, se saldrá del bucle principal y dejará de esperar triggers. Por tanto, la condición de salida del bucle es que los espectros tomados sean iguales a los espectros especificados en BSTON.

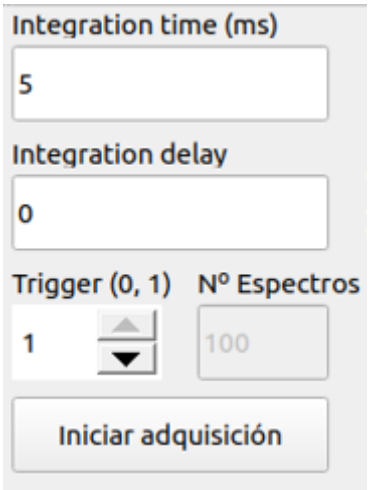


Figura 4.15: Mejora interfaz triggers.

4.4.6. Añadidos a la interfaz

En esta sección se hablará de varias mejoras de comodidad para el usuario y otra forma de representar los espectros que ayudará al investigador con el estudio de las muestras al instante.

Para ello, se hablará a continuación y en profundidad sobre cada mejora que se ha realizado.

- Se pueden modificar los modos de trigger al instante mediante un spin box que permite establecer el modo software o el modo hardware, para trigger interno o externo, correspondientes a 0 el modo software y a 1 el modo hardware; e introducir el número de espectros ya sea mediante la casilla de número de espectros o mediante la casilla de BSTON, ver Figura 4.15; se observa que la parte de introducir los espectros está deshabilitada, esto es porque, como se ha dicho antes, de ahí se toman para el trigger por software, mientras que para trigger por hardware se toma el número de espectros de los parámetros del láser (BSTON) al ser esto más lógico, ya que se tomarán tantos espectros como disparos realice el láser, ver Figura 4.13.

  Al cambiar el trigger se habilitan o deshabilitan los campos de espectros para que el usuario sepa cual se va a usar.

- Se puede elegir visualizar un espectro de entre todos los obtenidos mediante un slider o mediante una casilla de texto. Para esto se han guardado previamente todos los espectros en memoria gracias a un vector de C++ (que no un array), ver referencia [20]. Estos vectores funcionan mucho mejor que los arrays en situaciones en las que se quiere guardar gran cantidad de información y no se sabe de qué tamaño, como es el caso de los espectros porque no se sabe cuantos se van a pedir.

  Estos vectores internamente usan punteros que hacen que funcionen dinámicamente reservando memoria automáticamente y permitiendo el uso de comandos como *push back* que "empuja" el dato al final del vector, permitiendo acceder a ese dato de igual forma que un array, mediante corchetes. Gracias a esto se guarda el número de espectros pedidos asegurándose que no va a haber desbordamiento. A continuación, al acceder a esos espectros bien mediante el slider o bien mediante la elección del

espectro mediante la casilla, se dibuja en el gráfico el espectro con los valores de la posición. También se dispone de un indicador para saber qué medición se está visualizando. Ver Figura 4.16

- Se crea otra forma de representación de los espectros, muy útil a la hora de obtener información de todos los espectros que se han tomado con una sola gráfica. En este nuevo gráfico, en vez de dibujar la intensidad de los espectros uno a uno, se dibujan los valores individuales de intensidad de todos los espectros tomados correspondientes a una longitud de onda determinada. A este nuevo gráfico se llama de profundidad.

  Observando la Figura 4.17 se entiende de una forma visual lo que se ha explicado; en este caso se representa el valor de intensidad correspondiente a la longitud de onda 235,946 de las 300 mediciones hechas. Para esto se ha hecho uso de un *spin box* de valores con decimales al que se le define los valores mínimos y máximos que se le pueden introducir y el cambio o step que hay cada vez que se utilizan las flechas arriba y abajo, que corresponde a la distancia entre valores horizontales. Estos datos se han obtenido de las funciones del espectrómetro que dan información sobre las longitudes de onda en las que funciona el espectrómetro.

  Para representar los valores se utiliza un bucle para relacionar los valores de la longitud de onda disponibles, que corresponden a 4096 valores, el número de valores máximos que toma el espectrómetro, con el valor deseado en decimal; este valor introducido se resta a continuación al valor individual de cada longitud de onda correspondiente al array de los 4096 valores y el que se aproxime más a cero es el que se usará para representarlo en el gráfico.

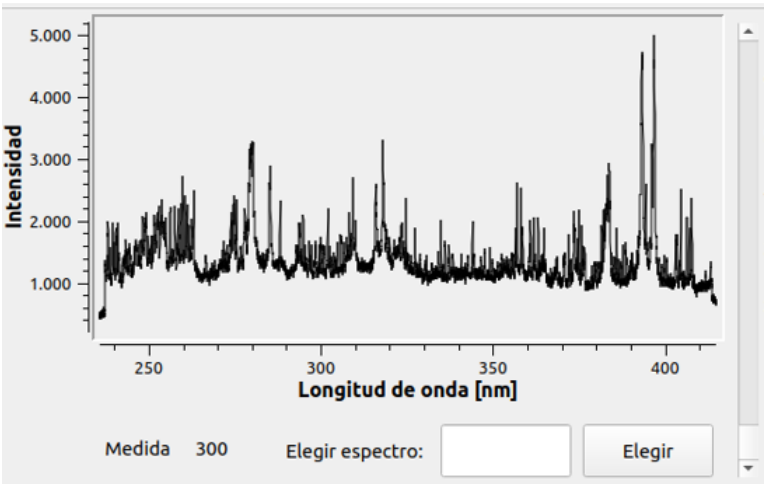


Figura 4.16: Representación de espectros.

4.4.7. Relé y control de la seguridad

A continuación se hablará de la implementación del uso de un relé para el control de la seguridad gracias a la información que da el sensor de distancia. Esto será útil en situaciones que el sensor láser no de una medida de distancia previamente establecida porque el sistema se haya desviado de la trayectoria, cuando está montado en un dron, por ejemplo.

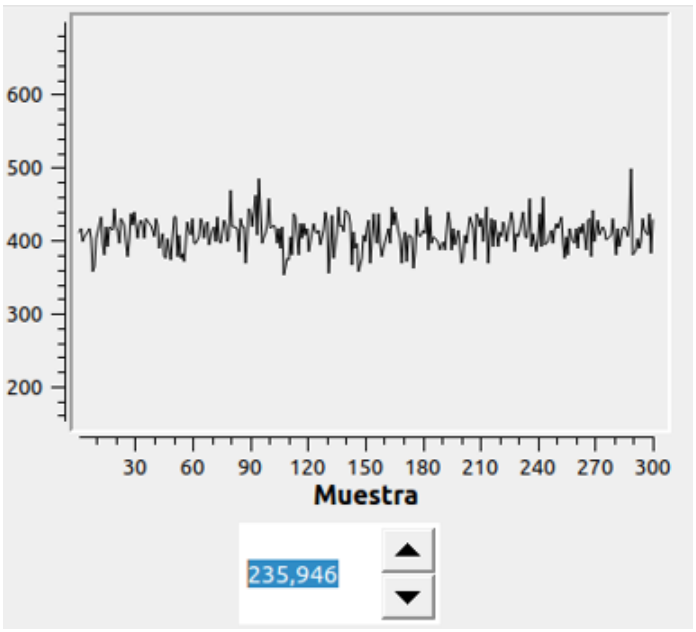


Figura 4.17: Otra forma de representación de espectros.

El láser tiene la capacidad de detener el disparo si se abre un contacto entre dos conectores que tiene en la parte trasera y que siempre estarán conectados salvo cuando el relé dictamine lo contrario.

Se actuará sobre este relé en dos situaciones de peligro:

- La distancia medida por el telémetro no es la establecida previamente; esta se fijará al apuntar a una superficie.
- La conexión entre el control remoto y el arduino que lleva incorporado el LattePanda se pierde, por ejemplo, si se da un fallo en el programa o se cierra la ventana de control remoto y el láser está disparando.

Para poder gestionar estas situaciones se hace uso del arduino y se usa para enviar por el puerto serie la distancia del telémetro. En arduino se aumenta el bucle principal para introducir un *keep alive*[1] que, mientras reciba desde el control remoto una consigna, el relé permitirá disparar el láser y, cuando deje de recibirla, el relé abrirá los contactos del láser e impedirá que dispare. Estas consignas recibidas llegarán por el puerto serie.

Esto en arduino se gestiona mediante tiempos, cada vez que llega un dato desde el control remoto se toma ese tiempo y se guarda en una variable; si dejan de llegar datos desde el control remoto, se vuelve a medir ese tiempo y se guarda en otra variable. Si la resta de ambas variables es mayor de 50 ms, para que de tiempo a confirmar que de verdad es una desconexión, se abrirán los contactos del relé hasta que se vuelva a recibir un dato desde el control remoto. Véase la Figura 4.18 para una mayor aclaración.

Cabe destacar también la gestión de la lectura o el envío de datos por el puerto serie que hace arduino; para que no haya solapamiento con los datos que se envían y se leen, se hace uso de una variable que irá conmutando y permitirá al programa de arduino la lectura y vaciado del puerto serie para saber si en este hay un dato del control remoto o el envío

[1] Forma de comunicación que se mantiene hasta que el cliente o el servidor la interrumpe.

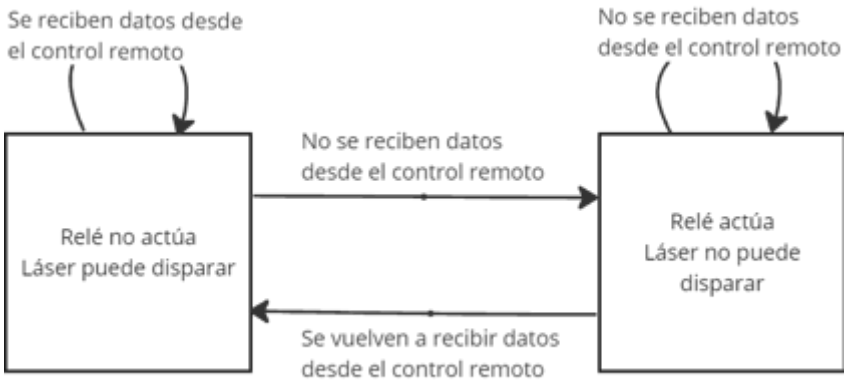


Figura 4.18: Diagrama de transición del relé.

de la lectura del sensor láser que utilizará el control remoto.

Ahora se hablará de cómo se gestiona este keep alive desde el control remoto.

Para realizar esto desde Qt Creator se usa una sola condición, que la distancia sea correcta, ya que, si se cumple esta condición y se mandan mensajes a arduino, ya se satisfacen ambas situaciones libres de peligro, la distancia es la apropiada y hay una comunicación entre el control remoto y arduino.

Para saber si la distancia es apropiada se utilizará una distancia fijada por el usuario que tendrá que ser igual a la recibida en cada instante de tiempo con una tolerancia de 20 cm. Para ello se hace uso de un nuevo botón que permite fijar la distancia que se tenga en ese momento, porque se está apuntando a la superficie de interés, y esa distancia se tomará como la distancia fijada que se tiene que cumplir para que el láser dispare, ver para ello la Figura 4.19 donde se muestra la interfaz con el nuevo botón. Previamente se había utilizado un campo de texto donde el usuario introdujera el valor de distancia, pero se optó por la nueva forma al ser esta más práctica y sencilla de usar usando el botón Fijar distancia.

Figura 4.19: Distancia actual y fijada.

Por tanto, si la distancia fijada es igual a la distancia actual ±20 cm, el control remoto envía información para que arduino permita el disparo y si la distancia fijada no es igual a la distancia actual ±20 cm, o no hay conexión entre el control remoto y el arduino, el control remoto no escribirá nada y arduino no permitirá el disparo.

También se implementa el uso de un nuevo botón que se incorpora en la sección de los comandos del láser llamado “DISTANCE OVERRIDE” que se encargará de omitir la distancia fijada para poder disparar sin que la distancia sea la apropiada, ver Figura 4.20.

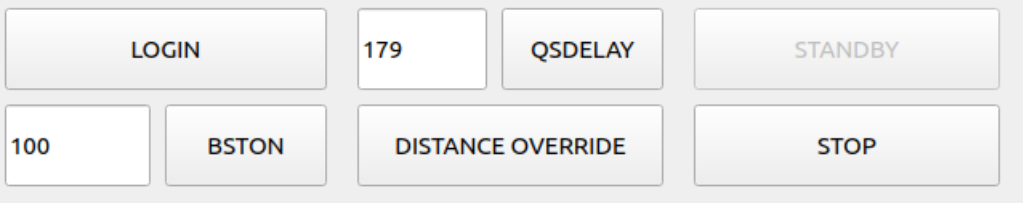


Figura 4.20: Distance Override.

4.4.8. Cámara

En esta parte se hablará de como se implementa en el código de la interfaz gráfica el otro componente hardware, la cámara. Esta funcionará también por trigger externo y tomará una imagen por cada disparo del láser, al igual que el espectrómetro.

Para hacer funcionar la cámara, se buscan las librerías del fabricante y se instala un Debian de Ubuntu con todos los binarios y librerías necesarias [21]. También se busca un ejemplo de funcionamiento básico de la cámara sin trigger.

Se hablará a continuación de algunas funciones importantes que se han utilizado para hacer funcionar la cámara:

- *is_InitCamera*: inicializa la cámara y devuelve un manejador que utilizar luego para referirnos a la cámara. Esto se utiliza para identificar una cámara cuando se usan varias cámaras a la vez.
- *is_ParameterSet*: toma los valores por defecto de la cámara, como la resolución o los bits por píxel. Estos parámetros se cargan desde la eeprom de la cámara.
- *is_AllocImageMem*: para que las librerías se encarguen de dar memoria suficiente a las imagenes que se van a obtener.
- *is_FreezeVideo*: función para esperar en cada iteración al trigger y cuando llega, toma una imagen y la guarda en memoria.

A la hora de guardar las imágenes para mostrarlas posteriormente, estas se guardan en un directorio especificado (igual que los ficheros de Matlab) con un nombre que contendrá el número de la medida; este número se usará para identificarlas posteriormente y mostrarlas a la vez que se muestran los espectros con el *slider*.

Como las imágenes nuevas tomadas se sobrescribían con las imágenes de mediciones anteriores, a veces el funcionamiento no era el adecuado porque se podían llegar a mostrar imágenes de mediciones anteriores. Para arreglar esto, cada vez que se inician las mediciones, se borran todas las imágenes guardadas y se deja la carpeta vacía para llenarla con las nuevas.

Una vez se empiezan a tomar imágenes con la cámara conectada y mostrarlas en la interfaz en tiempo real, se observa que no se toman todas las imágenes que se piden para un número de disparos del láser. Se llega a la conclusión que esto es debido a que la resolución de las imágenes es demasiado grande y el sistema no es capaz de procesarlas cada 50 ms (o a 20 Hz). La resolución de las imágenes por defecto es de 1280×720 y cada una pesa entorno a 4 Mbs. Se piensa en reducirlas a 640×480 y ver si así es capaz de procesarlas.

Una vez realizada la compresión de las imágenes, se consigue que el sistema sea capaz de

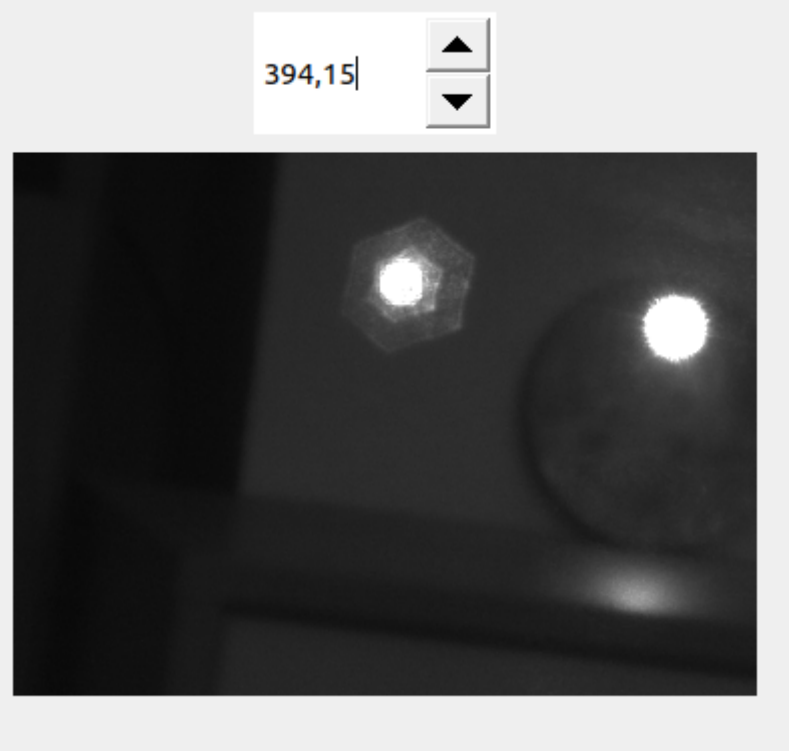


Figura 4.21: Imagen tomada con la cámara de enfoque fijo, imagen no enfocada (primeras pruebas con el sistema).

procesarlas durante la ejecución pero sin mostrarlas en directo, mostrándolas solamente al final de la ejecución mediante el slider. Estas imágenes se colocarán en la interfaz gráfica mediante un campo de texto de tamaño aceptable y se establece que en vez de texto, ahí salgan las imágenes en formato .bmp. En la Figura 4.21 se observa una imagen tomada con el sistema y mostrada en la esquina inferior derecha de la aplicación.

Posteriormente, al unir la ejecución de la cámara con la ejecución del espectrómetro, el programa no es capaz de gestionar ambas cosas a la vez ya que se toman la mitad de espectros y la mitad de imágenes, esto es debido a que se realizan ambas cosas en serie en vez de en paralelo.

La solución más lógica es realizar todo en paralelo mediante el uso de *threads* o hebras que inicien la cámara y el espectrómetro por separado y cada uno espere sus triggers de manera independiente. Para ello se busca la clase de Qt Creator que gestiona el uso de *threads* a la vez que se busca información de cómo usarla. La clase en cuestión se llama QThread y permite el uso de hebras de una forma muy sencilla [22].

El funcionamiento de la clase es el siguiente: se crea una nueva clase que hereda de QThread todos sus comportamientos y se crea una función llamada *run* dentro de la clase que ejecute exactamente lo mismo que se ejecutaba en serie para la cámara pero con un bucle *for*, ya que al estar dentro de un *thread* no bloquea el programa. A continuación, en el programa principal (MainWindow) se crea un objeto de la clase creada previamente y este se inicializa y se le asocian los parámetros necesarios cuando se inicia el espectrómetro, para que esté esperando a que lleguen los triggers a la vez que el otro componente. Ver esquema de funcionamiento en la Figura 4.22.

Con esto se consigue que todo se sincronice y funcione correctamente, obteniendo una

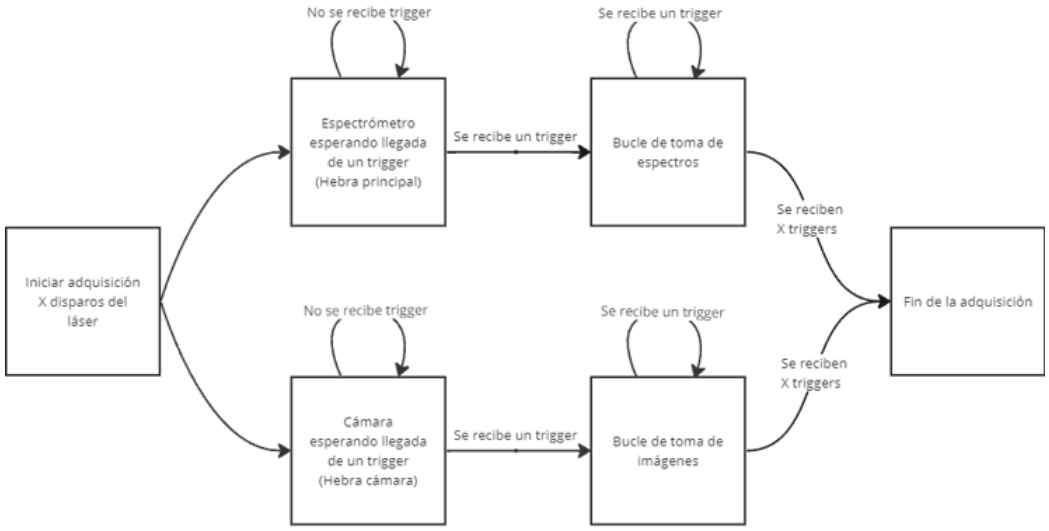


Figura 4.22: Esquema de transición de los componentes de adquisición.

imagen y un espectro por cada disparo del láser.

4.4.9. Mejoras finales

Ahora se hablará de varias mejoras necesarias y útiles a la hora de realizar la adquisición y algún que otro cambio que se tiene que realizar para que todo funcione correctamente.

- A la hora de iniciar la adquisición, los espectros se muestran en la gráfica solamente al terminar la ejecución porque se observa que esto ralentiza el sistema en algunas mediciones, haciendo que no se lleguen a tomar todas las medidas; entonces, al igual que con la cámara, una vez que se ha terminado el proceso de medición, se puede elegir un espectro y una imagen que visualizar mediante el *slider* o introduciendo el valor numérico de la medida.

  Esto es, por lo general, lo que se quiere utilizar, pero en casos muy concretos, como al poner a punto el espectrómetro mediante un proceso de ajuste de la óptica realizado por el investigador, se quieren ver los espectros mientras se realizan las adquisiciones, para ver si realmente se está recibiendo información o si, por el contrario, es todo ruido; recordando los primeros espectros tomados, Figuras 4.8 y 4.9, el primero es el ruido que devuelve el espectrómetro aunque no entre luz del plasma y el segundo es el espectro de emisión del mercurio presente en un tubo fluorescente.

  Para que la transición sea cómoda y no haya que comentar y des comentar código, se introduce un botón que realiza esta función llamado *Mostrar espectros*. Este botón que se introduce tiene un nuevo funcionamiento: actuar como un botón que se queda pulsado, o sin pulsar, teniendo dos estados que permiten saber muy bien si está actuando o no dicho botón, aunque realmente se observa esto también mirando el gráfico.

- Al realizar varios experimentos con el software y tomar espectros de diversos materiales, se observa que el sistema a veces no realiza todas las medidas que se le piden, normalmente una medida menos de lo pedido, o si son gran cantidad de

disparos, incluso dos o tres. Se reconoce este fallo como la pérdida de triggers del láser, problema eléctrico que raramente ocurre y que hace que un trigger en cualquier momento de la serie se pierda y no le llegue a la cámara y al espectrómetro, no tomándose en ese caso las medidas correspondientes.

Esto introduce un gran problema en el bucle de toma de espectros e imágenes porque recordemos que ambos bucles terminarán y permitirán su comienzo de nuevo únicamente cuando todos los disparos y sus respectivos triggers hayan ocurrido, recordemos Figura 4.22.

Entonces se piensa en una serie de soluciones, como introducir un contador de tiempos que cuando supere un umbral ignore el trigger, o forzar la detención con un demonio de Ubuntu que se encargue de cerrar la ejecución cuando se quede esperando triggers mucho tiempo. La solución por la que se opta es la que se considera más sencilla y menos propensa a fallos, que es la de introducir un nuevo botón de detención de la adquisición que fuerce la salida de los bucles usando funciones que ya vienen preparadas para ello.

Así, al presionar este nuevo botón, se llamarán a funciones como "*StopMeasure*" que se encargan de detener la espera de triggers y salir del bucle principal, permitiendo así obtener nuevas mediciones. Este nuevo botón también permite detener la adquisición en cualquier momento de cualquier medida o al comienzo de esta cuando se espera el primer trigger para introducir nuevos parámetros porque alguno era erróneo, por ejemplo. En la Figura 4.23 se observa un nuevo esquema de transición donde se tiene en cuenta la nueva problemática.

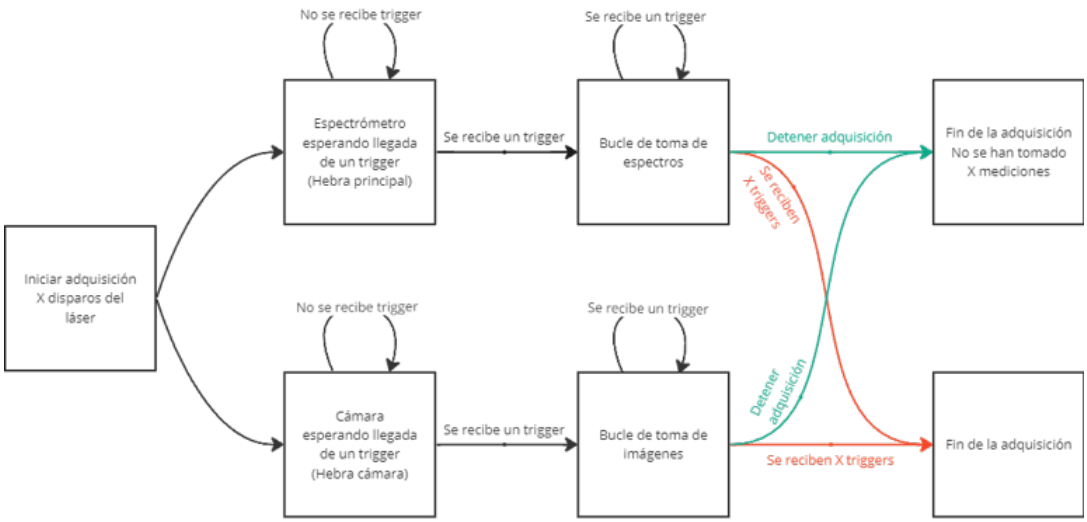


Figura 4.23: Esquema de transición con finalización mediante botón.

También es importante destacar que el usuario sabrá cuando faltan triggers porque el botón de iniciar adquisición estará deshabilitado y solamente se habilitará cuando finalice la adquisición normal por cualquiera de las formas.

- Cuando se empiezan a adquirir espectros con valores con mucha intensidad, se piensa interesante implementar un *autoscale* del gráfico para que se ajuste automáticamente a los valores máximos y mínimos de los valores a representar. Hasta ahora, los valores

eran establecidos por defecto ya que no se habían obtenido espectros de materiales, solamente ruido.

Para ello, en los bucles que dibujan los espectros en la interfaz gráfica cuando se mueve el *slider*, se introducen dos variables, máximo y mínimo que irán actualizando su valor conforme encuentren valores mayores o menores, respectivamente. Una vez se termine el bucle, se ajustará la escala con los valores que se hayan alcanzado en máximo y mínimo, con un poco de tolerancia por arriba y por abajo para que se vea bien. Con esto se consigue que para cada medida se ajuste automáticamente la escala, independientemente de los valores que la medida contenga.

También se realiza algo similar para el bucle que dibuja los espectros al instante si el botón de *Mostrar espectros* está pulsado.

- Los investigadores también sugieren como muy interesante a la hora de tomar muestras y verificarlas el guardado del gráfico de profundidad (Figura 4.17). La idea es la de establecer un valor de longitud de onda para visualizarlo en el gráfico y, si interesa, poder guardarlo igual que se hace con los espectros normales para poder utilizarlo posteriormente.

  Para ello se introduce un nuevo botón de *Guardar* que, igual que se hace con los espectros, se guardan las mediciones en un archivo de Matlab. En este caso constará de un solo vector que irá desde el 1 hasta el número de espectros tomados con un valor de intensidad para cada valor, ver Figura 4.24.

  El nombre del archivo será la fecha y hora actuales seguido de *wf* y el valor de la longitud de onda sin punto ni coma decimal para que Matlab no de problemas al cargarlo directamente. Para esto último se toma el valor que se haya introducido de longitud de onda, se separa por coma (si tiene valor decimal) y se concatenan los valores para tener el *string* de salida. Si no tiene coma se concatena directamente.

- Se muestran en la interfaz gráfica los nombres de los archivos para asociarlos con las anotaciones que hagan en papel. Para ello se toman los *strings* de salida de ambos ficheros y se muestran en dos campos de texto en la parte inferior de la interfaz gráfica. Estos valores se actualizarán cada vez que se haga una adquisición o se guarden valores con el campo de *Guardar*.

- Se introduce otra forma de guardar la información de los espectros, aparte de la ya comentada por si los investigadores no tuvieran suficientes conocimientos de Matlab. Para ello, se añade la salida de los ficheros en formato csv y se organiza el fichero principal de espectros en columnas, teniendo en la primera columna los valores de la longitud de onda y en las siguientes los valores de intensidad de cada medición correspondiente a cada longitud de onda.

  Esto se realiza incluyendo un bucle que itera el vector de espectros una vez ha acabado la ejecución y escribiendo ahora en un csv los valores como se han comentado anteriormente. El resultado, ahora en formato csv, se observa en la Figura 4.25.

date1505time101816wl235946.m

```
%Ejecutar con run para guardar la longitud de onda, graficar
%con plot(Wavelengthx(:,1),Wavelengthx(:,2))
%donde x es el valor de la longitud de onda
Wavelength235946=[
1   422
2   381
3   426
4   404
5   401
6   418
7   397
8   356
9   401
10  432
11  412
12  390
13  409
14  413
15  400
16  430
17  406
18  376
19  384
20  403
21  397
22  399
23  405
24  393
```

Figura 4.24: Fichero de Matlab.

4.5. Conexión de escritorio remoto

El objetivo final del software es, como se ha dicho anteriormente, su implantación en un sistema móvil; por esto se necesita de una comunicación con el controlador abordo que el investigador usará desde tierra. Esto se realizará creando una red WIFI privada desde el LattePanda que, al conectarse desde el control de tierra, permitirá utilizar el software mediante el uso de escritorio remoto.

1	Columna1	Columna2	Columna3	Columna4	Columna5	Columna6	Columna7	Columna8	Columna9	Columna10
2	235.796	480	517	523	442	466	491	486	478	458
3	235.844	429	371	377	402	385	443	421	379	405
4	235.892	560	509	514	517	542	498	530	490	528
5	235.94	415	389	395	411	408	399	393	377	411
6	235.988	534	562	526	554	531	534	572	557	580
7	236.036	430	438	435	431	439	443	423	443	429
8	236.084	528	545	466	509	528	517	556	495	526
9	236.132	478	442	463	450	449	451	443	437	454
10	236.18	523	502	475	503	538	511	468	522	525
11	236.228	431	401	407	373	395	401	410	412	376
12	236.276	561	521	541	557	535	523	541	533	536
13	236.325	394	408	417	436	462	382	434	421	425
14	236.373	561	521	525	519	534	531	511	536	545
15	236.421	423	425	431	413	399	443	444	441	436
16	236.469	549	540	553	568	525	514	561	552	548
17	236.517	458	432	478	458	477	487	454	452	438
18	236.565	551	540	556	548	555	533	537	554	540
19	236.613	424	434	422	425	385	451	441	454	440
20	236.661	543	552	537	548	555	549	562	522	502
21	236.709	446	442	478	462	414	449	464	450	444
22	236.757	553	534	555	533	546	559	553	528	553

Figura 4.25: Fichero csv.

Capítulo 5

Resultados experimentales

En este capítulo se mostrarán diversos experimentos realizados sobre varias muestras y los resultados obtenidos usando el software desarrollado en este trabajo. Con estos resultados, los investigadores son capaces de saber de qué sustancias están compuestas las muestras que se están analizando, observando para ello la forma que tienen los espectros.

5.1. Resultados con la interfaz

En primer lugar, se muestran los resultados obtenidos con la interfaz gráfica al realizar los experimentos. La versión de la interfaz gráfica es una versión un poco anterior a la final y no tiene el botón de detener (Figura 4.23) ya que al realizar estas pruebas es cuando se observaron los fallos con los triggers. Tampoco tiene incorporado el botón de guardar los espectros secundarios y mostrar los nombres de los archivos, ya que esto se piensa posteriormente.

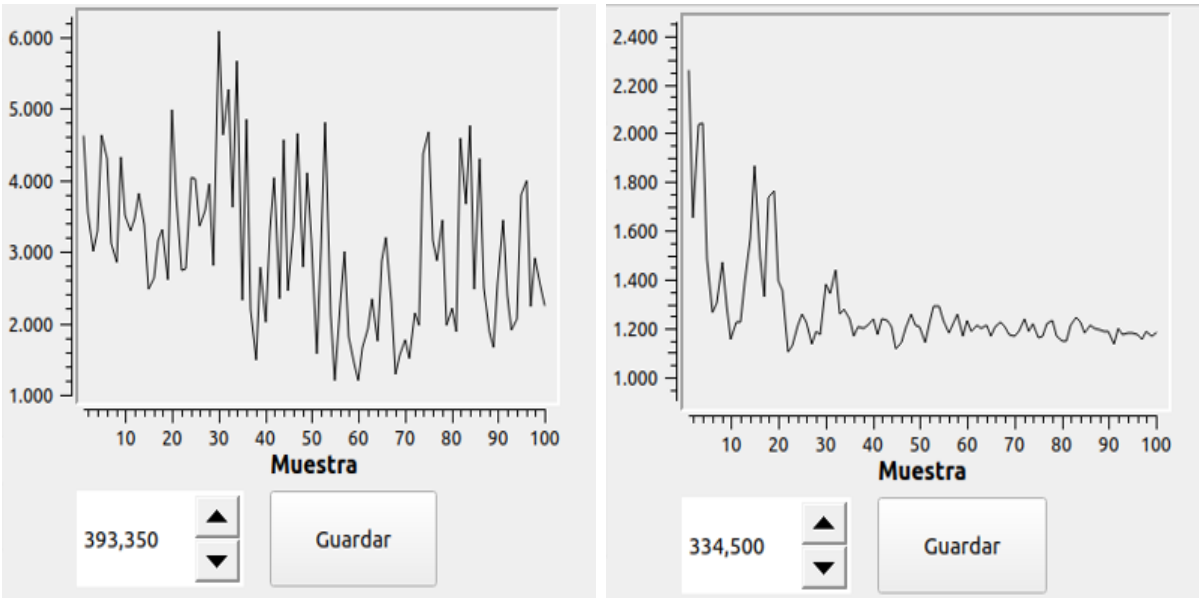
En la Figura 5.1 se muestra toda la interfaz con los resultados de disparar a un tipo de lava del volcán de la Palma donde el investigador tomó varias muestras. También se observa en la Figura 5.1, mediante la foto incrustada en la interfaz, cómo se dispara a la muestra.

A continuación se muestran los espectros de todos los materiales de los cuales se hicieron pruebas, Figura 5.3. En el montaje del equipo se tarda un rato porque el investigador tiene que poner todo a punto, pero en disparar y tomar espectros se tarda muy poco, es por esto que se pudieron tomar mediciones de muchas muestras en poco tiempo (alrededor de 15 segundos por muestra).

La Figura 5.3h se corresponde a una de las varias muestras que se necesitaban tomar para obtener la composición de los materiales del campo de enfrente de la Escuela de Ingenierías de la Universidad de Málaga. La Figura 5.3i es de un tipo de piedra que salía despedida del volcán de la Palma.

Por otro lado, en las Figuras 5.2a y 5.2b se muestran las gráficas de profundidad de granito y acero galvanizado. En la de acero galvanizado se observa un fenómeno muy curioso ya que la longitud de onda que se ha establecido es en la que se encuentra el cinc y se observa que con el paso de las mediciones el cinc desaparece. Esto es debido a que al disparar el láser repetidamente sobre un mismo punto de la muestra se elimina gradualmente la capa superficial de cinc del acero galvanizado.

Figura 5.1: Resultados experimento.



(a) Profundidad granito.

(b) Profundidad acero galvanizado.

Figura 5.2: Gráficas de profundidad.

5.2. Comprobación de los resultados

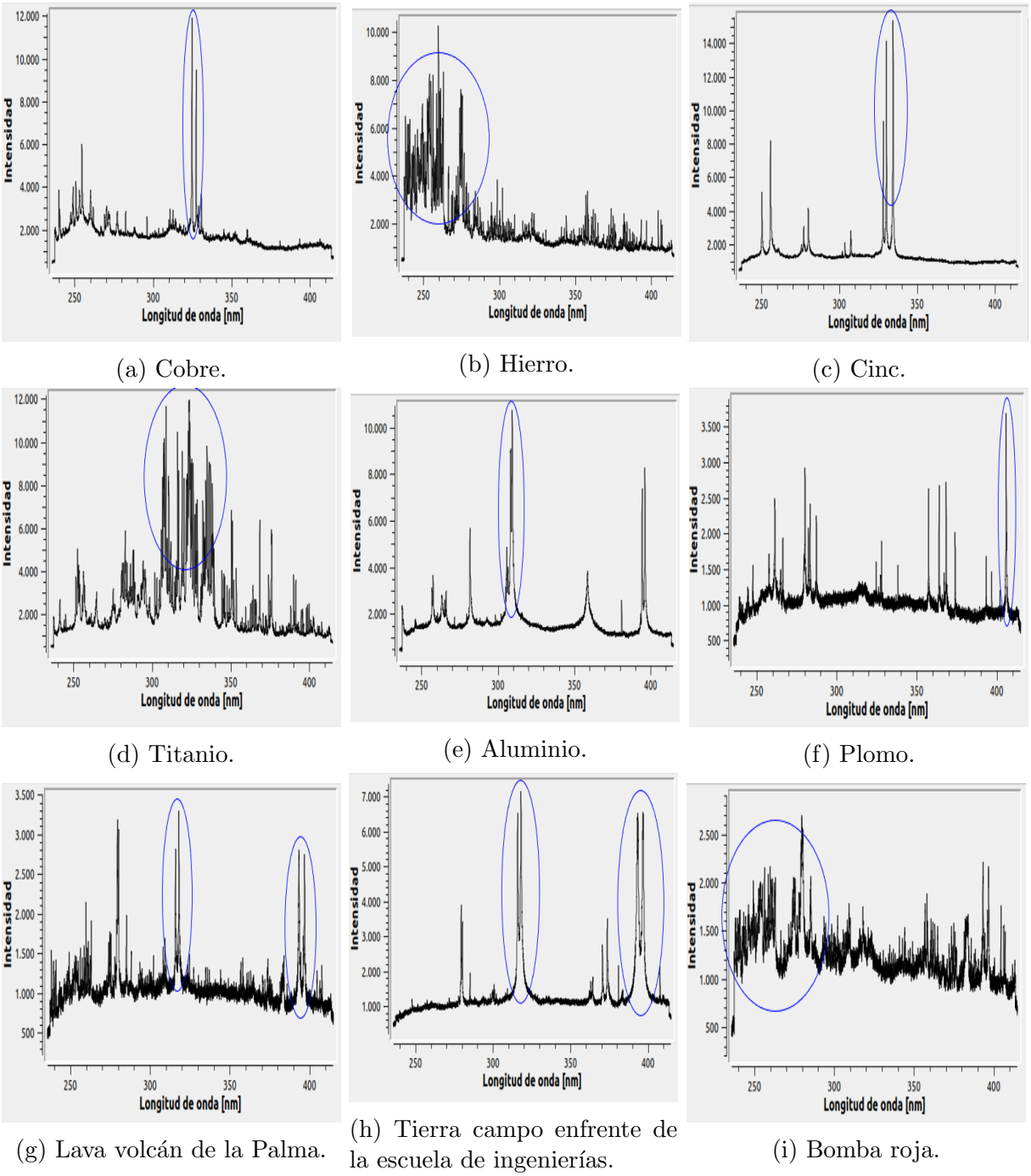
Para terminar este Capítulo de validación de la herramienta desarrollada, se pasa a continuación a comprobar mediante la realización de un experimento en laboratorio con el instrumento, que los resultados obtenidos, medidos y visualizados son correctos y que los espectros obtenidos se corresponden con los resultados esperados con varios patrones de referencia de los que se conocen las emisiones características de sus espectros. En las

Figuras 5.4 y 5.5, obtenidas con permiso de [23], se observan estos espectros.

La práctica habitual es identificar los componentes de una muestra problema comparando su espectro y con los espectros de los patrones; es lo que se denomina análisis cualitativo. En este caso se han tomado tres muestras problema: una roca magmática, una muestra de tierra y una bomba volcánica. A continuación se describe cada una de las Figuras mostradas anteriormente, comprobándose que los resultados obtenidos corresponden con los resultados esperados:

- Cobre. En la Figura 5.3a se observan los dos picos de intensidad del cobre alrededor de los 325 nm, como se observa en la Figura 5.5 mirando el cobre (Cu).
- Hierro. En la Figura 5.3b se observan los picos de intensidad alrededor de 280 nm, como se observa en la Figura 5.5 mirando el hierro (Fe).
- Cinc. En la Figura 5.3c se observan los picos de intensidad alrededor de los 330 nm.
- Titanio. En la Figura 5.3d se observan los picos de intensidad alrededor de los 320 nm, como se observa en la Figura 5.4 mirando el titanio (Ti).
- Aluminio. En la Figura 5.3e se observan los picos de intensidad alrededor de los 310 nm, como se observa en la Figura 5.4 mirando el aluminio que viene representado con el nombre de *2_AVG*.
- Plomo. En la Figura 5.3f se observa el pico predominante alrededor de los 410 nm.
- Lava volcán de la Palma. En la Figura 5.3g vemos un espectro parecido al del aluminio por lo que podemos decir que tiene bastante aluminio además de otros componentes como hierro.
- Tierra campo enfrente de la facultad de ingenierías. En la Figura 5.3h vemos un espectro parecido al del aluminio por lo que podemos decir que tiene bastante aluminio, además de algo de cobre.
- Bomba roja. En la Figura 5.3i vemos un espectro parecido al del hierro con lo que podemos afirmar que tiene una gran cantidad de hierro.

Por comparación de las tres muestras problema con los patrones y de los nueve espectros obtenidos con los de las Figuras 5.4 y 5.5 se comprueba que el instrumento con el nuevo software desarrollado es funcional y genera resultados precisos que permiten la identificación elemental de los componentes de una muestra.



(a) Cobre.

(b) Hierro.

(c) Cinc.

(d) Titanio.

(e) Aluminio.

(f) Plomo.

(g) Lava volcán de la Palma.

(h) Tierra campo enfrente de la escuela de ingenierías.

(i) Bomba roja.

Figura 5.3: Espectros de diversos elementos.

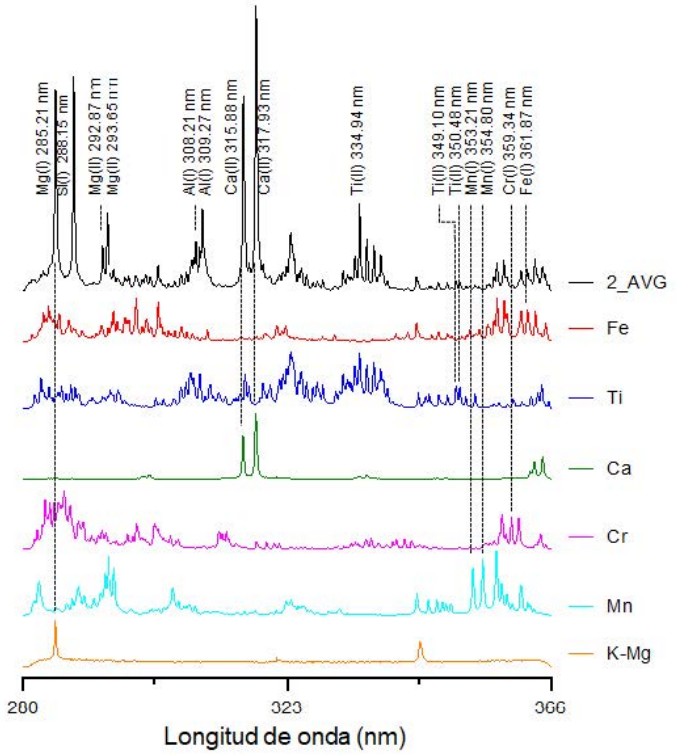


Figura 5.4: Intensidades de varios materiales (I).

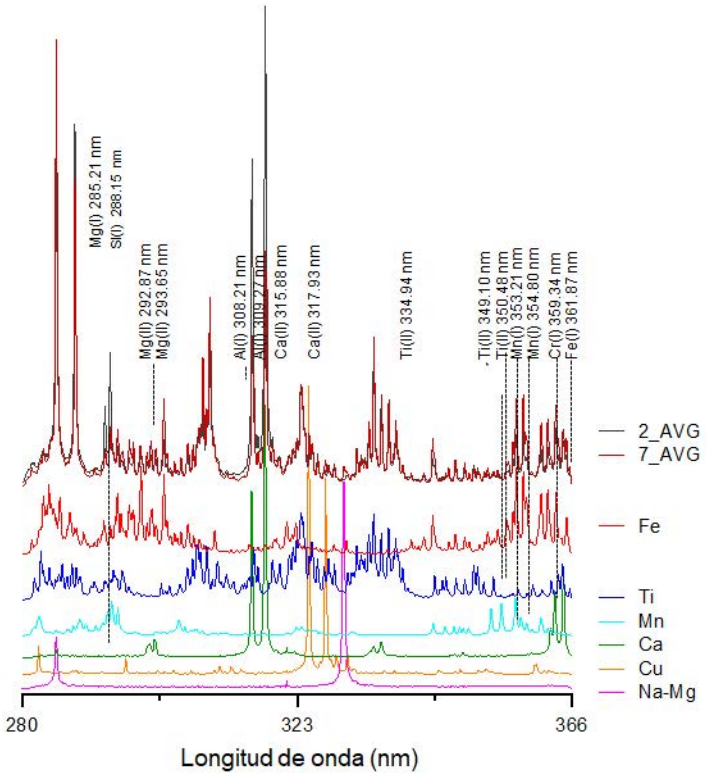


Figura 5.5: Intensidades de varios materiales (II).

Capítulo 6

Conclusiones y trabajos futuros

En este Capítulo se van a resumir los hitos alcanzados, las conclusiones y las líneas de trabajo futuro que se pueden realizar como continuación del trabajo aquí desarrollado.

Al principio del proyecto, se comenzó realizando el conexionado de los distintos dispositivos hardware, desarrollando un código que permitiera la obtención de datos del sensor de distancia y del espectrómetro. A continuación, se unificó todo lo que ya se tenía en un solo programa que, con una interfaz gráfica, permitiera controlar un láser y obtener información de los instrumentos previos.

Los objetivos que se plantearon como elementos indispensables que debía realizar la interfaz gráfica se han completado satisfactoriamente, incluyendo, además, elementos adicionales que los tutores del proyecto han visto oportunos, como mostrar en una gráfica los espectros de una forma alternativa o el almacenamiento de los mismos para un posterior análisis de los experimentos realizados.

Para cumplir, por tanto, los objetivos inicialmente planteados, se ha realizado una interfaz gráfica capaz de controlar el sistema LIBS y de obtener, procesar y almacenar información de los experimentos realizados con este. La interfaz gráfica, muy sencilla y fácil de usar, permitirá al usuario obtener resultados de las muestras que tome de ahora en adelante, además de poder montarlo en un sistema móvil y utilizarlo desde tierra y de forma remota al estar creado en un mini PC alimentado a baterias (LattePanda).

Con todo esto, las conclusiones que se sacan son las siguientes:

- La realización de experimentos en laboratorio y la comparación de muestras problema con patrones y con resultados en la bibliografía permite concluir que el instrumento con el nuevo software desarrollado en este Trabajo de Fin de Grado es plenamente funcional y genera resultados precisos que permiten la identificación elemental de los componentes de una muestra.
- El entorno Qt Creator permite la creación de interfaces gráficas de manera muy sencilla y permite enlazar fácilmente las señales que lleva el programa por detrás.
- Por otro lado, el sistema LIBS es una manera muy cómoda y rápida (una vez que el sistema está montado) de obtener información de la composición de los materiales deseados, permitiendo saber con certeza de qué está formada la muestra en cuestión.

Además, este proyecto ha facilitado y permitido la obtención de resultados a un estudiante

de doctorado de la Escuela de Industriales de la Universidad de Málaga y se ha facilitado el software a un estudiante de máster de química de la Facultad de Ciencias, también de la Universidad de Málaga, autor del proyecto [9], para realizar los experimentos necesarios para su Trabajo Fin de Máster.

A continuación, se hablará de algunas posibles mejoras que se pueden realizar sobre la interfaz como trabajos futuros:

- El auto escalado de la ventana de control remoto. La ventana principal tiene una resolución de 1280×720, la resolución estándar de cualquier tableta o dispositivo móvil; esta resolución se puede cambiar ajustando los valores de la pestaña, pero sería interesante incorporar *sliders* o el uso de una ventana que permita el desplazamiento por la misma, para permitir el uso del software en monitores antiguos, por ejemplo.
- La elección de la longitud de onda para la gráfica secundaria haciendo clic con el ratón en la gráfica principal. Para hacer uso de la segunda gráfica, se introduce un valor de longitud de onda que será el que se usará para dibujar todas las medidas tomadas en esa longitud de onda concreta. Sería muy interesante incorporar la gestión de los clics del ratón (o tocar con el dedo si es una tableta) en la gráfica de los espectros y, al tocar una longitud de onda concreta, guardar y utilizar ese valor en la gráfica secundaria.

Anexo A

Manual de Instalación

Para realizar una instalación correcta de Qt Creator y abrir el compilador con el código desarrollado en este trabajo se seguirán los siguientes pasos:

1. Instalar Ubuntu (versión recomendada 22.04).
2. Instalar librerías esenciales para que funcione Qt Creator:
   - Usar el comando "sudo apt-get update && sudo apt-get upgrade" para actualizar el instalador.
   - Usar el comando "sudo apt-get -y install build-essential openssl libssl-dev libssl1.0 libgl1-mesa-dev libqt5x11extras5" para instalar las librerías.
3. Descargar el instalador de Qt Creator accediendo al siguiente enlace: https://www.qt.io/download-qt-installer. Este enlace descargará automáticamente Qt Creator dependiendo del sistema operativo.
4. Instalar la versión más actual de Qt Creator. Para ello, acceder a la carpeta de descargas y compilar lo que se ha descargado como un ejecutable usando los siguientes comandos:
   - "chmod +x qt*.run" y luego ejecutarlo con "sudo ./qt*.run".
5. Instalar diversas librerías con los comandos "sudo apt-get install libqt5serialport5*" y "sudo apt-get install qtbase5-dev* qtchooser qtcreator qt5-qmake cmake".
6. Instalar paquete de librerías de Qt Creator siguiendo el tutorial del siguiente enlace: https://qwt.sourceforge.io/qwtinstall.html. Además,
   - Descargar la última versión de *Qwt Files*.
   - Acceder a la carpeta descargada y usar los comandos "make", luego "sudo make install".
7. Ya se puede ejecutar el programa y probar que funciona. Para abrir el programa, ya desde Qt Creator se debe dar a *open project* y buscar dentro de la carpeta con el código el archivo que termine en .pro que es el archivo del proyecto.

El código se encuentra disponible en un repositorio de GitHub. [1] Para acceder a dicho

[1] https://github.com/spaceuma/CHEMOCOPTER-GUI

repositorio habrá que pedir acceso a srl@uma.es.

Anexo B

Manual de Usuario

Para ejecutar la interfaz gráfica disponible en el LattePanda se seguirán una serie de pasos:

1. Realizar el conexionado de los distintos elementos y las conexiones de los pines de arduino como se muestran en las Figuras 3.2 y 3.4.
2. Encender el LattePanda, se iniciará la sesión automáticamente.
3. Botón derecho en el archivo *.shell* que se encuentra en el escritorio y hacer click en "Ejecutar como un programa". Esto abrirá la interfaz gráfica al pasar unos segundos.

En caso de no estar en el LattePanda y haberlo instalado en otro dispositivo, se abrirá Qt Creator usando el comando *sudo qtcreator* y se abrirá el archivo .pro disponible con el código. Esto abrirá el compilador y pulsando en botón de *run* se podrá ejecutar.

Una vez dentro de la interfaz gráfica se disponen de una serie de formas de interacción que se explican brevemente a continuación. Para ello, en la Figura B.1 se muestra la interfaz gráfica señalando cada forma de interacción:

1. Parámetros de adquisición del espectrómetro configurables.
2. Trigger interno o externo.
3. Mostrar/ No espectros durante la ejecución.
4. Detener la adquisición de espectros.
5. Iniciar la adquisición de espectros.
6. Número de espectros a tomar durante la ejecución con trigger interno.
7. Formas de elegir y visualizar los espectros de la serie adquirida.
8. Distancia actual y fijada, junto con la opción de Fijar distancia a la actual por medio de un botón.
9. Actuaciones sobre el láser:
    - Conectar/Desconectar: Conectar al láser por Telnet, actualizará el valor del botón y del campo de "Estado Telnet" al pulsarse.

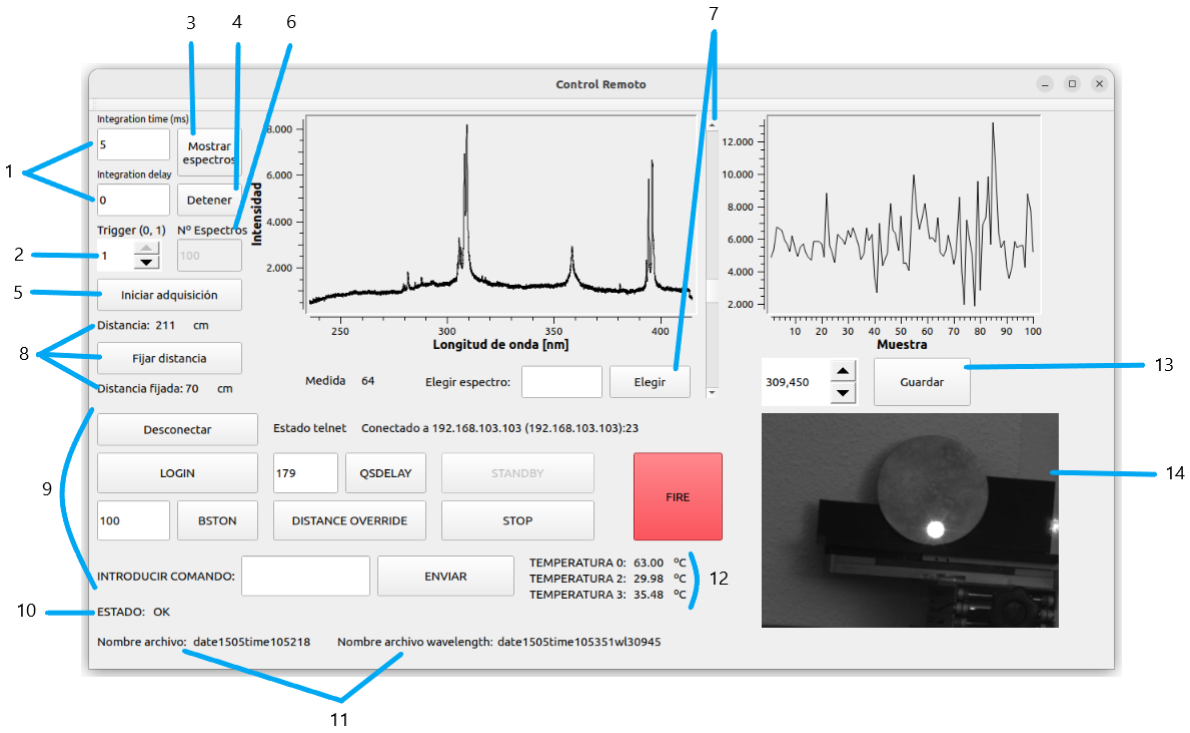


Figura B.1: Interfaz gráfica.

- Login: Hacer un login al láser para poder actuar sobre él. Primer comando necesario para una comunicación con el mismo.
- Bston: Número de disparos del láser.
- Qsdelay: Intensidad del pulso del disparo del láser.
- Distance override: Omitir la distancia fijada para disparar.
- Standby: Inicia el calentamiento del láser. Necesario para disparar.
- Stop: Detiene el calentamiento del láser.
- Fire: Dispara el láser.
- Introducir comando: Para introducir cualquier comando que acepte el láser.

10. Estado actual del láser. Cuando está en OK se puede disparar.
11. Nombres de los archivos de salida.
12. Temperaturas internas del láser.
13. Guardar espectro de profundidad.
14. Imagen obtenida por la cámara.

Bibliografía

[1] *Chemocopter*. https://www.uma.es/sala-de-prensa/noticias/utilizan-una-tecnologia-inedita-de-la-uma-para-medir-la-composicion-quimica-de-la-lava-del-volcan-de-la-palma/. Accessed: 27-04-2023.

[2] *Modelo en cascada*. https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/. Accessed: 27-04-2023.

[3] David A Cremers y Leon J Radziemski. *Handbook of laser-induced breakdown spectroscopy*. John Wiley & Sons, 2013.

[4] Andrzej W Miziolek, Vincenzo Palleschi e Israel Schechter. *Laser induced breakdown spectroscopy*. Cambridge university press, 2006.

[5] *Láser de Rubí*. https://es.wikipedia.org/wiki/Lser_de_rub. Accessed: 25-05-2023.

[6] *A brief history of laser-induced breakdown spectroscopy*. https://www.sciencedirect.com/science/article/abs/pii/S058485471300116X. Accessed: 25-05-2023.

[7] *Principales aportaciones de la técnica LIBS al estudio de biomateriales*. https://www.upo.es/cms1/export/sites/upo/moleqla/documentos/Numero19/Destacado_1.pdf. Accessed: 25-05-2023.

[8] *Uso LIBS en el patrimonio arqueológico sumergido*. https://dialnet.unirioja.es/servlet/articulo?codigo=5796710. Accessed: 25-05-2023.

[9] Alberto Bernal Asensio. *Trabajo Fin de Grado en Química. Evaluación de la espectroscopia de plasmas inducidos por láser para la caracterización de materiales de origen volcánico*. UMA, 2022.

[10] Rodney Loudon. *The quantum theory of light*. OUP Oxford, 2000.

[11] *Lidar Lite v3 Operation Manual and Technical specifications*. https://static.garmin.com/pumac/LIDAR_Lite_v3_Operation_Manual_and_Technical_Specifications.pdf. Accessed: 29-04-2023.

[12] Ray Rischpater. *Application development with qt creator*. Packt Publishing Birmingham, 2013.

[13] *Qt Widgets*. https://doc.qt.io/qt-6/qtwidgets-index.html. Accessed: 27-04-2023.

[14] *Completing Code*. https://doc.qt.io/qtcreator/creator-completing-code.html. Accessed: 27-04-2023.

[15] *GetDistanceI2c*. https://github.com/garmin/LIDARLite_Arduino_Library/blob/master/examples/v3/GetDistanceI2c/GetDistanceI2c.ino. Accessed: 27-05-2023.

[16] *Espectrómetros Avantes*. https://www.antaresinstrumentacion.com/avantes-espectrometros/. Accessed: 30-04-2023.

[17] *QSerialPort class*. https://doc.qt.io/qt-6/qserialport.html. Accessed: 29-04-2023.

[18] *QTelnet class*. https://github.com/silderan/QTelnet. Accessed: 01-05-2023.

[19] *QStringList class*. https://doc.qt.io/qt-6/qstringlist.html. Accessed: 01-05-2023.

[20] *std::vector*. https://cplusplus.com/reference/vector/vector/. Accessed: 02-05-2023.

[21] *Cámaras industriales ueye*. https://en.ids-imaging.com/downloads.html. Accessed: 07-05-2023.

[22] *QThread Class*. https://doc.qt.io/qt-6/qthread.html. Accessed: 08-05-2023.

[23] *Atomic Spectroscopy*. https://www.sciencedirect.com/journal/spectrochimica-acta-part-b-atomic-spectroscopy. Accessed: 27-05-2023.