UNIVERSIDAD DE EXTREMADURA

CENTRO UNIVERSITARIO DE MÉRIDA

MÁSTER UNIVERSITARIO DE INVESTIGACIÓN EN INGENIERÍAS Y ARQUITECTURAS

TRABAJO FIN DE MÁSTER

Detección de variedad y estado de maduración del ciruelo japonés mediante Deep-learning.

Autor: Borja Rodríguez Puerta

Mérida, Julio de 2018

UNIVERSIDAD DE EXTREMADURA

CENTRO UNIVERSITARIO DE MÉRIDA

MÁSTER UNIVERSITARIO DE INVESTIGACIÓN EN INGENIERÍAS Y ARQUITECTURAS

TRABAJO FIN DE MÁSTER

Detección de variedad y estado de maduración del ciruelo japonés mediante Deep-learning.

AUTOR: D. Borja Rodríguez Puerta

Fdo:

DIRECTOR: Dr. D. Francisco Chávez de la O

Fdo:

Mérida, Julio de 2018

PALABRAS CLAVES

Análisis imágenes hiperespectrales, machine learning, redes neuronales.

RESUMEN

Food and Agriculture Organization of the United Nations sitúa a España como el séptimo productor de ciruelas del mundo y el tercero a nivel europeo. La importancia económica que tiene el cultivo de este fruto en nuestro país es evidente. En Extremadura el ministerio de Agricultura, Pesca, Alimentación y Medio Ambiente Español, cifra en 6500Has el territorio dedicado a esta actividad. Una mejora en el proceso de recolección, recolectar el fruto en su momento óptimo, puede suponer una gran ventaja competitiva de las Empresas Extremeñas sobre sus competidoras.

Actualmente los agricultores utilizan técnicas clásicas basadas en su larga experiencia para valorar la calidad de la fruta a lo largo de su periodo de crecimiento. Este método tiene grandes limitaciones ya que las decisiones que se toman dependen en gran medida de la experiencia de estos, se trata pues de métodos altamente subjetivos, que puede derivar en errores en la cosecha, ya sea por recoger el fruto antes de tiempo o incluso con una fecha posterior a su estado óptimo. Para intentar solucionar este problema se están introduciendo nuevas técnicas que ayuden en la correcta toma de decisiones. Entre estas técnicas novedosas se encuentran la visión por computador y algoritmos de machine learning, que si son bien entrenados, podrán obtener la fecha óptima de cosecha.

En este trabajo nos ocuparemos de la detección de la variedad de la ciruela en su fase temprana de maduración, así como su estado mismo de maduración en sus diferentes fases del ciclo de maduración. Se utilizarán para ello el análisis de imágenes HiperEspectrales de diferentes variedades de ciruelo japonés divididas en su estado de maduración.

Para hacer frente a estas identificaciones, el primer objetivo es la automatización de la adquisición de las imágenes. En nuestro caso tenemos que desarrollar una serie de herramientas que permitan tomar un conjunto necesario de imágenes hiperespectrales por cada ciruela. El segundo objetivo es el análisis de las imágenes obtenidas utilizando técnicas de machine learning, lo que nos permitirá obtener diferentes clasificadores para las ciruelas, ya sea por variedad y/o estado de maduración.

KEY WORDS

Analysis of hyperspectral images, machine learning, neural networks

ABSTRACT

Food and Agriculture Organization of the United Nations (FAO) ranks Spain as the seventh largest producer of plums in the world and the third largest in Europe. The economic importance of growing this fruit in our country is evident. In Extremadura there are 6500 hectares of land dedicated to this activity, according to the Spanish Ministry of Agriculture, Fisheries, Food and the Environment. Improvements in the harvesting process, harvesting the fruit at its optimum time, can give Extremadura companies a great competitive advantage over their competitors.

Farmers today use classical techniques based on their long experience to assess the quality of the fruit throughout its growing period. This method has great limitations due to the decisions are taken depending on this experience. This technique is based on highly subjective methods, which can lead to errors in the harvest, either by picking the fruit early or even with a date later than its optimal state. Trying to solve this problem, new techniques to help in the correct decision making are being introduced. Among these novel techniques we can found the computer vision and machine learning algorithms, which, if they are well trained, can optimize the harvest date.

Today, farmers use classic techniques based on their long experience to assess the quality of the fruit over the years. This method has great limitations as the decisions made depend largely on the following of their growth spurt. Therefore, the farmers experience, is highly subjective and can lead to errors in the harvest, as is the case with the either by harvesting the fruit early or even after its optimum condition. Trying to solve this problem, new techniques are being introduced to help in the correct decision making. Among these techniques, we can found the newest, such as computer vision and machine learning algorithms, which, if they are well trained, would be able to optimize the harvest date.

In this work we will deal with the detection of the plum variety in its different stages of the ripening cycle. Hyperspectral image analysis of different varieties of plums divided into their ripeness stage will be used for this purpose. To achieve these identifications, the first objective is the automation of image acquisition process. It will be necessary to develop a set of tools that allows us to take a necessary set of hyperspectral images for each plum. Once the set of images is completed, in the second objective the set of images will be analyzed with machine learning techniques that will allow us the creation of plum classifiers by variety and ripeness stage.

Agradecimientos

A la primera persona que se lo quiero agradecer es a mi tutor el Dr. Francisco Chávez de la O, que sin su ayuda, conocimientos y su apoyo incondicional en los momentos más difíciles, no habría sido capaz de realizar este proyecto. Agradecer a los componentes del grupo de investigación GEA, en especial a Francisco Javier Rodríguez Díaz, de la universidad de Extremadura por su ayuda y colaboración con este proyecto. También me gustaría agradecerle su colaboración al Doctor Pedro José Pardo Fernández, grupo Orión, ya que sin sus conocimientos de óptica este proyecto no podría haberse realizado

A mis padres, por haberme proporcionado la mejor educación y lecciones de vida. Gracias a ellos aprendí que con esfuerzo, trabajo y constancia todo se consigue.

A todos mis familiares, por su apoyo. En especial a mi pareja Marta Iglesias, por cada día hacerme ver la vida de una forma diferente y apoyarme en mis decisiones. En último lugar, pero no por ello menos importante, mostrar mi agradecimiento a todos los profesores del Centro Universitario de Mérida por haberme hecho, además de un gran profesional, una mejor persona.

Índice general

Índice de figuras

Índice de Código

Índice de Tablas

Capítulo 1

Introducción

En la actualidad, España ocupa el séptimo puesto como productor de ciruelas a nivel mundial y el tercero a nivel europeo según la Organización de las Naciones Unidas para la Alimentación y la Agricultura. La importancia que tiene el cultivo de esta fruta en nuestro país es evidente, siendo mayor en las Comunidades Autónomas que centran su actividad económica en el sector primario. Este es el caso de Extremadura, donde según datos del Ministerio de Agricultura, Pesca, Alimentación y Medio Ambiente Español cifra en 6500Has el territorio dedicado a esta actividad. En este contexto, una mejora en la calidad de las ciruelas supondría una ventaja competitiva de las empresas extremeñas sobre sus competidoras.

La calidad de un producto es percibida por el consumidor como un conjunto de atributos que son evaluados de forma subjetiva, con el fin de obtener una medida que le permita escoger el mejor. Si atendemos a las frutas, la calidad es por tanto una percepción subjetiva de los diferentes componentes de esta: apariencia, aroma, sabor, etc. que nos ayuda a decidir si consumir el producto. La calidad en la fruta se puede mejorar de diversas maneras: mejorando los procesos de cultivo o recolectando el fruto en el momento óptimo, de esta manera, cuando llegue al consumidor, sus cualidades se encuentren en su estado ideal.

La clasificación de la ciruela por su estado de maduración es un proceso que se realiza de manera manual, lo que puede llevar a clasificaciones erróneas. Este proceso tradicionalmente ha sido realizado por operarios humanos, agricultores o técnicos, que con la experiencia adquirida durante los años de trabajo, son capaces, de manera visual, de clasificar las ciruelas por su calidad. Este método tiene grandes limitaciones, ya que las decisiones que se toman dependen en gran medida de la experiencia de estos, se trata pues de un método altamente subjetivo, que puede derivar en errores en la cosecha, ya sea por recoger el fruto antes de tiempo o incluso con una fecha posterior a su estado óptimo. Para intentar solucionar este problema y ayudar a los operarios, las nuevas líneas de investigación están introduciendo nuevas técnicas [1] que tienen como finalidad ayudar en la correcta toma de decisiones.

Entre estas técnicas novedosas se encuentran las técnicas basadas en visión por computador [2],[4],[5] y algoritmos de machine learning[3]. Este tipo de algoritmos requieren de un proceso intensivo de aprendizaje, que una vez concluido, obtienen potentes clasificadores que ayudarán a los agricultores y técnicos a tomar la decisión más acertada, en función a los parámetros óptimos de cosecha de las ciruelas.

La correcta clasificación de las ciruelas por su estado de maduración es de gran importancia, esto es debido a que ciertas propiedades internas del fruto (solidos solubles, brix, firmeza, etc.) están directamente relacionadas con el estado de maduración del fruto. Para poder conocer el estado de estas variables, es necesario la utilización de técnicas analíticas en un laboratorio. Los principales inconvenientes de estas técnicas son diversos, a continuación, se detallan algunos. En un primer lugar, se trata de técnicas destructivas, que mediante la destrucción del fruto, se obtienen sus propiedades químicas. En un segundo lugar, estás técnicas consumen una gran cantidad de tiempo ya que se trata de complejos análisis que tienen que ser llevados a cabo por técnicos cualificados, que sumado al elevado coste de los materiales necesarios, hace que este tipo de técnicas solo sean aplicables en centros de investigación. Para finalizar, el último inconveniente de estas técnicas es que solo son realizadas a un número limitados de muestras y por lo tanto no son representativas de la variabilidad química que se puede encontrar dentro de un lote de frutas.

La inspección visual de la fruta ha sido utilizada por la industria durante muchos años, pero como se cita con anterioridad, esta puede dar lugar a inconsistencias y variaciones a pesar de la capacitación profesional de los operarios. La variabilidad asociada a la evaluación realizada por los humanos puede ser mejorada o corregida con la implementación de tareas de inspección automatizadas que sean capaces de proporcionar información confiable del estado de las ciruelas.

Los sistemas de control de calidad basados en visión por computador han sido utilizados ampliamente por la industria agroalimentaria. Estos sistemas basan su funcionamiento en la utilización de cámaras digitales que junto con algoritmos de machine learning son capaces de clasificar los alimentos según la calidad de estos. La utilización de estas técnicas en el proceso de selección aporta las siguientes ventajas: tiempo de análisis reducido en comparación con las pruebas de laboratorio y alta precisión en la selección a un coste reducido.

La utilización de técnicas de machine learning por las industrias alimentarias en los procesos selectivos es habitual [3], se han aplicado Random decision forests [6], support vector machines[7] y redes neuronales en la evaluación de la calidad del producto [8]. En la actualidad, la creación de sistemas de control basados en algoritmos de machine learning es un campo muy prolífico, son numerosos los estudios que se publican todos los años proponiendo nuevas metodologías y comparándolas con los métodos clásicos, esto es debido a que no existe un consenso claro sobre que técnicas funcionan mejor.

El objetivo del trabajo que aquí se presenta es la detección de la variedad de

la ciruela en su fase temprana de maduración, así como su estado de maduración. Se utilizarán para ello el análisis de imágenes RGB e Hiperespectrales de diferentes variedades de ciruelas divididas en diferentes semanas de maduración. El uso de imágenes hiperespectrales para el análisis de calidad en productos agroalimentarios es ampliamente utilizado [9][10][11][12][13], obteniendo buenos resultados.

1.1. Justificación

El trabajo que aquí se presenta se ha desarrollado en el ámbito de tareas de dos proyectos de investigación que han sido concedidos al grupo de investigación del cual es miembro el director de este trabajo. Un primer proyecto, es un proyecto del plan nacional de investigación concedido por el Ministerio de Economía y Competitividad, titulado "Nuevos modelos de computo bioinspirados para entornos masivamente complejos"(TIN2017-85727-C4-4-P). El segundo proyecto, más centrado en el trabajo aquí desarrollado, lleva por título .Estudio de evolución y maduración del ciruelo japonés mediante análisis hiperespectral y sistemas inteligentes"(IB16035), concedido por la Consejería de Economía e Infraestructuras de la Junta de Extremadura.

El primer proyecto mencionado anteriormente es una colaboración entre diversas universidades españolas que tiene como objetivo investigar el uso de algoritmos bioinspirados en entornos masivamente complejos. La idea de este tipo de algoritmos es que tengan la capacidad de gestionar recursos computacionales heterogéneos y grandes colecciones de datos a la vez. La problemática estudiada considerará la resiliencia, el consumo energético y la autogestión de estas técnicas. Para poner a prueba las técnicas desarrolladas consideraremos diferentes áreas, entre ellas el análisis de dataset con multitud de imágenes. El segundo proyecto, en el que se encuentra enmarcado el trabajo de investigación aquí presentado, tiene por título .Estudio de evolución y maduración del ciruelo japonés mediante análisis hiperespectral y sistemas inteligentes"(IB16035). El objetivo de este segundo proyecto es el desarrollo de herramientas autoadaptables que ayuden a la toma de decisiones durante el ciclo de cultivo del ciruelo japonés, con la finalidad de obtener un fruto de mayor calidad. El proyecto cuenta con un grupo multidisciplinar compuesto por el grupo de investigación de evolución artificial de la universidad de Extremadura (GEA) y El Centro de Investigaciones Científicas y Tecnológicas de Extremadura (CICYTEX).

1.2. Estado del Arte

La literatura relativa a la temática que nos compete en este trabajo fin de máster podemos encontrar una de las referencias más relevantes, que tiene como objetivo el estudio y detección de la maduración en los frutos con hueso [1].

La aplicación de técnicas de visión artificial para el análisis de los alimentos ha aumentado considerablemente en los últimos años [2] [3] [4]. La diversidad de las aplicaciones depende, entre otras cosas, del hecho de que los sistemas de visión artificial proporcionan información sustancial acerca de la naturaleza y los atributos de los objetos presentes en una escena. Otra característica importante de tales sistemas es que abren la posibilidad de estudiar estos objetos en las regiones del espectro electromagnético donde el ojo humano es incapaz de operar, como en el ultravioleta (UV), infrarrojo cercano (NIR) o infrarrojo (IR).

Investigadores de todo el mundo han estudiado el potencial de diversas técnicas para conocer la calidad de la fruta [15][5]. Una de las más utilizadas han sido las diferentes técnicas de espectroscópicas, como la NIR [16]. El éxito de la utilización de estas técnicas reside en las ventajas que aportan a los investigadores, siendo las siguientes:

- El proceso de medición es simple y rápido, no son necesarios pretratamientos complejos o reacciones químicas sobre el fruto.
- Se trata de una técnica no destructiva, no es necesario destruir el fruto para conocer sus parámetros de calidad.
- Permiten conocer varios atributos de la fruta al mismo tiempo.

El inconveniente que tienen las técnicas de espectroscopia es que solamente nos aportan información de los componentes químicos y físicos de la fruta en el punto medido. Para poder obtener más información de la fruta es necesario combinar esta técnica con la adquisición de imágenes. Cuando utilizamos las técnicas de espectroscopia, en realidad tenemos que conocer que necesitamos una combinación de técnicas espectroscópicas con adquisición de imágenes clásicas. Esto nos proporcionará dos tipos de información diferentes. Por un lado, tenemos la información obtenida de la adquisición de la imagen del fruto, que se trata de una información espacial: morfología, tamaño, etc. Por otro lado, mediante la espectroscopia, obtenemos información sobre los componentes químicos y propiedades físicas que componen la fruta. Sin embargo, si nos centramos en las técnicas espectrales de imágenes, nos permiten la adquisición de imágenes de frutas e información espectral simultáneamente, con las ventajas de una alta resolución espectral y múltiples bandas de ondas. De acuerdo con la resolución espectral, la espectroscopia de imágenes se puede dividir en imágenes multiespectrales, imágenes hiperespectrales e imágenes ultraespectrales. Las imágenes multiespectrales y las imágenes hiperespectrales son factibles para la medición de los parámetros de calidad de la fruta [14].

Una nueva técnica ha surgido en los últimos años con fuerza en el campo del aprendizaje profundo[17] (Deep Learning, en inglés), se trata de las redes neuronales convolucionales [8]. Estas redes neuronales basan su funcionamiento en un aprendizaje jerarquizado en el cual estructuras de alto nivel son construidas de manera

automática a partir de estructuras de más bajo nivel llamadas capas, comenzado por los datos sin procesar: los píxeles de una imagen. Deep learning surge como una alternativa frente a una mayoría de técnicas de aprendizaje que están basadas solamente en una o, a lo sumo, dos capas de transformaciones no lineales de características.

El deep learning a través de redes neuronales convolucionales[17] es una alternativa a los métodos clásicos de clasificación que requerían de una cuidadosa selección de las características realizada a mano. Los métodos clásicos han demostrado ser bastante eficaces para resolver problemas simples o problemas bien delimitados, pero tropiezan con dificultades para hacerles frente con problemas complejos del mundo real tales como objetos y reconocimiento de voz. Es en estos problemas complejos donde dep learning está resultando ser verdaderamente efectivo.

Capítulo 2

Objetivos

Los objetivos principales de este trabajo fin de máster se presentan a continuación:

- Creación de herramienta software para la captura de datos de imágenes RGB e imágenes hiperespectrales de ciruelas en entornos controlados de laboratorio.
- Análisis de imágenes hiperespectrales con técnicas de deep learning para diseño y optimización de clasificadores de ciruelas, tanto por variedad como por estado de maduración.

El estudio del arte presentado en la sección 1.2 nos avala para el alcance de los objetivos marcados anteriormente, ya que, como demuestran los trabajos, la utilización de diferentes técnicas han arrojado buenos resultados. Si estas propuestas, las unimos a los métodos que a lo largo de este documento detallaremos, el éxito es evidente.

Tal y como se comenta anteriormente, el primer objetivo de este trabajo será la creación de una herramienta software que posibilite la adquisición de imágenes RGB e hiperespectrales simultáneamente. Se trata de la creación de un software específico que nos permita obtener de forma semi-automática las imágenes que se necesitarán para los procesos de aprendizaje y optimización de las técnicas que se utilicen en este trabajo. Esta captura de imágnes se realizará siempre en un entorno controlado de luminosidad en un laboratorio. Es importante que la aplicación posibilite al usuario las siguientes acciones:

- Rotación de las piezas de fruta: de cada una de las ciruelas es necesario fotografiar toda su superficie, por ese motivo se tomarán imágenes rotando el fruto 90 grados entre captura y captura.

- Control de las cámaras: Las piezas de frutas serán fotografiadas con dos cámaras diferentes, una cámara RGB y otra cámara hiperespectral. La aplicación tiene que permitir el control simultaneado de ambas cámaras de forma remota, de esta manera se evita la manipulación de éstas por parte del usuario y los posibles desajustes en las calibraciones del hardware, apartado muy importante a la hora de generar un correcto dataset.

El segundo objetivo descrito tiene como finalidad el análisis de las imágenes hiperespectrales, para poder desarrollar varios clasificadores que posibiliten diferenciar las ciruelas por su variedad, además del estado de maduración de las mismas. Se pretende analizar las imágenes hiperespectrales de las piezas de fruta que conforman el dataset, para localizar espectros de la fruta que nos permitan clasificarla de forma correcta, bajo criterios de variedad y maduración. Debido a la complejidad de este objetivo, debe ser dividido en las siguientes fases:

- Obtención de imágenes hiperespectrales: a través de la herramienta software desarrollada en el primer objetivo.
- Creación dataset: con las imágenes hiperespectrales obtenidas, seis por cada pieza de fruta, se debe generar un dataset, lo suficientemente extenso como para que las técnicas de deep learning obtengan los mejores resultados posibles.
- Utilización de redes neurolales convoluciones (Convolutional Neutal Network, CNN): se usarán CNNs, que haciendo uso del dataset anteriormente descrito, deberán de obtenerse redes optimizadas para los tipos de clasificación mencionados.
- Análisis de los datos: se estudiarán los resultados obtenidos con las CNNs para tratar de determinar qué longitudes de onda son más prometedoras para la clasificación de las ciruelas tanto por variedad como por estado de maduración.

Capítulo 3

Metodología y desarrollo

A lo largo de este capítulo se describe la metodología que se ha llevado a cabo para el análisis de las imágenes de los frutos del ciruelo japonés, por medio de técnicas de análisis hiperespectral.

El núcleo del trabajo aquí presentado se centra en el análisis de un gran conjunto de imágenes procedentes de los frutos del ciruelo japonés. Para alcanzar los objetivos descritos en el capítulo 2 se realizarán dos tareas o fases claramente diferenciadas. La primera fase es la creación de un dataset, compuesto por imágenes hiperespectrales. Se estudiarán tres variedades de ciruelo japonés, cada una de ellas cuenta con su propio ciclo de maduración. Las variedades que forman parte de este estudio son las siguientes: Angeleno, Owent y Black Splendor. El proceso consta de una captura múltiple de cada fruto, compuesta por cuatro imágenes por fruto, que se obtienen girando 90 grados el fruto y tomando la fotografía hiperespectral de la cara indicada, con lo que se puede llegar a tener una reconstrucción hiperespectral del fruto. En la segunda fase se aplicará el uso redes neuronales convolucionales (CNN) que nos permitirá poder clasificar las ciruelas por variedad. Un segundo proceso de ajuste de las CNNs nos permitirá clasificar las ciruelas por estado de maduración. Además el proceso tiene como objetivo el poder detectar capas hiperestrales prometedoras para la clasificación, no teniendo que utilizar todo el espectro capturado.

Para poder obtener las imágenes en el entorno de laboratorio, será necesario el desarrollo de herramientas específicas que permitan fotografiar la imagen con las exigencias requeridas por el estudio. La aplicación debe permitir la captura de las imágenes hiperespectrales, a la vez que se toman imágenes convencionales RGB, al utilizar dos tipos de cámaras simultáneamente. Teniendo como requisito la automatización del proceso, se ha creado una herramienta software, que a través de un entorno específico de trabajo, permite el control de ambas cámaras y además permite la rotación del fruto 90 grados gracias a una plataforma giratoria, lo que nos permite obtener una reconstrucción del fruto.

La segunda fase de trabajo que aquí se presenta, utilizará el conjunto de imágenes

generado para un estudio en profundidad, donde, aplicando técnicas de deep learning, podamos obtener clasificadores que permitan indicar la variedad de la imagen de la fruta analizada, así como el estado de su maduración. Dentro de las posibles técnicas que nos podemos encontrar en la literatura, las CNNs aportan una herramienta muy potente que se ajusta muy bien al tipo de problemas que estamos intentando resolver en este trabajo, de ahí que se utilicen como base en el trabajo propuesto. Este tipo de algoritmos, al tratarse de algoritmos en el entorno de la inteligencia artificial, requieren un proceso de aprendizaje, el cual puede llegar a ser muy costoso según al problema que se enfrenten. La finalidad perseguida con este entrenamiento es lograr que la CNN aprenda de manera autónoma a clasificar imágenes del fruto por su variedad y maduración. La red propuesta para este proyecto ha sido Alexnet (ver descipción en la sección3.2.2).

A lo largo de este capítulo se presenta el desarrollo de la aplicación generada para el control automático de toma de imágenes 3.1, junto con la descripción de análisis hiperespectral del data set obtenido 3.2.

3.1. Desarrollo de la aplicación

Una parte importante del proyecto de investigación que aquí se presenta es el desarrollo de una herramienta software que permita la captura de manera automática de imágenes RGB e hiperespectrales de las ciruelas. Estás últimas conformarán el dataset sobre el que posteriormente trabajaremos en la fase análisis, para lograr clasificar los frutos por variedad y estado de maduración. El software anteriormente citado permite la captura de imágenes de forma semi-automática, apenas existe interación del usuario. Gracias a las herramientas desarrolladas y aquí descritas, se permite controlar de manera remota los diferentes componentes hardware necesarios en la captura de las imágenes (cámara RGB, cámara hiperespectral y plataforma giratoria). A continuación, se describe en detalle la configuración que se ha realizado de dichos componentes hardware y cómo estos han sido integrados para dar lugar a la herramienta software que se anexa junto a esta memoria.

3.1.1. Configuración cámara RGB

En la captura de las imágenes RGB se ha utilizado una cámara digital científica Qimaging modelo Retiga-Exi -QImaging, Canadá. Esta cámara está equipada con un sensor CCD con una resolución de 1392 X 1040 píxeles, cuenta con una precisión de 12 bits por píxel. En la fig. 3.1 se muestra una imagen de la cámara. La montura de la lente es tipo C y la interfaz de comunicaciones es FireWire, lo que proporciona hasta 10 fotogramas por segundo a resolución completa. Las principales ventajas que obtenemos al utilizar esta cámara son la gran sensibilidad y linealidad en la respuesta sobre un amplio rango de valores de luminancia. También es importante

citar que posee muy poco ruido térmico, esto es debido al sistema de enfriamiento basado en una célula termoeléctrica Peltier que tiene instalada.

Figura 3.1: Cámara digital científica modelo Retiga-Exi -QImaging

Todo el proceso de captura de imágenes se va a realizar con una cabina de iluminación Matcher Modelo MM-4e - ver fig 3.2 - equipada con cuatro fuentes de luz: Simulador de luz diurna 6500K y de 5000K y una fuente ultravioleta para medir la fluorescencia si es necesario. En particular, el simulador D65 ha sido elegido como fuente luminosa para realizar las capturas fotográficas. La cabina de luz tiene un fondo gris espectral neutro N7 que facilita las tareas de segmentación de imágenes y además es robusto contra los cambios de fuente de luz.

Figura 3.2: cabina de iluminación Matcher Modelo MM-4e

La polarización electromagnética es un fenómeno natural que puede producirse en las ondas electromagnéticas, como la luz, a través de varios medios. Uno de ellos es la polarización por reflexión, que genera una polarización lineal de la luz al reflejarse sobre una superficie pulida, haciendo que el campo eléctrico de la onda reflejada oscile sobre un único plano. Este fenómeno, asociado a los reflejos a la hora de realizar fotografías, puede ser controlado mediante la inserción de un filtro polarizador lineal que solo deje pasar la luz cuyo plano de oscilación del campo eléctrico sea perpendicular al plano de polarización de la luz reflejada, eliminando así el reflejo. Debido a la superficie pulida de las ciruelas, se ha tenido que utilizar un filtro polarizador para minimizar la incidencia de los reflejos nos deseados a la

hora de tomar las imágenes. Este filtro está compuesto por un cristal polarizador, que al ser rotado, alineamos perpendicularmente al plano de polarización y de esta manera se pueden minimizar los reflejos no deseados sobre la pieza de fruta. Podemos imaginar, el filtro polarizador es como una rejilla que permite únicamente el paso de la luz que oscila en el plano paralelo al vector normal a la superficie de la reja. La luz transmitida al otro lado del polarizador se considera luz polarizada.

Para conseguir que los colores que se obtienen con un cámara digital reflejen fielmente la realidad, es necesario realizar una calibración antes de comenzar con el proceso de toma de imágenes. Para la calibración de la cámara Retiga-Exi -QImaging se ha utilizado el toolbox de Matlab conocido como imaqtool. Los pasos realizados para la correcta calibración han sido los siguientes:

- Abrimos Matlab y ejecutamos el comando imaqtool. Aparecerá una interfaz gráfica presentada en la Fig 3.3, donde podremos seleccionar la cámara a utilizar, en nuestro caso BAY16_1392x1040.

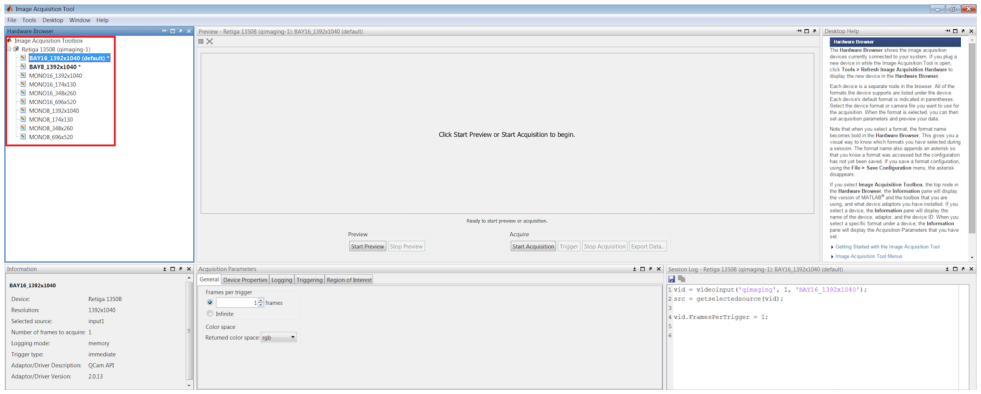

Figura 3.3: Toolbox imaqtool, Matlab

- En la parte inferior de la interfaz, Acquisition Parameters, tenemos que seleccionar la pestaña Device Properties.
- Ponemos todos los valores de los canales RGB a 1, en nuestro caso a 1000, como se indica en la Fig3.4. Es importante fijarse que los colores vienen definidos en el formato BGR.
- Jugando con el tiempo de exposición (exposure), tenemos que realizar una fotografía sobre una carta de color. para ello realizamos clic en el botón Start Acquisition y posteriormente tenemos que exportarla al workspace de Matlab.

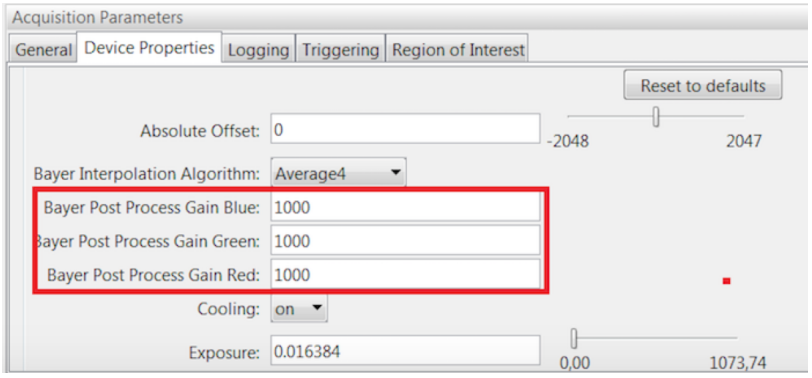

Figura 3.4: Configuración canales RGB

- En la consola de Matlab (ventana del programa de Matlab no en el toolbox) tenemos una nueva variable con el nombre indicado en el paso anterior. Como hemos utilizado una cámara que realiza fotografías en 12bit es necesario convertir la imagen a 8 Bit. El proceso sería el siguiente:

Listado de Código 3.1: Script para pasar imágen de 12bit a 8 bit

```
%El valor 4095 se obtiene por 2^12, donde 12 se
   corresponde con los bits de la camara
resultado=uint8(double(nombre_variable)./4095.*255);
%Mostramos la imagen
imshow(resultado)
```

- En la imagen que se muestre por pantalla, la carta de color fotografiada con anterioridad, tenemos que realizar clic sobre el blanco puro de la imagen para obtener sus valores RGB.
- Hay que repetir el proceso hasta lograr obtener un valor comprendido entre 220-245 en el canal G. Para ello, tenemos que ir modificando el parámetro de exposición, hasta alcanzar el valor deseado.
- Una vez obtenido el valor deseado en el canal G calculamos los otros dos canales. Como ejemplo de este proceso se muestra el código 3.2.

Listado de Código 3.2: Ejemplo obtención canales R y B

```
%Supongamos que obtenemos los siguientes valores en el
   blanco puro de la carta de color: RGB(127,223,180)

R = (valor canal G / valor canal R ) = 223/127 = 1,755
```

```
B = (valor canal G / valor canal B ) = 223/180 = 1,238
```

- Para comprobar que los canales están bien calculados, con el uso del toolbox, configuramos los valores RGB y capturamos una imagen de nuevo. Es importante tener en cuenta que los valores en el toolbox se muestran de manera BGR, Fig3.5.

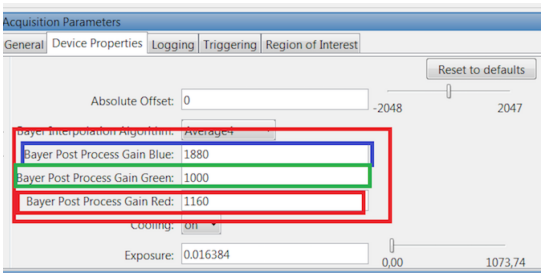

Figura 3.5: Configuración RGB

Una vez se ha finalizado la configuración de la cámara, el siguiente paso es introducir la configuración realizada dentro de la herramienta software que hemos desarrollado. Para capturar imágenes RGB a través de la cámara BAY16_1392x1040 hemos creado una función específica, ver Código 3.3.

Listado de Código 3.3: Función capturar imagen RGB

```
function capturarRGB(app,path,modo)
    vid = videoinput('qimaging', 1, 'BAY16_1392x1040');
    src = getselectedsource(vid);
    vid.FramesPerTrigger = 1;
    src.BayerPostProcessGainBlue = 1460;
    src.BayerPostProcessGainGreen = 1000;
    src.BayerPostProcessGainRed = 1740;
    src.Exposure = 0.08;
    vid.ROIPosition = [276 561 652 418];
    start(vid);
    Img=uint8((double(getdata(vid))/4095)*255);
    stop(vid);
    I = imshow(Img, 'Parent', app.UIAxesImg);
    app.UIAxesImg.XLim = [0 I.XData(2)];
    app.UIAxesImg.YLim = [0 I.YData(2)];
    if modo==1
```

```
        split=strsplit(path,'\');
        nombreFichero=string(strcat(path,split(6),'grados_Q8.
            png'));
        imwrite(Img,char(nombreFichero));
        nombreFichero=string(strcat(path,split(6),'grados.mat'
            ));
        save(nombreFichero,'Img');
    end
end
```

3.1.2. Configuración cámara hiperespectral

Al igual que la cámara convencional requiere una calibración previa, la cámara hiperespectral, al tratarse de un tipo de cámara más sofisticado, requiere nuevamente una fase muy importante de calibración. A lo largo de esta sección se explica el proceso que se debe llevar a cabo para su correcta calibración.

Es conveniente definir bien lo que se entiende como imagen hiperespectral antes de continuar. Una imagen hiperespectral se puede entender como un set de imágenes de un mismo objeto. Cada una de las imágenes, conocidas como bandas, se componen de una imagen espectral que representa a dicho objeto en una longitud de onda diferente. La estructura de una imagen multidimensional sería la mostrada en la Fig. 3.6.

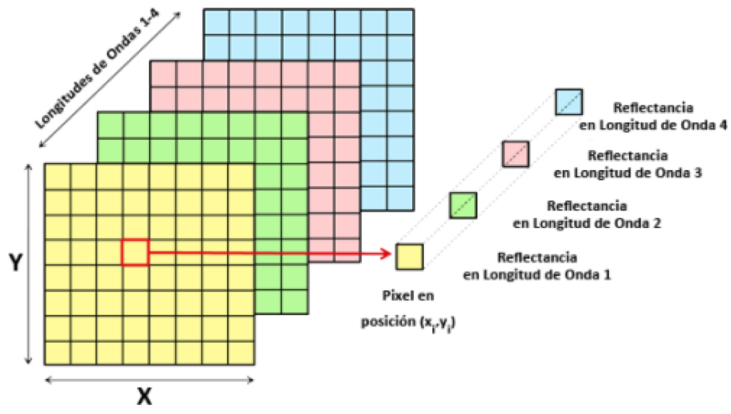

Figura 3.6: Estructura imagen multisimensional de cuatro bandas

donde se muestra una imagen multiespectral, si hablamos de una imagen hiperes-

pectral tiene la misma composición, pero con un numero de bandas más elevado. En este caso el orden de magnitud de N, donde N es igual al número de bandas, permite realizar una distinción a la hora de hablar de imágenes multidimensionales. Así, cuando el valor de N es reducido, típicamente unas cuantas bandas espectrales, se habla de imágenes multiespectrales, mientras que, cuando el orden de magnitud de N es de cientos de bandas, se habla de imágenes hiperespectrales.

Para la captura de las imágenes hiperespectrales se ha utilizado la cámara hiperespectral Cuber UHD 285. Dicha cámara cuenta con un rango de longitud de onda comprendido entre los 450 - 950 nm con un intervalo de submuestreo cada 4 nm. Los fabricantes garantizan 125 canales de información hiperespectral, aunque la cámara es capaz de ofrecer hasta los 138 canales. Estos canales extras también son tenidos en cuenta en este proyecto con la esperanza de descubrir si aportan información sustancial que pueda ser utilizada en fases posteriores. Una de las principales ventajas que aporta la cámara anteriormente citada, es la simplicidad en la captura de las imágenes ya que su proceso, de apuntar y disparar, es muy parecido al de las cámaras tradicionales RGB, permitiendo la obtención de imágenes hiperespectrales en un corto periodo de tiempo, este hecho no es común dentro de las cámaras hiperespectrales ya que la obtención de las imágenes se realiza mediante un barrido de la escena a capturar, obteniendo en cada pasada una única línea de la imagen. Gracias a la rapidez de la adquisición de las imágenes es posible, por ejemplo, la captura y procesamiento de imágenes hiperespectrales en tiempo real.

Al igual que sucede con la cámara RGB es necesario calibrar la cámara hiperespectral, aunque los motivos son diferentes. Mientras que la cámara RGB se calibraba para lograr una fidelización del color óptima en las imágenes, la cámara hiperespectral necesita ser calibrada para poder medir la reflectancia de los diferentes materiales que componen los objetos a fotografiar.

Para poder realizar una correcta calibración es necesario utilizar un patrón de reflectancia difusa certificado, ver Fig. 3.7. Al tratarse de un patrón certificado nos aseguramos que refleja el 100 % de la radiación para cualquier longitud de onda de nuestra cámara.

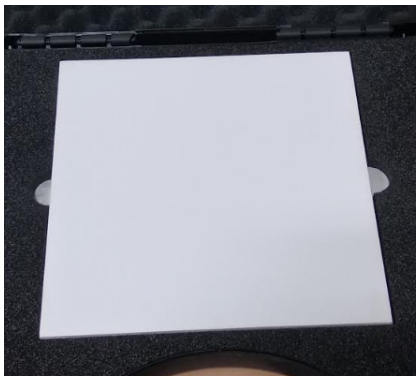

Figura 3.7: Patrón de reflectancia difuso

La calibración de la cámara hiperespectral Cuber UHD 285 se realiza con el software proporcionado por el fabricante, no siendo posible introducir este proceso en nuestro software específico. Los pasos a realizar para la calibración de la cámara son los siguientes:

- Inicializamos el software propietario de la cámara, Cube-pilot. La interfaz de este programa se muestra en la Fig3.8.

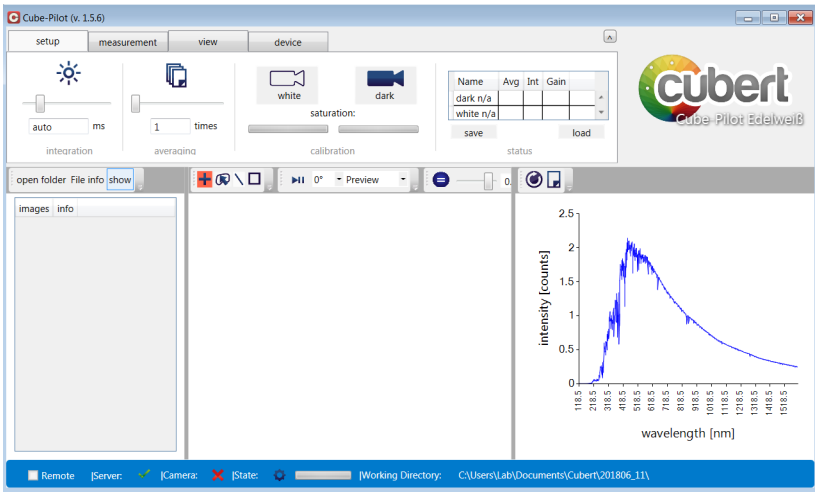

Figura 3.8: Interfaz programa Cube-Pilot

- Para un correcto funcionamiento de la cámara se utilizará un patrón de reflectancia difuso capaz de reflejar el 100 % de la radiación. Es necesario introducir el patrón dentro de la cabina de iluminación, como se muestra en la Fig. 3.9.

Figura 3.9: Colocación placa de calibrado para realizar la calibración de la cámara hiperespectral

- Una vez está colocado como se indica en el punto anterior, el siguiente paso con la cámara hiperespectral es, apuntando al patrón, realizar la medida. Para ello, realizaremos clic en el botón White como se muestra en la Fig.3.10.

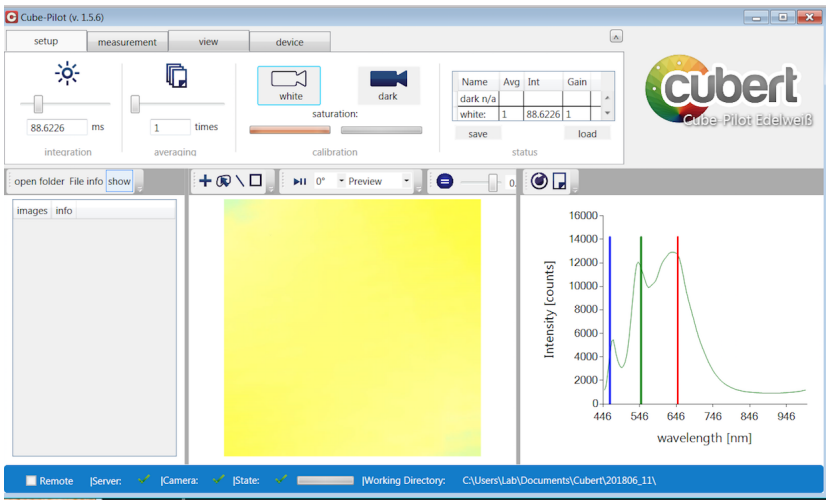

Figura 3.10: Medición intensidad blanco espectralmente neutro

- Para obtener una correcta calibración tenemos que minimizar el ruido introducido por los diferentes componentes hardware, sensores y componentes eléctricos que posee la cámara. Por lo tanto, para finalizar la calibración taparemos el objetivo de la cámara con un elemento opaco y procederemos a la medición de lo que se conoce como dark current, que indica la señal que recibe el sensor de la cámara aún sin llegarle luz alguna. Al finalizar se debe obtener un resultado similar al que se muestran en la Fig.3.11, donde se aprecia una línea completamente recta.

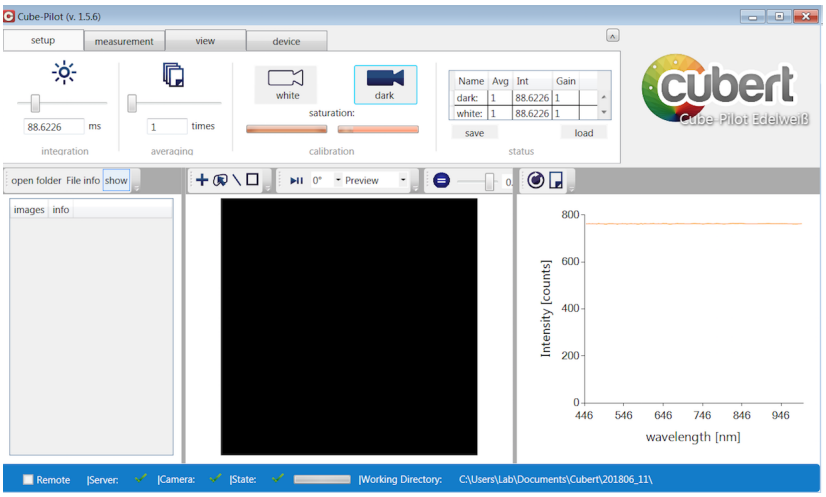

Figura 3.11: Medición intensidad negro

Con la cámara calibrada correctamente ya se pueden capturar imágenes hiperespectrales. Este proceso ya se ha automatizado en un script de Matlab que ha sido añadido a la aplicación que se ha desarrollado. El proceso de captura de imágenes consta de dos scripts: i) un script que permite la comunicación entre la cámara y nuestra aplicación, ii) un segundo script se encarga de indicar a la cámara las acciones que queremos realizar. La sencillez del primer script desarrollado, hace posible que no sea necesaria su explicación en profundidad, dejando su código en el material aportado en este TFM. Por otro lado, la complejidad del segundo script utilizado requiere de una explicación más profunda, que pasamos a detallar.

Listado de Código 3.4: Script para obtener imágnes hiperespectrales

```
function capturarHiperespectral(app,path)
    %Creamos un waitBar para que la aplicacion no de
    %sensacion de estar colgada.
    f = waitbar(0,'1','Name','Procesando Hiperespectral');
```

```
    %Obtenemos path original de la camara hiperespectral
    originalPath=client_send('localhost',3000,1,['<Cmd>
        GetPath</Cmd>']);
    %Indicamos a la camara el path donde queremos que se
    %almacenen las imagenes toamdas
    message=client_send('localhost',3000,1,['<Cmd>SetPath="'
        path '"</Cmd>']);
    split=strsplit(path,'\');
    fich=strcat(split(6),'_ImgHiper');
    waitbar(.15,f,'Obteniendo_Int_and_Avg');
    %obtenemos parametros esenciales para la captura de las
        imagenes
    Int = str2double(client_send('localhost',3000,1,['<Cmd>
        GetInt</Cmd>']));
    Avg = 1%str2double(client_send('localhost',3000,1,['<Cmd>
        GetAvg</Cmd>']));
    duration=((Int)*Avg)/1000;
    waitbar(.37,f,'Creando_.cue');
    %Capturamos el Cube, fichero que alamcena la informacion
    %hiperespectral de la imagen.
    message=client_send('localhost',3000,1,['<Cmd>CapCube="'
        char(fich) '"</Cmd>'],duration);
    client_send('localhost',3000,1,['<Cmd>GetPath</Cmd>']);
    waitbar(.67,f,'Exportando_.cue_a_Envi');
    waitbar(1,f,'Finanlizando');
    close(f);
end
```

Una vez ejecutado el script 3.4 se genera un fichero tipo CUE, que es necesario exportarlo a formato ENVI y posteriormente procesar para obtener una imagen hiperespectral por cada longitud de onda capturada por la cámara. De esta manera obtendremos tantas capas -imágenes hiperespectrales en formato PNG- como longitudes de ondas, en nuestro caso 138 capas.

3.1.3. Plataforma automática de captura de imágenes

Con la finalidad de automatizar el proceso de captura de las imágenes al máximo y que el usuario no tenga que intervenir en el giro de la fruta, lo que causaría posibles desajustes de los procesos de calibración, se ha desarrollado una plataforma giratoria que permite rotar la fruta los grados deseados. Para la ejecución de esta parte de la aplicación hemos basado en un proyecto desarrollado por empresa bq para la creación de un escáner 3d. Este escáner se encuentra bajo la licencia Creative Commons. Del proyecto anteriormente citado, hemos tomado los planos de la plataforma giratoria

para poder imprimirla en 3D y montarla. En la fig. 3.12 se muestra como ensamblar la plataforma. A continuación, se indican las piezas que componen la estructura anteriormente citada.

1. Motor paso a paso.
2. Soporte del motor.
3. M3 x 10 mm tornillo Allen.
4. Tuerca M3.
5. Acoplamiento de eje.
6. M8x30 mm tornillo Allen.
7. Cojinete.
8. Rodamientos de bolas 16014.
9. Soporte de disco.

Figura 3.12: Plataforma giratoria para escáner 3D

Los planos del soporte motor, item 2, y los correspondientes al soporte del disco, item 9, se pueden obtener de la siguiente dirección web:http://diwo.bq.com/en/ciclop-released/

Una vez ensamblada la plataforma giratoria, el siguiente paso es configurar el dispositivo hardware que la controla de manera remota. En este proyecto se ha utilizado una placa ARDUINO UNO que junto con CNC shield permite controlar un máximo de cuatro motores paso a paso. En la fig.3.13 se muestran los componentes hardware anteriormente citados.

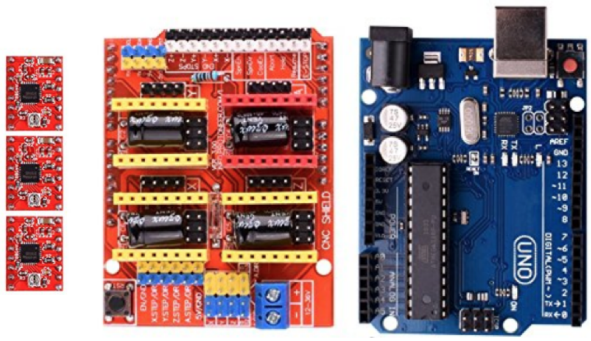

Figura 3.13: Arduino ONE + Shield CNC

El último componente hardware que compone la estructura de la plataforma giratoria es una fuente de alimentación, esta aporta los 12 voltios necesarios para lograr que la estructura gire.

Una vez finalizada la construcción de la plataforma giratoria, el siguiente paso es desarrollar el software que permita controlarla de manera remota. Para poder comunicar la placa Arduino con la plataforma giratoria es necesario la instalación de la librería GRBL dentro de la placa. Dicha librería se encuentra accesible en el repositorio GitHub:

https://github.com/Protoneer/GRBL-Arduino-Library/archive/master.zip. La carga de la librería requiere de los siguiente pasos:

1. Descomprimir la biblioteca y copiarla en la carpeta "Bibliotecas"donde instaló su software Arduino. P.ej. C:\arduino-1.0.3\libraries\
2. Cambiar el nombre de la carpeta a "GRBL". (Esto evitará que Arduino IDE muestre un aviso de error indicando que el nombre de la carpeta es demasiado largo)
3. Abre el Arduino IDE.
4. Haga clic en el menú: Archivo -> Ejemplos -> GRBL (o el nombre nuevo) -> Arduino UNO.

5. Subir la librería a la placa Arduino UNO

La integración del control de la plataforma giratoria, junto con los procesos de captura de imágenes, se ha llevado a cabo mediante un script en Matlab, que a través de un puerto COM del PC, posibilita la comunicación con la placa Arduino, de esta manera podemos indicar al motor paso a paso que gire un número determinado de grados. El script mostrado en 3.5, gestiona el movimiento del motor de la plataforma giratoria, que se unirá al proceso de captura de imágenes, tanto por la cámara RGB como por la cámara hiperespectral, para automatizar el proceso de generación del dataset de imágenes.

Listado de Código 3.5: Script para controlar la plataforma Arduino

```
classdef motor<matlab.mixin.SetGet
    properties(Access=public)
        com %Puerto al que se conecta el arduino.
        aceleracion %Aceleracion usada en la vuelta.
        grados %Grados que avanza el motor.
        conexion=0 %Indica si el motor esta conectado.
        arduino %Declaro el objeto arduino
    end
    methods (Access=public)
        function conectar(obj)
            try
                obj.arduino=serial(obj.com);
                set(obj.arduino,'BaudRate',115000);
                fopen(obj.arduino);
                pause(5);
                fprintf(obj.arduino,'$X');
                fprintf(obj.arduino,'$101=1');
                fprintf(obj.arduino,'$131=1');
                obj.conexion=1;
            catch ME
                obj.conexion=0;
            end
            return
        end
        function mover(obj)
            if obj.conexion==1
                orden=strcat('$121=',num2str(obj.aceleracion
                    ));
                fprintf(obj.arduino,orden);
                orden=strcat('G91 Y',num2str((obj.grados
                    *200)/360));
```

```
                fprintf(obj.arduino,orden);
            end
            return
        end
        function desconectar(obj)
            if obj.conexion==1
                obj.conexion=0;
                fclose(obj.arduino);
            end
            return
        end
    end

end
```

3.2. Análisis de imágenes hiperespectrales con CNN

La segunda fase de la metodología que se describe en este capítulo consta del análisis del dataset generado, para obtener los clasificadores marcados como objetivos del proyecto, un clasificador especializado en la detección de la variedad de la ciruela analizada, y otro nuevo clasificador especializado en obtener la fecha de maduración de la misma.

Como paso previo se ha generado el dataset utilizando las herramientas software específicas descritas en la sección anterior. En relación con las imágenes utilizadas, se han utilizado fotografías hiperespectrales de tres variedades de ciruelas: Black Splendor, OwenT y Angelino. Cada una de las variedades posee un ciclo de maduración propio, de esta manera nos encontramos con que las variedades tienen las siguientes maduraciones:

- Black Splendor: se trata de una ciruela con un ciclo de maduración temprana, alrededor de la mitad de Junio.
- OwenT: Su maduración y fecha de recolección corresponde con principios del mes de Julio.
- Angeleno: esta variedad tiene una maduración tardía, finales de Agosto, y su fecha de recolección corresponde con principios del mes de Septiembre.

En la Tabla 3.2 se detallan los dataset creados, el número de imágenes que compone cada uno y las características de cada una de las diferentes variedades usadas en este trabajo.

Cuadro 3.1: Características de las ciruelas

Dataset	Num. Imágenes	Fecha recolección	Semanas de maduración
MW1	121 x 138	9–13 Mayo	6 Angelino, 7 Owent y 7 BlackSplendor
MW2	147 x 138	23–27 Mayo	8 Angelino, 9 Owent y 9 BlackSplendor
MW3	127 x 138	13–17 Junio	11 Angelino, 12 Owent y 12 BlackSplendor
MW4	130 x 138	4–9 of Julio	14 Angelino, 15 Owent y 15 BlackSplendor
All_MW	525 x 138	-	-

Recordemos que el objetivo de esta fase de análisis es el desarrollo de un sistema inteligente capaz de clasificar imágenes hiperespectrales de ciruelas por la variedad a la que pertenezcan y el estado de maduración en el que se encuentren.

Entre los múltiples algoritmos que se pueden encontrar para crear un clasificador: existen algunos como: las redes neuronales convolucionales, los autocodificadores y las redes recurrentes (RNNs) que pueden resultar de gran ayuda a la hora de analizar imágenes. El gran potencial que tienen estas técnicas para el análisis de imágenes reside en su velocidad y eficacia. Para su correcto funcionamiento es necesario contar con una gran cantidad de datos y entrenarlas de forma correcta. Dichos algoritmos se pueden aplicar en tareas de clasificación de objetos dentro de una imagen. Debido a los buenos resultados obtenidos por las CNNs clasificando imágenes [18] se ha decidido utilizar una CNN conocida como Alexnet para crear los clasificadores anteriormente citados.

El primer paso a la hora de crear un clasificador basado en redes neuronales es la creación de un dataset con el que poder entrenarla. En el apartado anterior ya hemos abordado cómo obtener los ficheros tipo CUE, ahora toca procesarlos para obtener 138 imágenes por cada fichero correspondientes a las longitudes de onda con las que la cámara es capaz trabajar.

3.2.1. Procesamiento fichero tipo CUE

Como se ha explicado en apartados anteriores, cuando se realiza la captura de una imagen con la cámara hiperespectral y se exporta a formato envi se obtiene un fichero tipo CUE. Este fichero contiene toda la información hiperespectral de la imagen capturada. Es necesario procesar dicho fichero para obtener las imágenes PNG que se usarán para entrenar a la CNN.

Para procesar los ficheros tipo CUE se ha desarrollado un script en Matlab, ver código 3.6, que convierte dicho fichero en 138 imágenes PNG (una imagen por cada

longitud de onda que la cámara es capaz de capturar).

Listado de Código 3.6: Script procesar fichero tipo CUE

```
%Funcion que procesa un fichero tipo CUE
%Los parametros de entrada son el fichero y el path donde se
    encuentra.
function medida = procesarCUE( fich,path )
  originalPath=client_send('localhost',3000,1,['<Cmd>GetPath
     </Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>SetPath="'
     path '"</Cmd>']);
  client_send('localhost',3000,1,['<Cmd>GetPath</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>Load="' char
     (strcat(path,fich)) '"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>SetPanScale
     =1</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>
     SetExportFirstgray="1"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>ExportEnvi="
     ' char(strcat(path,fich)) '"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>SetPanScale
     ="20"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>
     SetExportFirstgray="1"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>ExportEnvi="
     ' char(strcat(path,fich)) '"</Cmd>']);
  message=client_send('localhost',3000,1,['<Cmd>SetPath="'
     originalPath '"</Cmd>']);

  tamBloq = 1000;
  notOpened = 1;
  cont = 1;

  while notOpened && cont < 3
    try
      X = multibandread(char(strcat(path,fich,'.cue')),[
          tamBloq,tamBloq,138],'uint16',0,'bsq','ieee-le');
      notOpened = 0;
    catch e
      pause(3);
      cont = cont + 1;
    end
  end

  for j = 1:138
```

```
    aux=X(:,:,j)./10000;
    imwrite(aux, char(strcat(path,int2str(j),'.png')), 'png'
        );
  end
end
```

La salida proporcionada por el script al procesar un fichero tipo CUE son 138 imágenes PNG del mismo fichero envi, que se corresponden con una única pieza de fruta. En la Fig.3.14 se muestra un ejemplo de algunas de las imágenes obtenidas, como se puede observar las imágenes son muy diferentes entre sí, ya que proporcionan información hiperespectral diferente, para una misma pieza de fruta.

Figura 3.14: Ejemplo imágenes procesadas fichero tipo CUE

3.2.2. Configuración estructura CNN

Una nueva técnica ha surgido en los últimos años con fuerza en el campo del aprendizaje profundo[17], se trata de las CNNs [8]. Estas redes basan su funcionamiento en un aprendizaje jerarquizado en el cual estructuras de alto nivel son construidas de manera automática a partir de estructuras de más bajo nivel, llamadas capas, comenzado por los datos sin procesar, los píxeles de una imagen.

El auge de este tipo de CNNs ha permitido el desarrollo de multitud de redes de propósito general o redes especializadas en el tratamiento de imágenes. Tal es el caso de Alexnet, una red que está compuesta por cinco capas convolucionales y tres capas fully conect, esta estructura se puede observar de manera gráfica en la Fig. 3.15. Su funcionamiento y estructura completa se describe en detalle en [8]. La diferencia entre las capas convoluciones y el resto de capas reside en las conexiones entre sus neuronas y las neuronas que componen la capa anterior.

Entre las capas de la red nos podemos encontrar con las capas de normalización, donde su misión es promover la competencia entre grupos cercanos de neuronas, disminuyendo las respuestas que son uniformemente grandes en el vecindario

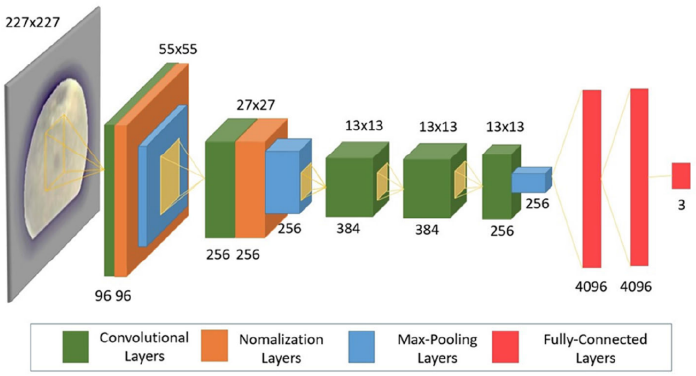

Figura 3.15: Estructura Alexnet

y aumentado las respuestas en secciones más pequeñas de la imagen. Las capas de Max-pooling son las encargadas de realizar una reducción de las imágenes obtenidas por las capas anteriores, normalmente capas de normalización cuya única función es la de normalizar los valores obtenidos en las capas de convolución. La Tabla 3.2 resume el tamaño del filtro, el número de filtros y stride para cada capa.

Cuadro 3.2: Parametros Alexnet para las capas de convolución y Pooling

Capa	Convolución			Pooling	
	Tam. Filtro	Num. Filtro	Stride	Tam. Filtro	Stride
Primera	11 x 11	96	4	3 x 3	2
Segunda	5 x 5	256	1	3 x 3	2
Tercera	3 x3	384	1	-	-
Cuarta	3 x 3	384	1	-	-
Quinta	3 x 3	256	1	3 x 3	2

El entrenamiento de este tipo de redes requiere de una estructura muy específica que posibilita todo el proceso de aprendizaje de la red. En el trabajo aquí presentado se ha utilizado el framework llamado Caffe [19], que nos permite, a través de una estructura fija en disco, lanzar el proceso de aprendizaje requerido por la red. A continuación, se describen los scripts necesarios para llevar a cabo dicho proceso.

Todo proceso estocástico, como el que nos compete en este trabajo, requiere realizar el mismo proceso de aprendizaje un determinado número de veces, para poder consolidar los resultados obtenidos por las diferentes ejecuciones. Para ello, en este trabajo hemos utilizado la técnica llamada K-fold cross validation, donde $K = 5$. Esta técnica nos obliga a dividir el conjunto completo de imágenes en 5 particiones, donde usará un conjunto compuesto por 4 de estas particiones para

entrenar a la red y 1 para validar los resultados. El proceso debe repite tantas veces como subconjuntos se hayan realizado, en nuestro caso 5. Posteriormente, con cada subconjunto se repite nuevamente un total de 6 veces el ajuste de la red, utilizando el mismo conjunto de datos de entrenamiento y validación, obteniendo así un total de 30 ejecuciones, lo que nos permite consolidar los resultados que arroje la red.

Como se ha comentado anteriormente, se ha utilizado el framework de trabajo Caffe, donde se pueden incorporar una serie de CNNs, entre ellas Alexnet. Para llevar a cabo el proceso de aprendizaje de la red es necesario crear una serie de directorios, los cuales tienen la siguiente estructura:

- Dir caffe_models: Este directorio contiene dos ficheros, el modelo y el configurador, estos ficheros son los encargados de almacenar tanto la estructura como la configuración de la CNN a utilizar.
  - model: El modelo define la estructura de una red neuronal, se tienen que realizar los cambios que se muestran el código 3.7.

Listado de Código 3.7: Configuración modelo CNN caffe

```
...
%Linea 14: indicar donde va a estar el fichero mean.
   binaryproto
 mean_file: "../fruta_alexnet/input77/mean.
    binaryproto"
...
%Linea 17: indicar donde va a estar la BBDD para
   entrenar .mlbd
source: "../fruta_alexnet/input77/train_lmdb"
...
%Linea 32: indicar donde va a estar el fichero mean.
   binaryproto
 mean_file: "../fruta_alexnet/input77/mean.
    binaryproto"
...
%Linea 35: indicar donde va a estar la BBDD para
   validar .mlbd
source: "../fruta_alexnet/input77/validation_lmdb"
...
%Configurar las capas de salida de la red, tenemos
   que tener una capa por cada clase que queramos
   clasificar.
%Linea 357
num_output: 3
```

- configurador: El solucionador es responsable de la optimización del modelo. Definimos los parámetros del solver en un archivo de configuración con la extensión *prototxt*. 3.8.

Listado de Código 3.8: Configuración solver CNN caffe

```
net: "../caffe_models/AlexNET.prototxt"
test_iter: 8
test_interval: 6
base_lr: 0.01
lr_policy: "step"
gamma: 0.1
stepsize: 60
display: 1
max_iter: 301
momentum: 0.9
weight_decay: 0.0001
snapshot: 301
snapshot_prefix: "../caffe_models"
solver_mode: GPU
solver_type: SGD
```

- Dir Master_BBDD: En este directorio se generan de manera automática varias bases de datos (BBDD) lmbd con las imágenes a ejecutar el script que ejecuta la CNN. Los ficheros contenidos dentro del Dir Master_BBDD son los siguientes:

  - train_lmdb:
    data.mdb
    lock.mdb
  - val_lmdb:
    data.mdb
    lock.mdb
  - Imagen media de los datos: Se resta la imagen media de cada imagen de entrada para asegurar que cada píxel característico tiene media cero. Este es un paso de pre-procesamiento común en la máquina de aprendizaje supervisado.

- Dir resultados: En este directorio se almacenan todos los resultados obtenidos al lanzar las diferentes ejecuciones de la CNN. Se almacenarán 138 carpetas, una por cada longitud de onda diferente que es capaz de captar nuestra cámara hiperespectral, y a su vez cada directorio contendrá 5 directorios más, uno por cada combinación del k-fold cross validation, con $k = 5$.

- Fichero lanzarCNN.py: Este fichero escrito en Python realiza varias tareas: genera las particiones para el cross validation, crea las BBDD lmbd, genera la

imagen media y lanza todas las ejecuciones necesarias para el aprendizaje de la CNN. A continuación, se explica en detalle el script:

- Generar particiones cross-validation. Código 3.9

Listado de Código 3.9: LanzarCNN.py creación cross-validation

```
def crossValidation(path):
 #Obtenemos el numero de imagenes de la sesion
 num_img=len(os.listdir(path+ '/2'))
 #Creamos un fichero con cinco grupos, en cada grupo
     estan las img
 #Obtenemos todos los ficheros del directorio'''
 img= os.listdir(path+ '/2')[:]
 grupo=[list(),list(),list(),list(),list()]
 for i in range(0,len(img)):
         rnd=random.randrange(len(img))
          if len(img)%5==0:
                  grupo[0].append(img[rnd])
          if len(img)%5==1:
                  grupo[1].append(img[rnd])
          if len(img)%5==2:
                  grupo[2].append(img[rnd])
          if len(img)%5==3:
                  grupo[3].append(img[rnd])
          if len(img)%5==4:
                  grupo[4].append(img[rnd])
          del img[rnd]
 #Creamos el fichero del crossValidation dentro de
    la carpeta Master_BBDD
 if not os.path.exists(path_raiz+"/Master_BBDD"):
         os.system("mkdir "+path_raiz+"/Master_BBDD")
 try:
         f=open('Master_BBDD/crossValidation.txt','w'
             )
         for i in range (0,5):
         f.write("###Grupo_"+str(i)+"\n")
         '''Guardamos las img de cada grupo'''
         for j in range (0,len(grupo[i])):
                 '''solo nos interesa el nombre del
                    fichero no la capa ni el formato
                    '''
                 nombre=grupo[i][j].split("_")
         f.write(str(nombre[0])+"_"+str(nombre[1])+"_
             "+str(nombre[2])+"_"+str(nombre[3])+"_\n"
             )
         f.close()
```

```
except ValueError:
        print ("Error al crear el fichero
            crossValidation")
```

- Generar BBDD lmdm y creación de la imagen media. Código 3.10

Listado de Código 3.10: LanzarCNN.py creación lmdb

```
def create_lmdb(num_layer):
 if os.path.exists(path_raiz+'/input'+str(num_layer)
    +'/train_lmdb'):
        os.system("rm -r "+path_raiz+'/input'+str(
            num_layer)+'/train_lmdb')
 if os.path.exists(path_raiz+'/input'+str(num_layer)
    +'/validation_lmdb'):
        os.system("rm -r "+path_raiz+'/input'+str(
            num_layer)+'/validation_lmdb')
 os.system("mkdir "+path_raiz+'/input'+str(num_layer
    )+'/train_lmdb')
 os.system("mkdir "+path_raiz+'/input'+str(num_layer
    )+'/validation_lmdb')
 train_lmdb = path_raiz+'/input'+str(num_layer)+'/
    train_lmdb'
 validation_lmdb = path_raiz+'/input'+str(num_layer)
    +'/validation_lmdb'
#os.system('rm -rf  ' + train_lmdb)
#os.system('rm -rf  ' + validation_lmdb)
 train_data = [img for img in glob.glob("./input"+
    str(num_layer)+"/train/*png")]
 test_data = [img for img in glob.glob("./input"+str
    (num_layer)+"/test/*png")]

 #Shuffle train_data
 random.shuffle(train_data)

 print 'Creating train_lmdb'
 in_db = lmdb.open(train_lmdb, map_size=int(1e12),
    subdir=True, create=True)
 with in_db.begin(write=True) as in_txn:
  for in_idx, img_path in enumerate(train_data):
        img = cv2.imread(img_path, 0)
        img = transform_img(img, img_width=
            IMAGE_WIDTH, img_height=IMAGE_HEIGHT)
        if 'angelino' in img_path:
                label = 0
        if 'blacksplendor' in img_path:
```

```
                label = 1
    if 'owent' in img_path:
                label = 2
        datum = make_datum(img, label)
                    in_txn.put('{:0>5d}'.format(in_idx),
                        datum.SerializeToString())
in_db.close()

print '\nCreating validation_lmdb'

in_db = lmdb.open(validation_lmdb, map_size=int(1
    e12))
with in_db.begin(write=True) as in_txn:
 for in_idx, img_path in enumerate(test_data):
        img = cv2.imread(img_path, 0)
        img = transform_img(img, img_width=
            IMAGE_WIDTH, img_height=IMAGE_HEIGHT)
        if 'angelino' in img_path:
                label = 0
        if 'blacksplendor' in img_path:
                label = 1
    if 'owent' in img_path:
                label = 2
        datum = make_datum(img, label)
                    in_txn.put('{:0>5d}'.format(in_idx),
                        datum.SerializeToString())

in_db.close()
        #Creamos la img media
        os.system("../src/caffe/build/tools/
            compute_image_mean -backend=lmdb "+
            path_raiz+"/input"+str(num_layer)+"/
            train_lmdb "+path_raiz+"/input"+str(
            num_layer)+"/mean.binaryproto")
```

- Lanzar la CNN. Esta sección del código genera la estructura de la red, creando un prototype y solver especifico en función de la capa a procesar y el cross-validation configurado. Código 3.11

Listado de Código 3.11: LanzarCNN.py launch CNN

```
def launch_CNN(num_layer,combinacion):
 #Creamos la carpeta resultado
 path_resultado=path_raiz+'/resultado/capa'+str(
    num_layer)+'/'
 if not os.path.exists(path_resultado):
```

```
        os.mkdir(path_resultado)
    os.system("mkdir -p "+path_resultado+str(
        combinacion))
#generamos un solver diferente para cada una de las
    ejecuciones
for l in range(0,3):
        #leemos el solver estandar y generamos 3
            solver para ser resultos
    try:
        f=open('caffe_models/solver_AlexNET.prototxt
            ','r')
        cont=-1
        contenido="\n"
        cont=0
        for line in f:
          if cont==12:
            contenido=contenido+"snapshot_prefix: "
                \"../fruta_alexnet_variedad/resultado
                /capa"+str(num_layer)+"/"+str(
                combinacion)+"\"\n"
          elif cont==0:
             contenido=contenido+"net: \"../
                fruta_alexnet_variedad/caffe_models/
                AlexNET_cap_"+str(num_layer)+"_"+str
                (l)+".prototxt\"\n"
          else:
            contenido=contenido+line
          cont=cont+1
        f.close()
    #Guardo el nuevo solver
        f=open('caffe_models/solver_AlexNET_cap_'+
            str(num_layer)+'_'+str(l)+'.prototxt','w'
            )
        f.write(contenido)
        f.close()
    except ValueError:
        print ("Error al crear el solver_AlexNET
            capa:"+str(num_layer))
#abrimos La Red y generamos una red diferente para
   cada ejecucion
    try:
        f=open('caffe_models/AlexNET.prototxt','r')
        cont=0
        contenido=""
        for line in f:
```

```
            if cont==13 or cont==31:
              contenido=contenido+"    mean_file: \"../fruta_alexnet_variedad/input"+
                 str(num_layer)+"/mean.binaryproto\" \
                 n"
            elif cont==16:
              contenido=contenido+"    source: \"../
                 fruta_alexnet_variedad/input"+str(
                 num_layer)+"/train_lmdb\"\n"
            elif cont==34:
              contenido=contenido+"    source: \"../
                 fruta_alexnet_variedad/input"+str(
                 num_layer)+"/validation_lmdb\"\n"
            else:
              contenido=contenido+line
            cont=cont+1
          f.close()
          #Guardo el nuevo solver
          f=open('caffe_models/AlexNET_cap_'+str(
             num_layer)+'_'+str(l)+'.prototxt','w')
          f.write(contenido)
          f.close()
      except ValueError:
        print ("Error al crear la red AlexNET capa:"+
           str(num_layer))
#Lanzamos las CNN en las diferentes GPU
t1 = threading.Thread(target=worker, args=(
   num_layer,combinacion,0,path_resultado+str(
   combinacion),0))
t2 = threading.Thread(target=worker, args=(
   num_layer,combinacion,1,path_resultado+str(
   combinacion),1))
t3 = threading.Thread(target=worker, args=(
   num_layer,combinacion,2,path_resultado+str(
   combinacion),0))
t1.start()
t2.start()
t1.join()
t2.join()
t3.start()
t3.join()
#Borramos los archivos alexnet y sus solver
for l in range(0,3):
        os.system('rm -r caffe_models/
           solver_AlexNET_cap_'+str(num_layer)+'_'+
```

```
            str(l)+'.prototxt')
    os.system('rm -r caffe_models/AlexNET_cap_'+str(
        num_layer)+'_'+str(l)+'.prototxt')
    os.system("rm "+path_resultado+"/*.caffemodel")
    os.system("rm "+path_resultado+"/*.solverstate")
```

- Script generarDatos.py: Este script recorre de forma automática la capeta resultados y aplica sobre cada resultado obtenido el script plot_learning_curve.py.
- Script plot_learning_curve.py. Este script genera una imagen PNG, Fig 3.16, con la salida de la red. También nos genera un fichero con el Log de cada red donde guarda los valores finales de la red.

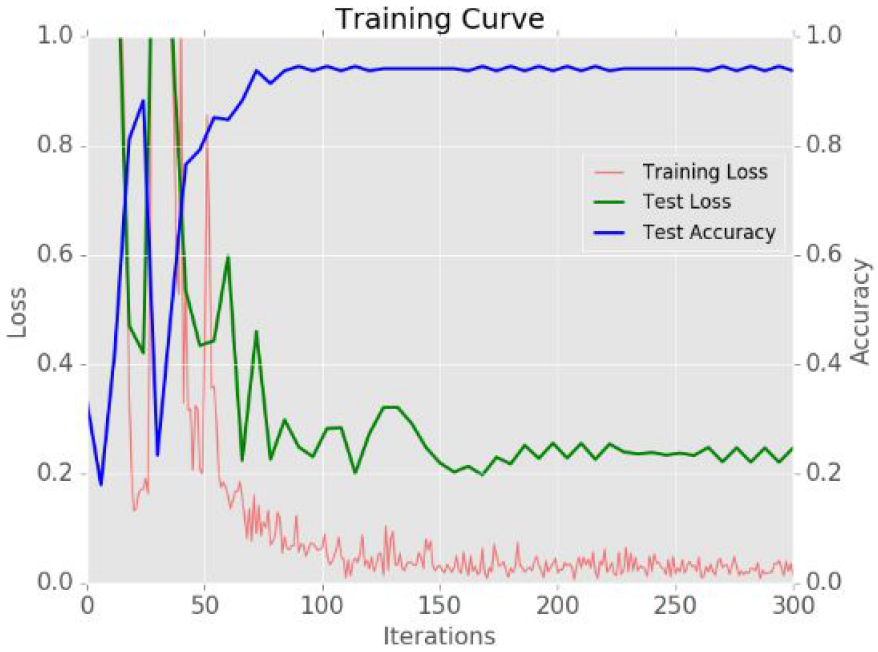

Figura 3.16: Ejemplo resultado script plot_learning_curve.py

Capítulo 4

Resultados

Entre los objetivos citados en el capítulo 2 se encuentra el análisis de la información hiperespectral de las imágenes de fruta, para obtener las longitudes de onda más prometedoras para su clasificación por variedad y madurez. Esto nos permitirá desechar gran cantidad de información hiperespectral y centrarnos en las bandas más prometedoras, que como se puede ver en esta sección, ofrecen resultados muy robustos en relación con la clasificación de la variedad de la fruta, así como su maduración.

En este apartado se presentan los resultados obtenidos por los diferentes clasificadores que se han optimizado utilizando Alexnet como CNN base en nuestro estudio. El objetivo era obtener clasificadores por variedad y/o por madurez, tales que, se puedan detectar las longitudes de ondas más prometedoras. Las variedades con las que se ha trabajado en este estudio han sido:

- Black Splendor
- Angeleno
- OwenT

Por otro lado, las fechas de maduración seleccionadas se presentan en la Tabla 3.2.

El trabajo que se ha llevado a cabo ha constado de la optimización de 138 clasificadores diferentes, ya que cada uno de ellos se debía especializar en una de las bandas obtenidas de las imágenes tomadas con la cámara hiperespectral. Para la optimización de este trabajo se ha utilizado un equipo compuesto por una tarjeta gráfica Tesla-k20 de la marca Nvidia. Dicha gráfica cuenta con 2496 núcleos cuda y una memoria GDDR5 de 5GB que le permiten ejecutar operaciones en coma flotante de doble precisión a una velocidad de 1.17 Tflops y operaciones en coma flotante de precisión simple a 3.52 Tflops.

A continuación, se presentan los resultados obtenidos por los clasificadores especializados en la variedad de la fruta, así como los especializados en la maduración. En ambos estudios se han utilizados los parámetros que se muestran en la Tabla. 4 para configurar la CNN Alexnet y los valores mostrados se corresponde con la media obtenida en cada una de las diferentes longitudes de onda del espectro.

Cuadro 4.1: Parámetros configuración CNN

Nombre Parámetro	Valor
base_lr	0.001
gamma	0.1
stepsize	60
max_iter	301
momentum	0.9
weight_decay	0.0001
snapshot	301
test_iter	8
test_interval	6
lr_policy	"step"
solver_mode	GPU
solver_type	SGD

4.1. Resultados clasificación por variedad

En esta subsección se describen e interpretan los datos obtenidos por los 138 clasificadores de las tres variedades diferentes de ciruelas utilizadas, todas ellas agrupadas según los datos mostrados en la Tabla 3.2. La idea es analizar, si en diferentes estados de maduración, podemos encontrar diferentes longitudes de onda que ayuden a la clasificación por variedad de las ciruelas. Se presenta un conjunto de gráficas que resumen el resultado de estos clasificadores, así como se puede observar en cada uno de ellos la presencia de ciertas bandas espectrales que obtienen excelentes resultados de clasificación.

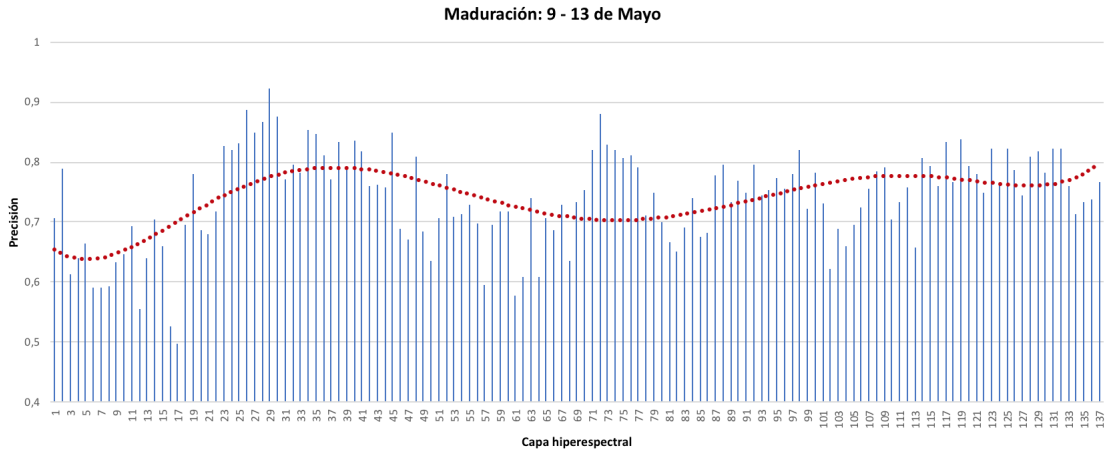

Figura 4.1: Resultado valor medio en clasificación por variedad Dataset MW1

Los resultados de la Fig. 4.1 se corresponden con la semana de maduración 9 - 13 de Mayo. En esta fecha todos los frutos se encuentran en una etapa muy temprana de su ciclo de maduración. Esto tiene como consecuencia que las ciruelas se encuentran poco desarrolladas y el parecido entre ellas es elevado, independiente de la variedad de las mismas. Como puede observarse en la figura, la línea de tendencia de color granate nos indica que entre las capas 25 y 35, siendo la mejor de ellas la capa número 29 con una precisión de un 96 %, se muestra una tendencia de mayor precisión en la clasificación.

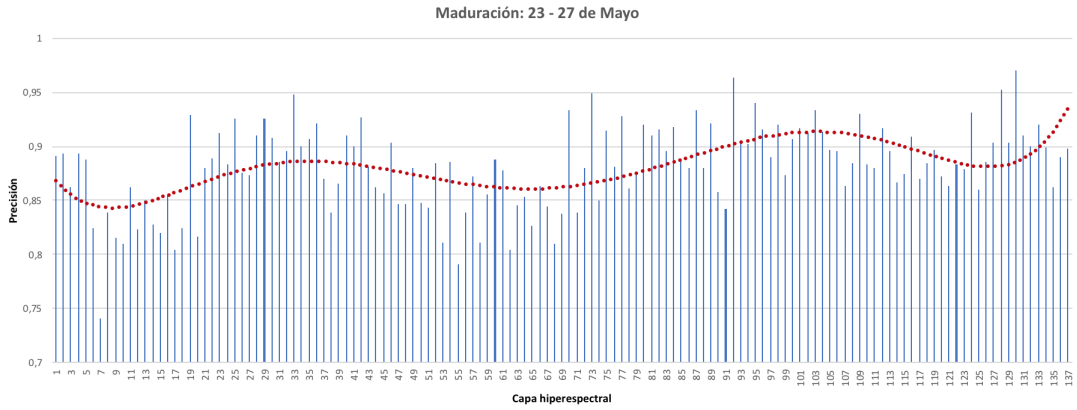

Figura 4.2: Resultado valor medio en clasificación por variedad Dataset MW2

Al clasificar las ciruelas por variedad utilizando el dataset MW2, 23 - 17 de Mayo, se obtienen mejores resultados que con el dataset MW1. Esto es debido a que las frutas, van avanzando en su proceso de maduración, lo que hace que se encuentren más desarrolladas y por tanto las diferencias entre variedades empiezan a ser palpables. Como puede observarse, atendiendo nuevamente a la línea de tendencia que se genera presenta en la Fig. 4.2, las últimas capas, entre las 129 y 137, ofrecen los mejores resultados de clasificación por variedad, teniendo en cuenta que todas las capas se obtienen una precisión mayor al 75 %. En este proceso de optimización del clasificador por variedad, se ha alcanzado el óptimo en la capa 130 con un 96 % de aciertos.

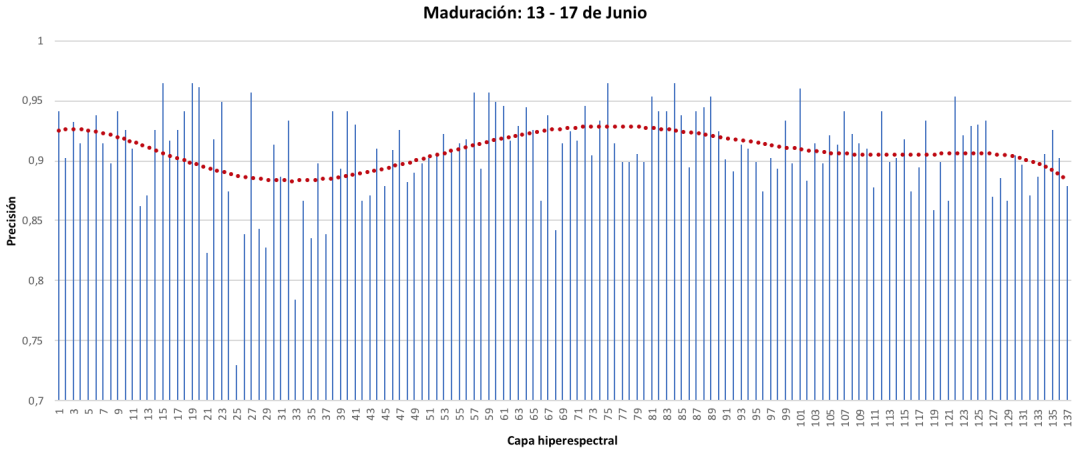

Figura 4.3: Resultado valor medio en clasificación por variedad Dataset MW3

A medida de avanzamos en las semanas de maduración, dataset MW3, 13 - 17 Junio, el clasificador mejora su precisión en todas sus capas respecto a los resultados obtenidos con los dataset anteriores. Casi todas las capas hiperespectrales del dataset MW3 obtienes una precisión superior al 75 %. La línea de tendencia dibujada en rojo sobre la Fig. 4.3 nos ayuda a confirmarlo. A diferencia de los dataset anteriores, en este se encuentran varias capas con una precisión superior al 96 %.

Los resultados obtenidos con el último dataset utilizado, MW4 4 - 9 Julio, aportan buenos resultados globales de todas las capas en cuanto a precisión media para clasificar las variedades de ciruelo estudiadas. En esta última fase de maduración las diferencias entre las variedades son notorias a simple vista y esto permite que su clasificación sea más sencilla, es por ese motivo por el cual los resultados globales obtenidos son mejores. La línea de tendencia presente en la gráfica Fig. 4.4 muestra que todas las capas obtienen buenos datos, estando la mayoría de las capas por encima del 80 %. El óptimo se encuentra en la capa 55 con un valor de precisión del 100 % en la clasificación por variedad.

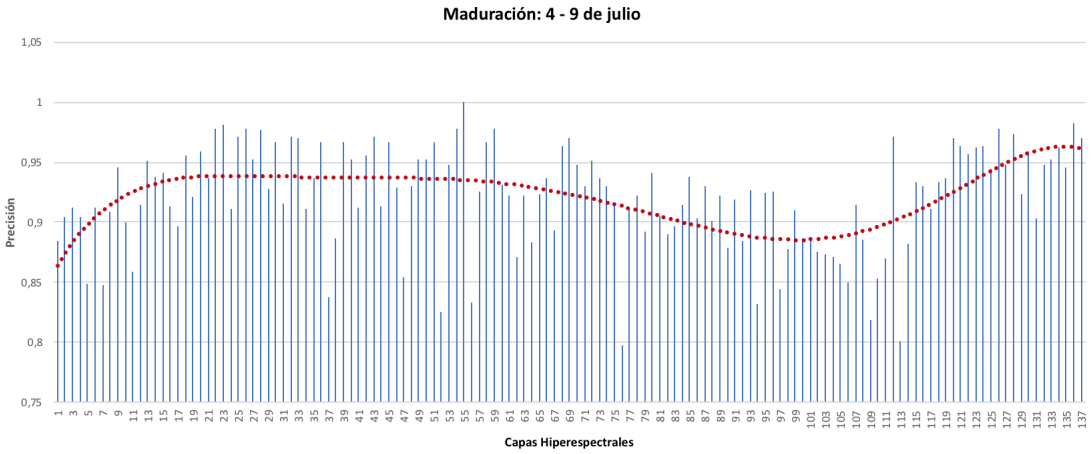

Figura 4.4: Resultado valor medio enclasificación por variedad Dataset MW4

Hasta aquí, podemos observar que a medida que la fruta madura, es más sencillo detectar su variedad, obteniendo un conjunto de capas hiperespectrales con mayor precisión. Estas características también pueden deberse a las diferentes fases de maduración que tienen las variedades que se han utilizado en este estudio. La variedad Angeleno es la que tiene mayor ciclo de maduración, de ahí que la fruta presenta menos cambios a lo largo del periodo estudiado, pero el resto si que presenta cambios, lo que permite a la red clasificar mejor las variedades. La Fig. 4.5 muestra un ejemplo de cómo evolucionan las variedades estudiadas de diferente forma.

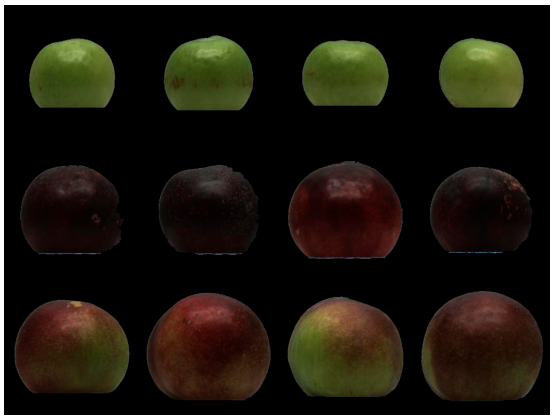

Figura 4.5: Evolución de maduración de las variedades de ciruela utilizadas

En el estudio que se detalla a continuación, se han mezclado todas las fases de maduración de las diferentes variedades, para poder desestacionar la componente del tiempo de maduración. Con esto se pretende no facilitar a la red la detección gracias a grandes cambios en ciertas variedades y cambios pocos significativos en las variedades de ciclo largo. La Fig. 4.6 muestra los resultados obtenidos de este nuevo estudio.

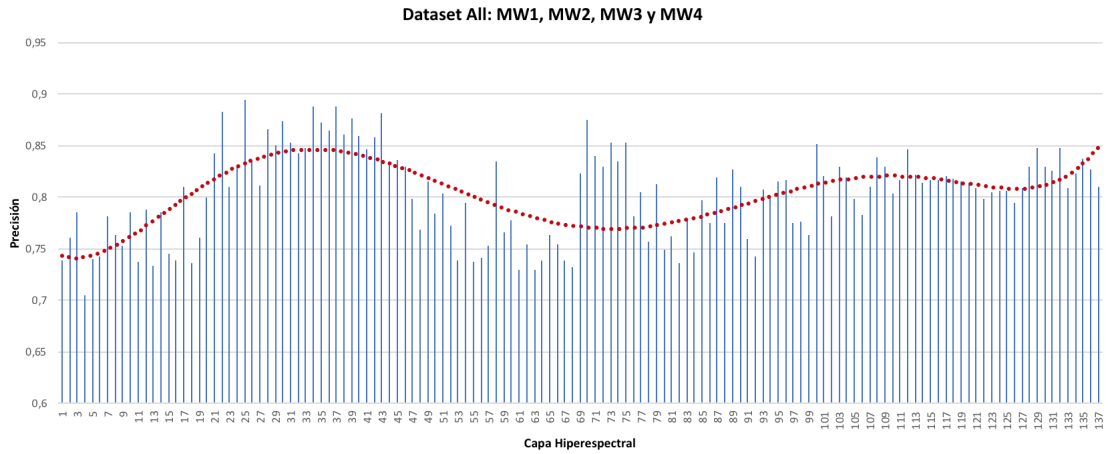

Figura 4.6: Resultado valor medio en clasificación por variedad Dataset ALL

El dataset utilizado donde la fecha de maduración del fruto no es relevante, arroja resultados del 70 % de precisión como mínimo. Si observamos la línea de tendencia presente en la Fig. 4.6 se aprecia que entre las capas 17 - 47 se obtienen muy buenos resultados, siendo la mejor capa la 25 con un 85 % de precisión. Este dato nos indica que es posible la clasificación de ciruelas por su variedad, gracias a la precisión obtenida del 85 %, siendo indiferente la fase de maduración en la que se encuentre el fruto.

La Tabla 4.1 muestra un resumen de las propuestas estudiadas con los diferentes dataset empleados. En esta tabla se resumen las capas que han obtenido mayores resultados, así como el intervalo de capas donde se aprecian diferencias con respecto al resto de capas.

Variedad	Num. Imágenes
Angeleno	219 x 138
Black-Splendor	171 x 138
OwenT	141 x 138

Cuadro 4.2: Proposición capas para clasificador

Semanas de maduración	DataSet	Capas mejor precisión	Rango de capas
9–13 Mayo	MW1	29	23 - 49
23–27 Mayo	MW2	92, 130	91 - 135
13–17 Junio	MW3	15, 19, 75, 84, 101	13 - 21 / 72 - 108
4–9 of Julio	MW3	55	115 - 137
-	All_MW	25	19 - 47

Podemos observar, según los datos presentados, que a diferentes fases de maduración de los frutos, la información relevante para poder clasificar las variedades según las imágenes hiperespectrales, se encuentran en diferentes bandas del espectro.

Por otro lado, también podemos observar que los resultados son excelentes, si nos centramos en la capacidad de clasificación de las redes optimizadas, ya que, independientemente de las capas, es posible generar una buena clasificación de las variedades empleadas.

Si comparamos los datos obtenidos en el estudio que aquí se presenta, con los datos obtenidos en estudios similares [20], donde su utilizaron únicamente imágenes RGB, podemos observar ciertas diferencias. Anteriormente se ha abordado la problemática de la clasificación de ciruelas por su variedad mediante el uso de la misma CNN -Alexnet- pero con la diferencia que, en este trabajo, las imágenes utilizadas eran imágenes RGB. Los experimentos que se realizaron en el estudio citado con anterioridad se muestran en la Tabla 4.1

Cuadro 4.3: Datos obtenidos y parámetros usados en el estudio de clasificación por variedad imágenes RGB [20]

Dataset	RGB	Parámetros
MW1	0.8960 ± 0.010	Época: 1.000 Learning rate: 0.001 Imagen: negro
MW2	0.9299 ± 0.015	Época: 1.000 Learning rate: 0.005 Imagen: negro
MW3	0.9739 ± 0.008	Época: 750 Learning rate: 0.01 Imagen: region
MW4	0.9674 ± 0.005	Época: 1.000 Learning rate: 0.005 Imagen: negro
All_MW	0.9071 ± 0.010	Época: 1.000 Learning rate: 0.01 Imagen: negro

Los datos de comparación de ambos estudios se presentan en la Tabla 4.1.

Cuadro 4.4: Resumen de resultados utilizando imágenes RGB frente a imágenes hiperespectrales

Dataset	RGB	Hiperespectral - Número capa
MW1	0.8960 ± 0.010	0.9232 ± 0.060 - 29
MW2	0.9299 ± 0.015	0.9695 ± 0.040 - 130
MW3	0.9739 ± 0.008	0.9648 ± 0.030 - 19
MW4	0.9674 ± 0.005	1.0000 ± 0.010 - 55
All_MW	0.9071 ± 0.010	0.8945 ± 0.030 - 25

Para estudiar si existen diferencias significativas entre ambos estudios, se ha utilizado el test de Wilcoxon. Se trata de un test no paramétrico para comprobar el rango medio de dos muestras relacionadas y determinar si existen diferencias significativas entre ellas. La hipótesis nula para La prueba de Wilcoxon es $H0 := 0\theta D$ la diferencia entre pares de observaciones sigue una distribución con mediana cero. La alternativa hipótesis es $H1 : \theta D \neq 0$. Para realizar la prueba, clasificamos las diferencias, en valor absoluto, entre las puntuaciones de rendimiento del dos algoritmos para cada caso. Entonces, calculamos R+ como la suma de rangos para las instancias en las que el primer algoritmo supera al primero la segunda y R- como la suma de los rangos para el caso inverso. Si $T = min(R^+, R^-)$ es inferior o igual al valor de la distribución de Wilcoxon para N (tamaño de las muestras) grados de libertad, el nulo se rechaza la hipótesis de la igualdad de medios. En particular, si R^+ es mayor que R^- y R^- es menor o igual que el valor crítico, el valor del primer algoritmo supera al segundo. Por el contrario, si R^+ es menor que R^- y menor o igual que el valor crítico, el segundo supera al primero. De lo contrario, la prueba no encontrar diferencias entre el rendimiento de los algoritmos. Los resultados obtenidos se muestran en la Tabla 4.1.

Cuadro 4.5: Resultados obtenidos utilizando el test de Wilcoxon

VS	R^+	R^-	P-value	P-value Asintótico
Hiper	8.0	7.0	≥ 0.2	0.787406

Como puede observarse, al comparar los datos obtenemos un valor de P-value igual al 0.2 esto significa que no existen diferencias significativas entre los experimentos, clasificación de ciruelas por su variedad usado RGB o hiperespectral. Para que existieran diferencias significativas en el test de Wilcoxon con un tamaño de 5 dataset el experimento A tendría que superar al experimento B en las 5 comparaciones realizadas, ya que el valor crítico de test de Wilcoxon para N=5 es 5. Aun así, el uso de imágenes hiperespectrales para realizar una clasificación por variedad nos aporta notables ventajas. La primera ventaja que nos aporta la utilización de imágenes hiperespectrales es la posibilidad de entrenar a la CNN en menos tiempo, esto es debido a que solo han sido necesarios 300 épocas para obtener resultados

similares al estudio con imágenes RGB, donde se necesitaron entre 750 y 1000 épocas. La otra ventaja que aporta la utilización de imágenes hiperespectrales es que, a diferencia de las RGB, se obtiene un rango de capas con buenos resultados, de esta manera obtenemos un dataset con el que trabajar y del que mediante combinación de longitudes de ondas se podrían obtener clasificadores con más precisión.

Se puede afirmar que el estudio de la imagen hiperespectral, al contener propiedades físico-químicas del fruto, ofrece mejores resultados frente a una clasificación por imagen convencional, donde únicamente la forma, tamaño y color serían las componentes que intervienen en la clasificación.

4.2. Resultados clasificación por estado de maduración

De igual forma que con los resultados presentados en la sección anterior, hemos obtenido buenos resultados de los clasificadores que se centran en la variedad de la fruta analizada, en esta sección se presenta un nuevo estudio donde se optimizan clasificadores orientados a determinar la fecha de maduración de la misma.

En este caso, los dataset con los que se ha trabajado están agrupados por variedad, utilizando 4 clases diferentes, los momentos de maduración de cuando fueron recogidos los frutos. Se utilizan para este estudio nuevamente las imágenes hiperespectrales, donde se intenta encontrar un conjunto de bandas que nos permitan clasificar las frutas por su estado de maduración.

Las Figuras 4.7, 4.8 y 4.9 muestran los resultados obtenidos.

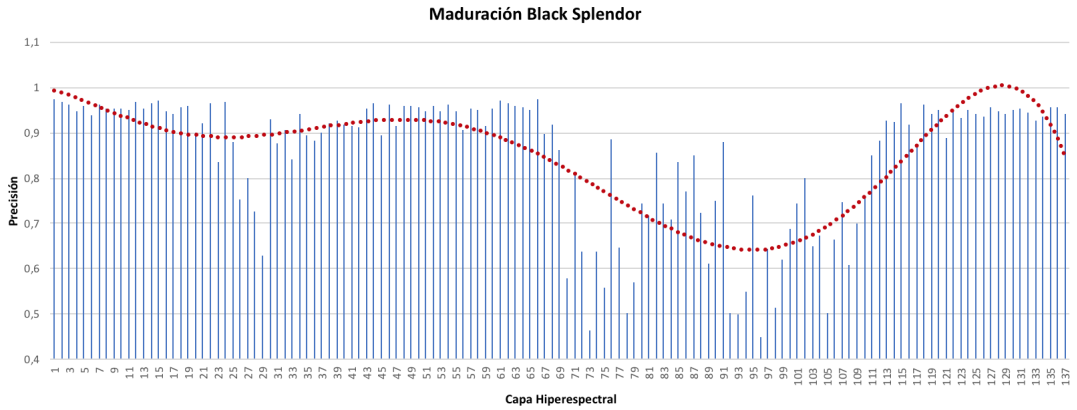

Figura 4.7: Resultado valor medio en clasificación por maduración, variedad Black Splendor

Si observamos los resultados obtenidos por el clasificador de la variedad Black Splendor por semana de maduración, se observa que existe una gran diferencia entre las diferentes capas de las imágenes hiperespectrales. Como se puede ver en la Figura 4.7, la precisión obtenida se puede dividir en tres bloques que son claramente diferenciables. Un primer bloque, de la capa 1 - 65, con un buen índice de precisión, entorno al 96 %. El segundo bloque, rango comprendido entre la capa 66 - 113, donde la precisión decae hasta valores cercanos al 40 %. Un último bloque, 114 - 137, donde la precisión se sitúa de nuevo al rededor del 96 %.

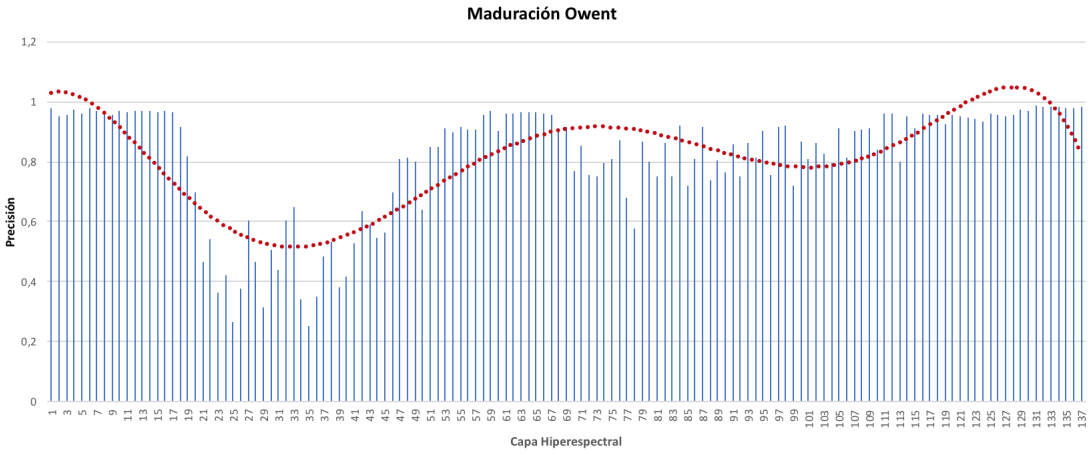

Figura 4.8: Resultado valor medio en clasificación por maduración, variedad OwenT

Los resultados obtenidos al clasificar imágenes hiperespectrales de la variedad OwenT -Figura 4.8- muestran, que al igual que sucedía con los resultados obtenidos en la variedad Black Splendor, sus resultados pueden dividirse en tres bloques. En el primer bloque, que comprende de la capa 1 hasta la capa 17 tienen una precisión cercana al 98 %. El segundo bloque, capa 18 - 57, obtiene mala precisión, algunas de sus capas tienen una precisión del 25 %. De la capa 58 hasta la capa 137 la precisión crece, situándose en un rango comprendido entre el 80 % y el 98 %.

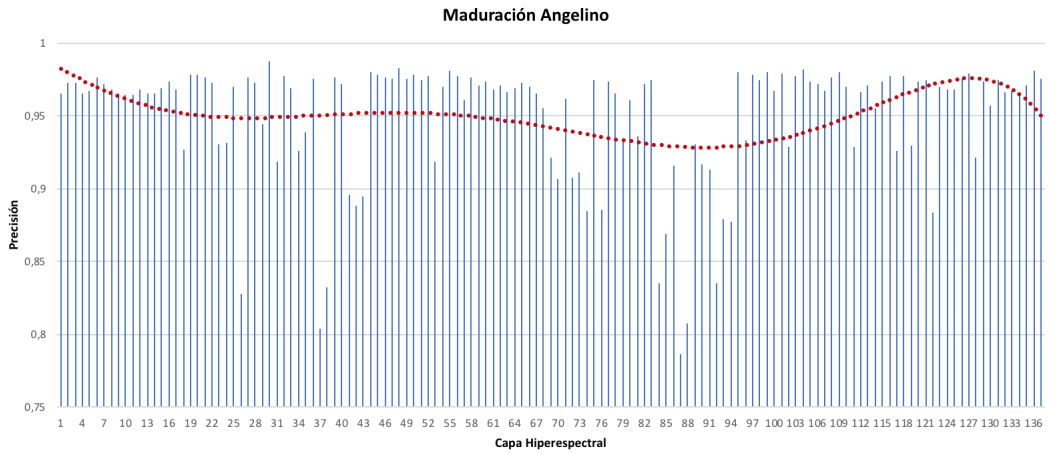

Figura 4.9: Resultado valor medio en clasificación por maduración, variedad Angelino

A diferencia de lo que ocurre con las variedades OwenT y Black Splendor los resultados obtenidos al clasificar la variedad Angeleno por maduración son muy similares en todas sus capas, 4.9. La precisión en ninguna de las capas es inferior al 80 %. La capa que ofrece mejor precisión es la capa 30 con un 98,78 %. Esto puede ser debido a que esta variedad es una variedad de ciclo muy largo, y las fechas en las que se ha recolectado los frutos, debido a su ciclo de maduración, no son lo suficientemente significativas en el tiempo.

A continuación, en la tabla 4.2 se muestra un resumen de las capas que obtienen los mejores resultados para cada una de las variedades.

Cuadro 4.6: Proposición capas para clasificar por estado de maduración

Variedad	Mejor Capa	Rango de capas
Angeleno	67	94 - 137
Black Splendor	7	1 - 65 / 114 - 137
OwenT	30	1 - 17 / 58 - 137

Capítulo 5

Conclusiones

A lo largo de este capítulo presentamos las conclusiones y las líneas futuras del trabajo aquí desarrollado. Se detalla el alcance de los objetivos propuestos al inicio de este trabajo, junto con el grado de consecución de los mismos, los problemas encontrados, así como las líneas futuras de trabajo que se han abierto con motivo de los resultados obtenidos.

5.1. Objetivos alcanzados

Al inicio de este proyecto se marcaron unos ambiciosos objetivos a cumplir. Como objetivo principal se presentó el diseño e implementación de un sistema capaz de clasificar ciruelas a través del análisis hiperespectral de las imágenes de dicho fruto. Este objetivo se dividía en dos subobjetivos: la optimización de clasificadores capaces de diferenciar entre tres variedades diferentes de ciruelas, en diferentes fases de su maduración. Por otra parte, un segundo subobjetivo que pretendía obtener un sistema clasificador que sitúe la fecha de maduración en la que se encuentra un determinado fruto de una variedad. A lo largo del estudio, como consecuencia del análisis hiperespectral de la información de las ciruelas, se pretende detectar zonas del espectro que nos permitan, gracias a las propiedades físico-químicas de las ciruelas, poder clasificar en función a los criterios establecidos anteriormente.

Los resultados presentados en el capítulo 4 muestran que es posible clasificar con una alto rango de acierto las variedades de ciruela seleccionadas para este estudio por medio de las imágenes hiperespectrales de las mismas tomadas en un entorno de laboratorio. Los datos obtenidos de clasificación demuestran que es posible clasificar la variedad de la ciruela atendiendo a su información hiperespectral y la semana de maduración en la que se encuentra, se han obtenido resultados del 92,32 %, 96,95 %, 96,48 % y 100,00 % respectivamente, para las cuatro fases de maduración empleadas en este estudio. Por otro lado, si la fase de maduración no se tiene en cuenta, se

obtiene unos resultados del 89,45 %.

Estos resultados nos permiten afirmar que es posible clasificar la variedad de una ciruela gracias al estudio de su espectro. Además, los datos obtenidos y presentados en el capítulo anterior nos permiten vislumbrar ciertas zonas de interés en el espectro de las diferentes variedades, atendiendo a su fecha de maduración.

Si comparamos los estudios realizados en este trabajo, con estudios previos en esta línea pero utilizando imágenes RGB, podemos llegar a la conclusión que ambos estudios obtienen muy buenos resultados y no se muestran diferencias significativas. Pero la ventaja de este estudio con respecto al anterior es que para llegar a estos niveles de precisión, se ha necesitado menos esfuerzo de cómputo, lo que permitirá en un futuro, aumentando el esfuerzo de cómputo y nuevos parámetros del espectro, llegar a mejores clasificadores. Incluso aumentando exponencialmente el dataset de imágenes, será posible obtener resultados en tiempos asumibles, ya que solo nos centraremos en las partes prometedores del espectro aquí detectadas.

5.2. Problemas encontrados

Durante la elaboración de este proyecto nos hemos encontrado con dificultades, en cada una de sus fases, que se han tenido que ir venciendo para alcanzar los objetivos marcados.

El principal problema que nos hemos encontrado en el desarrollo de la aplicación software ha sido el elevado número de dispositivos hardware que teníamos que controlar. Cada uno de los cuales - cámara RGB, cámara hiperespectral y la plataforma giratoria - tiene su propio protocolo de comunicación con el PC, por lo que tuvimos que formarnos en dichos protocoles para poder incorporarlos a la aplicación de captura de imágenes semi-automática. En el caso de las cámaras, su calibración se realiza a través de aplicaciones que se adquieren en el momento de comprarlas, estas aplicaciones son propietarias y no permiten su modificación. Esto ha supuesto un problema añadido que no se ha podido solucionar y es por ese motivo por el cual antes de comenzar con el uso de la aplicación de captura de datos, es necesario realizar la configuración y calibración de las cámaras con su software específico.

Otro problema con el que nos encontramos en este proyecto ha sido la utilización de CNNs. Aunque se trata de una técnica que en la actualidad es ampliamente utilizada, debido a los buenos resultados que están obteniendo en la clasificación de imágenes, tiene constantes cambios e incorporación de funcionalidades, lo que repercute en una elevada curva de aprendizaje. El uso se estas técnicas se complican por su continúa mejorada y gran rapidez con la que la comunidad realiza los cambios, dando lugar a numerosas variantes de la misma técnica. Aunque esto a priori es una ventaja que ayuda a avanzar a la ciencia, para el investigador supone una dificultad añadida ya que tiene que investigar en profundidad para escoger que variante de las

CNNs es la que mejor se adapta a su problema, en nuestro caso se decidió utilizar Alexnet.

Por último, otro problema que nos hemos encontrado es la necesidad de contar con el hardware que permita la utilización de las CNNs. No todos los equipos de cómputo están preparados para la utilización de dicha tecnología y es necesario contar con potentes GPUs que nos permitan ejecutar nuestros experimentos. En nuestro caso se ha contado con una GPU NVIDIA Tesla K20 que nos permitía ejecutar cada experimento en varios días de trabajo.

5.3. Líneas futuras

Una línea muy prometedora de este proyecto de investigación sería la creación de una aplicación, para dispositivos móviles, que contara con un sistema inteligente que permita la clasificación de las ciruelas por su estado de maduración o variedad. De esta manera el técnico agrario contaría con una herramienta que le ayudaría a tomar mejores decisiones, en el campo, en cuanto a escoger el momento óptimo de recolección, estado de su cosecha en un punto determinado, es decir, si la cosecha se encuentra en un determinado momento, pero las técnicas de inteligencia artificial detectan que se encuentra en otro diferente, esto puede ser debido a riegos deficitarios, estrés, etc. Todo ello, si es detectado a tiempo gracias a estas técnicas, podrá corregirse a lo largo del periodo de maduración y obtener una cosecha de mayor calidad.

Otra línea muy prometedora sería la clasificación de las ciruelas por el contenido de diversas características físicas del fruto, tales como sólidos solubles, PH, acidez o grados brix. En la actualidad estas características se obtienen por técnicos altamente cualificados en costosos laboratorios. Se puede desarrollar una herramienta software que incorpore modelos entrenados mediante algoritmos de machine learning que permitan el análisis de imágenes, RGB e hiperespectrales, para conocer la calidad del fruto durante toda su fase maduración, de esta forma se podrían tomar acciones correctoras para obtener ciruelas con la máxima calidad.

Bibliografía

[1] Wang, H., Peng, J., Xie, C., Bao, Y., He, Y. *"Fruit quality evaluation using spectroscopy technology: a review,"* Sensor 15(5), 11889-11927, 2015.

[2] J.A. Abbott *"Quality measurement of fruits and vegetables,"* Postharvest Biology and Technology 15(3):207–225, 1999.

[3] Sergio Cubero, Nuria Aleixos, Enrique Moltó, Juan Gómez-Sanchis, and Jose Blasco. *"Advances in machine vision applications for automatic inspec- tion and quality evaluation of fruits and vegetables,"* Food and Bioprocess Technology, 4(4):487–504, 2011.

[4] M.T. Riquelme, P. Barreiro, M. Ruiz-Altisent, and C. Valero. *"Olive classification according to external damage using image analysis,"* Journal of Food Engineering, 87(3):371–379, 2008.

[5] P.B. Pathare, U.L. Opara, and F. A. J. Al-Said. *"Colour measurement and analysis in fresh and processed foods: A review,"* Food and Bioprocess Technology, 6(1):36–60, 2013.

[6] Pal, M. *"Random forest classifier for remote sensing classification,"* International Journal of Remote Sensing, 26(1), 217-222. 2005.

[7] Danfeng Wang, Xichang Wang, Taiang Liu and Yuan Liu, *"Prediction of total viable counts on chilled pork using an electronic nose combined with support vector machine,"* Meat Science, Volume:90, pag:373 - 377, 2012.

[8] Krizhevsky, A., Sutskever, I., Hinton *"Imagenet classification with deep convolutional neural networks. In: Bartlett, P., Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.),"* Advances in Neural Information Processing Systems 25, pp. 1106–1114, 2012

[9] J. Xing, D. Guyer, D. Ariana, and R. Lu. *"Determining optimal wavebands using genetic algorithm for detection of internal insect infestation in tart cherry,"* Sensing and Instrumentation for Food Quality and Safety, 2(3):161– 167, 2008.

[10] C. Yang, W.S. Lee, and P. Gader. *"Hyperspectral band selection for detecting different blueberry fruit maturity stages,"* Computers and Electronics in Agriculture, 109:23–31, 2014.

[11] Lu, R., Ariana, D. P. *"Detection of fruit fly infestation in pickling cucumbers using a hyperspectral reflectance/transmittance imaging system,"* Postharvest Biology and Technology, 81, 44-50, 2013.

[12] Haff, R. P., Saranwong, S., Thanapase, W., Janhiran, A., Kasemsumran, S., Kawano, S. *"Automatic image analysis and spot classification for detection of fruit fly infestation in hyperspectral images of mangoes,"* Postharvest Biol. Technol, 86, 23-28, 2013

[13] Dubey, S. R., Jalal, A. S. *"Adapted approach for fruit disease identification using images,"* arXiv preprint arXiv:1405.4930, 2014

[14] Hailong Wang, Jiyu Peng, Chuanqi Xie, Yidan Bao y Yong He *"fruit quality evaluation using spectroscopy",* Sensor, 21 mayo 2015

[15] MJ Serradilla, MA Manzano, JR Mateos, F Pérez, J Prieto, V Alarcón, and Margarita López-Corrales. *"Influencia de diferentes patrones de cerezo en el comportamiento agronómico y calidad del fruto de las variedades "summit" y "sunburst","* ITEA, 104:3–11, 2008.

[16] S. Edward Law. *"Scatter of near-infrared radiation by cherries as a means of pit detection,"* Journal of Food Science, 38(1):102 – 107, 1973.

[17] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. *"Deep learning"* Nature 521, no. 7553 436-444. 2015

[18] Simonyan, K., Zisserman, A. *"Very deep convolutional networks for large-scale image recognition"* arXiv preprint arXiv:1409.1556. 2014

[19] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., ... , Darrell, T *"Caffe: Convolutional architecture for fast feature embedding,"* In Proceedings of the 22nd ACM international conference on Multimedia (pp. 675-678). ACM, 2014, November.

[20] Rodríguez, Francisco J. and García, Antonio and Pardo, Pedro J. and Chávez, Francisco and Luque-Baena, Rafael M. *Study and classification of plum varieties using image analysis and deep learning techniques* Progress in Artificial Intelligence, 7(2), 119-127, 2018