Abstract

In the present work a system for the identification of the product in the agricultural crops of zucchini is designed and developed. For the identification of the product we use images obtained by a camera within the visible spectrum in outdoor environments. The main complexity lies in the differentiation of the product of the leaves and the stem, since these three elements present similar textures that at times can be complicated to differentiate even by a human expert. The problem has been addressed by the analysis of textures obtained by the co-occurrence matrix and the histogram. With the information obtained through the cooccurrence matrix, a system based on supervised learning and a system is obtained able to identify the product of the zucchini using Euclidean distance and Mahalanobis distance.

Keywords:

Image processing – Grayscale – Histogram – Co-occurrence matrix – Texture analysis – Database – Classifier – Euclidean distance

Índice de figuras y tablas

Capítulo 1.Introducción

1.1. Identificación y estado del arte

A la hora de pensar sobre la forma de trabajar en los cultivos, aquella forma que existe desde el Neolítico y que se ha transmitido de padres a hijos dejando sus métodos, patrones, su forma de hacer que el cultivo produzca, cuyo producto es fundamental para la supervivencia de la humanidad, un producto que hace que el agricultor se sacrifique y trabaje de sol a sol, nos hace plantearnos si no existen formas de ayudarles, de hacer que el trabajo no suponga tal esfuerzo físico, de poder garantizarles que podrán trabajar de una forma más cómoda y asequible, de evitarles preocupaciones y regalarles más tiempo para su disfrute personal.

Tal forma se ha ido desarrollando con cada revolución, con cada avance, ocurrió en el siglo XII con la llegada del arreo de collar, siguió en el año 1700 con la barrena mecánica y continuó con maquinarias en la era de la revolución industrial. Ahora, que estamos en plena revolución tecnológica, donde es uno de los sectores más pioneros y con más auge, debemos obsequiarles con un avance que haga su trabajo más confortable y ameno, así garantizandoles una mejora en su calidad de vida.

Esta tecnología se ha ido incorporando en la vida cotidiana del ser humano, como iRobot [1] limpieza en el hogar, cuyo funcionamiento lo realiza gracias a un patrón de limpieza y un juego de sensores como la localización visual, que le ayuda a trazar un mapa de obstáculos, que se adapta a ello para conseguir la máxima cobertura posible.

O en el ámbito industrial como en la soldadura de punto y soldadura de arco, en transportación de materiales, moldeado en la industria plástica, entre otros.

Ahora bien específicamente en el sector de la agricultura se han desarrollado múltiples avances, en el caso más reciente, en el año 2016 una investigación en el control de los cultivos, para poder predecir la aparición de anomalías antes de tiempo, esta investigación fue realizada por el centro de automática y robótica de la Universidad Politécnica de Madrid [2], y se llevó a cabo mediante un robot terrestre y otro aéreo cuya finalidad era medir la temperatura, humedad, luminosidad y concentración de dióxido de carbono establecido en distintas etapas. La información recogida por ambos robot permitió conocer en todo momento las condiciones actuales de los cultivos y así poder detectar problemas. El desarrollo de esta tecnología hace que estén supervisados los cultivos las 24 horas del día, dando la posibilidad de que no aparezca ninguna anomalía y por consiguiente la mejora en productividad. Este proyecto se pudo realizar gracias a la combinación de algoritmos y sensores instalados en ambos robots.

Por otro lado, *the University of Lincoln* desarrolló el robot *Thorvald* [3] cuyo objetivo es dar apoyo en el área de calidad del suelo, predicciones de cosechas, realizar fotografías y diversas cuestiones de logística y transporte, cuya capacidad se traslada en operar en terrenos irregulares. Entre una de sus operaciones está la de poder dirigir una luz ultravioleta de intensidad extremadamente baja para así eliminar el moho de sus cultivos.

FamBot [4] se presenta como un robot cuyo avance tecnológico se centra en los cultivos de precisión, donde siembra, riega y cuida de los huertos de manera personalizada.

Ahora bien, el avance que más se asemeja a este proyecto consiste en una recolectora de frutas, el proyecto Zito del año 2010-2011 [5], donde se cosecha la fruta sin dañarla, ya que por medio de sensores realiza una identificación y localización del producto, verificando antes si es recomendable su cosechado, es decir, es capaz de tomar decisiones mediante la medición de color.

1.2. Propuesta

Como se puede observar los avances en tecnología han sido uno de las salidas profesionales más comunes en los últimos años, intentando modernizar todas las áreas posibles a través de la investigación y desarrollo. Por lo tanto, nuestro proyecto intentará dar un pequeño paso y aportar ideas a la continuación del desarrollo e investigación, haciendo hincapié en la inspección autónoma de los cultivos, dando una posible solución inicial a un problema existente. Se diseña un prototipo para la identificación y clasificación de productos de alto valor añadido, con el fin de que se pueda integrar a un robot móvil, donde poder realizar distintas operaciones eficientes como la recolección o extracción.

Para ello hacemos grandes esfuerzos en investigar y aprender distintos métodos para poder desarrollar, de la mejor manera posible, un prototipo para el robot automatizado.

1.3. Motivación y Objetivos

1.3.1. Motivación

Nuestra motivación al seleccionar este proyecto fue su planteamiento a la ayuda de un sector vital para nuestra sociedad y economía, así como el desarrollo e investigación en el ámbito de percepciones computacionales e inteligencia artificial y por supuesto en formar parte del desarrollo del robot móvil que realiza el CSIC-CAR, Centro de automática y robótica del Consejo Superior de Investigaciones Científicas.

Ayudó también el haber cursado asignaturas como percepción computacional e inteligencia artificial para decidir la elección, por el simple hecho de poder plasmar lo que hemos aprendido, afianzar y aprender los conceptos que integran estas asignaturas.

Y por último el poder desarrollar un proyecto que no solo servirá a un solo sector, sino que podría distribuirse a otros sectores que se planteen alguna similitud.

1.3.2. Objetivos

Como objetivo principal de este proyecto se centrará en la investigación orientada al diseño y desarrollo de un prototipo para la plataforma robótica móvil con capacidad para la inspección de cultivos, centrada en la identificación y clasificación supervisada de productos de alto valor añadido.

Como nuestro proyecto comenzó desde cero, nos marcamos como objetivos el poder identificar y clasificar solo un tipo de cultivo, dejando para futuros trabajos la posibilidad de añadir otro tipo de cultivos y otras formas de identificación.

A continuación, listamos los objetivos propuestos:

- Investigación en el sector del procesamiento de imágenes digitales.
- Investigación del procesamiento de imágenes en el campo de cultivos agrícolas.
- Determinar métodos para la identificación y clasificación de texturas en imágenes.
- Analizar las ventajas e inconvenientes de los distintos métodos obtenidos.
- Determinar los métodos a utilizar.
- Estudio de los diferentes clasificadores y elección del que mejor se ajuste a nuestro problema.
- Identificación de un determinado cultivo.
- Análisis de los tiempos de ejecución.

1.4. Metodología

Dada nuestra estancia en la universidad y el aprendizaje obtenido en esta, decidimos utilizar una metodología ágil, donde establecemos el desarrollo de pequeños prototipos o

versiones en un determinado tiempo para así poder obtener los objetivos establecidos y dar la mejor eficacia a nuestro proyecto.

En cada una de las etapas se distribuyen diferentes días para las reuniones y puestas en marchas del trabajo recopilado.

El proyecto se distribuye en diferentes fases:

1. Planificación, donde se atribuye el sector que vamos a desarrollar y el tiempo estimado.
2. Investigación, cuya finalidad fue inspeccionar e investigar los distintos métodos y formas que serían más convenientes para el desarrollo de nuestro proyecto.
3. Diseño, donde el objetivo es esbozar un boceto de la versión.
4. Implementación, donde se desarrolla el prototipo establecido con las herramientas que hemos obtenido en las etapas anteriores.
5. Validación del prototipo y pruebas a realizar.

Chapter 1.Introduction

1.1. Identification and state of the art

When thinking about how to work in crops, whose form there has existed since the Neolithic and has been transmitted from parents to children leaving their methods, patterns, how to make the crop produce, whose product is essential for the survival of humanity, a product for which makes the farmer sacrifice himself and work from sun to sun, makes us think if there is no way to help them, to make the work not involve such physical effort, to be able to guarantee that they can work in a way more comfortable and affordable, to avoid their worry and give them more time for their personal enjoyment.

Such shape has been developed with each revolution, with each advance, occurred in the XII century with the arrival of the collar, followed in 1700 with the mechanical auger and continued with machinery in the era of industrial revolution now that we are in the middle of a technological revolution, where it is one of the most pioneering and booming sectors, we must give them an advance that will make their work more comfortable and enjoyable, thus guaranteeing an improvement in their quality of life.

This technology has been incorporated into the daily life of the human being, such as iRobot [1] cleaning at home, whose operation is performed thanks to a cleaning pattern and a set of sensors such as visual localization, which helps you draw a map of obstacles, which is adapted to achieve maximum coverage.

Or in the industrial field as in spot welding and arc welding, in materials transportation, molding in the plastic industry, among others.

Specifically in the agricultural sector many advances have been made, in the most recent

case, in the year 2016, an investigation in the control of crops, in order to be able to predict the anomalies prematurely, this research was carried out by the center of automatic and robotics of the Universidad Politécnica de Madrid [2], and was carried out by means of a terrestrial and aerial robot whose purpose was to measure the temperature, humidity, luminosity and concentration of carbon dioxide established in different stages. The information collected by both robot allowed to know at all times the current conditions of the crops and to be able to detect problems, the development of this technology makes the crops monitored 24 hours a day, giving the possibility that no anomaly appears and therefore the improvement in productivity. This project could be realized thanks to the combination of algorithms and sensors installed in both robot.

On the other hand, the University of Lincoln developed the robot Thorvald [3] whose objective is to provide support in the area of soil quality, crop forecasts, photographs and various logistical and transport issues, whose capacity is transferred to operate in irregular ground. One of its operations is to be able to direct ultraviolet light of extremely low intensity to eliminate mold from its crops.

FamBot [4] is a presented as a robot whose technological progress focuses on precision crops, where it sows, water and care for the orchards in a personalized way.

However, the development that most closely resembles this project consists of the fruit picker, the Zito project of the year 2010-2011 [5], where fruit is harvested without damaging it, as it makes identification and localization through sensors Of the product, checking firstly if its harvesting is advisable, that is to say, it is able to make decisions through the measurement of color.

1.2. Proposal

As you can see the advances in technology have been one of the most common

professional exits in recent years, trying to modernize all possible areas through research and development. Therefore, our project will try to take a small step and contribute ideas To the continuation of development and research, with emphasis on the autonomous inspection of crops, giving a possible initial solution to an existing problem. The prototype is designed for the identification and classification of products with high added value, so that it can be integrated into a mobile robot, where it can carry out different efficient operations such as collection or extraction.

For this we make great efforts in investigating and learning different methods to be able to develop, in the best possible way, a prototype for the automated robot.

1.3. Motivation and Objectives

1.3.1. Motivation

Our motivation in selecting this project was its approach to the help of a vital sector for our society and economy, as well as the development and research in the field of computer perceptions and artificial intelligence and of course in being part of the development of the mobile robot that realizes The CSIC-CAR, Automatic Center and Robotics of the Higher Council of Scientific Research.

It also helped to have studied subjects such as computer perception and artificial intelligence to decide the choice, for the simple fact of being able to capture what we have learned, strengthen and learn the concepts that integrate these subjects.

And finally, the power to develop a project that will not only serve a single sector, but could be distributed to other sectors that have some similarity.

1.3.2. Objectives

The main objective of this project will be to focus on the design and development of a prototype for the mobile robotic platform with capacity for crop inspection, focused on the identification and supervised classification of high added value products.

As our project started from scratch, we set ourselves the objectives of being able to identify and classify only one type of crop, leaving for future work the possibility of adding other types of crops and other forms of identification.

Here are the proposed objectives:

- Research in the field of digital image processing.
- Investigation of image processing in the field of agricultural crops.
- To determine methods for the identification and classification of textures in images.
- Analyze the advantages and disadvantages of the different methods obtained.
- Determine the methods to use.
- Study the different classifiers and choose the one that best fits our problem.
- Identification of a particular crop.
- Analysis of execution times.

1.4. Methodology

Given our stay in the university and the learning obtained in this, we decided to use an agile methodology, where we establish the development of small prototypes or versions in a certain time in order to obtain the established objectives and give the best efficiency to our project.

In each of the stages different days are distributed for the meetings and marches of the work compiled.

The project is distributed in different phases:

1. Planning, where is attributed the sector that we are going to develop and the estimated time.
2. Research, whose purpose was to inspect and investigate the different methods and forms that would be most convenient for the development of our project.
3. Design, where the goal is to sketch a small sketch of the version.
4. Implementation, where the established prototype is developed with the tools we have obtained in the previous stages.
5. Validation of the prototype and tests to be performed.

Capítulo 2.Materiales y métodos

2.1. Análisis de requisitos y solución

2.1.1. Análisis de requisitos

Para poder definir correctamente los requisitos, ha de fijarse, en un principio en cómo funciona la inspección de cultivos en el mundo real, ya que el robot móvil ha de poder realizar las operaciones habituales de los seres humanos y conseguir mejorarlas en un tiempo más eficaz, ya que ha de poder mejorar la productividad.

Para cualquier tipo de operación a realizar, el primer paso es observar las percepciones que recibimos, como seres humanos nuestras percepciones son los sentidos, en el caso del robot sus percepciones son las que recibe del exterior a través de los sensores. Por ello uno de los requisitos fundamentales es la capacidad de integración de la imagen a estudiar, mediante percepción visual a través de sensores.

De la imagen recibida y a través de un estudio visual, el ser humano, como experto, se fijaría en pequeños sectores y sacaría características del objeto. Para que el robot pueda adquirir esta funcionalidad ha de poder analizar sectores pequeños de la imagen y sacar datos de ello, por lo que esta función ha de poder procesar o transformar la imagen y realizar distintos cálculos. Los pasos a seguir quedarían enumerados como:

1. Sacar pequeños sectores de la imagen.
2. Procesar o transformar estos pequeños sectores de la imagen.
3. Realizar distintos métodos para sacar datos.

Tras haber extraído características de la imagen, el ser humano extrae conclusiones de los

datos que recibe y actúa de acuerdo a ellos. Los pasos a seguir quedarían enumerados como:

4. Analizar los datos e integrarlos.
5. Sacar conclusiones de los datos obtenidos.
6. Realizar la clasificación.

Y por último, tras haber podido cumplir con los requisitos anteriores, se procede a la identificación del cultivo, dando por concluido el cumplimiento final de todos los objetivos.

2.1.2. Análisis de la solución

Tras haber analizado los requisitos del proyecto, se han planteado las siguientes soluciones:

1. A la hora de integrar una imagen a un determinado programa, se plantea las siguientes propuestas:
    a. Descarga de imagen. Esta propuesta hace referencia a la búsqueda de imagen y selección mediante archivo, donde vendrá alojado en una carpeta exclusivamente de imágenes.

       Esta opción es la más habitual a la hora de integrar cualquier tipo de archivo a un programa.
    b. Integración directa mediante cámara fotográfica. Esta propuesta es las más parecida a la hora de poder desarrollar para un robot, pero lleva recursos extras el poder implementarla.
2. En cuanto a sacar sectores de la imagen, ha de poder seleccionar fragmentos de la imagen, con sus correspondientes coordenadas.
3. En el ámbito de transformadas o procesamientos de imágenes, hay que tener en cuenta el preprocesamiento de la imagen, ya que no se debe de conformar solo con la imagen en RGB, si no se debe buscar el tratamiento de la imagen que más convenga a la hora de extraer información. Se plantean los siguientes procesamientos:
    a. Transformar la imagen a escala de grises, este tipo de procedimiento ayuda a la hora de poder obtener un histograma de la imagen y sacar datos de ella, como

puede ser media, varianza, entropía, etc. Además de poder realizar otros tratamientos de la imagen si es necesario.

b. Transformar la imagen al modelo CMY, donde se obtiene una imagen con ausencia de luz.
c. Transformación al modelo YIQ, donde se desacoplan la luminancia y la información del color.
d. Transformación al modelo HSI, donde se obtiene algunas propiedades del objeto mediante el color.

4. Tras haber procedido al procesamiento de imágenes, se ha de conseguir realizar cálculos que nos ayuden a sacar conclusiones. Para ello se plantean las siguientes técnicas:
    a. Realizar cálculos gracias al histograma, es decir, obtener datos de media, varianza y entropía.
    b. Detectar bordes mediante el algoritmo de Sobel, este algoritmo intenta extraer los pixeles con alta variación en la escala de grises que se encuentra en la imagen, es decir, calcula los gradientes con sus respectiva magnitud y dirección de cada pixel y este se compara con un determinado umbral.
    c. Detectar bordes mediante el algoritmo de Prewitt, este algoritmo es similar al Sobel a excepción de que tiene que comparar con un umbral diferente.
    d. Detectar bordes mediante el algoritmo de Roberts, este algoritmo es similar a los anteriores, pero es más sencillo, y con un umbral diferente.
    e. Detectar bordes mediante el operador Laplace, este algoritmo permite detectar bordes como los anteriores a excepción de que reduce la cantidad de bordes falsos, como consecuencia es mucho más complicado que los anteriores ya que dispone de cálculos de segunda derivada.
    f. Detección de puntos de interés mediante el algoritmo SIFT, este algoritmo se utiliza habitualmente para el reconocimiento de un objeto o escenario en una

imagen, este hace hincapié en detectar los puntos de la imagen donde existe información relevante, como por ejemplo la existencia de bordes o texturas.

g. Detección de puntos de interés mediante el algoritmo SURF, este algoritmo es similar al SIFT, intenta obtener puntos de interés en una imagen para así poder detectar un objeto, la principal diferencia radica en que utiliza una técnica de *Scale-space,* que asegura que los puntos de interés obtenidos sean invariantes en el escalado.
h. Cálculo de la matriz de coocurrencia, donde se hace análisis textural de la imagen y discriminación de patrones, sirve para reconocer diferencias texturales en la imagen.

5. Tras haber analizado y obtenido los datos de la imagen, se debe realizar las operaciones oportunas. Una de las operaciones importantes, que ha de poder atender el programa es el guardar, actualizar y cargar los datos obtenidos mediante el análisis que se haya hecho a la imagen, para ello se proponen las siguientes soluciones.
    a. Integración de datos mediante archivos de texto, este tipo de guardado es el más sencillo a la hora de implementar o programar, pero al mismo tiempo el menos productivo para poder guardar gran cantidad de datos, el más costoso en cuanto a almacenamiento secundario, así como el menos propenso en la recogida, validación y consulta de los datos.
    b. Integración de datos mediante base de datos, este tipo de guardado es más costoso a la hora de implementarla o programarla, ya que hace falta un buen diseño de la base de datos, así como un buen conocimiento para una utilización correcta y eficaz, pero mejora en cuanto a independencia de datos tanto lógica como física, coherencia de los resultados ya que los datos se recogen y almacenan una sola vez así eliminando en gran medida la redundancia de datos, mejora la disponibilidad de consulta o extracción de datos, y reduce el espacio de almacenamiento.

Al haber propuesto el guardado de datos, queda el último paso, la clasificación de las muestras para la identificación de los cultivos.

a. En esta primera propuesta se establece una clasificación de las muestras de la imagen para establecer la pertenencia o no a un determinado sector, mediante *machine learning* no supervisado del programa, este tipo de clasificación es más costoso a la hora de programar. Además requiere de una gran cantidad de muestras para de este modo se realice una fase de aprendizaje de calidad usando los patrones a referencia de datos. Si se realiza una fase de aprendizaje correcta el sistema será capaz de decidir independientemente si una muestra pertenece o no a una determinada clase.
b. En esta propuesta se establece una clasificación de muestras de la imagen para establecer la pertenencia o no a un determinado sector, mediante *machine learning* supervisado del programa, es decir, poder supervisar las muestras establecidas como conjunto de datos, establecer la pertenencia de una clase u otra.

Se propone la identificación del sector de la imagen a la pertenencia de una clase u otra mediante algoritmos de distancia.

2.1.3. Solución propuesta

Tras haber analizado en profundidad las diferentes opciones planteadas anteriormente se establece como solución al desarrollo del prototipo y en definitiva al programa con las siguientes propuestas, explicadas mediante las distintas fases por la que ha pasado el proyecto:

2.1.3.1 Fase 1: selección de la imagen a analizar

La integración de la imagen se estableció mediante búsqueda de archivo, esta búsqueda se realiza en una determinada carpeta, donde vendrán integradas todas las imágenes que se necesiten.

Se estableció como nombre al pulsador de carga como “añadir imagen”.

Se llegó al acuerdo de procesar la imagen a escala de grises por su amplia gama de posibilidades a trabajar con ella, obteniendo diversos resultados matemáticos, así como realizar diversos procedimientos partiendo de esta imagen.

Se designó la visualización de imagen tanto en RGB como en escala de grises [14], dentro de unos recuadros limitados para poder trabajar con ambas, si es necesario.

2.1.3.2 Fase 2: seleccionar Región de Interés (ROI)

Se estableció sacar fragmentos de la imagen, mediante marcación directa, tanto en la imagen RGB como en la imagen a escala de grises.

Esta marcación se estableció a recuadros de color rojo y con líneas discontinuas [15].

Se observó que era demasiado inconveniente y tedioso, trabajar con ambas en la misma vista, por lo que se separó en dos vistas bien distinguidas:

- Primera vista: destinada a buscar y cargar la imagen, así como la visualización de la imagen original en RGB.
- Segunda vista: destinada a mostrar la imagen en escala de grises, así como todas las posibles operaciones y cálculos con base en esta imagen.

2.1.3.3 Fase 3: actualización de las interfaces de usuario

Se establecieron varios puntos a mejorar en las dos vistas de la interfaz de usuario:

- Primera vista
    - Además de mostrar la imagen original en RGB, se estableció el procesar directamente el fragmento seleccionado dentro de la imagen RGB, es decir, transformar y mostrar solamente, en la segunda vista la parte seleccionada en escala de grises.
    - Se estableció el incluir el botón de procesamiento directo a escala de grises de toda la imagen, este pulsador se le puso como nombre “añadir imagen en gris”.
    - Se llegó al acuerdo de incluir un botón de selección de toda la imagen para

pasarla a escala de grises, este pulsador se le puso como nombre “seleccionar toda la imagen”.

Para aclarar, con respecto a los botones, tanto el añadir como el seleccionar tienen la misma finalidad y es procesar la imagen a escala de grises, pero partiendo de distintos estados.

“Añadir imagen en gris” parte directamente de la búsqueda de la imagen dentro de la carpeta, donde está situada todas las imágenes, en cambio “seleccionar toda la imagen”, parte del momento en el que ya se ha cargado la imagen en la primera vista.

Para no parecer repetitiva se establece desactivar el botón “seleccionar toda la imagen” hasta que se carga una imagen en la primera vista.

- Segunda vista
    - Se estableció el diseño de la segunda vista, dando como resultado la limitación de la visualización de la imagen a escala de grises aproximadamente a ⅘ del total de pantalla maximizada, dejando el ⅕ de la pantalla a informar de las distintas posibilidades de cálculos y visualización de resultados.
    - Se llegó al acuerdo de mostrar los puntos de coordenadas de la imagen referenciada de la original como “x”, “y”, “anchura” y “altura”.

Se pone de manifiesto el poder trabajar con varias segundas vistas simultáneamente, por lo que la creación de segundas vistas a escala de grises no está limitada.

Se fijó el poder desplazarse mediante barras de desplazamiento, en las dos vistas cuando la pantalla no tiene la maximización completa.

2.1.3.4 Fase 4: cálculo de los descriptores

Al concluir con la forma de visualización y funciones generales de esta, se realizó una investigación en el tratamiento de la imagen, y se establecieron como los cálculos más convenientes a la hora de implementar y trabajar con ellos, la utilización de histograma de la imagen para obtener tanto la media, varianza y entropía, así como el análisis de texturas, es decir

la matriz de coocurrencia.

Se seleccionó el análisis de texturas con respecto a la detección de bordes y puntos de interés, por la información encontrada y tiempo disponible para poder extraer e implementar.

2.1.3.5 Medidas del histograma de la imagen

El histograma se estableció con los 255 píxeles correspondientes de la imagen, y se calcularon las siguientes medidas:

- Media

$$g = \sum_{i=0}^{255} (i \times \frac{h(i)}{255})$$

Donde:

h es el histograma de la imagen.

- Varianza

$$\sigma^2 = \sum_{i=0}^{255} ((i-g)^2 \times \frac{h(i)}{255})$$

Donde:

g es la media.

h es el histograma de la imagen.

- Entropía

$$-e = \sum_{i=0}^{255} (\frac{h(i)}{255} \times log_2(\frac{h(i)}{255}))$$

Donde:

h es el histograma de la imagen.

2.1.3.6 Matriz de coocurrencia

Como ya habíamos presentado, la matriz de coocurrencia juega un importante papel para el análisis de texturas, exactamente la textura es una de las características utilizadas para la identificación de objetos o regiones de interés, ya que es una cuantificación de la variación espacial de valores de tono.

Basado en estadísticas, utilizaremos la matriz de coocurrencia para obtener variables de texturas de segundo orden.

El cálculo de la matriz de coocurrencia, es en realidad un histograma de los niveles de grises de dos dimensiones para un par de pixeles, es decir, la frecuencia de un nivel de gris que aparece en una relación espacial, dentro de una determinada ventana, en conclusión, la probabilidad de distribución de un par de pixeles.

Para el cálculo de la matriz necesitamos la componente espacial, es decir, la distancia entre el par de pixeles y el ángulo para el cómputo, este ángulo se expresa mediante 0º, 45º, 90º, y 135º, ya que el resto de ángulos posibles, exactamente 8, se pueden medir mediante una matriz simétrica.

Deduciendo de la cantidad de grados y distancia, de una misma imagen se puede obtener diferentes matrices de coocurrencia.

Ahora bien, para explicar mejor el cálculo de la matriz proponemos un ejemplo sencillo. Supongamos que tenemos esta simple imagen de 4x4 con valores de grises del 0 al 3.

0	0	1	1
0	0	1	1
0	2	2	2

2	2	3	3

Tabla 1. ejemplo escala de grises.

La matriz considera la relación entre un par de pixeles llamados, píxel de referencia y pixel vecino.

Hay que tener en cuenta los pixeles ubicados en los márgenes, ya que dependiendo de la componente espacial estos no serán usados en el cómputo.

Para establecer a qué distancia están estos pixeles se le atribuye la componente espacial, la distancia y el ángulo, que se expresa como una tupla con componente x e y.

x expresa la distancia en el eje x.

y expresa la distancia en el eje y.

(1,0), expresa 1 en dirección x y 0 en dirección y, es decir, grado 0º con distancia 1.

(1,1), expresa 1 en dirección x y 1 en dirección y, es decir, grado 45º con distancia 1.

(0,-1), expresa 0 en dirección x y -1 en dirección y, es decir, grado 90º con distancia 1.

(1,-1), expresa 1 en dirección x y -1 en dirección y, es decir, grado 135º con distancia 1.

Dependiendo de esta componente debemos contar la frecuencia de píxeles que cumplen, que el pixel de referencia está seguido del píxel vecino.

En la siguiente tabla, expresamos el resultado de la matriz con grado 0º y con distancia 1, del ejemplo de la imagen 4x4.

p. referencia/p.vecino	0	1	2	3
0	2	2	1	0

1	0	2	0	0
2	0	0	3	1
3	0	0	0	1

Tabla 2. Matriz de Coocurrencia.

A simple vista se puede observar que la matriz resultante no es simétrica, no es lo mismo contar el pixel referencia 0 y pixel vecino 1 cuyo resultado es 2, que el pixel referencia 1 y pixel vecino 0 cuyo resultado es 0, y esta cualidad es importante para poder realizar los cálculos oportunos.

Se puede obtener la matriz simétrica de varias formas.

Sumando cada par de pixel de derecha a izquierda y viceversa.

Sumando la matriz traspuesta con la original.

En cualquier caso, el resultado de la matriz simétrica, para este ejemplo es:

p. referencia/p.vecino	0	1	2	3
0	4	2	1	0
1	2	4	0	0
2	1	0	6	1
3	0	0	1	2

Tabla 3. Matriz simétrica de Coocurrencia.

Al obtener esta matriz, a continuación, se debe obtener la matriz de probabilidad, es decir, calcular el número de veces que un evento ocurre dividido por el número total de posibles eventos, cuya ecuación es:

$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=0}^{N-1} V_{i,j}}$$

Donde:

i es el número de filas y j el número de columnas.

V es el valor de la celda i,j.

Pi,j es la probabilidad en la celda i,j.

N es el número de filas o columnas.

A continuación, explicaremos los descriptores que hemos calculado con la matriz de coocurrencia.

- Homogeneidad

$$\sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1+(i-j)^2}$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

- Contraste

$$\sum_{i,j=0}^{N-1} (P_{i,j} * (i-j)^2)$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

- Disimilaridad

$$\sum_{i,j=0}^{N-1} (P_{i,j} * |i-j|)$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

- GLCM Media

$$\sum_{i,j=0}^{N-1} (i * P_{i,j})$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

- Desviación Standard

Para calcular la desviación standard, debemos primero calcular la varianza

$$\sigma_i^2 = \sum_{i,j=0}^{N-1} (P_{i,j} * (i - \mu_i)^2)$$

$$\sigma_j^2 = \sum_{i,j=0}^{N-1} (P_{i,j} * (i - \mu_i)^2)$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

σ^2 la varianza

μ la media.

Al obtener la varianza, podemos calcular la desviación standard

$$\sigma_i = \sqrt{\sigma_i^2}$$

$$\sigma_j = \sqrt{\sigma_j^2}$$

Siendo:

i es el número de filas y j el número de columnas.

σ^2 la varianza

σ la desviación standard.

- Entropía

$$\sum_{i,j=0}^{N-1} (-P_{i,j} * ln(P_{i,j}))$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

Asumiendo que 0*ln (0) =0

- Correlación

$$\sum_{i,j=0}^{N-1} (P_{i,j} * [\frac{(i-\mu_i)*(j-\mu_j)}{\sqrt{(\sigma_i^2)*(\sigma_i^2)}}])$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

σ^2 la varianza

μ la media.

- ASM(Angular Second Moment)

$$\sum_{i,j=0}^{N-1} P_{i,j}^2$$

Siendo:

i es el número de filas y j el número de columnas.

N es el número de filas o columnas.

Pi,j la probabilidad de la matriz coocurrencia de los valores de gris i,j.

Toda la información obtenida de la matriz de coocurrencia viene publicada en el punto [6] y [7] de la bibliografía.

Tras realizar la implementación de los descriptores, visualización de las funciones y posibles resultados, se procedió a realizar las pruebas o validación con diferentes segmentos de la imagen y su puesta en conocimiento.

2.1.3.7 Fase 5: integración de datos y clasificación

Tras la aceptación de las pruebas, se fijó el limitar tanto los grados a 0º, 45º, 90º, 135º y la

distancia interpixelar a 1,2,3 y 4, además se estableció construir un entrenador supervisado para establecer una clasificación y posteriormente la identificación del cultivo, es decir, *machine learning*.

Para ello se propuso establecer la integración de los datos mediante dos técnicas.

1. Guardado de los datos mediante archivos.
2. Guardado de los datos mediante base de datos, para ello se estableció el diseño y desarrollo mediante modelo-entidad, así como todos los requisitos necesarios para poner en práctica la creación y puesta en marcha.

   Las tablas se establecieron llamar “data”, “image”, “histogram”.

En cuanto al establecer la clasificación se propuso, inicialmente tener dos clases a explorar, esperando poder desarrollar estas clases para futuros proyectos.

- Correspondiente a los datos “es cultivo”, en este caso calabacín.
- Correspondiente a los datos “no es cultivo”, es decir, no es calabacín.

Se fijó integrar los datos al entrenador mediante supervisión, estableciendo si el segmento calculado de la imagen es o no un calabacín.

En conclusión, se estableció integrar los datos obtenidos en la base de datos de tres formas:

- Integración de datos a la base de datos, pero no al entrenador y sin pertenecer a ninguna clase.
- Integración de datos al entrenador y con pertenecía a la clase de calabacín.
- Integración de datos al entrenador y con pertenencia a la clase de no es un calabacín.

Se procedió a actualizar la vista a usuario y a realizar pruebas de la integración de datos.

2.1.3.8 Fase 6: identificación de cultivo mediante distancia

Tras haber superado las pruebas oportunas, Se produjo a reproducir la identificación del cultivo mediante técnicas de distancia.

Esta se añadió como “comprobar” en la visualización de la interfaz del usuario.

Para la realización de esta función se procedió a mejorar la base de datos, añadiendo una función trigger, con la integración y actualización de las medias aritméticas correspondientes a todos los descriptores de la matriz de coocurrencia de cada clase, dando así una nueva tabla en la base de datos cuyo nombre es “means”.

Ahora para poder explicar las técnicas de distancia seleccionadas, primero hemos de explicar que se pretende hacer.

Al obtener varios datos de diferentes fragmentos de la imagen, y tales fragmentos están clasificados en diferentes clases. Se calcula las medias de todos los descriptores de cada clase, y se obtiene la distancia estándar del sí y del no respecto a los descriptores del fragmento de imagen a probar.

Para concluir si el fragmento de imagen a probar corresponde a una clase u otra.

2.1.3.9 Distancia Euclídea

La distancia Euclídea es una de las técnicas más utilizadas y sencillas a la hora de comprender y obtener información.

$$d(x,y) = \sqrt{\sum_{i=0}^{N-1} (y_i - x_i)^2}$$

donde:

x: corresponde a los datos de la clase en cuestión.

y: corresponde a los datos del fragmento prueba.

i: es el número i-ésimo de lo descriptores.

N: número total de descriptores.

Así obtenemos la distancia tanto del fragmento prueba con la clase si y la distancia del fragmento prueba con la clase no.

Y a continuación se decide a qué clase pertenece dependiendo de la cercanía de sus descriptores.

Toda la información utilizada para la integración de la distancia Euclídea al proyecto corresponde a los puntos [8] [9] [10] y [11] de la bibliografía.

2.1.3.10 Distancia Mahalanobis

La distancia Mahalanobis es una técnica más complicada a la hora de programarla ya que necesita del cálculo de la matriz de varianza.

$$d(x_1, x_2) = \sqrt{\left(\frac{(x_{11}-x_{12})}{\sigma_1}\right)^2 + \left(\frac{(x_{21}-x_{22})}{\sigma_2}\right)^2}$$

Toda la información utilizada para la integración de la distancia Mahalanobis al proyecto corresponde a los puntos [8] [9] [10] y [12] de la bibliografía.

2.1.3.11 Fase 7: comprobación y funcionamiento

Tras haber implementado y actualizado el prototipo para todos los posibles estados, se hace la comprobación o validación, con la distancia Euclídea y Mahalanobis llegando a conclusiones explicadas en el apartado 3.

2.2. Análisis de herramientas

2.2.1 Sistema operativo

Hay tres sistemas operativos: Windows, Linux y Mac. Cada uno con unas características diferentes. En nuestro caso teníamos acceso a los tres sistemas operativos, Andres a Mac, Linux (virtualizado) y Windows (virtualizado) y Javier a Windows y Linux. Debido a que Javier no tenía acceso a Mac, descartamos ese sistema operativo.

En cuanto a los dos que nos quedan, nos decantamos por Microsoft Windows 10 debido a su compatibilidad con la mayoría de los softwares existentes y porque habíamos decidido usar un IDE específico que no tenía soporte en Linux

2.2.2 IDE

Estuvimos analizando 2 opciones de IDE para el desarrollo del proyecto: Xamarin Studio 6.3 y Microsoft Visual Studio 2015.

Xamarin Studio es una IDE desarrollado como software libre enfocada a los desarrollos .NET. Dicha solución era nueva para nosotros. Después de probar unos proyectos de prueba en él, no nos encontrábamos cómodos en el desarrollo. Por lo tanto pasamos a probar el Microsoft Visual Studio 2015.

El Microsoft Visual Studio 2015 es un IDE desarrollado por Microsoft con licencia de pago. Como alumnos de la complutense teníamos acceso a una licencia para cada uno, gracias al acuerdo entre la Universidad Complutense de Madrid y Microsoft. Este hecho y el haber usado versiones anteriores en la carrera, nos hizo sentirnos más cómodos en el entorno de desarrollo.

Por estos motivos hemos elegido como IDE el Microsoft Visual Studio 2015.

2.2.3 Control de Versiones

Teníamos 2 opciones en el control de versiones:

- GIT
- SVN

Empezamos probando el GIT, que es un sistema de control de versiones diseñado por Linus Torvalds y de código abierto. Hicimos varias pruebas con él, usando sus funcionalidades principales (fetch, merge, pull, commit, checkout...) y nos resultó complicado su uso de primeras.

Luego probamos el SVN, que es una solución de Apache para el control de versiones y

también de código abierto. En esta solución, Javier tenía experiencia de uso con ella. Y tras un rato de uso a Andrés le resultó también cómoda de uso. Usamos un cliente SVN llamado TortoiseSVN, gratuito y de código abierto, que se implementó como una extensión al shell de windows y se integra en el menú del botón derecho del ratón.

2.2.4 Base de datos

Como decidimos diseñar un base de datos relacional, buscamos opciones que se adaptaran al IDE y al lenguaje SQL. Nos centramos en dos opciones: MySQL y SQL Server.

Primero probamos MySQL, un sistema de gestión de bases de datos diseñado por Oracle y open source. Este sistema nos obligaba a tener un servidor MySQL. Probamos a montar un servidor MySQL sobre una versión WAMP (Windows Apache MySQL PHP). Esto podía ser estable de cara a versiones iniciales en desarrollo, pero después se necesitaría un servidor externo. Esto nos llevó a probar la siguiente opción.

SQL Server es la solución de servidor en lenguaje SQL de Microsoft. Investigando en la funcionalidades de Microsoft Visual Studio 2015, nos permitía generar una base de datos basada en SQL Server en un archivo .mdf e integrarla en el proyecto. Esta opción, nos permitía evitar usar un servidor externo para almacenar la base de datos y ganar rapidez en todo lo relacionado con la base de datos.

Debido a todo lo anterior decidimos usar la segunda opción, SQL Server.

2.2.5 Lenguaje de programación

En la primera reunión con María y Martín nos propusieron realizar el proyecto en el lenguaje C#. En mente, nosotros teníamos como opciones Java o C++ porque los habíamos usado anteriormente durante la carrera.

Tras la reunión empezamos a mirar documentación y código realizado con C# para familiarizarnos con él y comprobar si podíamos realizar el proyecto cómodamente.

Comprobamos que C# era parecido a JAVA en cuanto a la manera de usarlo y nos resultó cómodo adaptarnos a él. Tras estas pruebas decidimos aceptar la propuesta y desarrollar el proyecto con C#.

Estos son los aspectos más destacados de C# que nos hicieron desarrollar el proyecto con él:

- Lenguaje orientado a objetos
- Declaración en espacio de nombres, permitiendo definir más de una clase dentro de un mismo espacio de nombre.
- Por control de versiones ya que permite mantener múltiples versiones en forma binaria.
- Eliminación de uso puntero.
- Despreocupación de cabeceras .h
- No existencias de dependencias circulares.
- Por ser un lenguaje nuevo para aprender.
- Por ser uno de los lenguajes más utilizados en ámbito empresarial e investigación.

2.3. Arquitectura software

El diseño general de nuestro prototipo viene distinguido en el siguiente Figura 1, donde se pueden apreciar las comunicaciones entre los diferentes "paquetes" y el patrón vista-modelo.controlador que conforman el prototipo.

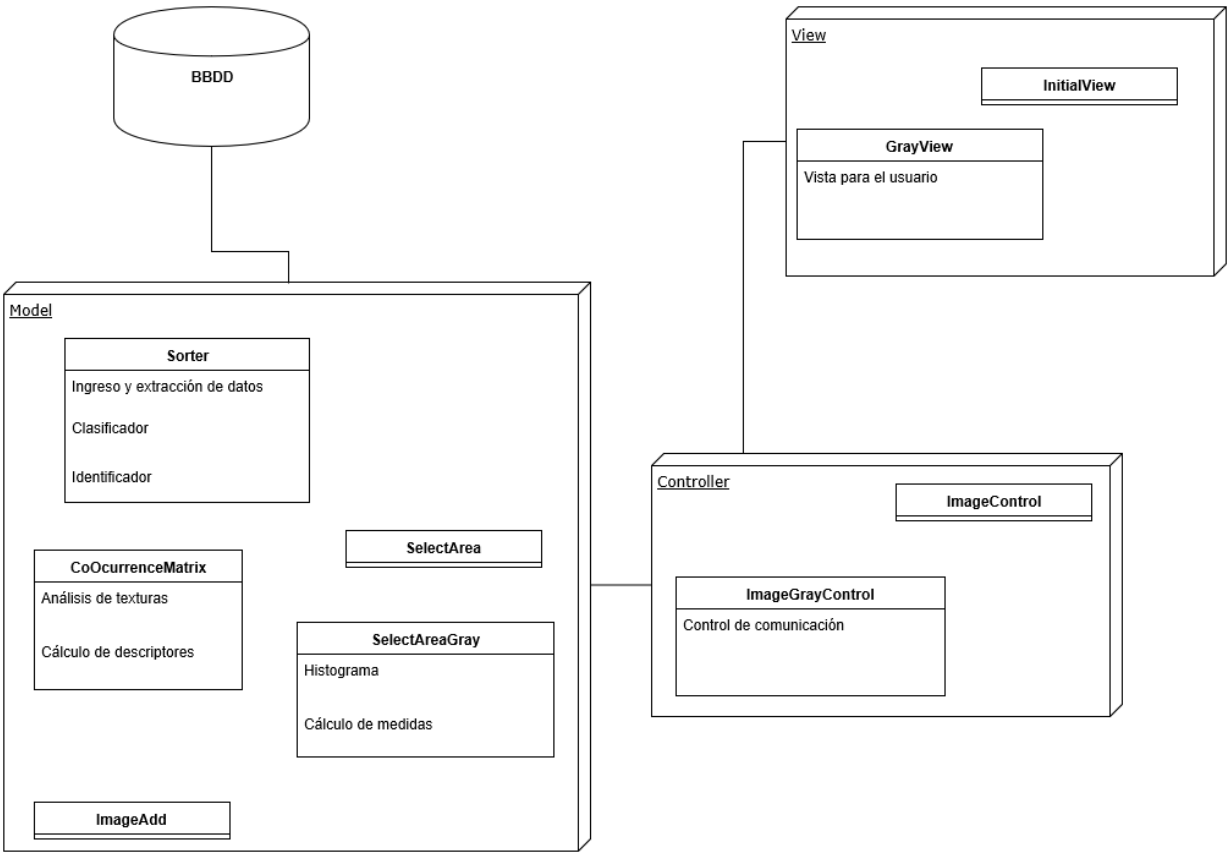

Figura 1. Arquitectura software de la aplicación.

2.3.1. Diseño de clases

El proyecto está diseñado mediante el seguimiento del patrón modelo-vista-controlador, este patrón está implementado para separar los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

El **Modelo,** contiene la representación y manejo de datos de nuestro prototipo y los cálculos realizados.

Este modelo contiene las siguientes clases:

- **coOcurrenceMatrix.cs**, destinada a realizar todos los cálculos correspondientes a la matriz de coocurrencia, al análisis de textura y descriptores, se puede apreciar su diseño

en la Figura 2.

También tiene integrada código de tiempo de ejecución para analizar el rendimiento de los cálculos obtenidos.

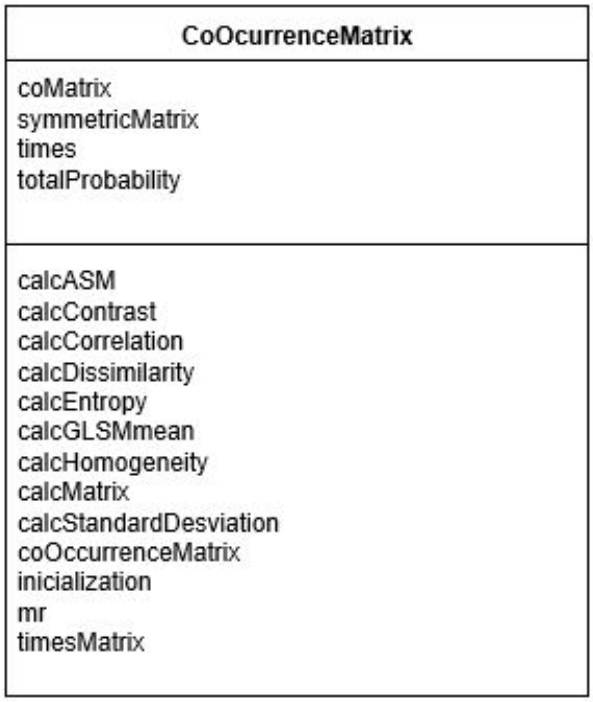

Figura 2. Diseño de la clase coOcurrenceMatrix con sus atributos, propiedades y métodos.

- **ImageAdd.cs**, es utilizada para tener información de la imagen original, realizar las funciones asignadas mostradas con la clase InitialView, como la identificar de los puntos de coordenadas al fragmento seleccionado de la imagen original. Se puede apreciar su diseño en la Figura 3.

  Esta clase crea objetos de las clases:

  1. Image, ya que la imagen integrada corresponde a esta clase.
  2. SelectedArea, para identificar el área seleccionada mediante puntos de coordenadas iniciales y finales.

3. Rectangle, para poder dibujar el rectángulo en la imagen original.

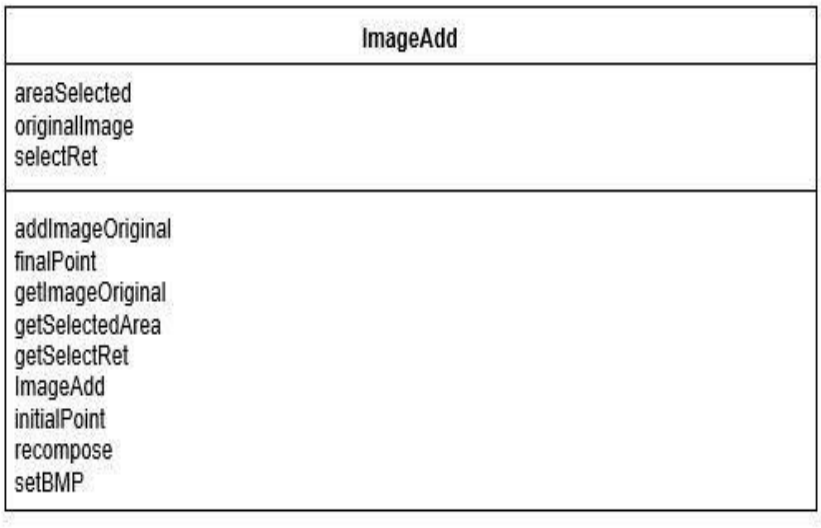

Figura 3. Diseño de la clase ImageAdd
con sus atributos, propiedades y métodos.

- **SelectArea.cs**, sirve para identificar el fragmento de imagen seleccionado para pasarla a escala de grises, además de conservar toda la información necesaria de esta. Se puede apreciar su diseño en la Figura 4.

  Esta clase crea objetos de la clase:

  1. Bitmap, mapa de bits tanto del fragmento seleccionado en RGB y en escala de grises.
  2. SelectAreaGray, para obtener todos los datos calculados en el área del fragmento en escala de grises.
  3. Rectangle, para poder dibujar un rectángulo dentro del fragmento de imagen en escala de grises, cuya área es la utilizada para realizar todos los cálculos.

Figura 4. Diseño de la clase SelectArea con sus atributos, propiedades y métodos.

- **SelectAreaGray.cs**, sirve para identificar el fragmento de imagen con el cual se van a realizar los cálculos necesarios tanto del histograma como de la matriz de coocurrencia. Se puede apreciar su diseño en la Figura 5.

  Esta clase realiza tanto el histograma como la media, entropía y varianza.

  Esta clase crea objetos de las clases:

1. Bitmap, ya que es el mapa de bits en escala de grises.
2. CoOcurrenceMatrix, ya que es la clase destinada a realizar las matrices de coocurrencia, así como los descriptores.

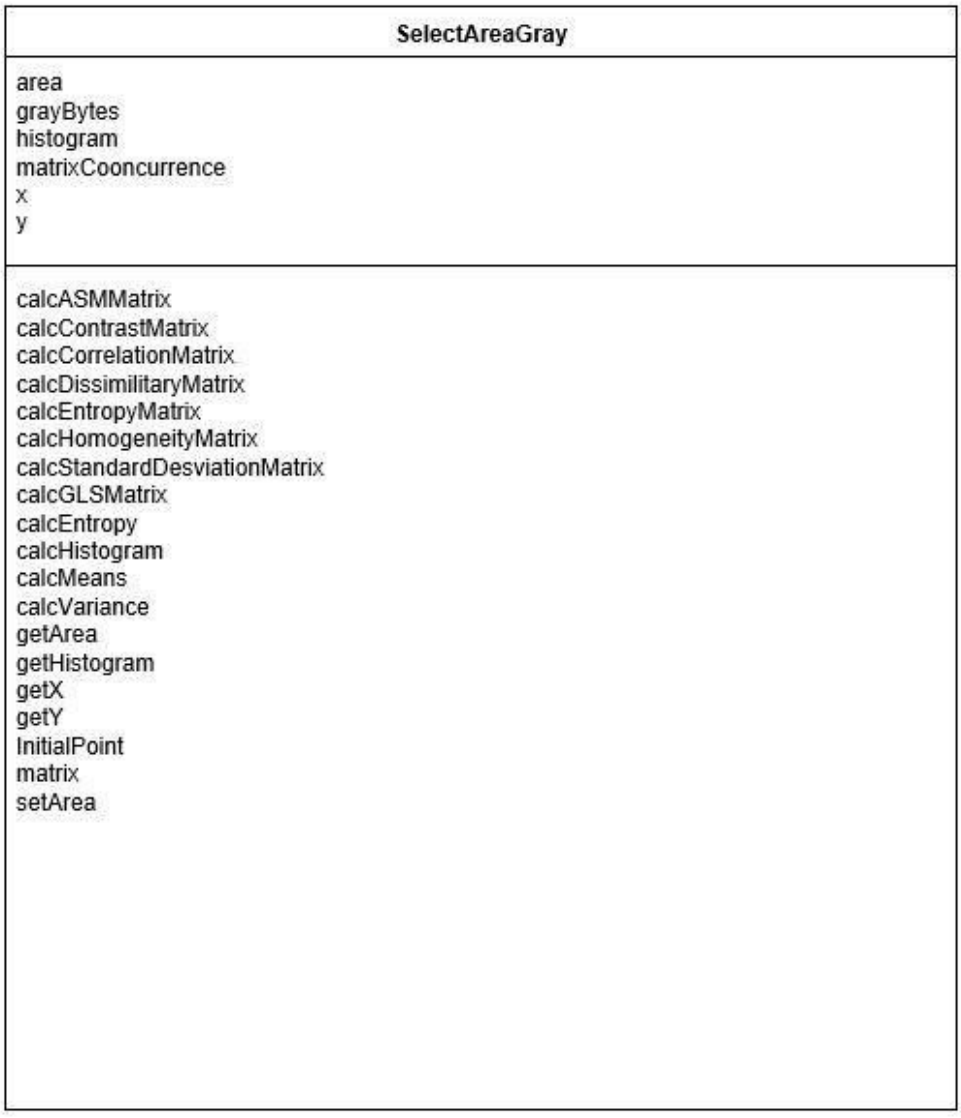

Figura 5. Diseño de la clase SelectAreaGray con sus atributos, propiedades y métodos.

- **Sorter.cs**, clase cuyo destino es establecer la conexión con la base de datos, intercambiar

datos con ella y realiza los algoritmos correspondientes a la distancia Euclídea y mahalanobis. Se puede apreciar su diseño en la Figura 6.

Esta clase crea objetos de la clase:

1. SqlConnection, para establecer la comunicación con la base de datos.
2. SqlCommand, para realizar un procedimiento o instrucción que se ejecuta en la base de datos.
3. SqlDataReader, donde proporciona una forma de leer las filas pertenecientes a la base de datos.

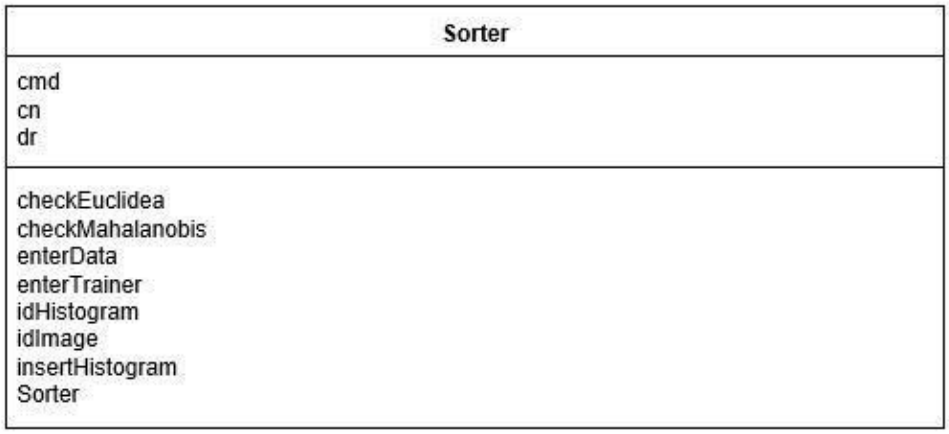

Figura 6. Diseño de la clase Sorter
con sus atributos, propiedades y métodos.

La **Vista**, contiene la interfaz del usuario, donde el usuario podrá visualizar la información que sea necesaria e interactuar con ella.

La vista está compuesta por:

- **InitialView.cs**, donde se visualiza al usuario la ventana inicial, con su correspondiente función de "añadir imagen", "seleccionar toda la imagen", "añadir la imagen a gris" y la visualización de la imagen original en RGB.

Esta clase es diseñada mediante herencia de la clase Form. Se puede apreciar su diseño en

la Figura 7.

Para poder concluir con todas estas funciones, la clase crea un objeto de la clase:

1. ImageControl, donde le manda todas las funcionalidades que necesita realizar.

Figura 7. Diseño de la clase InitialView
con sus atributos, propiedades y métodos.

- **grayView.cs,** destinada a visualizar al usuario, el fragmento de imagen seleccionada a escala de grises, al poder trabajar con esta área mediante la selección de diferentes

fragmentos, donde podrá realizar distintas funcionalidades como calcular su textura, visualizar los datos obtenidos, añadir si procede al entrenador y chequear a qué clase pertenece, es decir, identificarla.

Esta clase es diseñada mediante herencia de la clase Form. Se puede apreciar su diseño en la Figura 8.

Esta clase crea objeto de la clase:

1. ImageGrayControl, donde le manda todas las funcionalidades que necesita realizar.

Figura 8. Diseño de la clase grayView
con sus atributos, propiedades y métodos.

El **Controlador**, cuyo cometido es ser el intermediario entre el Modelo y la Vista, gestionando la conversación entre ellas y adaptando los datos que necesitan cada uno.

El controlador está compuesto por:

- **ImageController.cs**, destina a gestionar la comunicación entre las clases InitialView e ImageAdd. Se puede apreciar su diseño en la Figura 9.

  Esta clase crea un objeto de la clase ImageAdd.

Figura 9. Diseño de la clase ImageControl con sus atributos, propiedades y métodos.

- **ImageGrayController.cs**, cuya funcionalidad es gestionar la comunicación entre la clase grayView con todas las clases necesarias del modelo. Se puede apreciar su diseño en la Figura 10.

  Esta clase crea objetos de la clase:

  1. SelectArea, donde mandará las operaciones que habrá que realizar para la obtención de los resultados de los cálculos.
  2. Sorter, donde mandará las operaciones que habrá que realizar con la base de datos y el entrenador.

Figura 10. Diseño de la clase ImageGrayControl
con sus atributos, propiedades y métodos.

A continuación, en la Figura 11, aportamos el diseño de todas las clases conjuntas, así como su interacción entre ellas.

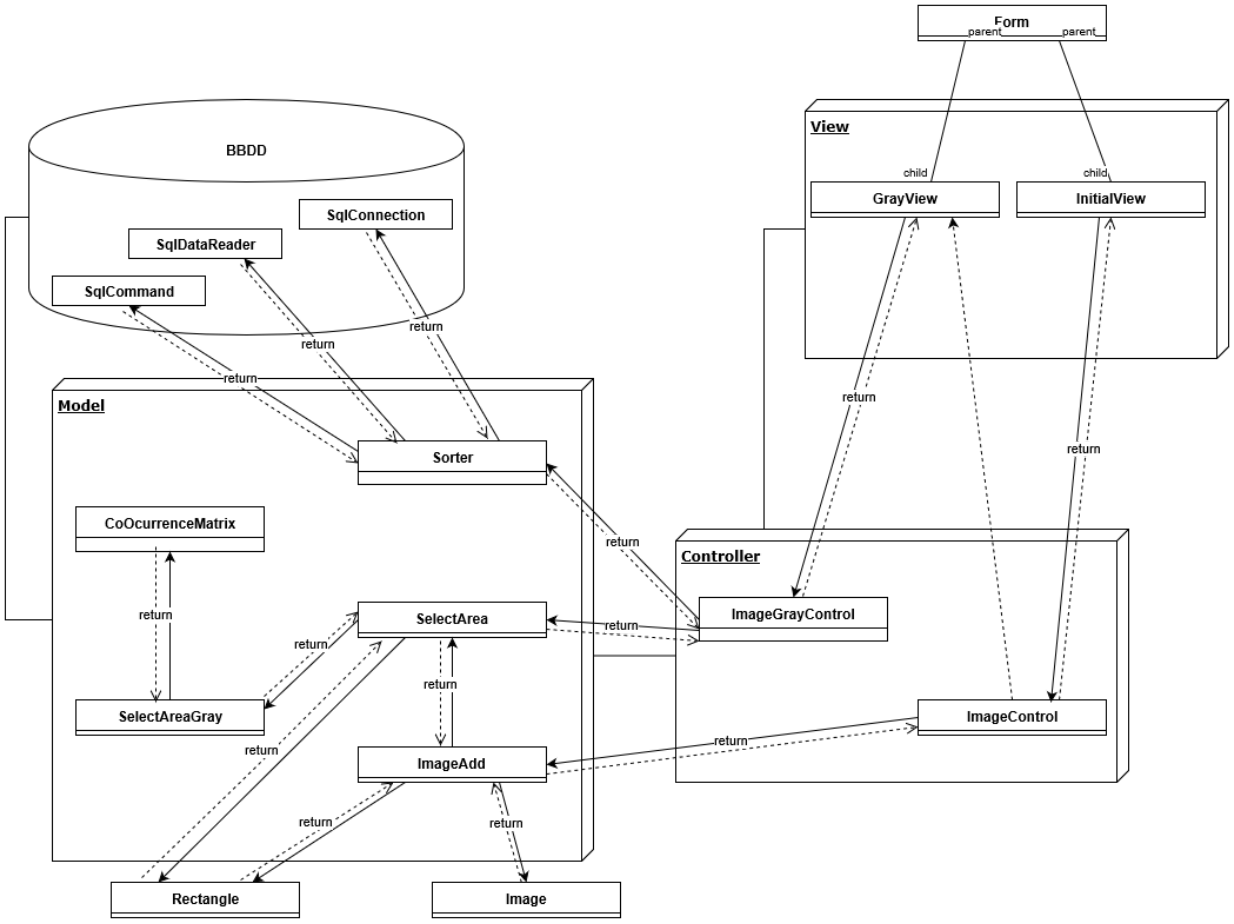

Figura 11. Diagrama de la estructura de la aplicación con sus clases y comunicación.

2.3.2. Base de datos

En un primer momento planteamos la solución de almacenar los datos calculados en archivos de texto con el siguiente formato:

"Coordenadas respecto a la imagen original: (x, y)

Ancho de la selección de la imagen: x

Alto de la selección de la imagen: y

Varianza: x

Entropía: x

Media: x

Histograma:

Nivel de gris - Cantidad

0: x

1: x

2: x

….

255: x

Matriz de coocurrencia:

Grado: x Distancia: x

ASM: x

Contraste: x

Dissimilarity: x

Entropía: x

GLS: x

Homogeneity

Desviación Estándar: x

Correlación: x”

En el cual, almacenamos los datos de referencia de la parte seleccionada de la imagen respecto de la original, los datos calculados respecto a la imagen en gris (varianza, entropía, media, histograma), los datos de cálculo de la matriz de coocurrencia (grado y distancia) y las

características calculadas sobre la matriz (ASM, Contraste, Dissimilarity, Entropía, GLS, Homogeneity, Desviación estándar y correlación).

Según iba pasando el desarrollo de la aplicación e iban evolucionando las características de la misma. Nos vimos en la necesidad de cambiar el sistema final de almacenamiento de datos para la utilización de la aplicación, sin embargo, continuamos ofreciendo al usuario la opción de guardar los datos en documentos de texto independientes.

Después de pensarlo detenidamente, decidimos que la mejor solución era diseñar e implementar un base de datos relacional, este diseño se muestra en la Figura 13. Primero identificamos las entidades necesarias para el desarrollo de la aplicación, estas fueron: data, image, histogram y means. En data hemos almacenado todos los datos, menos el histograma, que guardábamos en los archivos de texto que nombramos anteriormente. En image guardamos el nombre de la imagen de la cual se ha hecho la selección para los cálculos. En histogram guardamos el histograma de la imagen. En means se guardan las medias de las características que calculamos, en el caso que almacenemos en la parte de si o en la de no.

Decidimos que la base de datos está implementada en lenguaje SQL Server de Microsoft, dado que una de las utilidades del Microsoft Visual Studio 2015 nos permitía generar un archivo .mdf e incrustarlo en el proyecto para una gestión más rápida de los datos. Con esta solución desechamos la posibilidad de necesitar un servidor externo de mysql para realizar las consultas a la base de datos.

En la siguiente imagen podemos observar cómo nos quedó diseñado el esquema entidad-relación del diseño de la base de datos:

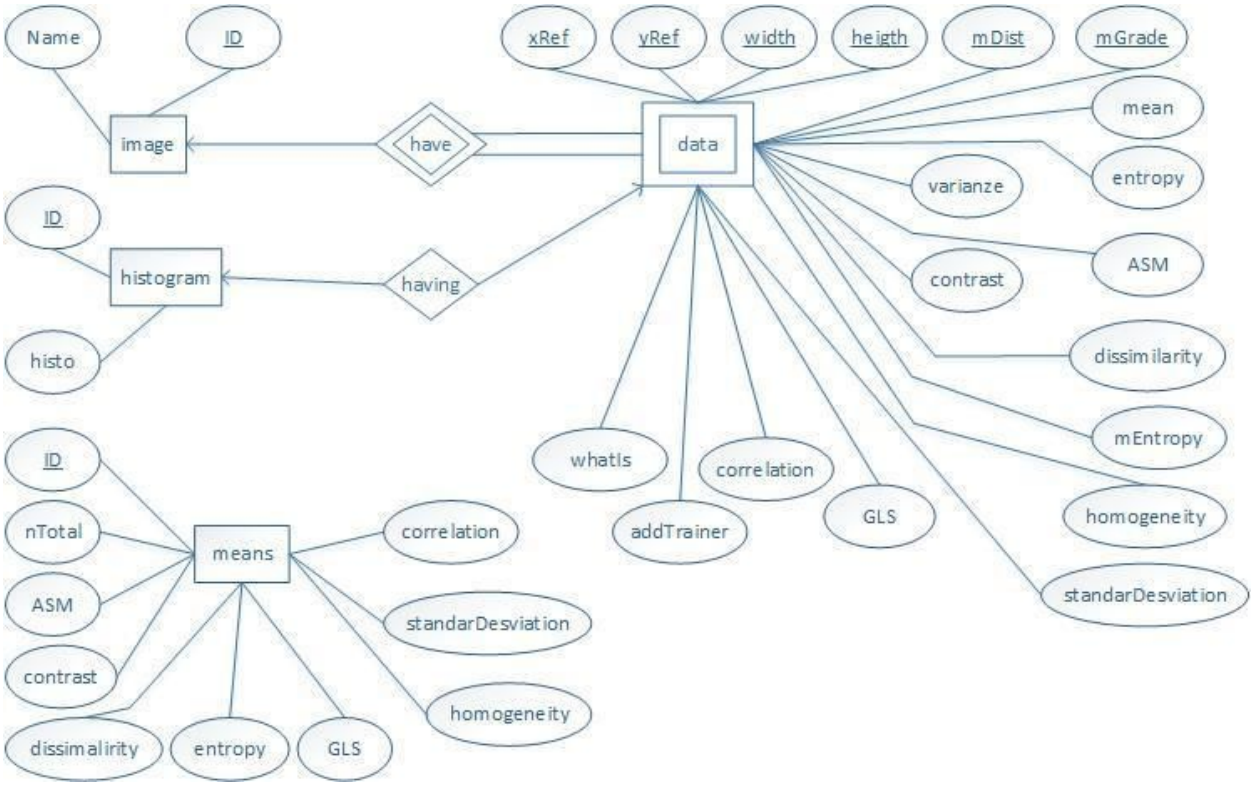

Figura 12. Diagrama entidad-relación de la base de datos.

En este esquema entidad-relación podemos observar que la entidad data es una entidad débil respecto a image y que está relacionada en relación 1:1 con histogram. La única entidad que quedó separada en el diagrama fue means, al ser un cálculo respecto a la entidad data.

Después de acordar este diseño entidad-relación, hemos transformado el mismo en álgebra relacional , como se muestra en la Figura 13.

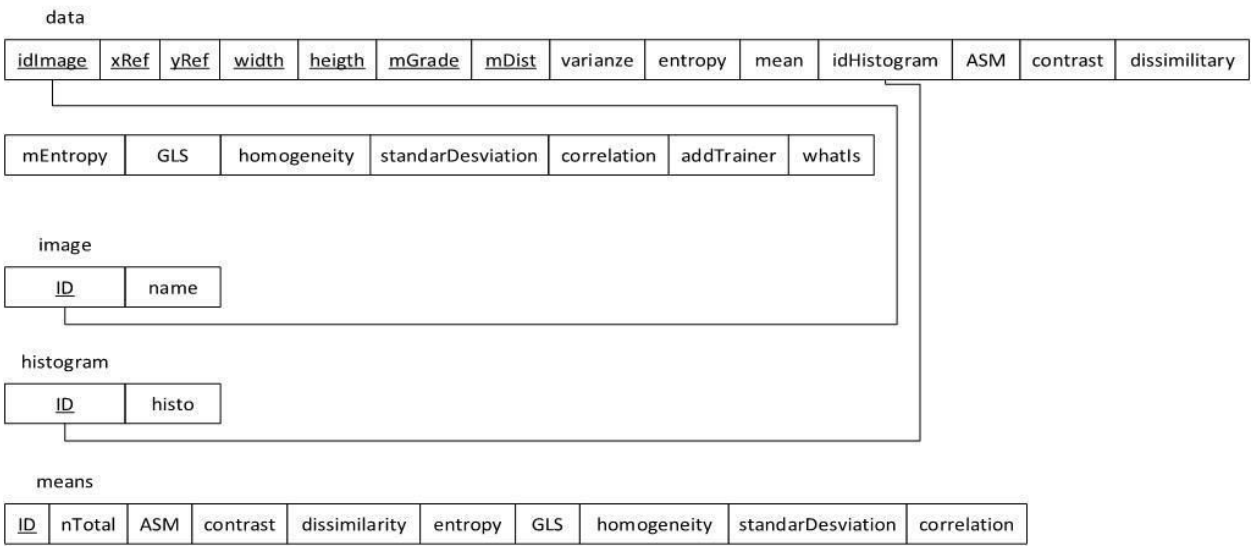

Figura 13. Diseño del modelo relacional de la base de datos.

Tras este esquema hicimos la traducción a tablas del esquema de álgebra relacional, quedándonos las siguientes tablas, mostradas en la Figura 14.

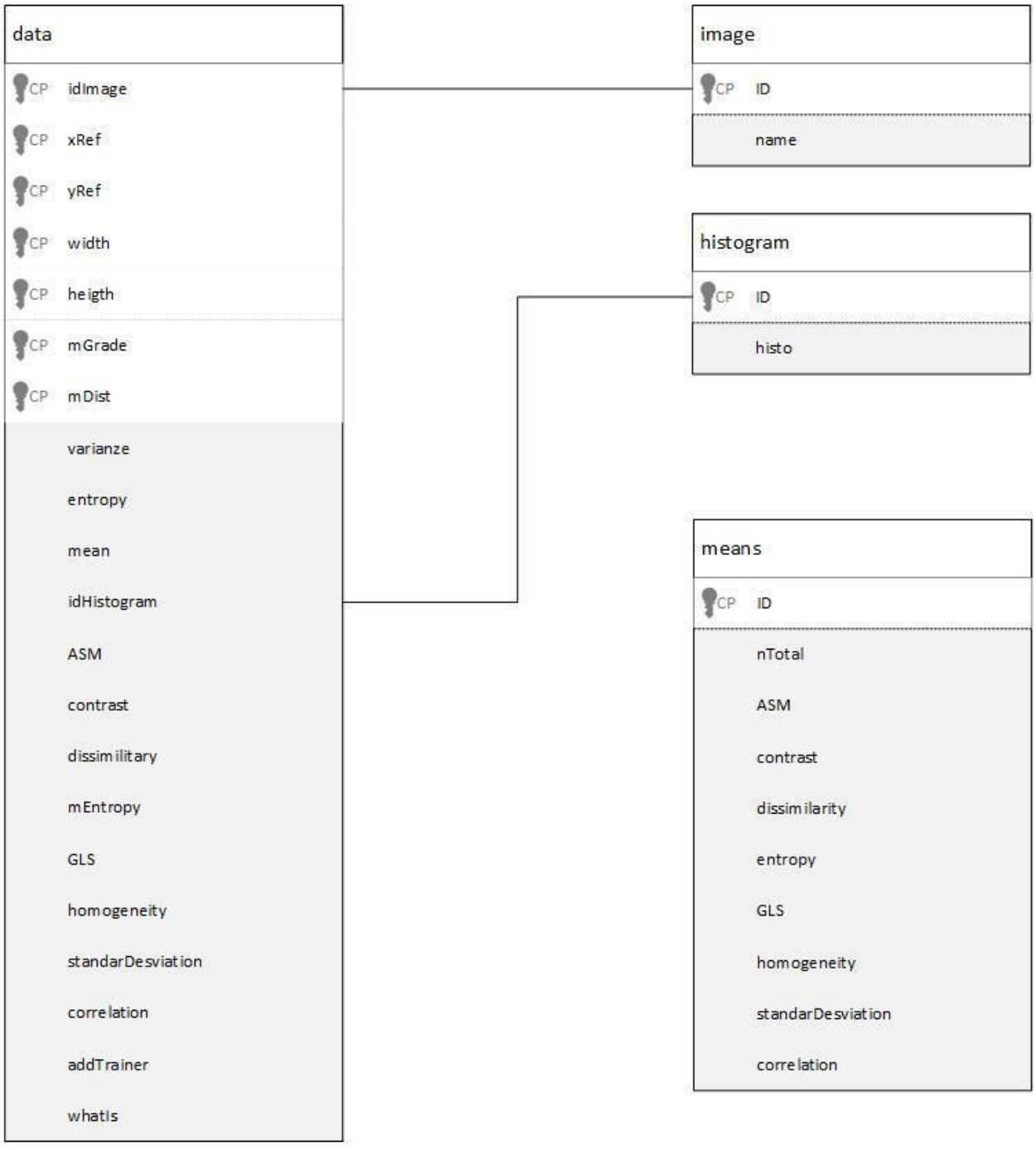

Figura 14. Tablas de la base de datos.

La tabla means decidimos actualizarla mediante un trigger en la tabla data. Este trigger

salta cuando se inserta una fila en la tabla data. Dicho trigger está programado así:

```
"CREATE TRIGGER [dbo].[updateTrainer]
   ON [dbo].[data]
   AFTER INSERT
   AS
   BEGIN
        DECLARE
             @trainer int,
             @tipo int
        SELECT @trainer = [addTrainer], @tipo = [whatIs] FROM inserted
        IF @trainer = NULL
             RETURN
        DECLARE
             @ASM float,
             @contrast float,
             @dissimilarity float,
             @entropy float,
             @GLS float,
             @homogeneity float,
             @standarDesviation float,
             @correlation float,
             @mASM float,
```

@mcontrast float,

@mdissimilarity float,

@mentropy float,

@mGLS float,

@mhomogeneity float,

@mstandarDesviation float,

@mcorrelation float,

@nTotal int

SELECT @ASM = ASM, @contrast = contrast, @dissimilarity = dissimilarity, @entropy = entropy, @GLS = GLS, @homogeneity = homogeneity, @standarDesviation = standarDesviation, @correlation = correlation FROM inserted;

SELECT @mASM = ASM, @mcontrast = contrast, @mdissimilarity = dissimilarity, @mentropy = entropy, @mGLS = GLS, @mhomogeneity = homogeneity, @nTotal = nTotal, @mstandarDesviation = standarDesviation, @mcorrelation = correlation FROM [dbo].[means] WHERE ID = @tipo;

SET @mASM = ((@mASM * @nTotal) + @ASM)/(@nTotal + 1);

SET @mcontrast = ((@mcontrast * @nTotal) + @contrast)/(@nTotal + 1);

SET @mdissimilarity = ((@mdissimilarity * @nTotal) + @dissimilarity)/(@nTotal + 1);

SET @mentropy = ((@mentropy * @nTotal) + @entropy)/(@nTotal + 1);

SET @mGLS = ((@mGLS * @nTotal) + @GLS)/(@nTotal + 1);

SET @mhomogeneity = ((@mhomogeneity * @nTotal) + @homogeneity)/(@nTotal + 1);

SET @mstandarDesviation = ((@mstandarDesviation * @nTotal) + @standarDesviation)/(@nTotal + 1);

SET @mcorrelation = ((@mcorrelation * @nTotal) + @correlation)/(@nTotal + 1);

UPDATE [dbo].[means] SET ntotal = @nTotal+1, ASM = @ASM, contrast = @mcontrast, dissimilarity = @mdissimilarity, entropy = @mentropy, GLS = @mGLS, homogeneity = @mhomogeneity, standarDesviation = @mstandarDesviation, correlation= @mcorrelation WHERE ID = @tipo;

END”

Capítulo 3. Resultados y conclusiones

3.1. Resultados

3.1.1 Resultado de los datos

Tras haber expuesto el material necesario y la construcción del prototipo, se han obtenido diferentes resultados dependiendo del sector de la imagen tratado y argumentos establecidos.

En las siguientes tablas se exponen ejemplos de cada uno de los cuatro sectores pertenecientes que conforman la imagen, calabacín, flor, tallo y tierra.

En cada una de ellas viene descrito:

- ➢ Tipo de sector de la imagen.
- ➢ El nombre de la imagen original.
- ➢ Representación del fragmento de imagen mediante puntos de coordenadas, anchura y altura respecto a la imagen original.
- ➢ Media, entropía y varianza del histograma a escala de grises del fragmento de la imagen seleccionado.
- ➢ Información de la componente espacial de la matriz de coocurrencia.
- ➢ Resultado de los descriptores de la matriz de coocurrencia.

Calabacín	Tierra	Flor	Tallo
Imagen: maxresdefault.jpg Coordenadas respecto imagen original: (622, 100)	Imagen: maxresdefault.jpg Coordenadas respecto imagen original: (870, 127)	Imagen: maxresdefault.jpg Coordenadas respecto imagen original: (514, 83)	Imagen: maxresdefault.jpg Coordenadas respecto imagen original: (198, 118)

Ancho: 158 Alto: 311 Varianza: 27,8855242417859 Entropía: 6,12538992345737 Media:168,737372298425 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,00177884435156346 Contraste: 21,1168820529625 Disimilaridad: 3,21330411452678 Entropía: 7,01395897229399 GLS: 170,621377516538 Homogeneidad: 9,2242416880238E-06 Desviación Estándar: 17759,5010572713 Correlación: 0,999999966523634	Ancho:310 Alto: 347 Varianza: 33,5685548988468 Entropía: 6,90863822829888 Media:194,20990982616 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,000273987391787067 Contraste: 155,601494082427 Disimilaridad: 9,20180371748599 Entropía: 8,56551372487689 GLS: 195,267111533906 Homogeneidad: 7,18385042517039E-06 Desviación Estándar: 13013,5989203046 Correlación: 0,999999540602172	Ancho: 71 Alto:87 Varianza: 26,8458531024287 Entropia: 5,87151804317694 Media: 135,277642868706 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,00162003445849207 Contraste: 29,504105090312 Disimilaridad: 3,95829228243021 Entropía: 6,94041550237081 GLS: 138,937274220033 Homogeneidad: 1,3373574744148E-05 Desviación Estándar: 22813,5820320142 Correlación: 0,999999971655715	Ancho: 32 Alto: 102 Varianza: 40,6445476835551 Entropía: 6,09111212800808 Media: 200,554534313725 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,000878404247785903 Contraste: 73,7128399746994 Disimilaridad: 6,92093611638204 Entropía: 7,27903352732044 GLS: 207,408918406071 Homogeneidad: 5,97300917340515E-06 Desviación Estándar: 10857,7785592593 Correlación: 0,999999687369615

Tabla 4. Ejemplo resultado de datos

Calabacín	Tierra	Flor	Tallo
Imagen: Calabacin0001 Coordenadas respecto imagen original: (1235, 228) Ancho: 77 Alto: 210 Varianza: 35,6568642016706 Entropía: 6,92773691419223 Media: 139,560420531849 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,000267627480355017 Contraste: 146,890476190475 Disimilaridad: 9,20526315789475 Entropía: 8,47899666275373 GLS: 141,3063283208 Homogeneidad:	Imagen: Calabacin0001 Coordenadas respecto imagen original: (22, 1463) Ancho: 203 Alto: 159 Varianza: 31,1695482037427 Entropía: 6,94119849680719 Media:98,750224618149 1 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,000261611006840506 Contraste: 154,821968989351 Disimilaridad: 9,15984806027738 Entropía: 8,55994294143141 GLS: 99,7665794881358 Homogeneidad:	Imagen: Calabacin0001 Coordenadas respecto imagen original: (639, 474) Ancho: 117 Alto: 73 Varianza: 46,7151170840322 Entropía: 7,27751643484528 Media:156,236623346212 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,000716498846751764 Contraste: 15,1313179026925 Disimilaridad: 2,92087860179499 Entropía: 7,59531838181681 GLS: 158,123405762872 Homogeneidad:	Imagen: Calabacin0001 Coordenadas respecto imagen original: (614, 73) Ancho: 78 Alto: 165 Varianza: 31,8870688502205 Entropía: 6,02688297488986 Media:122,207847707848 Matriz de coocurrencia: Grado: 0 Distancia: 1 ASM: 0,00296250349057046 Contraste: 4,01109799291617 Disimilaridad: 1,58244785517513 Entropía: 6,12205528854614 GLS: 125,004919323101 Homogeneidad: 1,81829637338449E-05

1,45502803893038E-05 Desviación Estándar: 22111,5116676805 Correlación: 0,999999849780334	3,77639900420039E-05 Desviación Estándar: 27172,1161704693 Correlación: 0,999999895153073	1,29983632337844E-05 Desviación Estándar: 19090,2410763533 Correlación: 0,999999979240174	Desviación Estándar: 24409,0670169676 Correlación: 0,999999996633871

Tabla 5. Ejemplo resultado de datos

3.1.2 Resultado de la identificación

Tras haber implementado lo correspondiente a la clasificación y entrenador, se realizaron 400 inserciones de fragmentos de imagen de calabacín y otros 400 fragmentos de no calabacín, para realizar el entrenamiento supervisado del entrenador.

Se realizó un muestreo de 50 fragmentos de calabacín, de flores, de tallo y de tierra con el algoritmo de distancia Euclídea obteniendo los siguientes resultados:

- Calabacín:teniendo un ratio de acierto del 86% (43 de 50).
- Flor: teniendo un ratio de acierto del 78% (39 de 50).
- Tallo: teniendo un ratio de acierto del 58% (29 de 50).
- Tierra: teniendo un ratio de acierto del 66% (33 de 50).

Estos resultados nos han demostrado la dificultad de diferenciar el tallo del calabacín, dado que falla un 42% de las veces diciendo que el tallo es calabacín.

3.1.3 Resultado en tiempo de ejecución

Se Midió el tiempo de ejecución del programa mediante algoritmos de la clase TimeSpan, tanto en resultados de datos como en resultados de identificación.

Tales resultados son:

Para el calabacín

tiempo de cálculo matriz en milisegundos 249

tiempo de cálculo ASM en milisegundos 1

tiempo de cálculo contraste en milisegundos 4

tiempo de cálculo dissimilarity en milisegundos 1

tiempo de cálculo entropía en milisegundos 0

tiempo de cálculo GLSmean en milisegundos 0

tiempo de cálculo de homogeneidad en milisegundos 1

tiempo de cálculo deviation en milisegundos 5

tiempo de cálculo correlation en milisegundos 7

tiempo de cálculo de ingreso de datos al trainer en milisegundos 63

Para la tierra

tiempo de cálculo matriz en milisegundos 379

tiempo de cálculo ASM en milisegundos 0

tiempo de cálculo contraste en milisegundos 17

tiempo de cálculo dissimilarity en milisegundos 1

tiempo de cálculo entropía en milisegundos 1

tiempo de cálculo GLS mean en milisegundos 0

tiempo de cálculo de homogeneidad en milisegundos 1

tiempo de cálculo deviation en milisegundos 0

tiempo de cálculo correlation en milisegundos 0

tiempo de cálculo de ingreso de datos al trainer en milisegundos 182`

Para la flor

tiempo de cálculo matriz en milisegundos 37

tiempo de cálculo ASM en milisegundos 0

tiempo de cálculo contraste en milisegundos 0

tiempo de cálculo dissimilarity en milisegundos 0

tiempo de cálculo entropía en milisegundos 0

tiempo de cálculo GLS mean en milisegundos 0

tiempo de cálculo de homogeneidad en milisegundos 0

tiempo de cálculo deviation en milisegundos 15

tiempo de cálculo correlation en milisegundos 0

tiempo de cálculo de ingreso de datos al trainer en milisegundos 15

Para el tallo

tiempo de cálculo matriz en milisegundos 20

tiempo de cálculo ASM en milisegundos 1

tiempo de cálculo contraste en milisegundos 0

tiempo de cálculo dissimilarity en milisegundos 0

tiempo de cálculo entropía en milisegundos 0

tiempo de cálculo GLS mean en milisegundos 0

tiempo de cálculo de homogeneidad en milisegundos 0

tiempo de cálculo deviation en milisegundos 15

tiempo de cálculo correlation en milisegundos 0

tiempo de cálculo de ingreso de datos al trainer en milisegundos 20

Tiempo en la identificación del calabacín

tiempo de cálculo del check con distancia Euclídea en milisegundos 0

tiempo de cálculo de identificación total en milisegundos 115

Tiempo en la no identificación del calabacín

tiempo de cálculo del check con distancia Euclídea en milisegundos 0

tiempo de cálculo de identificación total en milisegundos 6630

3.2. Conclusiones

3.2.1. Conclusiones de los resultados

Tras haber obtenido los resultados se ha concluido de tal forma:

En cuanto a los resultados obtenidos en los datos se concluye que hay pequeñas diferencias entre los cuatros sectores, algunas de estas diferencias son:

- ASM del calabacín suele ser superior al de la tierra.
- El contraste de la tierra suele ser superior al resto de los sectores.

Dejamos la posibilidad de conclusión a expertos en investigación de datos en la imagen.

En cuanto a los resultados obtenidos en la identificación del calabacín, se concluyen resultados efectivos utilizado la distancia Euclídea con la excepción de diferenciación entre la flor y el calabacín, dando a entender que sus texturas son similares.

Con la distancia Mahalanobis no se obtuvieron resultados concluyentes.

En cuanto a los resultados obtenidos en el tiempo de ejecución del programa, se concluye que son asequibles tanto para el cálculo de la matriz, como para el ingreso de datos en el trainer, dando unos valores aproximados en 200 milisegundos al cálculo de la matriz, valores despreciables al cálculo de los descriptores y aproximadamente 100 milisegundos al ingreso de datos.

Y para finalizar, unos valores bastante diferentes en la identificación del calabacín y del no calabacín, aproximadamente en 200 milisegundos y 6000 milisegundos respectivamente, esto

se debe a la comprobación de todas las posibilidades de la matriz de coocurrencia con el trainer.

3.2.2. Conclusiones sobre el proyecto

Se ha concluido que somos capaces de “crear” un prototipo de identificación de cultivos con los conocimientos adquiridos en la universidad, tanto del itinerario de computación como el de tecnologías de información con las siguientes capacidades [13]:

- Capacidad para la resolución de los problemas matemáticos y conceptos de matemática discreta, lógica, algorítmica y complejidad computacional.
- Capacidad para planificar, concebir, desplegar proyectos, así como conocimiento de metodología de la ingeniería del software.
- Capacidad para comprender los hábitos de trabajo efectivos y habilidades de comunicación en todos los entornos de desarrollo de software.
- Capacidad para analizar, diseñar, desarrollar, y evaluar aplicaciones, asegurando su fiabilidad, seguridad y calidad con el lenguaje de programación más adecuado.
- Capacidad de utilizar los procedimientos algorítmicos básicos de las tecnologías para diseñar soluciones al problema.
- Capacidad de utilizar de forma eficiente los tipos y estructuras de datos.
- Capacidad de extraer características, funcionalidades y estructura de las bases de datos, para realizar el diseño, análisis e implementación.
- Capacidad de utilizar las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas.
- Capacidad para formalizar y representar el conocimiento humano en una forma computable para la resolución de problemas mediante un sistema informático en aspectos de computación y percepción.
- Capacidad para desarrollar técnicas de aprendizaje.
- Capacidad de análisis y síntesis en la resolución de problemas.
- Capacidad para gestionar adecuadamente la información disponible y aplicándolos a la resolución de problemas informáticos utilizando el método científico.

Gracias a estas capacidades se han logrado:

- Se ha logrado de forma satisfactoria la investigación y análisis de los procedimientos y métodos en las imágenes para cultivos.
- Se ha logrado establecer una integración de datos robusta y eficaz.
- Se ha logrado establecer el principio de una clasificación de cultivos.
- Se ha logrado identificar parcialmente el cultivo correspondiente al calabacín.
- Se ha logrado obtener unos resultados en tiempo de ejecución asequibles.

3.2.3. Trabajo futuro

Tras marcarnos el alcance del proyecto y dado la posible ampliación del mismo, debido a su vinculación a un proyecto real, hemos definido una serie de posible de propuestas de ampliación del proyecto.

Se plantea investigar e implementar diversas soluciones para la diferenciación de la hoja del calabacín con el calabacín, para aumentar el número de cultivos posibles a los que se pueda enfocar el clasificador, una clasificación del resto de texturas que no forman parte del calabacín, una mejora dentro del clasificador, que mediante un aprendizaje automático, aumente la tasa de acierto cerca del 90% y métodos alternativos de clasificación para poder trabajar conjuntamente con este método de identificación, como la detención de puntos de interés.

3.2. Conclusions

3.2.1. Conclusions of the results

After obtaining the results, we have concluded:

As for the results obtained in the data, it is concluded that there are small differences between the four sectors, some of these differences are:

- ➢ The ASM of the zucchini that is usually higher to the one of the earth.
- ➢ The contrast that is usually the value of the earth higher to the rest of sectors.

We left the possibility of conclusion to data research experts in the images.

As for the results obtained in the identification of the zucchini, effective results were concluded using the Euclidean distance with the exception of the differentiation of the flower and the zucchini, implying that their textures are similar.

With the Mahalanobis distance, no conclusive results were obtained.

As for the results obtained in the execution time of the program, it is concluded that they are available both for the calculation of the matrix and for the input of data in the trainer, giving approximate values in 200 milliseconds to the calculation of the matrix, negligible values to the calculation of the descriptors and approximately 100 milliseconds to the data entry.

And to finish, quite different values in the identification of zucchini and non-zucchini, approximately in 200 milliseconds and 6000 milliseconds respectively, this is due to the verification of all possibilities of the co-occurrence matrix with the trainer.

3.2.2. Conclusions about project

It has been concluded that we are capable to "create" a prototype of crop identification with the knowledge acquired in the university, both of the computing itinerary and of information technologies with the following capacities [13]:

- Ability to solve mathematical problems and concepts of mathematical discrete, logical, algorithmic and computational complexity.
- Ability to plan, conceive, deploy projects, as well as knowledge of software engineering methodology.
- Ability to understand effective work habits and communication skills in all software

development environments.

- Ability to analyze, design, develop, and evaluate applications, ensuring their reliability, security and quality with the most appropriate programming language.
- Ability to use the basic algorithmic procedures of the technologies to design solutions to problem.
- Ability to efficiently use data types and structures.
- Ability to extract characteristics, functionalities and structure of the databases, to carry out the design, analysis and implementation.
- Ability to use the necessary tools for the storage, processing and access to the Systems.
- Ability to formalize and represent human knowledge in a computable way to solve problems through a computer system in aspects of computation and perception.
- Ability to develop learning techniques.
- Capacity for analysis and synthesis in problem solving.
- Ability to adequately manage available information and apply it to solving computer problems using the scientific method.

Thanks to these capabilities, we have achieved:

- Research and analysis of procedures and methods in crop images has been successfully achieved.
- Robust and effective data integration has been achieved.
- The principle of a crop classification has been established.
- It has been possible to partially identify the crop corresponding to the zucchini.
- Affordable runtime results have been achieved.

3.2.3. Future works

After marking the scope of the project and given the possible extension of it, due to its link to a real project, we have defined a series of possible proposals for project extension.

It is proposed to investigate and implement different solutions for the differentiation of

the zucchini sheet with the zucchini, to increase the number of possible cultures to focus the classifier, a classification of the rest of textures that are not part of the zucchini, an improvement Within the classifier, that through automatic learning, increase the success rate near 90% and alternative methods of classification to be able to work in conjunction with this method of identification, such as the detention of points of interest.

Capítulo 4. Aportaciones

4.1 Aportaciones de los miembros

4.1.1 Aportaciones de Roger Andrés Ortiz Londoño

Inicialmente decidimos utilizar un lenguaje de programación nuevo por el que no teníamos conocimiento, por lo que decidí investigar y analizar el lenguaje de programación, así como ver tutoriales de programación en el lenguaje c#.

Se decidió utilizar un repositorio SVN tortoise, en el cual no tenía idea, por lo que decidí investigar su funcionamiento y ventajas, además de descargar e instalar los programas correspondientes al uso de este servicio.

Tras haber aprendido a programar lo básico de la programación, me pase a descargar Visual Studio 2015, así como probar sus funcionalidades y programar pequeños fragmentos de prueba con código c#.

A continuación, me decidí a investigar y aprender las distintas técnicas de procesamiento y tratamiento de imagen, para así comprenderlas y debatirlas a la hora de implementar, algunos de los recursos encontrados y utilizados se encuentran expuestos en la bibliografía, se decidió en un principio utilizar el identificación de puntos de interés mediante SIFT Y SURF, pero al no encontrar suficiente información de la implementación se pasó a realizar mediante matriz de coocurrencia.

Leí diferentes artículos expuestos en la bibliografía para la comprensión de la matriz de coocurrencia y cálculo de los descriptores, cuando lo comprendí me dispuse a programar.

Tras realizar la programación me ubique a realizar el testeo de la programación para

establecer el funcionamiento del programa con la matriz de coocurrencia.

Se analizó las pruebas y se dispuso a rectificar la matriz de coocurrencia para su mejora en el funcionamiento.

Se realizó la segunda prueba de funcionamiento y se pasó satisfactoriamente.

Tras realizado las pruebas, se pasó a discutir el guardado de datos, en el cual se decidió mediante base de datos.

Diseñe una versión de la base de datos para la discusión del grupo y toma de decisión.

Tras haber decidido tanto el diseño de la base de datos y la arquitectura asociada a ella, en este caso SQL server, me preparé para comprenderla mediante artículos y tutoriales, e instalar los correspondientes programas para la puesta en marcha de la base de datos.

Se discutió el diseño de la base de datos para mejorar el funcionamiento de esta.

Se pasó a realizar la identificación del cultivo mediante distancias por lo que leí varios artículos expuestos en la bibliografía para la comprensión e implementación, se destacó la distancia Euclídea y la distancia Mahalanobis.

Implemente la primera versión de la integración de datos en la base de datos, así como la clasificación y distancias Euclídea y Mahalanobis.

Cambie parte del diseño de la vista de usuario para integrar el guardado de la base de datos, en el entrenador y el chequeo de la identificación de cultivo.

Implemente para sacar los tiempos de ejecución de la matriz de coocurrencia y de la integración de la base de datos y chequeo de la identificación.

Y me dispuse a realizar las pruebas.

Investigue la puesta en marcha del documento de la memoria, así como la forma de escribirlo y el contenido que debería de aparecer.

Y para finalizar, realice parte de la memoria ahora descrita.

4.1.2 Aportaciones de Javier Toledano Regaño

Tras mi paso por una empresa de desarrollo de juegos para móvil, entendí la importancia del uso de algún programa de gestión de versiones. Sólo conocía 2 soluciones GIT y SVN. Tras mirar ambas soluciones, el SVN me pareció de uso muy sencillo y con bastantes soluciones gratuitas para el uso del mismo. Tras mirar unas cuantas soluciones para el tema del servidor SVN, encontré la web RIOUXSVN.com. Dicha web nos daba la facilidad de con un usuario gratuito, gestionar 5 repositorios sin tener que pagar. Para la gestión del cliente SVN me decidí por TortoiseSVN, una solución gratuita que se integra en Windows perfectamente. Se lo comenté a Andrés y le pareció una muy buena idea.

Tras la decisión de implementar el proyecto en el lenguaje C# y el acceso que tenemos a la tienda Microsoft por ser alumnos de la complutense. Tomé la decisión de probar el Microsoft Visual Studio 2015. Tras unos días probando la funcionalidad básica para iniciar los proyectos en el Visual Studio, decidí decírselo a Andrés para que lo probara también. Tras probarlo acordamos utilizar la solución de Microsoft para desarrollar el proyecto. Una idea bastante acertada al ser C# un lenguaje creado por Microsoft y el Visual Studio creado para gestionar proyectos de sus lenguajes principalmente, acepta otros tipos de lenguaje.

Tras la primera reunión de cómo debía de ser la aplicación, me puse a realizar un pequeño diseño de la primera interfaz de la aplicación. Esta estaba dividida en 3 partes y se mostraba todo en esa vista. Era una vista muy sobrecargada. Tras otra reunión con María y Martín, se discutió la posibilidad de dividir la interfaz en dos vistas diferentes. Tras confirmar esa opción, me puse a rediseñar la vista inicial y diseñar la segunda vista.

Tras diseñar estas dos vistas, sin funcionalidad interna, empecé a diseñar y desarrollar el Modelo Vista Controlador de la aplicación. Este patrón acabó con dos vistas (la principal y la de la imagen en gris), 2 controladores (uno para cada vista, aunque el controlador de la vista principal instancia a la segunda vista) y 5 clases en el modelo. La funcionalidad de este patrón trata de que la vista solo interactúe con el controlador y que el modelo solo interactúe con el

controlador, sin que el modelo conozca nada de la vista y viceversa.

Empecé a investigar en internet como hacer el cambio a escala de grises de una imagen RGB. Tras ver algunas opciones de cómo realizarlo, tomé la decisión de optar por una adaptación de un código que estaba en la ayuda de Microsoft (14). El cual se adaptaba bastante a las necesidades del proyecto y nos ayudaba a simplificar el desarrollo de esa parte del proyecto. Dicho código multiplica los 3 bytes de cada pixel (Rojo, Verde y Azul) por un multiplicador de 0,33 y sumarlos para calcular el tono de gris. En ciertos estudios científicos muestran que el ojo humano no calcula el tono de gris con el multiplicador del 0.33 para cada byte, sino que usaban el 0,3 con el Rojo, el 0,59 con el verde y el 0,11 en el azul. Decidí usar esos porcentajes para el desarrollo de la aplicación.

Después lograr la transformación a gris de la imagen, me tocó ponerme a investigar cómo seleccionar una parte de la imagen y aplicar el algoritmo de transformación en escala de grises en la selección. Tras mirar los eventos de ratón y los eventos del picturebox donde encuadramos la imagen que hemos cargado en la vista principal y encontrar soluciones en los foros de ayuda de Microsoft (15), decidí realizar la versión actual de la selección de la imagen. La cual se usa en las dos vistas de la aplicación.

Tras esto, implementé la función para calcular el histograma al seleccionar una sección de la imagen en la segunda vista (vista de imagen en gris), dado que ya se hizo la transformación a escala de grises, me tocó volver a recorrer la imagen para almacenar los datos de cada pixel en un array.

Tras comentar con Andrés la posibilidad de realizar un base de datos en el proyecto y proponer desarrollar una base de datos relacional, empecé a investigar las distintas maneras de unir la base de datos y la aplicación. Tras investigar en el Microsoft Visual Studio 2015 las posibilidades que trae y revisar la documentación de C#, decidí utilizar una funcionalidad de Microsoft Visual Studio 2015. Dicha funcionalidad permite el uso de una base de datos SQL Server sobre un archivo .mdf el cual se incorpora al proyecto como un origen de datos en la aplicación y usa como servidor de SQL, el SQL Server Express incorporado en el Microsoft

Visual Studio 2015.

Tras decidir el tipo de base de datos que necesitaríamos y el lenguaje para desarrollarla, empezamos por diseñar a grandes rasgos un inicio de esquema entidad relación en modo borrador. Tras analizarlo fui realizando cambios para adecuar las entidades y las relaciones a las normas aprendidas en la asignatura de Bases de Datos. Tras terminar el diagrama entidad relación hice la transformación a álgebra relacional y el posterior paso a tablas.

Analicé la nomenclatura de SQL Server para realizar la creación de las tablas en la base de datos.

Tras esto empecé a diseñar y desarrollar el Trigger para la actualización de la tabla means. y empecé a realizar las consultas en la aplicación con la colaboración de Andrés.

Realice el testeo en modo debug de la aplicación para poder encontrar problemas de ejecución en la aplicación y poder depurar los mismos corrigiendo el código.

Bibliografía

[1] compañia iRobot, presentada en el sitio web: http://www.irobot.es/robots-domesticos/aspiracion

[2] artículo de innovaspain sobre robot para controlar los cultivos en invernaderos.4 de noviembre del 2016, presentado en el sitio web: http://www.innovaspain.com/robots-controlar-los-cultivos-invernaderos/

[3] artículo de noticias de la ciencia sobre experimentos en tierras de cultivo con un robot agrícola, 3 de noviembre del 2016, presentado en el sitio web: http://noticiasdelaciencia.com/not/21725/experimentos-en-tierras-de-cultivo-con-un-robot-agricola/

[4] artículo de la voz del muro sobre FarmBot, un robot que siembra, riega y cuida tu huerto, y que puedes fabricar tú mismo, julio del 2016, presentado en el sitio web: http://lavozdelmuro.net/farmbot-un-robot-que-siembra-riega-y-cuida-tu-huerto-y-que-puedes-fabricar-tu-mismo/

[5] artículo en blogspot sobre robot recolector de fruta, 10 de diciembre de 2010,presentado en el sitio web: http://proyectozito.blogspot.com.es/2010/12/robot-recolector-de-fruta.html

[6] documento de la Universidad de Sevilla mediante la Escuela Superior de ingenieros sobre caracerización de texturas, expuesta en la siguiente sitio web: http://bibing.us.es/proyectos/abreproy/11494/fichero/PROYECTO%252FCapitulo+5.pdf

[7] documento de la Universidad Nacional de la Plata del departamento de Ambiente y Recursos Naturales sobre la matriz de co-ocurrencia en la clasificación multiespectral: tutorial para la enseñanza de medidas texturales en cursos de grado universitario,13 de agosto del 2004, expuesto en el siguiente sitio web:

npe.br/unidades/cep/atividadescep/jornada/programa/t-9_trab_27.pdf

[8] documento de Miguel Cárdenas-Montes sobre las medidas de distancia expuesto en el siguiente sitio web: http://wwwae.ciemat.es/~cardenas/docs/lessons/MedidasdeDistancia.pdf

[9] documento de la Universidad del País Vasco-Euskal Herriko Unibertsitatea del departamento de Ciencias de la Computación e Inteligencia Artificial sobre clasificadores K-NN, expuesto en el siguiente sitio web: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t9knn.pdf

[10] documento sobre análisis cluster expuesto en el siguiente sitio web: http://www.ugr.es/~bioestad/_private/cpfund7.pdf

[11] explicación sobre distancia euclidea expuesto en el siguiente sitio web: http://www.uv.es/ceaces/multivari/cluster/d_euclidea.htm

[12] explicación sobre distancia Mahalanobis presentado en el siguiente sitio web: https://www.uv.es/ceaces/multivari/cluster/d_mahalanobis.htm

[13] capacidades del grado de ingeniería informática, expuestas en la página web: http://informatica.ucm.es/estudios/grado-ingenieriainformatica-estudios-competencias

[14] explicación sobre como convertir una imagen a escala de grises realizado por Microsoft Community Publishing Service, presentado en el siguiente sitio web: https://msdn.microsoft.com/es-es/communitydocs/net-dev/csharp/convertir-una-imagen-a-escala-de-grises

[15] explicación sobre seleccionar un trozo de una imagen en un PictureBox y poderlos guardar y dejar marcados de Microsoft Forum, presentado en el siguiente sitio web: https://social.msdn.microsoft.com/Forums/es-ES/212ab0d2-b8de-4b63-add8-e7b519dcef43/seleccionar-trozos-de-una-imagen-en-un-picturebox-y-poderlos-guardar-y-dejar-marcados?forum=vcses

[16] documento de procesamiento de imágenes, presentado en el siguiente sitio web: http://catarina.udlap.mx/u_dl_a/tales/documentos/msp/florencia_y_an/capitulo3.pdf

[17] explicación de la clase SQLCommand de Microsoft, octubre de 2016, presentado en el siguiente sitio web:
https://msdn.microsoft.com/es-es/library/system.data.sqlclient.sqlcommand(v=vs.110).aspx

[18] herramienta TortoiseSVN presentado en el siguiente sitio web: https://tortoisesvn.net/