



Master en Ingeniería de Sistemas y Control

**Identificación de estructuras vegetales en campos de cultivo de maíz**

Realizado por: Manuel David Franco Barrios


Dirigido por: D. Gonzalo Pajares ~~Martin-Sanz~~

Curso académico 2011/2012

Convocatoria de Septiembre

Master en Ingeniería de Sistemas y Control  
Proyecto Fin de Master  
Septiembre 2012

**Identificación de estructuras vegetales en campos de cultivo de maíz**

Realizado por: Manuel David Franco Barrios  
Dirigido por  Gonzalo Pajares Martin-Sanz

*(espacio reservado para calificaciones)*

## **Autorización**

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

## Resumen

A nivel general, se trata de un proyecto en el marco de la **Visión por Computador**, enfocado en particular en el ámbito del **reconocimiento de texturas**. El tema concreto estudiado es el de la **segmentación automática de texturas en imágenes agrícolas**.

Existe actualmente una necesidad general de automatizar las tareas agrícolas, y en los procedimientos empleados para ello se utilizan robots y máquinas cuyos sensores ópticos tienen un papel muy importante. Las imágenes que ofrecen dichos sensores deben ser procesadas adecuadamente, y en concreto una de las necesidades más importantes, que es la problemática aquí tratada, es la de identificar las plantas dentro de dichas imágenes, separándolas de otros elementos no relevantes, como pueden ser cielo y tierra, entre otros.

El proyecto trata el problema de la identificación de estructuras vegetales en imágenes agrícolas, en concreto de campos de cultivo de maíz, para su posterior tratamiento en el contexto de los procesos de automatización necesarios en la agricultura de precisión (por ejemplo aplicación de químicos o quema de mala hierba).

## Palabras clave

*Visión por computador, Reconocimiento de texturas, Segmentación de texturas, Imágenes agrícolas, Campos de cultivo, Agricultura de precisión, Automatización, ~~Identificación de estructuras vegetales~~, Espacios de color, CIELab, Filtros de Gabor, Clustering, K-Means*

*A Marta, a mis amigos, y a mi familia, por el tiempo robado.*

## Índice de contenido

1. Contexto.....	10
2. Descripción del problema y objetivos .....	11
3. Trabajos previos existentes.....	12
4. Alternativa planteada.....	13
4.1. Espacios de color.....	13
4.1.1. Percepción humana del color.....	14
4.1.2. Espacios de color perceptualmente uniformes.....	15
4.2. Espacios de características.....	17
4.2.1. Filtros de Gabor.....	17
4.2.2. Implementación como matrices de convolución.....	20
4.3. Agrupamiento mediante K-Means.....	21
5. Experimentaciones realizadas y resultados observados.....	24
6. Conclusiones.....	46
7. Trabajo futuro.....	48
8. Listado de acrónimos.....	49
9. Referencias .....	50
10. Anexo A: Código fuente .....	52



## Índice de ilustraciones

Ilustración 1: Transformación de espacio RGB a CIE XYZ.....	15
Ilustración 2: Transformación de espacio CIE XYZ a RGB.....	15
Ilustración 3: Luminosidad en Lab y Luv a partir de XYZ.....	16
Ilustración 4: Componentes de una función de Gabor en 1-D.....	18
Ilustración 5: Componentes de una función de Gabor en 2-D.....	18
Ilustración 6: Ejemplos de filtros de Gabor.....	19
Ilustración 7: Filtro de energía de Gabor en 1-D.....	20
Ilustración 8: Convergencia de los centros en algoritmo K-Means.....	22
Ilustración 9: Idea básica inicial del método.....	24
Ilustración 10: Descripción gráfica del método propuesto.....	25
Ilustración 11: Imagen original para ejemplo de aplicación.....	26
Ilustración 12: Canal L en CIELab.....	27
Ilustración 13: Canal b en CIELab.....	27
Ilustración 14: Canal a en CIELab.....	27
Ilustración 15: Filtro Gabor orientación 0 y frecuencia 1.....	28
Ilustración 16: Filtro Gabor orientación $\pi/4$ y frecuencia 1.....	28
Ilustración 17: Filtro Gabor orientación $\pi/2$ y frecuencia 1.....	28
Ilustración 18: Filtro Gabor orientación $3\pi/4$ y frecuencia 1.....	28
Ilustración 19: Filtro Gabor orientación 0 y frecuencia 1.5.....	28
Ilustración 20: Filtro Gabor orientación $\pi/4$ y frecuencia 1.5.....	28
Ilustración 21: Filtro Gabor orientación $\pi/2$ y frecuencia 1.5.....	28
Ilustración 22: Filtro Gabor orientación $3\pi/4$ y frecuencia 1.5.....	28
Ilustración 23: Salida de un filtro de Gabor de escala 17 píxeles, frecuencia 1.5 y orientación 0 sobre canal b.....	29
Ilustración 24: Salida de un filtro de Gabor de escala 33 píxeles, frecuencia 1 y orientación $\pi/4$ sobre canal a.....	30
Ilustración 25: Salida de un filtro de Gabor de escala 17 píxeles, frecuencia 1.5 y orientación $3\pi/4$ sobre canal a.....	30
Ilustración 26: Salida de un filtro de Gabor de escala 9 píxeles, frecuencia 1 y orientación $\pi/4$ sobre canal b.....	31
Ilustración 27: Agrupamiento en 3 clases.....	32
Ilustración 28: Superposición del agrupamiento en 3 clases a la imagen original.....	33
Ilustración 29: Agrupamiento en 4 clases.....	34
Ilustración 30: Superposición del agrupamiento en 4 clases a la imagen original.....	34
Ilustración 31: Agrupamiento en 6 clases.....	35
Ilustración 32: Superposición del agrupamiento en 6 clases a la imagen original.....	36
Ilustración 33: Agrupamiento en 8 clases.....	37
Ilustración 34: Superposición del agrupamiento en 8 clases a la imagen original.....	37
Ilustración 35: Píxeles clasificados como terreno sin vegetación.....	38
Ilustración 36: Píxeles clasificados como cultivo.....	39
Ilustración 37: Píxeles clasificados como pastos.....	40
Ilustración 38: Píxeles clasificados como hierbas.....	41
Ilustración 39: Píxeles clasificados como arboleda.....	42

Ilustración 40: Píxeles sin interés, tipo I.....	43
Ilustración 41: Píxeles sin interés, tipo II.....	44
Ilustración 42: Píxeles sin interés, tipo III.....	44

## 1. Contexto



~~La presente memoria responde a la descripción del trabajo realizado como Proyecto Fin de Master del Máster Universitario en Ingeniería de Sistemas y de Control de la UNED, en el curso académico 2011/2012.~~

~~Dentro de dicho Master, he cursado asignaturas relativas a robótica, inteligencia artificial, modelado y simulación de sistemas dinámicos, automatización industrial, y visión por computador. Con todas ellas he aprendido y disfrutado, pero sin duda ha sido el último dominio citado, el de la visión, el que me ha despertado más interés. Es por ello que me decidí por este ámbito a la hora de cursar las prácticas del master, y también lo elegí sin dudar como dominio de mi Proyecto Fin de Master, descrito en el presente documento.~~

~~Así, me puse en contacto con el profesor D. Gonzalo Pajares Martín-Sanz, con quien ya había cursado "Visión por Computador", y le comuniqué mi deseo de realizar mi proyecto sobre dicha temática, a lo cual accedió amablemente y de inmediato propusimos una serie de posibles proyectos concretos, de entre los cuales me decidí por el presente, titulado "Identificación de estructuras vegetales en campos de cultivo de maíz basado en imágenes", y que enfoca el área de reconocimiento de texturas en imágenes, con el objetivo básico de separar los distintos elementos o regiones presentes en imágenes agrícolas.~~

~~Mi experiencia previa en el ámbito de la visión artificial se ceñía a lo cursado en el Master, si bien sí que había trabajado antes con tratamiento de señales, y en particular con procesamiento de imágenes y vídeo. En concreto, no tenía ninguna experiencia en el reconocimiento de texturas. Es por ello que la mayoría del trabajo realizado ha sido de documentación, estudio y consulta; así como experimentación de los métodos que en los artículos científicos sobre la temática a los que he tenido acceso se mencionaban. Por supuesto dentro de esta experimentación se ha realizado un trabajo de implementación de algunos de dichos métodos. Dicho trabajo, tanto los diferentes métodos estudiados como lo finalmente implementado y los experimentos realizados con dicha implementación, y las conclusiones a las que he llegado, así como posibles alternativas que me hubiera gustado poder llegar a probar, quedan resumidos en la presente memoria.~~

## 2. Descripción del problema y objetivos

El propio título del proyecto, "*Identificación de estructuras vegetales en campos de cultivo de maíz basado en imágenes*", es muy descriptivo y resume la idea general del mismo. A nivel general, se trata de un proyecto en el marco de la **Visión por Computador**, enfocado en particular en el ámbito del **reconocimiento de texturas**. El tema concreto estudiado es el de la **segmentación automática de texturas en imágenes agrícolas**, descrito en [1].

Tal y como se explica en dicho artículo, ~~pero repito~~ aquí para mayor comodidad del lector, existe actualmente una necesidad general de automatizar las tareas agrícolas, y en los procedimientos empleados para ello los sensores ópticos tienen un papel muy importante. Las imágenes que ofrecen dichos sensores deben ser procesadas adecuadamente, y en concreto una de las necesidades más importantes, que es la problemática aquí tratada, es la de identificar las plantas dentro de dichas imágenes, separándolas de otros elementos no relevantes, como pueden ser cielo y tierra, entre otros.

En el caso concreto de nuestro problema, que es el planteado en dicho artículo, se trata de imágenes tomadas sobre campos de maíz, y se necesita incluso separar los distintos tipos de plantas, como serían el maíz y las malas hierbas, con el objetivo de llevar a cabo algún tipo de tratamiento físico sobre las segundas, como la aplicación de herbicidas o la quema de las mismas mediante un brazo robótico. Este último es de hecho un caso real de aplicación dentro del proyecto de investigación [2], ~~según me comentó el profesor D. Gonzalo Pajares.~~

Expuesta la problemática, el objetivo establecido para el presente proyecto era el proponer una solución alternativa a la planteada en el propio artículo [1]. Como material, además del propio artículo en sí, ~~se me proveyó con~~ un conjunto de imágenes de muestra sobre las que experimentar, tomadas como ya se ha dicho sobre campos de maíz, ~~y en todas las cuales~~ aparecen vegetales, tierra, cielo y algunas otras estructuras como edificios varios.

En resumen, el objetivo del proyecto era proponer un método para, dentro de las imágenes provistas, ser capaz de:

1. **Identificar la vegetación**, aislándola del resto de elementos no relevantes.
2. Dentro de las estructuras vegetales identificadas, **discriminar la planta de las malas hierbas**.

### 3. Trabajos previos existentes

En el artículo [1] se exponen como ya se ha dicho tanto el problema como una solución al mismo que consiste, a grandes rasgos, en calcular una serie de índices para cada pixel a partir de su color, componiendo una nueva imagen por cada índice, y luego fusionando la información de las mismas aplicando varios métodos de "thresholding". Dichos índices: ExG, ExR, CIVE, ExGR; ya se mencionaban en otros artículos referenciados por [1] y anteriores al mismo, pero se propone una forma de combinarlos para obtener mejor fiabilidad a la hora de separar plantas, tierra y cielo.

En [1] no se contempla el uso de análisis de frecuencia alegando que las características espectrales de los distintos tipos de vegetación a discriminar son muy similares, y que esto, unido a que las imágenes son tomadas al aire libre y expuestas por tanto a condiciones muy adversas como la exposición a diferentes condiciones de iluminación, sombras, o condiciones climáticas variables, factores todos que pueden afectar la composición espectral, dificulta mucho el empleo de métodos basados en esta técnica. En [6] y [9] se habla de métodos basados en el análisis de píxeles individuales y métodos basados en regiones, descartando los segundos por asumir que en las imágenes aéreas sobre las que se centran ambos artículos no existen patrones identificables como texturas.

No obstante sí que existen en la literatura artículos sobre métodos basados en el análisis espectral para la clasificación y segmentación de texturas, incluso en imágenes naturales ([11],[12],[18]). De hecho es ampliamente utilizado el análisis mediante filtros de Gabor (ver [11], [12], [13], [14], [15], [16], [17], [18]).

La referencia más importante a este respecto es [12], que trata concretamente del análisis de texturas en imágenes multiespectrales, y en particular de la clasificación de texturas no homogéneas en imágenes naturales, tratándose por tanto de un problema de la misma naturaleza que el nuestro. El método que proponen hace hincapié en la correspondencia con la percepción visual humana a la hora de realizar la clasificación, aunque curiosamente no utilizan un espacio de color perceptualmente uniforme, sino que se basan en la aplicación de filtros de Gabor multi-escala sobre las componentes de la imagen en el espacio de color HSI.

También para el análisis de imágenes aéreas en [8] se asume la variabilidad o ambigüedad de las características extraíbles para la identificación de texturas, y proponen la aplicación de la teoría de conjuntos difusos para afrontar la misma.

Por último, en [15] se proponen una serie de medidas de similitud o "distancia" entre texturas (caracterizadas mediante filtros de Gabor), que podrían ser de utilidad comparar a la hora de clasificar las imágenes.

## 4. Alternativa planteada

De entrada, y una vez ~~tenidas en cuenta~~ todos los métodos mencionados en la sección anterior, ~~opté~~ por explorar un camino lo más diferente posible de la solución expuesta en el artículo de referencia [1], de forma que la comparación de los resultados fuera más interesante.

Así, el primer punto de distanciamiento fue basar la clasificación de cada ~~pixel~~ de la imagen en un conjunto de características que describan el entorno del mismo, en contraste con el método empleado en [1] que utiliza exclusivamente la información de color de cada ~~pixel~~ individualmente para calcular los ~~índices~~ que luego utiliza para segmentar, si bien es verdad que este proceso de segmentación ya incluye información global de la imagen, pero en ningún momento se examina el entorno local de cada ~~pixel~~. De hecho, tanto en [1] como en [6] y [9] se desaconseja esta aproximación por la gran variabilidad existente en las texturas presentes en imágenes naturales y aéreas según el caso.

No obstante, un reconocimiento de texturas propiamente dicho ~~debe hacerse estudiando~~ regiones y no píxeles aislados, ya que de acuerdo al concepto de textura ~~encontrado en literatura como~~ [5] y ~~otros~~, no existe textura en un único punto, sino que una textura se caracteriza por la distribución de los valores de un conjunto de píxeles vecinos. Teniendo ~~además en cuenta~~ que se quieren distinguir diferentes tipos de estructuras vegetales, ~~pienso que debe ser casi imposible~~ diferenciar las mismas basándonos exclusivamente en la tonalidad de color, pues siempre van a existir vegetaciones diferentes con colores casi idénticos al estar jugando con una gama muy estrecha del espectro de color.

Por lo expuesto, ~~entiendo que no queda más remedio si quieren conseguirse resultados precisos que~~ intentar controlar la problemática variabilidad presente en las texturas vegetales, sin dejar de estudiarlas como tales ~~(es decir, intentando identificar texturas en regiones locales).~~

Para ello, se ~~han~~ empleado una combinación de técnicas ya descritas en las referencias mencionadas en la sección anterior, siendo las más importantes una selección apropiada del espacio de color sobre el que trabajar, descripción de texturas mediante filtros de Gabor, y agrupamiento de las mismas en ~~diferentes~~ clases. En las siguientes subsecciones se ~~pasa a exponer~~ los conceptos fundamentales de cada uno de estos elementos por separado, y en la sección posterior se describe su uso conjunto en el método propuesto.

### 4.1. Espacios de color

Según [12] la elección del espacio de color es esencial en el análisis de texturas en imágenes multispectrales.

En esta sección se presentan los conceptos básicos para entender qué es un espacio de color, cuáles son los más ampliamente usados, cuáles son sus características, y cuál ha sido la elección para el presente proyecto.

#### 4.1.1. Percepción humana del color

La percepción humana del color se basa en la incidencia de la luz visible (ondas electromagnéticas cuya longitud de onda va de 400nm a 700 nm) en la retina, en la cual existen tres tipos de células fotorreceptoras, cada uno con una curva de respuesta espectral diferente, por lo que en teoría todos los colores pueden especificarse mediante tres números que corresponderían a las salidas de dichos fotorreceptores.

En 1931 la "Commission Internationale de l'Eclairage" (CIE) estandarizó las curvas de respuesta de los fotorreceptores de un hipotético observador estándar, y definió el sistema CIE XYZ mediante el que cualquier color visible puede ser representado mediante valores positivos de X, Y, y Z. Puede verse por tanto dicho sistema XYZ como un espacio en tres dimensiones donde cada vector define un color, teniendo aquí por tanto la definición de espacio de color.

Dicho espacio XYZ comprende como se ha dicho todos los colores visibles, sin embargo no es el único espacio de color estándar existente, sino que son muy utilizados una serie de subespacios del mismo diseñados con propósitos específicos. Los más comunes son:

- RGB: Es un sistema de colores aditivo basado en la suma de los tres colores primarios rojo, verde y azul, utilizado comúnmente en pantallas CRT.
- CMYK: Sistema de color substractivo usado en imprenta, donde la ausencia de color (origen del espacio) es el blanco.
- HSL: Sistema de colores diseñado para ser intuitivo a la hora de permitir a un usuario especificar un color fácilmente. La componente de luminosidad está separada en un canal propio, lo que tiene ventajas para ciertas aplicaciones de procesamiento de imágenes.

Todos ellos son subespacios diferentes del espacio XYZ, lo que quiere decir que cada uno deja fuera un conjunto de colores que no puede representar.

Además, al estar compuesta la base de cada espacio por un conjunto diferente de vectores en XYZ, la distancia existente entre dos colores dados es diferente según se mida en uno u otro espacio. Esto es muy relevante de cara a muchas aplicaciones de tratamiento de imágenes, entre ellas la nuestra.

#### 4.1.2. Espacios de color perceptualmente uniformes

En ciertas aplicaciones es muy importante poder medir la distancia entre colores de forma que se corresponda lo más fielmente posible con la similitud entre esos colores percibida por un ser humano, para lo cual han sido diseñados una serie de espacios de colores llamados "perceptualmente uniformes" o simplemente "uniformes".

Un espacio de color es perceptualmente uniforme si un pequeño desplazamiento de un color C en dicho espacio (es decir, se desplaza una distancia  $d$  medida en dicho espacio) provoca un mismo grado de variación en la percepción independientemente de en qué lugar del espacio esté dicho color. Por ejemplo, en un espacio no uniforme como RGB, la diferencia perceptual entre un par de colores A y B (p.e. Violeta y rojo) puede ser equivalente a una mayor diferencia perceptual entre B y C (rojo y negro), existiendo la misma distancia entre los respectivos pares ( $AB = BC$ ).

En 1976, la CIE estandarizó dos espacios de este tipo, llamados *Luv* (o CIELuv) y *Lab* (o CIELab), compuestos ambos por tres canales en los que el canal "L" contiene la información relativa a la luminosidad, y los otros dos ("u" y "v", o "a" y "b", respectivamente) con tienen la relativa a la cromaticidad. Los dos espacios son también subespacios de CIE XYZ, y comparten un blanco de referencia  $X_n Y_n Z_n$ .

Dado que la gran mayoría de librerías existentes para el tratamiento de imágenes carga las mismas por defecto en espacio RGB, para transformarlas a espacio CIELab se pasa primero por el espacio de referencia XYZ, mediante las siguientes transformaciones.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Ilustración 1: Transformación de espacio RGB a CIE XYZ

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.240479 & -1.537150 & -0.498535 \\ -0.969256 & 1.875992 & 0.041556 \\ 0.055648 & -0.204043 & 1.057311 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Ilustración 2: Transformación de espacio CIE XYZ a RGB



Para la transformación entre XYZ y CIELab o CIELuv, la luminancia se define igual:

$$L^* = \begin{cases} 116 (Y/Y_n)^{1/3} - 16 & \text{if } Y/Y_n > 0.008856 \\ 903.3(Y/Y_n) & \text{otherwise} \end{cases}$$

Ilustración 3: Luminosidad en Lab y Luv a partir de XYZ

Los canales de cromaticidad sí que se definen de forma diferente en CIELuv y CIELab. En el primer caso:

$$\begin{aligned} u^* &= 13L^*(u' - u'_n) , \\ v^* &= 13L^*(v' - v'_n) , \\ u' &= \frac{4X}{X + 15Y + 3Z} , \\ v' &= \frac{9Y}{X + 15Y + 3Z} , \end{aligned}$$

Donde  $u'_n$  y  $v'_n$  se definen igual que  $u'$  y  $v'$  pero usando  $X_n$ ,  $Y_n$  y  $Z_n$ .

En el caso de CIELab:

$$\begin{aligned} a^* &= 500 (f(X/X_n) - f(Y/Y_n)) , \\ b^* &= 200 (f(Y/Y_n) - f(Z/Z_n)) , \end{aligned}$$

donde

$$f(t) = \begin{cases} t^{1/3} & \text{if } Y/Y_n > 0.008856 , \\ 7.787t + 16/116 & \text{otherwise} . \end{cases}$$

## 4.2. Espacios de características

Como ya se ha expuesto, el color es una propiedad a nivel de pixel, pero la textura implica en sí un espacio o *region*, y no existe textura en un sólo pixel [5] [18].

Para describir la textura en una región es normal utilizar una proyección de la función de intensidad de la imagen sobre un conjunto de funciones. Esto se suele denominar descomposición espectral, dado que cada una de dichas funciones suele medir la intensidad en un area diferente del dominio de la frecuencia espacial en dos dimensiones (aunque puede haber otras alternativas).

La escala, la frecuencia espacial, y la orientación son características muy importantes de las texturas presentes en una imagen, y son las que queremos extraer. En nuestro caso, una textura se describe como un vector de valores, donde cada uno corresponde a la energía relativa en una subbanda de frecuencia, de escala y de orientación específica.

El método de descomposición espectral que empleamos está basado en filtros de Gabor (~~ver~~ [11], [12], [13], [14], [15], [16], [17], [18] y [19]). Existe literatura que evidencia como puede modelarse mediante filtros de Gabor la respuesta de las neuronas de la corteza visual primaria del cerebro de algunos animales. Incluso se dan en [19] algunas guías sobre cómo modelarlas, aunque nosotros utilizaremos un banco mucho más simple que veremos que será suficiente como prueba de concepto.

### 4.2.1. Filtros de Gabor

Una función de Gabor es una función senoide modulada por una gaussiana. En las siguientes figuras se muestra de forma intuitiva el concepto en el caso de una y de dos dimensiones.

En el caso de 1-D una función como esta maximizará su respuesta en una frecuencia específica (dependiendo de la componente sinusoidal), y sólo en una parte localizada de la señal (dependiendo de la componente gaussiana). En el caso de 2-D, la salida dependerá también de la orientación de la senoide.

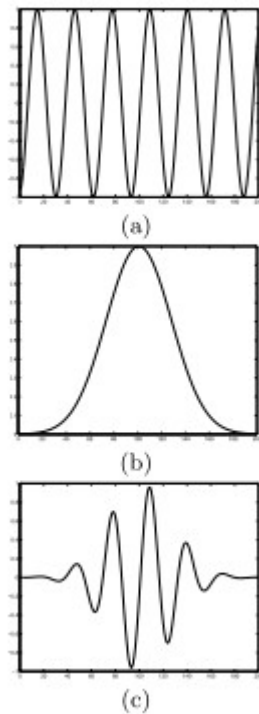


Ilustración 4:  
Componentes de una  
función de Gabor en 1-D

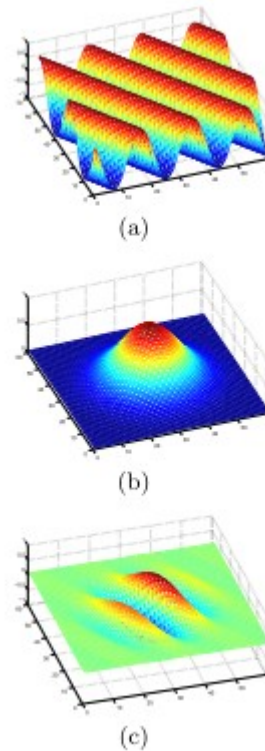


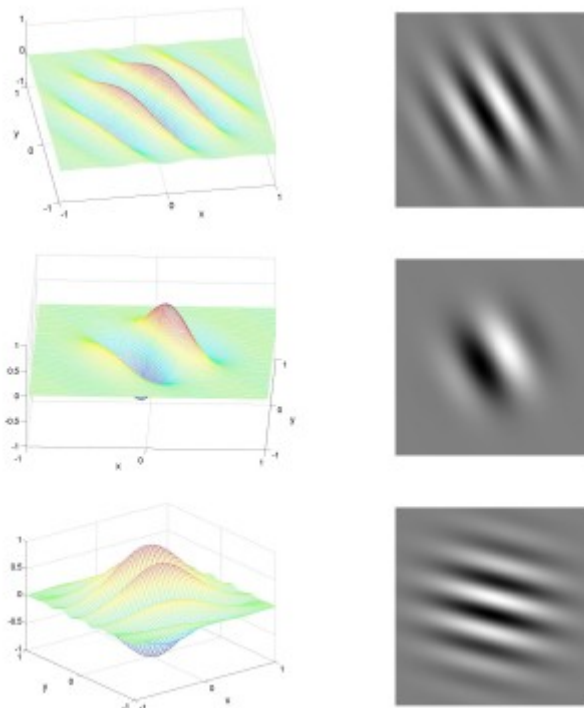
Ilustración 5: Componentes de  
una función de Gabor en 2-D

Sea  $g(x, y, \theta, \phi)$  una función de Gabor con una longitud de onda  $\theta$  y orientación  $\phi$ , se define mediante la siguiente fórmula:

$$g(x, y, \theta, \phi) = \exp\left(-\frac{x^2 + y^2}{\sigma^2}\right) \exp(2\pi i \theta (x \cos \phi + y \sin \phi))$$

Donde a su vez la longitud de onda se define como  $\theta = f / \lambda$  y la desviación estándar como  $\sigma = 0.5 \lambda$ , siendo  $f$  la frecuencia espacial y  $\lambda$  la escala del filtro.

Las figuras siguientes muestran la representación en 3-D de algunas funciones de Gabor, y su aspecto al representarlas en el plano de una imagen.



*Ilustración 6: Ejemplos de filtros de Gabor*

La respuesta de una función de Gabor a una imagen se obtiene como una función de convolución bidimensional. Sea  $I(x,y)$  la función de intensidad de la imagen y  $G(x, y, \theta, \phi)$  la respuesta del filtro de Gabor con longitud de onda  $\theta$  y orientación  $\phi$  a una imagen en el punto  $(x, y)$ ,  $G(\cdot)$  se obtiene como sigue:

$$G(x, y, \theta, \phi) = \int \int I(p, q) g(x - p, y - q, \theta, \phi) dp dq$$

En realidad, tal como se aprecia en la definición de  $g$ , estamos hablando de una función compleja, y las representaciones mostradas en las figuras anteriores corresponden sólo a la parte real, pero el filtro de Gabor realmente está compuesto de una parte real y una imaginaria que pueden verse como dos filtros iguales desfasados  $90^\circ$ . La parte real contiene el filtro

$$g_r(t) = w(t) \sin(2\pi f t + \theta)$$

y la parte imaginaria contiene

$$g_i(t) = w(t) \cos(2\pi f t + \theta)$$

Las respuestas de las componentes real e imaginaria de un filtro de Gabor complejo a una senoide son otra senoide, como se ve en la siguiente figura. Pero si lo que queremos es medir la presencia o energía de una señal de cierta frecuencia en una entrada, lo deseable es que la respuesta sea una señal positiva. Para ello, lo que haremos será tomar como salida del filtro Gabor la magnitud del mismo tomando parte real e imaginaria como coordenadas polares, es decir, la salida viene definida como la raíz cuadrada de la suma de los cuadrados de las salidas real e imaginaria en cada punto. A esto lo llamamos filtro de energía de Gabor y es el tipo que usamos en el presente proyecto.

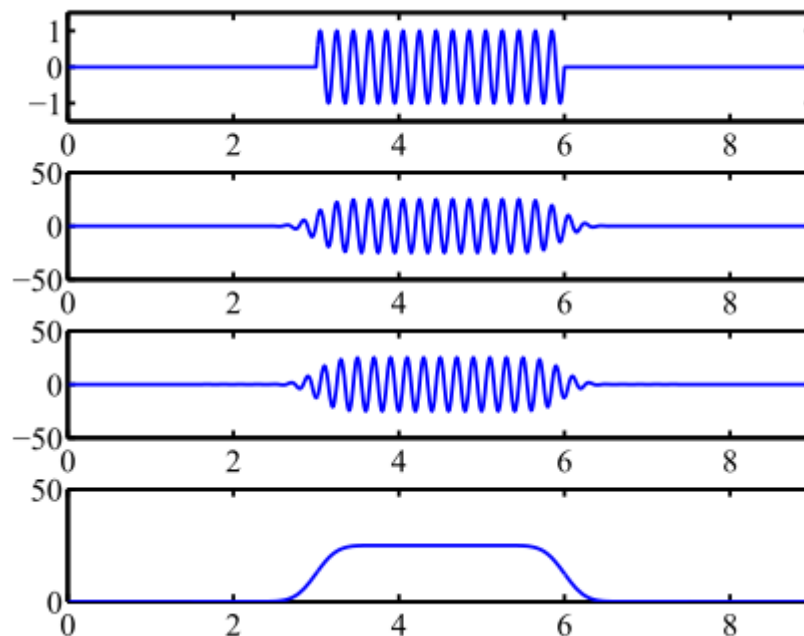


Ilustración 7: Filtro de energía de Gabor en 1-D

#### 4.2.2. Implementación como matrices de convolución

Sobre la aplicación eficiente de estos filtros a una imagen, también hay que contemplar que aunque en la teoría se ha definido anteriormente como la convolución del filtro sobre la imagen completa para calcular la salida en cada punto, esto es altamente costoso computacionalmente hablando.

Teniendo en cuenta que por concepto la salida del filtro va a depender de la región donde se centra la función gaussiana, la aportación del resto de puntos de la imagen que quedan fuera de dicha campana va a ser prácticamente nula, por lo que en este proyecto se ha optado por modelar cada

filtro de Gabor como una matriz cuadrada cuya dimensión depende de la escala del filtro (y por tanto de la desviación estándar de la componente gaussiana).

Esta matriz una vez calculada puede aplicarse como cualquier filtro de convolución lineal, es decir que su coste de computación es similar a aplicar por ejemplo un filtro de suavizado básico (si bien la dimensión de la matriz será seguramente bastante mayor).

De esta forma, en nuestro método tenemos para cada filtro Gabor dos matrices  $N \times N$ , una con la parte real y otra con la parte imaginaria del filtro, que se aplican por separado al canal original por convolución 2-D, y sus resultados respectivos son interpretados como coordenadas polares de las que se extrae la magnitud.

Esto provoca una diferencia respecto a la definición usual de filtros en la literatura, donde se caracteriza a los mismos con dos parámetros: frecuencia espacial y orientación, y se liga la escala del filtro a la propia frecuencia. Nosotros sin embargo desligamos ambos parámetros y por tanto necesitamos definir los tres para formar un filtro de Gabor como una matriz:

1. La escala nos dará el tamaño de dicha matriz de convolución (ligada a la desviación estándar de la componente gaussiana).
2. La frecuencia nos dará el número de ciclos de la senoide dentro de la escala indicada.
3. La orientación, igual que en la literatura.

#### 4.3. Agrupamiento mediante K-Means

Una vez descrita la textura que rodea cada pixel de la imagen mediante un vector de características, queda la tarea de clasificar los mismos en grupos que correspondan a las diferentes texturas buscadas o presentes en la imagen.


Notese aquí que el número de texturas buscadas, o texturas de interés para la aplicación, normalmente va a ser mucho menor que el número de texturas realmente presentes en la imagen, especialmente cuando estamos trabajando con imágenes reales.

Existen numerosos algoritmos de agrupamiento, de varios tipos, pero el empleado en el presente proyecto por su sencillez y eficiencia ha sido el algoritmo K-Means [4] [20] [18]. Se trata de un algoritmo de agrupamiento no supervisado que representa una distribución de datos usando K centros, donde K es un número elegido por el usuario.

El algoritmo K-Means tiende a encontrar grupos naturales en los datos. El usuario elige cuantos

grupos ~~quiere formar~~ y el algoritmo crea K centros que acaba situando, dentro del espacio de características, en el centro de los grupos naturales de datos existentes en el mismo.

El funcionamiento básico del algoritmo es el siguiente:

1. Como entrada se recibe el conjunto de datos y el número K de ~~clusters~~  deseados.
2. Se colocan aleatoriamente K centros en el espacio de datos.
3. Se asocia a cada dato el centro más cercano al mismo.
4. Se calcula, para cada centro, el punto medio de todos los datos asociados al mismo, y se produce un acercamiento del centro a dicho punto medio.
5. Se vuelve al punto 3 hasta que haya convergencia (los centros no se muevan de forma relevante).

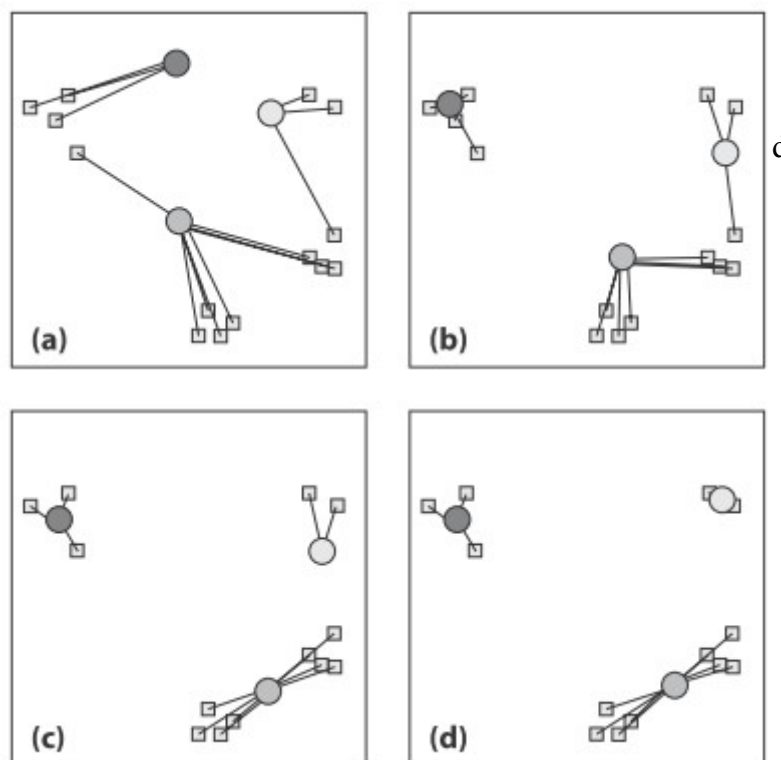


Ilustración 8: Convergencia de los centros en algoritmo K-Means

En la mayoría de los casos el algoritmo converge rápidamente, aunque a veces puede requerir un número mayor de iteraciones.

Por lo general se trata de un algoritmo muy efectivo, pero tiene tres grandes problemas:

1. No se garantiza una solución óptima: puede que se caiga en soluciones localmente óptimas.
2. El número de clusters elegido depende del usuario. Si por ejemplo en la ilustración anterior se hubieran elegido 2 clusters los resultados serían muy diferentes, probablemente poco intuitivos.
3. Supone que la covarianza en el espacio de datos no es relevante o ya ha sido normalizada.

Por suerte para cada uno de estos problemas hay alguna solución, o al menos alguna forma de reducirlo. Para los dos primeros puntos, se puede usar como medida del problema la varianza en los datos de cada cluster clasificado, de forma que un mejor agrupamiento es aquel que la minimiza sin usar demasiados grupos. Las soluciones a los problemas anteriores son respectivamente:

1. Ejecutar el algoritmo varias veces con centros iniciales diferentes, y elegir la solución con la menor varianza.
2. Comenzar ejecutando el algoritmo con  $K=1$ , y volver a ejecutar iterativamente incrementando  $K$ , midiendo la varianza total para cada iteración, de forma que esta irá disminuyendo hasta llegar a un punto de estancamiento, momento en el cual el nuevo cluster añadido no reduce de forma relevante la varianza.
3. Multiplicar los datos por la inversa de la matriz de covarianza, de manera que el espacio de datos se "deforma" para compensar las diferencias en significatividad entre distancias de diferentes características.

En especial el segundo punto es un problema que nos hemos encontrado en los experimentos realizados, y que veremos con más detalle en la siguiente sección.



## 5. Experimentaciones realizadas y resultados observados

Desde el principio el camino a explorar fue la aplicación de filtros de Gabor para clasificar cada pixel en función a su entorno. Es decir, producir una descripción, en forma de vector de características, de cada pixel en función del resultado obtenido de la aplicación de un banco de filtros de Gabor en su entorno, y utilizar este espacio de características como entrada de algún algoritmo de agrupamiento para realizar la clasificación.

~~Además, habiendo en la literatura referencias a la importancia del espacio de color utilizado, realicé algunas pruebas del mismo método sobre diferentes espacios, confirmando la importancia de la elección apropiada del mismo.~~

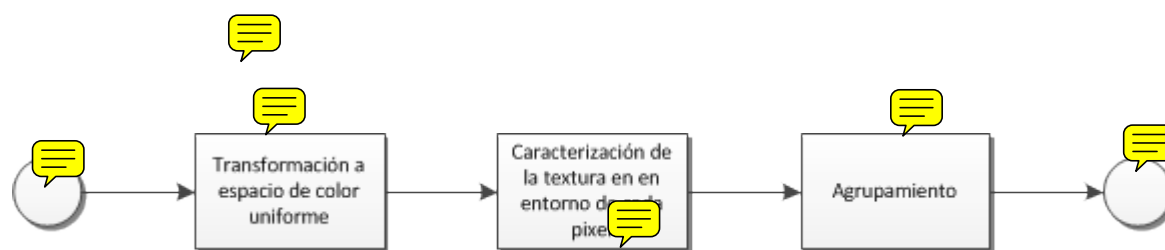


Ilustración 9: Idea básica inicial del método

Los resultados obtenidos con el espacio RGB utilizado inicialmente no eran nada alentadores, incluso empeorando si se pretrataba la imagen normalizando los valores para pasar al espacio cromático rgb, tratando de independizarnos de los efectos de la luminosidad en la imagen. Se probaron una enorme lista de variaciones de parámetros en el banco de filtros aplicado, no encontrando una mejora general con ninguna combinación de escalas y orientaciones, o dicho de otra forma, no se encontraba el espacio de características adecuado.

Se pasó entonces a probar sobre el espacio HSL, mejorando los resultados de forma que las clasificaciones resultantes de aplicar los mismos filtros de Gabor, es decir, para un mismo espacio de características, eran más parecidas a las esperadas.

No obstante, la mejora más notable vino con la utilización del espacio CIELab como base para la aplicación del banco de filtros, en concreto de sus componentes cromáticas "a" y "b", dejando fuera el canal de intensidad o luminosidad "L". Esto dio lugar a una clasificación mucho más cercana a la esperable y deseable, es decir resultante en la tendencia al agrupamiento en una misma clase de pixels pertenecientes al mismo tipo de elementos naturales, de la forma en que lo haría una persona. Además, es esperable que el hecho de descartar el canal de luminosidad implique cierta independencia respecto de las condiciones climáticas y de iluminación en las que se tomen las fotografías, aunque esto no se ha podido comprobar empíricamente por no disponer de las imágenes necesarias.

De todas formas, la mayor parte del tiempo de pruebas y experimentación se ha dedicado a realizar ejecuciones con diferentes parámetros en el diseño del banco de filtros de Gabor. Se han hecho pruebas con bancos de filtros con una sola y con varias escalas, con número diferente de orientaciones, y con varias frecuencias espaciales.

Por último, destacar también que para afinar los resultados se jugó con el suavizado tanto de la imagen original como de las resultantes de la aplicación de los filtros de Gabor, observando que en ambos casos la clasificación mejoraba al aplicar un filtro de suavizado gaussiano en las mismas, haciendo menos frecuente la aparición de pequeñas "islas" de píxeles mal clasificados, y aumentando la tendencia a que píxeles próximos se clasificasen como de una misma textura. Es de suponer que el filtrado de la imagen inicial procura este efecto al limar posibles defectos de la misma debido al ruido (el ruido por la compresión jpg en las imágenes originales era observable a simple vista). En cuanto al filtrado de las resultantes de los filtros, es una técnica recomendada en la literatura consultada, y su aplicación ha servido para amortiguar la variabilidad presente en las texturas naturales, como es nuestro caso.

En definitiva, después de todo el proceso de experimentación, el proceso general quedó como se describe en el diagrama de flujo de la figura siguiente.

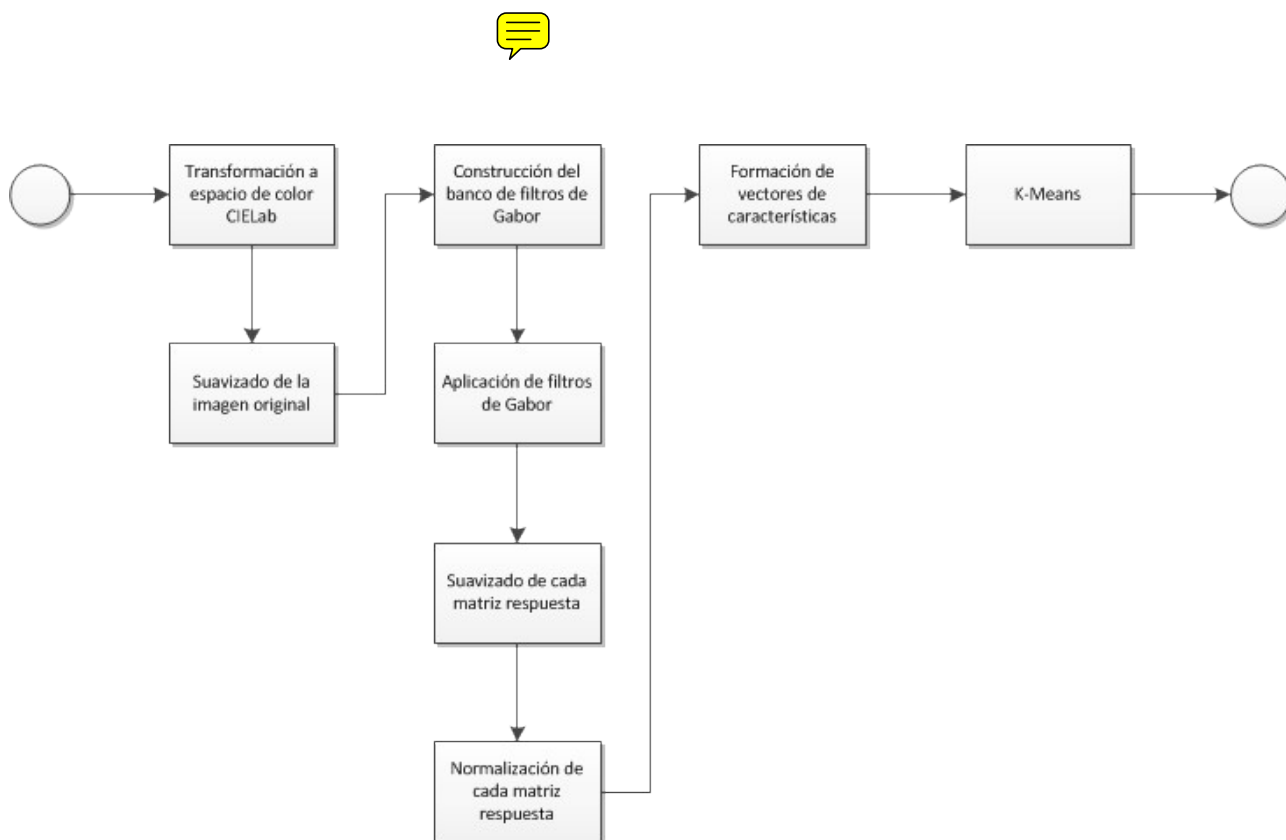


Ilustración 10: Descripción gráfica del método propuesto

Se expone a continuación un ejemplo de aplicación del método a una imagen concreta mostrada en la figura. Los resultados y conclusiones son similares a los obtenidos en el resto de imágenes analizadas en el transcurso del proyecto.



*Ilustración 11: Imagen original para ejemplo de aplicación*

La primera etapa del proceso es convertir la imagen al espacio de color CIELab.



*Ilustración 12: Canal L en CIELab*



*Ilustración 14: Canal a en CIELab*

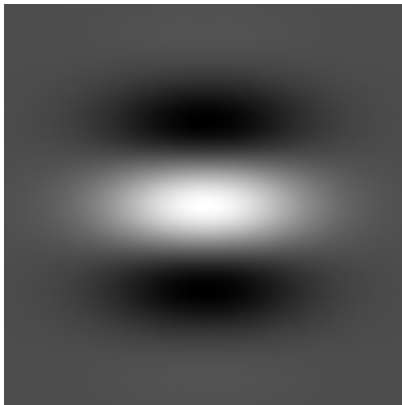


*Ilustración 13: Canal b en CIELab*

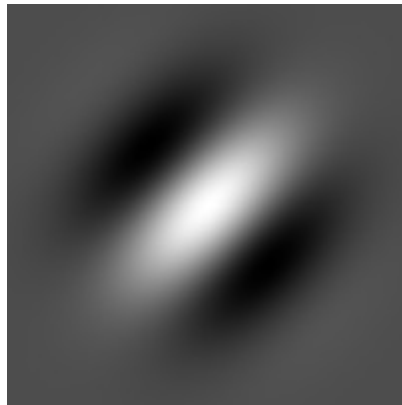
De los canales obtenidos, se descarta el canal "L" pasando a trabajar únicamente con la cromática de la imagen, canales "a" y "b".



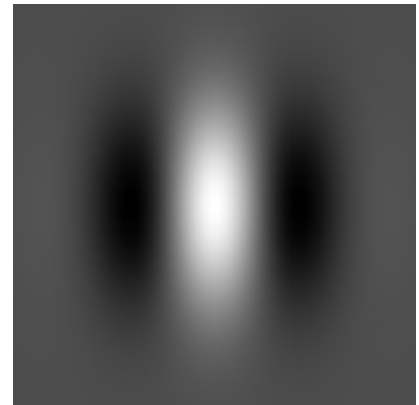
El siguiente paso en el proceso es generar el banco de filtros de Gabor que se aplicará sobre dichos canales. El diseño de este banco de filtros se ha obtenido de forma completamente empírica, jugando con los parámetros del mismo y comparando los resultados obtenidos. De los experimentos llevados a cabo las clasificaciones más correctas se han conseguido con un banco compuesto por 24 filtros en 3 escalas (9, 17 y 33 píxeles), 4 orientaciones (horizontal, vertical y diagonales) y 2 frecuencias espaciales (1 y 1.5).iendo los filtros los representados en las siguientes figuras (las diferentes escalas se obvian).



*Ilustración 15: Filtro Gabor  
orientación 0 y frecuencia 1*



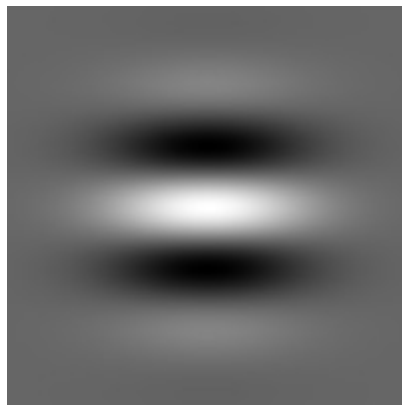
*Ilustración 16: Filtro Gabor  
orientación  $\pi/4$  y frecuencia 1*



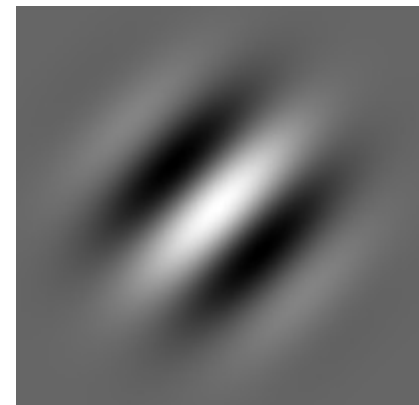
*Ilustración 17: Filtro Gabor  
orientación  $\pi/2$  y frecuencia 1*



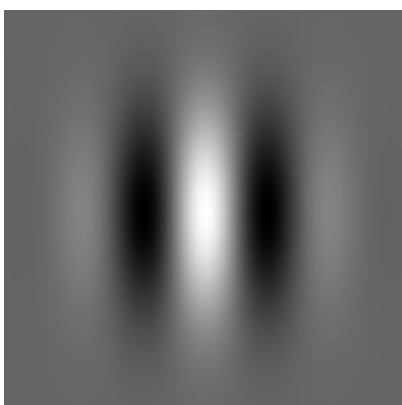
*Ilustración 18: Filtro Gabor  
orientación  $3\pi/4$  y frecuencia 1*



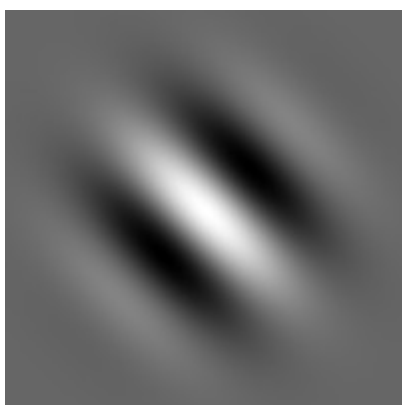
*Ilustración 19: Filtro Gabor  
orientación 0 y frecuencia 1.5*



*Ilustración 20: Filtro Gabor  
orientación  $\pi/4$  y frecuencia 1.5*



*Ilustración 21: Filtro Gabor  
orientación  $\pi/2$  y frecuencia 1.5*

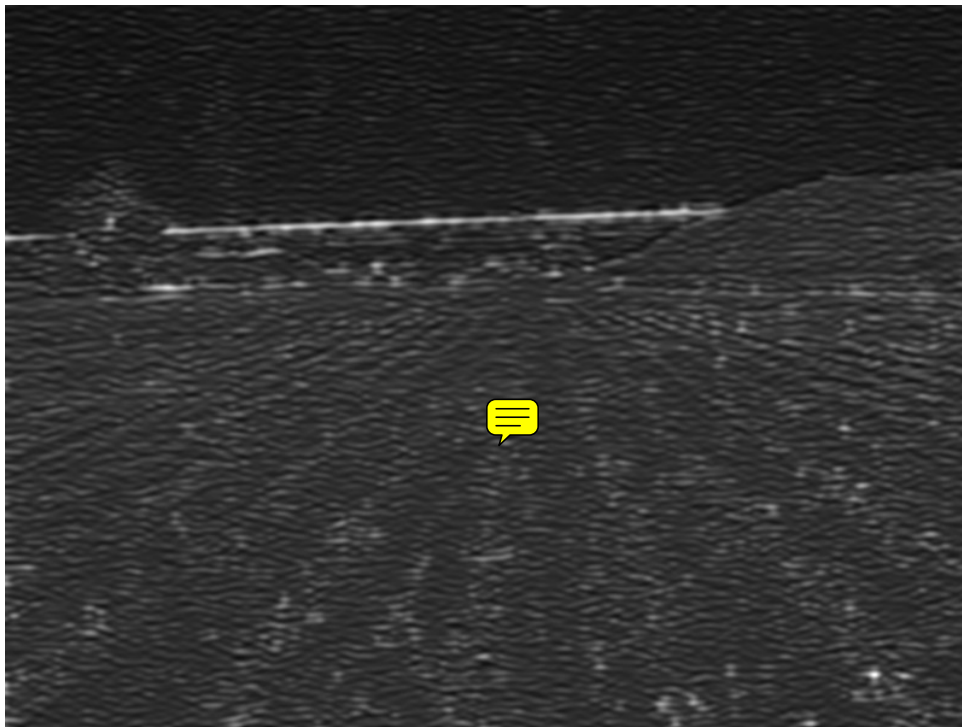


*Ilustración 22: Filtro Gabor  
orientación  $3\pi/4$  y frecuencia 1.5*

Realmente, la representación mostrada en las figuras corresponde a la parte real del filtro de Gabor complejo. La representación de la parte imaginaria sería idéntica pero con los colores invertidos, pues están en fase de  $90^\circ$ .

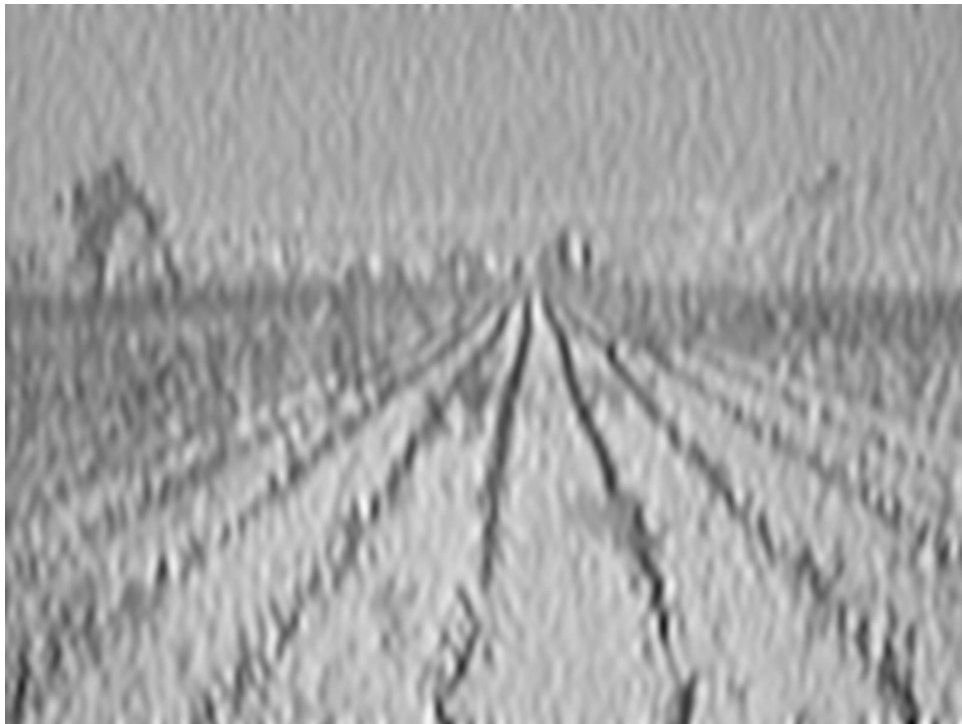
De la aplicación de este banco de 24 filtros sobre cada uno de los 2 canales considerados de la imagen original resultan 48 matrices que contienen las salidas de aquellos.

Dichas matrices son suavizadas mediante un filtro gaussiano, y sus valores normalizados, por el motivo que se explicará posteriormente. Se muestran como ejemplo algunas salidas en las imágenes a continuación:

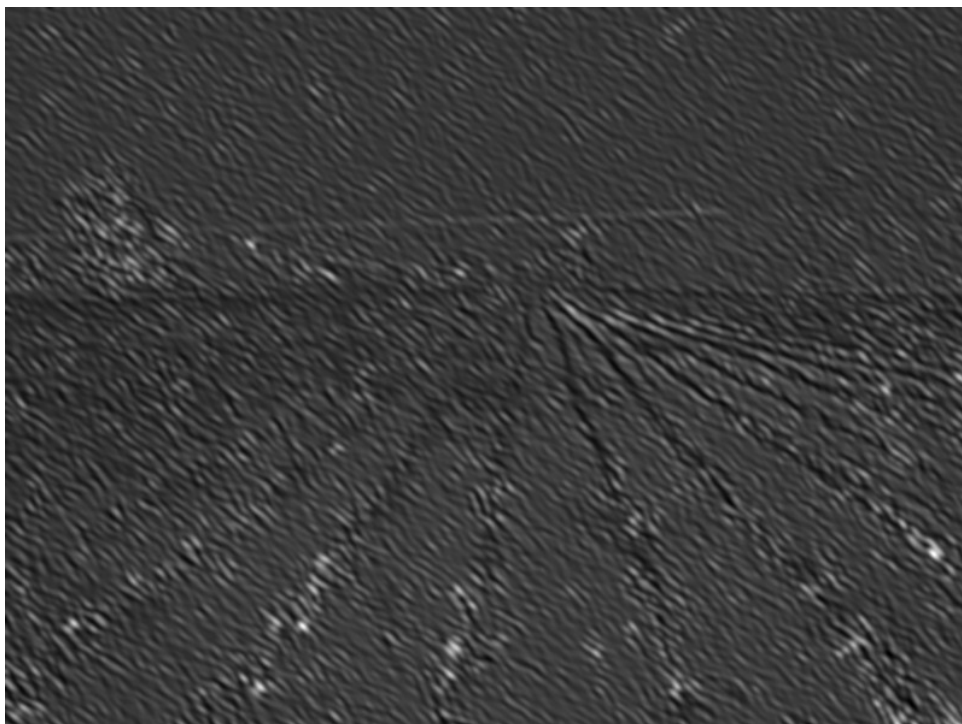


*Ilustración 23: Salida de un filtro de Gabor de escala 17 píxeles, frecuencia 1.5 y orientación 0 sobre canal b*





*Ilustración 24: Salida de un filtro de Gabor de escala 33 pixeles, frecuencia 1 y orientacion PI sobre canal a*



*Ilustración 25: Salida de un filtro de Gabor de escala 17 pixeles, frecuencia 1.5 y orientacion 3PI/4 sobre canal a*



Ilustración 26: Salida de un filtro de Gabor de escala 9 píxeles, frecuencia 1 y orientación  $\pi/4$  sobre canal b

Reorganizando la información de estas matrices, se forman  $N$  vectores de características de dimensión 48, uno por cada píxel de la imagen original. Dichos vectores son la descripción de la textura en el entorno centrado en el píxel correspondiente.

Al haber sido normalizadas las salidas de los filtros, el valor de una característica en un vector no indicará ya la energía en ese punto de la imagen de la salida filtro correspondiente, sino la importancia relativa de ese píxel en relación a los demás en cuanto a esa característica. Esta normalización influye también en el cálculo de las distancias entre los descriptores de textura a la hora del agrupamiento, puesto que si no estuvieran normalizados, y suponiendo que hablamos de distancia euclídea (o cualquier otra que no implique una normalización intrínseca del espacio de parámetros, como podría ser Mahalanobis), las variaciones en distancia en algunas de las características podrían ser mucho mayores que en el resto influyendo por tanto mucho más en la clasificación. En principio, no es lo deseable, y de hecho en las pruebas realizadas se ha comprobado empíricamente que los resultados son mejores aplicando esta normalización.

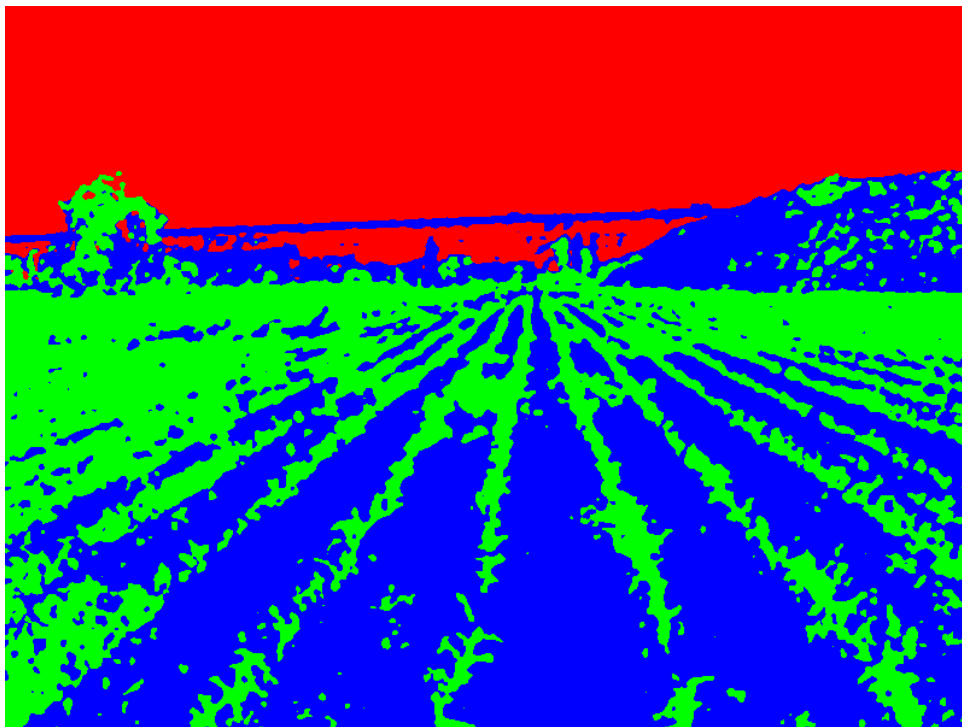
Una vez tenemos descrito cada píxel de la imagen en cuanto a la textura en su entorno local, nos queda realizar el agrupamiento.

Como se explicó en la sección anterior, se utiliza el algoritmo K-Means eligiendo el número de clusters en que se desea dividir la imagen. Veremos aquí el resultado de la clasificación dependiendo del número de clusters elegido.





En primera instancia, probamos a dividir la imagen en tres **clusters**, que intuitivamente **esperábamos** que resultaran ser cielo, tierra y estructuras vegetales. **Como vemos** en las imágenes **a continuación**, la clasificación resultante cumple de forma bastante aproximada con dicha expectativa, aunque no con la precisión deseable puesto que yerra clasificando algunos píxeles pertenecientes a estructuras vegetales en el mismo grupo que los píxeles de tierra. Para una mejor observación de los resultados, se superpone también la clasificación coloreada a la imagen original.



*Ilustración 27: Agrupamiento en 3 clases*



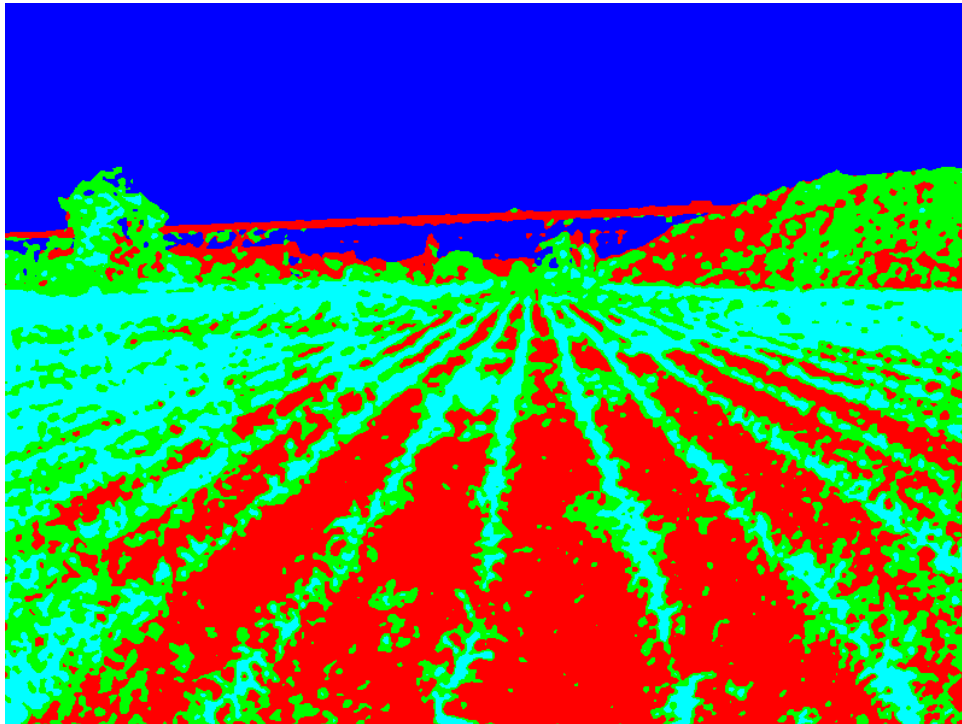
Ilustración 28: Superposición del agrupamiento en 3 clases a la imagen original

No obstante, el objetivo del presente proyecto no es sólo clasificar en estos tres grupos, sino llegar a distinguir ~~dentro de los~~ diferentes tipos de estructuras vegetales presentes, para lo que es necesario por tanto utilizar más clases. Se muestra **a continuación** el resultado que obtuvimos de aumentar el número de clases a 4.

Como se observa, efectivamente la clase adicional sirve para distinguir entre dos tipos de estructuras vegetales: una agrupa los surcos principales correspondientes a los cultivos, con un pequeño error pues incluye algunos píxeles correspondientes a vegetación arborea; y otra que agrupa la vegetación circundante a los surcos cultivados, los arbustos de la colina en el fondo de la imagen a la derecha, y la gran mayoría de la vegetación arborea. Es decir, la clasificación se aproxima bastante a una identificación, dentro de la primera clase, de las estructuras vegetales correspondientes a las plantas cultivadas.

Notese observando la imagen original que **la distinción entre las clases no depende sólo de la tonalidad del verde, sino que obedece bastante al tipo de vegetación en sí** (o al menos a diferentes estados de una misma planta, por ejemplo de crecimiento), aunque con los errores mencionados, que seguramente podrían corregirse ~~realizando un diseño más cuidadoso~~ del banco de filtros. Esto es una **aportación fundamental de la aplicación de filtros de Gabor** frente al análisis de la tonalidad de cada píxel de manera individual, que como por ejemplo en [1] distinguen varios tipos de planta donde en realidad hay píxeles con diferentes tonalidades de verde pertenecientes a

una misma planta, es decir, producen una sobreclasificación.

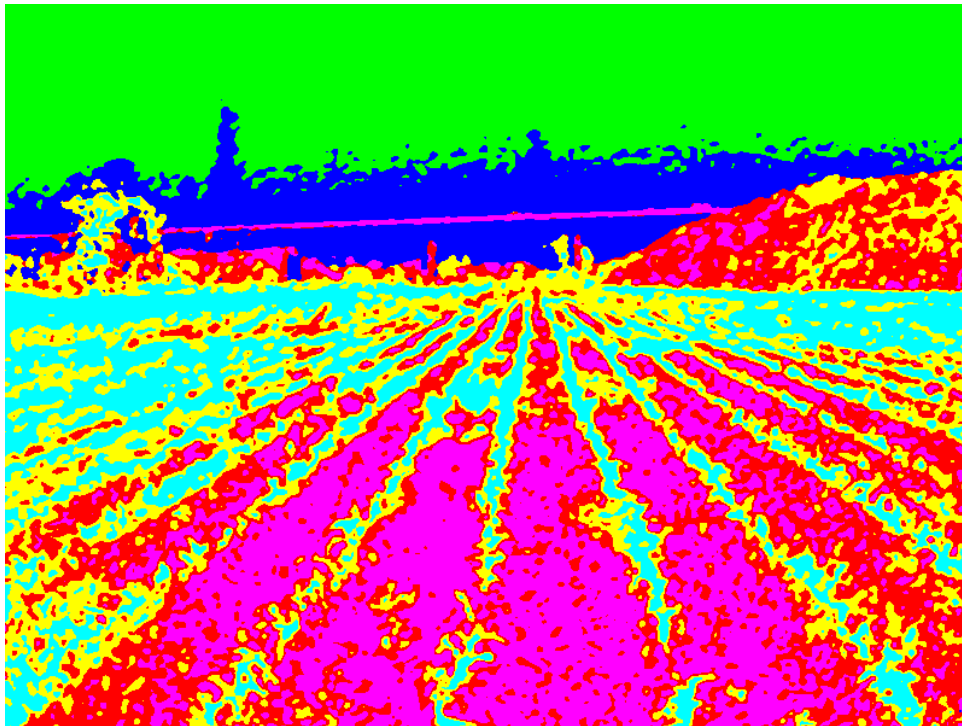


*Ilustración 29: Agrupamiento en 4 clases*



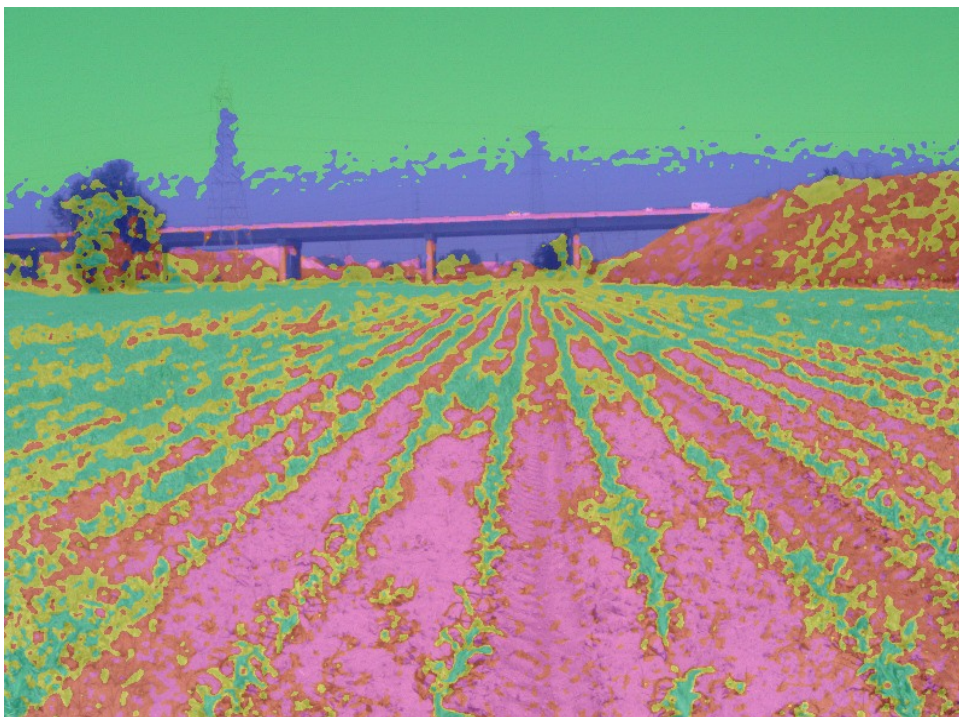
*Ilustración 30: Superposición del agrupamiento en 4 clases a la imagen original*

No obstante, que el objetivo sea identificar cielo, tierra, plantas cultivadas y otras plantas no quiere decir que en la imagen no existan más texturas, por lo que si forzamos a que la clasificación se haga en 4 grupos estamos obligando a clasificar erroneamente todas las texturas que no pertenezcan realmente a ninguno dichos grupos. Por ello, se hicieron pruebas utilizando 6 y 8 grupos de clasificación respectivamente, obteniendo los resultados que se visualizan en las imágenes a continuación.



*Ilustración 31: Agrupamiento en 6 clases*



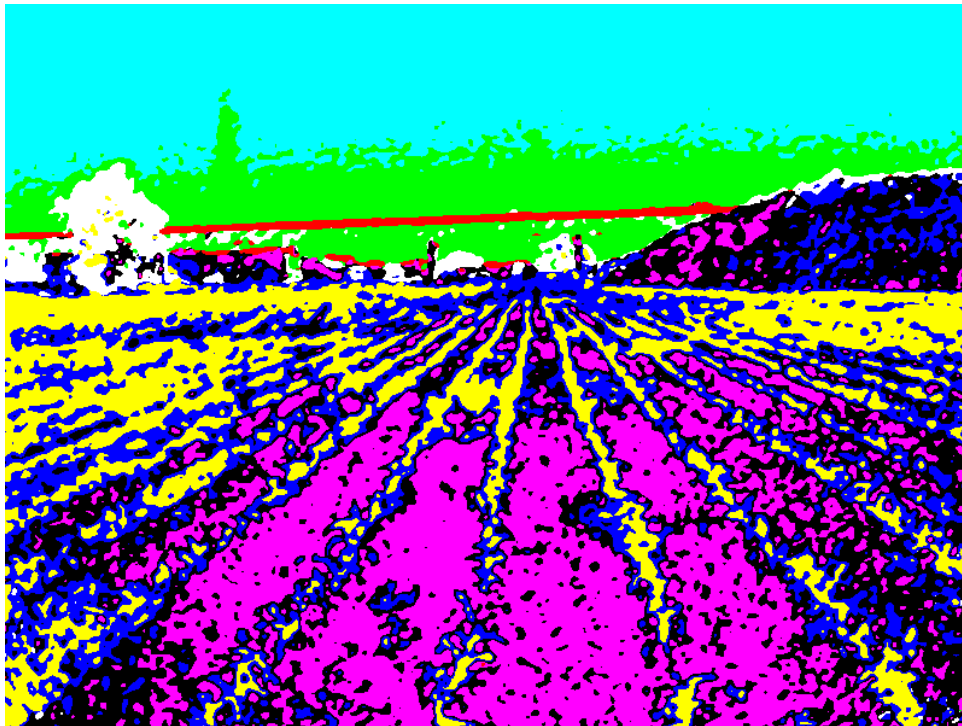


*Ilustración 32: Superposición del agrupamiento en 6 clases a la imagen original*

En la agrupación en 6 clases, puede observarse como uno de los dos nuevos grupos han servido para distinguir entre dos texturas diferentes presentes en lo que un humano clasificaría como cielo, y el otro se corresponde con zonas que un humano describiría como suelo con algo de vegetación.

Se siguen teniendo de todas formas clasificaciones ambiguas debidas a que el número de texturas en la imagen es mayor que el número de grupos empleados en el K-Means. Por ejemplo, el puente aparece clasificado en el mismo grupo que el suelo, cuando en realidad son dos elementos o texturas que nada tienen que ver y que serían clasificados separadamente por un humano. Este caso concreto no tiene mucho interés para nuestra aplicación porque lo que realmente nos interesa es la vegetación, pero de forma similar no se está separando bien la vegetación arbórea, que muestra más variabilidad que la vegetación en la zona de cultivo, y se está mezclando aquella, por partes, con los distintos grupos de esta.

Sin embargo, si seguimos aumentando el número de clases en el agrupamiento, vemos a continuación como la precisión mejora, al aproximarse el número de grupos al número real de elementos o texturas diferentes en la imagen.



*Ilustración 33: Agrupamiento en 8 clases*



*Ilustración 34: Superposición del agrupamiento en 8 clases a la imagen original*

En esta clasificación ya hay un nivel de precisión aceptable en cuanto a la homogeneidad de cada grupo clasificado, de forma que aunque no todos los grupos sean igual de relevantes de cara al objetivo del proyecto, al aumentar el número de estos se han conseguido principalmente dos cosas: poder descartar grupos sin interés con mayor fiabilidad, y disminuir el número de píxeles erróneos en los grupos relevantes, que ya no se ven tan "obligados" a absorber píxeles "huérfanos", entendiendo por pixel huérfano a un pixel que pertenece a una textura real que no tiene correspondencia con ninguno de las clases contempladas por K-Means (aunque aún sigue habiendo algunos, como las columnas del puente, que se han absorbido por uno de los grupos de suelo, lo que en principio y según lo expuesto se arreglaría si continuamos aumentando el número de clases a discernir).

Para ilustrar lo expuesto con una mayor claridad, a continuación se muestra la imagen original cortada en las diferentes 8 clases, nombrando descriptivamente cada una de ellas.

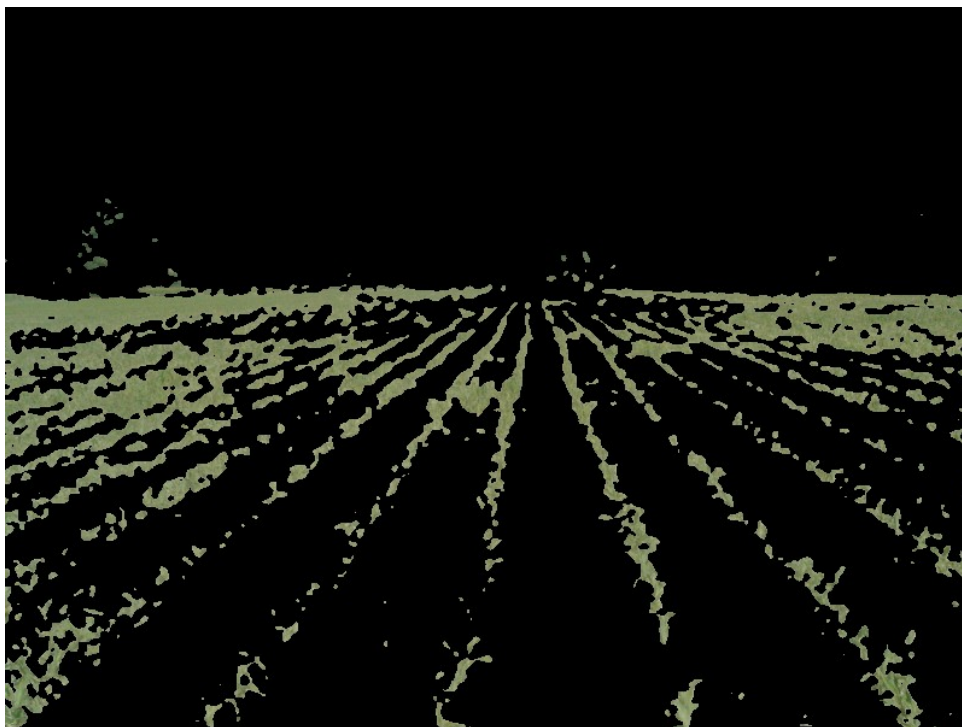
1. **Tierra:** Se agrupan los píxeles correspondientes a suelo sin apenas trazas de vegetación



*Ilustración 35: Píxeles clasificados como terreno sin vegetación*



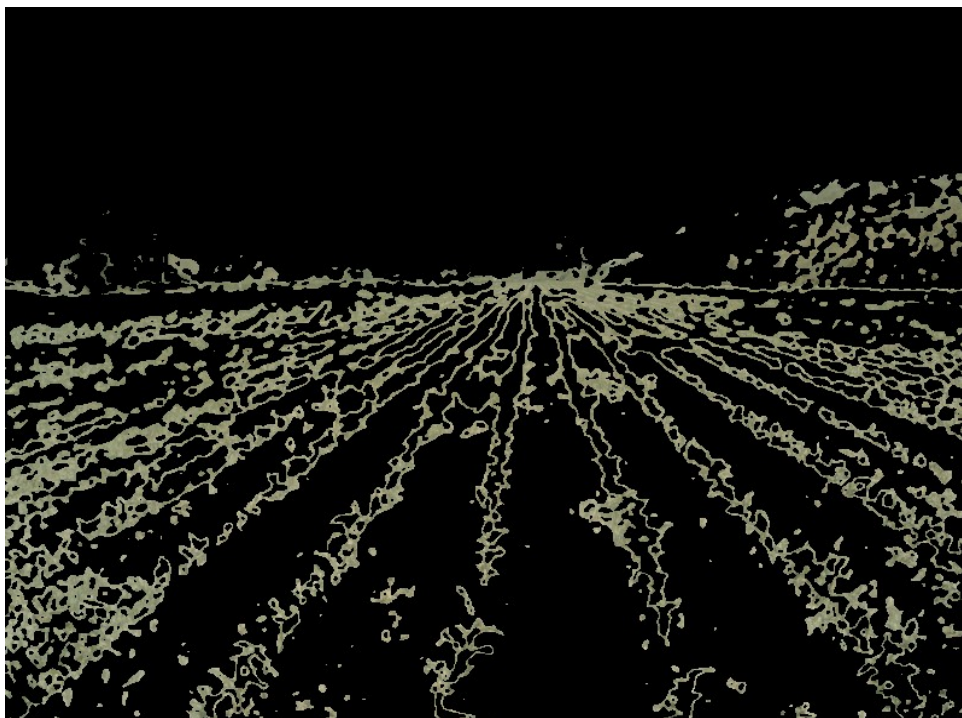
2. Cultivos: Agrupa los píxeles que un humano clasificaría como plantas cultivadas. Vease que aún se observan algunos píxeles que realmente pertenecen a un árbol mal incluidos en este grupo. Es de suponer que un mejor diseño del banco de filtros o un aumento del número de clases en la clasificación podrían solucionar este tipo de error.



*Ilustración 36: Píxeles clasificados como cultivo*



3. Pastos: Esta clase agrupa píxeles que un observador catalogaría como vegetación similar a los cultivos pero no correspondiente a los mismos, como por ejemplo pastos y hierbas altas, anexas a aquellos. Se puede observar ~~como~~ incluye tanto la vegetación que rodea al maíz como los pastos en la colina del fondo.



*Ilustración 37: Píxeles clasificados como pastos*

4. Hierbas: Incluye los píxeles correspondientes al suelo donde existe cierta vegetación baja que matiza la textura del mismo, como hierbas. Se incluyen ~~erroneamente~~ en esta clase las columnas del puente, ~~al no haber~~ una clase exclusiva para ellos y tener una textura muy parecida debido a la resolución de la imagen.



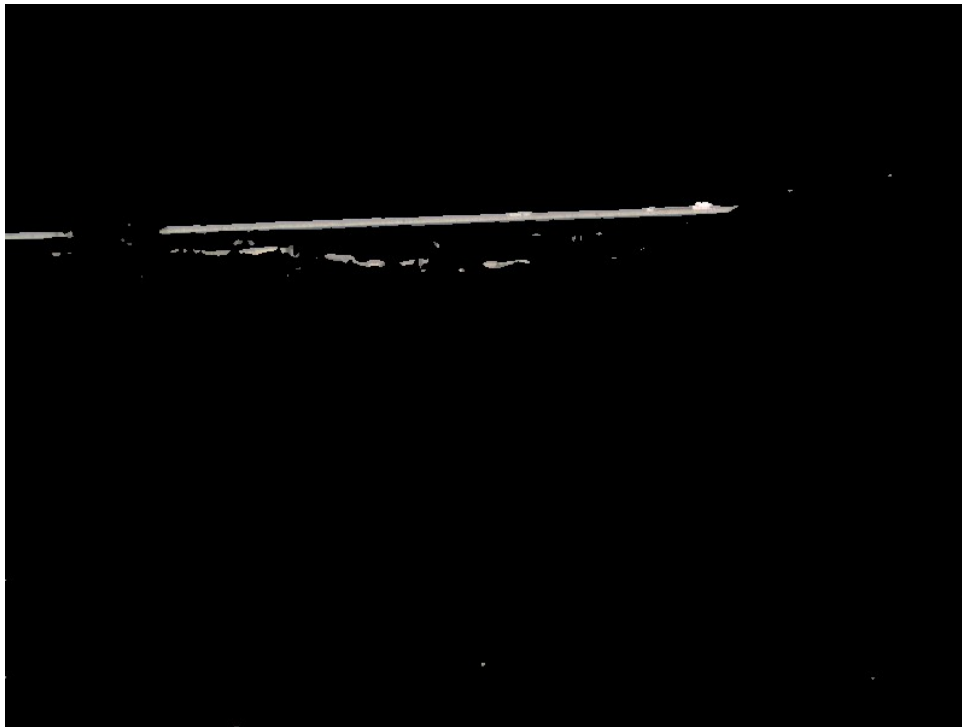
*Ilustración 38: Píxeles clasificados como hierbas*

5. **Arboleda:** Incluye los píxeles correspondientes a los árboles y arbustos presentes en la imagen. Incluye erróneamente algunos píxeles del puente con una tonalidad similar. Probablemente una imagen de mejor resolución, un diseño más óptimo del banco de filtros y el uso de más clases en el agrupamiento son factores que ayudarían a corregir este tipo de errores. No obstante la aproximación como se ve en la imagen es aceptable.



*Ilustración 39: Píxeles clasificados como arboleda*

6. Sin interés tipo I : incluye píxeles que corresponden a la parte superior del puente y de lo que ~~parecen ser casas~~ que sobresalen de entre la vegetación en el horizonte, similares en cuanto a la textura observada, pero sin interés ~~para nosotros~~.



*Ilustración 40: Píxeles sin interés, tipo I*

7. Sin interés tipo II: Agrupa los píxeles que corresponden a la parte más baja del cielo, cuya textura se ve matizada por la presencia de una serie de torres y tendido eléctrico.



*Ilustración 41: Píxeles sin interés, tipo II*

8. Sin interés tipo III: Píxeles correspondientes a cielo abierto.



*Ilustración 42: Píxeles sin interés, tipo III*

El hecho de tener estos tres últimos grupos es de relevancia no por el interés de los mismos, sino porque sirven para ~~sacar~~ dichos píxeles de los grupos donde se clasificaban ~~erroneamente~~ al usar ~~menos~~ clases. Una vez que se tienen aislados, pueden simplemente descartarse, y los grupos de interés son realmente más precisos y homogéneos.

## 6. Conclusiones

Del trabajo realizado puede llegarse a varias conclusiones respecto a los diferentes aspectos tratados en relación a la segmentación de texturas.

En primer lugar, y por seguir el orden en el que se han mencionado en el presente documento, respecto a la elección del espacio de colores se confirma lo afirmado en [12] sobre la criticidad de una correcta elección del espacio de color adecuado. Se ha comprobado en los experimentos realizados que los resultados de clasificación son más intuitivos trabajando con un espacio de colores perceptualmente uniforme, aunque dicho artículo se limitaba al espacio HSI. En nuestro caso, trabajando con CIELab conseguimos que:

- La distancia matemática entre colores se corresponda más fielmente con la diferencia percepción humana
- La luminosidad no sea necesaria para obtener buenos resultados de clasificación

Ambos puntos influyen en que la clasificación sea más acorde a la esperada según la percepción visual de la imagen (que corresponde a los elementos naturales reales presentes en la misma). Además el segundo punto nos protege en cierta medida de condiciones ambientales adversas habituales en imágenes naturales; e incluso disminuye la potencia de procesamiento necesaria respecto a RGB al trabajar sólo con dos canales.

En segundo lugar, respecto a la posibilidad de realizar un análisis espectral sobre imágenes naturales, puede concluirse por los resultados observados que es perfectamente posible hacerlo obteniendo buenos resultados de clasificación. Se abre aquí una puerta a mayor experimentación, incluso al aprendizaje supervisado. No obstante, también se ha observado la criticidad del diseño del banco de filtros empleado. El implementado en el presente proyecto es el que mejor resultado ha ofrecido tras una gran cantidad de pruebas con diferentes variaciones de parámetros, pero probablemente un estudio más detallado y cuidadoso matemáticamente hablando del problema permitiría encontrar un diseño más óptimo que ofreciera mejores resultados.

Además, en cuanto a la eficiencia, hemos comprobado que aplicar un filtro de Gabor puede reducirse a la convolución de filtros lineales precalculados.

Por último, respecto al agrupamiento en clases de las texturas descritas, vemos cierta problemática con la aplicación del K-Means, sobre todo respecto a dejar en manos del usuario decidir cuántos clusters hay que formar. Por un lado, formar demasiados clusters introduce una complejidad innecesaria, pero por otro, formar menos implica que los resultados no son intuitivos debido a que se "fuerza" al algoritmo a incluir píxeles que no deberían pertenecer a ninguno de los grupos en alguno de ellos.

Una solución a este respecto podría ser la propuesta en la sección 4.3 y en [20] de ir iterando sobre diferentes valores de  $K$  hasta ~~llegar~~ al valor óptimo. Pero quizá aún más adecuado sería probar otros algoritmos de agrupamiento más sofisticados que sean capaces de ajustar ellos mismos el número de ~~clústeres~~ a la complejidad de la imagen.



## 7. Trabajo futuro

La principal fuente de imprecisiones en el método presentado ~~parece venir~~ del empleo del algoritmo K-Means para el agrupamiento. Si bien es verdad que es un algoritmo simple y rápido, de gran eficiencia, quizá pudiera mejorarse la eficacia de la clasificación utilizando algún método más sofisticado capaz de ajustar el número de ~~clusters~~ necesarios a la complejidad de la imagen analizada.

Además, sería muy interesante comprobar el efecto de utilizar medidas de distancia alternativas a la hora de agrupar los vectores de características. De especial interés sería probar con la **EMD** o con la distancia de Mahalanobis, y analizar qué efectos provocan.

También hubiera sido de ~~mucho~~ interés haber dispuesto de imágenes de una misma zona en diferentes condiciones ambientales de iluminación, para comprobar hasta qué punto el haber prescindido del canal de luminosidad en el espacio CIELab nos ~~aisla~~ de errores debido a dichas condiciones.

En cuanto a la caracterización en sí de las texturas, el diseño del banco de filtros utilizado en este ~~proyecto~~ ha resultado ~~de~~ meramente ~~de~~ un proceso iterativo de prueba y error, ~~pero un~~ estudio cuidadoso de los parámetros más adecuados al tipo de imágenes tratados seguramente llevaría a un aumento definitivo en la precisión de la clasificación. En cuanto a la variabilidad de las texturas en las imágenes naturales ~~hay~~ aún una serie de problemas a atacar, a la hora de analizarlas por regiones, como son:


1. La escala básica de algunas texturas puede ser y será con frecuencia mayor que la mayor escala utilizada en los filtros de Gabor, lo que impide que los mismos sean capaces de capturar correctamente la misma.
2. Siempre habrá píxeles frontera en los cuales el filtro aplicado abarcará zonas con texturas diferentes, lo que producirá información no válida y que incluso puede ~~suponer~~ ruido en el espacio de características.
3. Condiciones de ruido debidas a iluminación, desenfoques, y similares aumentan aún más la variabilidad en la imagen.
4. Incluso en condiciones ideales, las texturas naturales se caracterizan por una gran variabilidad espacial, siendo regulares sólo desde un punto de vista estadístico.

Estos puntos son los que ~~veo~~ como grandes retos y objetivos de ~~futuro trabajo~~. Hay ya ciertas propuestas, como por ejemplo el pretratamiento de las imágenes expuesto en [16] y [18] mediante un método de suavizado preservando los bordes en el espacio de características. Sería interesante comprobar los resultados de su aplicación en nuestro problema.

## 8. Listado de acrónimos

- CIE: Commission Internationale de l'Eclairage
- CMYK: Cyan, Magenta, Yellow, black
- CRT: Cathod Ray Tube
- EMD: Earth Mover's Distance
- HSI: Hue, Saturation, Intensity
- HSL: Hue, Saturation, Lightness
- RGB: Red, Green, Blue

## 9. Referencias

- 
- [1] M. Guijarro a,\*, G. Pajares , I. Riomoros , P.J. Herrera , X.P. Burgos-Artizzu , A. Ribeiro; **Automatic segmentation of relevant textures in agricultural images**; Computers and Electronics in Agriculture 75 (2011) 75–83
- [2] Robot Fleets for Highly Effective Agriculture and Forestry Management;  
<http://www.rhea-project.eu/>
- [3] Gonzalo Pajares, J.M. de la Cruz; **Ejercicios Resueltos de Visión por Computador**; 1ª edición, RA-MA, Madrid, 2007
- [4] Gonzalo Pajares y Jesús M. de la Cruz; **Visión por computador: imágenes digitales y aplicaciones**; 2ª edición, RA-MA, Madrid, 2007
- [5] Ramesh Jain, Rangachar Kasturi, Brian G. Schunck ; **Machine Vision**; McGraw Hill 1995 (Capítulo 7, Texture)
- [6] G. Pajares, M. Guijarro, P.J. Herrera, A. Ribeiro; **Combining classifiers through fuzzy cognitive maps in natural images**;
- [7] Gonzalo Pajares, María Guijarro, Angela Ribeiro; **A Hopfield Neural Network for combining classifiers applied to textured images**; Neural Networks 23 (2010) 144–153
- [8] D. Puig, S. Alvarez, P. Sobrevilla, E. Montseny; **Analysis of natural textures using fuzzy techniques**;
- [9] María Guijarro, Raquel Abreu, Gonzalo Pajares; **On combining Learning Vector Quantization and the Bayesian classifiers for natural textured images**; II Congreso Español de Informática, IV Taller de Minería de Datos y Aprendizaje, 196-201
- [10] V. Shiv Naga Prasad, Justin Domke; **Gabor Filter Visualization**;
- [11] Naotoshi Seo; **Texture Segmentation using Gabor Filters**; ENEE731 Project, 2006
- [12] Leena Lepistö, Iivari Kunttu, Jorma Autio, Ari Visa; **Classification Method for Colored Natural Textures Using Gabor Filtering**;
- [13] Thomas P. Weldon, William E. Higgins, and Dennis F. Dunn; **Efficient Gabor filter design for texture segmentation**;
- [14] Yong Huang, Kap Luk Chan, Zhihua Zhang; **Texture classification by multi-model feature integration using Bayesian networks**;
- [15] Yossi Rubner, Jan Puzicha, Carlo Tomasi, Joachim M. Buhmann; **Empirical Evaluation of Dissimilarity Measures for Color and Texture**;
- [16] Yossi Rubner, Carlo Tomasi; **Coalescing Texture Descriptors**; ARPA Image Understanding Workshop 1996
- [17] Yossi Rubner, Carlo Tomasi; **Texture Metrics**;
- [18] Yossi Rubner; **Perceptual metrics for database image navigation**; PhD Thesis 1999

- [19] Javier R. Movellan; **Tutorial on Gabor Filters**;
- [20] Gary Bradski and Adrian Kaehler; **Learning OpenCV**; O'Reilly, September 2008, Primera Edición.

## 10. Anexo A: Código fuente

La implementación del método expuesto utilizada en las pruebas fue escrita en lenguaje C, haciendo uso intensivo de la librería OpenCV.

No se incluye aquí el código fuente por no considerarlo necesario, pero el mismo está alojado en un repositorio privado en Bitbucket ([www.bitbucket.com](http://www.bitbucket.com)) y disponible bajo petición a [mafraba@gmail.com](mailto:mafraba@gmail.com).