

Sistemas de percepción

Máster Universitario en Ingeniería Industrial

Diseño e implementación de un algoritmo para identificar campos y bosques

Mini-Proyecto

Autores: Ernest Arranz Pasqual

Roger Sala Guadaña

Profesor: Alberto Sanfeliu

Data entrega: 19/5/2024



Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona



Índice

Índice.....	1
1. Resumen.....	1
2. Introducción.....	2
2.1 Pruebas previas.....	2
3. Explicación del método.....	3
3.2. Aplicación.....	5
4. Resultados.....	9
4.5 Otras pruebas.....	11
5. Conclusiones.....	13
6. Referencias.....	14

1. Resumen

Hoy en día la tecnología está presente en prácticamente todas las áreas tanto de la vida cotidiana, así como también de la profesional. Su evolución ha permitido crear nuevos sectores, pero también ha permitido que sectores que llevan prácticamente desde el origen del ser humano hayan mejorado exponencialmente. Un claro ejemplo de este fenómeno es la agricultura, con la que este proyecto está relacionado.

La evolución de la agricultura gracias a la tecnología ha sido notable a lo largo de la historia. En su momento, la aparición de máquinas permitió aumentar la productividad. Más tarde se introdujo el uso de fertilizantes y pesticidas gracias a la ingeniería química. Posteriormente, gracias a la biotecnología se crearon cultivos transgénicos más resistentes a plagas y enfermedades. Actualmente, en la era digital, se utilizan sistemas de automatización y robótica para realizar tareas agrícolas con mayor eficiencia y menos intervención humana, así como sistemas de visión por computador para monitorear y gestionar cultivos.

El objetivo de este mini-proyecto es desarrollar un programa capaz de identificar y etiquetar diferentes zonas de agricultura sobre imágenes aéreas, utilizando técnicas vistas en la asignatura de Sistemas de percepción y programando en MATLAB.

El enfoque principal que se ha utilizado es el uso de matrices de co-ocurrencia de niveles de grises o GLCM, por sus siglas en inglés *Gray Level Co-occurrence Matrix*. Dadas unas muestras de los diferentes elementos a identificar, el algoritmo obtiene las propiedades de la textura de cada una de ellas, que posteriormente usa para analizar diferentes escenas en las que va identificando y comparando subsecciones de estas mediante una ventana deslizante. Si la diferencia entre esas propiedades es menor a un umbral, se considera buena y etiqueta esa región.

2. Introducción

Cada vez más, se está utilizando la visión por computador en el campo de la agricultura. Es lo que se conoce como agricultura satelital [1]. Esta ofrece muchas ventajas complementarias a la agricultura convencional, como clasificación precisa de los diferentes cultivos, identificación de plagas y un monitoreo constante de los campos. La identificación precisa de las áreas forestales y agrícolas no solo afecta considerablemente de manera positiva al medio ambiente, sino que también es crucial para la planificación del uso del suelo y la toma de decisiones en la gestión sostenible de los recursos.

2.1 Pruebas previas

Para empezar buscamos cuáles son las técnicas existentes para tareas parecidas a fin de evaluarlas [2] [3].

Primero descartamos algoritmos basados en redes neuronales como U-Net, pues no encontramos un dataset adecuado de suficiente calidad para entrenar una, y no teníamos los medios disponibles para crearlo.

Hay unos cuantos algoritmos que aún y ser populares no son adecuados para esta tarea debido a que se limitan a segmentar, sin ser capaces de etiquetar. Esto descarta algoritmos como k-means [4], watershed [5] y segmentación basada en entropía [6].

Un algoritmo que puede ser usado para etiquetado son los filtros de Gabor [7], pero después de unas pruebas iniciales los resultados no eran suficientemente buenos y se desestimó.

Finalmente, el algoritmo elegido fue uno basado en matriz de co-ocurrencias [8]. Primeramente, se han tomado diferentes imágenes satelitales de campos de cultivos de Tarragona con Google Maps [9] en las que se aplicará el algoritmo de identificación. Asimismo, se han tomado muestras de cada uno de los diferentes elementos que se quieren identificar, con las que se extraerán las propiedades de la textura. Para este proyecto se identifican dos tipos de cultivos a los que llamamos campos y árboles. A continuación se muestran ejemplos de los campos a identificar:

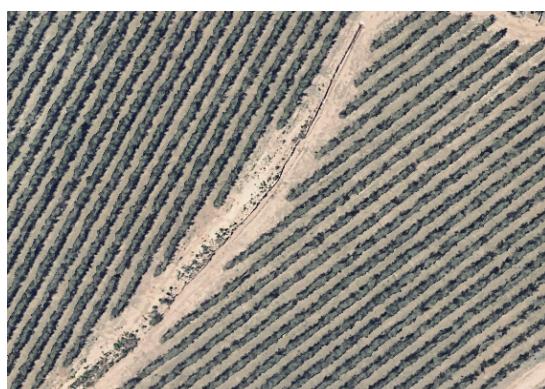


Figura 1. Ejemplo de campos



Figura 2. Ejemplo de Árboles

Para simplificar el proyecto, se han tomado todas las imágenes a una misma escala (mismos píxeles / metro), puesto que la textura se ve muy afectada por esta métrica y es muy difícil elaborar algoritmos de etiquetado de texturas invariantes a escala. Esto no invalida el propósito del mini-proyecto, ya que podría conseguirse fácilmente si tomásemos las fotos con un dron que siempre vuela a una distancia constante del suelo.

3. Explicación del método

La matriz de co-ocurrencias es una herramienta utilizada en la identificación y etiquetado de texturas en imágenes. Su objetivo es capturar la relación espacial entre los píxeles de una imagen. Básicamente, es una tabla que muestra la frecuencia con la que diferentes combinaciones de niveles de gris (valores de píxeles) ocurren en una imagen, considerando una relación espacial específica.

Esta matriz será de dimensión $n \times n$ siendo n los niveles de gris que puede tener la imagen. En otras palabras, los diferentes valores que pueden tener los píxeles de la imagen. Como se ha comentado, cada matriz de co-ocurrencias irá condicionada a la relación espacial entre píxeles que se quiera estudiar.

Consideremos que se quiere analizar la textura de la siguiente imagen:

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

Queremos analizar la relación entre píxeles en la dirección horizontal hacia la derecha. Puesto que los valores de la imagen van del 0 al 3 (4 niveles distintos) la matriz de co-ocurrencias será en este caso de 4×4 .

	0	1	2	3
0				
1				
2				
3				

Ahora contamos las ocurrencias de cada par de niveles de gris. Por ejemplo, buscamos la pareja (0,1), es decir, los 1 que tengan un 0 a la derecha y vemos que hay 3 casos en los que esto se cumple.

$$\begin{array}{cccc}
 0 & 1 & 2 & 3 \\
 1 & 0 & 1 & 2 \\
 2 & 1 & 0 & 1 \\
 3 & 2 & 1 & 0
 \end{array}$$

Colocamos en la casilla correspondiente el número:

	0	1	2	3
0				
1	3			
2				
3				

Repitiendo este procedimiento para todos los pares obtenemos la matriz de co-ocurrencias:

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

Una vez obtenida la matriz, toda entera son muchos parámetros, y no hay una manera clara de comparar dos matrices de este estilo, con lo que para simplificar el proceso y abstraer un poco más que define una textura, se pueden extraer las características de la matriz, que no son otra cosa que parámetros estadísticos de la distribución de valores en la matriz. Las características empleadas son 4:

Contraste: Diferencia de intensidad entre un píxel y sus vecinos.

Homogeneidad: Medida de cuán uniformes son los niveles de gris.

Energía: Suma de los cuadrados de los valores de la matriz, indicando uniformidad.

Entropía: Medida de la aleatoriedad en la imagen.

Se calculan mediante las siguientes fórmulas matemáticas:

$$\text{Entropy} = - \sum_i \sum_j P[i, j] \log P[i, j].$$

$$\text{Energy} = \sum_i \sum_j P^2[i, j]$$

$$\text{Contrast} = \sum_i \sum_j (i - j)^2 P[i, j]$$

$$\text{Homogeneity} = \sum_i \sum_j \frac{P[i, j]}{1 + |i - j|}$$

Estas características son las que utiliza el algoritmo para comparar las muestras con las escenas.

3.2. Aplicación

El programa consiste en un *live script* de MATLAB llamado [*ClassificationMain mlx*](#), estructurado en diferentes apartados::

Clean the workspace

Limpiamos el entorno de trabajo de variables y figuras restantes de ejecuciones anteriores.

Parameters

Cargamos todos los parámetros que usará el programa. Aquí se definen la localización de las muestras y escenas, su pre procesado, así como la configuración de la extracción de propiedades y los umbrales para las comparaciones, entre otros.

Load sample filepaths

Para no tener que definir la localización de todas y cada una de las muestras, el programa en su lugar carga como muestras todas las imágenes en las carpetas indicadas en la sección de parámetros.

Process samples

Antes de extraer las características, primero se pre procesan las muestras para adecuarlas a la tarea. Se convierten las muestras a escala de grises si no lo son y también, puesto que todas las regiones del mismo campo no tienen por qué tener la misma textura exacta, se aplica un filtro gaussiano para homogeneizar ligeramente la textura y hacerla más general.



Figura 3. Imagen antes del filtro



Figura 4. Imagen después del filtro

El filtro gaussiano es una técnica utilizada en procesamiento de imágenes para suavizar y reducir el ruido en una imagen. Se basa en la función gaussiana, que da más peso a los píxeles cercanos al punto central y menos a los más alejados, creando un efecto de desenfoque suave.

Seguidamente, el algoritmo aplica permutaciones de rotación y extrae las características de las muestras.

Sample Properties

A fin de buscar relaciones entre las diferentes muestras de un mismo grupo, se puede elegir graficar unos histogramas de las diferentes propiedades para buscar semejanzas, aunque no parece haber un patrón claro.

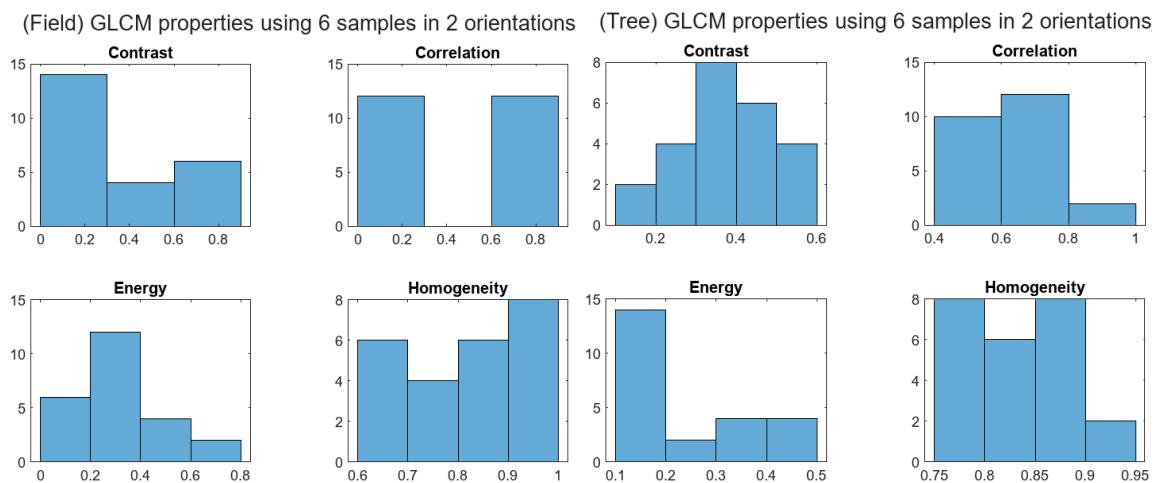


Figura 5. Histogramas sobre las propiedades extraídas

Process Scene

Parecido al pre procesado de las muestras, se convierte la escena a escala de grises y se aplica un suavizado.

Compute Errors

Para detectar texturas, mediante una ventana deslizante del mismo tamaño que las muestras con un paso variable estipulado como parámetro, se va analizando cada región de la escena y extrayendo sus características.



Figura 6. Aplicación de la ventana deslizante

Una vez analizadas estas regiones, el algoritmo compara las propiedades con las de las muestras y calcula el error como la suma de las diferencias absolutas entre las diferentes características.

Find matching cases

Si la diferencia es menor a un umbral, la imagen identifica esa región en una máscara como campo o árboles. Este procedimiento se repite para los dos tipos de elementos a identificar.

Una vez completado, se grafican las dos máscaras parciales, una de cada tipo, donde se puede ver una primera etiquetación.

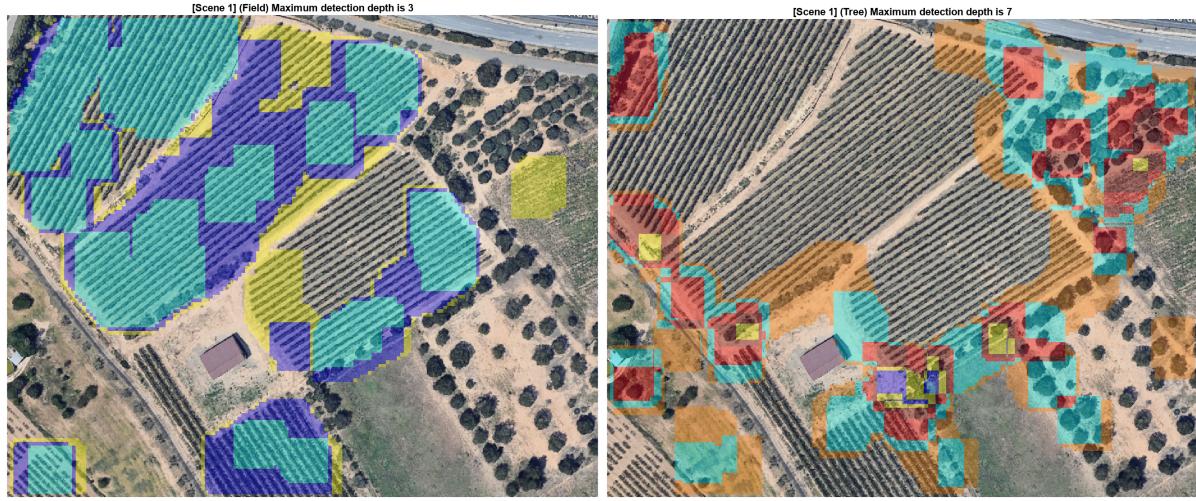


Figura 6. Máscaras parciales para la escena 1
(Campos y árboles respectivamente)

Combine masks

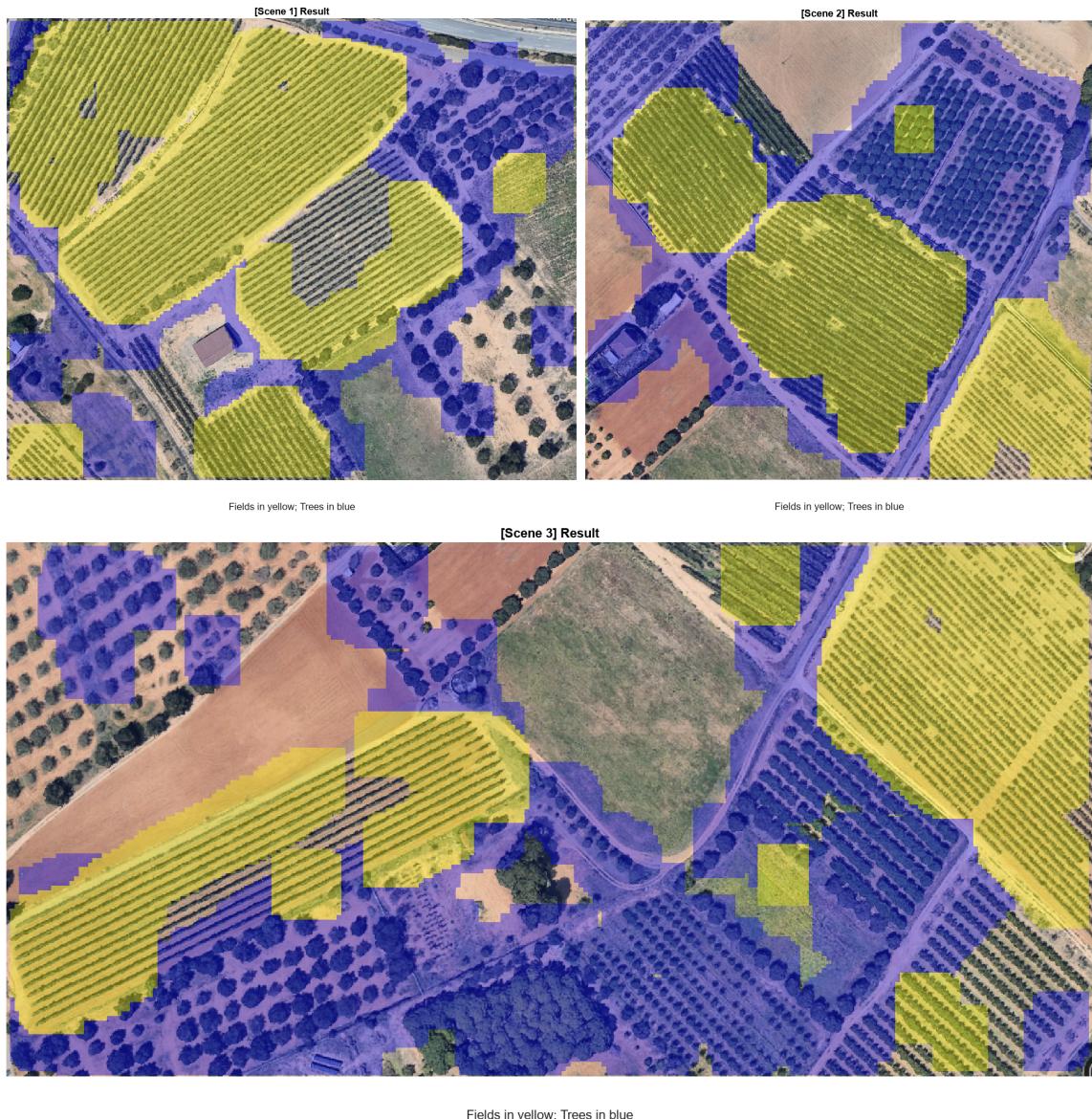
Es probable que haya regiones que hayan sido identificadas tanto como campos como árboles. En este caso, se comparan los errores respectivos para regiones en conflicto y se asigna finalmente la etiqueta del tipo que menor error tiene mediante una comparación ponderada. Se pondera para poder compensar imágenes que inherentemente tienen más error, puesto que son menos únicas o más difíciles de abstraer.

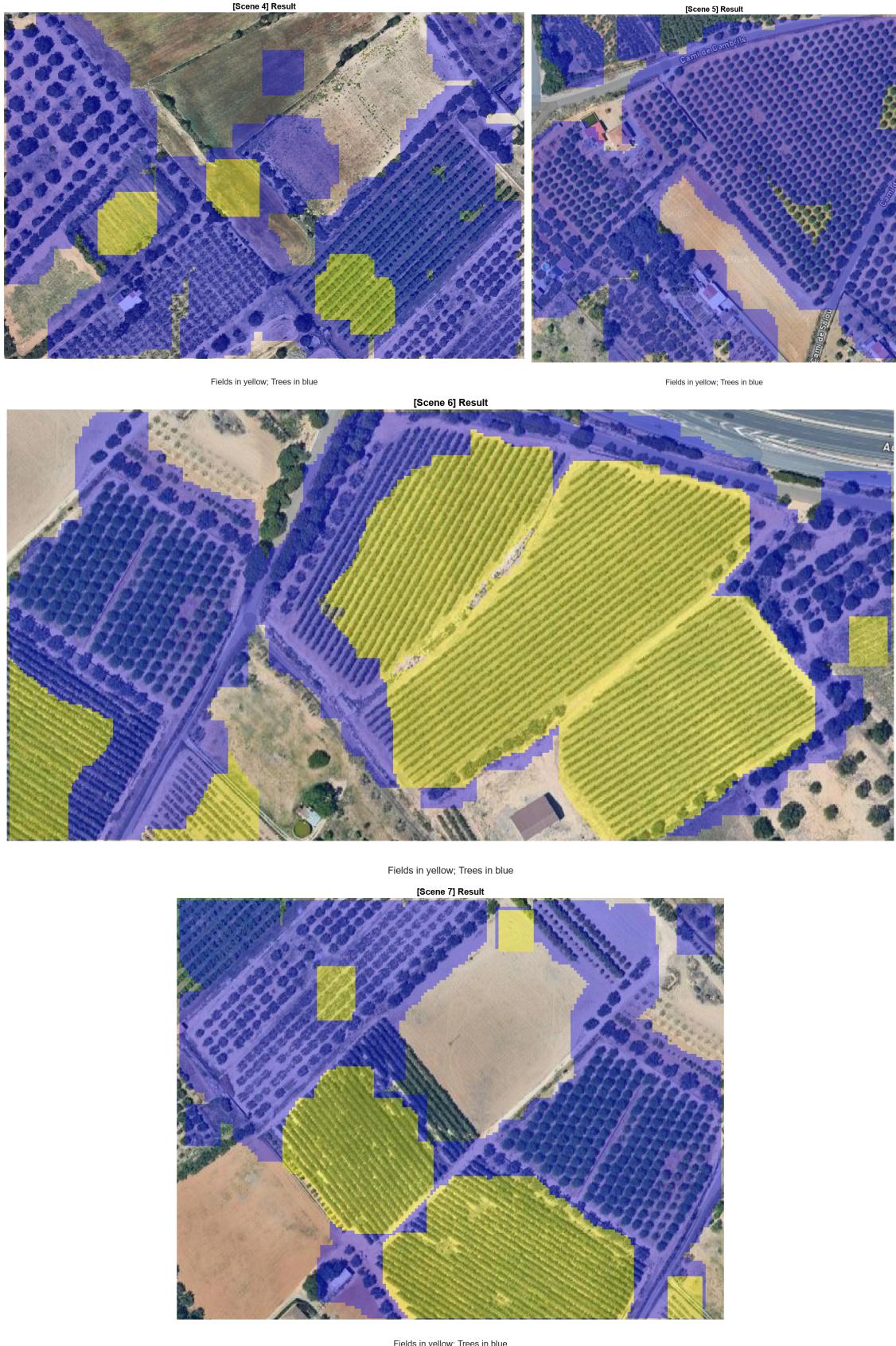


Figura 7. Resultado final para la escena 1
(Campos en amarillo y árboles en azul)

4. Resultados

Finalmente, se ajustaron estos parámetros para obtener los mejores resultados posibles en tres escenas diferentes y se aplicó el mismo algoritmo para 5 escenas más para comprobar su efectividad. Los resultados obtenidos se muestran a continuación (campos en amarillo y árboles en azul):



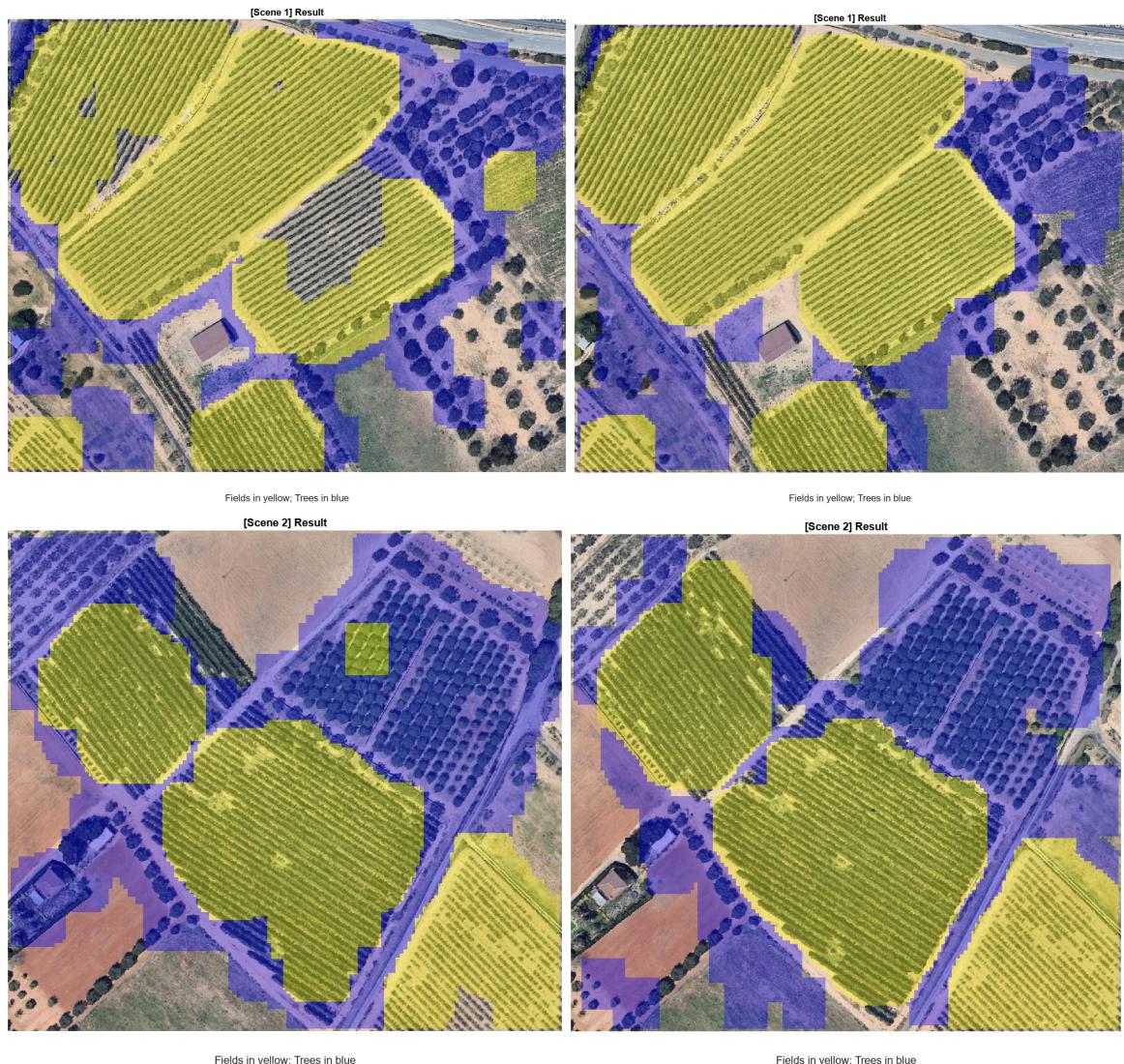


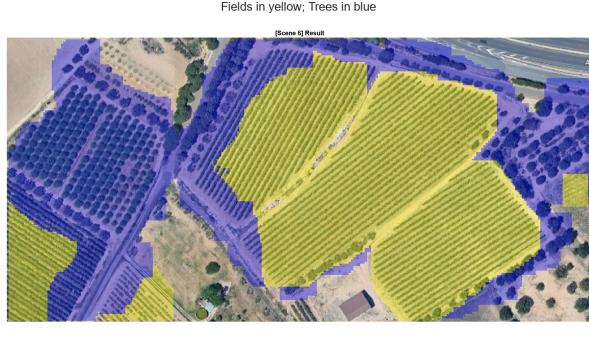
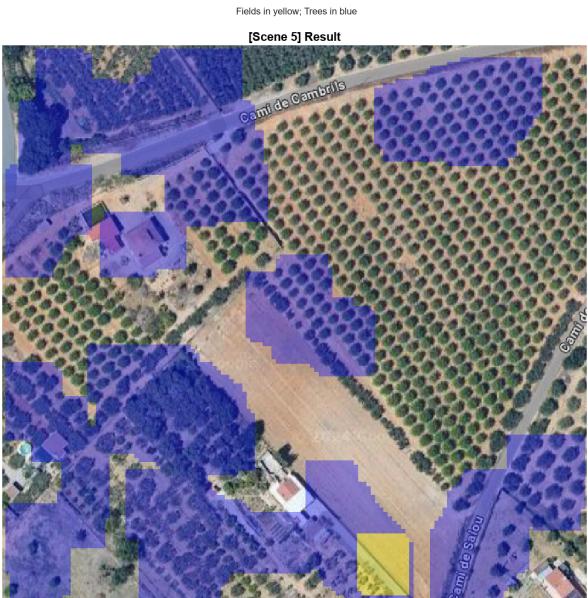
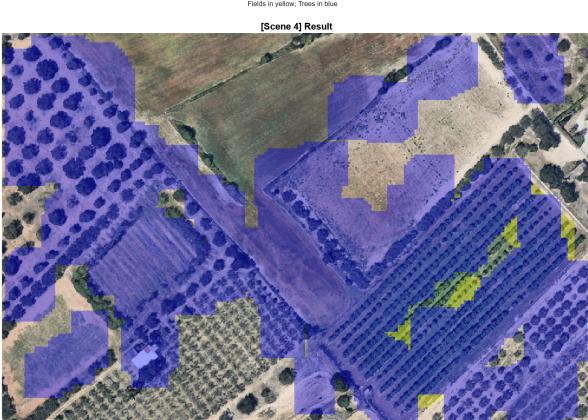
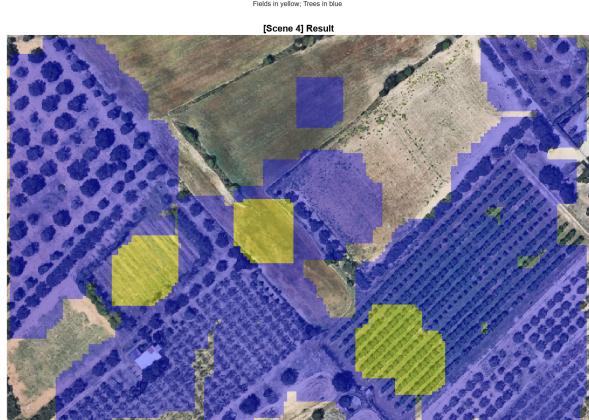
**Figura 8. Resultados finales para las diferentes escenas
(Campos en amarillo y árboles en azul)**

Como se puede observar, las grandes extensiones se identifican bastante bien. También hay algunas falsas identificaciones de árboles que se identifican como campo y viceversa. Hay también algunas regiones que no se identifican con ningún elemento, así como algunos campos sin cultivo que quedan identificadas como campo.

4.5 Otras pruebas

Para intentar mejorar los resultados se ha probado a aplicar el algoritmo, no sobre la imagen en escala de grises, sino sobre cada canal de color (implementación en [*ClassificationRGBTest.mlx*](#)). A continuación puede verse las diferencias (sobre escala de grises a la izquierda, por canal de color a la derecha; una escena por fila):





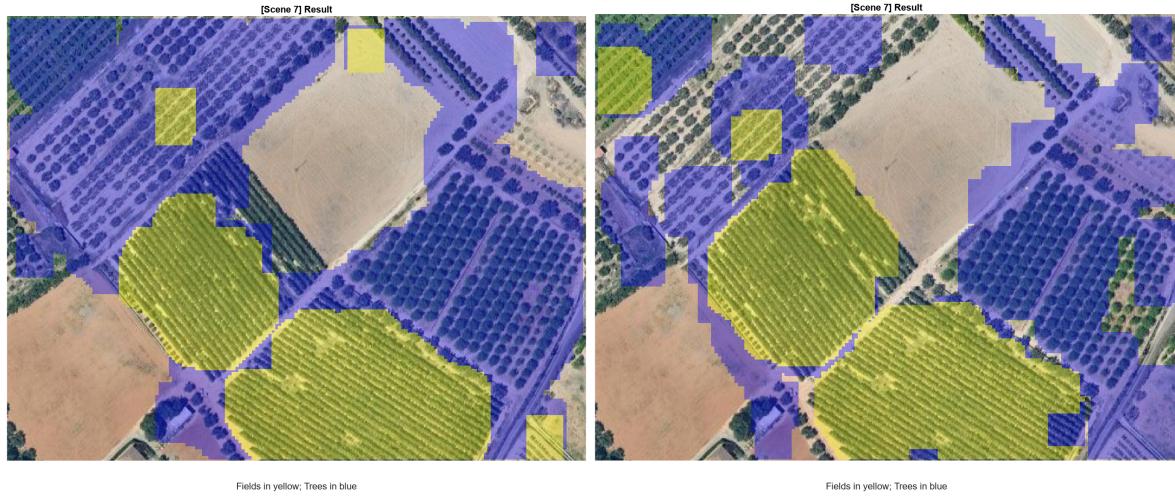


Figura 9. Comparación entre aplicación sobre escala de grises (izquierda) y por canal (derecha).
(Campos en amarillo y árboles en azul)

Los resultados son variados, en algunos casos mejora (como en las escenas 1, 2, y 6) y en otros empeora (como en las escenas 4 y 5). Parece que la información adicional aportada por los colores en general beneficia a la detección de campos, pero dificulta la detección de árboles. Combinado con el aumento de la carga computacional que supone la modificación, no parece compensar, aunque puede que con más pruebas y ajuste de los parámetros en un futuro si pueda superar al algoritmo basado en escala de grises.

5. Conclusiones

Si bien el resultado no es perfecto y tiene margen de mejora, se ha logrado cierto nivel de identificación y etiquetado.

En un futuro y a fin de mejorar el resultado, se podría probar a pre procesar las muestras y escenas para estandarizar la escala, haciendo más flexible la toma de datos y también se podría buscar la manera de incluir de alguna manera la información aportada por los colores que en la implementación actual basada en escala de grises no se usan.

6. References

- [1] Teledetección e imágenes satelitales en agricultura gana representatividad. Agrawdata:
<https://agrawdata.com/blog/teledeteccion-imagenes-satelitales-agricultura/>
- [2] Review of texture image analysis and texture classification methods. Arxiv:
<https://arxiv.org/pdf/1904.06554>
- [3] Comparison between Five Traditional Algorithms and a DL U-Net Architecture. MDPI:
<https://www.mdpi.com/536090>
- [4] k-means L*a*b color segmentation. Matlab Examples:
<https://es.mathworks.com/help/images/color-based-segmentation-using-k-means-clustering.html?lang=en>
- [5] Watershed segmentation. Matlab Examples:
<https://es.mathworks.com/help/images/marker-controlled-watershed-segmentation.html?lang=en>
- [6] Entropy based texture segmentation. Matlab Examples:
<https://es.mathworks.com/help/images/texture-segmentation-using-texture-filters.html?lang=en>
- [7] Texture segmentation based on Gabor filters. Matlab Examples:
<https://es.mathworks.com/help/images/texture-segmentation-using-gabor-filters.html?lang=en>
- [8] Gray level co-occurrence matrix (GLCM). Matlab:
<https://es.mathworks.com/help/images/ref/graycomatrix.html?lang=en>
- [9] Imágenes de los cultivos. Google Maps:
<https://www.google.es/maps/@41.0992122,1.1413426,491m/data=!3m1!1e3?hl=ca&entry=ttu>