Universidad Autónoma de Madrid
Escuela Politécnica Superior

Trabajo de fin de grado

CREAR ESCENARIOS 3D A PARTIR DE FOTOS

Grado en Ingeniera Informática

Carlos López Cruces
Mayo 2013

CREAR ESCENARIOS 3D A PARTIR DE FOTOS

AUTOR: Carlos López Cruces
TUTOR: Héctor Menéndez
PONENTE: David Camacho

Grupo de la EPS AIDA (Applied Intelligence and Data Analysis)
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo 2013

Resumen

Resumen

Durante los últimos años, se han utilizado muchos sistemas para reconstruir escenarios tridimensionales partiendo de imágenes o vídeos. La expansión de estas herramientas va acompañada del proliferante mercado de los videojuegos y de los avances en el mundo de la robótica, especialmente dentro de la navegación espacial[1]. Existen más aplicaciones, por ejemplo, en la Medicina[2] o la Astrofísica[3].

Algunos de estos sistemas utilizan los algoritmos de segmentación de imagen[4] para identificar objetos y zonas de interés, y así poder estimar una profundidad, ya sea partiendo de una[5] o varias imágenes[6].

Este trabajo se basa en una de estas aplicaciones: Make3D[7, 8, 9]. Esta aplicación se utiliza para realizar una reconstrucción 3D a partir de una única imagen. Este proyecto aporta ampliaciones funcionales del framework de la herramienta Make3D permitiendo la utilización de filtros de imagen y varios algoritmos de segmentación. El enfoque principal es dar una mayor versatilidad a la herramienta, mejorando los resultados de la reconstrucción a través de la mejora de los resultados de la segmentación.

Para poder cuantificar la mejora de la segmentación, se añade a la arquitectura un sistema de evaluación automática que obtiene las métricas de Precision y Recall a través de imágenes previamente segmentadas por personas.

Palabras Clave

Segmentación Imagen, Reconstrucción 3D, Filtro de Imagen, Clustering, Make3D

Abstract

Over the last few years, several systems have been applied generate a three-dimensional environment through pictures or videos. These methodologies have been recently improved is several fields such as video-game and Robotics[1], amongs others. Others relevant applications are, for example, Medical field[2] and Astrophysics[3].

They use image segmentation algorithms to recognize objects and areas of interest. This information is used to estimate a depth, using a single image[5] or more images[6].

This proyect is based on one of these tools: Make3D[7, 8, 9]. It is used to generate a three-dimensional scene using a single image. This project provides extensions for the Make3D framework such as the use of image filters and different image segmentation algorithms. The goal is to provide more versatility to the tool. This new framework improves the reconstruction quality enriching the segmentation results.

To measure the segmentation results, an automatic evaluation system to calculate Precision and Recall metrics (compared with previous human-based segmentations) has been included in the framework.

Key words

Image Segmentation, 3-d Reconstruction, Image Filter, Clustering, Make3D

Agradecimientos

Índice general

Índice de figuras

Índice de tablas

1 Introducción

En Data Mining[10] se utilizan técnicas como la segmentación para separar instancias en grupos que tengan características similares. Un caso especial es la segmentación en imágenes, que permiten identificar objetos distintos o zonas. Uno de los usos que se le pueden dar a esta técnica es la reconstrucción de una imagen simple a un escenario 3D. Esta técnica aprovecha la segmentación para aproximar la profundidad geométrica de la imagen, suponiendo una perspectiva cónica.

Este proyecto trata de crear una arquitectura que aplique filtros de imagen y algoritmos de segmentación de imágenes para poder después, con la herramienta Make3D[7, 8, 9], reconstruir dichas imágenes a una representación 3D. Además de los filtros, los algoritmos de segmentación y la reconstrucción 3D, se han usado técnicas de evaluación clásicas como Precision y Recall[11], para calcular la calidad del resultado del proceso.

La representación 3D final no es exacta sino aproximada, por lo que se pueden observar valores atípicos que no concuerdan con la realidad (por ejemplo que el cielo esté más cerca o que no se tengan en cuenta objetos como árboles o farolas). Depende mucho de la resolución de la imagen y la correcta segmentación de la misma. Debido a esto, esta arquitectura pretende, mediante filtros de imagen, ayudar a la segmentación de la misma y flexibilizar su ejecución.

Finalmente se ha realizado, entre los distintos procesos de filtrado y segmentación, una comparativa sobre un conjunto de 25 imágenes para valorar qué combinación de técnicas ofrece un mejor resultado.

Este trabajo se estructura de la siguiente manera: la Sección 2 introduce el estado del arte, la Sección 3 presenta la arquitectura, la Sección 4 se enfoca las pruebas y los resultados y, finalmente, la última sección expone las conclusiones y el trabajo futuro.

1.1. Motivación del proyecto

La motivación del proyecto es la implementación de una herramienta que pruebe y analice distintos algoritmos de segmentación de imágenes, y que, mediante estas segmentaciones, se reconstruya con la herramienta Make3D [7, 8, 9] una representación 3D en formato VRML (Virtual Reality Modeling Language).

Para esto, se parte de la hipótesis de que si se obtiene una mejor segmentación de la imagen se obtendrá una reconstrucción más definida y de mayor calidad. Cabe destacar que no toda

imagen es válida, se han utilizado, por lo general, imágenes a color en formato RGB (Red-Green-Blue) y que tuviesen una profundidad geométrica dentro de la perspectiva cónica. Estas imágenes se han seleccionado de la base de datos de imágenes de Berkley[12].

Para evaluar el correcto funcionamiento del algoritmo de segmentación se han llevado a cabo una serie de pruebas. Los resultados se han comparado con las imágenes segmentadas por personas de la base de datos[12]. Para la evaluación de se han utilizado las métricas clásicas de Precision y Recall[11].

El proceso se ha dividido en:

1. **Un filtrado de la imagen original**: que resalta las partes u objetos fundamentales de ella.
2. **El clustering o segmentación real de ésta**: que nos traduce la imagen a áreas u objetos de interés (de forma ciega).
3. **Un filtrado de la imagen segmentada resultado**: que elimina ciertas áreas que por su tamaño pueden no ser importantes.

1.2. Objetivos del proyecto

Los objetivos del proyecto son:

1. Desarrollar una base de datos de imágenes y segmentaciones (realizadas por humanos).
2. Implementar una serie de filtros de pre-procesado de imagen que remarquen el cambio de zona a zona y que homogenicen los píxeles de una misma zona.
3. Aplicar algoritmos de clustering de imágenes.
4. Implementar una serie de filtros de imagen segmentada que elimine las zonas pequeñas o poco importantes.
5. Implementar una automatización del cálculo de métricas de evaluación: Precision, Recall y F-measure.
6. Integrar la herramienta Make3D para realizar las pruebas.
7. Obtener los resultados y comprobarlos empíricamente con reconstrucciones 3D.

Se ha decido, de cara a cumplir estos objetivos, un enfoque dinámico que no diese prioridad a un objetivo en concreto. Se han llevado a cabo pruebas unitarias y globales enfocados a modificar o eliminar nuevos módulos.

2

Estado del arte

Esta sección introduce de forma general las técnicas aplicadas en el desarrollo del trabajo. Primero se enfoca en los filtros de pre-procesado y post-procesado, después se centra en la segmentación y, por último, introduce los procesos de reconstrucción de imagen a un escenario 3D.

2.1. Filtros de imagen

Con la aparición de la fotografía digital y su popularización, las técnicas de procesamiento de imagen se han visto impulsadas. Estas técnicas han tenido cantidad de usos distintos: mejorar la calidad de la imagen [13], eliminar ruido [14], extraer información [15], comprimir el tamaño de la imagen [15] y reconocimiento de patrones (Biometría)[16], entre otras.

Los filtros de imagen se pueden dividir en tres grandes grupos:

- **Puntuales**: El resultado de un píxel solamente depende de su valor. Los filtros puntuales más comunes son:
    1. **La umbralización**[17]: divide en dos secciones la imagen permitiendo las operaciones binarias o la extracción de información (por ejemplo en textos escaneados). La división es mediante un umbral dado que funciona de la siguiente manera: los píxeles que estén por debajo de ese umbral estarán en una sección y los iguales o superiores en otra. En el caso, por ejemplo, de imágenes de documentos de texto en blanco y negro es posible escoger un umbral automáticamente que más se ajuste. Se puede hacer global (un umbral para toda la imagen) o local (cortando la imagen en secciones y aplicando una umbralización automática en cada sección).
    2. **El recorte**[17]: selecciona regiones en imágenes con correspondencia entre valores y regiones, básicamente consiste en seleccionar uno o varios objetos de la imagen que estén en un intervalo de valores fijo.
    3. **Negativo**[17]: se invierte la intensidad de la imagen.
    4. **Seccionado de niveles**[17]: extrae unos niveles de intensidad para resaltarlos o procesarlos.
- **Locales**: El resultado de un píxel depende de su valor y del valor de los píxeles a su alrededor en un vecindario finito. Usualmente se utiliza una ventana deslizante o máscara

para su implementación. Se distinguen a su vez en cómo manejan los bordes físicos de la imagen, es decir, los píxeles que se encuentran en los límites de la imagen (izquierdo, derecho, superior e inferior):

1. **Filtrado Parcial**[17]: los bordes físicos no se tienen en cuenta, por lo que se obtiene un resultado más pequeño. Si la máscara es de tamaño $N \times N$ el resultado tendrá $\frac{N-1}{2}$ píxeles menos por lado.
2. **Padding**[17]: los valores que faltan para completar la máscara se toman con el valor neutro, comúnmente cero.
3. **Convolución circular**[17]: se toma la imagen como una representación en forma de esfera. Es decir, los píxeles anteriores al límite izquierdo son los del límite derecho, los posteriores al límite derecho son los del límite izquierdo, los superiores al límite superior son los del límite inferior y los inferiores al límite inferior son los del límite superior.

Los filtros locales más comunes son:

1. **El suavizado**[18]: permite eliminar ruido, difuminar bordes o desenfocar.
2. **El filtro Gaussiano**[19]: tipo especial de suavizado. Es un filtro que suaviza la imagen mediante una máscara obtenida de la curva tridimensional de Gauss. Sirve para promediar priorizando el centro de la ventana deslizante antes que sus extremos.
3. **Filtro de mediana**[20, 21]: suavizado mediante la función estadística de la mediana, es decir, se ordenan los valores de los píxeles de la ventana y el resultado es el valor en el medio. Es útil para eliminar ruido.
4. **Realce de contornos**[17]: mediante segundas derivadas como en filtros de Laplace[22, 21], o con desplazamientos de imagen, se resaltan los cambios de intensidad.
5. **Dilatación y erosión**[17]: operadores morfológicos, aquellos basados en la teoría matemática de conjuntos, utilizados frecuentemente en imágenes binarias para eliminar o destacar elementos de la imagen, y corregir errores en bordes.

- **Globales**: El resultado de un píxel depende de la información de toda la imagen. Los filtros globales más comunes son:

1. **El filtrado por frecuencia** [23, 24]: como los filtros paso bajo, paso banda o paso alto, que eliminan un rango de frecuencias de la imagen, para comprimir, eliminar ruido o detectar bordes.
2. **Transformaciones**: como la Transformada de Fourier[25] que se obtiene una representación mediante números complejos, o la Transformada de Hough[26] que identifica figuras geométricas en una imagen.
3. **Extracción de puntos de interés**: obtiene esquinas o puntos de cruce de una imagen. Sirven para comparar imágenes o buscar patrones, como en las huellas dactilares [27].
4. **Extracción de bordes**: extrae los bordes de una imagen como el filtro Laplaciano [22, 21], basado en la derivada. Otros más actuales son los contornos activos[28].
5. **Extracción de regiones**[17]: se suele utilizar la umbralización [29]. Una clase especial es la segmentación que no sólo reconoce las distintas regiones sino que las divide y las etiqueta.

2.2. Clustering

Las técnicas de clustering se han usado sobre todo en Data Mining y en Machine Learning. El objetivo del clustering es dividir un conjunto de datos en grupos (clusters) de forma ciega,°12q según un criterio de similitud, básicamente los elementos de cada grupo son similares entre sí y distintos con elementos de distintos grupos.

Los algoritmos de clustering se dividen generalmente en tres tipos de métodos:

- **Métodos Jerárquicos**: Sin saber a priori cuantos clusters hay, estos métodos producen una secuencia de particiones; o bien, partiendo de un único cluster y dividiendo hasta tener un cluster por instancia, o bien, partiendo de un cluster por instancia y uniendo cluster hasta quedar uno. La unión o división de clusters sigue un criterio especificado en el algoritmo. Algunos ejemplos son:

  1. **Dendrogramas**[30]: una representación gráfica en forma de árbol. Al inicio del algoritmo cada instancia pertenece a un clúster distinto, después el algoritmo itera uniendo los dos clusters que sean los más semejantes.
  2. **Métodos de Enlace**[31]: los métodos de enlace utilizan la proximidad de las instancias para unir en clusters. Se diferencian según el modo:
     a) **Enlace simple (Single Linkage)**[32]: utiliza la distancia mínima entre dos instancias de cada cluster.
     b) **Enlace completo (Complete Linkage)**[33]: utiliza la distancia máxima entre dos instancias de cada cluster.
     c) **Enlace promedio (Average Linkage)**[34]: utiliza la media de las distancias entre todas las instancias de los dos clusters.
     d) **Enlace de centroides (Centroid Linkage)**[35]: utiliza la distancia entre los centroides de cada cluster.
     e) **Método de Ward (Ward Linkage)**[36]: utiliza la suma de las distancias al cuadrado a los centroides de cada cluster.

- **Métodos de Particionado**: Al saber a priori el número de clusters en el conjunto de datos, estos métodos producen una partición de las instancias en ese número de clusters siguiendo un criterio de optimización.

  1. **K-means**[37]: es el algoritmo más común de clustering. Divide en grupos los píxeles según la distancia euclídea a los centroides de cada grupo. Hay variaciones de K-means, por ejemplo, que mejoran la eficiencia filtrando los datos usados[38].
  2. **Expectation-Maximization**[39]: se usa para encontrar estimadores de máxima verosimilitud en modelos probabilísticos.
  3. **Spectral Clustering**[40]: utiliza el espectro de la matriz de similitud para reducir su dimensión y agrupar las instancias con este número menor de dimensiones.

- **Clustering solapado (Overlapping Clustering)**: El clustering solapado tiene la diferencia de que una misma instancia puede pertenecer a uno, ninguno o varios clusters. Uno de los algoritmos de clustering solapado más conocido es una variación del k-means, el Fuzzy C-means Clustering[41]. Existen dos tipos:

  1. **Clustering solapado difuso (Fluzzy Overlapping Clustering)**[42]: en el clustering difuso una instancia tiene un grado (como en la lógica difusa) de pertenencia a un cluster, y puede estar en varios clusters con distintos grados cada uno.
  2. **Clustering solapado absoluto (Soft Overlapping Clustering)**[43]: en este clustering no existen grados, una instancia puede estar en uno, ninguno o varios clusters, pero siempre de forma absoluta.

2.3. Segmentación de imagen

La segmentación de imagen[17] tiene por objetivo dividir la imagen en distintos objetos o zonas representativas, como muestra la Subsección 2.1, la diferencia entre la extracción de regiones y la segmentación es que, no solo se reconocen las zonas, sino que se dividen y se etiquetan. Para la división se puede tener en cuenta la intensidad de imagen, la frecuencia (cambio de intensidades), el reconocimiento de figuras geométricas, la textura... Sirve sobre todo en obtener una representación más significativa y más fácil de analizar.

En la segmentación de imágenes se suelen utilizar técnicas de clustering, tomando los píxeles como elementos y los objetos o zonas como clusters. Además de K-means, se suele utilizar algoritmos de segmentación basados en grafos[44]. Estos traducen la imagen a un grafo en el que cada píxel se toma como un vértice del grafo y las aristas indican el grado de similitud entre los nodos, comúnmente se utiliza la distancia euclídea de la intensidad o de los colores de los píxeles. Después se trata de agrupar aquellos vértices que tienen mayor grado de similitud.

También se utiliza la segmentación mediante el particionamiento de regiones[45, 46]. Esta técnica es, a diferencia de la anterior, sencilla y de bajo coste computacional. Es un proceso iterativo que trata de subdividir en regiones más pequeñas hasta que todas sean homogéneas o que las regiones heterogéneas que quedan estén por debajo de un tamaño mínimo. Para saber si la región es homogénea se utiliza una función de similitud y un umbral.

Otras técnicas de segmentación utilizan filtros de extracción de regiones (como la umbralización[29]) y después aplican un algoritmo de etiquetado que se basa en recorrer cada zona y otorgarle un identificador único.

También existen técnicas de segmentación basada en modelos[47]. Esta técnica se basa en que en la imagen hay una serie de modelos geométricos que se repiten y que, usando un sistema probabilístico, se pueden reconocer. El principal problema de este sistema es, no solo el conocer que modelos son los más habituales, sino las posibles combinaciones de estos modelos en la imagen, teniendo que hacer uso de rotaciones y cambios de tamaño, y obtener una valoración estadística precisa que compare el modelo con el ejemplo seleccionado. En esta técnica se suele utilizar una variación de la Transformada de Hough[26].

Por último, cabe mencionar la Transformada divisoria [48] (en inglés wathershed). Esta técnica se basa en el uso de un filtro de realce de contornos, como un filtro de Laplace[22, 21], y en la interpretación topográfica de su resultado. El filtro deja una imagen resultado con valores altos en los bordes de la imagen (haciendo de montañas) y bajos en el resto con valores bajos (haciendo de valles). Si dejáramos agua en un píxel al azar, este agua caería a un valle, un mínimo local. Los píxeles cuyo agua cayese en el mismo mínimo local se etiquetarían en una misma zona.

2.4. Reconstrucción 3D

Con la popularización de los videojuegos, las tecnologías 3D y los motores gráficos han obtenido una gran repercusión mediática, lo que ha permitido un avance en la reconstrucción 3D automática a partir de imágenes, vídeos o información de varias cámaras para reducir costes o interactuar mejor con el usuario. Además, cada día aumentan los proyectos y trabajos en el campo de la Robótica que utiliza representaciones tridimensionales para poder auto-dirigirse[1]. Un ejemplo es la conducción automática de vehículos[49].

Los algoritmos de reconstrucción 3D han utilizando distintos sistemas. Existen casos con hardware especial, como los sistemas que utilizan sensores láser o radar[50]. Existen los que utilizan dos o más imágenes[6] o una secuencia de vídeo[51].

La mayoría de los trabajos se basan en nuestra visión llamada estereopsis[6, 52, 53, 54], es decir, en la reconstrucción tridimensional de las imágenes de cada ojo que hace nuestro cerebro.

Otros trabajos son específicos para un tipo concreto de imágenes a tratar o el tipo de objeto a reconstruir. Como reconstruir ciudades con imágenes terrestres y aéreas[55] o la reconstrucción de cuerpos humanos [56].

Otros trabajos se basan en aproximar la profundidad de una única imagen. Entre estos existen trabajos que utilizan una configuración muy específica, como la reconstrucción de objetos conocidos como manos o caras de personas [57], la forma de la sombra[58, 59] o la forma de la textura[60, 61, 62]. Estos trabajos por lo general asumen un color y/o textura uniforme y, por lo tanto, funcionan muy mal con imágenes sin restricciones de color o con muchos tipos de textura.

En los sistemas que como entrada tienen una única imagen, pueden utilizar objetos de manera de asistentes, con forma conocida y próximos al objetivo destino[5]. En los sistemas de secuencias de vídeo se pueden reconstruir la forma de superficies no rígidas[63]. Otros sistemas utilizan el análisis frecuencial mediante la transformada de Fourier para estimar la profundidad media de una escena[64].

Otro sistema estima distancias unidimensionales a obstáculos utilizan aprendizaje automático supervisado para la conducción autónoma de un coche pequeño[65].

Otro sistema se basa en reconstruir modelos tridimensionales de interiores que contienen solamente paredes y suelo a partir de una única imagen[66].

Otro sistema se basa en una vista única que asume que las líneas y los puntos de fuga son conocidos en la escena y calcula los ángulos entre líneas paralelas para inferir la estructura 3-D a partir de imágenes de Manhattan[67].

Este trabajo se basa en la herramienta Make3D[7, 8, 9] un sistema que utiliza una única imagen para representar un escenario tridimensional. Este sistema utiliza un aprendizaje supervisado para la estimación de la profundidad, utiliza la técnica jerárquica MRF[68] (Markov Random Field). Esta técnica consiste en un grafo no dirigido que permite hacer inferencias entre distintas variables. El sistema tiene en cuenta características multiescala globales y locales de la imagen y relaciona profundidades entre distintos puntos de la imagen.

3

Arquitectura del sistema

3.1. Diseño y Arquitectura

La arquitectura de la aplicación está divida en módulos, todos ellos conectados en el proceso de reconstrucción. Aunque algunos de estos módulos son opcionales, su utilización ha demostrado mejorar notablemente el proceso de reconstrucción. La figura 3.1 muestra un esquema de la arquitectura completa.

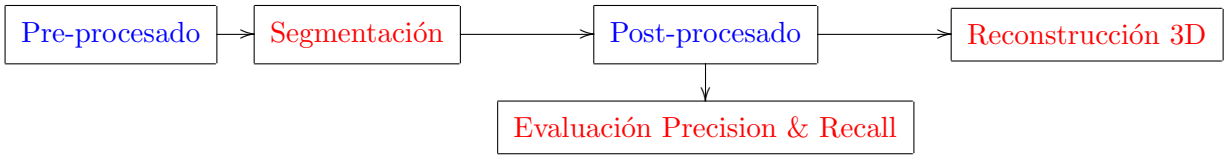


Figura 3.1: Módulos de la aplicación. Se marcan en rojo los módulos que formen el núcleo de la arquitectura y en azul los optativos.

1. **Módulo de Filtros de Pre-procesado**: contiene la implementación de los filtros de pre-proceso de imagen.
2. **Módulo de Segmentación**: contiene los algoritmos de segmentación.
3. **Módulo de Filtros de Post-procesado**: contiene la implementación de los filtros de post-proceso de imagen segmentada.
4. **Módulo de Evaluación**: contiene las funciones del cálculo de las métricas de Precision y Recall.
5. **Módulo de Reconstrucción 3D (Make3D)**: contiene el código de la herramienta Make3D y las funciones para la reconstrucción del escenario 3D.

3.2. Filtros de pre-procesado de la imagen

El objetivo de estos filtros de imagen utilizados es el de mejorar la segmentación de las imágenes, es decir, que las zonas sean más homogéneas consigo mismas y más heterogéneas con respecto a otras zonas. Para ello se ha utilizado una serie de distintos filtros que cumplen este objetivo:

1. **Ecualización del histograma**[17] (Figura 3.2): tiene por objetivo obtener un nuevo histograma de la imagen, a partir del original, con una distribución uniforme de los diferentes niveles de intensidad. Puesto que la imagen es a color, una posible implementación es dividir en canales de colores; rojo, verde y azul, e individualmente ecualizar individualmente sus histogramas.

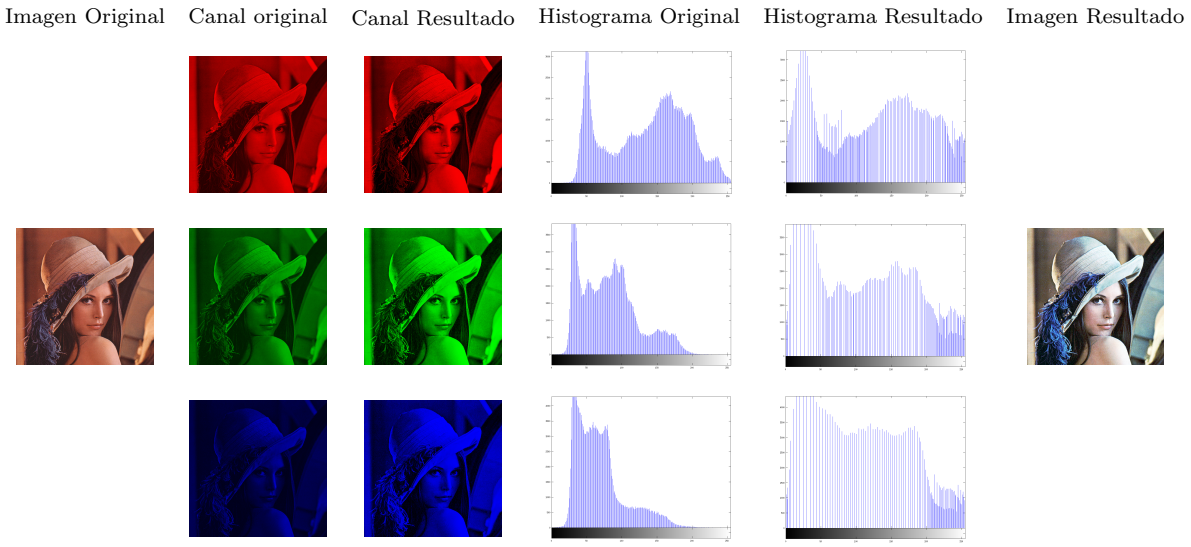


Figura 3.2: Ecualización de histograma de cada canal RGB.

2. **Resaltado de la frontera**[17] (Figura 3.3): se utiliza un algoritmo de detección de bordes [69] de la imagen. Se suma la imagen original con la imagen resultado de multiplicar un número pasado por parámetro y los bordes obtenidos de la imagen. Este número es el que relaciona cuanto se quiere remarcar la frontera, a mayor valor mayor será la diferencia. El objetivo del filtro es realzar los bordes de la imagen para separar las zonas de esta.

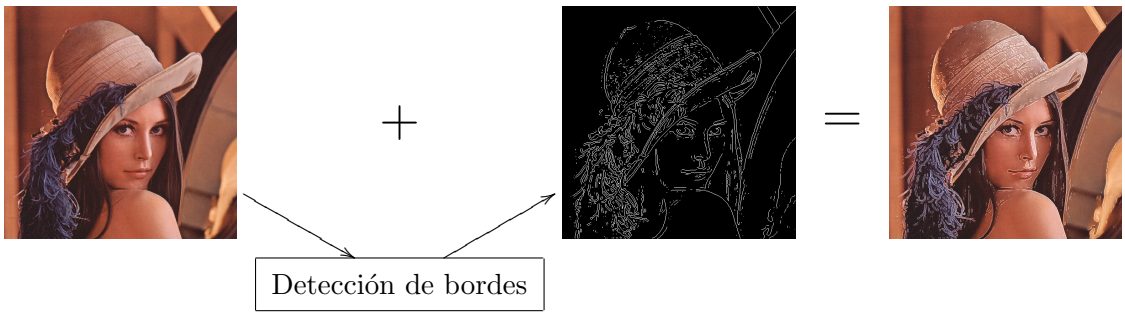


Figura 3.3: Resaltado de bordes de la imagen.

3. **Suavizado y resaltado de la frontera** (Figura 3.4): el proceso es el mismo que el del filtro anterior, con la diferencia de que en vez de sumar los bordes con la imagen original se hace con la imagen resultado de un filtro de suavizado [17] o promediado de imagen. El proceso de este filtro de suavizado es utilizar una ventana deslizante de dimensión N (mejor un número impar) por la imagen y realizar la media de los píxeles que se encuentren en ella. Esta dimensión especifica el nivel de suavizado de la imagen, a mayor valor mayor difusión en el resultado.

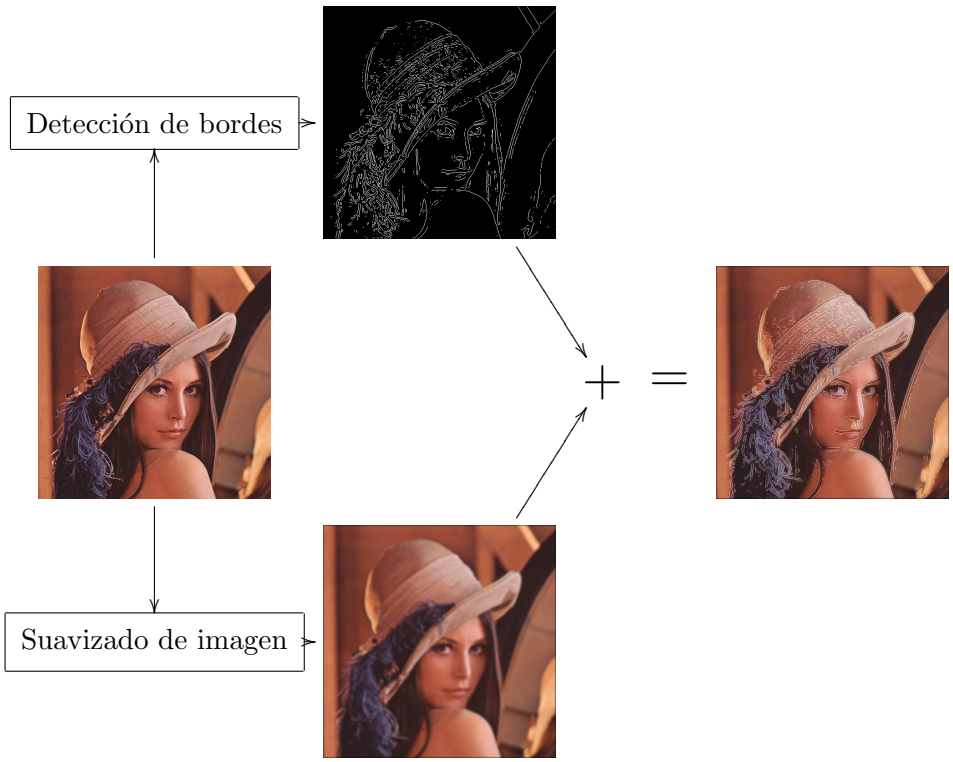


Figura 3.4: Suavizado y resaltado de bordes de la imagen.

4. **Filtro bilateral [70]** (Figura 3.5): este filtro es parecido al Filtro Gaussiano [19]. El Filtro Gaussiano o desenfoque Gaussiano es un filtro que suaviza la imagen perdiendo nitidez, utiliza una curva de Gauss para promediar por lo que se prioriza el centro de la ventana más que sus extremos. La diferencia del Filtro Gaussiano y el bilateral es que el primero tiene exclusivamente en cuenta la distancia euclídea (diferencia del valor de un píxel a otro) y el segundo también toma en cuenta la distancia en la gama (intensidad del color). Es decir, el filtro tiene dos máscaras de promediado, una igual al Filtro Gaussiano y una máscara adicional no lineal que mida las variaciones de intensidad con respecto al píxel central. Se busca, por tanto, suavizar la imagen pero respetando los bordes.

Figura 3.5: Filtrado bilateral de la imagen.

En conclusión, se han utilizados todos estos filtros de forma individual para aumentar la diferencia entre distintas zonas y homogeneizar los píxeles de una misma zona. En la arquitectura se ha creado una estructura que contiene el nombre del filtro y los nombres y valores de sus parámetros.

3.3. Segmentación de la imagen

Los algoritmos de segmentación en imágenes dividen en grupos de píxeles zonas u objetos. En este proyecto, esto sirve para calcular una aproximación a la profundidad geométrica de la imagen, suponiendo una perspectiva cónica en la misma. Se han usado técnicas de clustering para la implementación de los algoritmos de segmentación.

Las algoritmos de segmentación aplicados en este proyecto son:

1. **K-means**[37]: divide en k zonas según la media de la distancia euclídea. El pseudo-código de K-means se muestra en el cuadro del Algoritmo 1.

Algoritmo 1 Pseudo-codigo del algoritmo K-means

Require: Un conjunto de n elementos $X = \{x_1, \ldots, x_n\}$ y un número fijo de clusters k.
Ensure: Un conjunto de clusters $C = \{C_1, \ldots, C_k\}$ que divide X

1: Asigna k instancias siendo los centroides iniciales. Se define el conjunto de centroides como $Y = \{y_1, \ldots, y_k\}$ y $Y' = \emptyset$
2: **while** $Y \neq Y'$ **do**
3: Asigna todos $C_j = \emptyset$.
4: $Y' \leftarrow Y$.
5: **for all** $x_i \in X$ **do**
6: Calcula la distancia mínima del centroide a x_i. Siendo y_j la distancia mínima a x_i.
7: Introduce x_i en C_j.
8: **end for**
9: Calcula los centroides de C y establece $y_i \leftarrow centroide(C_i)$.
10: **end while**
11: **return** C

K-means inicialmente establece los centroides (línea 1) y añade los elementos al clúster cuyo centroide está más cerca de ellos (líneas 5-7). Después, se calcula la nueva posición de los centroides de cada cluster (línea 9) y, de nuevo, añade los elementos más cercanos a cada clusters (líneas 5 a 7). Se continúa hasta que la posición de los centroides converge en un punto fijo (línea 2).

2. **Algoritmo eficiente de segmentación de imágenes basado en grafos**[71]: Se define la medición de una frontera entre dos áreas utilizando una representación basada en grafos. Se traduce la imagen a un grafo en el que cada píxel se toma como un vértice del grafo y cada arista indica el grado de similitud entre dos píxeles. Este método conserva los detalles en las áreas de imagen de baja variabilidad e ignora detalles en las regiones de alta variabilidad. El pseudo-código del algoritmo de segmentación basada en grafos se muestra en el cuadro del Algoritmo 2.

Algoritmo 2 Pseudo-codigo del algoritmo eficiente de segmentación de imágenes basado en grafos

Require: Un grafo $G = (V, E)$ con n vértices y m aristas.
Ensure: La segmentación de V en r número de áreas $S = \{C_1, \ldots, C_r\}$
1: Ordena de forma no decreciente E en $\pi = \{O_1, \ldots, O_m\}$
2: Empezar con la segmentación S^0 donde cada vértice v_i es su propia área
3: **for** $q = \{1, \ldots, m\}$ **do**
4: v_i y v_j = los vértices conectados por una q-arista en el orden $O_q = (v_i, v_j)$
5: $C_i^{(q-1)}$ = el área de $S^{(q-1)}$ que contiene v_i
6: $C_j^{(q-1)}$ = el área que contiene v_j
7: **if** $C_i^{(q-1)} \neq C_j^{(q-1)}$ y $w(O_q) \leq$ diferencia interna mínima de $(C_i^{(q-1)}, C_j^{(q-1)})$ **then**
8: S^q se obtiene de $S^{(q-1)}$ mediante la unión de $C_i^{(q-1)}$ y $C_j^{(q-1)}$
9: **else**
10: $S^q = S^{(q-1)}$
11: **end if**
12: **end for**
13: **return** $S = S^m$

Ahora establecemos que una segmentación S producido por el algoritmo 2 obedece a las propiedades globales de no ser ni demasiado fino ni muy tosco cuando se utiliza la región de comparación del predicado, es decir, de la función de similitud. Aunque el algoritmo hace sólo decisiones codiciosas, produce una segmentación que satisface las propiedades globales. Por otra parte, se muestra que cualquiera de los posibles no decreciente ordenamientos borde de peso que podrían ser recogidos en el paso 0 del algoritmo producen la misma segmentación.

En el proyecto se utilizan siempre imágenes a color ("true color", sin índice o mapa de colores) en formato RGB (Red Green Blue) y, estos algoritmos de segmentación devuelven una matriz de dos dimensiones de números enteros. Cada valor hace referencia a el identificador de su zona. Para tener un formato único se especifica que este identificador sea un número entero del 1 al número de segmentaciones y que ordenados de menor a mayor por tamaño de zona (de mayor a menor). Por lo que la mayor área tendrá valor 1 y la menor el número de segmentaciones. Por tanto la imagen tiene un tamaño, un número de filas y un número de columnas iguales a la imagen original, y un número de segmentaciones.

Los problemas que pueden tener estos algoritmos son dos:

1. **La sobresegmentación[17]**: Que divida en demasiadas zonas o grupos
2. **La infrasegmentación[17]**: Que divida insuficientemente en zonas o grupos

Para obtener una visualización de la imagen segmentada se utiliza un pseudo-color. Se obtiene dividiendo el espectro de colores entre el número de segmentaciones y asignando a cada identificador de zona un valor RGB aleatorio distinto de ese espectro dividido en secciones del mismo tamaño, como en el ejemplo de la Figura 3.6.

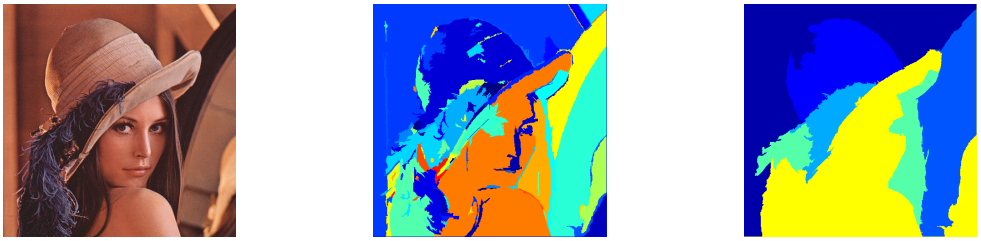
Figura 3.6: Círculo cromático.

3.4. Filtros de post-procesado de la imagen

Estos filtros nos permiten eliminar zonas que podríamos considerar como irrelevantes en una imagen segmentada, ya que son demasiado pequeñas y lo más probable es que pertenezcan a zonas mayores. Eliminar zonas irrelevantes mejora la segmentación, ya que es frecuente que el algoritmo sobresegmente la imagen, es decir, que divida en demasiados grupos de píxeles la imagen.

Estos filtros son iterativos y eliminan zonas de menor a mayor tamaño dependiendo cada uno de una variable:

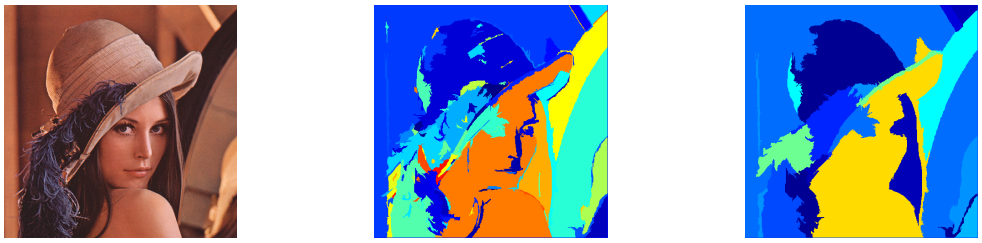
1. **Según un número máximo de áreas[17] (Figura 3.7)** : si el número de zonas excede este máximo se eliminan las zonas sobrantes más pequeñas. Una persona tiende a no segmentar demasiado, solamente lo importante, por lo que existiría un máximo relativo. Como los algoritmos de segmentación no actúan igual, este filtro intenta adecuarlo.



(a) Imagen original. (b) Segmentación. (c) Resultado del filtro.

Figura 3.7: Ejemplo de ejecución del filtro según el número máximo de áreas.

2. **Según un número mínimo de píxeles por área**[17] (Figura 3.8): se eliminan las áreas que no cumplan el tamaño mínimo de píxeles. A diferencia del anterior este se relaciona con el tamaño de la imagen ya que cuanto mayor tamaño sea la imagen mayor será el de sus segmentaciones y con ello evitamos que un algoritmo sobresegmente una image solo porque sea más grande.



(a) Imagen original. (b) Segmentación. (c) Resultado del filtro.

Figura 3.8: Ejemplo de ejecución del filtro según el número mínimo de píxeles por área.

Lo que se hace es unir la zona eliminada con la zona con la que esté más conectada, creando una única zona distinta. Para ello todos los píxeles de la zona a eliminar cuentan qué zonas vecinas y qué número de píxeles están conectados con ellos. Se obtiene la zona que tenga un mayor número de conexiones y se unen las dos zonas.

Cuando se habla de los píxeles alrededor de otro píxel, suponiendo píxeles cuadrados, se puede entender de dos formas:

1. **Una conexión de cuatro**: donde los píxeles alrededor son el próximo superior, inferior, el de la izquierda y el de la derecha.
2. **Una conexión de ocho**: donde los píxeles alrededor, además de los de conexión cuatro, son los próximos en las diagonales, es decir, el superior izquierdo, superior derecho, inferior izquierdo e inferior derecho.

Es posible utilizar cualquiera de ellas, pero la segunda es más fácil de implementar.

3.5. Evaluación mediante Precision y Recall

La evaluación de la segmentación consiste en obtener los valores de Precision y Recall comparando las imágenes segmentadas por la herramienta con las segmentaciones de las mismas imágenes originales hechas por personas. Las imágenes, tanto las originales como las segmentaciones hechas por personas, se recogen de la base de datos de de Berkley[12].

Para poder entender las métricas de Precision y Recall, es necesario definir los siguientes conceptos:

- **True-Positive** (tp): La instancia ha sido correctamente clasificada como parte de la clase.
- **False-Positive** (fp): La instancia ha sido incorrectamente clasificada como parte de la clase.
- **True-Negative** (tn): La instancia ha sido correctamente no clasificada como parte de la clase.
- **False-Negative** (fp): La instancia ha sido incorrectamente no clasificada como parte de la clase.

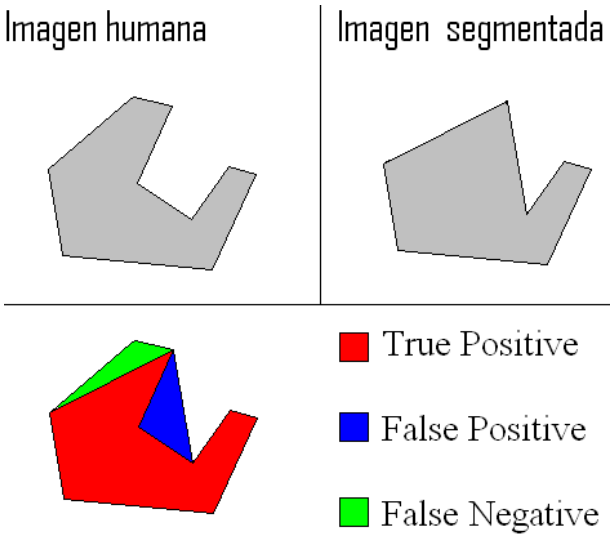


Figura 3.9: Aplicación de la metodología a la segmentación de imágenes a través de áreas, en este caso, las imágenes de arriba son: la segmentada por un humano (izquierda) y el resultado de la segmentación (derecha). La imagen de abajo muestra las áreas que corresponden con los verdaderos positivos (rojo), falsos positivos (azul) y falsos negativos (verde).

Las métricas de Precision, Recall y F-Measure se definen como:

$$Precision = \frac{tp}{tp + fp} \tag{3.1}$$

$$Recall = \frac{tp}{tp + fn} \tag{3.2}$$

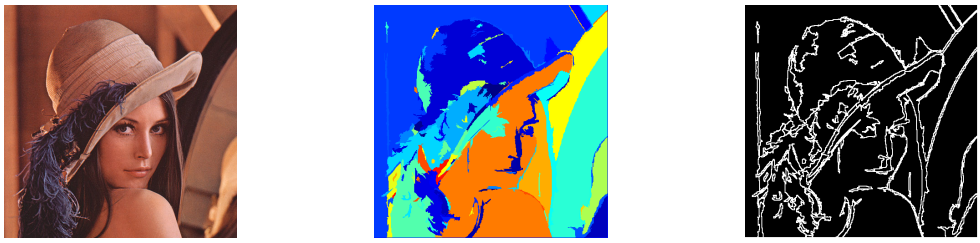
$$F - Measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.3}$$

Precision, valor que indica cuántas de las instancias son correctas (positivas) sobre el ratio de las recuperadas. **Recall**, mide la situación en la que una instancia se encuentra debidamente tipificado de acuerdo a su clase. **F-Measure**, es una métrica que proporciona un valor medio a partir de las anteriores.

Estas métricas se pueden obtener de dos formas:

1. Las métricas se pueden obtener de cada área independientemente y sacar la media entre todas (ver figura 3.9).

2. Las métricas se pueden obtener utilizando una transformación de la imagen segmentada a un formato frontera, que básicamente las líneas divisoras de las zonas toman el valor verdadero y el resto de píxeles el cero. Para obtenerla basta un filtro local que utilice una ventana deslizante de tamaño 3x3, si todos los valores son iguales toma el valor de falso y si hay alguno distinto toma el valor de verdadero. Después, se compara las fronteras de la imagen segmentada por la aplicación y las fronteras segmentadas por humanos. Para ello se utiliza la anterior opción pero con solamente dos clases, frontera y no frontera. En la Figura 3.10 se expone un ejemplo de transformación a formato frontera. En este proyecto se ha utilizado la esta opción ya que es la más óptima, porque se tiene en cuenta un menor conjunto de datos.



(a) Imagen original. (b) Segmentación. (c) Segmentación en frontera.

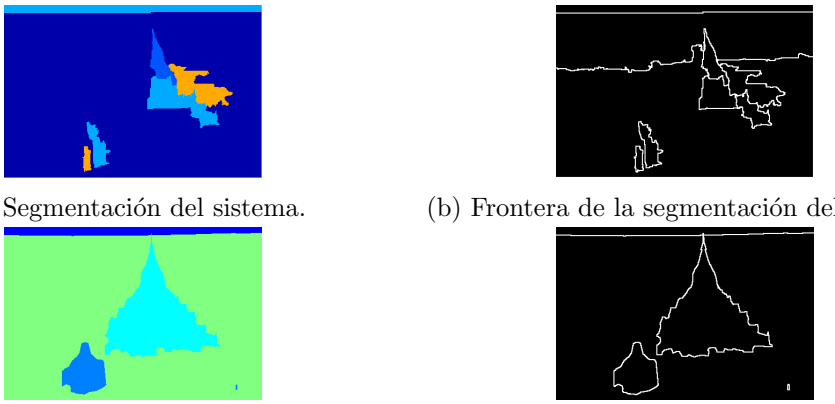
Figura 3.10: Ejemplo de transformación de una segmentación a formato frontera.

Para una mejor explicación se presenta un ejemplo de ejecución de una imagen con una segmentación humana. En primer lugar, se calcula todo el proceso del sistema, el filtro de preprocesado, la segmentación de la imagen y el filtro de postprocesado, tal y como presentamos en la Figura 3.11.

(a) Imagen original. (b) Preprocesado. (c) Segmentación. (d) Postprocesado.

Figura 3.11: Ejemplo de procesado del sistema: preprocesado, segmentación y postprocesado.

Después se pasan las segmentaciones, tanto el resultado del postprocesado como la segmentación humana cargada, a su transformada en formato frontera. Estas transformaciones se exponen en la Figura 3.12

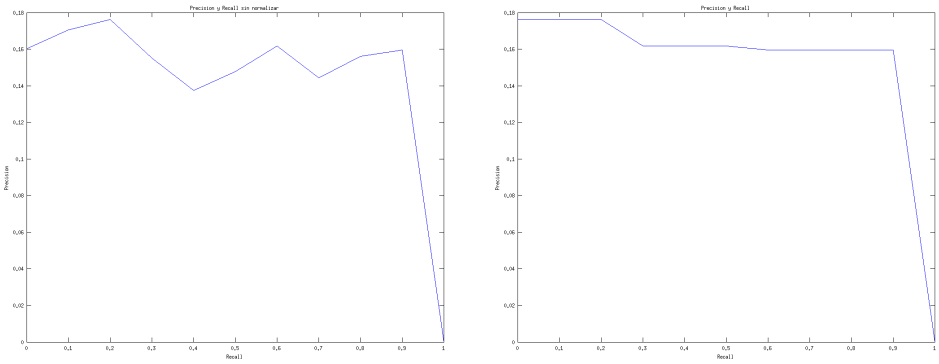


(a) Segmentación del sistema. (b) Frontera de la segmentación del sistema.

(c) Segmentación humana. (d) Frontera de la segmentación humana.

Figura 3.12: Transformaciones de las segmentaciones a su frontera.

Por último, se calculan las métricas de Precision y Recall. Para ello se utiliza un proceso acumulativo que recoge un porcentaje (p) de píxeles aleatoriamente (comúnmente un 10 %), se calcula su Precision con la Ecuación 3.1. El Recall es el porcentaje acumulado en ese instante de tiempo. Después se normaliza esta métrica para que Precision sea, o bien constante, o bien decreciente con un Recall decreciente de 100 % a 0 %. El proceso de normalización se presenta en la Figura 3.16.



(a) Precision y Recall sin normalizar. (b) Precision y Recall normalizadas.

Figura 3.13: Ejemplo de la normalización de las métricas Precision y Recall. El eje X representa la métrica Recall y el eje Y la Precision.

Con este resultado podemos calcular la F-measure con la Ecuación 3.3. El valor de F-measure del resultado anterior se presenta en la Figura 3.14.

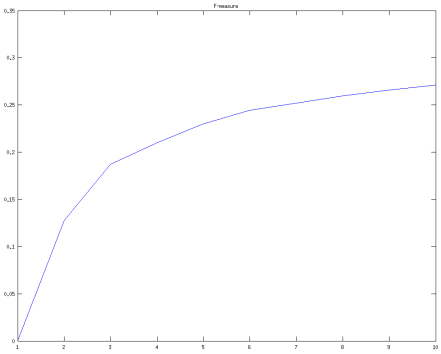


Figura 3.14: Ejemplo del cálculo de la métrica F-measure.

Debido a que los algoritmos de segmentación no son deterministas, es decir, para una misma entrada puede haber distintas salidas, se ha ejecutado un número determinado de veces la misma combinación (en este caso 10) y se ha promediado la métrica de Precision. También se tiene que promediar según el número de segmentaciones humanas de cada imagen contenidas en la base de datos. Todo esto para una sola imagen, para tener un resultado global también se promedia las métricas de Precision de todas las imágenes de utilizadas, que en este proyecto han sido 25.

3.6. Reconstrucción 3D

En esta arquitectura, para hacer la reconstrucción del escenario tridimensional se utiliza la herramienta Make3D. Esta herramienta utiliza una estimación de profundidad de una sola imagen fija mediante un enfoque de aprendizaje supervisado. Este aprendizaje se basa, entre otras cosas, en la segmentación. El algoritmo de segmentación que aplica la herramienta es el algoritmo basado en grafos [71].

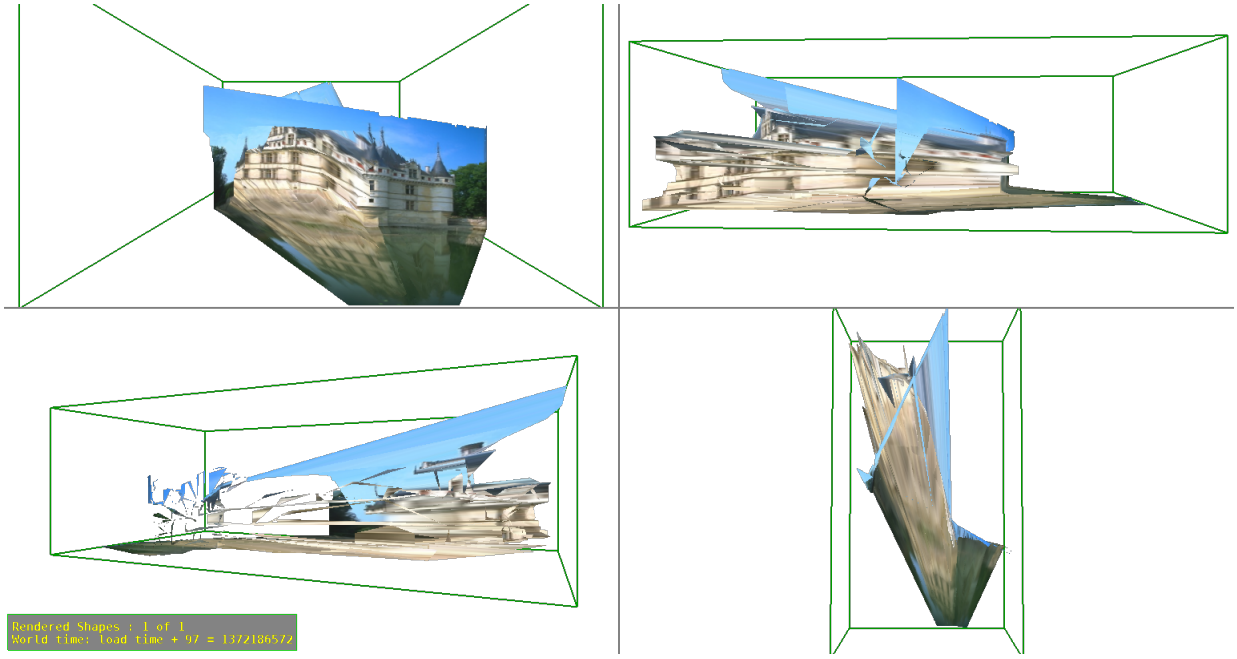
El conjunto de datos que usa la herramienta para el aprendizaje es una serie de imágenes de ambientes no estructurados (interiores, exteriores, aceras, bosques...) y los mapas de profundidad, resultado ideal de lo que la herramienta intenta predecir. La herramienta tiene en cuenta el contexto global de la imagen y utiliza una estructura jerárquica MRF (Markov Random Field), un grafo no dirigido que permite hacer inferencias entre distintas variables. Esta estructura añade características de imagen multiescala ya sea local o global, y relación entre profundidades entre distintos puntos de la imagen.

Una de las características que la herramienta tiene en cuenta para predecir la profundidad de la imagen es la segmentación. La hipótesis de este proyecto es que si se mejora la segmentación de la imagen con otro algoritmo se observaría una mejora en la calidad del resultado de la proyección 3D.

Para aclarar mejor el comportamiento de esta herramienta se exponen dos ejemplos. El primer ejemplo en el que se observa una profundidad y el segundo ejemplo que no tiene profundidad ya que se trata de un primer plano.

El primer ejemplo de imagen usada como entrada es la vista de un edificio. La fotografía utilizada se encuentra en la Sub-figura 4.3 y el resultado de su reconstrucción 3D en la Sub-figura 4.4.

(a) Imagen original.



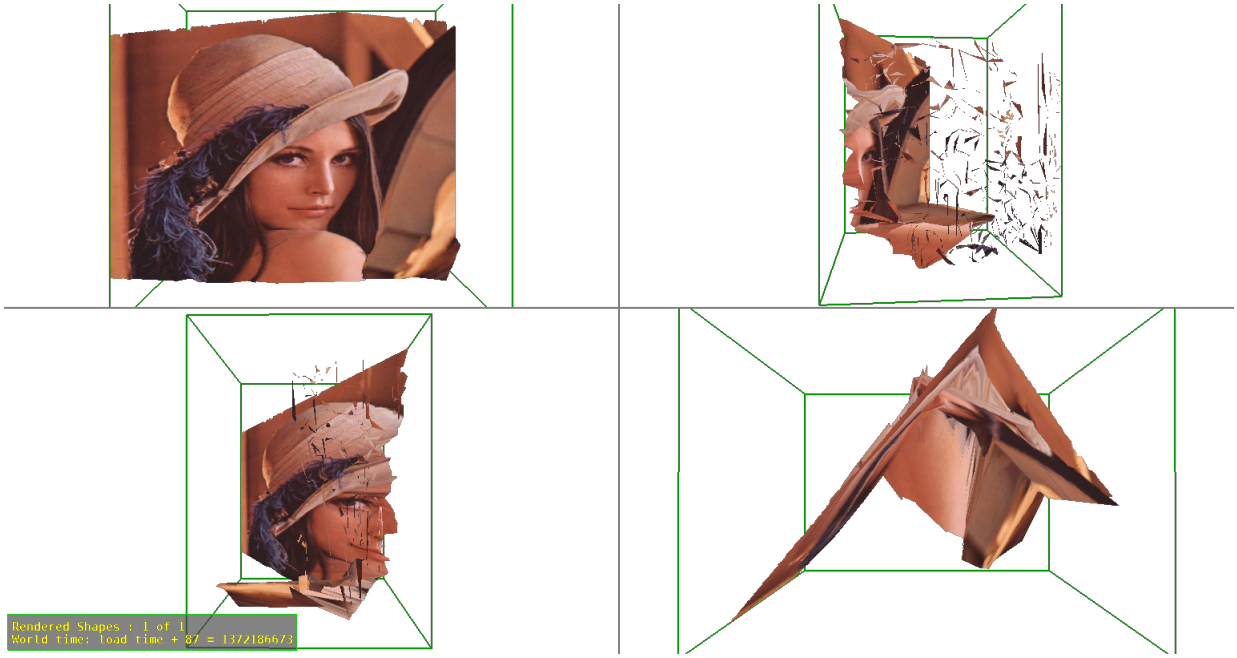
(b) Reconstrucción 3D.

Figura 3.15: Ejemplo de reconstrucción 3D normal. En la reconstrucción 3D se observa, de izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

Se observa en este ejemplo que la herramienta busca utilizar la esquina vertical y central del edificio dejándola en la posición más próxima del edificio en el escenario 3D. También se observa que intenta dejar al fondo el lado izquierdo del edificio y en menor medida el derecho, pero es evidente que utiliza demasiada profundidad. Un aspecto correcto es que trata el agua como un suelo perfecto sin que el reflejo del edificio lo desvirtúe. Aunque el cielo no lo reconoce como debería y hace que forme parte del edificio.

El segundo ejemplo de imagen es el primer plano utilizado para explicar el funcionamiento de los filtros y de la segmentación, Sub-Secciones 3.2, 3.3 y 3.4. La fotografía utilizada se encuentra en la Sub-figura 4.6 y el resultado de su reconstrucción 3D en la Sub-figura 4.7.

(a) Imagen original.



(b) Reconstrucción 3D.

Figura 3.16: Ejemplo de reconstrucción 3D errónea. En la reconstrucción 3D se observa, de izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

Se observa que la herramienta es incapaz de resolver una imagen en primer plano ya que divide la cara en dos debido a que reconoce la esquina que se encuentra al fondo de la imagen. Además busca un suelo que no existe en este caso y por ello de hombro hacia abajo lo toma erróneamente como suelo y el resto no. Por esto es necesario utilizar una imagen con profundidad ya que la herramienta busca líneas y puntos de fuga para la reconstrucción.

4

Pruebas Realizadas y Resultados

4.1. Parámetros experimentales

Para las pruebas realizadas se han escogido 25 imágenes de la base de datos de Berkley[12] que cumplían los criterios de color RGB y cierta profundidad geométrica suponiendo perspectiva cónica. Para la evaluación de las métricas Precision y Recall se ha usado un intervalo común del 10 %.

Debido a que los algoritmos usados de segmentación no son deterministas, es decir, que para una misma entrada pueden devolver distinta salida, es necesario realizar un promedio de un determinado número de ejecuciones. En estas pruebas hemos escogido 10 ejecuciones. Los parámetros de los algoritmos de segmentación se muestran en la Tabla 4.1.

Algoritmo	**Nombre**	**Valor**
K-means	número de clusters	20
Segmentación basada en grafos	número de klusters	20
”	sigma	0.5
”	tamaño mínimo	500

Tabla 4.1: Argumentos de algoritmos de segmentación.

Haciendo un estudio estadístico del número de segmentaciones humanas por imagen de la base de datos de Berkley[12] obtenemos el histograma de la Figura 4.1.

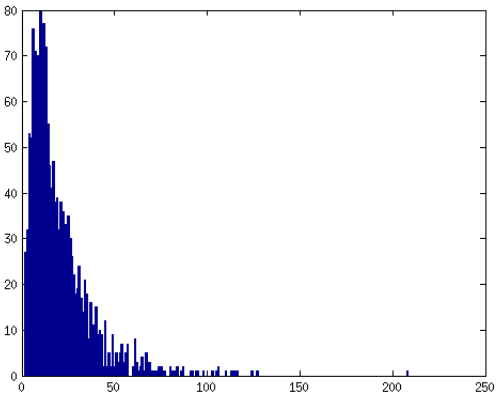


Figura 4.1: Histograma del número de áreas distintas por segmentación. El eje X representa el número de áreas distintas por segmentación y el eje Y el número de segmentaciones realizadas por personas.

Se obtiene que entre el 1 y el 50 está el 93.9375 % del conjunto de segmentaciones, es decir, que la gran mayoría de las personas solamente necesita entre 1 y 50 áreas distintas para segmentar una imagen. Se obtiene que la media es 20.3521 áreas distintas por segmentación. Por ello se ha escogido el número 20 como número de clusters para los algoritmos de segmentación (Tabla 4.1).

A diferencia del filtro de post-procesado, el tamaño mínimo del algoritmo de segmentación basada en grafos solo une dos áreas si éstas son menores al mínimo y están conectadas, no si una es menor al mínimo y la otra es con la que está más conectada. El parámetro sigma=0.5 sirve para difuminar la imagen, se ha escogido el valor dado como ejemplo dado en la documentación del código.

Los parámetros de filtros de pre-proceso se exponen en la Tabla 4.2.

Filtro	**Nombre**	**Valor**
Ecualización de histograma	nivel	256
Resaltado de frontera	método	'LoG'
”	multiplicador	100
Suavizado y resaltado de frontera	dimensión	7
”	método	'LoG'
”	multiplicador	100
Filtro bilateral	sigma	[3, 0.1]
”	w	5

Tabla 4.2: Argumentos de algoritmos de segmentación.

De la Tabla 4.2 cabe destacar el nivel de la Ecualización de histograma, que hace referencia al número de niveles discretos en los que se divide el histograma. Los filtros de resaltado de frontera, y suavizado y resaltado de frontera, comparten dos argumentos, el método de detección de bordes, que no cambia demasiado el resultado, y el número multiplicador que multiplica la frontera para resaltarla en la imagen. Además el suavizado y resaltado de frontera tiene la

dimensión de la ventana deslizante que promedia la imagen. Los argumentos del filtro bilateral hacen referencia a: sigma, donde la desviación espacial se da por sigma[1] (cuyo valor es 3) y la desviación de la intensidad se da por sigma[2] (cuyo valor es 0.1), y w, que es el tamaño medio de la ventana de filtro bilateral Gaussiano.

Los argumentos de los filtros de post-proceso se muestran en la Tabla 4.3.

Filtro	**Nombre**	**Valor**
Mínimo tamaño de área	Mínimo número de píxeles por zona	1000
Máximo número de áreas	Máximo número de áreas	10

Tabla 4.3: Argumentos de algoritmos de segmentación.

De la tabla 4.3 tanto el mínimo número de píxeles por área como el máximo número de áreas se ha escogido probando manualmente varios casos al azar en el conjunto de 25 imágenes utilizadas como base de datos.

4.2. Pruebas del sistema completo

En esta sección se exponen las pruebas de la herramienta. Se demostrará que es posible mejorar la calidad de la segmentación y su posterior reconstrucción 3D, mediante filtros de imagen. Para demostrar la mejora de la segmentación se comparará utilizando la media de la métrica F-measure (Sección 3.5). Para demostrar la mejora de la calidad de la reconstrucción 3D se comparará visualmente un ejemplo entre dos representaciones: con y sin filtros.

En las siguientes tablas de esta sub-sección se presentan los resultados de los filtros y algoritmos anteriormente descritos (Sección 4.1). En primer lugar se presentarán los resultados del sistema sin filtros; únicamente se usan los algoritmos de segmentación, en segundo lugar se presentan los resultados obtenidos con cada filtro de pre-proceso, en tercer lugar se presentan los resultados obtenidos con cada filtro de post-proceso, y en cuarto y último lugar se presentan los resultados obtenidos de todas las combinaciones entre filtros de preproceso y post-proceso. Se destacará en negrita aquellos valores que destaquen significativamente en comparación al resultado obtenido sin filtros.

En primer lugar se ha obtenido el valor de F-measure para cada segmentación sin filtros. Como era esperable, se observa una gran diferencia entre la segmentación basada en grafos y K-means. Esto se debe a que K-means solamente tiene en cuenta la intensidad de los píxeles y la segmentación basada en grafos la intensidad y la frecuencia, es decir, los cambios de intensidad entre píxeles.

Pre-Proceso	**Segmentación**	**Post-proceso**	**F-measure**
-	K-means	-	0.088393
-	segmentación basada en grafos	-	0.160648

Tabla 4.4: Resultados del sistema sin filtros.

A continuación, se muestran los resultados utilizando un solo filtro. Primero con los filtros de pre-procesado, mostrados en la Tabla 4.5. Se observa que el filtro que mejor funciona es el filtro bilateral ya que tanto en K-means como en la segmentación basada en grafos, aumenta notablemente el valor de la métrica (ver Tabla 4.4). Esto se debe a que es el filtro que más respeta el objetivo de homogeneizar las zonas comunes respetando los bordes.

Pre-Proceso	**Segmentación**	**Post-proceso**	**F-measure**
Ecualización de Histograma	K-means	-	0.079933
Resaltado de la frontera	K-means	-	0.088677
Suavizado y resaltado de la frontera	**K-means**	-	**0.105355**
Filtro bilateral	**K-means**	-	**0.116337**
Ecualización de Histograma	Segmentación basada en grafos	-	0.152609
Resaltado de la frontera	Segmentación basada en grafos	-	0.162786
Suavizado y resaltado de la frontera	Segmentación basada en grafos	-	0.155080
Filtro bilateral	**Segmentación basada en grafos**	-	**0.176872**

Tabla 4.5: Resultados de la arquitectura con filtros de pre-procesado.

Y después, con los filtros de post-procesado, mostrados en la Tabla 4.6. Se observa que el filtro que mejor funciona es el filtro basado en el máximo número de áreas. Aunque en el caso de K-means no consigue un resultado notable, si lo hace en la segmentación basada en grafos (ver Tabla 4.4). Aunque el sistema entre los dos filtros puede ser equivalente, el filtro basado en el mínimo tamaño de área depende mucho de la imagen a procesar mientras que el filtro basado en el máximo número de áreas tiene un componente más humano ya que tenemos una predisposición a no segmentar demasiado tal y como se expone en la Sección 4.1.

Pre-Proceso	**Segmentación**	**Post-proceso**	**F-measure**
-	**K-means**	**Máximo número de áreas**	**0.093333**
-	K-means	Mínimo tamaño de área	0.088520
-	**Segmentación basada en grafos**	**Máximo número de áreas**	**0.199727**
-	Segmentación basada en grafos	Mínimo tamaño de área	0.166927

Tabla 4.6: Resultados del sistema con filtros de post-procesado.

Finalmente, se usan ambos filtros: pre-proceso y post-proceso mostrados en la Tabla 4.7. El mejor resultado se ha obtenido con la combinación entre los dos mejores filtros procesados individualmente, el filtro bilateral y el filtro basado en el máximo número de áreas. También se observa que esta combinación es mejor que los filtros procesados individualmente (ver Tablas 4.4, 4.5 y 4.6).

Pre-Proceso	**Segmentación**	**Post-proceso**	**F-measure**
Ecualización de Histograma	K-means	Máximo número de áreas	0.090493
Ecualización de Histograma	K-means	Mínimo tamaño de área	0.079953
Resaltado de la frontera	K-means	Máximo número de áreas	0.092992
Resaltado de la frontera	K-means	Mínimo tamaño de área	0.088699
Suavizado y resaltado de la frontera	K-means	Máximo número de áreas	0.111169
Suavizado y resaltado de la frontera	K-means	Mínimo tamaño de área	0.105504
Filtro bilateral	**K-means**	**Máximo número de áreas**	**0.122902**
Filtro bilateral	K-means	Mínimo tamaño de área	0.116545
Ecualización de Histograma	**Segmentación basada en grafos**	**Máximo número de áreas**	**0.192666**
Ecualización de Histograma	Segmentación basada en grafos	Mínimo tamaño de área	0.156313
Resaltado de la frontera	Segmentación basada en grafos	Máximo número de áreas	0.191044
Resaltado de la frontera	Segmentación basada en grafos	Mínimo tamaño de área	0.166996
Suavizado y resaltado de la frontera	Segmentación basada en grafos	Máximo número de áreas	0.186427
Suavizado y resaltado de la frontera	Segmentación basada en grafos	Mínimo tamaño de área	0.161592
Filtro bilateral	**Segmentación basada en grafos**	**Máximo número de áreas**	**0.221288**
Filtro bilateral	Segmentación basada en grafos	Mínimo tamaño de área	0.186753

Tabla 4.7: Resultados de la sistema con filtros de pre-procesado y post-procesado.

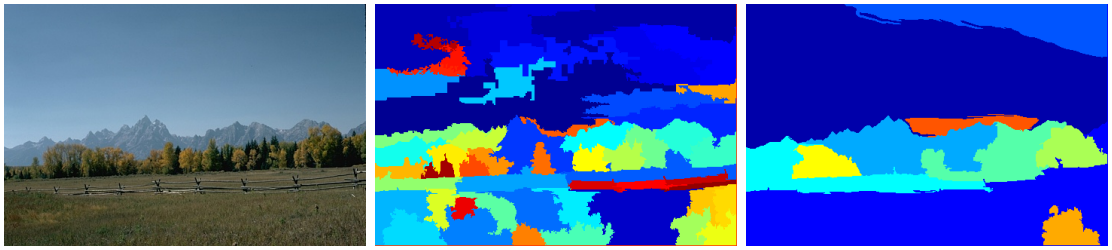
En conclusión, obtenemos que el mejor algoritmo de segmentación de imagen usado en este proyecto ha sido el algoritmo basado en grafos ya que gana significativamente al mejor K-means incluso sin filtros. Como se deduce el algoritmo de segmentación utilizado es fundamental para el resultado de la métrica, pero también se observa que aumentan significativamente ambos resultados de los algoritmos de segmentación con los filtros. En ambos algoritmos de segmentación, su mejor resultado es combinando un filtro de preproceso, el filtro bilateral, y un filtro de post-procesado, el filtro basado en el máximo número de áreas. Esto se debe a que los objetivos de ambos filtros no chocan entre si, y ha que son los filtros que mejor han funcionado individualmente.

4.3. Resultados de la Reconstrucción 3D

En esta sub-sección se expondrán dos ejemplos de reconstrucción 3D, uno utilizando el algoritmo de segmentación basado en grafos y otro utilizando el algoritmo de K-means. En ambos ejemplos se mostrará la segmentación con y sin filtros y su posterior reconstrucción 3D. Los filtros usados son los que mejores resultados obtenidos en la sub-sección anterior (Sub-sección 4.2).

La herramienta Make3D busca diferenciar suelos y paredes para generar el escenario tridimensional. Cabe destacar que en exteriores la reconstrucción más usual es básicamente una forma de "L" en los lados, es decir, una representación con un suelo y una única pared frontal alejada. También mencionar que el cielo no se toma como una profundidad infinita sino, idílicamente, como la máxima del escenario 3D en cuestión.

En primer lugar se muestra el ejemplo con la segmentación basada en grafos. La imagen utilizada para la representación es la siguiente, Figura 4.2a, y sus segmentaciones sin filtros, 4.2b, y con filtros, 4.2c.



(a) Imagen original. (b) Segmentación sin filtros. (c) Segmentación con filtros.

Figura 4.2: Imagen original y segmentaciones con y sin filtros mediante segmentación basada en grafos.

Se puede observar que la segmentación sin filtros, en la Figura 4.2b, tiene demasiadas áreas diferentes, más específicamente se sobresegmenta la hierba y el cielo. En la segmentación con filtros, en la Figura 4.2c, se observa una mejora en estas sobresegmentaciones aunque sigue sin llegar a ser perfecta.

Primero se expone la reconstrucción sin filtros en la Figura 4.3. Se puede observar que no existe un cambio claro en la forma de "L" (pared frontal y suelo) por lo que la reconstrucción no añade mucho valor. Es decir, no se muestra el cambio de profundidades entre los árboles y las montañas.

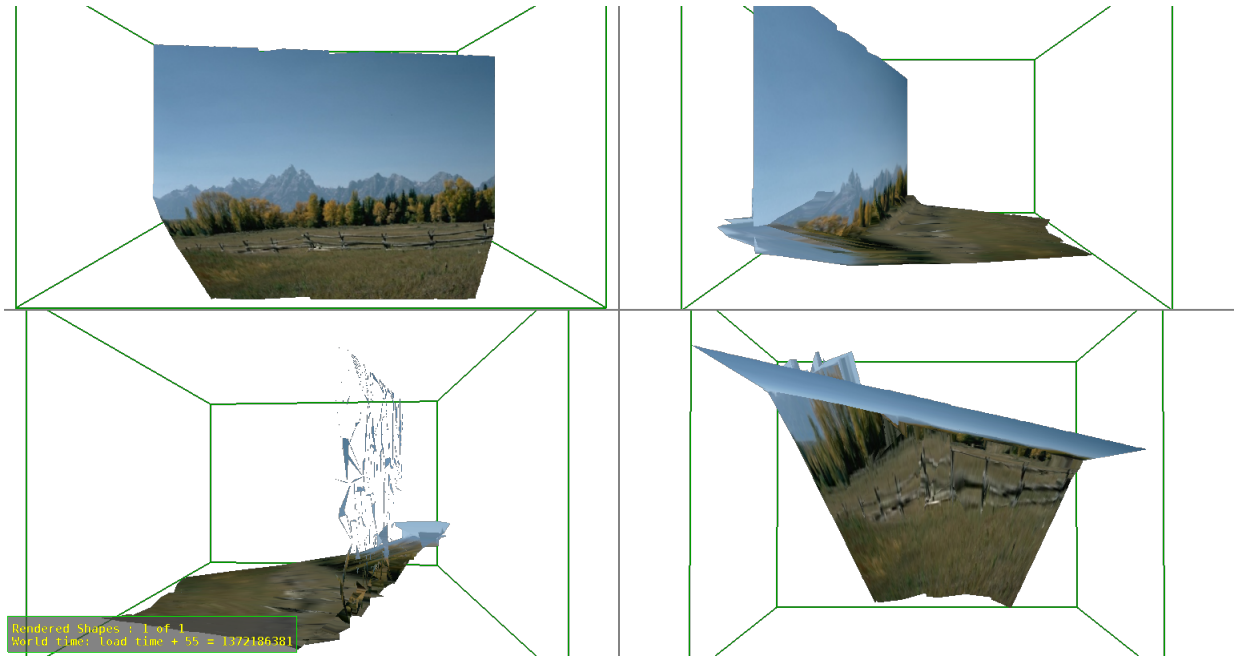


Figura 4.3: Ejemplo de reconstrucción 3D sin filtros mediante segmentación basada en grafos. De izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

Como puede verse, en la reconstrucción con filtros en la Figura 4.4, se produce un cambio claro ya que se tiene en cuenta en mayor medida las profundidades de las montañas y los árboles pero también se observan mayor número de errores debido a que hay puntos en el escenario demasiado alejados.

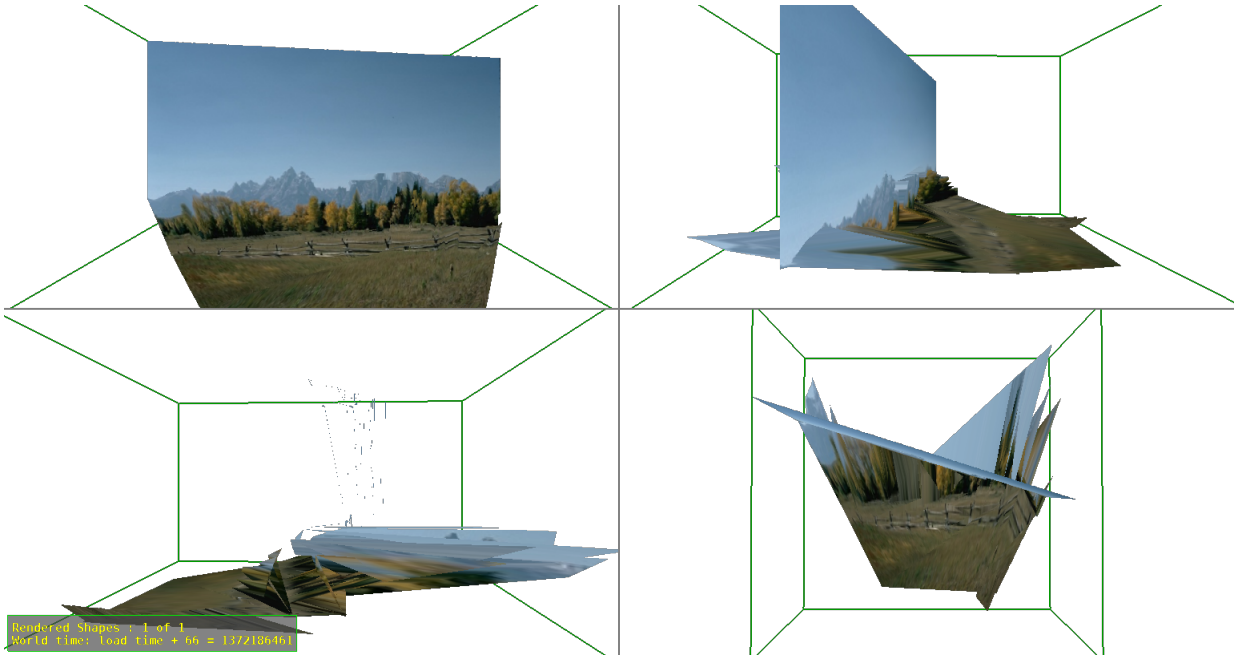
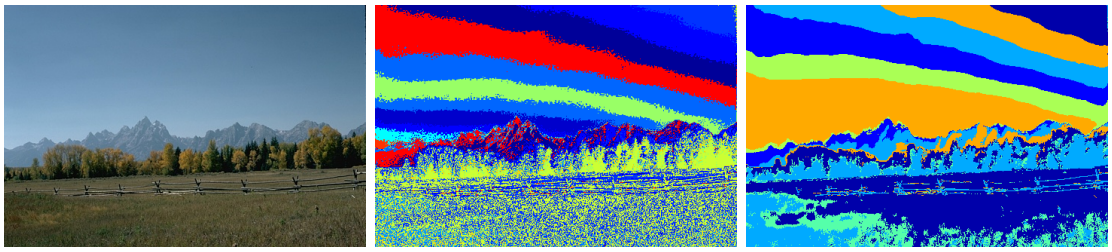


Figura 4.4: Ejemplo de reconstrucción 3D con filtros mediante segmentación basada en grafos. De izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

La mayoría de las reconstrucciones probadas dan este tipo de error. Como se ha expresado anteriormente, en la Subsección 3.6, la herramienta Make3D tiene un componente de aprendizaje automático por lo que es posible que al cambiar la segmentación la herramienta no interprete como debería. Para solucionarlo se debería volver al realizar el aprendizaje automático de la

herramienta y, posiblemente, sea necesario cambiar las salidas esperadas de la herramienta ya que se trata de aprendizaje supervisado.

A continuación, se expone el ejemplo utilizando el algoritmo de segmentación K-means. La imagen utilizada para la representación es la misma que la anterior, Figura 4.5a, y sus segmentaciones sin filtros, 4.5b, y con filtros, 4.5c.



(a) Imagen original. (b) Segmentación sin filtros (c) Segmentación con filtros

Figura 4.5: Imagen original y segmentaciones con y sin filtros mediante kmeans.

Podemos observar que a diferencia del caso del algoritmo de segmentación los píxeles de una misma zona no tienen porque estar unidos. Esto se debe a que K-means solamente utiliza la intensidad de los píxeles. Se observa que en la segmentación sin filtros (Figure 4.5b) existen en el suelo y el las laderas de las montañas una nube de puntos de distintas áreas. En la segmentación con filtros (Figura 4.5c) se observa una mejora.

En primer caso, se muestra la reconstrucción sin filtros en la Figura 4.6. Se puede observar una cantidad considerable de errores mayores que en el anterior ejemplo.

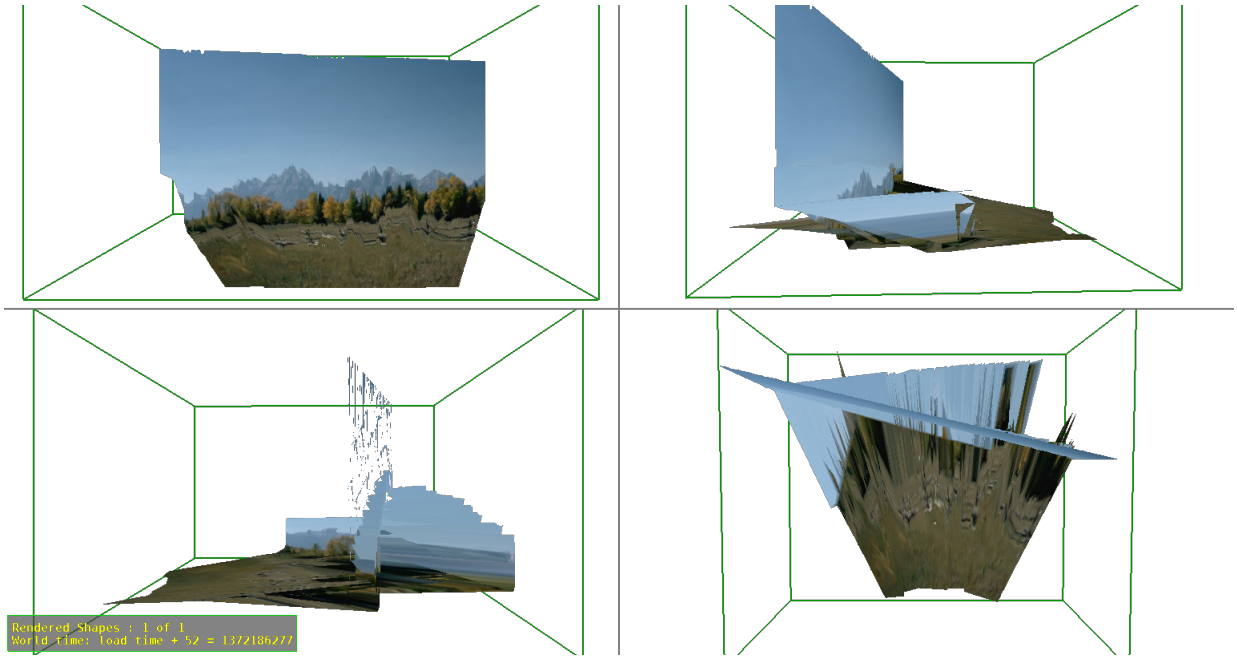


Figura 4.6: Ejemplo de reconstrucción 3D sin filtros mediante kmeans. De izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

En segundo caso, en la reconstrucción con filtros en la Figura 4.7, se observa que parece aumentar los casos de error en la aproximación de la profundidad. Esto puede deberse a que al tener una segmentación más definida visualmente se observa los errores más claramente. Como ocurre en el ejemplo anterior no se puede saber si realizando el proceso de aprendizaje de la herramienta con este algoritmo de segmentación de imagen se obtendrían mejores resultados.

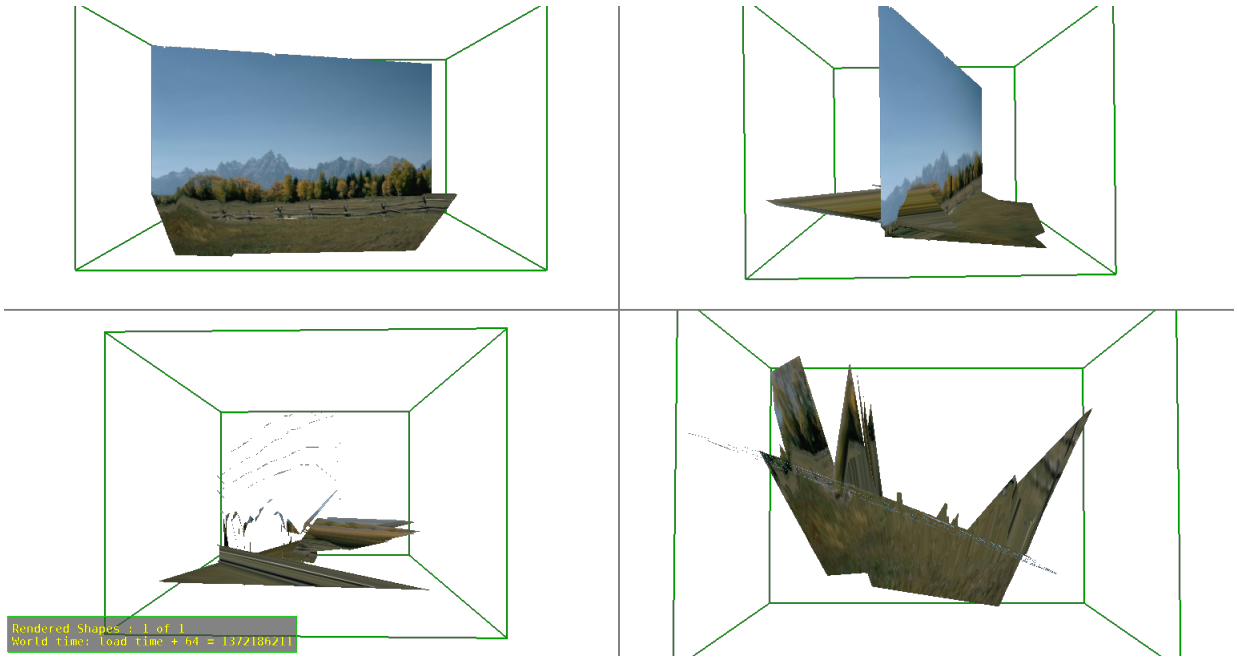


Figura 4.7: Ejemplo de reconstrucción 3D con filtros mediante kmeans. De izquierda a derecha y de arriba a abajo, el plano de frontal, izquierdo, derecho y superior.

En conclusión, se observa una clara diferencia cualitativa en el resultado de la reconstrucción 3D entre los algoritmos de segmentación de imagen basada en grafos y K-means. También se concluye que para obtener unos resultados más sólidos se necesita volver a realizar el proceso de aprendizaje de la herramienta Make3D.

5

Conclusiones y trabajo futuro

5.1. Conclusiones

Se ha diseñado un sistema que permite flexibilizar el uso de filtros de imagen y de algoritmos de segmentación para obtener un resultado de mejor calidad a la hora de realizar una reconstrucción 3D, que únicamente con la segmentación. El sistema es más flexible porque permite elegir, modificar y combinar entre distintos filtros y algoritmos de segmentación sin afectar al funcionamiento principal. Además se han integrado métricas de evaluación que permiten analizar la calidad de los resultados obtenidos.

Los algoritmos de segmentación de imagen usados en este proyecto han sido: K-means y segmentación basada en grafos. Según los resultados obtenidos de la métrica F-measure, el algoritmo de segmentación basado en grafos aproxima mucho mejor que K-means a una segmentación hecha por una persona.

Los filtros se dividen en preproceso y post-proceso, los primeros buscan facilitar el objetivo del algoritmo de segmentación y los segundos eliminar zonas consideradas irrelevantes debido ha su tamaño.

Los filtros de preprocesado usados en este proyecto han sido: la ecualización del histograma, el resaltado de la frontera, el suavizado y resaltado de la frontera y el filtro bilateral. Tanto para la segmentación basada en grafos como para K-means, el filtro que mejores resultados ha obtenido ha sido el filtro bilateral, con una mejora notable en ambos.

Los filtros de post-procesado usados en este proyecto han sido: el filtro basado en un número máximo de áreas y el filtro basado en un número mínimo de píxeles por área. Tanto para la segmentación basada en grafos como para K-means, el filtro que mejores resultados ha obtenido ha sido el filtro basado en un número máximo de áreas. Utilizando ambos filtros, tanto el filtro de preproceso como el de post-procesado, se obtiene un resultado mejor que utilizándolos individualmente, obteniendo el mejor resultado del sistema.

Este sistema se ha integrado con la herramienta Make3D, que reconstruye una imagen a una proyección 3D. También se concluye que, aunque la segmentación se aproxime a una segmentación hecha por una o varias personas, no es motivo suficiente para garantizar la calidad de la reconstrucción obtenida.

Por último, se ha realizado diversas pruebas donde se ha obtenido mejoras significativas en la segmentación gracias a los resultados presentados en la sección de experimentación.

5.2. Trabajo futuro

El trabajo futuro del sistema desarrollado tiene muchas vertientes. En general se puede escribir en dos grandes líneas de trabajo:

- **Mejora de la segmentación de imágenes**:
  1. Ampliación del el número de filtros de imagen, algoritmos de segmentación y métricas de evaluación para obtener mayor variedad y efectividad en la segmentación.
  2. Aplicación de técnicas de aprendizaje automático para obtener qué combinaciones, tanto de filtros y algoritmos como de sus respectivos argumentos, tienen los mejores resultados.
- **Mejora de la reconstrucción 3D**:
  1. Utilización de distintos métodos de reconstrucción 3D y poder promediar sus resultados.
  2. Ampliación del número de imágenes utilizadas para la reconstrucción, y sumar sus proyecciones, para obtener un resultado más veraz.

En resumen es posible ampliar enormemente la capacidad y funcionalidad de la arquitectura mediante la combinación adecuada de técnicas de filtrado y segmentación, junto con técnicas de clustering, que permitan la reconstrucción fidedigna (o de calidad) de entornos o escenarios 3D.

Bibliografía

[1] Giuseppina C. Gini and Alberto Marchi. Indoor robot navigation with single camera vision. In José Manuel Iñesta Quereda and Luisa Micó, editors, *PRIS*, pages 67–76. ICEIS Press, 2002.

[2] Holger Scherl, Stefan Hoppe, Markus Kowarschik, and Joachim Hornegger. Design and implementation of the software architecture for a 3-d reconstruction system in medical imaging. In *Proceedings of the 30th international conference on Software engineering*, ICSE '08, pages 661–668, New York, NY, USA, 2008. ACM.

[3] Ioannis Pachoulakis. 3d reconstruction and visualization of astrophysical wind volumes using physical models. *ACM Trans. Model. Comput. Simul.*, 18(4):14:1–14:36, September 2008.

[4] Jordi Freixenet, Xavier Muñoz, D. Raba, Joan Martí, and Xavier Cufí. Yet another survey on image segmentation: Region and boundary information integration. In *Proceedings of the 7th European Conference on Computer Vision-Part III*, ECCV '02, pages 408–422, London, UK, UK, 2002. Springer-Verlag.

[5] Aaron Hertzmann and Steven M. Seitz. Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1254–1264, August 2005.

[6] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1-3):7–42, April 2002.

[7] A. Saxena, Min Sun, and A.Y. Ng. Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):824–840, 2009.

[8] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(5):824–840, May 2009.

[9] Ashutosh Saxena, Andrew Ng, and Sung Chung. Learning Depth from Single Monocular Images. *NIPS*, 18, 2005.

[10] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[11] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[12] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[13] Ashish Thakur and R. S. Anand. Image quality based comparative evaluation of wavelet filters in ultrasound speckle reduction. *Digit. Signal Process.*, 15(5):455–465, September 2005.

[14] Punyaban Patel, Abhishek Tripathi, Banshidhar Majhi, and C. R. Tripathy. A new adaptive median filtering technique for removal of impulse noise from images. In *Proceedings of the 2011 International Conference on Communication, Computing & Security*, ICCCS '11, pages 462–467, New York, NY, USA, 2011. ACM.

[15] Seongwook Youn and Dennis McLeod. Improved spam filtering by extraction of information from text embedded image e-mail. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 1754–1755, New York, NY, USA, 2009. ACM.

[16] Cancelable biometric filters for face recognition. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3 - Volume 03*, ICPR '04, pages 922–925, Washington, DC, USA, 2004. IEEE Computer Society.

[17] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1992.

[18] Cliff Reiter. With j: image processing 1: smoothing filters. *SIGAPL APL Quote Quad*, 34(2):9–15, March 2004.

[19] Mark Nixon and Alberto S. Aguado. *Feature Extraction & Image Processing, Second Edition.* Academic Press, 2nd edition, 2008.

[20] William K. Pratt. *Digital image processing.* John Wiley & Sons, Inc., New York, NY, USA, 1978.

[21] Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. Local laplacian filters: edge-aware image processing with a laplacian pyramid. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 68:1–68:12, New York, NY, USA, 2011. ACM.

[22] Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. Local laplacian filters: edge-aware image processing with a laplacian pyramid. *ACM Trans. Graph.*, 30(4):68:1–68:12, July 2011.

[23] Charles Han, Bo Sun, Ravi Ramamoorthi, and Eitan Grinspun. Frequency domain normal map filtering. *ACM Trans. Graph.*, 26(3), July 2007.

[24] Charles Han, Bo Sun, Ravi Ramamoorthi, and Eitan Grinspun. Frequency domain normal map filtering. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[25] Naga K. Govindaraju, Brandon Lloyd, Yuri Dotsenko, Burton Smith, and John Manferdelli. High performance discrete fourier transforms on graphics processors. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, pages 2:1–2:12, Piscataway, NJ, USA, 2008. IEEE Press.

[26] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972.

[27] Davide Maltoni, Dario Maio, Anil K. Jain, and Salil Prabhakar. *Handbook of Fingerprint Recognition.* Springer Publishing Company, Incorporated, 2nd edition, 2009.

[28] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[29] Ruey-Ming Chao, Hsien-Chu Wu, and Zi-Chun Chen. Image segmentation by automatic histogram thresholding. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ICIS '09, pages 136–141, New York, NY, USA, 2009. ACM.

[30] Christos Nikolopoulos and Parupalli Prassad. On topologically equivalent dendrograms. In *Proceedings of the 17th conference on ACM Annual Computer Science Conference*, CSC '89, pages 464–464, New York, NY, USA, 1989. ACM.

[31] Sei-Ichiro Sakata, Fumihiro Ashida, and Hiroyoshi Tanaka. Kriging-based convex subspace single linkage method with path-based clustering technique for approximation-based global optimization. *Struct. Multidiscip. Optim.*, 44(3):393–408, September 2011.

[32] William F. Eddy, Audris Mockus, and Shingo Oue. Approximate single linkage cluster analysis of large data sets in high-dimensional spaces. *Comput. Stat. Data Anal.*, 23(1):29–43, November 1996.

[33] P. Dawyndt, H. De Meyer, and B. De Baets. The complete linkage clustering algorithm revisited. *Soft Comput.*, 9(5):385–392, May 2005.

[34] Morteza Haghir Chehreghani, Hassan Abolhassani, and Mostafa Haghir Chehreghani. Improving density-based methods for hierarchical clustering of web pages. *Data Knowl. Eng.*, 67(1):30–50, October 2008.

[35] Jinling Zhang, Yinghua Lu, Lin Wang, Hongxin Zhang, Bo Zhang, Yeqiu Wang, Kai Wu, and Stefan Wolfl. Uncentered (absolute) correlation clustering method fit for establishing theoretical sapk/jnk signaling pathway in human soft tissue sarcoma samples. In *Proceedings of the 7th international conference on Intelligent Data Engineering and Automated Learning*, IDEAL'06, pages 1329–1336, Berlin, Heidelberg, 2006. Springer-Verlag.

[36] Florian Landis, Thomas Ott, and Ruedi Stoop. Hebbian self-organizing integrate-and-fire networks for data clustering. *Neural Comput.*, 22(1):273–288, January 2010.

[37] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[38] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):881–892, July 2002.

[39] Maya R. Gupta and Yihua Chen. Theory and use of the em algorithm. *Found. Trends Signal Process.*, 4(3):223–296, March 2011.

[40] Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 563–572, New York, NY, USA, 2010. ACM.

[41] Ming-Chuan Hung and Don-Lin Yang. An efficient fuzzy c-means clustering algorithm. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 225–232, Washington, DC, USA, 2001. IEEE Computer Society.

[42] Dinh Khac Dong, Tran Dinh Khang, and Phan Anh Phong. Fuzzy clustering with hedge algebra. In *Proceedings of the 2010 Symposium on Information and Communication Technology*, SoICT '10, pages 49–54, New York, NY, USA, 2010. ACM.

[43] K. Lin and Ravikuma Kondadadi. A similarity-based soft clustering algorithm for documents. In *Proceedings of the 7th International Conference on Database Systems for Advanced Applications*, DASFAA '01, pages 40–47, Washington, DC, USA, 2001. IEEE Computer Society.

[44] Jingdong Wang. *Graph Based Image Segmentation: A modern approach.* VDM Verlag, Saarbrücken, Germany, Germany, 2008.

[45] Deepali Kelkar and Surendra Gupta. Improved quadtree method for split merge image segmentation. In *Proceedings of the 2008 First International Conference on Emerging Trends in Engineering and Technology*, ICETET '08, pages 44–47, Washington, DC, USA, 2008. IEEE Computer Society.

[46] John R. Smith and Shih fu Chang. Quad-tree segmentation for texture-based image query. In *In Proceedings of ACM Multimedia 94*, pages 279–286, 1994.

[47] Tobias Heimann. *Statistical Shape Models for 3D Medical Image Segmentation.* VDM Verlag, Saarbrücken, Germany, Germany, 2009.

[48] Ming Zhang, Ling Zhang, and H. D. Cheng. A neutrosophic approach to image segmentation based on watershed method. *Signal Process.*, 90(5):1510–1517, May 2010.

[49] Su-Nan Huang, Steven C. Chan, and Wei Ren. Mixture of automatically- and manually-controlled vehicles in intelligent transport systems. *J. Intell. Robotics Syst.*, 24(2):175–205, February 1999.

[50] M. Quartulli and M. Datcu. Bayesian model based city reconstruction from high resolution isar data. In *Remote Sensing and Data Fusion over Urban Areas, IEEE/ISPRS Joint Workshop 2001*, pages 58–63, 2001.

[51] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. 3d city modeling using cognitive loops. In *Video Proceedings for CVPR 2006 (VPCVPR'06)*, June 2006.

[52] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *Int. J. Comput. Vision*, 12(1):43–77, February 1994.

[53] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach.* Prentice Hall Professional Technical Reference, 2002.

[54] Subhodev Das and Narendra Ahuja. Performance analysis of stereo, vergence, and focus as depth cues for active vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(12):1213–1219, December 1995.

[55] Christian Frueh and Avideh Zakhor. Constructing 3d city models by merging ground-based and airborne views. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–562. IEEE, 2003.

[56] Sebastian Thrun and Ben Wegbreit. Shape from symmetry. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2*, ICCV '05, pages 1824–1831, Washington, DC, USA, 2005. IEEE Computer Society.

[57] Takayuki Nagai, T. Naruse, M. Ikehara, and Akira Kurematsu. Hmm-based surface reconstruction from single images. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 2, pages II–561–II–564 vol.2, 2002.

[58] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):690–706, August 1999.

[59] Atsuto Maki, Mutsumi Watanabe, and Charles Wiles. Geotensity: Combining motion and lighting for 3d surface reconstruction. *Int. J. Comput. Vision*, 48(2):75–90, July 2002.

[60] T. Lindeberg and J. Garding. Shape from texture from a multi-scale perspective. In *Computer Vision, 1993. Proceedings., Fourth International Conference on*, pages 683–691, 1993.

[61] Jitendra Malik and Ruth Rosenholtz. Computing local surface orientation and shape from texture forcurved surfaces. *Int. J. Comput. Vision*, 23(2):149–168, June 1997.

[62] J Malik and P Perona. Preattentive Texture-Discrimination With Early Vision Mechanisms. *Journal Of The Optical Society Of America A-Optics Image Science And Vision*, 7(5):923–932, MAY 1990.

[63] Lorenzo Torresani and Aaron Hertzmann. Automatic non-rigid 3d modeling from video. In *IN ECCV*, pages 299–312, 2004.

[64] Antonio Torralba and Aude Oliva. Depth estimation from image structure. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(9):1226–1238, September 2002.

[65] Jeff Michels, Ashutosh Saxena, and Andrew Y. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 593–600, New York, NY, USA, 2005. ACM.

[66] Erick Delage, Honglak Lee, and AndrewY. Ng. Automatic single-image 3d reconstructions of indoor manhattan world scenes. In Sebastian Thrun, Rodney Brooks, and Hugh Durrant-Whyte, editors, *Robotics Research*, volume 28 of *Springer Tracts in Advanced Robotics*, pages 305–321. Springer Berlin Heidelberg, 2007.

[67] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. *Int. J. Comput. Vision*, 40(2):123–148, November 2000.

[68] Stan Z. Li. *Markov random field modeling in image analysis*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001.

[69] Tony Lindeberg. Edge detection and ridge detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):117–156, November 1998.

[70] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pages 839–, Washington, DC, USA, 1998. IEEE Computer Society.

[71] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, September 2004.