UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Industrial

Diseño de un algoritmo para la optimización de la instalación de paneles solares fotovoltaicos y el cálculo del potencial solar a gran escala en entornos urbanos

Trabajo Fin de Grado

Grado en Ingeniería en Tecnologías Industriales

AUTOR/A: Pic Munuera, Emmanuel Jean Daniel

Tutor/a: Bastida Molina, Paula

Cotutor/a: Gómez Navarro, Tomás

CURSO ACADÉMICO: 2023/2024

AGRADECIMIENTOS

"En un momento tan importante para mí como lo es el final de mi carrera universitaria, me gustaría agradecer a todos los que me han apoyado en este trayecto, especialmente a mis padres, Nathalie Munuera y Laurent Pic, a mi pareja, María García-Romeu, y a mis amigos. De igual modo, quisiera expresar mi especial agradecimiento a mis tutores, Paula Bastida Molina y Tomás Gómez Navarro, por todo su apoyo y por ayudarme a realizar este trabajo sobre un tema que me apasiona tanto."

RESUMEN

La complejidad de los problemas causados por el cambio climático exige explorar nuevas soluciones y optimizar las existentes. Con el incremento global de la demanda de energía, se espera que las energías renovables jueguen un papel crucial en el suministro energético de los centros urbanos, destacando especialmente la energía solar fotovoltaica debido a su desarrollo comercial y la madurez de su tecnología. Sin embargo, uno de los mayores problemas que afronta esta tecnología es la escasez de superficies con características adecuadas para su correcto aprovechamiento.

Con esta idea en mente, este trabajo plantea una metodología para estimar el potencial solar fotovoltaico de una localidad a gran escala, basada en buscar superficies aptas para la óptima instalación de paneles solares fotovoltaicos por medio del uso de visión artificial para eliminar posibles obstáculos. La metodología propuesta agrupa una serie de parámetros necesarios para maximizar la energía anual generada por cada superficie como lo es la radiación solar recibida, las sombras proyectadas, la pendiente y la orientación cardinal. Además de facilitar, de forma automática, la mejor configuración de paneles valorando distintos tamaños, posiciones, inclinaciones y orientaciones con respecto al sol.

El algoritmo desarrollado utiliza ortofotos de alta definición y archivos de nubes de puntos 3D (LIDAR) obtenidos del Centro de Descargas del Centro Nacional de Información Geográfica (CNIG). También emplea información catastral proporcionada por las páginas de datos abiertos del Gobierno y de los ayuntamientos, así como datos de generación de energía de la herramienta web PVGIS.

La información es tratada con los softwares MATLAB y QGIS donde se hace uso de conceptos como visión artificial, máscaras binarias, filtrado de nubes de puntos, modelos digitales de elevaciones, mapas de irradiación, georreferenciación de capas, optimización con limitaciones geométricas, entre otros.

El estudio se aplica al barrio de Illa Perduda en Valencia como caso práctico. Esto debido a que Valencia fue elegida Capital Verde Europea 2024 y presenta, junto a la Delegación de Calidad Medioambiental, un gran apoyo a las iniciativas ambientales y programas de actuación basados en la racionalización de recursos y respeto al entorno. En dicha localidad, y tras aplicar el algoritmo completo, se obtuvo un valor de energía generable de 4,45 GWh/año para una superficie útil de 21.159 m^2. Además, se demostró que, entre otras cosas:

1. No siempre un arreglo con un panel de mayor potencia nominal será el que mayor energía genere.
2. La inclinación de panel más adecuada para maximizar la generación de energía es cerca de los 35°.
3. Orientar los paneles al sur en una cubierta con otra orientación no compensa el mayor número de paneles (y por tanto la mayor potencia instalada) que la configuración paralela a los bordes de la cubierta es capaz de acomodar en las superficies.

Palabras Claves: Energía Solar Fotovoltaica; Visión Artificial (AV); Optimización del Layout de Paneles; Light Detection and Ranging (LIDAR); Plan Nacional de Ortofotografía Aérea (PNOA); Formato de Archivo de Imagen Etiquetada (TIF); Formato comprimido LAS (LAZ); Formato de archivo láser (LAS)

ABSTRACT

The complexity of climate change issues necessitates exploring innovative solutions and optimizing existing ones. As global energy demand rises, renewable energy sources are anticipated to play a critical role in urban energy supply, with solar photovoltaic (PV) energy being particularly prominent due to its commercial development and technological maturity. However, a significant challenge for this technology is the limited availability of surfaces suitable for optimal utilization.

In response to this challenge, this study proposes a methodology for estimating the solar PV potential of a large urban area. The approach involves identifying suitable surfaces for optimal solar panel installation using computer vision to eliminate potential obstacles. The methodology integrates various parameters essential for maximizing annual energy generation from each surface, such as solar radiation received, projected shadows, slope, and cardinal orientation. It also automates the optimal configuration of panels by evaluating different sizes, positions, inclinations, and orientations relative to the sun.

The developed algorithm utilizes high-resolution orthophotos and 3D point cloud data (LIDAR) obtained from the National Geographic Information Center's (CNIG) Download Center. It also leverages cadastral information from government and municipal open data portals, as well as energy generation data from the PVGIS web tool.

The data is processed using MATLAB and QGIS software, incorporating concepts such as computer vision, binary masks, point cloud filtering, digital elevation models, irradiation maps, layer georeferencing, and geometric constraint optimization, among others.

This methodology is applied to the Illa Perduda neighborhood in Valencia as a practical case study. Valencia was selected as the European Green Capital 2024 and demonstrates strong support for environmental initiatives and action programs focused on resource rationalization and environmental sustainability, with significant backing from the Environmental Quality Delegation. At this location, and after applying the full algorithm, a generable energy value of 4.45 GWh/year was obtained for a usable area of 21,159 m^2. In addition, it was shown that, among other things:

1. An array with a higher rated power panel will not always generate the most energy.
2. the most suitable panel tilt to maximise energy generation is around 35°.
3. Orienting the panels to the south on a roof with another orientation does not compensate for the higher number of panels (and therefore the higher installed power) that the configuration parallel to the roof edges is able to accommodate on the surfaces.

Keywords: Photovoltaic Solar Energy; Artificial Vision (AV); Panel Layout Optimisation; Light Detection and Ranging (LIDAR); National Aerial Orthophotography Plan (PNOA); Tagged Image File Format (TIF); LAS compressed format (LAZ); Laser Archive Format (LAS).

RESUM

La complexitat dels problemes causats pel canvi climàtic exigix explorar noves solucions i optimitzar les existents. Amb l'increment global de la demanda d'energia, s'espera que les energies renovables juguen un paper crucial en el subministrament energètic dels centres urbans, destacant especialment l'energia solar fotovoltaica a causa del seu desenvolupament comercial i la maduresa de la seua tecnologia. No obstant això, un dels majors problemes que afronta aquesta tecnologia és l'escassetat de superfícies amb característiques adequades per al seu correcte aprofitament.

Amb aquesta idea en ment, aquest treball planteja una metodologia per a estimar el potencial solar fotovoltaic d'una localitat a gran escala, basada a buscar superfícies aptes per a l'òptima instal·lació de panells solars fotovoltaics per mitjà de l'ús de visió artificial per a eliminar possibles obstacles. La metodologia proposada agrupa una sèrie de paràmetres necessaris per a maximitzar l'energia anual generada per cada superfície com ho és la radiació solar rebuda, les ombres projectades, el pendent i l'orientació cardinal. A més de facilitar, de manera automàtica, la millor configuració de panells valorant diferents grandàries, posicions, inclinacions i orientacions respecte al sol.

L'algoritme desenvolupat utilitza ortofotos d'alta definició i arxius de núvols de punts 3D (LIDAR) obtinguts del Centre de Descàrregues del Centre Nacional d'Informació Geogràfica (CNIG). També empra informació cadastral proporcionada per les pàgines de dades obertes del Govern i dels ajuntaments, així com dades de generació d'energia de l'eina web PVGIS.

La informació és tractada amb els softwares MATLAB i QGIS on es fa ús de conceptes com a visió artificial, màscares binàries, filtrat de núvols de punts, models digitals d'elevacions, mapes d'irradiació, georeferenciació de capes, optimització amb limitacions geomètriques, entre altres.

L'estudi s'aplica al barri d'Illa Perduda a València com a cas pràctic, pel fet que va ser triada Capital Verda Europea 2024 i presenta, al costat de la Delegació de Qualitat Mediambiental, un gran suport a les iniciatives ambientals i programes d'actuació basats en la racionalització de recursos i respecte a l'entorn. En aquesta localitat, i després d'aplicar l'algoritme complet, es va obtindre un valor d'energia generable de 4,45 *GWh/any per a una superfície útil de 21.159 m². A més, es va demostrar que, entre altres coses:

1. No sempre un arranjament amb un panell de major potència nominal serà el que major energia genere.
2. La inclinació de panell més adequada per a maximitzar la generació d'energia és prop dels 35°.
3. Orientar els panells al sud en una coberta amb una altra orientació no compensa el major nombre de panells (i per tant la major potència instal·lada) que la configuració paral·lela a les vores de la coberta és capaç d'acomodar en les superfícies.

Paraules Claus: Energia Solar Fotovoltaica; Visió Artificial (AV); Optimització del Layout de Panells; Light Detection and Ranging (LIDAR); Pla Nacional d'Ortofotografia Aèria (PNOA); Format d'Arxiu d'Imatge Etiquetada (TIF); Format comprimit LES (LAZ); Format d'arxiu làser (LES).

ÍNDICE

Contenido

INDICE DE FIGURAS

INDICE DE TABLAS

Capítulo 1. Introducción

1.1 Antecedentes y motivación

La actual tendencia de crecimiento en las grandes sociedades desarrolladas tiene importantes repercusiones sobre el medio ambiente: escasez de recursos, de espacio, aumento del consumo energético debido al crecimiento de la población y el aumento de consumo per cápita y, como consecuencia, mayor presión sobre los recursos del planeta. Relacionado con esto, se estima que el consumo de energía mundial aumenta cerca de 2% cada año, y se cree que aumentará aún más en años futuros (Enerdata [1]).

La realidad es que la energía es indispensable en la sociedad moderna. Como consecuencia, para gestionar el problema se han implementado medidas para aumentar la participación de las energías renovables en el mix energético de la generación de electricidad. En Europa, desde los años 2000 se ha visto un incremento notable de la contribución de las energías renovables. Sin embargo, ese incremento siempre ha sido menor en España, lo que da lugar a una posible oportunidad de mejora (Anuario Estadístico BP [2]).

Siguiendo esa idea, se destaca el hecho de que España es el país con el mayor potencial solar de Europa por su alta media de radiación por metro cuadrado, tal y como se ve en la Figura 1.

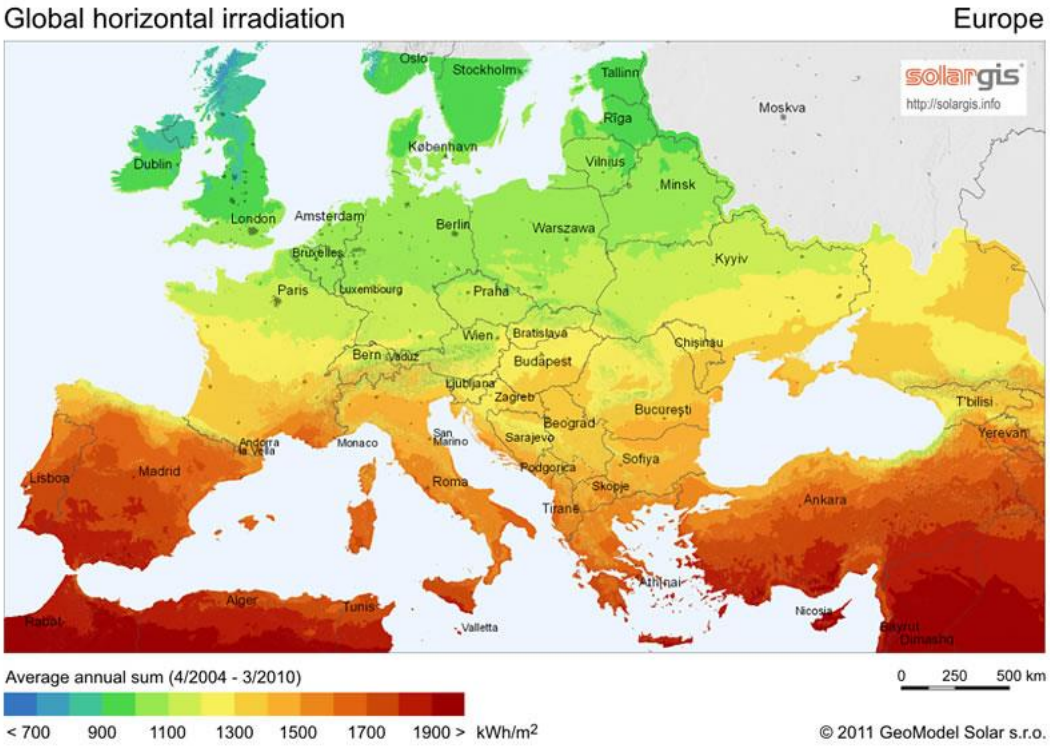


Figura 1. Mapa irradiación Solar en Europa [3].

Como se ha comentado, la presencia de energías renovables en el parque generador de energía eléctrica en España ha presentado un crecimiento notorio en los últimos años; las instalaciones de energía renovable representan el 61,3 % del total de 2023 como se ve en la Figura 2.

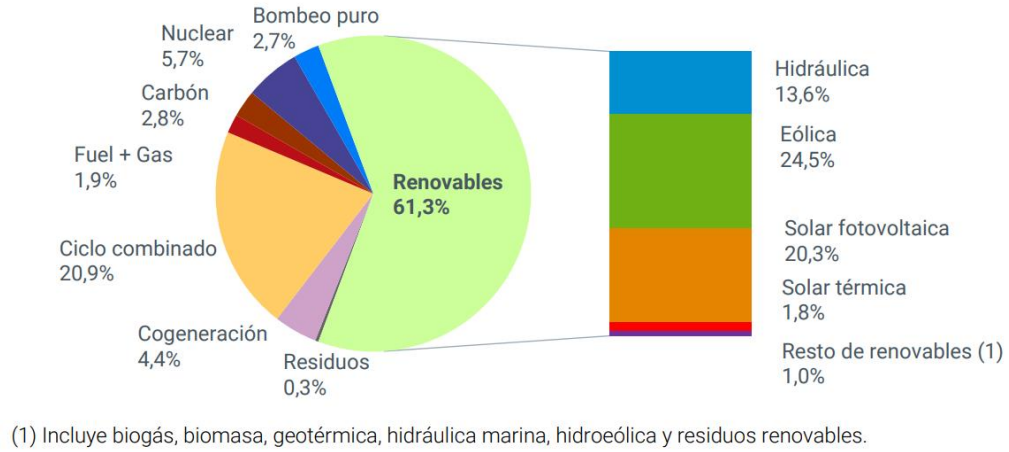


Figura 2. Estructura de potencia instalada (%) en España en 2023 [4].

De acuerdo con los informes de la Red Eléctrica de España (REE) este aumento se debe principalmente al auge de la potencia solar fotovoltaica, la cual supone un 20,3% del reparto y un 89.3% del crecimiento de la potencia instalada respecto al año anterior (Figura 3).

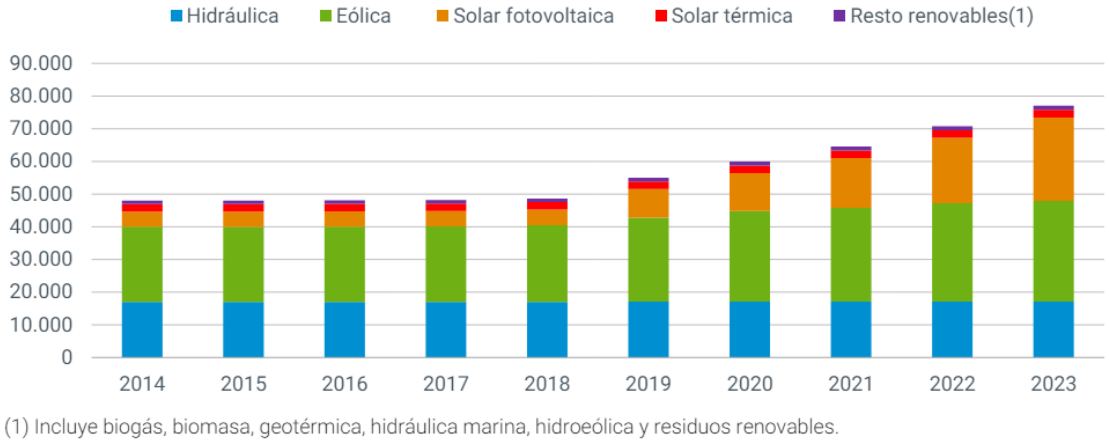


Figura 3. Evolución de la potencia instalada renovable (MW) [4].

Con esto, la energía solar fotovoltaica se coloca como la tercera fuente de energía más importante del país a nivel de potencia instalada y la cuarta en cuanto a generación con un 14% del total en 2023 (Figura 4).

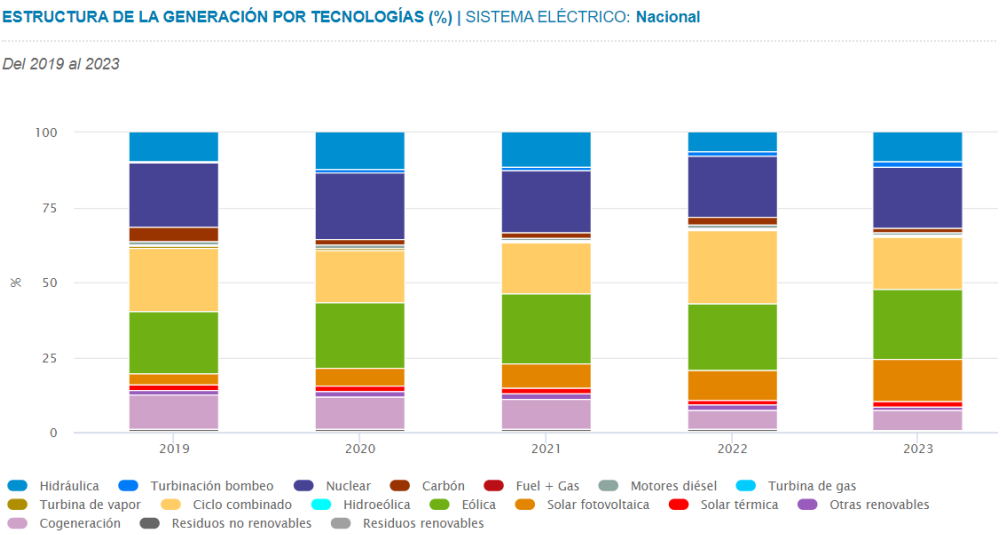


Figura 4. Estructura de generación por tecnologías a nivel nacional [5].

De este modo se demuestra su alto potencial, sobre todo frente a la inevitable reducción del uso de energías contaminantes (como los combustibles fósiles) en las próximas décadas, de los cuales se estima un descenso del 75% para 2035 a nivel mundial [6], lo cual requerirá de un sustituto para afrontar la fracción de generación que estos ocupaban.

Este trabajo plantea contribuir a dicha producción desde los hogares, los cuales, de acuerdo con la Unión Europea, consumen el 40% de la energía total [7]. Además, una instalación de generación próxima de su punto de consumo supone ahorros considerables en pérdidas y reduce la dependencia a la red y por lo tanto también a los combustibles fósiles de los que obtiene parte de dicha energía. Esta idea viene ligada al concepto de prosumidor, es decir, que las viviendas produzcan y consuman a la vez.

Por otro lado, de entre las iniciativas propuestas se encuentra el Pacto Verde Europeo y la Agenda 2030, las cuales tienen entre sus objetivos la reducción de las emisiones en un 55% en 2030 y la neutralidad climática en 2050 [8]. La energía fotovoltaica es un factor esencial en la transición europea hacia la neutralidad climática, ya que presenta una mejora frente a otras fuentes de energía por la reducción de forma indirecta de las emisiones de CO_2 para su obtención.

Continuando con el problema que suponen las emisiones, las ciudades son responsables de cerca del 70% de las emisiones de efecto invernadero a nivel mundial incluso ocupando una fracción de tierra correspondiente a menos del 2% de la superficie del planeta de acuerdo con informes de la ONU [9]. En otras palabras, existe una alta concentración de emisiones en poco espacio sumado al hecho de que los espacios que se podrían destinar para la colocación de paneles fotovoltaicos tienen una gran demanda para otros usos, como pueden ser la calefacción con chimenea, antenas para telecomunicaciones o entretenimiento, jardinería, etc.

Surge entonces un problema (la necesidad de energía y las altas emisiones) con una posible solución (el uso de paneles fotovoltaicos en tejados) que presenta algunas complicaciones (la falta de espacio). En relación al tema, otro problema que existe es la dificultad de estimar el potencial fotovoltaico de grandes cantidades de tejados en el marco de la planificación energética. Efectivamente, habitualmente el potencial de generación de electricidad mediante generación fotovoltaica de una serie grande de cubiertas, por agilidad, se calcula grosso. Se utilizan factores de conversión como 1 kW de potencia instalada por cada 10 m2 de cubierta, una generación eléctrica

de 1.400 kWh/(kWinstalado*año) en Valencia, etc. [10]. Es decir, se hacen cálculos poco precisos del potencial de generación de las cubiertas, y en este trabajo se buscará mejorar estos métodos, en particular para el entorno urbano.

1.2 Grado de alineación del trabajo con los Objetivos de Desarrollo Sostenible (ODS)

A continuación, se comentarán los ODS con un mayor grado de relación por el trabajo realizado.

En primer lugar, se destaca la amplia relación con el ODS 7 “Energía asequible y no contaminante”, el cual busca garantizar el acceso a una energía limpia y asequible para el desarrollo de las sociedades. Mediante los resultados de este trabajo se pretende ayudar a cumplir este objetivo facilitando una metodología capaz de optimizar la generación de energía limpia y no contaminante en uno de los entornos con más demanda.

En segundo lugar, se podría relacionar en menor medida con el ODS “Trabajo decente y crecimiento económico” por las oportunidades de empleo que podría generar esta metodología para el sector de la planificación energética.

En tercer lugar, la relación con el ODS 9 “Industria, innovación e infraestructuras” y el ODS 11 “Ciudades y comunidades sostenibles” viene dada por el objetivo del trabajo de ayudar a crear barrios sostenibles que puedan contribuir o incluso cubrir completamente su demanda energética de una forma eficiente.

En cuarto lugar, por la naturaleza del proyecto y su relación con las energías renovables como lo es la solar fotovoltaica, este colabora con el ODS 12 “Producción y consumo responsables” al reducir la dependencia de los recursos naturales no renovables, que es el fundamento principal de este ODS. Además, promueve el uso de la energía solar, un recurso natural renovable.

Por último, el ODS 13 “Acción por el clima” plantea aumentar la ambición de los proyectos en contra del cambio climático mediante el aumento de la escala de las medidas. En este proyecto se plantea una metodología capaz de abarcar una localidad del tamaño de un barrio completo en busca de contribuir a lograr las emisiones netas nulas, lo cual también abarca este ODS.

A continuación, se muestra la Tabla 1 a modo de resumen:

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No procede
ODS 1. Fin de la pobreza.				X
ODS 2. Hambre cero				X
ODS 3. Salud y bienestar.				X
ODS 4. Educación de calidad.				X
ODS 5. Igualdad de género.				X
ODS 6. Agua limpia y saneamiento.				X
ODS 7. Energía asequible y no contaminante.	X			
ODS 8. Trabajo decente y crecimiento económico.			X	
ODS 9. Industria, innovación e infraestructuras.	X			
ODS 10. Reducción de las desigualdades.				X
ODS 11. Ciudades y comunidades sostenibles.	X			
ODS 12. Producción y consumo responsables.	X			
ODS 13. Acción por el clima.	X			
ODS 14. Vida submarina.				X
ODS 15. Vida de ecosistemas terrestres				X
ODS 16. Paz, justicia e instituciones sólidas				X
ODS 17. Alianzas para lograr objetivos.				X

Tabla 1. Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030.

1.3 Objetivos:

Este trabajo de fin de grado busca desarrollar una metodología automatizada que permita calcular el potencial solar fotovoltaico a gran escala de una localidad, tomando un barrio de Valencia como zona de estudio y planteando soluciones a los diversos desafíos que supone trabajar en un entorno urbano. Los objetivos del estudio se muestran a continuación:

- Desarrollar un código automatizado por medio de visión artificial capaz de detectar zonas susceptibles de poder usarse para la colocación de paneles solares fotovoltaicos, evitando obstáculos con la ayuda de ortofotos de Valencia e información catastral.

- Realizar un estudio de la irradiación disponible en la zona para descartar zonas en sombra. Este análisis permitirá descartar superficies con un bajo porcentaje de horas de sol anuales.

- Calcular la pendiente de las distintas cubiertas de los edificios y su orientación cardinal para, en consecuencia, descartar aquellas con características inadecuadas y aplicar de forma más precisa el algoritmo que se desarrolló.

- Desarrollar un algoritmo que, de forma automática y a gran escala, sea capaz de calcular la mejor configuración de paneles que se podría disponer tanto de forma paralela al borde de la cubierta como orientados hacia el sur. Esto, con la idea de obtener la máxima energía anual. Este algoritmo evaluará las combinaciones de una serie de parámetros: las dimensiones y potencias de los distintos paneles solares comerciales, la inclinación de los paneles con respecto a la superficie en la que están apoyados, el lado del panel que está apoyado con el suelo, la orientación con respecto al sol de los paneles y la posibilidad de disponerlos de forma coplanar a la superficie.

1.4 Estado del arte:

A continuación, se ahondará en la literatura existente con la finalidad de realizar un repaso de los enfoques que diversos autores han implementado para afrontar problemas similares.

Para realizar un estudio del potencial fotovoltaico con enfoque en afrontar la problemática del espacio útil real aprovechable, Zhong et al. [11] desarrolló una estructura de 4 pasos: i) identificar segmentos de tejado contiguos, ii) delimitar áreas candidatas, iii) calcular la irradiación solar y iv) disponer los paneles solares fotovoltaicos (FV para abreviar) de forma óptima (este último paso con bases en otro trabajo que realizó junto a Tong [12]). Esto, usando el programa de pago ArcGIS.

Sin embargo, pese a que su estructura es bastante coherente y funcional, presenta varios puntos débiles:

Por un lado, si bien considera las sombras implícitamente a través del estudio de la irradiación media anual, no toma en consideración las sombras que los propios paneles puedan generar sobre otros. Tampoco considera, un espacio entre filas y/o columnas para ventilar mediante el paso de aire. Descarta otras configuraciones aparte de las disposiciones paralelas a la cubierta. Y finalmente, su método de detección basado en cambios bruscos de pendiente podría llevar a ignorar obstáculos o considerar obstáculos superficies que no lo son.

De entre los trabajos que han centrado sus esfuerzos en la precisión de la disposición de los paneles solares FV considerando la sombra proyectada por los mismos, se podría destacar al de Mignoni [13] quien elaboró un algoritmo en lenguaje de programación Python basado en disponer los paneles solares FV de tal forma que el coste sea mínimo para abastecer una demanda dada. Presenta una estructura que valora distintas orientaciones modificando el ángulo que forman los paneles con el norte o azimut. Al final, llega a una conclusión que apoya la idea comúnmente aceptada de que la mejor orientación es hacia el sur para el hemisferio norte.

Sin embargo, este último resulta poco versátil debido al hecho de que parte de una superficie que considera plana, por lo que su algoritmo no sería aplicable a muchas situaciones de la realidad, o al menos no con total precisión en el caso de aplicarlo a superficie con cierta pendiente. Este problema también está presente en otros estudios como el de Alharbi [14], quien ahonda más en la disposición multi-azimut en la que trabajó Zhong. También en el estudio de Rehman [15], cuya innovación fue crear dos modelos: uno variando la pendiente entre cada fila de paneles y otro variando la separación entre filas demostrando que este último enfoque generaba una mayor energía anual a costa de una gran complejidad de automatización e instalación.

Un estudio más cercano geográficamente con además resultados de mucho interés es el de González [16], quien concluye que el ángulo óptimo de los paneles solares FV en la península para obtener la máxima producción anual es cercano a los 34°, aunque varía si el enfoque es por estaciones del año bajando hasta 20° en verano y subiendo a 55° en invierno. También comenta que el uso de soportes da mucha versatilidad y permite ajustar el ángulo al óptimo frente a una disposición coplanar la cual se ajustaría solo a la pendiente de la cubierta. Pese a esto, no ahonda mucho en el tema al no buscar un criterio claro para elegir una configuración sobre la otra ni mencionar las ventajas de la disposición coplanar más allá del coste de instalación.

Añadido a las debilidades encontradas en los trabajos citados ninguno parece buscar y comparar diferentes tamaños de paneles o hacer distinciones claras entre una colocación coplanar y una con soportes. Además, suelen limitar sus zonas de estudio a una o pocas superficies.

De igual manera existen programas tales como PVsyst [17] y Homer[18] desarrollado por el laboratorio nacional de energías renovables de Estados Unidos (NREL) los cuales ofrecen un estudio para planificar disposiciones de paneles solares fotovoltaicos pero que solo funcionan para zonas geográficas limitadas, requieren de indicar la disposición de los paneles manualmente y su enfoque se basa exclusivamente en optimizar parámetros económicos.

Resumiendo, tras el estudio del estado del arte y añadiendo lo que Burkhart y Jones estudiaron sobre el sombreado [19], se podría decir que los parámetros clave para la optimación de superficies para su aprovechamiento para disponer paneles solares FV teniendo en cuenta las sombras proyectadas y buscando la máxima energía anual son los siguientes:

- Latitud de la superficie.
- Las dimensiones de los paneles.
- El lado en el que se apoyan sobre las superficies.
- La inclinación de los paneles.
- La irradiación solar en la zona.
- El ángulo con respecto de azimut de los paneles.

A estos parámetros, se le añadió el estudio de la influencia de la pendiente propia de la superficie de apoyo y distinciones relativas a la disposición coplanar o con soportes. En este proyecto se busca resolver los problemas mencionados incluyendo las separaciones entre filas y columnas para la ventilación, mejorar la detección de obstáculos, comparar las configuraciones paralelas al borde de la cubierta con la alineada con el sur, y diseñar un método ágil que se pueda aplicar a una zona de estudio de la escala de un barrio completo. Los resultados se compararon con los estudios agrupados por E. Fakhraian, M.A. Forment, F.V. Dalmau et .al [20].

Capítulo 2. Metodología

En este capítulo se explicará la metodología desarrollada en este estudio para su uso a gran escala. A continuación, se presenta un flujograma simplificado de dicha metodología:

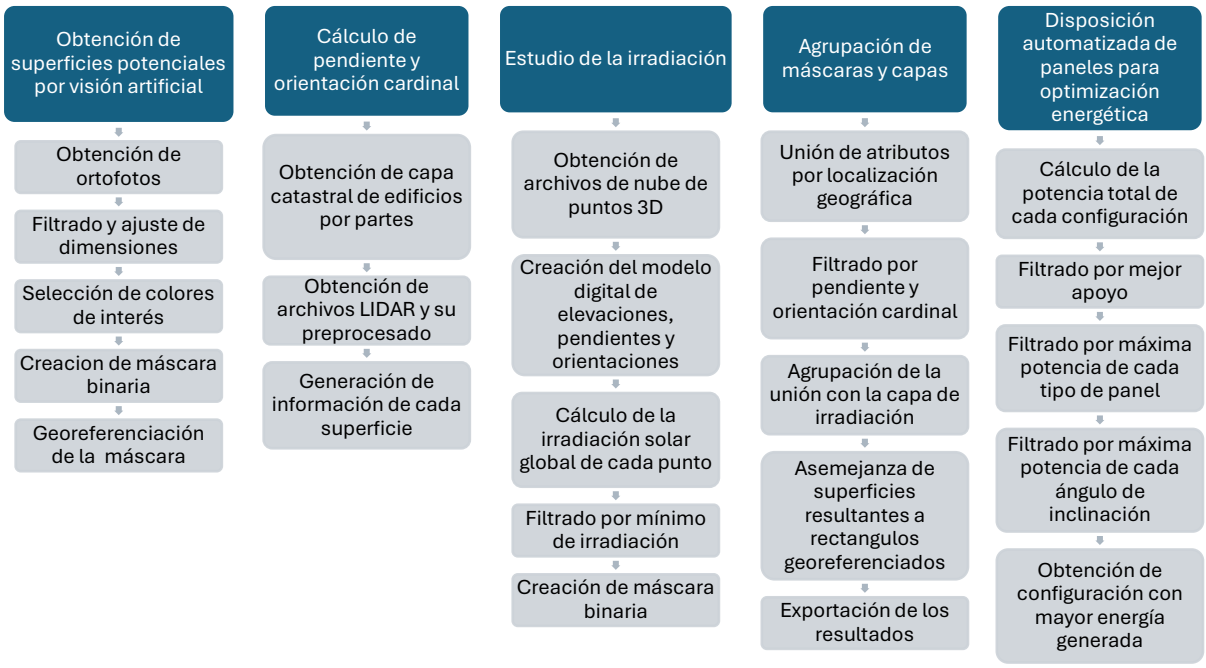


Figura 5. Flujograma de la metodología.

2.1 Obtención de superficies potenciales por visión artificial

En este apartado se desarrollará el primer paso de la metodología de este estudio. Su finalidad es obtener una capa georreferenciada con todas las posibles superficies aptas para la instalación de paneles solares FV. Se entienden como "aptas" todas aquellas que forman parte de un tejado/cubierta sin obstáculos de acuerdo con una coincidencia de colores previamente definidos y mediante el uso de visión artificial, como lo haría el ojo humano. Se realizó de esta manera ya que ninguna otra fuente de datos es tan fiable. Otras fuentes como las catastrales o LIDAR carecen de información sobre obstáculos, muretes, reformas posteriores a cuando se tomaron los datos, etc. Para conseguir el objetivo del apartado se desarrollaron dos códigos de MATLAB:

1. MyRoofTypeSet.m: código que permite crear regiones de color o modificar existentes mediante la selección de pixeles en una imagen TIF (archivo de imagen que facilita almacenar una gran cantidad de datos de alta calidad sin pérdida de información), propiciando una selección precisa de superficies de interés para su uso en la detección de diferentes tipos de tejados de forma automática.

2. MyRoofTypeDetection.m: código que detecta de forma automática los tejados coincidentes con una base de datos creada con el código "MyRoofTypeSet" al introducir una imagen TIF de la zona de interés de la que se busca identificar los tejados para finalmente crear una máscara binaria con ellos. Cuenta con técnicas de creación de máscaras binarias (ceros y unos), georreferenciación, visualización, procesado de imagen por erosión (reducción del tamaño) y dilatación (aumento del tamaño) y herramientas interactivas para realizar modificaciones manuales de considerarlo oportuno.

Previamente al uso de los códigos de MATLAB es necesario llevar a cabo un pequeño preprocesado de las imágenes para un funcionamiento preciso y rápido:

Primero, es necesario obtener las ortofotos de alta definición a utilizar según la zona a estudiar. Para el caso de España, dichas ortofotos se pueden obtener del Centro Nacional de Información Geográfica (CNIG) [21] en la pestaña "Fotos e imágenes aéreas". Si el área de interés se encuentra entre dos zonas se pueden unir los archivos que la conforman usando la herramienta *Ráster > Miscelánea > Combinar.* Esta herramienta, así como muchas que se usarán en la metodología pertenecen al software gratuito QGIS[1] [22] el cual se recomienda para esta operación y las posteriores por su comodidad y gran variedad de recursos. Una vez obtenidas las imágenes más actuales, es altamente recomendado recortarlas al área de estudio para agilizar los procesos posteriores, sobre todo por las exigencias en materia de RAM del dispositivo utilizado y para reducir los tiempos de carga. Esto se puede conseguir con la herramienta *Ráster > Extracción > Cortar ráster por capa de máscara* de QGIS. Este recorte podría ser más o menos necesario según la potencia del equipo.

Como paso opcional, es posible eliminar todo lo que no sean edificios mediante el uso de información catastral en QGIS y con la herramienta ya mencionada. Esto con la finalidad de ir un paso más allá en lo referente a la calidad del resultado. Se ahondará en las ventajas y desventajas de este paso opcional en el caso de estudio del trabajo (más adelante).

Una vez los pasos previos estén completados, el siguiente paso es crear la máscara binaria. Este proceso empieza con el código *MyRoofTypeSet.m*, donde inicialmente se le pedirá al usuario un archivo TIF correspondiente a la zona de estudio completa o un fragmento de ésta que se considere que cuente con toda la variedad de colores de tejados. A continuación, se le preguntará si desea crear una nueva matriz de tejados o añadir colores a una existente, todo dando la opción de elegir el nombre tanto de la matriz como del color mediante ventanas emergentes. Posteriormente, el usuario interactuará con la imagen que introdujo para seleccionar muestras de pixeles a añadir a las regiones de colores que ha creado. De ese modo, el código calculará de forma automática las estadísticas de cada color en los modelos RGB y HSV (modelos basados en la mezcla de rojo, verde y azul donde cada color se asocia a tres números comprendidos de 0 a 255 y 0 a 1 respectivamente).

Una vez se cuente con la estructura o matriz de tejados (llamada "RoofType") con sus respectivos colores (llamados "colorRegions") y sus estadísticas, se procede a hacer uso del código *MyRoofTypeDetection.m*. En primer lugar, se le pedirá al usuario introducir la imagen TIF de la zona de la que desea detectar los tejados. A continuación, creará dos máscaras binarias usando la información de colores en la matriz de tejados: una primera a partir de los valores umbrales de los RGB y HSV, y una segunda a partir de sus medias y desviaciones estándar. Seguidamente, filtrará y fusionará dichas máscaras para obtener una única que refleje las superficies aptas en la zona de

[1] QGIS es un Sistema de Información Geográfica profesional, gratuito y de código abierto, que permite crear, visualizar, analizar, editar y publicar información geoespacial.

interés. Finalmente, el código georreferencia la máscara con la información de la imagen TIF original y guarda los resultados. Para identificarla a posteriori se llamará la máscara final georreferenciada como “MaskGeo”.

2.2 Cálculo de pendiente y orientación cardinal

A continuación, se explicará el segundo paso de la metodología propuesta. Este paso tiene como objetivo determinar la pendiente y la orientación cardinal de cada superficie obtenida en el paso 2.1 para poder aprovechar dicha información para la óptima colocación de paneles solares fotovoltaicos. Para conseguir este objetivo se hará uso de nuevo del software gratuito QGIS. Originalmente se planteó realizarlo por Matlab, pero esta idea se descartó debido a lo complejo que resultaba el filtrar las nubes de puntos para cada edificio particular, la falta de precisión en el cálculo de la pendiente y la imposibilidad de asignar una orientación a dicha pendiente. El software QGIS cuenta con complementos pensados para lograr estos objetivos de forma cómoda y rápida.

Como pasos previos, es altamente recomendable tener a disposición una capa vectorial de la superficie de interés dividida por partes. Dicha capa se puede obtener gracias a la información catastral libre que ofrecen los ayuntamientos, sitios del gobierno o complementos de QGIS (se ahondará más en el caso práctico más adelante). Una vez obtenida, se recortará para conservar solo las partes de edificios pertenecientes a la zona de interés mediante la herramienta *Vectorial > Herramientas de geoproceso > Cortar*.

Por otro lado, es necesario obtener los archivos LAZ de la zona de interés. Para ello se puede recurrir de nuevo al centro de descargas del Centro Nacional de Información Geográfica (CNIG) [21] en la pestaña “Modelos Digitales de Elevaciones”. Los archivos LAZ son archivos formados por nubes de puntos en 3D capturados con tecnología LIDAR (Light Detection and Ranging) la cual consiste en un sistema de sensores equipados en aviones que emiten pulsos de luz y obtienen la información con su retorno. Dichos archivos cuentan con una resolución dada por su densidad de puntos la cual es de 0.5 puntos por metro cuadrado.

Una vez obtenidos los archivos más actuales, podría ocurrir que la zona de interés esté entre dos o más archivos LAZ ya que estos abarcan una superficie de 2x2 km2. Si este fuera el caso, basta con unirlos con la herramienta “*lasmerge*” del complemento “*LAStools*” de QGIS. Tanto si este paso previo fue necesario como si no, el paso siguiente consiste en recortar la nube de puntos para conservar solo el área deseada. Esto se puede conseguir mediante el uso de la herramienta propia de QGIS “Cortar nube de puntos”. Se llamará a esta capa “*LASRecortado*” para referenciarla en pasos futuros.

A continuación, es preciso filtrar la nube de puntos LASRecortado mediante la función propia de QGIS “Filtrar nube de puntos”, donde habrá que seleccionar como expresión de filtrado:

$$Classification = 6$$

Esto es posible ya que en la información interna de los puntos de los archivos LAS existe una clasificación la cual se indica a continuación:

- 0: Creado, nunca clasificado
- 1: No clasificado
- 2: Suelo

- 3: Baja vegetación
- 4: Vegetación media
- 5: Alta vegetación
- 6: Edificio
- 7: Punto bajo (ruido)
- 8: Reservado
- 9: Agua
- 10: Vía férrea
- 11: Superficie de rodamiento
- 12: Reservado
- 13: Cable-Guarda (protección)
- 14: Cable-Conductor (fase)
- 15: Torre de transmisión
- 16: Conector Estructura-Cableado (aislante)
- 17: Plataforma de puente
- 18: Ruido alto

Como se puede apreciar, la clasificación que resulta de interés es únicamente la de edificios, es decir la número 6, de allí la expresión de filtrado. Esto permitirá asociar los puntos pertenecientes a edificios a las divisiones que los conforman, evitando que se tomen en consideración puntos del entorno.

Teniendo la capa vectorial de divisiones catastrales y la nube de puntos filtrada para los edificios de la zona de interés, se procede a hacer uso de la función "*LidarRoofTopAnalysis"* perteneciente al complemento "WhiteBoxTools". Una vez seleccionada habrá que facilitarle las dos capas mencionadas anteriormente y ajustar los parámetros según las necesidades. Los valores por defecto suelen ser adecuados salvo por el "radio de búsqueda" el cual habrá que ajustar a las dimensiones de la zona de estudio.

Una vez ejecutado se obtendrá como resultado una capa que presenta las divisiones por partes de los edificios cada una con información relativa a su pendiente, orientación, máxima elevación, área, etc. Esta información fue posible de calcular gracias a la nube de puntos 3D del archivo LAS.

Cabe mencionar que QGIS referencia la orientación cardinal con valores de 0° a 360°, es decir con el sistema sexagesimal como en una circunferencia, siendo 0 alineado con el Norte, 90 alineado con el Este, 180 al Sur y 270 al Oeste (como se ve en la Figura 6).

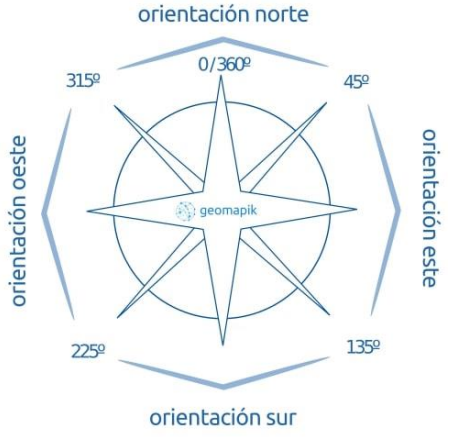


Figura 6. Sistema de medición cardinal de QGIS.

Finalmente, es posible filtrar la capa final a través del Constructor de consultas (el constructor de consultas es una herramienta para filtrar datos que se encuentra en las propiedades de cada capa) para eliminar el ruido que puede haberse generado por puntos que no se pudieron asociar a ninguna superficie gracias a una expresión lógica como, por ejemplo:

$$Área > Área\ mínima$$

Cabe mencionar que generalmente los puntos aislados son mínimos debido a lo riguroso del preprocesado. La capa final se llamará "Pendiente" para referenciarla en etapas futuras.

2.3 Estudio de la irradiación

En la tercera parte de la metodología propuesta, se buscará obtener una capa binaria georreferenciada de la zona de interés donde los puntos con valor 1 sean aquellos con una irradiación global mayor a un mínimo establecido. La finalidad última de este apartado es conseguir eliminar las áreas que permanecen en sombra gran parte del día, ya que no serían adecuadas para el objetivo final que es generar electricidad con paneles solares fotovoltaicos.

Lograr este objetivo es posible siguiendo una serie de pasos en el software QGIS, los cuales se describen a continuación.

Partiendo de la capa *LASRecortado* obtenida en el paso 2, se hará uso de la función "*blast2dem*" del complemento *LAStools* para convertir la nube de puntos en un archivo DEM (modelo digital de elevaciones) en formato TIF. Esta función resulta muy cómoda ya que es capaz de procesar millones de puntos secuencialmente para conseguir tratar grandes volúmenes de datos eficientemente.

Obtenido el DEM de la zona de interés, se procede a usarlo para calcular un modelo digital de pendientes (MDP) y un modelo digital de orientaciones (MDO). Para ello se utiliza la herramienta *Ráster > Análisis > Pendiente y Ráster > Análisis > Aspecto* respectivamente.

Estos tres modelos contienen toda la información necesaria para poder realizar el estudio de la irradiación de una zona. Para lograr este objetivo se recurrirá a la función "*r.sun.insoltime*" del complemento GRASS a la que habrá que facilitarle los tres modelos calculados previamente. Se elige la opción de irradiación global y se ajustan los parámetros según la necesidad. El inconveniente es que esta función solo calcula la irradiación para un día del año escogido manualmente. Para solucionar este problema basta con copiar el comando de la función como código de Python e introducirlo en la consola de Python de QGIS añadiendo un pequeño bucle como el que se muestra en el "ANEXO 1. Códigos de QGIS".

Una vez se tengan las capas con la irradiación global de cada día del año, en *Ráster > Calculadora Ráster* se suman y se divide el resultado por 365 para obtener una media diaria de la irradiación anual.

$$IrradiaciónMediaAnual = \frac{\sum_{i=1}^{365} Irradiación(i)}{365} \quad (1)$$

Con la capa resultante se hace uso de nuevo de la calculadora ráster y de una expresión lógica para conservar solo los puntos con irradiación mayor a la mínima requerida.

$$IrradiaciónMediaAnual > IrradiaciónMínima \quad (2)$$

Los resultados se pueden comparar con la información que facilitan aplicaciones de acceso web como “Huella Solar” la cual es una aplicación web diseñada para generar y visualizar mapas de radiación solar y soleamiento en entornos urbanos desarrollada por el arquitecto Alejandro Díaz Morales[23]. Para este estudio no es posible usarla directamente por la imposibilidad de exportar los mapas generados de forma que contengan su información por puntos y de forma georreferenciada.

Finalmente, para poder trabajar con la capa obtenida en los pasos siguientes, es necesario poligonizarla usando la herramienta *Ráster > Conversión > Poligonizar (ráster a Vectorial)* obteniendo así una capa binaria donde los “1” son los puntos que cumplen tener una irradiación mayor a la mínima (ecuación 2) y “0” los que no. Para apreciarlo, en el Constructor de consultas de la capa polinizada se indica conservar solo los valores iguales a 1. Para identificarla a posteriori se llamará la capa final obtenida “IrradiacionFinal”.

Cabe mencionar que no es necesario filtrar la capa de IrradiacionFinal por edificios debido a que ya se ha hecho en el paso 2.1.

2.4 Agrupación de máscaras y capas

El cuarto paso de la metodología tiene como objetivo agrupar las capas resultantes de los pasos anteriores (la capa sin obstáculos del paso 2.1, la capa con información de pendiente/orientación del paso 2.2 y la de irradiación del paso 2.3) para unificar la información y así determinar la superficie final útil para la disposición de paneles solares fotovoltaicos.

En primer lugar, se trasladará la información de inclinación y orientación de la capa *Pendiente* a la máscara *MaskGeo*. La forma más fácil de conseguir esto sería mediante la opción de “Unión” en las propiedades de capa, pero al no tener ningún atributo clave en común, es decir, una columna en su tabla de atributos con valores idénticos en ambas capas, esta opción no es realizable. En su lugar se usará la herramienta propia de QGIS “Unir atributos por localización” eligiendo añadir los campos “*Slope*” y “*Aspect*” con la opción “*Tomar atributos del traslape más grande solamente*”. Esto es debido a que, por la naturaleza de las agrupaciones de una nube de puntos, las superficies generadas en las dos capas de entrada no coinciden exactamente. Como consecuencia, esta opción es la más adecuada para asociar los atributos a la máscara sin obstáculos.

En este punto se cuenta con una capa libre de obstáculos que contiene cada superficie con su respectiva información de pendiente y orientación “*Union_MASK_Pend*”. El paso siguiente es usar esa información para restringir las superficies disponibles a aquellas que cumplan con la pendiente y orientación deseada.

Los criterios para valorar la adecuación de una superficie para la instalación de paneles solares FV pueden resumirse en dos aspectos fundamentales: la planitud y la orientación.

1. **Planitud**:
    - La planitud de una superficie se determina mediante la inclinación de esta. Una superficie se considera plana si su inclinación se encuentra dentro de un intervalo específico.
    - Este intervalo tiene un mínimo de 0 grados (completamente horizontal) y un máximo que varía según la tolerancia deseada del proyecto. Por ejemplo, se puede

considerar un máximo de inclinación entre 1 y 5 grados, dependiendo de los requisitos específicos de la instalación.

2. **Orientación**:

    - La orientación de la superficie debe enfrentar el recorrido del sol durante el día. La orientación óptima varía según el hemisferio y la latitud en la que se encuentre la zona de estudio.
    - Para el hemisferio norte, la orientación ideal es hacia el sur, mientras que, para el hemisferio sur, es hacia el norte. Este criterio también debe ajustarse dentro de un intervalo de tolerancia adecuado que maximice la exposición solar [16].

Para ello, se procede de la siguiente forma:

En el constructor de consultas de la capa "*Union_MASK_Pend*" se escribe la expresión lógica:

$$(Slope > 0 \ \& \ Slope < MaxPlanitud) \mid (Aspect > MinOrientacion \ \& \ Aspect < MaxOrientacion)$$

Tras realizar esta operación se habrán eliminado las superficies con una pendiente y orientación no deseada de acuerdo con los límites establecidos.

A continuación, se juntará la capa resultante con la capa vectorizada "*IrradiancionFinal*" para ya contar con una única capa con las superficies que cumplen todas las condiciones impuestas: libre de obstáculos; pendiente y orientación adecuada; y superior a la irradiación mínima.

Para conseguir esto se hará uso de nuevo de la herramienta *Vectorial > Herramientas de geoproceso > Cortar* eligiendo como capa de entrada la capa "*Union_MASK_Pend*" ya filtrada y la capa "*IrradiancionFinal*" como capa de superposición. La ejecución de esta herramienta generará como resultado una capa vectorial donde solo las partes de las superficies que queden dentro de la capa "*IrradiancionFinal*" estarán presentes. A esta capa se le llamará "*Union_MASK_Pend_RadAnual".*

Finalmente, como resultado de las operaciones vistas hasta este punto, se han conseguido las partes de la localidad de partida que son óptimas para la disposición de paneles solares. El problema que surge ahora es que estas superficies son en su mayoría irregulares y para el siguiente paso (el algoritmo automatizado para optimizar la disposición de paneles, paso 2.5) es necesario asemejarlas a superficies rectangulares. Dicha acción es necesaria ya que permite la automatización y facilita la futura instalación.

Para lograr convertir las geometrías irregulares a rectángulos se siguen los siguientes pasos:

1. Primero, se generará una cuadrícula sobre toda la zona de interés de forma similar a como lo han hecho autores como Alharbi et al. [14] pero a gran escala. Para ello, se usa la *herramienta Vectorial > Herramientas de investigación > Crear cuadricula*. Una vez ahí se elige "Rectángulo" como tipo de cuadrícula, la extensión (seleccionando la capa ráster cortada obtenida en el paso 2.1) y las dimensiones de los recuadros según los requerimientos (dimensión horizontal y dimensión vertical). Es importante tener en cuenta que cuanto más pequeños sean los recuadros de la cuadrícula mejor se ajustarán a las formas irregulares pero el tiempo de

cómputo aumentará proporcionalmente. La cuadrícula generada también se puede girar al gusto habilitando las opciones de digitalización avanzada en la barra de herramientas.

2. En segundo lugar, se recortará la cuadrícula para que solo permanezcan aquellos recuadros que queden dentro de la zona útil. Previamente a esta operación es muy aconsejable crear un índice espacial para los elementos de la cuadrícula para reducir en gran medida los tiempos de carga (en Propiedades de la capa> Fuentes > Crear índice espacial). El recorte de la cuadrícula se consigue fácilmente con la herramienta *Vectorial > Herramientas de geoproceso > Cortar* eligiendo como capa de entrada la cuadricula y como capa de superposición la capa "*Union_MASK_Pend_RadAnual".* Inevitablemente ocurrirá que algunos recuadros de la cuadrícula aparezcan cortados, esto se soluciona añadiendo un campo con el área en la tabla de atributos de la capa y creando una consulta en el Constructor de consultas para conservar solo los recuadros con área igual al área completa indicada al crear la cuadrícula. La expresión para introducir en el constructor de consultas es:

$$Área = Dimensión\ vertical\ x\ Dimensión\ horizontal \qquad (3)$$

3. El siguiente paso es seleccionar los recuadros para construir los rectángulos deseados con la función "Seleccionar objetos espaciales por polígono" y unirlos con la herramienta Vectorial > *Herramientas de geoproceso > Disolver.* Una vez estén todas se combinan con la herramienta "*Unir capas vectoriales"* creando la capa *"Combinado".* Puede ocurrir que en la información interna de los rectángulos se conserven los vértices de los recuadros de los que se formaron, esto se resuelve aplicando la función "Envolvente convexa" seguida de la herramienta de geometría vectorial "Simplificar" con el método de simplificación por distancias de Douglas-Peucker [24]. Esto conservará solo los cuatro vértices del rectángulo y además los numerará de igual forma para todos lo cual facilita mucho los siguientes pasos.

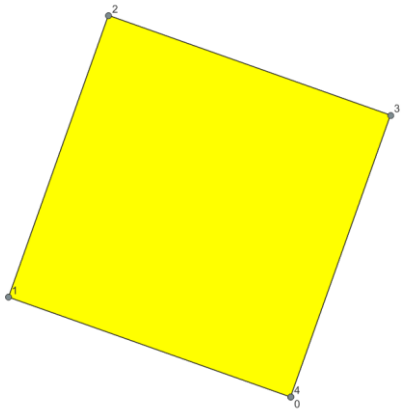


Figura 7. Rectángulo de cubierta con vértices numerados.

El método de simplificación llama "vértice 0" al vértice más al sur como prioridad y en caso de igualdad de condiciones pasará a ser el que posea mayor proximidad al oeste. El resto se numeran en sentido horario.

4. A continuación, se pasará la información de la capa "*Union_MASK_Pend_RadAnual"* a los rectángulos generados con la herramienta "Unir atributos por localización" eligiendo añadir los

campos "*Slope*" y "*Aspect*", con el tipo de unión "*Crear objeto separado para cada objeto coincidente (uno a muchos)*". Así, la pendiente y la orientación cardinal de cada rectángulo se encontrará organizada en su tabla de atributos. Por otro lado, es necesario generar otros tres campos en la tabla: el ancho, largo y el ángulo de medianera de cada rectángulo. Esto se puede conseguir fácilmente con trigonometría introduciendo las expresiones siguientes dentro de la tabla de atributos de la capa:

- Se considera que el ancho es la dimensión que une los vértices 1 y 2 o lo que es lo mismo los vértices 0 y 3. Se eligió usar los puntos 1 y 2, quedando la expresión en lenguaje de QGIS de la siguiente manera:

$$sqrt((\$y\_at(2) - \$y\_at(1))^2 + (\$x\_at(2) - \$x\_at(1))^2)$$

  Siendo:

  sqrt la función que devuelve la raíz cuadrada.

  $y\_at(i)$ la coordenada y en el vértice i.

  $x\_at(i)$ la coordenada x en el vértice i.

- Se considera que el largo es la dimensión que une los vértices 0 y 1º lo que es lo mismo los vértices 2 y 3. Se eligió usar los puntos 1 y 2, quedando la expresión en lenguaje de QGIS de la siguiente manera:

$$sqrt((\$y\_at(1) - \$y\_at(0))^2 + (\$x\_at(1) - \$x\_at(0))^2)$$

- Finalmente, en este trabajo se referirá al ángulo de medianera como el mínimo ángulo que forma la cubierta con el sur geométrico, siendo 0° totalmente alineado con el sur, -90° totalmente al este y 90° totalmente orientado al oeste. Se planteó de esta manera ya que así es el sistema de referencia en PVGIS el cual se comentará más adelante. De esta forma es posible determinar dicho ángulo con la siguiente expresión en QGIS:

$$if(abs(degrees(azimuth(make\_point(\$x\_at(0), \$y\_at(0)), make\_point(\$x\_at(3), \$y\_at(3)))) - 90) \geq abs(degrees(azimuth(make\_point(\$x\_at(0), \$y\_at(0)), make\_point(\$x\_at(1), \$y\_at(1))))) - 270), degrees(azimuth(make\_point(\$x\_at(0), \$y\_at(0)), make\_point(\$x\_at(1), \$y\_at(1)))) - 270, degrees(azimuth(make\_point(\$x\_at(0), \$y\_at(0)), make\_point(\$x\_at(3), \$y\_at(3)))) - 90\ )$$

  Siendo:

  abs la función que devuelve el valor absoluto.

  degrees la función que convierte radianes a grados sexagesimales.

  azimuth la función que devuelve el ángulo de azimut en radianes.

  make_point la función que crea un punto a partir de sus coordenadas.

- Como paso adicional, es muy recomendable para organizar la información y asociar los resultados futuros a cada cubierta crear un índice numérico de cada rectángulo de cubierta basado en la posición geográfica de estos. Esto debido a que QGIS crea por defecto una

columna de Índices para los elementos de las capas, pero de forma arbitraria. Con la siguiente expresión es posible generar un índice para cada elemento basado con su proximidad al norte:

$$array\_find(array\_sort(array\_agg(ymax(@geometry)), False), ymax(@geometry)) + 1$$

Siendo:

array_find la función que devuelve un índice numérico, en este caso empezando en 1 y aumentando +1 con cada iteración.

array_sort la función que organiza las filas según la expresión facilitada.

ymax la función que devuelve la mayor coordenada y de una geometría.

- Si se desea más limpieza en la tabla de datos se pueden eliminar todas aquellas columnas que puedan haber regenerado las funciones usadas habilitando la edición y con la opción "Borrar campos" dentro de la tabla de atributos.

Finalmente, se han llevado a cabo las acciones necesarias para exportar los resultados al algoritmo de disposición automática de paneles solares FV de optimización energética. Para que los distintos rectángulos puedan ser leídos por el algoritmo de MATLAB primero se debe convertir la información a un formato compatible. Se ha escogido el formato XLSX o hoja de cálculo MS Office open XML por su facilidad de lectura y para visualizar resultados. Es importante guardar el archivo en la misma carpeta donde estén los códigos de MATLAB que componen el algoritmo para que este pueda leerlo con facilidad (sin tener que habilitar otras carpetas en el programa) y para que todos los recursos estén agrupados.

2.5 Disposición automatizada de paneles para optimización energética

En este quinto y último paso de la metodología propuesta se explicará el funcionamiento del algoritmo de MATLAB para el cálculo de la mejor distribución de paneles para una superficie rectangular dada, así como las opciones que toma en consideración. El objetivo último de este paso es determinar la configuración que más energía anual sería capaz de generar teniendo en cuenta una serie de parámetros que se detallarán en los próximos párrafos.

El criterio de partida para el código principal del algoritmo, llamado *"DisposicionOptimizadaCompleta",* es valorar si es mejor hacer un arreglo o array (conjunto de paneles solares FV) donde se dispongan los paneles solares FV de forma coplanar a la superficie o en caso contrario mediante el uso de soportes inclinados. Esta decisión se realiza según la pendiente con la que cuente la superficie, si esta es suficiente (en tejados inclinados) se dispondrán de forma coplanar, de no ser así se dispondrán con soportes inclinados. Para este trabajo se consideró que la inclinación mínima para plantear una configuración coplanar era de 12°, esto se decide así por las

recomendaciones de las normas de referencia UNE 136020-2004 y UNE 127100-1999 para el montaje de cubiertas con tejas.

El segundo criterio del código es el relativo a la orientación de los paneles solares. Las dos grandes configuraciones planteadas en este trabajo son:

1. **Configuración paralela al borde de la cubierta**: se trata de la configuración que, por la naturaleza de la geometría de la mayoría de los paneles comerciales, permite acomodar un mayor número de estos en el interior de las superficies rectangulares disponibles. Para estudiar esta configuración se desarrolló la función llamada "Sur" en MATLAB.

2. **Configuración orientada al sur**: se trata de la configuración que, por la posición geográfica que gozan las cubiertas y tejados en el hemisferio norte, resulta la que mayor generación de energía por kW de potencia instalada produce. Para estudiar esta configuración se desarrollaron dos funciones en MATLAB llamadas "Paralelo" y "ParaleloCoplanar".

El algoritmo no descartará una orientación sobre la otra hasta que haya comparado el mejor arreglo de cada una. Además, es posible elegir las pérdidas del sistema mediante un control deslizante numérico (Figura 8), entendiendo como pérdidas del sistema aquellas pérdidas que reducen la cantidad de energía entregada a la red eléctrica en comparación con la producida por los módulos fotovoltaicos de acuerdo con la ecuación(4.

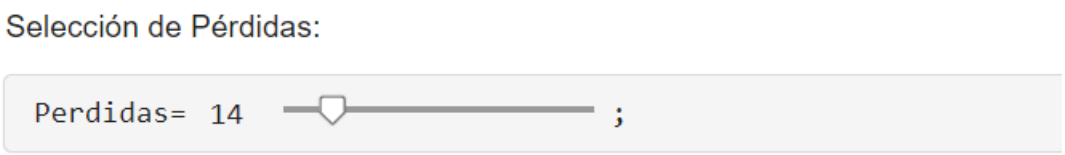


Figura 8. Control deslizante para selección de pérdidas

$$Energía = \frac{(100 - Perdidas)}{100} \times EnergíaProducida \qquad (4)$$

Las funciones "Sur" y "Paralelo" son llamadas por "DisposicionOptimizadaCompleta" si resulta que la pendiente de la superficie es muy baja como para plantear un arreglo coplanar. En caso contrario se llamará a la función "ParaleloCoplanar".

A continuación, se explicará el funcionamiento de las tres grandes funciones mencionadas.

2.5.1 *Funcionamiento de la función Sur*

La función "Sur" es la responsable de valorar todas las posibles configuraciones que resultan de combinar los parámetros:

- Tipo de panel: los paneles comerciales de identificarán por su potencia nominal, cada uno tiene asociado unas dimensiones y se permiten introducir 5 tipos de paneles distintos con sus dimensiones.

- Inclinación del panel: por defecto valora los ángulos 25°, 30°, 35°, 40° y 45°.
- Lado apoyado del panel: refiriéndose a si se apoya el panel por su lado corto o su lado largo (W y L respectivamente).
- Orientación: manteniendo una orientación alineada con el sur.

Su ejecución empieza llamando a su vez a la función “AlSur” la cual es la encargada de, dados los parámetros de la superficie, calcular el número de paneles de cada combinación que cabrían en la superficie de entrada. Para lograr esto primero dibuja cada cubierta con las dimensiones de las superficies obtenidas en el paso 2.4, se aprecia un ejemplo en la Figura 9.

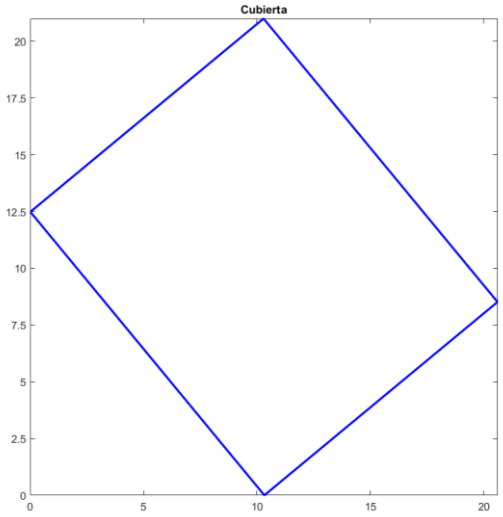


Figura 9. Ejemplo de representación de cubierta.

Es importante recalcar que lo que se ve en las figuras explicativas es una vista en planta de la disposición por lo que la dimensión del lado no apoyado del panel se ve modificada como se ve en la Figura 10.

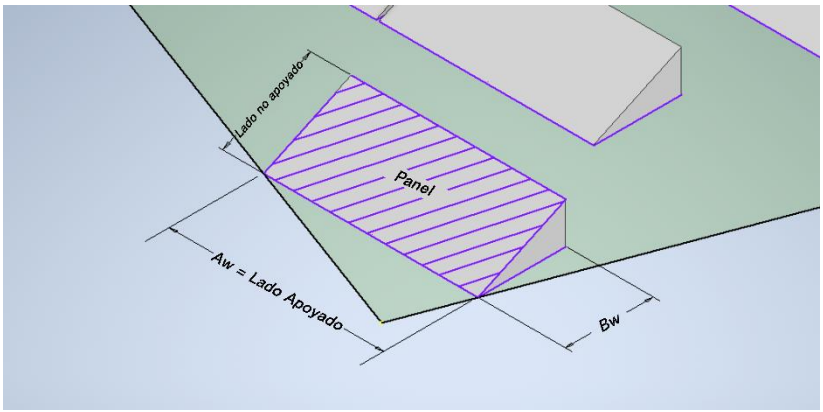


Figura 10. Representación de las dimensiones usadas en el algoritmo.

Siendo Aw igual a la dimensión del lado apoyado del panel y Bw la proyección de la dimensión del lado no apoyado la cual depende de la inclinación del panel.

$$Bw = Lado\ no\ apoyado\ \times \cos(\Theta) \quad (5)$$

Siendo:

Θ: la inclinación del panel (°).

En primer lugar, el algoritmo empieza a buscar la altura vertical "y" necesaria para que quepa un primer panel dentro del rectángulo de cubierta mediante iteraciones aumentando dicha altura hasta que la dimensión horizontal "h" sea mayor que el lado del panel (como se ve en la Figura 11). Las ecuaciones que rigen esta iteración son las siguientes:

$$c_1 = \frac{(y_i - P0_y)}{sen(Azimut)} \quad (6)$$

$$c_2 = \frac{(y_i - P0_y)}{sen(90° - Azimut)} \quad (7)$$

$$h = \sqrt{(c_1)^2 + (c_2)^2} \quad (8)$$

Las variables de las anteriores tres ecuaciones se representan en la Figura 11 (los subíndices x e y representan la coordenada del punto):

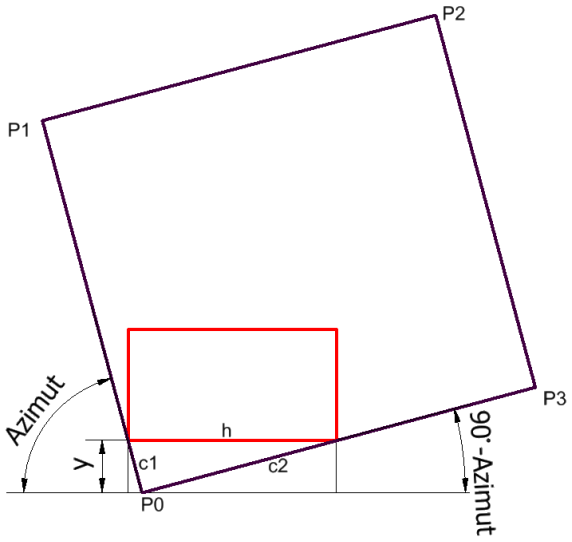


Figura 11. Representación de los parámetros para la colocación del primer panel.

Una vez que sabe la altura a la que debe colocar el panel, obtiene la coordenada horizontal restando desde el vértice 0 (P0):

$$Q1_x = P0_x - cosd(Azimut) \times c_1 \quad (9)$$

Siendo $Q1_x$ la coordenada x del vértice 1 del rectángulo del panel como se ve en la Figura 12.

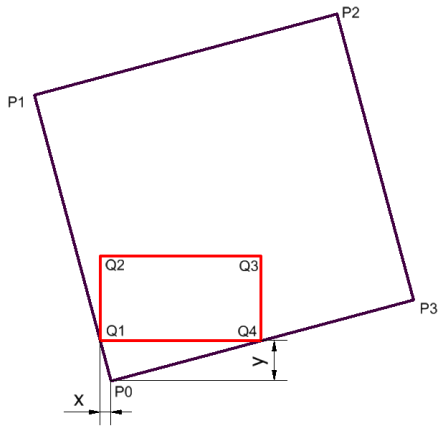


Figura 12. Numeración de los vértices del rectángulo del panel

También tiene en cuenta, por supuesto, que el panel no sobresalga por arriba. Esto gracias a que verifica que los vértices que forman el rectángulo del panel no tengan mayor coordenada “y” que las rectas que unen los vértices del rectángulo de la cubierta.

Si el espacio disponible es suficiente para colocar el panel, se pasa a disponer la siguiente hilera de paneles. Para ello, el algoritmo parte de la altura “y” encontrada para disponer el primer panel y suma la dimensión vertical del panel además de la separación necesaria para evitar que la hilera anterior proyecte sombra sobre la siguiente, esto de acuerdo con las recomendaciones del Instituto para la Diversificación y Ahorro de la Energía (IDAE)[25]. La expresión que recomienda se puede expresar de la siguiente forma:

$$d = \frac{h}{\tan\left(61° - latitud\right)} = h \times k \qquad (10)$$

Siendo:

d = separación entre filas para evitar sombras proyectadas (m).

h = diferencia de altura entre el objeto que proyecta sombra y el panel (m).

latitud = la latitud de la zona donde está ubicada la superficie (°).

k = factor adimensional que depende de la latitud.

La coordenada x del vértice 1 del primer panel de las hileras siguientes se obtiene igual que el primer panel de la disposición con la ecuación(6 y(9 pero con la nueva “y”. Teniendo el primer panel colocado, el algoritmo procede a disponer tantos paneles como pueda a su derecha, manteniendo una separación entre columnas de 2 cm para permitir un mínimo paso de aire entre paneles (distancia “ed”), hasta que las dimensiones de la cubierta no permitan añadir otro. De este modo el algoritmo va recorriendo el borde inferior de la superficie hasta que una hilera sobrepase supere el borde superior del panel (Figura 13).

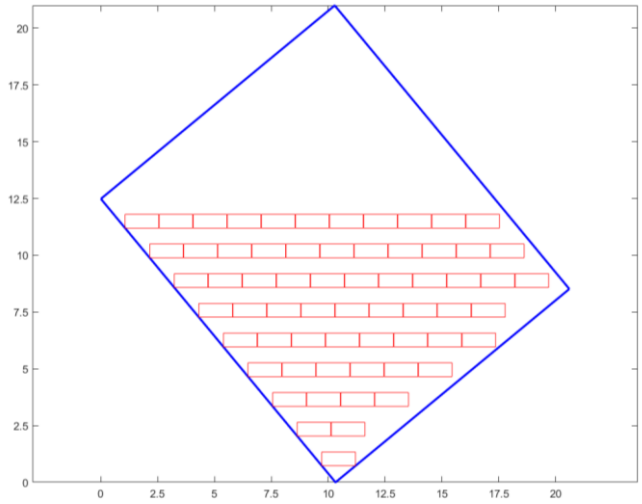


Figura 13. Ejemplo de disposición de paneles inferiores orientados al sur.

En ese punto la función “AlSur” procede a disponer los paneles superiores. La coordenada “y” de la primera hilera superior se calcula sumando la separación por sombra (ecuación(10) como se ha hecho hasta ahora para las hileras inferiores, pero hallar la coordenada “x” para disponer el primer panel de las hileras superiores requiere de una ecuación más complicada.

Esta nueva ecuación parte de las coordenadas del vértice 2 del rectángulo de cubierta. Para llegar a la coordenada x del vértice apoyado del primer panel superior es necesario calcular la variable “ys” con la siguiente expresión:

$$y_s = P2_y - y - Bw \qquad (11)$$

De este modo la variable x del vértice apoyado del primer panel superior (S2x) se puede obtener aplicando ecuaciones trigonométricas, quedando la expresión de la siguiente forma:

$$S2_x = S1_x = P2_x - tan(Azimut) \times y_s \qquad (12)$$

Estas variables se ven representadas en la Figura 14:

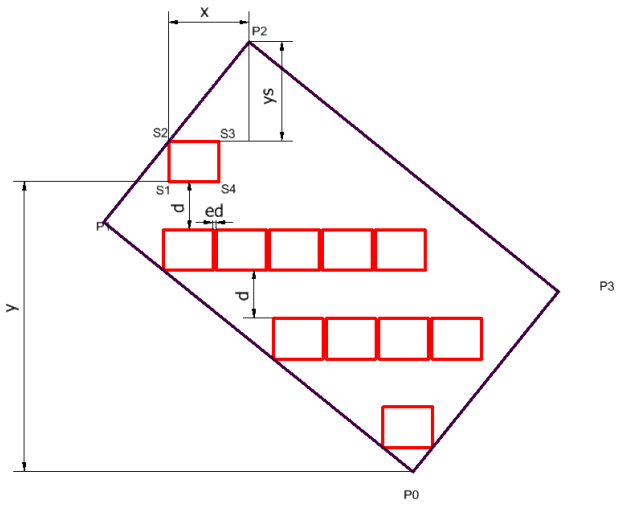


Figura 14. Representación de los parámetros para la colocación de los paneles superiores

De este modo y respetando la separación, el algoritmo recorre lo que queda de superficie hasta disponer todos los paneles faltantes sin sobresalir por la parte superior del rectángulo. Esto último de nuevo gracias a las restricciones impuestas referentes a que los vértices de los paneles deben tener coordenadas inferiores a las rectas que unes los vértices del rectángulo de la cubierta. Una disposición al sur completa queda como en la Figura 15.

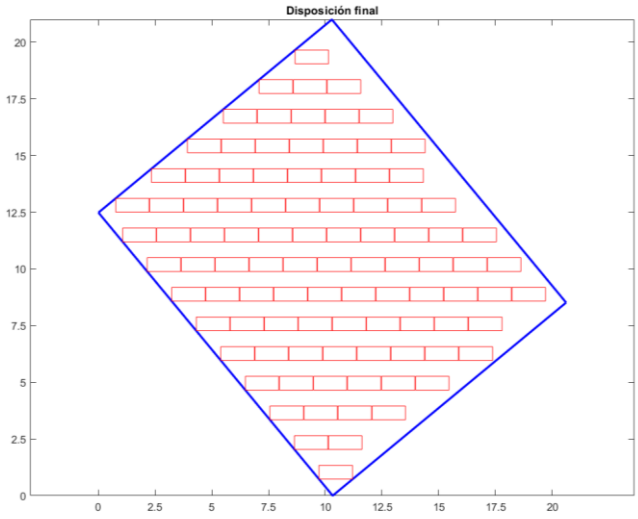


Figura 15. Ejemplo de disposición orientada al sur completa.

Mediante un contador, la función AlSur almacena el número de paneles que cabrían en cada configuración y lo multiplica por la potencia nominal del panel de la configuración, obteniendo así la potencia total instalable del array.

Una vez la función AlSur devuelve a la función Sur la potencia total de cada combinación de parámetros, esta última empieza a filtrar dichas combinaciones para quedarse con aquella que genera la mayor energía anual. Para ello hace uso de la función Optimizacion1 la cual determina el apoyo (lado largo o corto) que más potencia total instalable permite obtener para cada tipo de panel e inclinación de este, pasando de tener 50 configuraciones a 25. A continuación, llama a la función Optimización2 para guardar el arreglo con mayor potencia instalable para cada tipo de inclinación, de este modo pasa de 25 a 5 configuraciones y finalmente se convierte a energía multiplicando la potencia total instalable en kW por la energía asociada a cada inclinación de panel, es decir, la energía que se produciría al año por cada kW de potencia instalada según la inclinación del panel en kWh/(kWPotencia_Instalada*año). Esta energía se estima mediante una base de datos creada a partir de la herramienta web PVGIS[2] [26]. De este modo se puede hacer una comparación exacta y conservar solo el arreglo que mayor energía sería capaz de generar. Tras esta última comparación, Sur devuelve su resultado a DisposicionOptimizadaCompleta.

2.5.2 *Funcionamiento de la función Paralelo*

La función "Paralelo" es la responsable de valorar todas las posibles configuraciones que resultan de combinar los parámetros:

- Tipo de panel: los paneles comerciales de identificarán por su potencia nominal, cada uno tiene asociado unas dimensiones y se permiten introducir 5 tipos de paneles distintos con sus dimensiones.
- Inclinación del panel: por defecto valora los ángulos 25°, 30°, 35°, 40° y 45°.
- Lado apoyado del panel: refiriéndose a si se apoya el panel por su lado corto o su lado largo sobre la cubierta (W y L respectivamente).
- Orientación: alineada con el borde de la cubierta. Esto da pie a la posibilidad de apoyar los paneles de forma paralela a dos caras de la cubierta o lo que es lo mismo, con un Azimut de α o β como se ve en la Figura 16.

[2] PVGIS (Photovoltaic Geographical Information System) es una herramienta desarrollada por la Comisión Europea que proporciona datos y recursos relacionados con la energía solar fotovoltaica.

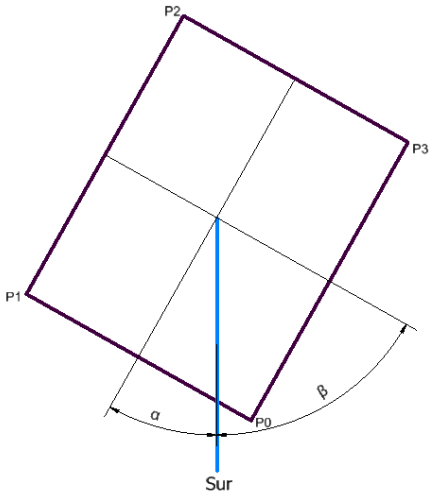


Figura 16. Representación de los ángulos α y β.

El ángulo α o ángulo de medianera se define como el ángulo más pequeño entre la cubierta y el sur geográfico y β como el ángulo complementario. Es importante recordar que para este trabajo un ángulo de 0° corresponde con totalmente alineado con el sur, 90° totalmente al oeste y -90° totalmente orientado al este. De modo que el ángulo complementario β se obtiene como resultado de sumar 90° al ángulo de medianera si este es negativo o restar 90° si es positivo.

La ejecución de la función Paralelo empieza llamando a la función "potenciaTotal" la cual es la encargada de, dados los parámetros de la superficie, calcular el número de paneles de cada combinación que cabrían en la superficie de entrada y de hallar la potencia total instalable de dichos paneles. Para lograr este objetivo, el algoritmo primero estudia la configuración con ángulo de Azimut igual a α:

Por un lado, itera aumentando el número de columnas mientras que la longitud total en dirección del apoyo llamada Awtotal sea menor al largo del rectángulo de cubierta:

$$AwTotal = ncolumnas \times DimensionPanel + ed \times (\, ncolumnas \, - 1) \qquad (13)$$

A su vez itera aumentando el número de filas mientras que la longitud total en dirección del lado no apoyado (BwTotal) sea menor al ancho del rectángulo de cubierta:

$$BwTotal = nfilas \times DimensionPanel \times \cos(\theta) + d \times (nfilas - 1) \qquad (14)$$

Siendo DimensionPanel la dimensión del lado apoyado del panel. Si se está estudiando la configuración con el lado corto su valor corresponde al ancho del panel y si se está estudiando el lado largo corresponde con el largo del panel. La Figura 17 muestra estos parámetros gráficamente:

Figura 17. Representación de los parámetros necesarios para la disposición paralela a la cubierta.

Posteriormente el algoritmo estudia la configuración con ángulo de Azimut igual a β la cual se rige con las mismas ecuaciones que la configuración α (ecuaciones (13 y (14) pero comparando la longitud total en dirección del apoyo "Awtotal" con el ancho de la cubierta y la longitud total en dirección del lado no apoyado "Bwtotal" con el largo de la cubierta, es decir, invirtiéndolas.

De este modo, se obtiene el número de filas y columnas para cada una de las 100 combinaciones posibles. Al multiplicar dichos valores entre ellos y por la potencia nominal del panel se obtiene la potencia total de cada arreglo.

A continuación, la función Paralelo comienza a filtrar los arreglos para conservar el que mayor energía anual sería capaz de generar. Para ello, igual que la función Sur, hace uso de la función Optimizacion1 la cual determina el apoyo (lado largo o corto) que más potencia total permite obtener para cada tipo de panel e inclinación de este, pasando de tener 100 configuraciones a 50. A continuación, llama a la función Optimización2 para guardar el arreglo con mayor potencia para cada tipo de inclinación, de este modo pasa de 50 a 10 configuraciones. En tercer lugar, hace uso de la función Optimizacion3 para hallar la energía que generaría cada configuración restante teniendo en cuenta su inclinación de panel y su ángulo de azimut. Esta energía se estima mediante la base de datos obtenida de PVGIS e interpolando en un polinomio de orden 4. De ese modo compara los arreglos orientados con azimut α y β consiguiendo la mejor configuración para cada inclinación.

Gracias a este proceso se puede hacer una comparación exacta y conservar solo el arreglo que mayor energía sería capaz de generar. Tras esta última comparación, la función Paralelo devuelve su resultado a DisposicionOptimizadaCompleta.

2.5.3 *Funcionamiento de la función ParaleloCoplanar*

La función "ParaleloCoplanar" es la equivalente de la función Paralelo para arreglos coplanares a la superficie disponible. Es llamada por DisposicionOptimizadaCompleta cuando la pendiente de la superficie disponible es suficiente para un arreglo coplanar. Su funcionamiento resulta muy similar a su contraparte con soportes, pero presenta una serie de diferencias en su funcionamiento los cuales se comentarán a continuación.

Por un lado, al disponer los paneles de forma coplanar a la superficie ya no se valoran distintas inclinaciones de panel, esta viene restringida por la propia inclinación de la cubierta. Sin embargo, aún valora las combinaciones de cada tipo de panel y del lado sobre el que esté apoyado. Otra diferencia es que ya no hay distinción entre un arreglo orientado con azimut igual al ángulo de medianera α y uno con el ángulo complementario β.

La ejecución de la función ParaleloCoplanar empieza llamando a la función "potenciaTotalCoplanar" la cual es la encargada de, dados los parámetros de la superficie, calcular el número de paneles de cada combinación que cabrían en la superficie de entrada y calcular la potencia total que generarían esos paneles. Su funcionamiento es similar a la función potenciaTotal ya mencionada, pero está adaptada a las restricciones de pendiente y ajusta las separaciones entre filas debidas a las sombras considerando solo el espesor de los paneles como se vio en la ecuación(15.

$$h = \frac{ProfundidadPanel}{sen(90° - PendienteCubierta)} = \frac{ProfundidadPanel}{cos(PendienteCubierta)} \qquad (15)$$

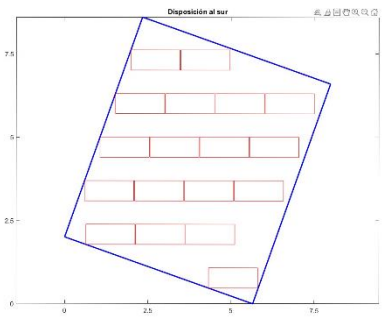
Esta h es la diferencia de alturas que se introduce en la ecuación(10 para los casos coplanares.

Como se adelantó, al no tener que comparar distintas inclinaciones de panel, no puede usar la función optimizacion2 en su lugar hace uso de la función optimización2Coplanar para elegir la mayor potencia total para posteriormente calcular la energía que dicho arreglo produciría (de nuevo con la base de datos de PVGIS). Tampoco es necesario hacer uso de la función optimizacion3 ya que todas las configuraciones tienen las misma pendiente y orientación (por ende, su energía también será proporcional a su potencia instalada). Al final de su ejecución determina el arreglo con la mayor potencia instalada y calcula la energía que sería capaz de generar teniendo en cuenta su azimut y pendiente.

Del mismo modo que las funciones Sur y Paralelo devuelve su resultado a DisposicionOptimizadaCompleta.

Tras recibir todos los resultados de las funciones llamadas, DisposicionOptimizadaCompleta muestra al usuario los resultados de ambos estudios y destaca el que más energía anual en kWh/año sería capaz de producir como se ve en la Figura 18. Ejemplo de una salida del código DisposicionOptimizadaCompleta.

Como comentario adicional, la creación de una función "SurCoplanar" para estudiar disposiciones donde los paneles se dispongan de forma coplanar a la superficie y además orientados al sur no tendría sentido, ya que, la orientación de los arreglos coplanares está condicionada a la orientación de la superficie (al no usar soportes). En otras palabras, sería igual que los arreglos de ParaleloCoplanar a nivel de generación de energía por panel, pero logrando disponer de menos paneles en total lo cual va en contra del objetivo de maximizar la energía total generada.



Cubierta 3.

Mejor_arregloParalelo = 1×6 table

	Mayor Pot (kW)	Energía (kWh/año)	Conf	Pot panel (W)	Apoyo	Ángulo
1	5.3600	8.2917e+03	α	670	W	25

Mejor_arregloSur = 1×5 table

	Mayor Pot (kW)	Energía (kWh/año)	Pot panel (W)	Apoyo	Ángulo
1	3.6000	5.6084e+03	200	L	25

Es preferible utilizar soportes para apoyar los paneles.

La mayor Energía total anual para los parametros planteados con orientación paralela a la cubierta es:
8291.66 kWh/año con una potencia instalada de 5.36 kW y usando 8 paneles de 670 W.

La mayor Energía total anual para los parametros planteados con orientación al sur es:
5608.43 kWh/año con una potencia instalada de 3.60 kW y usando 18 paneles de 200 W.

La mayor Energía total anual se dará con orientación paralela a la cubierta.

Figura 18. Ejemplo de una salida del código DisposicionOptimizadaCompleta.

En la Figura 18 se aprecia un ejemplo de la salida del código DisposicionOptimizadaCompleta, este ejemplo en particular es de un caso donde la pendiente de la superficie rectangular es menor a 12º por lo que el algoritmo estudió los arreglos con soportes llamando a las funciones “Paralelo” y “Sur”. En las tablas devueltas por las funciones se aprecian los datos del mejor arreglo (el que más energía generaría) de ambas orientaciones:

- Mayor Pot: Es la potencia instalada en kW del arreglo que mayor energía genera.
- Energía: Energía del mejor arreglo en kWh/año.
- Conf: Orientación del arreglo para el caso paralelo a los bordes de la cubierta (α o β).
- Pot Panel: Potencia del panel solar FV usado en el mejor arreglo en W.
- Apoyo: Lado del panel apoyado sobre la superficie (W o L) en el mejor arreglo.
- Ángulo: Inclinación del panel en grados del mejor arreglo.

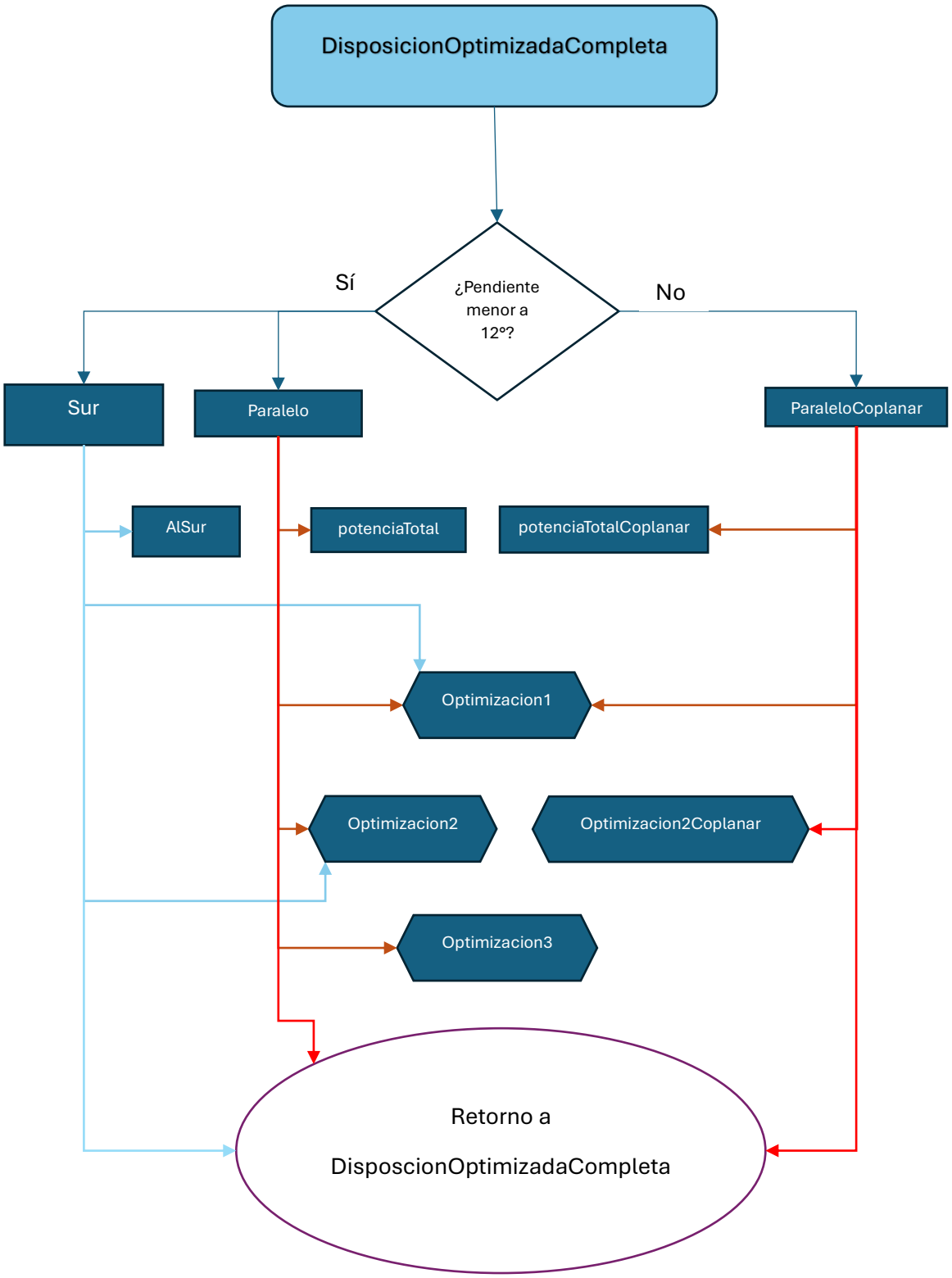


Figura 19. Diagrama de funciones del algoritmo de disposición de paneles

Capítulo 3. Caso de estudio: Illa Perduda, Valencia

En este capítulo se presentará el caso de estudio de este trabajo, así como la ejecución de la metodología propuesta en él. Para este trabajo se eligió el barrio de Illa Perduda localizado en la ciudad de Valencia como caso de estudio.

La ciudad de Valencia es una ciudad con clima mediterráneo caracterizada por veranos cálidos e inviernos apacibles. La ciudad es bien conocida por ser una de las tres ciudades más grandes de España y por tener ambiciosas estrategias de sostenibilidad lo que le valió el título de Capital Verde Europea 2024. L'Illa Perduda es uno de los 87 barrios de Valencia, ubicado en la zona Este de la ciudad (ver Figura 20).

Figura 20. Mapa resaltando el Barrio de Illa Perduda.

3.1 Eliminación de obstáculos

En la siguiente sección se procederá de acuerdo con la metodología propuesta en el punto 2.1.

Las ortofotos del PNOA (Plan Nacional de Ortofotografía Aérea) empleadas fueron obtenidas del centro de descargas del CNIG en el apartado de "Fotos e imágenes aéreas". En dicho sitio se dividen las zonas por cuadrantes y los relativos a la ciudad de Valencia son los llamados 0696-3, 0696-4, 0722-1 y 0722-2 como se ve en la Figura 21. La zona de estudio del trabajo, Illa Perduda, pertenece al cuadrante 0722-2.

Figura 21. Ciudad de Valencia dividida en cuadrantes.

Para trabajar exclusivamente con el barrio seleccionado, se procede a recortar la imagen TIF de alta calidad del cuadrante 0722-2 usando como capa de máscara la capa "barris-barrios.geojson" obtenida del portal del ajuntamiento de valencia en la sección de "Urbanismo e Infraestructura" y "Datos abiertos"[27]. El barrio de Illa Perduda extraído del conjunto de barrios se ve en la Figura 22 mientras que el recorte obtenido con dicha capa se ve en la Figura 23.

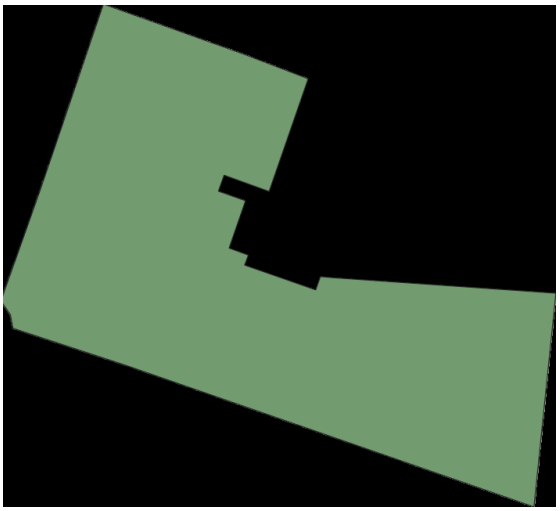


Figura 22. Capa vectorial del barrio de Illa Perduda.

Figura 23. Barrio de Illa Perduda.

Como se puede apreciar el barrio cuenta con tejados en su gran mayoría con tonos rojizos, anaranjados y grisáceos. Esos colores serán los que se seleccionarán en el código de MATLAB MyRoofTypeSet. Para el funcionamiento de este, basta con facilitarle la imagen TIF del barrio completo o un fragmento de este que se considere cuente con toda la variedad de colores de tejados del barrio para a continuación seleccionar pixeles de la imagen a modo de muestras para que el código pueda obtener una media de dichas muestras y clasificarlas. Para el color rojizo, anaranjado y gris se obtuvieron unas medias RGB de (182.42, 122.60, 111.86), (198.77, 169.62, 149.56) y (177.76, 176.69, 167.86) respectivamente.

Con esta información se procede a detectar los colores seleccionados en la imagen completa mediante el código MyRoofTypeDetection.m.

Un problema que surge es que las cubiertas grises comparten un color muy similar a las calles y para la visión artificial no es posible distinguirlos. Por ello es recomendable hacer previamente un segundo recorte usando esta vez la información catastral de los edificios del barrio para eliminar las calles. Dicha información se puede obtener mediante el complemento de QGIS “Spanish Inspire Catastral Downloader” el cual permite obtener, entre otras cosas, una capa vectorial con todos los edificios de una provincia y municipio seleccionados. Dicho complemento obtiene sus datos del sitio oficial de la Sede Electrónica del Gobierno de España [28].

La idea a continuación es recortar la capa de edificios, llamada A.ES.SDGC.BU.46900.building o buildings para abreviar, de modo que se ajuste a solo los edificios ubicados en el barrio. Dicha operación es un simple recorte vectorial (como se explica en la metodología) usando como capa de entrada la capa el barrio extraído de barris-barrios.geojson.

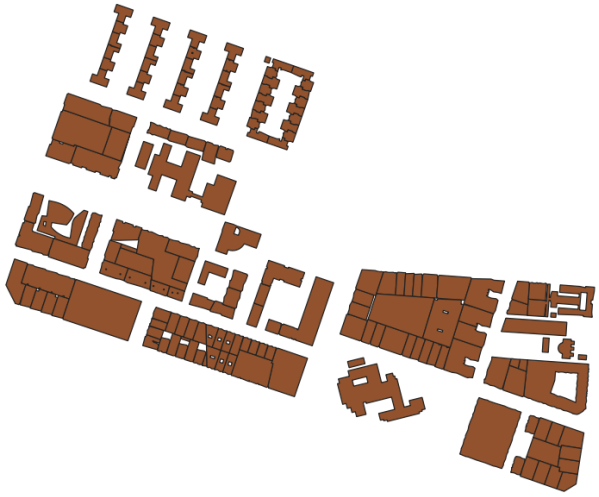


Figura 24. Edificios de Illa Perduda obtenidos del catastro.

Para obtener una capa TIF recortada conteniendo únicamente los edificios del barrio, bastaría repetir el mismo corte por capa de máscara usado para obtener la capa de la Figura 23 pero usando como máscara la capa buildings de la Figura 24 en vez de la capa del barrio de la Figura 22. Un problema que surge al intentar realizar esta operación directamente es que algunos vértices de los polígonos que componen los edificios del catastro se encuentran mal ubicados de modo que los polígonos se solapan impidiendo realizar el corte. Un ejemplo de este inconveniente se puede apreciar en la Figura 25.

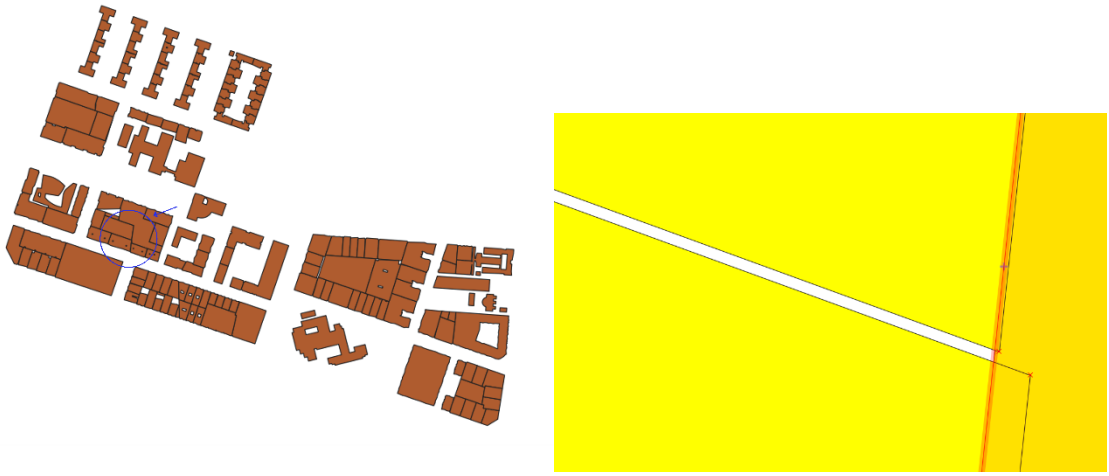


Figura 25. Vértices de edificios solapados.

Se cree que este error pueda ser debido a la incorporación de edificios nuevos al registro catastral existente de forma apresurada, a divisiones de una propiedad o a modificaciones de estas. Estos errores se pueden reparar usando la herramienta de vértices de QGIS y la función Corregir geometrías (lo cual crea una versión válida de las geometrías existentes sin modificar el número de vértices de estas). Finalmente, pese al contratiempo, se obtiene la imagen TIF del barrio sin calles (Figura 26).

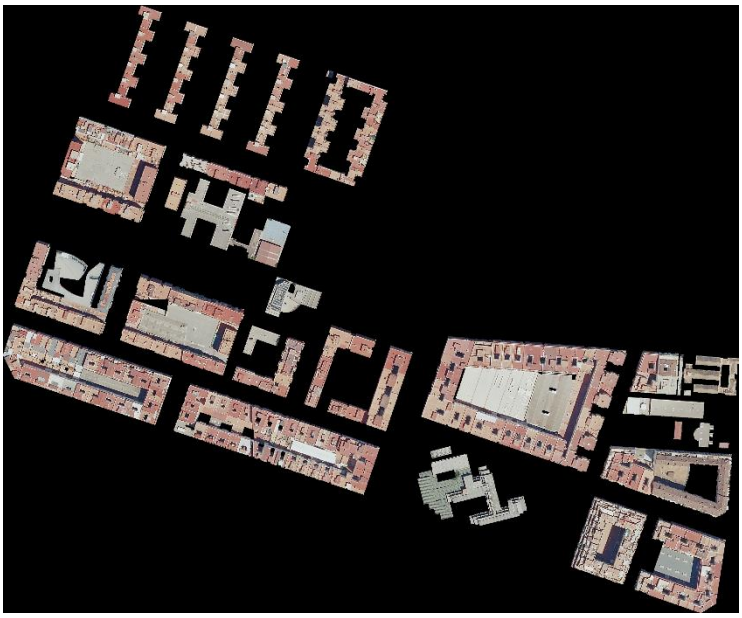


Figura 26. Illa Perduda sin calles.

Contando finalmente con la capa de entrada del código se procede a introducirla en MyRoofTypeDetection.m donde el código detectará todos los pixeles que formen parte de una región de colores creando una máscara binaria para cada una y posteriormente combinándolas.

Por defecto considera que un píxel pertenece a una región de color si sus valores RGB o HSV están incluidos dentro de un intervalo definido por el mínimo y máximo valor de los pixeles seleccionados o por aquellos que cumplan estar en el intervalo de tolerancia:

$$media \pm 3 \times desviacion\ estandar$$

Se ese modo se alcanza un resultado más preciso. Cabe recordar que tras este paso el código da la opción al usuario de añadir o suprimir mediante polígonos las zonas que considere para un resultado más próximo al deseado. Para el caso de estudio se consiguió la máscara binaria mostrada en la Figura 27 la cual cuenta con una superficie disponible de 52.782,6 m^2. Aunque tras vectorizarla y aplicar un pequeño filtro de área mínima de 10m^2 para eliminar pequeños puntos aislados que se hayan podido detectar, la superficie disponible queda en 51.142,3 m^2.

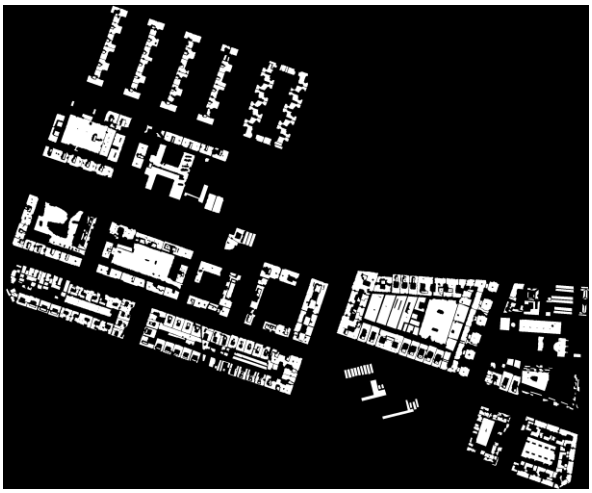


Figura 27. Máscara binaria sin obstáculos.

3.2 Cálculo de la pendiente y orientación de las superficies

En la siguiente sección se procederá de acuerdo con la metodología propuesta en el punto 2.2.

En primer lugar, fue necesario conseguir una capa vectorial con las divisiones por partes del barrio, esta capa se consigue a la vez que la capa buildings usada en el apartado 3.1 por medio del complemento Spanish Inspire Catastral Downloader. Una vez obtenida la capa A.ES.SDGC.BU.46900.buildingpart o buildingparts para abreviar, al igual que con la capa buildings, se procedió a recortarla para conservar solo los edificios (con sus partes) del barrio de Illa Perduda como se aprecia en la Figura 28.

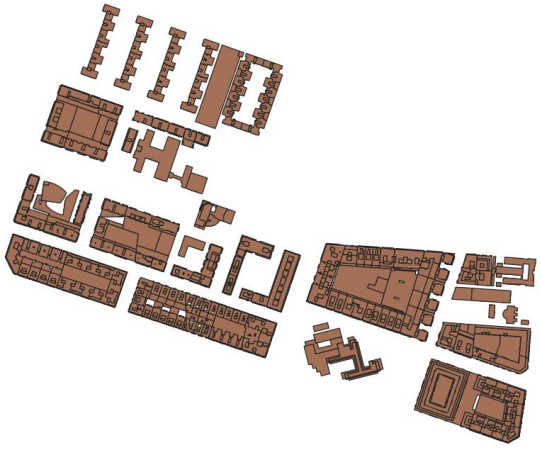


Figura 28. Partes de los edificios de Illa Perduda.

Por otro lado, los archivos de puntos LAZ (formato de compresión de ficheros LAS) se descargarán del centro de descargas del CNIG en el apartado de “Modelos digitales de elevaciones”. Se usarán los archivos más actuales disponibles (cobertura 2015-2021). Se descargaron los siguientes dos archivos:

- PNOA-2015-VAL-728-4372-ORT-CLA-RGB.LAZ
- PNOA-2015-VAL-728-4374-ORT-CLA-RGB.LAZ

Esto ya que el barrio de Illa Perduda se encontraba entre las áreas comprendidas por ambos. Por lo tanto, es necesario combinar ambos archivos mediante la herramienta "*lasmerge*" del complemento "*LAStools*" de QGIS y posteriormente recortar el resultado para conservar solo el área correspondiente al barrio, quedando como la Figura 29 y finalmente filtrar dicha nube de puntos para eliminar aquellos que no correspondan a edificios (esto aprovechando la clasificación interna que poseen los puntos como se explica en la metodología). Este paso optimiza los tiempos de carga y mejora el resultado final.

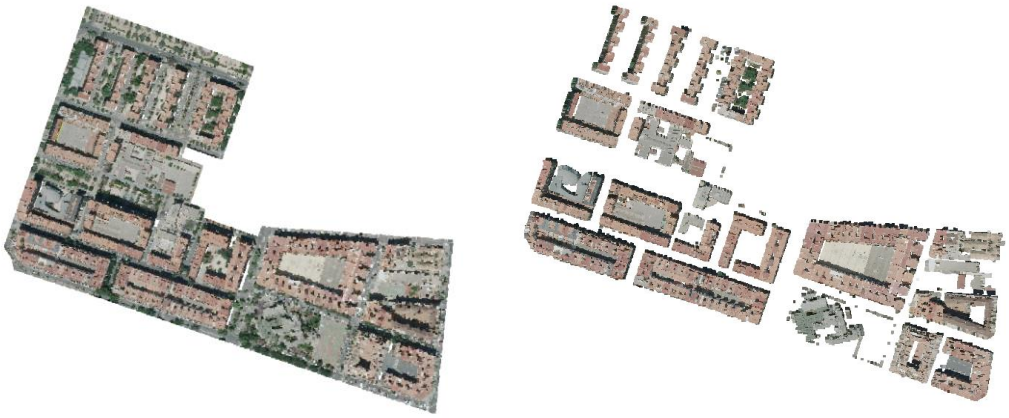


Figura 29. Nube de puntos de Illa Perduda antes y después del filtrado.

Los puntos aislados que se aprecian han sido eliminados en gran parte filtrando adicionalmente por una altura mínima ya que se trata de puntos del suelo con una clasificación errónea. Sin embargo, no causarán ningún problema en la operación siguiente ya que esta consiste en crear nuevos polígonos agrupando puntos de la nube de puntos tomando como referencia la capa buildingparts, y al no haber edificios en esa zona no se asignarán a ninguna superficie y posteriormente serán eliminados.

La operación de asignado de puntos a superficies se realizó mediante la función *LidarRoofTopAnalysis* perteneciente al complemento WhiteBoxTools introduciendo la capa de nube de puntos recortada y la capa buildingparts. El resultado se filtra por un área mínima de 2 m^2 para eliminar puntos que no se pudieron asignar a superficies y se reparan las geometrías inválidas con la función *Corregir geometrías* vista anteriormente. En este punto el resultado es el que se aprecia en la Figura 30.

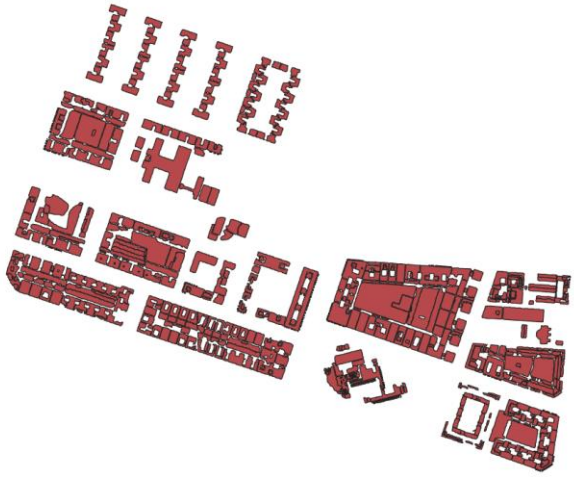


Figura 30. Partes de edificios con pendiente y orientación.

El uso de nubes de puntos permitió a la función *LidarRoofTopAnalysis* crear nuevos polígonos, los cuales cuentan con la información de su pendiente y orientación en su tabla de atributos gracias a que ajusta un plano a cada superficie utilizando un modelo de regresión.

Para decidir qué superficies son aptas no se dispone de las características estructurales reales de las cubiertas de los edificios. Se puede estimar su sobrecarga de uso y otras variables a partir del proyecto TABULA-EPISCOPE [29]. Se trata de un estudio muy importante realizado a nivel europeo, que caracteriza estructuralmente los edificios según la región, la relación altura/envolvente, el año de construcción, etc. Según este proyecto, los edificios del barrio de Illa Perduda son en su mayoría edificios plurifamiliares de 4 o más plantas, construidos en el periodo 1960-1979 (125 edificios) y 1980-2000 (25 edificios). Todos estos edificios tienen cubiertas transitables con una inclinación inferior a 20° y una sobrecarga de hasta 500 kg/m². En otras palabras, pueden soportar fácilmente las cargas de peso de los paneles fotovoltaicos (aproximadamente 20 kg/m2 con soportes). Por tanto, se han considerado como superficies aptas todas las cubiertas transitables y como no aptas las no transitables: cubiertas de panel sándwich, de paneles de fibra de cemento, etc.

3.3 Cálculo de la Irradiación

En la siguiente sección se procederá de acuerdo con la metodología vista en el punto 2.3.

Para obtener la capa vectorial correspondiente a los puntos del barrio que cuenten con una irradiación suficiente se partió de la nube de puntos recortada al barrio de Illa Perduda que se aprecia en la Figura 29 (no es necesario que esté filtrada por edificios) y con ella se generó el modelo digital de elevaciones, o DEM por sus siglas en inglés (Figura 31), gracias a de la función "*blast2dem*" del complemento *LAStools*.

Figura 31. DEM de Illa Perduda.

Contando con el modelo digital de elevaciones es relativamente sencillo obtener el modelo digital de aspectos (orientaciones) y el modelo digital de pendientes mediante las opciones de análisis ráster de QGIS. Estos dos modelos se muestran a continuación:

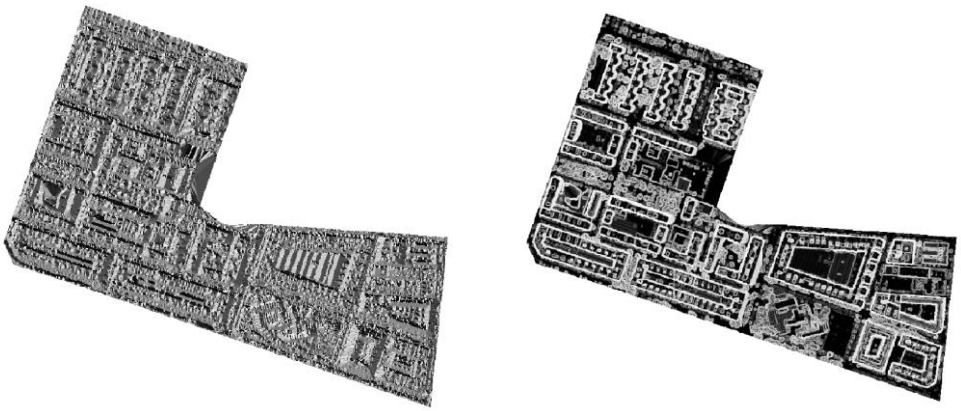


Figura 32. Modelo digital de orientaciones y modelo digital de pendientes.

Contando con los tres modelos, se usó la función "*r.sun.insoltime*" del complemento GRASS para generar una capa raster con la irradiación de cada punto de la zona de estudio. Como ya se comentó en la metodología, esta función genera un modelo de irradiación solo para el día del año seleccionado lo que es un problema ya que se busca hacer un estudio de la irradiación anual en la zona. Para lograr este objetivo, se seleccionó solo opción de radiación total (suma de radiación difusa, directa y reflejada) se copió el código de Python mediante la opción avanzada que ofrece *r.sun.insoltime* y se modificó para que mediante un bucle generase el modelo de irradiación de cada día del año (dicho bucle se puede consultar en el ANEXO 1. Códigos de QGIS).

Tras haber generado cada modelo se combinaron y mediaron dando como resultado un modelo cuyos puntos contaban con la información del valor medio de la irradiación media anual en kWh/día*año. En la Figura 33 se ve dicho modelo, cuanto más claro el color en la escala de grises mayor la irradiación anual de dicho punto.

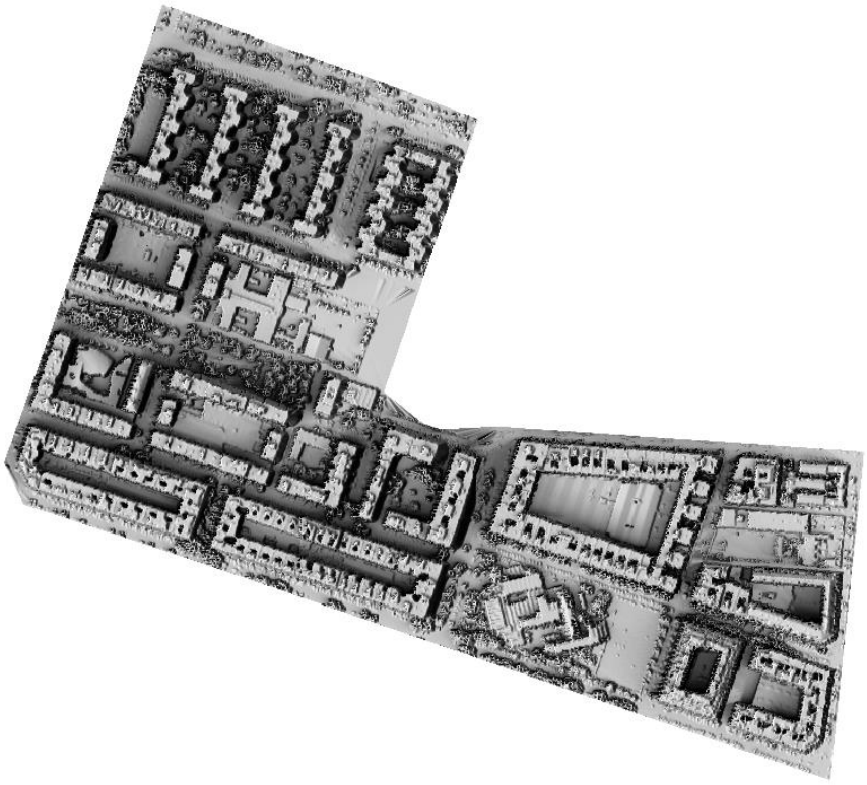


Figura 33. Irradiación global anual en Illa Perduda.

Los puntos de dicho modelo cuentan con valores entre los 462 y 7125 Wh/(m^2*día). Para este trabajo se tomó como valor mínimo 1000 kWh/(m^2*año), o lo que es lo mismo, 2740 Wh/(m^2*día). Es decir, se trabaja con aquellos puntos con una irradiación mayor a ese umbral. Esto de acuerdo con la recomendación del IDAE en su evaluación del potencial de energía solar y fotovoltaica derivado del cumplimiento del código técnico de edificación [30].

Aplicando este mínimo en la calculadora ráster se obtiene la capa binaria de la Figura 34, donde los puntos en negro son los que no cumplen con el mínimo de irradiación.

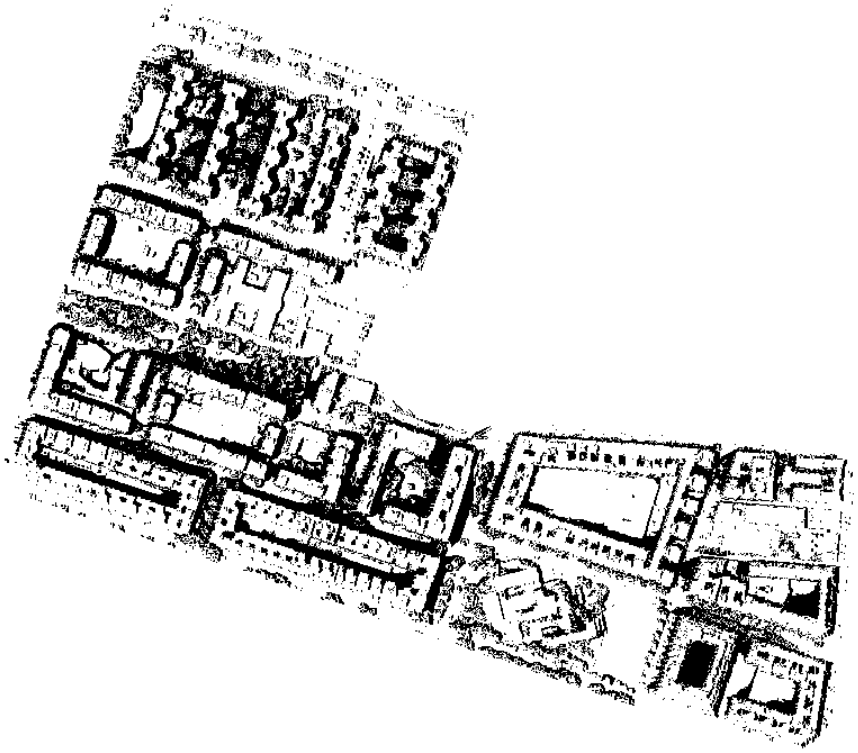


Figura 34. Mapa de Irradiación filtrado.

Para poner usar dicha capa para restringir las superficies de estudio fue necesario vectorizar tanto esta capa como la capa resultante del paso 3.1 con la herramienta *Ráster > Conversión > Poligonizar (ráster a Vectorial)*.

3.4 Unión de las capas y preparación de las superficies

En la siguiente sección se procederá de acuerdo con la metodología vista en el punto 2.4.

Siguiendo el orden de la metodología planteada, el primer paso es usar la información de la capa de pendientes y orientaciones obtenida en el punto 3.2 para restringir las superficies disponibles en la máscara binaria sin obstáculos obtenida en el punto 3.1. Para ello es necesario pasar la información de la capa de pendientes y orientaciones (llamada simplemente Pendiente en la metodología) a la capa sin obstáculos (llamada MaskGeo en la metodología).

Para ello se usó la función unir atributos por localización indicando unir atributos con traslape más grande solamente. Esta opción hace que la información de cada superficie de la capa Pendiente se traspase a la superficie que más se asemeje a ella en la capa Maskgeo, pero no modifica su apariencia.

A continuación, ya contando con la información en la capa deseada, se procede a filtrarla de modo que solo se conservarán las superficies planas o aquellas inclinadas, pero con una orientación adecuada. Para este caso de estudio se consideró plana a toda superficie con pendiente entre 0° y 5°, esto atendiendo a las consideraciones para la seguridad de utilización y accesibilidad del Código Técnico de la Edificación[31]. Mientras que aquellas con orientación adecuada serian aquellas con una orientación entre el Este y el Suroeste o lo que es lo mismo este 90° y 225°. Esto para favorecer las superficies entorno a la orientación Sur la cual es la óptima para el hemisferio norte [16]. Sin

embargo, se optó por relajar esta restricción a 67,5° y 247,5° para incluir al menos la mitad de la circunferencia (180° en total) como se ve en la Figura 35.

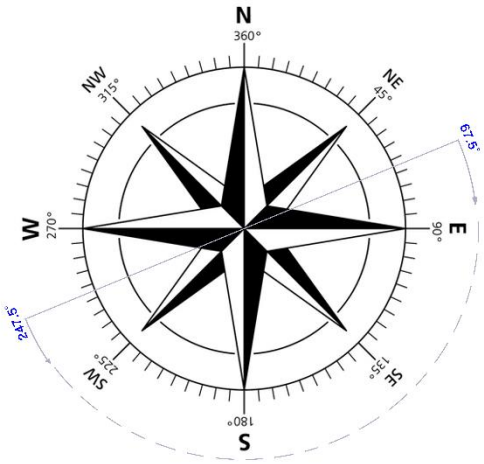


Figura 35. representación del intervalo de orientaciones aceptado.

Tras aplicar la expresión de filtrado en el constructor de consultas de la capa Union_MASK_Pend se obtiene una capa compuesta de superficies libres de obstáculos y con pendientes y orientaciones adecuadas (Figura 36). En este punto el área total disponible se redujo de 51.142,3 m^2 a 49.607,5 m^2.

Figura 36. Capa sin obstáculos con pendiente y orientaciones adecuadas.

Este paso resulta especialmente útil para eliminar cubiertas a dos aguas mal orientadas, para apreciarlo mejor se muestra a continuación la Figura 36 pero con un aumento para apreciar un ejemplo de esas superficies eliminadas (en verde las superficies eliminadas respecto al paso anterior):

Figura 37. Ejemplo de superficies a dos aguas eliminadas por mala orientación.

El paso siguiente es restringir las superficies actuales por irradiación mínima. Esto, con la finalidad de eliminar zonas en sombra o que por obstáculos u otros edificios no reciban suficiente radiación solar al año. Para conseguirlo, se recortó la capa Union_MASK_Pend ya filtrada con la capa de

irradiación obtenida en el punto 3.3 (se referirá a esta capa como IrradiacionFinal como en la metodología). Tras realizar el corte vectorial de modo que solo las zonas con irradiación superior a la mínima queden se obtiene las superficies tal como se muestra en la Figura 38 (en rosa las superficies eliminadas de la etapa anterior) esta capa es la llamada Union_MASK_Pend_RadAnual.

Figura 38. Superficies resultantes de aplicar el filtro de irradiación.

Finalmente, solo resta asemejar las superficies resultantes para poder exportarlas al algoritmo de disposición de paneles solares fotovoltaicos de MATLAB. Este proceso empieza generando cuadrículas sobre nuestra zona de estudio con la herramienta crear cuadrícula. Estas cuadriculas se pueden girar para adecuarse a la orientación de los edificios como se comenta en la metodología. Para este trabajo se crearon cuadriculas cuadradas de 0.5x0.5m^2 para facilitar que se asemejen a las formas irregulares se las superficies. A continuación, se realizó otro corte vectorial para conservar únicamente los recuadros de la cuadricula contenidos dentro de las superficies de la capa Union_MASK_Pend_RadAnual y que además estuviesen enteros, quedando como en la Figura 39 (se muestra de cerca para poner apreciar la cuadrícula). En este punto el área total disponible se redujo de 49.607,5 m^2 a 42.626,1 m^2.

Figura 39. Muestra de la cuadrícula recortada sobre la capa Union_MASK_Pend_RadAnual.

Posteriormente se agruparon los recuadros de forma que resultasen en rectángulos de tamaños considerablemente más grandes. El conjunto de rectángulos es filtrado para conservar solo aquellos con un área mínima de 10 m^2. El resultado puede ser apreciado en la Figura 40 junto con un ejemplo con la imagen TIF del barrio de fondo. El área final disponible para disponer paneles solares fotovoltaicos es de 21.159,1 m^2 lo que supone la mayor reducción de área entre etapas.

Figura 40. Cubiertas asemejadas a rectángulos.

En este punto solo queda trasladar la información de la capa Union_MASK_Pend_RadAnual, es decir la pendiente y orientación, a cada rectángulo y generar los parámetros de dimensiones (largo y ancho), ángulo de medianera y una numeración basada en su posición geográfica (para diferenciar cada cubierta). Esto se consiguio empleando las expresiones formuladas en la metodología dentro de la tabla de atributos de la capa vectorial que contiene los rectangulos.

Una vez que cada superficie rectangular contaba con sus parámetros asociados, se exportó la capa a formato XLSX o hoja de cálculo MS Office open XML y se guardó en la carpeta que contiene las funciones de MATLAB para la etapa de disposición automatizada de paneles para optimización energética con el nombre de “Superficies”.

3.5 Disposición automatizada de paneles para optimización energética en Illa Perduda

En la siguiente sección se procederá de acuerdo con la metodología vista en el punto 2.5.

En primer lugar, se eligieron los siguientes paneles solares FV como aquellos a considerar por el algoritmo:

DimensionPaneles = 5×4 table

	potencias	longitud	anchura	profundidad
1	150	1.4780	0.6740	0.0350
2	200	1.4850	0.6680	0.0300
3	330	1.9560	0.9920	0.0400
4	510	2.1870	1.1020	0.0350
5	670	2.3840	1.3030	0.0350

Figura 41. Tabla de parámetros de los paneles solares FV.

Donde:

potencias: potencia nominal del panel (W).

longitud: Largo del panel (m).

anchura: Ancho del panel (m).

profundidad: espesor del panel (m).

Se eligieron cinco paneles solares comerciales representativos, con características variadas asegurando una muestra diversa y relevante para el análisis. La selección sigue la tendencia de que, a mayor potencia nominal del panel, mayores son sus dimensiones. Los paneles comerciales usados fueron, en orden:

1. Panel Solar 150W Policristalino 12V [32].
2. Panel Solar 200W 12V Monocristalino PERC Solar [33].
3. Panel Solar 330W Policristalino 12V [34].
4. Panel Solar 510W TSM-510DE18M.08(II) VERTEX [35].
5. Panel Solar 670W | Mono-PERC Trina Solar [36].

Esta tabla editable es guardada automáticamente y leída por el código *DisposicionOptimizadaCompleta.* Posteriormente, se procedió a elegir un coeficiente de pérdidas del 14% de acuerdo con las recomendaciones de PVGIS [26]. A continuación, se ejecutó el código *DisposicionOptimizadaCompleta* donde se pudieron apreciar los resultados, tanto los datos del mejor arreglo paralelo a los bordes de la cubierta como los del orientado al sur (con la

representación gráfica de este último), de las 578 superficies rectangulares que se generaron en el paso previo (paso 3.4). Un ejemplo de la salida de los resultados se aprecia en la Figura 18.

La energía total que se podría producir en el barrio completo aplicando el algoritmo de optimización es de 4,45 GWh/año con una potencia instalada de 2,93 MW.

Capítulo 4. Resultados y discusión

En este capítulo se resumirán e interpretarán los resultados obtenidos en el estudio.

4.1 Evolución del área disponible

Como ya se ha adelantado, inicialmente se contaba con una superficie disponible de 52.782,6 m^2 la cual se redujo tras cada etapa de la siguiente forma:

- En el paso 3.1 tras el filtrado de puntos aislados en la detección de áreas libres de obstáculos: 51.142,3 m^2.
- En el paso 3.4 tras filtrar las superficies por su orientación cardinal y pendiente: 49.607,5 m^2.
- En el paso 3.4 tras filtrar por irradiación mínima: 42.626,1 m^2.
- Por último, al final del paso 3.4 tras asemejar las superficies a rectángulos: 21.159,1 m^2.

Esta área final está compuesta por las ya mencionadas 578 superficies rectangulares que se exportaron al código de disposición automatizada de paneles para optimización energética. A continuación, se presentan algunos resultados de la ejecución del código:

4.2 Comparación de la relación m^2/kW resultante

La energía total que se podría producir en el barrio completo aplicando el algoritmo de optimización resultó de 4,45 GWh/año con una potencia instalada de 2,93 MW, lo que nos da una relación de 7,22 m^2/kWp_inst.

Esta relación es mucho más exacta que las estimaciones mencionadas en el estado del arte, donde se estimaban 10m^2 para disponer cada kW de potencia instalada [10]. Este resultado también se puede comparar con los estudios llevados a cabo en Europa que se agrupan en el estado del arte realizado por E. Fakhraian, M.A. Forment, F.V. Dalmau et al. [20] donde se presentan una serie de resultados relativos al potencial fotovoltaico de los tejados de diversas localizaciones aplicando variados métodos de estudio. Un ejemplo de estos estudios es el realizado en Suiza elaborado por Assouline D. usando un proceso de "Machine learning" (subconjunto de la inteligencia artificial que se enfoca en enseñar a las computadoras para que aprendan de los datos y mejoren con la experiencia) con un algoritmo basado en vectores, el valor que arrojó dicho estudio fue una relación de 19,28 m^2/kWp_inst de acuerdo a la ecuación 16 la cual permite estimar dicha relación.

$$\frac{Superficie\ Disponible\ (m^2)}{Energía\ Generada\ (kWh)} \times Producción\ Media\ de\ Energía\ por\ kW\ \left(\frac{kWh}{kW}\right) \qquad (16)$$

En el caso anteriormente mencionado la superficie disponible era de 328 km^2, la energía generada de 17,86 TWh y la producción media de energía por kW instalado en Suiza de acuerdo a PVGIS [26] es de unos 1050 kWh/kWp_inst.

Otro caso que menciona este autor es el llevado a cabo en Ludwigsburg en Alemania hecho por Romero Rodriguez L. con modelos 3D y la plataforma SimStadt el cual arroja una relación de 16,89 m2/kW de acuerdo de nuevo con la ecuación 16 (la superficie disponible era de 22,26 km^2, la

energía generada de 1318 GWh y la producción media de energía por kW instalado en Ludwigsburg de acuerdo a PVGIS [26] es de unos 1000 kWh/kWp_inst).

En la Península Ibérica encontramos los casos de Lisboa en Portugal elaborado por Brito MC usando el software ArcGIS con complementos de análisis solar obteniendo una relación de 8,61 m2/kW (la superficie disponible era de 538 tejados estimados en 115 m^2 cada uno, la energía generada de 11.5 GWh y la producción media de energía por kW instalado en Lisboa de acuerdo a PVGIS [26] es de unos 1600 kWh/kWp_inst) y el caso de España hecho por Gomez-Exposito A. usando un algoritmo de interpolación en el Matlab, ArcGIS y PVGIS, dicho estudio arroja una relación de 5,86 m2/kW (la superficie disponible era de 1134 km^2, la energía generada de 291 TWh y la producción media de energía por kW instalado en España de acuerdo a PVGIS [26] es de unos 1500 kWh/kWp_inst).

Comparando estos resultados se interpreta que la relación obtenida en este trabajo para el caso de Illa Perduda es similar a los estudios agrupados por E. Fakhraian, M.A. Forment, F.V. Dalmau et al., destacando una mejora en cuanto a la superficie necesaria para disponer cada kW de potencia instalada respecto al estudio llevado a cabo en Suiza, en Ludwigsburg y en Lisboa, pero presenta un resultado ligeramente peor al del caso realizado en España por Gomez-Exposito A. En la siguiente tabla se muestra un resumen de la comparación:

Autor	Localización del estudio	Metodología	Relación m^2/kW
Pic Munuera E	Illa Perduda, Valencia, España	Matlab, QGIS y PVGIS	7,22
Assouline D	Suiza	Machine learning, Vectores	19,28
Romero Rodriguez L	Ludwigsburg, Alemania	modelos 3D y SimStadt	16,89
Brito MC	Lisboa, Portugal	ArcGIS	8,61
Gomez-Exposito A	España	Matlab, ArcGIS y PVGIS	5,86

Tabla 2. Resumen de estudios de potencial fotovoltaico.

4.3 Ratios de configuraciones

A continuación, se presentarán de forma numérica y gráfica las frecuencias con las que el algoritmo seleccionó cada configuración como la mejor, en relación con el total de casos.

Por un lado, de las 578 superficies solo 11 de ellas resultaron tener la pendiente suficiente para aplicar la configuración coplanar lo que supone un 2% del total como se muestra en la Figura 42.

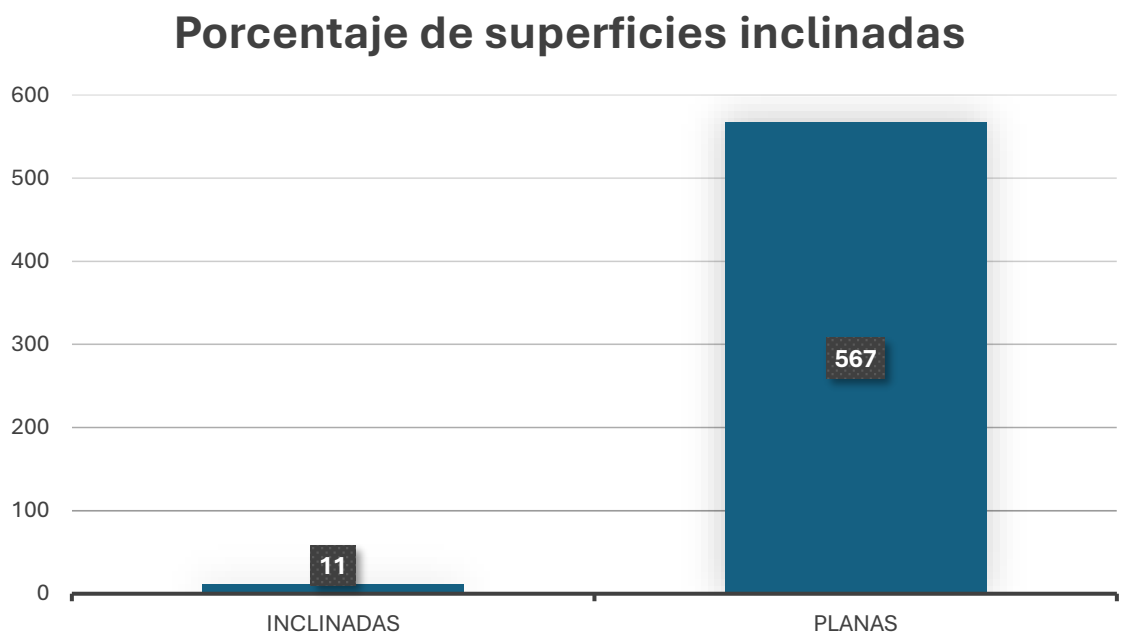


Figura 42. Ratio de superficies inclinadas o con arreglo coplanar respecto al total.

De entre las superficies planas (en las que se usaron estructuras con soportes), se aprecia que la inclinación más frecuente de los paneles del mejor arreglo de cada superficie es de 35º con 443 casos de los 567, seguido por la inclinación de 30º con 63 casos, luego la de 25º con 46 casos, seguida de la de 40º con 15 casos y por último la de 45º la cual no fue la mejor en ningún caso (Figura 43).

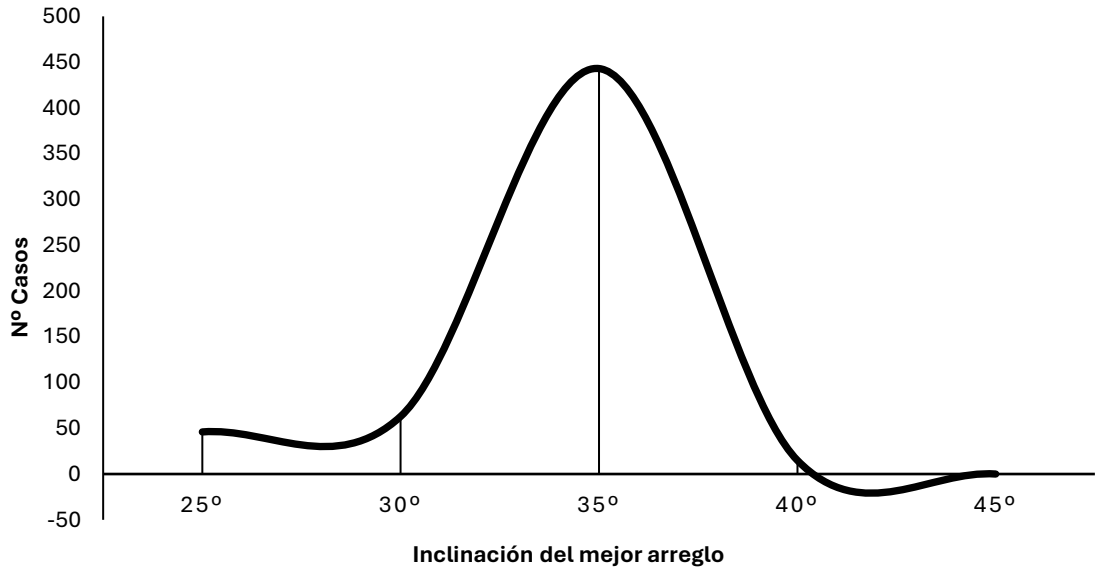


Figura 43. Representación de la distribución de las mejores inclinaciones para cada superficie.

Con este resultado es posible validar la simplificación comúnmente hecha la cual afirma que una buena estimación para la inclinación de un panel fotovoltaico es la latitud de la zona geográfica en la que este está situado, esta estimación se menciona en el artículo de E. González [16] ya

mencionado anteriormente. También respalda su resultado final el cual expresa que el ángulo óptimo para maximizar la producción de energía en la península ibérica es 34°.

Otro aspecto que valorar, son los tipos de panel que más veces se presentaron en el arreglo que más energía sería capaz de generar. Se aprecia en la Figura 44 como el panel usado en la mayoría de los arreglos es el de mayor potencia nominal, de 670W, pese a ser el que más espacio ocupa por panel con 353 arreglos. En segundo lugar, se encuentran las configuraciones con paneles de 510 W con 123 arreglos. En tercer lugar, los arreglos con paneles de 200W con 77 arreglos. En cuarto lugar, los arreglos con paneles de 330W con 25 arreglos de los 578 que hay en total. Por último, en ningún caso se dio que el mejor arreglo de una superficie fuese con paneles de 150W.

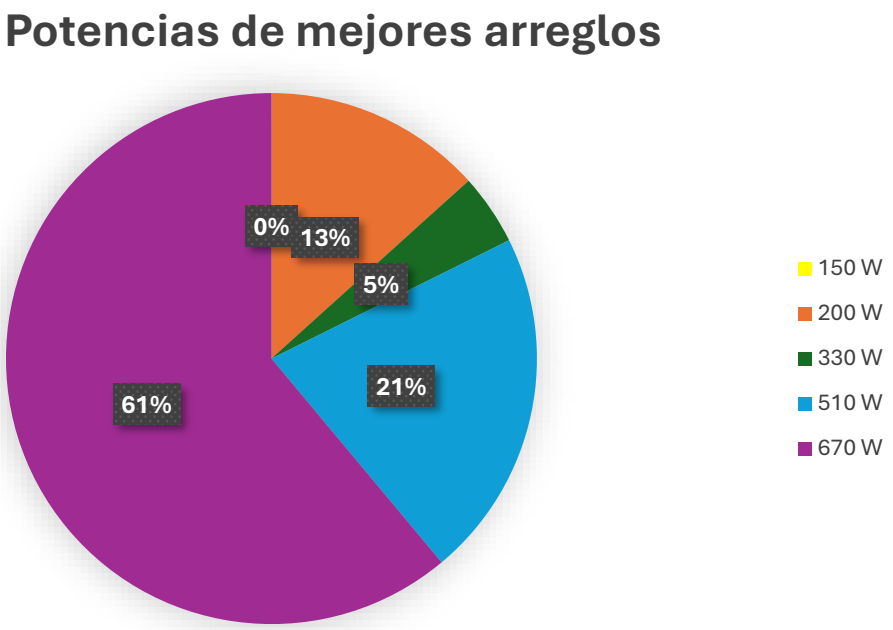


Figura 44. Ratio de potencias de panel respecto del total.

Con estos resultados se interpreta que no siempre el arreglo que mayor energía total es capaz de generar es aquel con paneles de mayor potencia nominal, a veces un panel más pequeño permite crear un arreglo capaz de generar una mayor energía total porque sus dimensiones se acomodan mejor a la superficie.

Otra ratio interesante de observar es la proporción de veces que un arreglo orientado al sur ha resultado el mejor frente a uno paralelo a los bordes de la cubierta. Del total de superficies, en 511 el arreglo que más energía generaría ha resultado ser con paneles solares FV orientados de forma paralela a los bordes de la cubierta, mientras que en 67 casos la superficie ya estaba orientada al sur por lo que ambos arreglos eran iguales (Figura 45). En ningún arreglo la configuración orientada al sur logró generar más energía que la paralela pese a generar tanta o más energía por kW instalado. Esto se debe a que esa mejor orientación no compensaba el mayor número de paneles (y por tanto la mayor potencia instalada) que la configuración paralela era capaz de acomodar en las superficies.

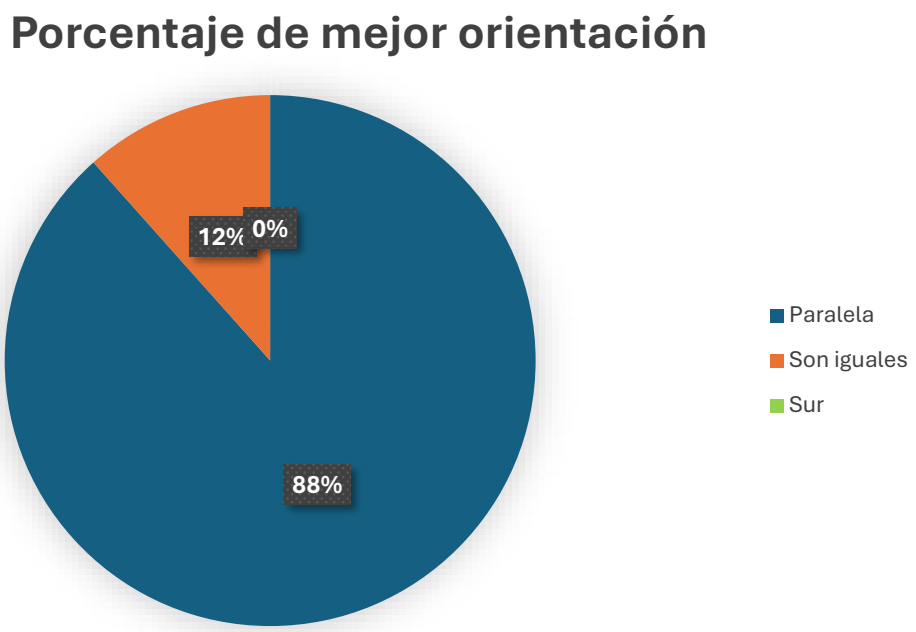


Figura 45. Porcentaje de mejores orientaciones.

Por último, se calculó también el número de arreglos en el que el lado del panel apoyado sobre la superficie era el más corto y el número de veces que era mejor apoyar el lado largo para obtener una mayor energía total. Los resultados de dichos cálculos son los siguientes:

Por un lado, el número de arreglos donde la mejor configuración resultó tener los paneles solares FV apoyados sobre su lado corto (llamado W en la metodología) fue de 484, los arreglos donde la mejor configuración resultó tener los paneles solares FV apoyados sobre su lado largo (llamado L en la metodología) fue de 61 y los casos donde en el mejor arreglo era indiferente disponer los paneles apoyados sobre su lado corto o largo fue de 33 como se aprecia en la Figura 46.

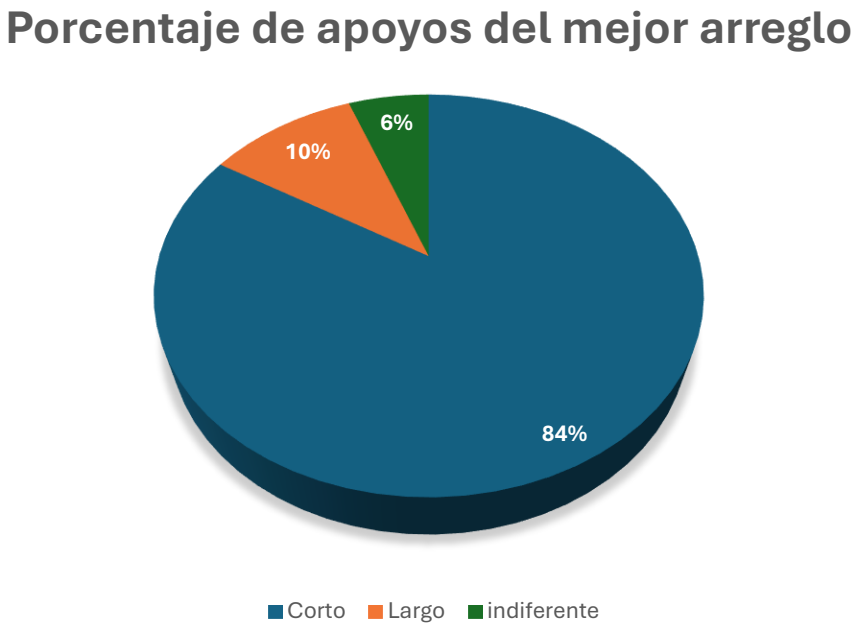


Figura 46. Porcentaje de apoyos del mejor arreglo.

De estos resultados se puede deducir que, pese a que habitualmente se disponen los paneles solares FV de forma que el lado mayor esté apoyado en el suelo para aumentar su resistencia al viento o reducir su carga frente a esfuerzos producidos por el viento, puede resultar que en muchos casos disponer los paneles solares FV de forma opuesta, es decir, apoyando el lado corto, se consiga un arreglo que genere mucha más energía en total.

Capítulo 5. Conclusiones

En este apartado se buscará recapitular los objetivos que se han conseguido con este trabajo, ofrecer unas reflexiones sobre las aplicaciones que se le podrían dar y proponer futuras ideas de desarrollo.

Como se ha podido apreciar en los puntos anteriores, se ha conseguido desarrollar una metodología capaz de calcular el potencial solar fotovoltaico de una localidad a gran escala mediante el uso de un código automatizado en Matlab que elimina las zonas con obstáculos gracias al uso de visión artificial. Se hizo un estudio de la irradiación disponible en QGIS mediante modelos digitales de elevación, de pendiente y de orientación para filtrar superficies en sombra. También se calculó la pendiente y orientación de cada superficie de la zona de estudio y se usó dicha información para conservar solo aquellas con características adecuadas de acuerdo con el uso que se plantea darles (generación de energía mediante paneles solares FV) y de acuerdo con la zona geográfica donde se encuentra la localidad estudiada. Por último, también se consiguió desarrollar un código que, de forma automatizada, es capaz de calcular el mejor arreglo en lo relativo a generar la máxima energía total posible para una superficie dada. Esto valorando el disponer distintos paneles comerciales, además de valorar disponerlos de forma coplanar a la superficie, usando soportes con distintas inclinaciones, sobre su lado corto o largo y orientados de forma paralela a los bordes de la cubierta u orientados al sur.

Esta metodología podría ser de gran utilidad para planificadores energéticos del territorio y de las ciudades, instaladores de paneles solares FV y para el sector de la producción y fabricación de paneles. Debido a que permitiría acercarse a la situación real de los clientes y poder tomar decisiones con información más precisa que la encontrada en la literatura hasta ahora.

5.1 Oportunidades de mejora a futuro

Entre las propuestas de mejora cabría destacar la posibilidad de utilizar un software más especializado como podría ser el caso de ArcGIS en lugar de QGIS, que fue el usado en este trabajo. ArcGIS, cuenta con mayor número de herramientas y complementos para el estudio de la irradiación solar lo que justificaría su uso tan extendido por parte de los autores citados anteriormente.

Por otro lado, otra forma de mejorar los resultados del estudio sería contar con recursos tales como ortofotos y nubes de puntos “LIDAR” más actualizadas, además de registros catastrales libres de errores como los comentados en la Figura 25.

En adición, podría implementarse o desarrollarse un algoritmo mediante lenguajes de programación más ágiles que Matlab para mejorar la etapa vista en el paso 2.4 relativa a adecuar las superficies irregulares resultantes de los procesos de la metodología a rectángulos para su posterior exportación.

Por último, de conocerse las características estructurales reales de los edificios estudiados, se podría añadir una etapa adicional a la metodología donde se estime si es posible para el edificio soportar las cargas estructurales que suponen los arreglos de paneles solares FV.

En síntesis, este trabajo ha permitido evidenciar que las simplificaciones comúnmente hechas para estimar el potencial solar fotovoltaico de una localidad son poco precisas y que una superficie determinada puede tener una configuración de paneles solares FV óptima distinta a lo que se podría proponer en una planificación simple.

Capítulo 6. Presupuesto

En este capítulo se detallarán los costes asociados al desarrollo del proyecto desde actividades principales como secundarias, hasta recursos y equipos usados durante el estudio.

Es importante tener en cuenta que el coste de los equipos informáticos se calcula suponiendo una amortización a los 5 años. Los softwares utilizados: QGIS (versión 3.36.2) y Matlab (versión R2023b) son de uso gratuito y de pago respectivamente, pero, este último no tuvo ningún coste al haberse usado bajo la licencia otorgada a la comunidad UPV.

En lo relativo al desarrollo de los códigos de Matlab, se incluyó el tiempo necesario para elaborar tanto los códigos "MyRooftypeSET.m" y "MyRooftypeDetection.m" como el código "DisposicionOptimizadaCompleta" y sus funciones "Sur", "Paralelo"," ParaleloCoplanar", etc.

Finalmente, los archivos tales como ortofotos, registros catastrales y nubes de puntos LIDAR fueron descargados de centros de libre descarga y no supusieron ningún coste económico.

Cuadro de precios descompuestos:

N.º Orden	Descripción de unidades de obra	Ud.	Rendimiento	Precio (€)	Importe (€)
01	**Elaboración del estado del arte incluyendo recopilación de información**				
01.01	Ingeniero Junior	h	45	30	1.350
01.02	Ordenador Asus X571GT	h	45	0,09	4,05
			Coste Total		**1.354,05**
02	**Obtención de datos, incluyendo ortofotos, archivos catastrales y nubes de puntos LIDAR**				
01.01	Ingeniero Junior	h	10	30	300
01.02	Ordenador Asus X571GT	h	10	0,09	0,9
			Coste Total		**300,9**
03	**Diseño de la metodología y sus pasos desde la obtención de superficies potenciales por visión artificial hasta la disposición automatizada de paneles para optimización energética**				
01.01	Ingeniero Junior	h	80	30	2.400
01.02	Ordenador Asus X571GT	h	80	0,09	7,2
03.01	Software QGIS	h	60	0	0
03.02	Software Matlab	h	10	0	0
			Coste Total		**2.407,2**
04	**Desarrollo de los códigos de Matlab**				
01.01	Ingeniero Junior	h	100	30	3.000
01.02	Ordenador Asus X571GT	h	100	0,09	9
03.02	Software Matlab	h	100	0	0

		Coste Total			**3.009**
05	**Cálculos, análisis e interpretación de resultados**				
01.01	Ingeniero Junior	h	20	30	600
01.02	Ordenador Asus X571GT	h	20	0,09	1,8
		Coste Total			601,8
06	**Redacción del trabajo**				
01.01	Ingeniero Junior	h	80	30	2.400
01.02	Ordenador Asus X571GT	h	80	0,09	7,2
		Coste Total			**2.407,2**
07	**Transporte**				
07.01	Desplazamiento en bicicleta de alquiler (Valenbisi)	Viajes	120	0,52	32,4
		Coste Total			**32.4**
08	**Tutorías**				
08.01	Profesor Ayudante Doctor	h	20	40	800
08.02	Profesor Titular	h	20	50	1.000
		Coste Total			**1.800**

Presupuesto:

N.º Orden	Descripción de unidades de obra	Mediciones	Precio (€)	Importe (€)
01	**Elaboración del estado del arte**	1	1.354,05	1.354,05
			Coste Total	**1.354,05**
02	**Obtención de datos, incluyendo ortofotos, archivos catastrales y nubes de puntos LIDAR**	1	300,9	300,9
			Coste Total	**300,9**
03	**Diseño de la metodología y sus pasos desde la obtención de superficies potenciales por visión artificial hasta la disposición automatizada de paneles para optimización energética**	1	2.407,2	2.407,2
			Coste Total	**2407,2**
04	**Desarrollo de los códigos de Matlab**	1	3.009	3.009
			Coste Total	**3.009**
05	**Cálculos y análisis**	1	601,8	601,8
			Coste Total	**601,8**
06	**Redacción del trabajo**	1	2.407,2	2.407,2
			Coste Total	**2.407,2**
07	**Transporte**	1	32,4	32,4

		Coste Total		**32,4**
08	**Tutorías**	1	1.800	1.800
		Coste Total		**1.800**
		Total Presupuesto		**11.912,55**

Resumen del presupuesto:

Presupuesto de ejecución material (PEM)	11.912,55 €
6% Beneficio Industrial	714,75
Presupuesto de ejecución por contrata (PEC)	12.627,30 €
21% IVA	2.651,73
Presupuesto Global	15.279,04 €

El presupuesto asciende a la cantidad de EUROS: QUINCE MIL DOSCIENTOS SETENTA Y NUEVE CON CUATRO CÉNTIMOS.

Capítulo 7. Bibliografía:

[1] ‘Estadísticas de consumo energético mundial | Enerdata’. Accessed: Aug. 09, 2024. [Online]. Available: https://datos.enerdata.net/energia-total/datos-consumo-internacional.html

[2] bp, ‘Statistical Review of World Energy 2022’.

[3] ‘Solar resource maps & GIS data for 200+ countries | Solargis’. Accessed: Aug. 09, 2024. [Online]. Available: https://solargis.com/es/resources/free-maps-and-gis-data?locality=europe

[4] wwwreees, ‘Informe del Sistema Eléctrico: Informe resumen de energías renovables 2023’, 2023.

[5] ‘REData - Estructura generacion | Red Eléctrica’. Accessed: Aug. 02, 2024. [Online]. Available: https://www.ree.es/es/datos/generacion/estructura-generacion

[6] J. Khan and M. H. Arsalan, ‘Solar power technologies for sustainable electricity generation – A review’, *Renewable and Sustainable Energy Reviews*, vol. 55, pp. 414–425, Mar. 2016, doi: 10.1016/J.RSER.2015.10.135.

[7] ‘Residencial, comercial e institucional’. Accessed: Aug. 09, 2024. [Online]. Available: https://www.miteco.gob.es/es/cambio-climatico/temas/mitigacion-politicas-y-medidas/edificacion.html

[8] ‘Pacto Verde Europeo - Consilium’. Accessed: Aug. 02, 2024. [Online]. Available: https://www.consilium.europa.eu/es/policies/green-deal/

[9] ‘Las ciudades, “causa y solución” del cambio climático | Noticias ONU’. Accessed: Aug. 02, 2024. [Online]. Available: https://news.un.org/es/story/2019/09/1462322

[10] T. Gómez-Navarro, T. Brazzini, D. Alfonso-Solar, and C. Vargas-Salgado, ‘Analysis of the potential for PV rooftop prosumer production: Technical, economic and environmental assessment for the city of Valencia (Spain)’, *Renew Energy*, vol. 174, pp. 372–381, Aug. 2021, doi: 10.1016/J.RENENE.2021.04.049.

[11] Q. Zhong, J. R. Nelson, D. Tong, and T. H. Grubesic, ‘A spatial optimization approach to increase the accuracy of rooftop solar energy assessments’, *Appl Energy*, vol. 316, p. 119128, Jun. 2022, doi: 10.1016/J.APENERGY.2022.119128.

[12] Q. Zhong and D. Tong, ‘Spatial layout optimization for solar photovoltaic (PV) panel installation’, *Renew Energy*, vol. 150, pp. 1–11, May 2020, doi: 10.1016/J.RENENE.2019.12.099.

[13] N. Mignoni, R. Carli, and M. Dotoli, ‘Layout Optimization for Photovoltaic Panels in Solar Power Plants via a MINLP Approach’, *IEEE Transactions on Automation Science and Engineering*, 2023, doi: 10.1109/TASE.2023.3322786.

[14] A. Alharbi, Z. Awwad, A. Habib, and O. de Weck, ‘Economical sizing and multi-azimuth layout optimization of grid-connected rooftop photovoltaic systems using Mixed-Integer Programming’, *Appl Energy*, vol. 335, p. 120654, Apr. 2023, doi: 10.1016/J.APENERGY.2023.120654.

[15] N. ur Rehman, M. Uzair, and U. Allauddin, 'An optical-energy model for optimizing the geometrical layout of solar photovoltaic arrays in a constrained field', *Renew Energy*, vol. 149, pp. 55–65, Apr. 2020, doi: 10.1016/J.RENENE.2019.12.040.

[16] E. González-González, J. Martín-Jiménez, M. Sánchez-Aparicio, S. Del Pozo, and S. Lagüela, 'Evaluating the standards for solar PV installations in the Iberian Peninsula: Analysis of tilt angles and determination of solar climate zones', *Sustainable Energy Technologies and Assessments*, vol. 49, p. 101684, Feb. 2022, doi: 10.1016/J.SETA.2021.101684.

[17] 'PVsyst – Photovoltaic software'. Accessed: Aug. 09, 2024. [Online]. Available: https://www.pvsyst.com/

[18] 'HOMER Pro - Microgrid Software for Designing Optimized Hybrid Microgrids'. Accessed: Aug. 09, 2024. [Online]. Available: https://homerenergy.com/products/pro/index.html

[19] R. E. Jones and J. F. Burkhart, 'Shading effects of collector rows tilted toward the equator', *Solar Energy*, vol. 26, no. 6, pp. 563–565, Jan. 1981, doi: 10.1016/0038-092X(81)90171-7.

[20] E. Fakhraian, M. A. Forment, F. V. Dalmau, A. Nameni, and M. J. C. Guerrero, 'Determination of the urban rooftop photovoltaic potential: A state of the art', *Energy Reports*, vol. 7, pp. 176–185, Sep. 2021, doi: 10.1016/J.EGYR.2021.06.031.

[21] 'Centro de Descargas del CNIG (IGN)'. Accessed: Jun. 25, 2024. [Online]. Available: https://centrodedescargas.cnig.es/CentroDescargas/index.jsp#

[22] 'Spatial without Compromise · QGIS Web Site'. Accessed: Aug. 09, 2024. [Online]. Available: https://www.qgis.org/

[23] 'huellasolar. Aplicación Web. Mapas de soleamiento y radiación de ciudades'. Accessed: Jun. 27, 2024. [Online]. Available: https://www.huellasolar.com/

[24] 'Algoritmo de Douglas-Peucker'. Accessed: Aug. 09, 2024. [Online]. Available: https://www.aplitop.com/Products/TcpMDT/v8/help/es/referencia/algoritmo_de_douglas_peucker.htm

[25] 'IDAE Instituto para la Diversificación', Accessed: Jul. 22, 2024. [Online]. Available: www.idae.es

[26] 'JRC Photovoltaic Geographical Information System (PVGIS) - European Commission'. Accessed: Aug. 05, 2024. [Online]. Available: https://re.jrc.ec.europa.eu/pvg_tools/es/

[27] 'Portal de Datos Abiertos del Ayuntamiento de València — Portal de l'Ajuntament de la ciutat de València'. Accessed: Jul. 30, 2024. [Online]. Available: https://valencia.opendatasoft.com/pages/home/

[28] 'Sede Electrónica del Catastro - Inicio'. Accessed: Jul. 30, 2024. [Online]. Available: https://www.sedecatastro.gob.es/

[29] 'TABULA-EPISCOPEEnergy Performance Indicator Tracking Schemes for the Continuous Optimisation of Refurbishment Processes in European Housing Stocks |

IVE'. Accessed: Aug. 29, 2024. [Online]. Available: https://www.five.es/project/episcope/

[30] 'Evaluación del potencial de energía solar térmica y fotovoltaica derivado del cumplimiento del Código Técnico de Edificación | Idae'. Accessed: Jul. 31, 2024. [Online]. Available: https://www.idae.es/publicaciones/evaluacion-del-potencial-de-energia-solar-termica-y-fotovoltaica-derivado-del

[31] D. Básico SUA, 'MINISTERIO DE VIVIENDA Y AGENDA URBANA'.

[32] 'Panel Solar 150W Policristalino 12v'. Accessed: Aug. 19, 2024. [Online]. Available: https://www.wccsolar.net/panel-solar-150w-12v-policristalino/

[33] 'Panel Solar 200W 12V Monocristalino PERC Solar Components | Solar Components'. Accessed: Aug. 19, 2024. [Online]. Available: https://solarcomponents.es/producto/panel-solar-200w-12v-monocristalino-perc-solar-components/

[34] 'Panel Solar 330W Policristalino para instalaciones solares 12V 24V 48V'. Accessed: Aug. 19, 2024. [Online]. Available: https://www.wccsolar.net/panel-solar-330w-policristalino/

[35] 'Panel Solar 510W TSM-510DE18M.08(II) VERTEX - Mundosolar'. Accessed: Aug. 19, 2024. [Online]. Available: https://www.mundosolar.es/solar-fotovoltaica/tsm-510de18m-08ii-vertex/

[36] 'Panel Solar 650W Monocrystaline 37.4V 132 cells 2384×1303×35 mm | VERTEX Series | TRINA SOLAR | Techno Sun Webportal B2B'. Accessed: Aug. 19, 2024. [Online]. Available: https://b2b.technosun.com/shop/sol0427-panel-solar-670w-mono-perc-382v-1755a-132-celulas-2384x1303x35mm-tsm-de21-670-vertex-trina-solar-37925?srsltid=AfmBOorPyfVNBwYRC64NLhMpVZrC7KFN50PKEij4pkWBFKYZpg_MXGco

ANEXO 1. Códigos de QGIS

Bucle para el cálculo de la irradiación anual

```
import processing

for x in range(1, 366):

   fp="Dirección_Carpeta/Nombre_Carpeta" + str(x) +".tif"

   processing.run("grass:r.sun.insoltime",
{'elevation':'C:/ArchivosLas/Semana27/DEM_ISLAPERDUDA.tif','aspect':'C:/ArchivosLas/Semana27
/Modelo_Aspecto.tif','aspect_value':270,'slope':'C:/ArchivosLas/Semana27/Modelo_Pendiente.tif'
,'slope_value':0,'linke':None,'albedo':None,'albedo_value':0.2,'lat':None,'long':None,'coeff_bh':No
ne,'coeff_dh':None,'horizon_basemap':None,'horizon_step':None,'day':1,'step':0.5,'declination':N
one,'distance_step':1,'npartitions':1,'civil_time':None,'-p':False,'-
m':False,'glob_rad':fp,'GRASS_REGION_PARAMETER':None,'GRASS_REGION_CELLSIZE_PARAMETE
R':0,'GRASS_RASTER_FORMAT_OPT':'','GRASS_RASTER_FORMAT_META':''})
```

ANEXO 2. Códigos de MATLAB

MyRooftypeSET.m

```
% Roof Type Generation

clc;

clear;


% Roof type structure loading

load("MATLABData\roofType.mat","roofType");


%% Image

% Definir la ruta del directorio

folder_path ='D:\Universidad\UPV cuarto año\TFG\Cosas de Qgis\Catastro';
%'C:\Users\Tomas\Desktop\TFM\Ortofotos Valencia\';


% Solicitar al usuario seleccionar un archivo dentro del directorio
```

```
disp('Seleccione un archivo dentro del folder.');
[filename, folder] = uigetfile(fullfile(folder_path, '*.*'), 'Seleccionar archivo');

% Verificar si se seleccionó un archivo válido
if isequal(filename, 0)
   error('No se seleccionó ningún archivo válido.');
end

% Obtener la ruta completa del archivo seleccionado
image_path = fullfile(folder, filename);

% Cargar la imagen del tejado
tAerialImage = Tiff(image_path); %crea tiff
AerialImage = tAerialImage.read; %la lee
AerialImageHSV = rgb2hsv(AerialImage); %pasa de RGB a HSV

% Pixel detection
answer = questdlg("Would you like to add a new roofType for this image?");
while answer=="Yes"
   roofName = input("Please, write down the name of the new roofType.\n","s");
   colorRegionName = input("Please, write down the name of the new colorRegion for this roofType.\n","s");
   pixelClicks = selectRoofPixels(AerialImage);

   newRoofType = struct();
   newRoofType.name = convertCharsToStrings(roofName);

   colorRegion = struct();
   colorRegion.name = convertCharsToStrings(colorRegionName);
   colorRegion.pixelValuesRGB = getPixelValues(AerialImage,pixelClicks);
   colorRegion.pixelValuesHSV = getPixelValues(AerialImageHSV,pixelClicks);
```

```
    colorRegion.statistics = getStatistics(colorRegion);
    newRoofType.colorRegions = [];
    newRoofType.colorRegions = [newRoofType.colorRegions colorRegion];

    roofType = [roofType newRoofType];

    answer = questdlg("Would you like to add a new roofType for this image?");
end

answer = questdlg("Would you like to add a new colorRegion for an existing roofType?");

while answer=="Yes"

    [idx,tf] = listdlg('ListString',[roofType.name],'SelectionMode','single','PromptString',"Select the roofType to modify:");

    if isempty(idx)
        break
    end

    colorRegionName = input("Please, write down the name of the new colorRegion for this roofType.\n","s");
    pixelClicks = selectRoofPixels(AerialImage);
    newColorRegion = struct();
    newColorRegion.name = convertCharsToStrings(colorRegionName);
    newColorRegion.pixelValuesRGB = getPixelValues(AerialImage,pixelClicks);
    newColorRegion.pixelValuesHSV = getPixelValues(AerialImageHSV,pixelClicks);
    newColorRegion.statistics = getStatistics(newColorRegion);
    roofType(idx).colorRegions = [roofType(idx).colorRegions newColorRegion];

    answer = questdlg("Would you like to add a new colorRegion for an existing roofType?");
end
```

```
answer = questdlg("Would you like to add more points to an existing colorRegion?");

while answer=="Yes"

   [roofTypeIdx,tf] = listdlg('ListString',[roofType.name],'SelectionMode','single','PromptString',"Select the roofType to modify:");

   if isempty(roofTypeIdx)
      break
   end

   [colorRegionIdx,tf] = listdlg('ListString',[roofType(roofTypeIdx).colorRegions.name],'SelectionMode','single','PromptString',"Select the colorRegion to modify:");

   if isempty(colorRegionIdx)
      break
   end

   pixelClicks = selectRoofPixels(AerialImage);

   roofType(roofTypeIdx).colorRegions(colorRegionIdx).pixelValuesRGB = [roofType(roofTypeIdx).colorRegions(colorRegionIdx).pixelValuesRGB;
getPixelValues(AerialImage,pixelClicks)];
   roofType(roofTypeIdx).colorRegions(colorRegionIdx).pixelValuesHSV = [roofType(roofTypeIdx).colorRegions(colorRegionIdx).pixelValuesHSV;
getPixelValues(AerialImageHSV,pixelClicks)];
   roofType(roofTypeIdx).colorRegions(colorRegionIdx).statistics = getStatistics(roofType(roofTypeIdx).colorRegions(colorRegionIdx));

   answer = questdlg("Would you like to add more points to an existing colorRegion in this image?");
end
```

```
close all;
% Save the roofType struct
save("MATLABData\roofType","roofType");
%%
% Save the roofType struct
save("MATLABData\roofType","roofType");

%% Lo siguiente es una funcion

function pClicks = selectRoofPixels(Image)

   f = figure('Name','Selected building pixeels');
   imshow(Image);
   hold on
   pClicks = zeros(size(Image,[1 2]));
   pC=imshow(pClicks);
   pC.AlphaData = 0.4;

   key = 0;

   while key ~= 13

      % Wait until some key or button is pressed and get the position of the
      % mouse and the pressed key.
      [x,y,key] = ginput(1);
      xPixel = round(x);
      yPixel = round(y);

      % If it was not enter
      if key ~= 13
```

```
            % If it was the left click add that FA to the deletion list.
            if key == 1
                pClicks(yPixel,xPixel) = 1;
                % If it was the right click remove the FA from the deletion
                % list.
            elseif key == 3
                pClicks(yPixel,xPixel) = 0;
            end
        end

        % Refresh the image and show the selected FAs in red.
        figure(f);
        delete(pC);
        pC=imshow(conv2(pClicks,ones(5),"same")>0);
        pC.AlphaData = 0.4;

    end

end

function pixelValues = getPixelValues(Image,selectedPixels)

    [yPixel,xPixel] = find(selectedPixels);
    yPixelcoord = repelem(yPixel,3);
    xPixelcoord = repelem(xPixel,3);
    zPixelcoord = repmat((1:3)',length(xPixel),1);
    PixelIdx = sub2ind(size(Image),yPixelcoord,xPixelcoord,zPixelcoord);
    pixelValues = reshape(Image(PixelIdx),[3,length(xPixel),])';

end
```

```
function regionStats = getStatistics(colorRegion)

    regionStats.RGB = struct();
    regionStats.RGB.min = min(colorRegion.pixelValuesRGB);
    regionStats.RGB.max = max(colorRegion.pixelValuesRGB);
    regionStats.RGB.mean = mean(colorRegion.pixelValuesRGB);
    regionStats.RGB.std = std(double(colorRegion.pixelValuesRGB));

    regionStats.HSV = struct();
    regionStats.HSV.min = min(colorRegion.pixelValuesHSV);
    regionStats.HSV.max = max(colorRegion.pixelValuesHSV);
    regionStats.HSV.mean = mean(colorRegion.pixelValuesHSV);
    regionStats.HSV.std = std(colorRegion.pixelValuesHSV);

end
```

MyRooftypeDetection.m

```
%% 1) RoofType struct loading

clear;

clc;


% Roof type structure loading

load("MATLABData\roofType.mat","roofType"); %ajustar a dirección


%% 2) Image struct creation / import


% Definir la ruta del directorio

folder_path ='D:\Universidad\UPV cuarto año\TFG\Cosas de Qgis\Catastro'; % ajustar a
dirección


% Solicitar al usuario seleccionar un archivo dentro del directorio

disp('Seleccione un archivo dentro del folder.');

[filename, folder] = uigetfile(fullfile(folder_path, '*.*'), 'Seleccionar archivo');


% Verificar si se seleccionó un archivo válido

if isequal(filename, 0)

   error('No se seleccionó ningún archivo válido.');

end


ruta_imagen = fullfile(folder, filename);

tAerialImage = Tiff([ruta_imagen]);


AerialImage = tAerialImage.read;

AerialImageHSV = rgb2hsv(AerialImage);


[~,imageName,~] = fileparts(tAerialImage.FileName);
```

```
fileStructPath = "MATLABData\imageMasks\" + imageName + ".mat";


if isfile(fileStructPath)
   load(fileStructPath);
   imageStruct.image.tiff = tAerialImage;
else
   imageStruct = struct();
   imageStruct.name = imageName;
   imageStruct.image = struct();
   imageStruct.image.tiff = tAerialImage;
   imageStruct.image.RGB = AerialImage;
   imageStruct.image.HSV = AerialImageHSV;
   len = length(roofType);
   for i = 1:len
      imageStruct.roofTypes(i) =
struct("name",{roofType(i).name},"colorRegion",{struct("name",{roofType(i).colorRegions.nam
e})});
   end
   imageStruct.image.medianFiltered = struct();
   imageStruct.image.medianFiltered.RGB =
cat(3,medfilt2(AerialImage(:,:,1),[5,5]),medfilt2(AerialImage(:,:,2),[5,5]),medfilt2(AerialImage
(:,:,3),[5,5]));
   imageStruct.image.medianFiltered.HSV =
rgb2hsv(imageStruct.image.medianFiltered.RGB);
end


%% 3) Create mask for roofType


while true


   medianFiltered = 1;
```

```
  [idx,tf] = listdlg('ListString',[roofType.name],'SelectionMode','single','PromptString',"Select the roofType to create the mask for:");

  if isempty(idx)
    break
  end

  imageStruct = createMask(imageStruct,roofType,idx,medianFiltered);
  % imageStruct = filterMask(imageStruct,idx);

  answer = questdlg("Would you like to create another mask for this median filtered image for some roofType?");

  if strcmp(answer, 'No')
    break
  end
end

%% 4) Trials I. Filter individual masks

% Tamaños de roofType y colorRegion
numRoofTypes = numel(imageStruct.roofTypes);
for i = 1:numRoofTypes
  numColorRegions = numel(imageStruct.roofTypes(i).colorRegion);

  for j = 1:numColorRegions
    % Filtrar las máscaras RGB
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB = struct();
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMeanStdMask = filterIndividualMask(imageStruct.roofTypes(i).colorRegion(j).meanStdMask.RGB, 100, 500);
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMinMaxMask = filterIndividualMask(imageStruct.roofTypes(i).colorRegion(j).meanStdMask.RGB, 100, 500);
```

```
    % Filtrar las máscaras HSV
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV = struct();
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMeanStdMask =
filterIndividualMask(imageStruct.roofTypes(i).colorRegion(j).meanStdMask.HSV, 100, 500);
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMinMaxMask =
filterIndividualMask(imageStruct.roofTypes(i).colorRegion(j).minMaxMask.HSV, 100, 500);

    % Calcular operación blobOR para máscaras RGB y HSV
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total = struct();
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.minMax = struct();
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.minMax.blobOR =
blobOR(imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMinMaxMask,
imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMinMaxMask);
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.minMax.OR =
imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMinMaxMask |
imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMinMaxMask;

    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.meanStd = struct();
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.meanStd.blobOR =
blobOR(imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMeanStdMask,
imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMeanStdMask);
    imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.meanStd.OR =
imageStruct.roofTypes(i).colorRegion(j).filteredMask.RGB.filteredMeanStdMask |
imageStruct.roofTypes(i).colorRegion(j).filteredMask.HSV.filteredMeanStdMask;

    close all;
  end
end

%% 5) Trials I. Fuse all the masks (LIGHT)

numRoofTypes = numel(imageStruct.roofTypes); % Obtener el número total de tipos de
tejados
```

```
for i = 1:numRoofTypes

  numColors = numel(imageStruct.roofTypes(i).colorRegion); % Obtener el número total de regiones de color para el tipo de tejado actual


  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax = struct();

  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.OR = false;

  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.blobOR = false;


  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd = struct();

  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.OR = false;

  imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.blobOR = false;


  for j = 1:numColors

    % Fusionar las máscaras de cada región de color del tipo de tejado actual

    imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.OR = imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.OR | imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.minMax.OR;

    imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.blobOR = imageStruct.roofTypes(i).mask.fusedFilteredMask.light.minMax.blobOR | imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.minMax.blobOR;


    imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.OR = imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.OR | imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.meanStd.OR;

    imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.blobOR = imageStruct.roofTypes(i).mask.fusedFilteredMask.light.meanStd.blobOR | imageStruct.roofTypes(i).colorRegion(j).filteredMask.total.meanStd.blobOR;

  end

end

%

% imageStruct.roofTypes(1).colorRegion(7).filteredMask = struct();

%

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.RGB = struct();

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMeanStdMask = filterIndividualMask(imageStruct.roofTypes(1).colorRegion(7).meanStdMask.RGB,200,800);
```

```
% imageStruct.roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMinMaxMask =
filterIndividualMask(imageStruct.roofTypes(1).colorRegion(7).meanStdMask.RGB,200,800);

%

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.HSV = struct();

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMeanStdMask =
filterIndividualMask(imageStruct.roofTypes(1).colorRegion(7).meanStdMask.HSV,200,800);

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMinMaxMask =
filterIndividualMask(imageStruct.roofTypes(1).colorRegion(7).minMaxMask.HSV,200,800);

%

% % Blob OR

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total = struct();

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.RGB = struct();

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.RGB.blobOR =
blobOR(imageStruct.roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMinMaxMask,ima
geStruct.roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMeanStdMask);

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.RGB.OR =
imageStruct.roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMinMaxMask|imageStruct.
roofTypes(1).colorRegion(7).filteredMask.RGB.filteredMeanStdMask;

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.HSV = struct();

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.HSV.blobOR =
blobOR(imageStruct.roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMinMaxMask,imag
eStruct.roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMeanStdMask);

% imageStruct.roofTypes(1).colorRegion(7).filteredMask.total.HSV.OR =
imageStruct.roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMinMaxMask|imageStruct.
roofTypes(1).colorRegion(7).filteredMask.HSV.filteredMeanStdMask;

%

% close all;

%

% %% Trials I. Fuse all the masks (SHADOW)

%

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow = struct();

%

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.RGB = struct();

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.RGB.OR =
imageStruct.roofTypes(1).colorRegion(4).filteredMask.total.RGB.OR|imageStruct.roofTypes(1
).colorRegion(3).filteredMask.total.RGB.OR;
```

```
% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.RGB.blobOR =
imageStruct.roofTypes(1).colorRegion(4).filteredMask.total.RGB.blobOR|imageStruct.roofTypes(1).colorRegion(3).filteredMask.total.RGB.blobOR;

%

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.HSV = struct();

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.HSV.OR =
imageStruct.roofTypes(1).colorRegion(4).filteredMask.total.HSV.OR|imageStruct.roofTypes(1).colorRegion(3).filteredMask.total.HSV.OR;

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.HSV.blobOR =
imageStruct.roofTypes(1).colorRegion(4).filteredMask.total.HSV.blobOR|imageStruct.roofTypes(1).colorRegion(3).filteredMask.total.HSV.blobOR;


%% 6) Trials I. Fuse light and shadow


% Fuse Light

 imageStruct.roofTypes(1).mask.fusedFilteredMask.light.total.blobOR =
blobOR(blobOR(imageStruct.roofTypes(1).mask.fusedFilteredMask.light.meanStd.blobOR,imageStruct.roofTypes(1).mask.fusedFilteredMask.light.meanStd.OR),...


blobOR(imageStruct.roofTypes(1).mask.fusedFilteredMask.light.minMax.blobOR,imageStruct.roofTypes(1).mask.fusedFilteredMask.light.minMax.OR));

% imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.total.blobOR =
blobOR(blobOR(imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.RGB.OR,imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.RGB.blobOR),...

%
blobOR(imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.HSV.OR,imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.HSV.blobOR));

% Fuse Shadow n Light

imageStruct.roofTypes(1).mask.fusedFilteredMask.total = struct();

imageStruct.roofTypes(1).mask.fusedFilteredMask.total =
imageStruct.roofTypes(1).mask.fusedFilteredMask.light.total.blobOR;%|imageStruct.roofTypes(1).mask.fusedFilteredMask.shadow.total.blobOR;


%% 7) PRE-Visualize it

%visualizeMask(imageStruct.image.RGB,imageStruct.roofTypes(1).mask.fusedFilteredMask.total);
```

```
%Solo mascara

visualizeMask(imageStruct.roofTypes(1).mask.fusedFilteredMask.total);

%% 7.1) Continue Work


imageStruct.roofTypes(1).mask.fusedFilteredMask.total=imageStruct.roofTypes(1).mask.finalMask;


%% 8) Trials I. Crop blobs

croppedMask = cropBlobs(imageStruct.image.RGB,imageStruct.roofTypes(1).mask.fusedFilteredMask.total);


%% 8.1) More cropping if necessary

close all;

croppedMask = cropBlobs(imageStruct.image.RGB,croppedMask);

close all;

%% 9) Trials I. Add blobs


figure("Name", "Add Blobs");

imshow(imageStruct.image.RGB); % Asegúrate de que estás visualizando la imagen original.

hold on;


visualizeMask(imageStruct.image.RGB,croppedMask)


while true
  % Permite al usuario dibujar un polígono
  hPoly = drawpolygon;


  % Espera a que el usuario termine de dibujar
  wait(hPoly);


  % Obtiene las coordenadas del polígono seleccionado
  polygonPosition = hPoly.Position;
```

```
    % Crea una máscara binaria para el polígono seleccionado
    polygonMask = poly2mask(polygonPosition(:, 1), polygonPosition(:, 2), size(croppedMask, 1), size(croppedMask, 2));

    % Combina el polígono con la máscara previamente creada (croppedMask)
    croppedMask = croppedMask | polygonMask;

    % Cierra la figura de dibujo de polígonos
    delete(hPoly);

    % Pregunta al usuario si desea dibujar otro polígono
    disp("Presiona 'Enter' para dibujar otro polígono o cualquier otra tecla para salir.");
    key = waitforbuttonpress;

    if key == 0
        % El usuario presionó 'Enter' para dibujar otro polígono, continúa el bucle
        continue;
    else
        % El usuario presionó una tecla diferente, sale del bucle
        break;
    end
end

% Cierra las ventanas de visualización

close(1); % Cierra la ventana de la imagen original
close(2); % Cierra la ventana de visualización de la máscara

%% 9.1) More cropping if necessary
close all;
```

```
% Llama a cropBlobs y obtiene la máscara recortada
croppedMask = cropBlobs(imageStruct.image.RGB, croppedMask);

%% 10) Visualize it
%visualizeMask(imageStruct.image.RGB,croppedMask);
visualizeMask(croppedMask);
%% 10.5) Finally (don't) close the mask
% First, fuse super close blobs.
%imageStruct.roofTypes(1).mask.finalMask = myClose(croppedMask,ones(3),3,3);
imageStruct.roofTypes(1).mask.finalMask = croppedMask;
visualizeMask(imageStruct.image.RGB,imageStruct.roofTypes(1).mask.finalMask);
close all;

% Especifica el nombre del archivo de salida
outputPath = "D:\Universidad\UPV cuarto año\TFG\Matlab TFG\Imagen Inteligencia
Artificial\";  % Reemplazar con la ubicación deseada
outputFileName = sprintf('%sMASK_%s.tif', outputPath, imageName);

% Guarda la máscara como un archivo .tif
imwrite(imageStruct.roofTypes(1).mask.finalMask, outputFileName);

fprintf('Máscara guardada como %s\n', outputFileName);

%% 11) Finally close the mask
% First, fuse super close blobs.
imageStruct.roofTypes(1).mask.finalMask =
myFill(myClose(croppedMask,ones(3),3,3),2000);
visualizeMask(imageStruct.image.RGB,imageStruct.roofTypes(1).mask.finalMask);

% Second, fuse individual blobs.
lastLabels = bwlabel(imageStruct.roofTypes(1).mask.finalMask);
blobs = unique(lastLabels(:));
```

```
finalBlobs = zeros(size(lastLabels));
for i = blobs(blobs>0)'
   blobMask = ismember(lastLabels,i);
   blobClosedMask = myClose(blobMask,ones(3),4,4);
   finalBlobs = finalBlobs|blobClosedMask;
end

imageStruct.roofTypes(1).mask.finalMask = myFill(finalBlobs,2000);
close all;

% Especifica el nombre del archivo de salida
outputPath = "D:\Universidad\UPV cuarto año\TFG\Matlab TFG\Imagen Inteligencia Artificial\"; % Reemplaza con la ubicación deseada
outputFileName = sprintf('%sMASK_%s.tif', outputPath, imageName);

% Guarda la máscara como un archivo .tif
imwrite(imageStruct.roofTypes(1).mask.finalMask, outputFileName);

fprintf('Máscara guardada como %s\n', outputFileName);

%% 12) Georeference

% Ruta del archivo de imagen original
ruta_mascara = outputFileName;

% Cargar imagen original y obtener información de georreferenciación
info = geotiffinfo(ruta_imagen);

% Leer la máscara
mascara = imread(ruta_mascara);
```

```
% Genera un nombre de archivo para la máscara georreferenciada
[~, name, ext] = fileparts(ruta_mascara);
nombre_mascara_georreferenciada = sprintf('%s_GEO%s', name, ext);

% Ruta completa donde se guardará la máscara georreferenciada
ruta_mascara_georreferenciada = fullfile(outputPath, nombre_mascara_georreferenciada);

% Guardar la máscara georreferenciada con la misma información de georreferenciación que la imagen original
geotiffwrite(ruta_mascara_georreferenciada, mascara, info.SpatialRef, 'CoordRefSysCode', info.GeoTIFFTags.GeoKeyDirectoryTag.ProjectedCSTypeGeoKey);

fprintf('Máscara georreferenciada guardada como %s\n', ruta_mascara_georreferenciada);

%% 12.1) Visualize it

visualizeMask(imageStruct.image.RGB,imageStruct.roofTypes(1).mask.finalMask);

%% 13) Trials I. Calculate the area of each blob

XResolution=72/0.0254; %pxm; 72 es la resolución de las mascaras
YResolution=72/0.0254; %pxm; 72 es la resolución de las mascaras
blobLabels = bwlabel(imageStruct.roofTypes(1).mask.finalMask);
blobProps = regionprops(blobLabels,"Area");
blobAreas = (469286.667*[blobProps.Area])/(XResolution*YResolution);
imageStruct.roofTypes(1).area=[];
imageStruct.roofTypes(1).area = struct();
imageStruct.roofTypes(1).area.totalBlobArea = sum(blobAreas)*0.0001;
disp(imageStruct.roofTypes(1).area.totalBlobArea);

% Threshold areas lower than 50,75,100m2
```

```
imageStruct.roofTypes(1).mask.thresholdedMask = struct();

k = find(blobAreas>50);
imageStruct.roofTypes(1).mask.thresholdedMask.fifty = ismember(blobLabels,k);
imageStruct.roofTypes(1).area.fifty = sum(blobAreas(k));
figure("Name","Threshold50");
imshow(imageStruct.roofTypes(1).mask.thresholdedMask.fifty);
k = find(blobAreas>75);
imageStruct.roofTypes(1).mask.thresholdedMask.seventyFive = ismember(blobLabels,k);
imageStruct.roofTypes(1).area.seventyFive = sum(blobAreas(k));
figure("Name","Threshold75")
imshow(imageStruct.roofTypes(1).mask.thresholdedMask.seventyFive);
k = find(blobAreas>100);
imageStruct.roofTypes(1).mask.thresholdedMask.hundred = ismember(blobLabels,k);
imageStruct.roofTypes(1).area.hundred = sum(blobAreas(k));
figure("Name","Threshold100")
imshow(imageStruct.roofTypes(1).mask.thresholdedMask.hundred);

close all;

% 14) Sum area to rooftype sum. Initialise

sectionName = "Valencia_" + imageName;
roofType(1).area.(sectionName) = struct();
roofType(1).area.(sectionName).totalBlobArea = 0;
roofType(1).area.(sectionName).fifty = 0;
roofType(1).area.(sectionName).seventyFive = 0;
roofType(1).area.(sectionName).hundred = 0;

roofType(1).area.(sectionName).totalBlobArea = 0;
```

```
roofType(1).area.(sectionName).totalBlobArea =
roofType(1).area.(sectionName).totalBlobArea +
imageStruct.roofTypes(1).area.totalBlobArea;

roofType(1).area.(sectionName).fifty = (roofType(1).area.(sectionName).fifty +
imageStruct.roofTypes(1).area.fifty)/10000;

roofType(1).area.(sectionName).seventyFive = (roofType(1).area.(sectionName).seventyFive
+ imageStruct.roofTypes(1).area.seventyFive)/10000;

roofType(1).area.(sectionName).hundred = (roofType(1).area.(sectionName).hundred +
imageStruct.roofTypes(1).area.hundred)/10000;


% Crear un formato para imprimir los valores tabulados

formatString = 'Total Blob Area for %s:\t%.2f\nFifty for %s:\t%.2f\nSeventyFive for
%s:\t%.2f\nHundred for %s:\t%.2f\n';


% Imprimir los valores en el formato tabulado

fprintf(formatString, sectionName, ...

  roofType(1).area.(sectionName).totalBlobArea, ...

  sectionName, ...

  roofType(1).area.(sectionName).fifty, ...

  sectionName, ...

  roofType(1).area.(sectionName).seventyFive, ...

  sectionName, ...

  roofType(1).area.(sectionName).hundred);


% Guardar la cadena con los valores en el portapapeles

clipboard('copy', sprintf(formatString, sectionName, ...

  roofType(1).area.(sectionName).totalBlobArea, ...

  sectionName, ...

  roofType(1).area.(sectionName).fifty, ...

  sectionName, ...

  roofType(1).area.(sectionName).seventyFive, ...

  sectionName, ...

  roofType(1).area.(sectionName).hundred));
```

```
% Save the struct

save("MATLABData\imageMasks\"+imageStruct.name,"imageStruct",'-v7.3');

%% Functions

function IS = createMask(IS,RT,idx,medianFilt)

  if medianFilt
    imageHSV = IS.image.medianFiltered.HSV;
    imageRGB = IS.image.medianFiltered.RGB;
  else
    imageHSV = IS.image.HSV;
    imageHSV = IS.image.RGB;
  end

  nColorRegion = length(RT(idx).colorRegions);

  imageH = IS.image.HSV(:,:,1);
  imageS = IS.image.HSV(:,:,2);
  imageV = IS.image.HSV(:,:,3);

  imageR = IS.image.RGB(:,:,1);
  imageG = IS.image.RGB(:,:,2);
  imageB = IS.image.RGB(:,:,3);

  for i = 1:nColorRegion

    colorRegion = RT(idx).colorRegions(i);
```

```
    %minMax mask


    minMaxRoofMaskH = imageH < colorRegion.statistics.HSV.max(1) & imageH >
colorRegion.statistics.HSV.min(1);

    minMaxRoofMaskS = imageS < colorRegion.statistics.HSV.max(2) & imageS >
colorRegion.statistics.HSV.min(2);

    minMaxRoofMaskV = imageV < colorRegion.statistics.HSV.max(3) & imageV >
colorRegion.statistics.HSV.min(3);


    minMaxRoofMaskR = imageR < colorRegion.statistics.RGB.max(1) & imageR >
colorRegion.statistics.RGB.min(1);

    minMaxRoofMaskG = imageG < colorRegion.statistics.RGB.max(2) & imageG >
colorRegion.statistics.RGB.min(2);

    minMaxRoofMaskB = imageB < colorRegion.statistics.RGB.max(3) & imageB >
colorRegion.statistics.RGB.min(3);


    IS.roofTypes(idx).colorRegion(i).minMaxMask = struct();

    IS.roofTypes(idx).colorRegion(i).minMaxMask.R = minMaxRoofMaskR;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.G = minMaxRoofMaskG;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.B = minMaxRoofMaskB;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.RGB = minMaxRoofMaskR &
minMaxRoofMaskG & minMaxRoofMaskB;


    IS.roofTypes(idx).colorRegion(i).minMaxMask.H = minMaxRoofMaskH;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.S = minMaxRoofMaskS;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.V = minMaxRoofMaskV;

    IS.roofTypes(idx).colorRegion(i).minMaxMask.HSV = minMaxRoofMaskH &
minMaxRoofMaskS & minMaxRoofMaskV;


    IS.roofTypes(idx).colorRegion(i).minMaxMask.total =
IS.roofTypes(idx).colorRegion(i).minMaxMask.RGB & ...

      IS.roofTypes(idx).colorRegion(i).minMaxMask.HSV;


    % meanStd mask
```

```
    meanStdRoofMaskH = imageH < colorRegion.statistics.HSV.mean(1)+3*colorRegion.statistics.HSV.std(1) & imageH > colorRegion.statistics.HSV.mean(1)-3*colorRegion.statistics.HSV.std(1);

    meanStdRoofMaskS = imageS < colorRegion.statistics.HSV.mean(2)+3*colorRegion.statistics.HSV.std(2) & imageS > colorRegion.statistics.HSV.mean(2)-3*colorRegion.statistics.HSV.std(2);

    meanStdRoofMaskV = imageV < colorRegion.statistics.HSV.mean(3)+3*colorRegion.statistics.HSV.std(3) & imageV > colorRegion.statistics.HSV.mean(3)-3*colorRegion.statistics.HSV.std(3);


    meanStdRoofMaskR = imageR < colorRegion.statistics.RGB.mean(1)+3*colorRegion.statistics.RGB.std(1) & imageR > colorRegion.statistics.RGB.mean(1)-3*colorRegion.statistics.RGB.std(1);

    meanStdRoofMaskG = imageG < colorRegion.statistics.RGB.mean(2)+3*colorRegion.statistics.RGB.std(2) & imageG > colorRegion.statistics.RGB.mean(2)-3*colorRegion.statistics.RGB.std(2);

    meanStdRoofMaskB = imageB < colorRegion.statistics.RGB.mean(3)+3*colorRegion.statistics.RGB.std(3) & imageB > colorRegion.statistics.RGB.mean(3)-3*colorRegion.statistics.RGB.std(3);


    IS.roofTypes(idx).colorRegion(i).meanStdMask = struct();

    IS.roofTypes(idx).colorRegion(i).meanStdMask.R = meanStdRoofMaskR;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.G = meanStdRoofMaskG;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.B = meanStdRoofMaskB;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.RGB = meanStdRoofMaskR & meanStdRoofMaskG & meanStdRoofMaskB;


    IS.roofTypes(idx).colorRegion(i).meanStdMask.H = meanStdRoofMaskH;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.S = meanStdRoofMaskS;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.V = meanStdRoofMaskV;

    IS.roofTypes(idx).colorRegion(i).meanStdMask.HSV = meanStdRoofMaskH & meanStdRoofMaskS & meanStdRoofMaskV;


    IS.roofTypes(idx).colorRegion(i).meanStdMask.total = IS.roofTypes(idx).colorRegion(i).meanStdMask.RGB & ...
```

```
        IS.roofTypes(idx).colorRegion(i).meanStdMask.HSV;

    end

    meanStdMaskArray = [IS.roofTypes(idx).colorRegion.meanStdMask];
    minMaxMaskArray = [IS.roofTypes(idx).colorRegion.minMaxMask];

    IS.roofTypes(idx).mask = struct();
    IS.roofTypes(idx).mask.firstMask = struct();
    IS.roofTypes(idx).mask.firstMask.meanStd = max(cat(3,meanStdMaskArray.total),[],3);
    IS.roofTypes(idx).mask.firstMask.minMax = max(cat(3,minMaxMaskArray.total),[],3);

end


function openedIm = myClose(ID,SE,nErosion,nDilation)

    openedIm = ID;

    for i = 1:nDilation
        openedIm = imdilate(openedIm,SE);
    end
    for i = 1:nErosion
        openedIm = imerode(openedIm,SE);
    end
end



function filledFilteredMask = filterIndividualMask(initialMask, minSize1, minSize2)
```

```
filledMask = myFill(initialMask,0);
openedFilledMask = imopen(filledMask,[0 1 0 ; 1 1 1 ; 0 1 0]);
figure("Name","Opened filled mask");
imshow(filledMask);

% Suppress smaller blobs
% Extract blobs and blob area
imLabels = bwlabel(openedFilledMask);
props = regionprops(openedFilledMask,imLabels,'Area');

% Surpress small blobs        %%elimina agujeros pequeños dentro

k = find([props.Area] < minSize1);
suppressedLabels = ismember(imLabels,k);
filteredMask = xor(openedFilledMask,suppressedLabels);
figure("Name","First filtered mask");
imshow(filteredMask);

closedMask = myClose(filteredMask,[0 1 0 ; 1 1 1 ; 0 1 0],2,2);
filledFilteredMask = myFill(closedMask,0);

% Suppress smaller blobs again
% Extract blobs and blob area
imLabels = bwlabel(filledFilteredMask);
props = regionprops(filledFilteredMask,imLabels,'Area');

% Surpress small blobs
k = find([props.Area] < minSize2);
suppressedLabels = ismember(imLabels,k);
filteredMask = xor(filledFilteredMask,suppressedLabels);
figure("Name","Twice filtered mask");
```

```
    imshow(filteredMask);

    % Finally close the image
    closedMask = myClose(filteredMask,[0 1 0 ; 1 1 1 ; 0 1 0],4,4);
    filledFilteredMask = myFill(closedMask,0);
    figure("Name","Final filled closed filtered mask");
    imshow(filledFilteredMask);

end



function orMask = blobOR(mask1,mask2)

    mask1Labels = bwlabel(mask1);
    mask2Labels = bwlabel(mask2);
    blob1Values = unique(mask1Labels(mask2));
    blob2Values = unique(mask2Labels(mask1));
    orMask =
ismember(mask1Labels,blob1Values(blob1Values>0))|ismember(mask2Labels,blob2Values
(blob2Values>0));

end



function croppedBlobs = cropBlobs(baseImage, blobMask)
    f = figure('Name', 'Selected FAs for elimination');

    % Superponer la imagen base con la máscara utilizando imfuse
    fusedImage = imfuse(baseImage, blobMask, 'blend', 'Scaling', 'joint');
    imshow(fusedImage);
```

```
% Inicializar la máscara total
totalMask = false(size(blobMask));

% Espera hasta que se complete el dibujo del polígono o se presione Enter
while true
    % Permite al usuario dibujar un polígono
    h = drawpolygon;

    % Espera a que el usuario termine de dibujar
    wait(h);

    % Obtiene las coordenadas del polígono seleccionado
    polygonCoordinates = h.Position;

    % Crea una máscara binaria para el polígono seleccionado
    polygonMask = poly2mask(polygonCoordinates(:, 1), polygonCoordinates(:, 2), size(blobMask, 1), size(blobMask, 2));

    % Combina el polígono con la máscara previamente creada
    totalMask = totalMask | polygonMask;

    % Cierra la figura de dibujo de polígonos
    delete(h);

    % Pregunta al usuario si desea dibujar otro polígono
    disp("Presiona 'Enter' para dibujar otro polígono o cualquier otra tecla para salir.");
    key = waitforbuttonpress;

    if key == 0
        % El usuario presionó 'Enter' para dibujar otro polígono, continúa el bucle
```

```
        continue;
      else
        % El usuario presionó una tecla diferente, sale del bucle
        croppedBlobs = blobMask & ~totalMask;
        break;
      end
    end

    % Cierra la figura principal y la figura de la imagen superpuesta
    close(f);
end



function visualizeMask(img,mask)
    figure("Name","Mask visualisation")
    imshow(img);
    hold on
    mask3D = cat(3,mask*0.1,mask*0.6,mask*0.7);
    I = imshow(mask3D);
    I.AlphaData = mask*0.6;
end



function finalMask = myFill(mask,maxArea)

    filledMask = imfill(mask,"holes");
    holeMask = xor(mask,filledMask);

    labelledHoles = bwlabel(holeMask);
```

```
    props = regionprops(holeMask,labelledHoles,'Area');

    k = find([props.Area] < maxArea);
    holes2fill = ismember(labelledHoles,k);

    finalMask = mask|holes2fill;

end
```

DisposicionOptimizadaCompleta.mlx

```
clc
clear
T = readtable("Superficies.xlsx");
T=sortrows(T,"ID_1","ascend");

load Datos_Paneles.mat DimensionPaneles

EnergiaBarrio=0;
PotBarrio=0; %Conjunto de contadores para estudio
conti=0;
cont1=0;
cont2=0;
cont3=0;
cont4=0;
contCoplanar=0;
ContL=0;
ContW=0;
ContLoW=0;
TablaPot=zeros(5,2);
TablaPot(:,1)=DimensionPaneles.potencias;
```

```
for F=1:size(T,1)
   fprintf('Cubierta %d.\n',T.ID_1(F));
   Longitud=T.Largo(F);
   Ancho=T.Ancho(F);
   Pendiente=T.U_SLOPE(F);
   Azimut=T.Azimut(F);
```

Datos de la cubierta:

Selección de Perdidas:

```
Perdidas=14;

%Ancho=Ancho/cosd(Pendiente); %hecho en Qgis para todos

if Pendiente<=12
      [Mejor_arregloParalelo]=Paralelo(Longitud,Ancho,Azimut,Perdidas)
  if Azimut~=0 %lo llama si no está la cubierta alineada al sur
      [PotSur,conf,Inclinacion,EnergiaSur]=Sur(Longitud,Ancho,Azimut,Perdidas);
  end

else %coplanar
      [Mejor_arregloParalelo]=ParaleloCoplanar(Longitud,Ancho,Azimut,Pendiente,Perdidas)

end
```

```
if Azimut~=0 && Pendiente<=12
```

Dibujo Sur:

```
if conf=="W" || conf=="L o W"
   conf=1;
elseif conf=="L"
```

```
  conf=2;
end


Correción por si el azimut es negativo
if Azimut<0
  Azimut=Azimut+90;
  Invert=Longitud; %paso intermedio para invertir valores
  Longitud=Ancho;
  Ancho=Invert;
end


%datos panel (dimensiones): %se podira hacer con datos de DimensionPaneles

for i=1:size(DimensionPaneles,1)
  if PotSur==DimensionPaneles.potencias(i)
    LongitudPanel=DimensionPaneles.longitud(i);
    AnchuraPanel=DimensionPaneles.anchura(i);
    ProfundidadPanel=DimensionPaneles.profundidad(i);
  end
end


%datos panel (dimensiones en planta sacadas de algoritmo):
  if Pendiente<=12 %Caso con soporte-datos panel (dimensiones en planta sacadas de
algoritmo):
    if conf==1
      Aw=AnchuraPanel;
      Bw=cosd(Inclinacion)*LongitudPanel;
```

```
        %separacion por sombra:
        d=sind(Inclinacion)*LongitudPanel*1/tand(61-39); %1/tand(61-39)=2.4751

    elseif conf==2
        Aw=LongitudPanel;
        Bw=cosd(Inclinacion)*AnchuraPanel;
       %separacion por sombra:
        d=sind(Inclinacion)*AnchuraPanel*1/tand(61-39);
    end
end

if Pendiente>12 %Caso Coplanar
    %separacion por sombra:
        d=ProfundidadPanel/cosd(Pendiente)*1/tand(61-39); %1/tand(61-39)=2.4751
    if conf==1
        Aw=AnchuraPanel;
        Bw=LongitudPanel;



    elseif conf==2
        Aw=LongitudPanel;
        Bw=AnchuraPanel;
    end
end

%Separación en dirección del lado apoyado (entre columnas)
ed=0.02;

P1=[0 sind(Azimut)*Longitud];
P2=[cosd(90-Azimut)*Ancho sind(Azimut)*Longitud+sind(90-Azimut)*Ancho];
P3=[cosd(Azimut)*Longitud 0];
```

```
P4=[cosd(Azimut)*Longitud+cosd(90-Azimut)*Ancho sind(90-Azimut)*Ancho];

P1x=P1(1);
P1y=P1(2);

P2x=P2(1);
P2y=P2(2);

P3x=P3(1);
P3y=P3(2);

P4x=P4(1);
P4y=P4(2);

Dibujar rectangulo de la cubierta
plot([P1x, P2x,P4x], [P1y, P2y, P4y], 'b-', 'LineWidth', 2);
hold on
plot([P1x, P3x, P4x], [P1y, P3y, P4y], 'b-', 'LineWidth', 2);
hold off

axis equal
xlim([0 P4x])
ylim([0 P2y])
title("Cubierta")

xticks(0:5:P4x)
yticks(0:2.5:P2y)
```

Primera hilera de paneles

```
y=0;h=0;
```

```
%q1x=0,q1y=0, q2x=0, q2y=0, q3x=0, q3y=0, q4x=0, q4y=0;


  while h<Aw
  y=y+0.01;
  c1=(y-P3y)/sind(Azimut);
  c2=(y-P3y)/sind(90-Azimut);
  h=sqrt(c1^2+c2^2);

  % x=(y-p(2))/p(1)
  % alternativas para c1 y c2: sqrt((x-P3x)^2+(y-P3y)^2) o norm([x y]-[P3])

  end


  q1x=P3x-cosd(Azimut)*c1;
  q1y=y;

  q2x=q1x;
  q2y=y+Bw;

  q3x=q1x+Aw;
  q3y=q2y;

  q4x=q3x;
  q4y=q1y;

p12=polyfit([P1x P2x],[P1y P2y],1); %recta superior izquierda
p34=polyfit([P3x P4x],[P3y P4y],1); %recta inferior derecha
p24=polyfit([P2x P4x],[P2y P4y],1); %recta superior derecha
```

Salto si no cabe:

```
contador=0;
%sqrt((P2y-P4y)^2+(P4x-P2x)^2)
if q3y<polyval(p24,q3x) && q2y<polyval(p12,q2x) &&Aw<sqrt((P2y-P4y)^2+(P4x-P2x)^2)




plot([P1x, P2x,P4x], [P1y, P2y, P4y], 'b-', 'LineWidth', 2);
hold on
plot([P1x, P3x, P4x], [P1y, P3y, P4y], 'b-', 'LineWidth', 2);
hold on
plot([q1x, q2x,q3x], [q1y, q2y, q3y], 'r-', 'LineWidth', 0.5);
hold on
plot([q1x, q4x,q3x], [q1y, q4y, q3y], 'r-', 'LineWidth', 0.5);
hold off

title("Primer panel")
axis equal
```

Segunda hilera de paneles

```
plot([P1x, P2x,P4x], [P1y, P2y, P4y], 'b-', 'LineWidth', 2);
hold on
plot([P1x, P3x, P4x], [P1y, P3y, P4y], 'b-', 'LineWidth', 2);
hold on
plot([q1x, q2x,q3x], [q1y, q2y, q3y], 'r-', 'LineWidth', 0.5);
hold on
plot([q1x, q4x,q3x], [q1y, q4y, q3y], 'r-', 'LineWidth', 0.5);
hold on
axis equal
```

```
contador=1;
X=0;
y2=y+Bw+d;
g1x=P3x-cosd(Azimut)*(y2-P3y)/sind(Azimut);
g4x=g1x-ed;
%yline(y2);
%hold on



 while polyval(p12,g1x)>(y2+Bw) % g1x

      while polyval(p34,g4x+Aw+ed)<y2 && polyval(p24,g4x+Aw+ed)>(y2+Bw)
        c1=(y2-P3y)/sind(Azimut);
        c2=(y2-P3y)/sind(90-Azimut);
        %h=sqrt(c1^2+c2^2);




        g1x=P3x-cosd(Azimut)*c1+X;
        g1y=y2;

        g2x=g1x;
        g2y=y2+Bw;

        g3x=g1x+Aw;
```

```
        g3y=g2y;

        g4x=g3x;
        g4y=g1y;



         plot([g1x, g2x,g3x], [g1y, g2y, g3y], 'r-', 'LineWidth', 0.5);
         hold on
         plot([g1x, g4x,g3x], [g1y, g4y, g3y], 'r-', 'LineWidth', 0.5);
         hold on
        xticks(0:2.5:P4x)
        yticks(0:2.5:P2y)
        axis equal

         X=X+Aw+ed;
         %polyval(p,g4x)
         contador=contador+1;
      end
 X=0;
 y2=y2+Bw+d;




c1=(y2-P3y)/sind(Azimut);
g1x=P3x-cosd(Azimut)*c1;
g4x=g1x-ed;

%chequeos
 % polyval(p34,g4x+Aw+ed)
```

```
  %polyval(p24,g4x+Aw+ed)

 end
% hold off

%yline(y2+Bw)
% tand(Azimut)*(P2y-y2-Bw)
% polyval(p12,P2x-tand(Azimut)*(P2y-y2-Bw))

% tand(90-Azimut)*(P2y-y2-Bw)
%  polyval(p24,P2x+tand(90-Azimut)*(P2y-y2-Bw))

    %    s1x=P2x-tand(Azimut)*(P2y-y2)+X
    %    s1y=y2

    %    s2x=s1x
    %    s2y=s1y+Bw



     %   s3x=s1x+Aw
     %   s3y=s2y

     %   s4x=s3x
     %   s4y=s1y
```

Paneles superiores

```
% yline(y2)
```

```
X=0;
s1x=P2x-tand(Azimut)*(P2y-y2-Bw);
s4x=s1x-ed; %originalmente valor de s1x


 while polyval(p12,s1x)<(P2y) %%%%%%%%% es igual añadir || evitar bucle

      while polyval(p34,s4x+Aw+ed)<y2 && polyval(p24,s4x+Aw+ed)>(y2+Bw)
       %prueba
       %laquedebesermenor=polyval(p34,s4x+Aw+ed)
       % s4x+Aw+ed
       %y2


       %x=tand(Azimut)*(P2y-y2)
       s1x=P2x-tand(Azimut)*(P2y-y2-Bw)+X;
       s1y=y2;

       s2x=s1x;
       s2y=s1y+Bw;



       s3x=s1x+Aw;
       s3y=s2y;

       s4x=s3x;
       s4y=s1y;
```

```
        plot([s1x, s2x,s3x], [s1y, s2y, s3y], 'r-', 'LineWidth', 0.5);
        hold on
        plot([s1x, s4x,s3x], [s1y, s4y, s3y], 'r-', 'LineWidth', 0.5);
        hold on
       xticks(0:2.5:P4x)
       yticks(0:2.5:P2y)

        X=X+Aw+ed;
        %polyval(p,g4x)
        contador=contador+1;
    end
    X=0;
    y2=y2+Bw+d;
    s1x=P2x-tand(Azimut)*(P2y-y2-Bw);
    s4x=s1x-ed;
 end
hold off
title("Disposición al sur")
axis equal

end %fin de if de condicion no cabe ni uno

%fprintf('Caben %d paneles orientados paralelamente al borde de la
cubierta.\n',Mejor_arregloParalelo.("Mayor Pot (kW)")*1000/Mejor_arregloParalelo.("Pot panel
(W)"));
%fprintf('Caben %d paneles orientados al sur.\n',contador);

%fprintf('La potencia total será %4.2f kW.\n',contador*PotSur/1000);
%fprintf('La energía generada será %4.2f kWh/año.\n',EnergiaSur);
%Ang25=[113.61, 118.67, 152.4, 163.52, 182.99, 184.9, 192.27, 180.25, 151.21, 133.39,
108.21, 104.25];
```

```
%Energia por KW a 25º sum(Ang25,2)


end


Resultados:

if Pendiente<=12

   fprintf('Es preferible utilizar soportes para apoyar los paneles.');

  % fprintf('El mejor ángulo respecto del suelo para el arreglo al sur es: %4.2f',Inclinacion-
Pendiente);

   % fprintf('El mejor ángulo respecto del suelo para el arreglo paralelo es:
%4.2f',Mejor_arregloParalelo.("Ángulo")-Pendiente);


else

   fprintf('Es preferible apoyar los paneles de forma coplanar a la cubierta.');

   contCoplanar=contCoplanar+1;

end


fprintf('La mayor Energía total anual para los parametros planteados con orientación paralela
a la cubierta es:\n %4.2f kWh/año con una potencia instalada de %4.2f kW y usando %d
paneles de %d W.\n',Mejor_arregloParalelo.("Energía (kWh/año)"),
Mejor_arregloParalelo.("Mayor Pot (kW)"),Mejor_arregloParalelo.("Mayor Pot
(kW)")*1000/Mejor_arregloParalelo.("Pot panel (W)"),Mejor_arregloParalelo.("Pot panel
(W)"));


if Azimut==0

   fprintf('La cubierta está orientada al sur.')

   fprintf('Tanto el arreglo al Sur como el paralela a la cubierta serían iguales.')

   EnergiaBarrio=EnergiaBarrio+Mejor_arregloParalelo.("Energía (kWh/año)");

   PotBarrio=PotBarrio+Mejor_arregloParalelo.("Mayor Pot (kW)");

   cont1=cont1+1;

   if Mejor_arregloParalelo.("Apoyo")=="L" %contador apoyo

      ContL=ContL+1;

   elseif Mejor_arregloParalelo.("Apoyo")=="W"

      ContW=ContW+1;
```

```
  elseif Mejor_arregloParalelo.("Apoyo")=="L o W"
    ContLoW=ContLoW+1;
  end
  for i=1:5 %contador por panel
    if Mejor_arregloParalelo.("Pot panel (W)")==DimensionPaneles.potencias(i)
      TablaPot(i,2)=TablaPot(i,2)+1;
    end
  end
elseif Pendiente<=12 %no al sur ni coplanar
  fprintf('La mayor Energía total anual para los parametros planteados con orientación al sur
es:\n %4.2f kWh/año con una potencia instalada de %4.2f kW y usando %d paneles de %d
W.\n',EnergiaSur, contador*PotSur/1000, contador, PotSur);
  if EnergiaSur>=Mejor_arregloParalelo.("Energía (kWh/año)")
    fprintf('La mayor Energía total anual se dará con orientación al sur. \n');
    EnergiaBarrio=EnergiaBarrio+EnergiaSur;
    PotBarrio=PotBarrio+contador*PotSur/1000;
    cont2=cont2+1;

    if Mejor_arregloParalelo.("Apoyo")=="L" %contador apoyo
    ContL=ContL+1;
    elseif Mejor_arregloParalelo.("Apoyo")=="W"
    ContW=ContW+1;
    elseif Mejor_arregloParalelo.("Apoyo")=="L o W"
    ContLoW=ContLoW+1;
    end

    for i=1:5 %contador por panel
      if PotSur==DimensionPaneles.potencias(i)
        TablaPot(i,2)=TablaPot(i,2)+1;
      end
    end
  else
```

```
    fprintf('La mayor Energía total anual se dará con orientación paralela a la cubierta. \n ');
    EnergiaBarrio=EnergiaBarrio+Mejor_arregloParalelo.("Energía (kWh/año)");
    PotBarrio=PotBarrio+Mejor_arregloParalelo.("Mayor Pot (kW)");
    cont3=cont3+1;

    if Mejor_arregloParalelo.("Apoyo")=="L" %contador apoyo
    ContL=ContL+1;
    elseif Mejor_arregloParalelo.("Apoyo")=="W"
    ContW=ContW+1;
    elseif Mejor_arregloParalelo.("Apoyo")=="L o W"
    ContLoW=ContLoW+1;
    end

    for i=1:5 %contador por panel
      if Mejor_arregloParalelo.("Pot panel (W)")==DimensionPaneles.potencias(i)
        TablaPot(i,2)=TablaPot(i,2)+1;
      end
    end
  end
else
  fprintf('La mayor Energía total anual se dará con orientación paralela a la cubierta. \n ');
  EnergiaBarrio=EnergiaBarrio+Mejor_arregloParalelo.("Energía (kWh/año)");
  PotBarrio=PotBarrio+Mejor_arregloParalelo.("Mayor Pot (kW)");
  cont4=cont4+1;

  if Mejor_arregloParalelo.("Apoyo")=="L" %contador apoyo
    ContL=ContL+1;
  elseif Mejor_arregloParalelo.("Apoyo")=="W"
    ContW=ContW+1;
  elseif Mejor_arregloParalelo.("Apoyo")=="L o W"
    ContLoW=ContLoW+1;
```

```
  end

  for i=1:5 %contador por panel
    if Mejor_arregloParalelo.("Pot panel (W)")==DimensionPaneles.potencias(i)
      TablaPot(i,2)=TablaPot(i,2)+1;
    end
  end
end

%waitforbuttonpress
conti=conti+1;
end
```

Sur.mlx

function [Pot,Apoyo,ElAngulo,Energia]=Sur(LongitudCubierta,AnchuraCubierta,Orientacion,Perdidas)

Table of Contents

Datos de entrada:

```
%LongitudCubierta=20
%AnchuraCubierta=5
%Orientacion=-30 % "α" ángulo de medianera entre cubierta y sur
%AngCompl=Orientacion+90; % "β" ángulo complementario

%Separación en dirección del lado apoyado (entre columnas)
%ed=0.02;
```

%Separación en dirección perpendicular al lado apoyado "d"

%d=ancho*sen(algulo)/tan(61-latitud)=ancho*sen(algulo)*cteSep

%Constante de separación en dirección perpendicular al lado apoyado

%cteSep=1/tand(61-39);

%número de columnas y filas máximo (a elegir pero que sean iguales)

%nCol=(0:30)';

%nFilas=(0:30)';

%Ángulos de inclinación del panel

angulo=25:5:45;

load Datos_Paneles.mat DimensionPaneles

"W" corresponde a la configuración del panel apoyado en el lado corto mientras que "L" corresponde a la configuración del panel apoyado en el lado largo

Cálculo de la potencia total de cada tipo de panel según la inclinación y con orientación α (conf=1)

conf=1;

% vectores de 1x5 para cada angulo

PotTotal150=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,1,conf,angulo,Orientacion);

PotTotal180=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,2,conf,angulo,Orientacion);

PotTotal200=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,3,conf,angulo,Orientacion);

```
PotTotal510=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,4,conf,angulo,Ori
entacion);

PotTotal670=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,5,conf,angulo,Ori
entacion);


conf=2;

PotTotal150(:,2)=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,1,conf,angulo,
Orientacion);

PotTotal180(:,2)=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,2,conf,angulo,
Orientacion);

PotTotal200(:,2)=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,3,conf,angulo,
Orientacion);

PotTotal510(:,2)=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,4,conf,angulo,
Orientacion);

PotTotal670(:,2)=AlSur(DimensionPaneles,LongitudCubierta,AnchuraCubierta,5,conf,angulo,
Orientacion);


Presentación en forma de tabla para mejor entendimiento

PotTotalconf1=[PotTotal150,PotTotal180,PotTotal200,PotTotal510,PotTotal670];

tablaPotTotal=array2table(PotTotalconf1);

viejosNombres = 1:width(tablaPotTotal);

nuevosNombres=["Pot 150 W","Pot 150 L","Pot 180 W","Pot 180 L","Pot 200 W","Pot 200 L","Pot
510 W","Pot 510 L","Pot 670 W","Pot 670 L"];

tablaPotTotalconf1=renamevars(tablaPotTotal,viejosNombres,nuevosNombres);

tablaPotTotalconf1.Angulos=[25,30,35,40,45]';
```

Relación área-Potencia (m^2/kW) de cada configuración

```
%RelacionAreaPotconf1=(AnchuraCubierta*LongitudCubierta)./PotTotalconf1;


TablaRelacionAreaPotconf1=(AnchuraCubierta*LongitudCubierta)./tablaPotTotalconf1;

TablaRelacionAreaPotconf1.Angulos=[25,30,35,40,45]';
```

Optimización

Comparación 1:

Encontrar la mayor potencia total para cada tipo de panel, cada inclinación y cada orientación indicando tipo de apoyo (L=0, W=1).

[Comparacion1conf1,~]=optimizacion1(PotTotalconf1); %%aunque se llame conf1 contiene ambas

Comparación 2:

Encontrar la mayor potencia total de cada inclinación indicando panel y tipo de apoyo.

[Comparacion2conf1,~]=optimizacion2(Comparacion1conf1);

Comparación 3:

Hallar potencias equivalentes con respecto a la orientación α de cada inclinación indicando panel y tipo de apoyo.

%No tiene sentido aqui

%[Comparacion3,tablaComparacion3]=optimizacion3(Comparacion2conf1,Comparacion2conf2,Orientacion);

Mejor arreglo:

%potencia en kilovatios por cada mes obtenido de PVGIS para Illa Perduda

Ang25=(100-Perdidas)/100*1811.51;

Ang30=(100-Perdidas)/100*1832.36;

Ang35=(100-Perdidas)/100*1842.36;

Ang40=(100-Perdidas)/100*1841.4;

Ang45=(100-Perdidas)/100*1829.09;

%energía anual con 1 kW de potencia instalada según la inclinación con azimut 0

%BaseDatos=table(Ang25, Ang30, Ang35, Ang40, Ang45);

Comparacion4(:,1)=Comparacion2conf1(:,1);

```
Comparacion4(1,2)=Comparacion4(1,1)*Ang25;
Comparacion4(2,2)=Comparacion4(2,1)*Ang30;
Comparacion4(3,2)=Comparacion4(3,1)*Ang35;
Comparacion4(4,2)=Comparacion4(4,1)*Ang40;
Comparacion4(5,2)=Comparacion4(5,1)*Ang45;

Comparacion4(:,3:5)=Comparacion2conf1(:,2:4);

tablaComparacion4=array2table(Comparacion4);
viejosNombres = 1:width(tablaComparacion4);
nuevosNombres=["Mayor Pot (kW)","Energía (kWh/año)","Pot panel (W)","Apoyo","Ángulo"];
tablaComparacion4=renamevars(tablaComparacion4,viejosNombres,nuevosNombres);



tablaComparacion4.Apoyo=categorical(tablaComparacion4.Apoyo,[0 1 2],["L","W","L o W"]);



[~,index]=max(Comparacion4(:,2));
Mejor_arregloSur=tablaComparacion4(index,:)
%Mejor_arreglo=removevars(Mejor_arreglo,"Mayor Pot")

Pot=Mejor_arregloSur.("Pot panel (W)");
Apoyo=Mejor_arregloSur.Apoyo;
ElAngulo=Mejor_arregloSur.("Ángulo");
Energia=Mejor_arregloSur.("Energía (kWh/año)");
```

end

Paralelo.mlx

function
[Mejor_arregloParalelo]=Paralelo(LongitudCubierta,AnchuraCubierta,Orientacion,Perdidas)

Mejor arreglo:

Datos de entrada:

%LongitudCubierta=15

%AnchuraCubierta=5

%Orientacion=-30 % "α" ángulo de medianera entre cubierta y sur

%AngCompl=Orientacion+90; % "β" ángulo complementario

%Separación en dirección del lado apoyado (entre columnas)

ed=0.02;

%Separación en dirección perpendicular al lado apoyado "d"

%d=ancho*sen(algulo)/tan(61-latitud)=ancho*sen(algulo)*cteSep

%Constante de separación en dirección perpendicular al lado apoyado

cteSep=1/tand(61-39);

%número de columnas y filas máximo (a elegir pero que sean iguales)

%nCol=(0:30)';

%nFilas=(0:30)';

%Ángulos de inclinación del panel

angulo=25:5:45;

load Datos_Paneles.mat DimensionPaneles

Cálculo de la potencia total de cada tipo de panel según la inclinación y con orientación α (conf=1)

```
conf=1;


PotTotal150=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,1,conf,angulo);
PotTotal180=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,2,conf,angulo);
PotTotal200=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,3,conf,angulo);
PotTotal510=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,4,conf,angulo);
PotTotal670=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,5,conf,angulo);


Presentación en forma de tabla para mejor entendimiento
PotTotalconf1=[PotTotal150,PotTotal180,PotTotal200,PotTotal510,PotTotal670];
tablaPotTotal=array2table(PotTotalconf1);
viejosNombres = 1:width(tablaPotTotal);
nuevosNombres=["Pot 150 W","Pot 150 L","Pot 180 W","Pot 180 L","Pot 200 W","Pot 200 L","Pot
510 W","Pot 510 L","Pot 670 W","Pot 670 L"];
tablaPotTotalconf1=renamevars(tablaPotTotal,viejosNombres,nuevosNombres);
tablaPotTotalconf1.Angulos=[25,30,35,40,45]';
tablaPotTotalconf1;
```

Cálculo de la potencia total de cada tipo de panel según la inclinación y con orientación β (conf=2)

```
conf=2;


PotTotal150=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,1,conf,angulo);
PotTotal180=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,2,conf,angulo);
PotTotal200=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,3,conf,angulo);
PotTotal510=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,4,conf,angulo);
PotTotal670=potenciaTotalNew(LongitudCubierta,AnchuraCubierta,5,conf,angulo);
Presentación en forma de tabla para mejor entendimiento
PotTotalconf2=[PotTotal150,PotTotal180,PotTotal200,PotTotal510,PotTotal670];
tablaPotTotal=array2table(PotTotalconf2);
viejosNombres = 1:width(tablaPotTotal);
nuevosNombres=["Pot 150 W","Pot 150 L","Pot 180 W","Pot 180 L","Pot 200 W","Pot 200 L","Pot
510 W","Pot 510 L","Pot 670 W","Pot 670 L"];
```

tablaPotTotalconf2=renamevars(tablaPotTotal, viejosNombres,nuevosNombres);

tablaPotTotalconf2.Angulos=[25,30,35,40,45]';

%tablaPotTotalconf2

Relación área-Potencia (m^2/kW) de cada configuración

RelacionAreaPotconf1=(AnchuraCubierta*LongitudCubierta)./PotTotalconf1;

RelacionAreaPotconf2=(AnchuraCubierta*LongitudCubierta)./PotTotalconf2;

TablaRelacionAreaPotconf1=(AnchuraCubierta*LongitudCubierta)./tablaPotTotalconf1;

TablaRelacionAreaPotconf1.Angulos=[25,30,35,40,45]';

TablaRelacionAreaPotconf2=(AnchuraCubierta*LongitudCubierta)./tablaPotTotalconf2;

TablaRelacionAreaPotconf2.Angulos=[25,30,35,40,45]';

Optimización

Comparación 1:

Encontrar la mayor potencia total para cada tipo de panel, cada inclinación y cada orientación indicando tipo de apoyo (L=0, W=1).

[Comparacion1conf1,tablaComparacion1conf1]=optimizacion1(PotTotalconf1);

[Comparacion1conf2,tablaComparacion1conf2]=optimizacion1(PotTotalconf2);

%tablaComparacion1conf1

%tablaComparacion1conf2

Comparación 2:

Encontrar la mayor potencia total de cada inclinación indicando panel y tipo de apoyo.

[Comparacion2conf1,tablaComparacion2conf1]=optimizacion2(Comparacion1conf1);

[Comparacion2conf2,tablaComparacion2conf2]=optimizacion2(Comparacion1conf2);

tablaComparacion2conf1;

tablaComparacion2conf2;

Comparación 3:

Hallar potencias equivalentes con respecto a la orientación α de cada inclinación indicando panel y tipo de apoyo.

[Comparacion3,tablaComparacion3]=optimizacion3(Comparacion2conf1,Comparacion2conf2,Orientacion,Perdidas);

Comparacion3;

```
tablaComparacion3;
```

Comparación 4:

Hallar maximo entre α y β equivalente y el equivalente de esa potencia con respecto a la inclinación 25° indicando panel y tipo de apoyo.

```
Comparacion4=zeros(1,6);


if max(Comparacion3(:,2))>max(Comparacion3(:,4))
   [Comparacion4(1,2),Index]=max(Comparacion3(:,2));
   Comparacion4(1,1)=Comparacion3(Index,1);
   Comparacion4(1,3)=1; %α
elseif max(Comparacion3(:,2))<max(Comparacion3(:,4))
   [Comparacion4(1,2),Index]=max(Comparacion3(:,4));
   Comparacion4(1,1)=Comparacion3(Index,3);
   Comparacion4(1,3)=2; %β
else
   [Comparacion4(1,2),Index]=max(Comparacion3(:,2));
   Comparacion4(1,1)=Comparacion3(Index,1);
   Comparacion4(1,3)=3; %α o β
end


 Comparacion4(1,4:6)=Comparacion3(Index,5:7);



tablaComparacion4=array2table(Comparacion4);
viejosNombres = 1:width(tablaComparacion4);
nuevosNombres=["Mayor Pot (kW)","Energía (kWh/año)","Conf","Pot panel (W)","Apoyo","Ángulo"];
tablaComparacion4=renamevars(tablaComparacion4,viejosNombres,nuevosNombres);



tablaComparacion4.Apoyo=categorical(tablaComparacion4.Apoyo,[0 1 2],["L","W","L o W"]);
```

tablaComparacion4.Conf=categorical(tablaComparacion4.Conf,[1 2 3],["α","β","α o β"]);

Mejor arreglo:

Mejor_arregloParalelo=tablaComparacion4;

ParaleloCoplanar.mlx

function [Mejor_arregloParalelo]=ParaleloCoplanar(LongitudCubierta,AnchuraCubierta,Azimut,Pendiente,Perdidas)

"W" corresponde a la configuración del panel apoyado en el lado corto mientras que "L" corresponde a la configuración del panel apoyado en el lado largo

Cálculo de la potencia total de cada tipo de panel y apoyo

%Son 1x2

[PotTotal150]=potenciaTotalCoplanar(LongitudCubierta,AnchuraCubierta,Pendiente,1);

[PotTotal180]=potenciaTotalCoplanar(LongitudCubierta,AnchuraCubierta,Pendiente,2);

[PotTotal200]=potenciaTotalCoplanar(LongitudCubierta,AnchuraCubierta,Pendiente,3);

[PotTotal510]=potenciaTotalCoplanar(LongitudCubierta,AnchuraCubierta,Pendiente,4);

[PotTotal670]=potenciaTotalCoplanar(LongitudCubierta,AnchuraCubierta,Pendiente,5);

Presentación en forma de tabla para mejor entendimiento

PotTotal=[PotTotal150,PotTotal180,PotTotal200,PotTotal510,PotTotal670];

tablaPotTotal=array2table(PotTotal);

viejosNombres = 1:width(tablaPotTotal);

nuevosNombres=["Pot 150 W","Pot 150 L","Pot 180 W","Pot 180 L","Pot 200 W","Pot 200 L","Pot 510 W","Pot 510 L","Pot 670 W","Pot 670 L"];

tablaPotTotal=renamevars(tablaPotTotal,viejosNombres,nuevosNombres);

%tablaPotTotal

Optimización

Comparación 1:

Encontrar la mayor potencia total para cada tipo de panel, cada inclinación y cada orientación indicando tipo de apoyo (L=0, W=1).

[Comparacion1,tablaComparacion1]=optimizacion1(PotTotal);

tablaComparacion1;

Comparación 2:

Encontrar la mayor potencia total de cada inclinación indicando panel y tipo de apoyo.

[Comparacion2,tablaComparacion2]=optimizacion2Coplanar(Comparacion1);

tablaComparacion2;

Comparación 3:

Hallar potencias equivalentes con respecto a la orientación α de cada inclinación indicando panel y tipo de apoyo.

%[Comparacion3,tablaComparacion3]=optimizacion3(Comparacion2,Comparacion2conf2,Orientacion);

%tablaComparacion3

Comparación 4:

Hallar maxximo entre α y β equivalente y el equivalente de esa potencia con respecto a la inclinación 25° indicando panel y tipo de apoyo.

Comparacion4(1,1)=Comparacion2(:,1);

Comparacion4(1,3:4)=Comparacion2(:,2:3);

tablaComparacion4=array2table(Comparacion4);

viejosNombres = 1:width(tablaComparacion4);

nuevosNombres=["Mayor Pot","Pot 25° equiv (kW)","Pot panel (W)","Apoyo"];

tablaComparacion4=renamevars(tablaComparacion4,viejosNombres,nuevosNombres);

tablaComparacion4.Apoyo=categorical(tablaComparacion4.Apoyo,[0 1 2],["L","W","L o W"]);

Mejor arreglo:

%potencia en kilovatios por cada mes obtenido de PVGIS para Illa Perduda

```
matriz=zeros(7,19);


matriz(1,:)=(100-Perdidas)/100*[1372.68, 1409.2, 1447.02, 1484.08, 1519.81, 1553.29, 1583.72, 1610.19, 1632.55, 1650.2, 1662.46, 1669, 1669.23, 1663.16, 1651.15, 1633.53, 1610.83, 1583.65, 1552.89];

matriz(2,:)=(100-Perdidas)/100*[1327.93, 1376.24, 1424.56, 1471.43, 1517.18, 1559.98, 1598.8, 1632.78, 1661.93, 1684.7, 1699.89, 1708.02, 1708.31, 1700.77, 1685.71, 1663.89, 1634.59, 1599.96, 1560.78 ];

matriz(3,:)=(100-Perdidas)/100*[1281.63, 1340.61, 1400.19, 1457.69, 1511.23, 1561.29, 1606.98, 1647.08, 1681.2, 1708.93, 1728.03, 1736.98, 1737.27, 1728.91, 1711.18, 1683.73, 1648.98, 1608.52, 1561.76 ];

matriz(4,:)=(100-Perdidas)/100*[1236.21, 1304.55, 1371.41, 1436.97, 1498.59, 1555.52, 1608.56, 1654.62, 1692.17, 1723.06, 1745.53, 1755.93, 1756.35, 1747.31, 1725.8, 1694.47, 1656.46, 1611.37, 1557.38 ];

matriz(5,:)=(100-Perdidas)/100*[1188.25, 1265.79, 1341.02, 1412.82, 1481.02, 1543.16, 1600.32, 1652.1, 1695.92, 1727.64, 1752.56, 1765.08, 1765.7, 1754.68, 1730.73, 1698.68, 1655.95, 1603.75, 1544.57 ];

matriz(6,:)=(100-Perdidas)/100*[1142.54, 1225.67, 1307.63, 1384.66, 1458.7, 1527.03, 1586.8, 1640.28, 1688.04, 1724.11, 1749.39, 1763.37, 1764.37, 1751.81, 1727.48, 1692.47, 1644.78, 1588.99, 1527.53 ];

matriz(7,:)=(100-Perdidas)/100*[1093.68, 1183.06, 1269.6, 1352.08, 1428.58, 1500.64, 1566.4, 1621.57, 1670.6, 1710.83, 1736.22, 1751.31, 1752.37, 1738.97, 1714.98, 1675.61, 1625.73, 1570.5, 1503.14 ];


p15=polyfit(-112:10:68, matriz(1,:),4);

p20=polyfit(-112:10:68, matriz(2,:),4);

p25=polyfit(-112:10:68, matriz(3,:),4);

p30=polyfit(-112:10:68, matriz(4,:),4);

p35=polyfit(-112:10:68, matriz(5,:),4);

p40=polyfit(-112:10:68, matriz(6,:),4);

p45=polyfit(-112:10:68, matriz(7,:),4);


if Pendiente>12 && Pendiente<=20

   Energia= polyval(p15,Azimut);

elseif Pendiente>20 && Pendiente<=25

   Energia= polyval(p20,Azimut);
```

```
elseif Pendiente>25 && Pendiente<=30
   Energia= polyval(p25,Azimut);
elseif Pendiente>30 && Pendiente<=35
   Energia= polyval(p30,Azimut);
elseif Pendiente>35 && Pendiente<=40
   Energia= polyval(p35,Azimut);
elseif Pendiente>40 && Pendiente<=45
   Energia= polyval(p40,Azimut);
elseif Pendiente>45
   Energia= polyval(p45,Azimut);

end

Comparacion4(1,2)=Comparacion4(1,1)*Energia;




tablaComparacion4=array2table(Comparacion4);
viejosNombres = 1:width(tablaComparacion4);
nuevosNombres=["Mayor Pot (kW)","Energía (kWh/año)","Pot panel (W)","Apoyo"];
tablaComparacion4=renamevars(tablaComparacion4,viejosNombres,nuevosNombres);



tablaComparacion4.Apoyo=categorical(tablaComparacion4.Apoyo,[0 1 2],["L","W","L o W"]);

[~,index]=max(Comparacion4(:,2));
Mejor_arregloParalelo=tablaComparacion4(index,:);
%Mejor_arreglo=removevars(Mejor_arreglo,"Mayor Pot")
end
```