

Paquete de python para la fusión de imágenes satelitales sobre arquitecturas homogéneas (CPU) y heterogéneas (CPU/GPU) (`satellite-images-fusion`), utilizando las técnicas: Gram Schmit, Modulación de altas frecuencias, Transformación High Pass Filter y

Valor medio simple

Manual de Usuario

Tabla de contenido

1	Descripción	2
2	Requisitos	2
3	Descripción de funcionalidades	2
3.1.1	<code>satellite_images_fusion.invoker</code>	2

1 Descripción

La fusión de imágenes satelitales es un proceso digital que permite reunir en una imagen procesada la riqueza espectral de una imagen multiespectral y la resolución espacial de una imagen pancromática. Para llevar esto a cabo se cuenta con algunos algoritmos definidos previamente como técnica de valor medio, filtro paso alto, gran Schmit y modulación de altas frecuencias, entre otras. A continuación, se da una breve descripción de estas técnicas.

Gram Schmit

La imagen pancromática se combina con el resto de las bandas de menor resolución espacial mediante una transformación matemática de los datos originales. La transformación Gram-Schmidt es una técnica común en álgebra lineal y análisis multivariante; en este caso se aplica con objeto de ortogonalizar las bandas de una imagen digital. La ortogonalización de los datos elimina la información redundante que contienen, algo bastante acusado en las imágenes de satélite, especialmente en las bandas del espectro visible. En el caso de que exista una correlación perfecta entre las bandas de entrada, la ortogonalización Gram-Schmidt produciría una banda en la que todos los píxeles tendrían valor 0. En el caso más realista de que las correlaciones entre las bandas sean muy altas, el resultado será una imagen con valores muy bajos.

Modulación de altas frecuencias

Se trata de una variación de los denominados métodos de fusión en el dominio espacial, cuya idea es la de transferir las altas frecuencias de la imagen de alta resolución a la imagen de baja resolución. Como ya es sabido, las altas frecuencias contienen la información relativa a los detalles de una imagen y pueden extraerse mediante operaciones de filtrado o convolución. Básicamente, estos métodos consisten en la suma de las altas frecuencias de la imagen pancromática a cada banda de la imagen, la eficiencia de este filtro se basa en la existencia de una correlación radiométrica elevada entre los componentes de altas frecuencias de ambas imágenes.

Un caso particular de los métodos espaciales es el llamado modulación de las altas frecuencias (HFM, High Frequency Modulation). Consiste en la multiplicación de cada banda de baja resolución (MS)i, por la imagen de alta resolución (Pan), se obtienen mejores resultados si el filtro paso-bajo se diseña de forma que se ajuste a la función de dispersión puntual (PSF, Point Spread Function) relativa entre ambas imágenes. La PSF es una función de ponderación sobre la señal electrónica que se produce a la salida de los detectores y que depende de factores ópticos, del movimiento de la imagen en su adquisición, del propio detector y de los componentes electrónicos que entran en juego durante el proceso.

Transformación High Pass Filter - HPF

Este método consiste en añadir la información espacial de la banda pancromática a la información multiespectral de menor resolución espacial aplicando un filtro de paso alto en combinación con una operación de álgebra de mapas. El

funcionamiento del algoritmo, descrito en Leica-Geosystems (2006), consta de cinco pasos:

1. Calcular el parámetro R a partir del tamaño del píxel de la capa pancromática (t_{pan}) y de la multiespectral.
2. Aplicar un filtro de paso alto a la imagen pancromática; el tamaño de la ventana de filtrado es proporcional a R. Todos los elementos de la ventana de filtrado toman el valor -1, con la excepción del valor central. Existen tres posibilidades para este valor; el menor de los tres es el que se utiliza por defecto y es el que se ha empleado en este trabajo.
3. Remuestrear la imagen multiespectral al tamaño del píxel de la resolución espacial de la imagen filtrada.
4. Sumar la imagen filtrada a las capas multiespectrales. Pero antes, la imagen filtrada se pondera en función de la desviación típica de la imagen multiespectral y el valor de R, a este factor de ponderación se le denomina W, donde σ_{mis} es la desviación típica de cada una de las bandas; σ_{pan} es la desviación típica de la imagen filtrada y M es un factor que determina la intensidad en la aplicación del filtro. Donde P_{out} es el píxel de salida de cada una de las bandas multiespectrales ya fusionadas; P_{in} es el píxel de entrada de cada una de las bandas multiespectrales originales y $PHPF$ es el píxel de la imagen filtrada.
5. Expandir linealmente los niveles digitales (ND) de la imagen multiespectral fusionada; esta operación reescalía la imagen resultante, de forma que la media y la desviación típica coincida con las de la imagen original.

Valor medio simple

El método de transformación de valor medio aplica el promedio de valor medio simple, a cada una de las combinaciones de banda de salida. Por ejemplo, la nueva banda fusionada será la suma de la banda original con la banda pancromática dividido en dos. Así sucesivamente según el número de bandas que conforme la imagen multiespectral.

Por otra parte, Una vez se ha aplicado alguno de los algoritmos de fusión, es necesario realizar una evaluación de la riqueza espectral y espacial de la imagen generada por los métodos de fusión expuestos anteriormente, es posible se utilizar algunos índices como mean square error (mse), root mean square error (rmse), Bias y el índice de correlación.

En este orden de ideas, al implementar los algoritmos de fusión de imágenes satelitales en forma serial, es decir, realizando su ejecución exclusivamente en CPU, se presentan tiempos elevados al utilizar imágenes de dimensiones superiores, es por esto que este proyecto busca realizar la implementación de las transformadas mencionadas anteriormente mediante procesamiento heterogéneo CPU/GPU con el fin de optimizar los tiempos de ejecución para este algoritmo. Así mismo, se tiene como objetivo proporcionar herramientas para la comparación en términos de tiempos de ejecución y evaluación de la calidad de la imagen obtenida.

2 Requisitos

Para el funcionamiento de este software, es necesario el siguiente hardware:

1. Tarjeta gráfica NVIDIA, solo si desea realizar la fusión en GPU

Librerías necesarias para ejecutar:

1. Python 3.7 o mayor: <https://www.python.org/download/releases/3.6/>
2. Numpy 1.21.6: <https://pypi.org/project/numpy/>
3. Scikit-image 0.18.3: <https://pypi.org/project/scikit-image/>
4. Scipy 1.7.3: <https://pypi.org/project/scipy/>
5. Pycuda 2022.1: <https://pypi.org/project/pycuda/>
6. Scikit-cuda 0.5.3: <https://scikit-cuda.readthedocs.io/en/latest/install.html>
7. Cupy-cuda111 10.2.0: <https://pypi.org/project/cupy/>

3 Descripción de funcionalidades

Esta sección, tiene como propósito definir el módulo invoker, el cual se encarga de realizar la fusión de imágenes y el cálculo de los índices matemático-estadísticos de comparación de calidad.

Nota: Para hacer uso de este paquete la imagen multiespectral y pancromática deben tener la misma dimensión cuadrada y ancho de píxel.

3.1.1 satellite_images_fusion.invoker

3.1.1.1 Generación de fusión de imágenes

Realiza la fusión de imágenes satelitales a partir de la definición y configuración de parámetros determinada por el usuario.

satellite_images_fusion.invoker.generate_fusion_images (*multispectral_path*,
panchromatic_path, *method_fusion*, *fusioned_image_path*, *device_fusion="cpu"*,
geographical_info=True)

Parámetros:

<i>multispectral:</i>	<i>string</i>
	Dirección local de la imagen multiespectral que se desea cargar.
<i>panchromatic:</i>	<i>string</i>
	Dirección local de la imagen pancromática que se desea cargar.
<i>method_fusion:</i>	<i>string</i>
	Método de fusión que se desea aplicar a las imágenes multiespectral y pancromática. Las opciones son: <i>high_pass_filter</i> , <i>mean_value</i> , <i>high_frequency_modulation</i> y <i>gram_schmidt</i> .
<i>fusioned_image_path:</i>	<i>string</i>

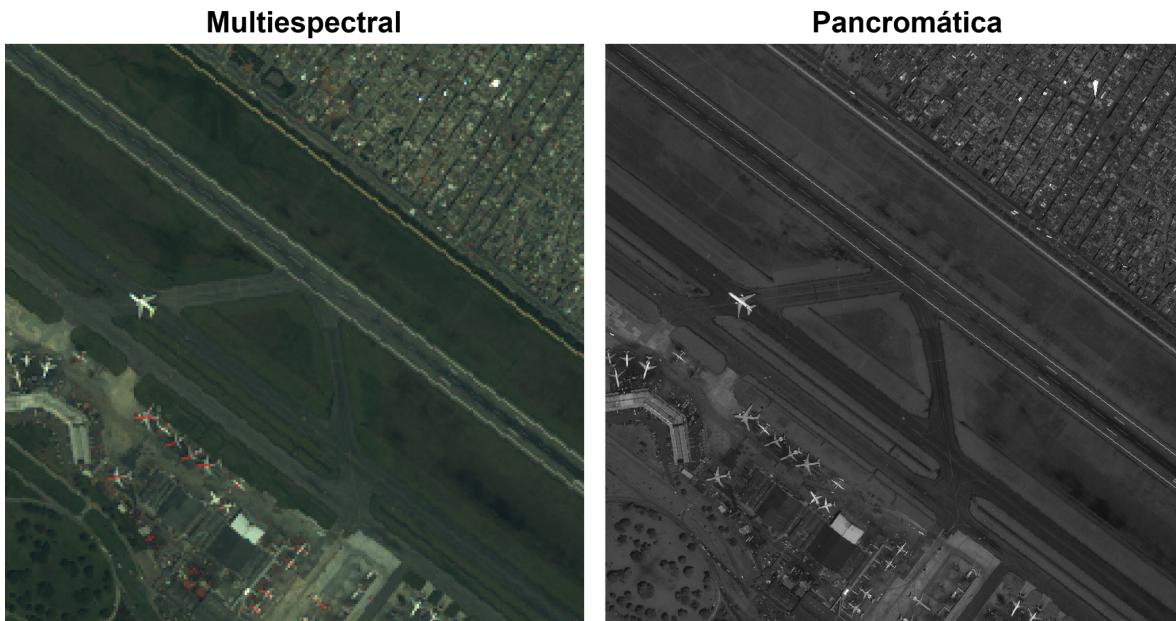
device_fusion:	Dirección local donde se desea almacenar la imagen. string, opcional
geographical_info:	Determina el dispositivo donde se desea aplicar el método de fusión. Sus opciones son 'cpu' y 'gpu'. bool, opcional

Determina si se desea conservar la información geográfica a la imagen generada a partir del método de fusión

Ejemplo 1:

```
>>> multi_path = 'multispectral.tif'
>>> pan_path = 'panchromatic.tif'
>>> method = "gram_schmidt"
>>> architecture = "gpu"
>>> geographical_info = True
>>> fused_path = f"fusion_{method}_{architecture}.tif"
>>> invoker.generate_fusion_images(multi_path, pan_path, method,
fused_path, architecture, geographical_info)
```

En el anterior ejemplo, se utilizaron las siguientes imágenes como entrada.



Generando la siguiente imagen, a partir de la fusión mediante la técnica de Gram Schmidt procesada en GPU.



3.1.1.2 Cálculo de métricas de calidad

Realiza el cálculo de métricas de calidad. Dentro de las métricas disponibles se encuentra: mse (Error Mínimo Cuadrado), rmse (Raíz del error mínimo cuadrado), bias (Sesgo) y correlation (Coeficiente de correlación).

```
satellite_images_fusion.invoker.generate_quality_metrics(fusioned_image_path,  
original_image_path, metrics=['mse', 'rmse', 'bias', 'correlation'])
```

Parámetros:

fusioned_image_path: *string*

Dirección local de la imagen fusionada que desea cargar.

original_image_path: *string*

Dirección local de la imagen de referencia con la cual desea comparar la fusionada.

metrics:

list[string]

Lista de métricas que desea obtener, su valor por defecto son todas las métricas, 'mse', 'rmse', 'bias' y 'correlation'.

Retorno

results:

dict{'metric':np.array()}

Diccionario con los resultados de las métricas proporcionadas por parámetro. Cada llave corresponde a la métrica y su valor, serán

arreglos de numpy con tantas posiciones como canales tenga la imagen fusionada.

Ejemplo 2:

```
>>> fuseded_path = 'fuseded_image.tif'  
>>> reference_path = 'reference_image.tif'  
>>> metrics_custom = ['rmse', 'bias', 'correlation']  
>>> results_metrics=invoker.generate_quality_metrics( fuseded_path,  
reference_path, metrics = metrics_custom)  
>>> results_metrics  
>>> {'bias': [-0.007974930918606793, -0.009433883111631403,  
-0.007752823074063109], 'correlation': [0.8071270263736467,  
0.714081438666127, 0.6690165994258338], 'rmse': array([12.34318889,  
14.94367754, 12.34059014])}
```

Para el ejemplo anterior, se tomó las siguientes imágenes como entrada.



Más ejemplos:

<https://colab.research.google.com/drive/1tuWc60ub4scOSNI3wTQ8EoI5b9E-tyNS>