21-6-2023

La Matemática dentro del modelado en 2D

Ortomosaico de la Universidad Central del Ecuador.

UNIVERSIDAD CENTRAL DEL ECUADOR
Omnium potentior est sapientia

GUERRÓN BURBANO DARIO JAVIER.
FACULTAD DE CIENCIAS – PROYECTO AGRICULTURA INTELIGENTE.

Contenido

Indice de Figuras.

La matemática dentro del modelado en 2D

1. Introducción.

El siguiente trabajo tiene como finalidad conocer y entender los procesos matemáticos que intervienen en la realización de un modelo en 2 dimensiones (2D) de la Universidad Central del Ecuador, para ello, se presenta una descripción general de algunos conceptos matemáticos, algunas propiedades de una imagen representada como una matriz, una explicación teorica general del algortimo "Transformación de Características Invariantes de Escala" (SIFT por sus siglas en inglés) y su aplicación en lo que se conoce como Panorama Stitching[1] para obtener el ortomosaico[2] en 2D.

En la actualidad ha incrementado el uso de métodos para el reconocimiento de un objeto en una escena, uno de ellos es el algoritmo SIFT propuesto por Lowe (Lowe, 2011), este ha logrado establecerse como un estándar para dicho propósito debido a su alta precisión y su bajo tiempo de procesamiento.

Existen varios trabajos relacionados con del algoritmo SIFT, como el trabajo de Takacs (Takacs et al., 2008), el cual implementa en un móvil un algoritmo SIFT el cual hace una comparación con ciertas imágenes contenidas en una base de datos arrojando como resultado el nombre del lugar en donde se encuentra, obteniendo resultados satisfactorios, en las condiciones optimas, otro caso, es el trabajo de Zhang (Zhang & Košecká, 2005), el cual uso un enfoque basado en un "histograma de localizador de color" que era usado para limitar la búsqueda en la base de imágenes, con un paso final basado en el algoritmo SIFT, otro, es el de He (He et al., 2006), también utilizó el algoritmo SIFT pero empleo un método de aprendizaje en el tiempo para encontrar "características prototipo" las cuales fueran utilizadas para la localización y solucionar las variaciones que se presentaban conforme a los cambios naturales presentados en el lugar. (Olvera et al., 2013). En Tanzania, una organización Suiza, lo aplicaron para el mapeo de la ciudad con alta definición para rastrear con precisión edificios y carreteras. En Etiopía para mapear fuentes de agua, en Borneo para documentar el uso ilegal de terrenos y entre otras aplicaciones. (Hernández, 2017).

SIFT ha sido utilizado para recuperación de objetos (Sivic y Zisserman, 2003), emparejamiento 3D (Delponte et al., 2006), reconstrucción de escenas 3D (Yun et al., 2007), localización y mapeo de robots (Ogawa et al., 2007), considos de imágenes panorámicas (Ostiak, 2006) y seguimiento de movimiento (Battiato et al., 2007; Battiato et al., 2009). En cuanto a las aplicaciones en la

---

[1] El Photo Stitching (puntadas fotográficas en referencia a la costura) consiste en crear una toma panorámica a partir de la concatenación de varias imágenes del mismo paisaje que cubren diferentes áreas del mismo. (Tatay, 2003)

[2] Un ortomosaico corresponde al conjunto de imágenes tomadas desde una o varias maneras que presentan áreas de traslape entre sí y que son unidas o combinadas en una sola imagen para ampliar el rango de visión de la escena. (Hernández, 2017)

fotogrametría[3], destacan el modelado 3D de objetos pequeños (Kalantari y Kassera, 2004), el análisis de seguimiento de características espacio-temporales (Heinrichs et al., 2008), registro de datos de intensidad LIDAR e imágenes aéreas (Abedinia et al., 2008), y el mapeo e tiempo real de UAV (Forstner y Steffen, 2008). (Figueiras, 2018).

Así, este algoritmo es el más empleado como base de la mayoria de softwares actuales que busca automatización en la fotogrametría, tal como describen autores como McCarthy, 2014; Apolonio et al., 2014; Bhandari et al., 2015; Aicardi et al., 2016. Donde, este algoritmo es base para el emparejamiento de imágenes por caracteristicas, a partir de la detección espacial de puntos característicos de las imágenes, invariantes a cambios de escala y orientación, permitiendo así una correlación automática de multiples imágenes o multicorrelación (Figueiras, 2018).

Al final de este trabajo en cada uno de los resultados obtenidos se indicará un ortomosaico en 2D de la Universidad Central del Ecuador, los cuales resultan de la aplicación de los algotimos mencionados en este trabajo junto con algunas de las fotografias aéreas de la UCE.

2. Objetivo General.

Describir de manera general los conceptos matemáticos que intervienen en la generación de un ortomosaico en 2D de la Universidad Central del Ecuador.

3. Materiales y Herramientas.

3.1. Equipo de escritorio/portatil.

- Fabricante: Hewlett-Packard.
- Modelo: Hp Pavillion g6 Notebook PC.
- Procesador: AMD A4-4300M APU with Radeon(tm) HD Graphics. 2.50 GHz.
- RAM instalada: 4,00 GB.
- Memoria de video: 2,00 GB.
- Sistema operativo: Windows 11 Pro 64 bits.

3.2. Software.

Lenguaje de programación: Python, versión .3.9.7.

Software de programación:

- PyCharm, versión 2021.3.3.
- Entorno Anaconda Navigator 3, versión 2021.11.

[3] La fotogrametría es la ciencia o técnica cuyo objetivo es el conocimiento de las dimensiones y posición de objetos en el espacio, a través de la medida o medidas realizadas a partir de la intersección de dos o más fotografías. (Instituto Geográfico Agustín Codazzi, 2002)

4. Marco Teórico.

4.1. Conceptos básicos.

4.1.1. Matriz.

Una matriz de m filas y n columnas ($m \times n$) con elementos en $\mathbb{R}$ es un arreglo de la forma:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Donde a_{mn} pertenecen a $\mathbb{R}$ y m, n pertenecen a $\mathbb{Z}$, en forma abreviada, la matriz anterior puede expresarse como

$$[a_{ij}], con\ i = 1, 2, \dots, m\ y\ j = 1, 2, \dots, n.$$

4.1.2. Adición de matrices.

Esta operación puede efectuarse cuando las matrices tienen el mismo orden y el resultado se obtiene sumando los elementos correspondientes de ambas matrices de acuerdo a la siguiente definición:

Sean $A = [a_{ij}]$ y $B = [b_{ij}]$ dos matrices de orden $m \times n$ con elementos en $\mathbb{R}$, la suma $A + B$ es uma matriz $S = [s_{ij}]$ de ordem $m \times n$ definida por:

$$s_{ij} = a_{ij} + b_{ij}.$$

Para $i = 1, 2, \dots, m$ y $j = 1, 2, \dots, n$.

4.1.3. Sustracción de matrices.

Esta operación puede efectuarse cuando las matrices tienen el mismo orden y el resultado se obtiene restando los elementos correspondientes de ambas matrices de acuerdo a la siguiente definición:

Sean $A = [a_{ij}]$ y $B = [b_{ij}]$ dos matrices de orden $m \times n$ con elementos en $\mathbb{R}$, la resta $A - B$ es uma matriz $R = [r_{ij}]$ de ordem $m \times n$ definida por:

$$r_{ij} = a_{ij} - b_{ij}.$$

Para $i = 1, 2, \dots, m$ y $j = 1, 2, \dots, n$.

4.1.4. Multiplicación por un escalar.

Esta operación se define formalmente como:

Sean $A = [a_{ij}]$ una matriz de orden $m \times n$ con elementos en $\mathbb{R}$ y β que pertenece a $\mathbb{R}$, el producto βA es una matriz $E = [e_{ij}]$ de orden $m \times n$ definida por:

$$e_{ij} = \beta a_{ij}.$$

Para $i = 1, 2, \dots, m$ y $j = 1, 2, \dots, n$.

4.1.5. Multiplicación de matrices.

Sean $A = [a_{ij}]$ y $B = [b_{ij}]$ dos matrices de orden $m \times n$ y $n \times p$ respectivamente, con elementos en $\mathbb{R}$, el producto AB es uma matriz $P = [p_{ij}]$ de orden $m \times q$ definida por:

$$p_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}.$$

Para $i = 1, 2, \ldots, m$ y $j = 1, 2, \ldots, n$.

4.1.6. Convolución de matrices.

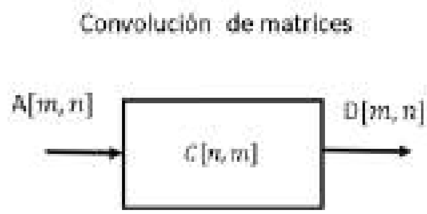


Figura 1. Convolución de matrices.

Sean $A_{m \times n}$ y una matriz $C_{(2N+1)\times(2N+1)}$con $2N + 1 < m, n$ se define la convolución de las matrices A y C, como una nueva matriz $D = A * C$ definida a partir de la expresión

$$D[m, n] = \sum_{i} \sum_{j} A[m, n] \cdot C[i - n, j - m].$$

4.2. Imagen.

Una imagen digital es una matriz de dimensión $m \times n$ compuesta por elementos muy pequeños llamados pixeles, donde cada pixel puede definir solamente un color y el número de pixeles define la dimensión o cantidad de información que contiene una imagen.

4.2.1 Canales de color RGB.

Cualquier color puede ser representado mediante combinaciones de colores rojo, verde y azul, cada uno en diferente proporción, la combinación RGB estándar indica 256 niveles por cada canal.

Así, sabiendo como podemos representar una imagen digital, formaremos una matriz de dimensión $m \times n$, con elementos vectores en donde cada vector está formado por 3 componentes (canales RGB) con valores contenidos en los enteros de 0 a 255 en un intervalo cerrado.

Por ejemplo, la imagen siguiente tiene una dimensión de 4000×2250 pixeles.

Figura 2. Fotografía original.

Tomando una pequeña zona de la fotografía anterior podemos indicar los valores de cada pixel.

(a) (b) (c)

Figura 3. Pixeles de una zona de la Figura 2.

Por ejemplo, de la imagen (c) de la Figura 3 podemos observar los valores siguientes.

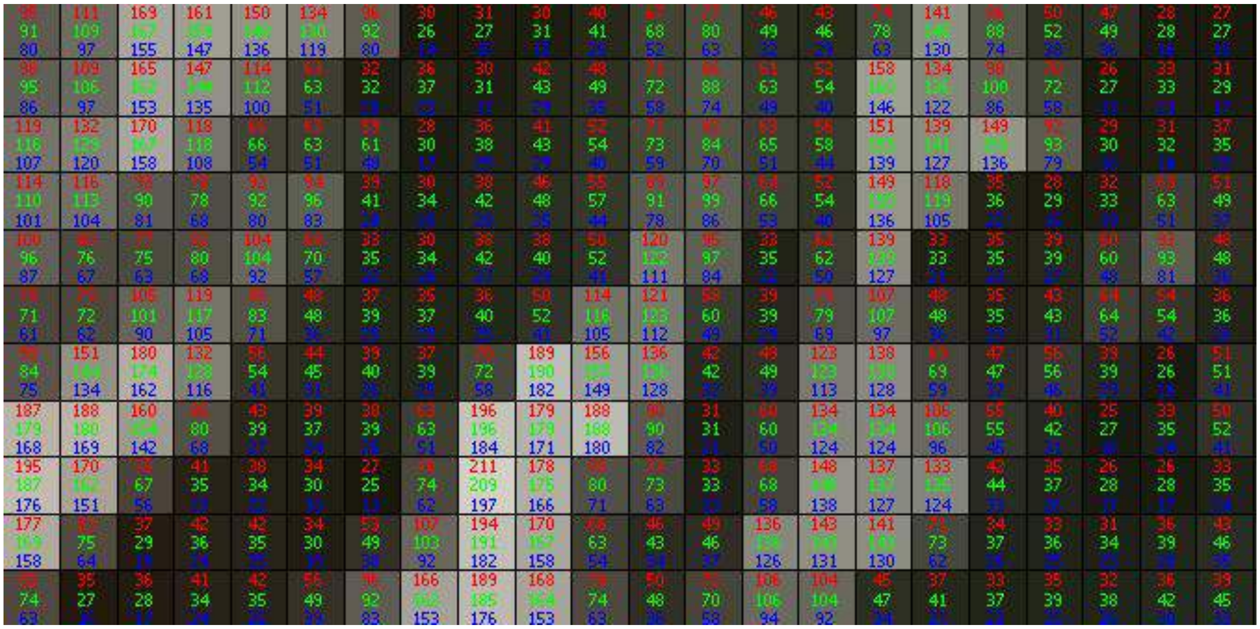


Figura 4. Valores de pixeles ampliados de la Figura 3 (c).

Si utilizamos el siguiente código realizado en Python se puede visualizar cada uno de los valores de los pixeles de los canales RGB en una imagen.

```
import cv2

imgorig = cv2.imread("Img1.jpg") # Leer imagen.

cv2.imshow("Imagen", imgorig) # Mostrar imagen.
cv2.waitKey(0) # cerrar ventana.
cv2.destroyAllWindows()
```

Figura 5. Código Python para visualizar pixeles de una imagen.

Observación: Para poder visualizar los pixeles de una imagen es necesario compilar el código anterior mediante un entorno Anaconda. Pasos: 1. Ejecutar “cmd”. 2. Crear entorno: “conda create name (nombre del entorno)”. 3. Activar entorno: “conda activate (nombre del entorno)”. 4. Instalar openCV en el entorno: “conda install -c conda-forge opencv”. 5. Entrar a la ruta del archivo y ejecutar el archivo. “python (nombre del archivo).py”.

Código de canales RGB en Python.

RBG.

```python
im = Image.open(ruta)
[ren, col] = im.size
out = im
i = 0
while i < ren:
    j = 0
    while j < col:
        niveles = im.getpixel((i, j))
        nivel_r = niveles[0]
        nivel_g = niveles[1]
        nivel_b = niveles[2]
        out.putpixel((i, j), (nivel_r, nivel_b, nivel_g))
        j += 1
    i += 1
out.save([illegible])
imagenL = PhotoImage(file=[illegible])
global rbg
rbg = canvas.create_image(500, 180, anchor=NW, image=imagenL)
ventana.mainloop()
```

Figura 6. Código de canales RGB en Python.

GBR.

```python
out.putpixel((i, j), (nivel_g, nivel_b, nivel_r))
```

Figura 7. Código de canales GBR.

GRB.

```python
out.putpixel((i, j), (nivel_g, nivel_r, nivel_b))
```

Figura 8. Código de canales GRB.

BRG.

```python
out.putpixel((i, j), (nivel_b, nivel_r, nivel_g))
```

Figura 9. Código de canales BRG.

BGR.

```python
out.putpixel((i, j), (nivel_b, nivel_g, nivel_r))
```

Figura 10. Canales de canales BGR.

Código de canales RBG con librería OpenCV.

```python
import cv2
# Cargar imagen
img = cv2.imread("RUTA-IMAGEN")

# Extraer canales
B, G, R = cv2.split(img)

# Mostrar imagen
cv2.imshow("Imagen Blue", B)
cv2.imshow("Imagen Green", G)
cv2.imshow("Imagen Red", R)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Figura 11. Código de canales RBG con librería OpenCV

Código de escala de grises.

```
ruta = ("C:/Users/dario/Imagenes/" + im)
im = Image.open(ruta)
im.show()
im2 = im
i = 0
while i < im2.size[0]:
    j = 0
    while j < im2.size[1]:
        r, g, b = im2.getpixel((i, j))
        g = (r + g + b) / 3
        gris = int(g)
        pixel = tuple([gris, gris, gris])
        im2.putpixel((i, j), pixel)
        j+=1
    i+=1
im2.show()
```

Figura 12. Código de escala de grises.

Observación: El consumo computacional al ejecutar el código anterior dependerá del tamaño y resolución de la imagen, es decir, entre más grande y más resolución de la imagen el tiempo de compilación será mucho mayor.

Código de escala de grises con librería OpenCV.

```
import cv2
# Imagen Gris
imgGris = cv2.imread("RUTA-IMAGEN", 0)
# Mostrar imagen
cv2.imshow("Imagen Gris", imgGris)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Figura 13. Código de escala de grises con OpenCV.

4.3. Orientación de fotografías.

4.3.1. Traslación de una imagen.

Es el movimiento de los pixeles de una imagen según un vector de movimiento.

La siguiente transformación muestra el resultado de trasladar el punto (x, y) según un vector de desplazamiento (d_x, d_y), obteniendo el punto (x', y').

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

4.3.2. Rotación de una imagen.

Es el giro de los pixeles de una imagen en torno al origen de coordenadas. La siguiente transformación muestra el resultado de rotar el punto (x, y) un ángulo θ, obteniendo el punto (x', y').

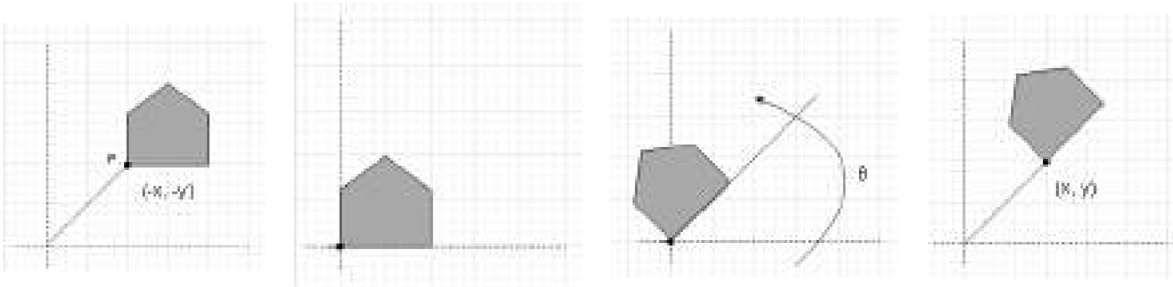
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\mathrm{sen}(\theta) & 0 \\ \mathrm{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

4.3.3. Escalado de una imagen.

Cambia el tamaño de una imagen. La siguiente transformación muestra el resultado de escalar el punto (x, y) en un factor (s_x, s_y), obteniendo el punto (x', y').

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

Ejemplo:



(a) (b) (c) (d)

Figura 14. Orientación de una imagen.

(a) Imagen original que se desea rotar en torno al punto P de coordenadas $(-x, -y)$.
(b) Resultado de la primera traslación.
(c) Resultado del giro.
(d) Resultado final después de la última traslación.

Utilizando el siguiente código en Python y la imagen de la Figura 2, podemos observar la traslación, rotación y escalado de la imagen.

```
import cv2
import numpy as np
import imutils

imgorig = cv2.imread("Img1.jpg")
# Escalado.
# Ancho: 500, Alto: 500
imgscale = imutils.resize(imgorig, width=500, height=500)
ancho = imgscale.shape[1] # N° Columnas.
alto = imgscale.shape[0] # N° Filas.

# Traslación.
# Trasladar 100 en "x" y 100 en "y".
M = np.float32([[1, 0, 100], [0, 1, 100]])
imgtrasl = cv2.warpAffine(imgscale, M, (ancho, alto))

# Rotación
# Grados de rotación: 45. # Valor de 1 para la escala
M1 = cv2.getRotationMatrix2D((ancho//2, alto//2), 45, 1)
imgrot = cv2.warpAffine(imgscale, M1, (ancho, alto))

cv2.imshow("Imagen Trasladada", imgtrasl)
cv2.imshow("Imagen Rotada", imgrot)
cv2.imshow("Imagen Escalada", imgscale)
cv2.waitKey(0)
```

Figura 15. Código en Python para la traslación, rotación y escalado de una imagen.

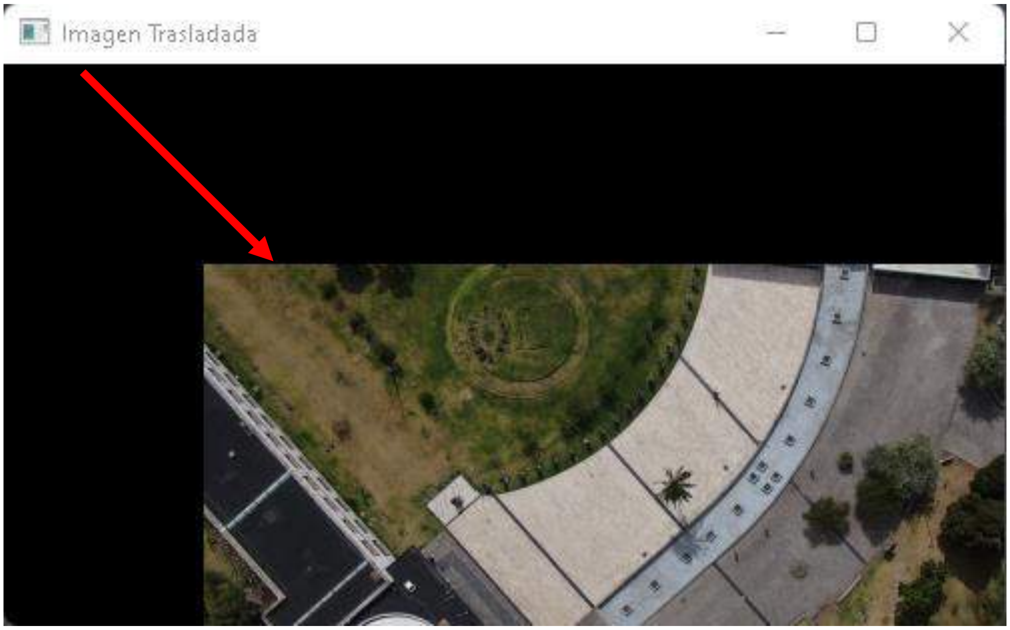


Figura 16. Figura 2 trasladada.

En la Figura 16, observamos que la Figura 2 fue trasladada 100 pixeles respecto a la esquina superior derecha en los ejes x e y.

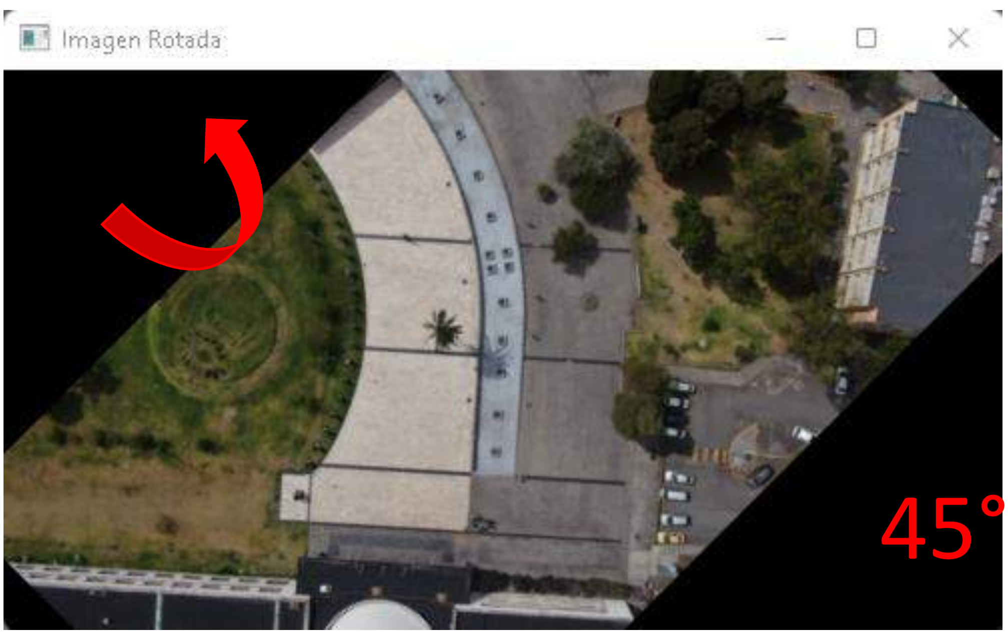


Figura 17. Figura 2 rotada.

En la Figura 17, observamos que la Figura 2 fue rotada a 45 grados.

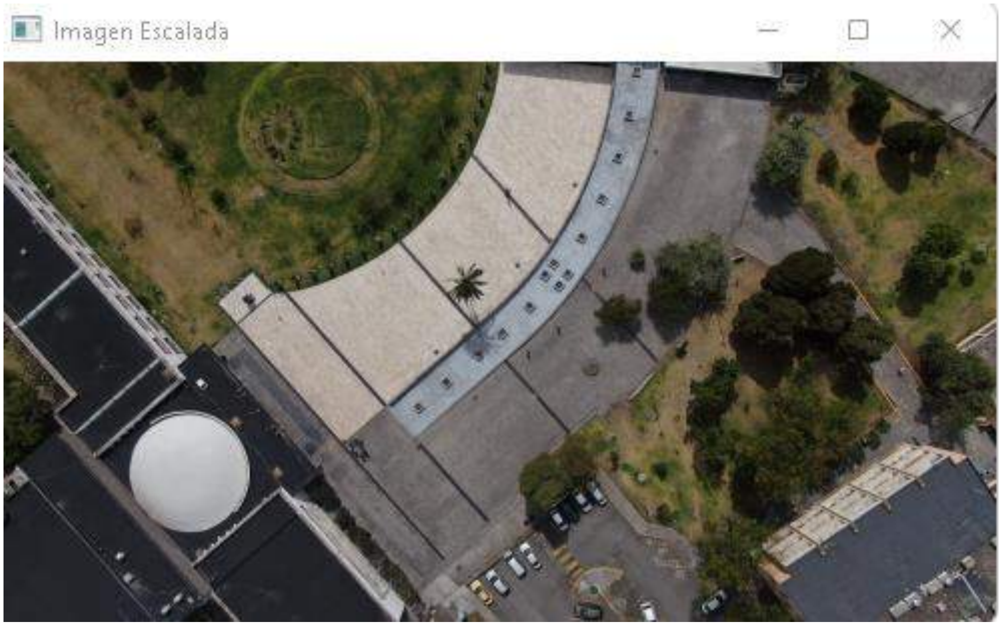


Figura 18. Figura 2 escalada.

La figura anterior es la Figura 2 pero redimensionada a 500×500 pixeles.

4.4. Traslape de imágenes.

Denominamos traslape, solape, superposición o recubrimiento, a la superposición parcial entre foto y foto de manera frontal (Traslape Longitudinal) y lateral (Traslape Transversal) en un determinado porcentaje, de acuerdo a las líneas de vuelo del dron.

En cuanto mayor sea el traslape, se conseguirá una reconstrucción del modelo más preciso, por lo que para terrenos ondulados o abruptos es recomendable considerar porcentajes de traslape mayores a los consideras en terrenos llanos. (Drone, 2019)

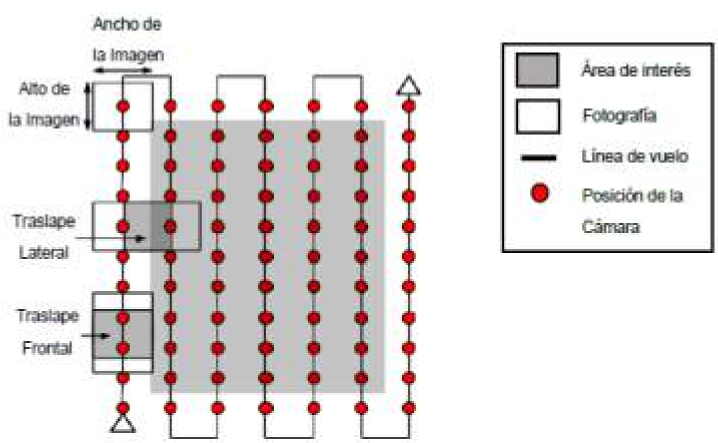


Figura 19. Traslape. (Duarte Jiménez, 2018).

Para realizar el traslape de las fotografías es necesario encontrar puntos de característicos de interés comunes, los cuales son correlacionados y triangulados para determinar la posición de cada uno de los miles o cientos de miles de puntos que conforman la fotografía, y poder formar el ortomosaico.

Así, para encontrar estos puntos característicos nos basaremos en el algoritmo siguiente.

- Transformación de características invariantes de escala. (Scale Invariant Feature Transform - SIFT).

4.5. Método SIFT para encontrar puntos clave.

Como hace mención en su investigación Herigert, para lograr detectar una imagen, en primer lugar es necesario encontrar los puntos clave o de interés que identifiquen de una manera unívoca a cada uno de los objetos de manera de poder encontrarlos nuevamente si estos aparecen en cualquier otra escena.

La idea principal del algoritmo SIFT es la transformación de la imagen a una presentación compuesta de puntos de interés, estos puntos contienen la información característica de la imagen que luego son usados para la detección de muestras. (Olvera et al., 2013).

El algoritmo SIFT detecta una gran cantidad de puntos característicos en cada imagen aplicando cuatro pasos bien estructurados. Estos puntos detectados en todas las fotografías son útiles al momento de querer buscar regiones solapadas, ya que permite comparar unos pocos pixeles entre imágenes para buscar similitudes. (Gómez Ortega, 2018).

Los puntos característicos, se refieren a los puntos donde el valor de gris de la imagen cambia drásticamente o los puntos con mayor curvatura en el borde de la imagen, como puntos de esquina, puntos de borde, puntos brillantes en áreas oscuras y puntos oscuros en áreas brillantes. Estos puntos característicos se extraen para identificar la imagen. (Li et al., 2015).

A continuación, se resume el algoritmo mediante 4 pasos principalmente:

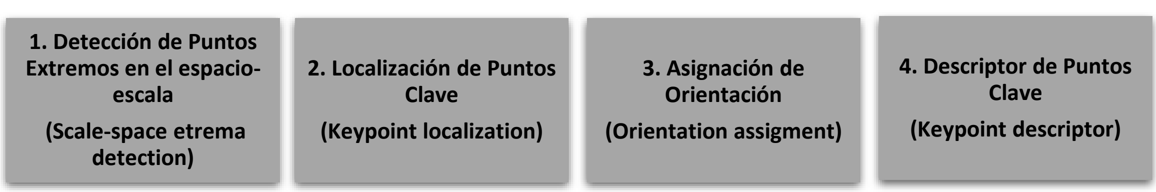


Figura 20. Diagrama de la metodología del algoritmo SIFT.

Paso 1. Detección de puntos extremos en el espacio-escala. Lo primero que se debe realizar es buscar puntos clave (o de interés) en la imagen completa, es decir, puntos que tengan una definición clara y matemáticamente bien fundada, con una posición bien definida en el espacio de la imagen, que posean estructuras de imagen locales alrededor del punto de interés que sean ricas en contenido de información, ser estables bajo deformaciones locales y globales, y que sean lo suficientemente distintos unos de otros. (Gómez Ortega, 2018).

Luego, se representa la imagen en diferentes escalas y tamaños. Se lleva a cabo de manera eficiente mediante el uso de la función de diferencia Gaussiana. Se utiliza este tipo de filtros debido a que la función Gaussiana es invariante a la traslación, a la rotación y al escalado de la misma, para la detección de puntos de interés. Además, elimina el ruido de la imagen. (Olvera et al., 2013).

Koenderink (1984) y Lindeberg (1994) demostraron que bajo una variedad de suposiciones razonables, el único filtro apropiado para estos efectos o núcleo (kernel) de espacio-escala posible es la función de Gauss. Así, el espacio-escala de una imagen se define como una función, $L(x, y, \sigma)$, tal que:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y).$$

Con $I(x, y)$ la imagen entendida como una matriz, $*$ la operación de convolución x e y, y $G(x, y, \sigma)$ la convolución Gaussiana aplicada sobre la imagen:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}.$$

Siendo σ la desviación estándar del filtro Gaussiano. (Figueiras, 2018).

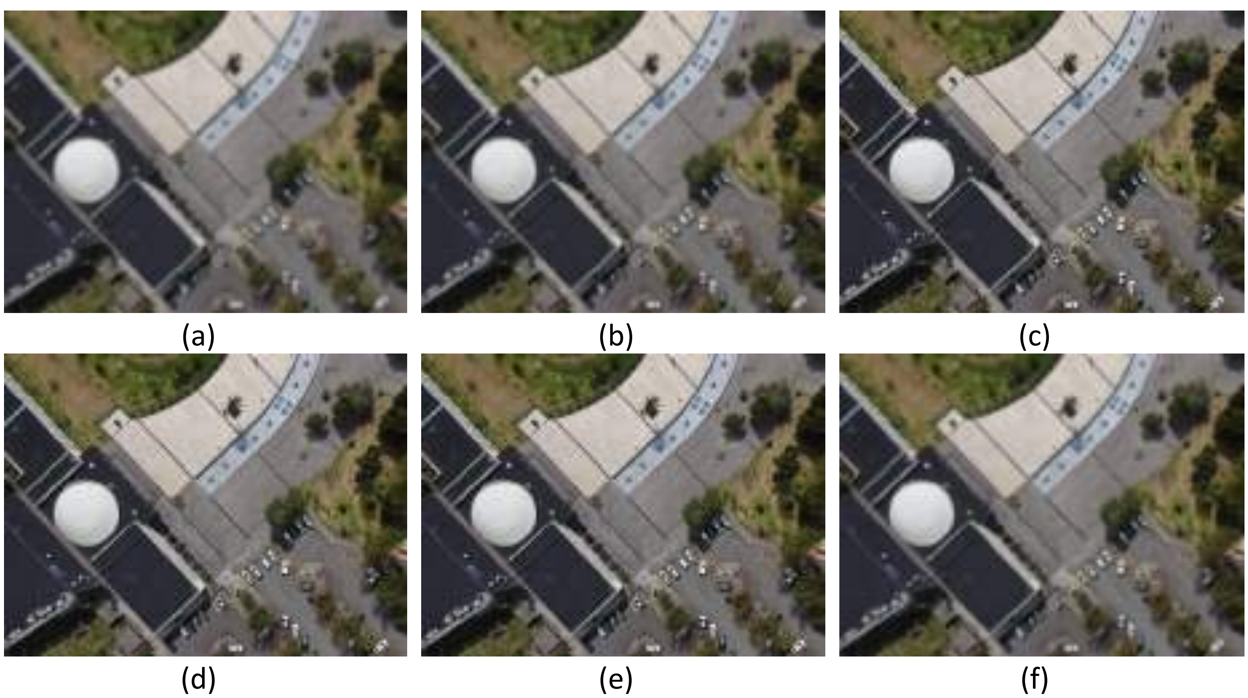


Figura 21. Filtro Gaussiano aplicado a la fotografía de la Figura 2. (a): $\sigma = 5$, (b): $\sigma = 3$, (c): $\sigma = 1$, (d): $\sigma = 0.5$, (e): $\sigma = 0.1$, (f): $\sigma = 0$.

Al aplicar diferentes valores de σ, se observa que cada una de las imágenes tienen un suavizado diferente, en el caso (a) cuando σ sea mayor el suavizado será más profundo, en cambio cuando $\sigma = 0$, el algoritmo calculara el valor de la desviación estándar.

La imagen con la que se trabaja es la convolución entre la imagen original y el filtro Gaussiano. Adicionalmente, se utiliza una distinta desviación estándar para el filtrado de la diferencia de dos filtros Gaussianos.

Para detectar puntos clave en el espacio-escala se propuso (Lowe, 1999) usar extremos del espacio-escala en la función diferencia de Gauss (o DoG) convolucionada con la imagen, $DoG(x, y, \sigma)$, computada mediante la diferencia de dos espacios-escala cercanos separados por un factor multiplicativo k:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y).$$
$$= L(x, y, k\sigma) - L(x, y, \sigma).$$

Encontrando las diferencias positivas y negativas entre los dos valores de σ, se detallan los puntos de borde o de contorno. (Universidad Miguel Hernández de Elche, 2021).

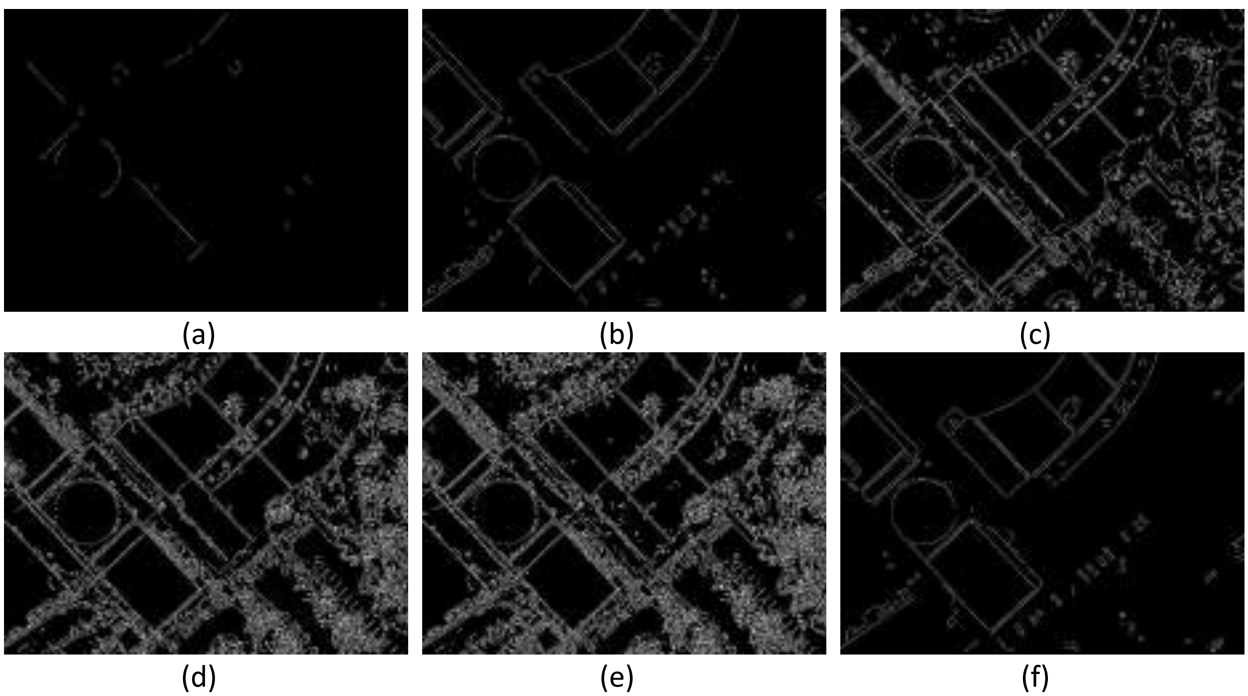


Figura 22. Bordes de cada imagen de la Figura 21. (a): $\sigma = 5$, (b): $\sigma = 3$, (c): $\sigma = 1$, (d): $\sigma = 0.5$, (e): $\sigma = 0.1$, (f): $\sigma = 0$.

Así, el algoritmo SIFT, lo que hace es calcular los puntos de borde a múltiples valores de σ y a múltiples escalas. (Min et al., 2012).

Se realiza a la imagen un número entero s a determinar, de diferencias Gaussianas, incrementando la desviación estándar teniendo en cuenta $k = 2^{1/s}$, lo que da lugar a una octava. Tras ello, la imagen se submuestrea tomando 1 de cada 2 pixeles en filas y columnas, y se procede de la misma manera, dando lugar a la siguiente octava. (Figueiras, 2018). De esta manera, se obtiene una pirámide de escalas (octavas) de la imagen, tal como se puede observar en la siguiente figura:

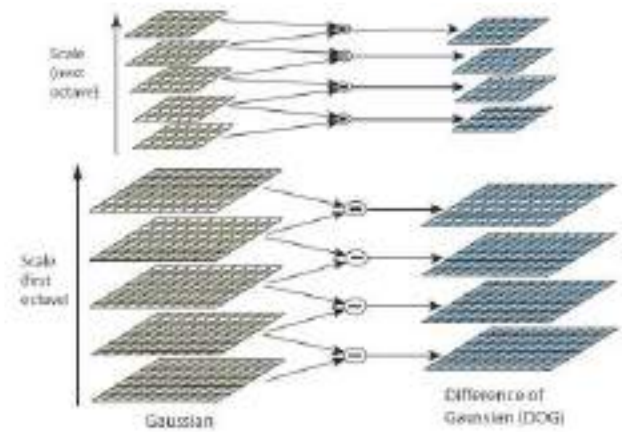


Figura 23. Pirámide de imagen espacial de escala Gaussiana y pirámide DoG. (Figueiras, 2018).

Como se observa en la figura anterior, a la izquierda, el conjunto de imágenes espacio-escala de cada octava tras la convolución Gaussiana para $s = 5$. A la derecha, el conjunto de imágenes resultado de la diferencia de Gauss. Después de cada octava, la imagen Gaussiana se sub muestrea con un factor de 2 y se repite el proceso.

Ahora, para cada imagen en cada octava de la diferencia de Gauss generada, se buscan los extremos locales, candidatos a puntos clave. Así, para una $D(x, y, \sigma)$ y una escala (octava) determinada, un punto (x_0, y_0) será un máximo o un mínimo relativo si es mayor o menor, respectivamente, a sus 8 puntos vecinos en su nivel y a sus 9 puntos vecinos de los niveles inferior y superior. Esto puede observarse en la figura siguiente. En el caso de las $D(x, y, \sigma)$ de transición entre octavas, se comparan con los puntos vecinos del nivel superior e inferior. (Figueiras, 2018).

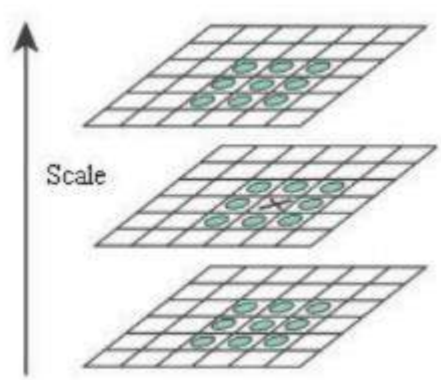


Figura 24. Detección de extremos de espacio de escala en imágenes DoG. (Figueiras, 2018).

La figura anterior representa la detección de la máxima o mínima diferencia de Gauss, comparando cada pixel (marcado con X) con sus 26 vecinos en las regiones 3x3 de los niveles adyacentes.

Demostración de la invarianza en la escala:

La función diferencia de Gauss proporciona una aproximación al Laplaciano o Gaussiano normalizado en escala, $\sigma^2 \nabla^2 G$, ya estudiado por Lindeberg (1994). Él mismo detalla que se requiere la normalización del Laplaciano con el factor σ^2, para garantizar invarianza en la escala. Además, Mikilajczyk (2002) encontró que el máximo y mínimo de $\sigma^2\nabla^2 G$ produce la mayor estabilidad en las características de la imagen. La relación entre la diferencia de Gauss D y $\sigma^2\nabla^2 G$ se entiende mediante la ecuación paramétrica:

$$\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G.$$

De la ecuación anterior, se deduce que $\nabla^2 G$ puede ser calculado como aproximación de la diferencia finita a $\frac{\partial G}{\partial \sigma}$, usando la diferencia entre espacio-escala en $k\sigma$ y σ:

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x,y,k\sigma) - G(x,y,\sigma)}{k\sigma - \sigma}.$$

De ahí:

$$G(x,y,k\sigma) - G(x,y,\sigma) \approx (k-1)\sigma^2\nabla^2 G.$$

De ello, se deduce que cuando la función de diferencia de Gauss tiene espacios-escala que difieren un factor constante k, entonces incorpora el factor de normalización de escalaσ^2 requerido para el Laplaciano invariante en escala.

Paso 2. Localización de puntos clave. Entre los puntos encontrados en el paso anterior, también se encuentran puntos de bajo contraste, inestables a cambios e iluminación y ruido. Para eliminarlos, se procede como sigue. Se estima la función diferencia de Gauss entorno a un punto $x_0 = (x_0, y_0, \sigma_0)^t$, mediante una serie de Taylor de grado 2:

$$D(X) = D + \frac{\partial D^t}{\partial X}X + \frac{1}{2}X^t\frac{\partial^2 D}{\partial X^2}.$$

Con D y sus derivadas elevadas siempre en x_0.

Derivando la ecuación anterior, igualando a 0 y simplificando, se obtiene:

$$\bar{X} = -\frac{\partial^2 D^{-1}}{\partial X^2}\frac{\partial D}{\partial X}.$$

Sustituyendo la ecuación anterior en la primera y simplificando se obtiene el valor del máximo local:

$$D(\bar{X}) = D + \frac{1}{2}\frac{\partial D^t}{\partial X}\bar{X}.$$

El criterio es, si $|D(\bar{X})| < 0.03$, el punto es eliminado de la lista de puntos clave.

Tras eliminar los puntos de bajo contraste, hay que quitar los puntos situados a lo largo de bordes. Para ello, se procede de manera análoga al algoritmo de Harris (Harris, 1988). Esto es, sea H la matriz Hessiana de $D(x, y, \sigma)$ evaluada en un punto (x_0, y_0, σ_0) del espacio-escala, se tendrá un borde, si sus autovalores α y β, son uno grande y otro pequeño, o lo que es lo mismo:

$$Traza(H) = \frac{\partial^2 D}{\partial x^2} + \frac{\partial^2 D}{\partial y^2} = \alpha + \beta.$$

$$Det(H) = \frac{\partial^2 D}{\partial x^2} \times \frac{\partial^2 D}{\partial y^2} = \alpha \cdot \beta.$$

Denominando $\alpha = r \cdot \beta$, la condición se reduce a:

$$\frac{Traza(H)^2}{Det(H)} < \frac{(r+1)^2}{r}.$$

Tras varios experimentos, Lowe (2004) propone un umbral de $r = 10$. (Figueiras, 2018).

Paso 3. Asignación de la orientación.

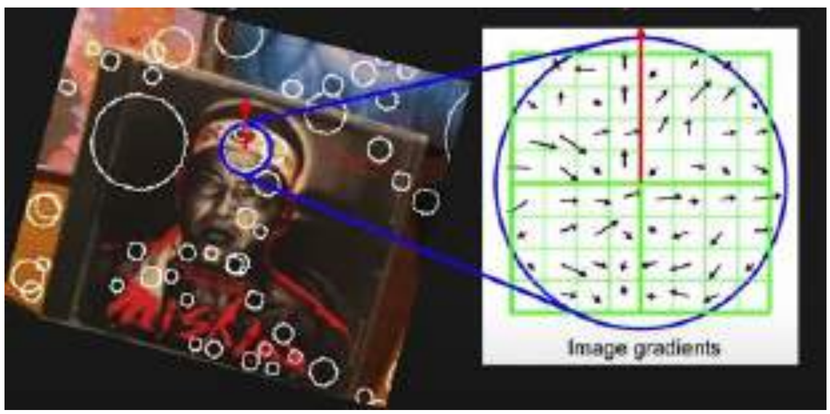


Figura 25. Orientación del gradiente. (Figueiras, 2018).

Mediante la asignación de orientación a cada punto clave de la imagen basada en características locales de la misma, se pueden lograr características invariantes a las rotaciones. Así pues, para cada punto de la imagen filtrada $L(x, y, \sigma)$, se puede determinar su gradiente $m(x, y)$ y fase $\theta(x, y)$, mediante diferencias entre pixeles:

$$m(x, y) = \sqrt{\left(L(x+1, y) - L(x-1, y)\right)^2 + \left(L(x, y+1) - L(x, y-1)\right)^2}.$$

$$\theta(x, y) = tg^{-1}\left(\left(L(x, y+1) - L(x, y-1)\right) / \left(L(x+1, y) - L(x-1, y)\right)\right).$$

Para determinar la orientación precisa de cada punto clave, se selecciona una región de 16x16 pixeles alrededor del punto, y de cada pixel se calcula su gradiente y fase. Con ello, se genera un histograma de direcciones, ponderado por una ventana de Gauss circular en torno al punto clave, de $\sigma_{h1} = 1.5 * (Nivel\ espacio\_escala)$. El máximo en el histograma corresponde a una dirección dominante en el gradiente local que será asignada al punto clave. (Figueiras, 2018).

Paso 4. Descriptor de puntos clave: Tras la asignación a cada punto clave de una escala, localización y orientación, lo siguiente es determinar para cada punto clave un descriptor relativamente invariante a cambios de iluminación y transformaciones afines. Ahora, una ventana Gaussiana de $\sigma_{h2} = 0.5$ centrada en el punto clave ya orientado, pondera los valores de su módulo y fase a partir de subregiones de 4×4 pixeles, obteniéndose 16 subregiones con 8 direcciones distintas cada una, tal como se muestra en la figura siguiente. La longitud de cada dirección corresponde con la suma de las magnitudes de los gradientes cercanos a esa dirección dentro de cada subregión. Así, se tiene para cada punto clave, un descriptor de $4 \cdot 4 \cdot 8 = 128$ elementos. (Figueiras, 2018).

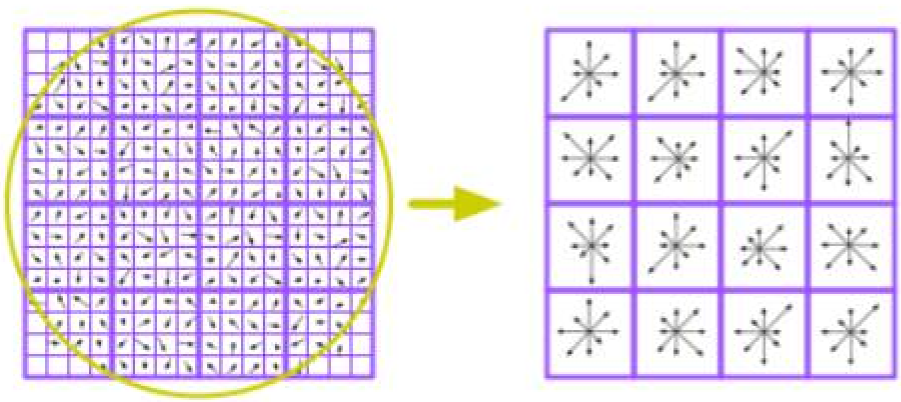


Figura 26. Descriptor de puntos clave. (Figueiras, 2018).

A la izquierda de la imagen anterior se indica los gradientes de la ventana circular de 16×16 pixeles. A la derecha el descriptor SIFT para un punto clave, con 8 direcciones por cada subregión de 4×4 pixeles.

Una vez que se tienen identificados los puntos característicos y representados por descriptores, para cada par de imágenes que comparten información, según ya fue identificado en la sección anterior, se buscan correspondencias entre sí, proceso llamado emparejamiento de puntos claves.

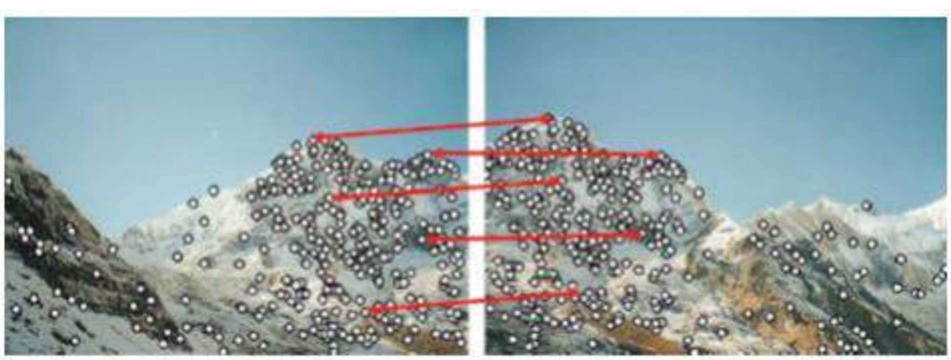


Figura 27. Emparejamiento de puntos clave entre dos imágenes basado en descriptores locales.

El objetivo de esta fase es emparejar los puntos encontrados de una imagen con los de las demás, utilizando como comparador los vectores de características como se indica en la figura anterior. Esto permite inferir qué imágenes están solapadas e identificar así qué puntos corresponden a la misma información de la escena fotografiada. Para este fin se define una medida de distancia entre los descriptores de los puntos de interés y generalmente, el desempeño del método depende tanto de las propiedades del punto como de la elección del descriptor de la imagen. (Gómez Ortega, 2018).

Para abordar este problema se define $F(I)$ como el conjunto de características encontradas en la imagen I. Para cada par de imágenes I y J, el sistema considera cada descriptor $f \in F(I)$ y encuentra su vecino más cercano (en el espacio de los descriptores) $f_{nn} \in F(J)$ como:

$$f_{nn} = arg \min_{f_J \in F(J)} \left\| f - f_J \right\|_2.$$

Para encontrar el argumento que minimiza la expresión se calcula la distancia euclidiana entre cada par posible, obteniendo así el emparejamiento entre ambos descriptores. Luego de emparejar las características de I y J, cada característica $f \in F(I)$ será emparejada con a lo más una característica en $F(J)$, y algunos de estos emparejamientos pueden estar equivocados. Si luego de este proceso, un par de imágenes posee menos que un mínimo de emparejamientos, las imágenes se consideran que no calzan, y todos los emparejamientos son removidos de estas.

4.6. Panorama Stitching.

4.6.1. Matriz de Homografía para superponer imágenes.

Una homografía corresponde a cualquier transformación proyectiva que genere una correspondencia entre dos figuras geométricas planas (fotografías), tal que a cada punto de ellas le corresponda un punto de la otra. Para definir cómo se realiza esta relación geométrica hay que representar los puntos de una imagen de la siguiente manera: Un punto (x, y) en una imagen puede representarse por un vector $P = (p_1, p_2, p_3)$, donde $x = \frac{p_1}{p_3}$ e $y = \frac{p_2}{p_3}$, lo que se llama representación homogénea de un punto. La homografía entre dos puntos equivalentes P' y P de dos figuras se calcula como $P' = HP$, donde P' corresponde a un punto de una imagen emparejado con el punto P de la otra, y H corresponde a la matriz de homografía. Si se realiza esta transformación con la matriz adecuada sobre dos imágenes completas que comparten información, se puede lograr superponer las fotografías para generar el ortomosaico. Esto significa que para calcular la homografía que mapea un punto P a su correspondiente P' en otra imagen, basta con calcular la matriz de homografía de 3×3. (Gómez Ortega, 2018).

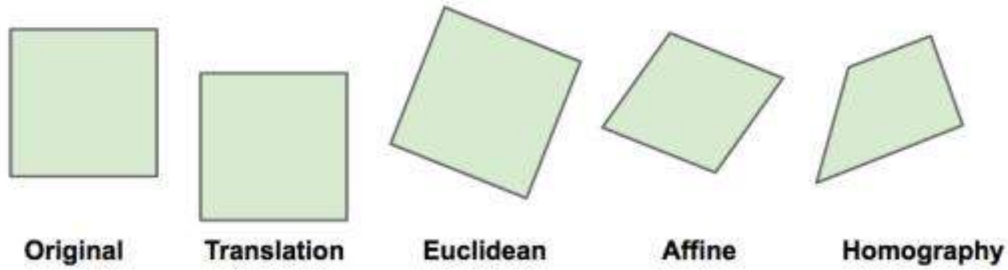


Figura 28. Tipos de homografías.

Una homografía puede tener distintos tipos de transformaciones visuales según los grados de libertad que esta tenga. El primer caso es la original, luego la traslación con 2 grados de libertad, la transformación euclidiana para 3 grados de libertad, transformación afín para 6 grados de libertad y homografía para 8 grados de libertad.

La matriz H produce proyecciones invariables en escala, demostración que se hará a continuación. Para ello se debe probar que la coordenada (x, y) obtenida por el vector P' generado por la proyección $P_a' = HP$ es el mismo que el generado por $P_b' = \lambda HP$. C

$$P_a' = \begin{pmatrix} p_{a1}' \\ p_{a2}' \\ p_{a3}' \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}.$$

$$P_a' = \begin{pmatrix} H_{11}p_1 + H_{12}p_2 + H_{13}p_3 \\ H_{21}p_1 + H_{22}p_2 + H_{23}p_3 \\ H_{31}p_1 + H_{32}p_2 + H_{33}p_3 \end{pmatrix}.$$

$$x_a' = \frac{p_{a1}'}{p_{a3}'} = \frac{H_{11}p_1 + H_{12}p_2 + H_{13}p_3}{H_{31}p_1 + H_{32}p_2 + H_{33}p_3}.$$

$$y_a' = \frac{p_{a2}'}{p_{a3}'} = \frac{H_{21}p_1 + H_{22}p_2 + H_{23}p_3}{H_{31}p_1 + H_{32}p_2 + H_{33}p_3}.$$

Luego, se realiza el mismo procedimiento para el punto $P_b{}'$ para verificar la igualdad:

$$P_b' = \begin{pmatrix} p_{b1}' \\ p_{b2}' \\ p_{b3}' \end{pmatrix} = \lambda \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}.$$

$$P_b' = \begin{pmatrix} \lambda H_{11}p_1 + \lambda H_{12}p_2 + \lambda H_{13}p_3 \\ \lambda H_{21}p_1 + \lambda H_{22}p_2 + \lambda H_{23}p_3 \\ \lambda H_{31}p_1 + \lambda H_{32}p_2 + \lambda H_{33}p_3 \end{pmatrix}.$$

$$x_b' = \frac{p_{b1}'}{p_{b3}'} = \frac{\lambda H_{11}p_1 + \lambda H_{12}p_2 + \lambda H_{13}p_3}{\lambda H_{31}p_1 + \lambda H_{32}p_2 + \lambda H_{33}p_3} = \frac{H_{11}p_1 + H_{12}p_2 + H_{13}p_3}{H_{31}p_1 + H_{32}p_2 + H_{33}p_3} = x_a'.$$

$$y_b' = \frac{p_{b2}'}{p_{b3}'} = \frac{\lambda H_{21}p_1 + \lambda H_{22}p_2 + \lambda H_{23}p_3}{\lambda H_{31}p_1 + \lambda H_{32}p_2 + \lambda H_{33}p_3} = \frac{H_{21}p_1 + H_{22}p_2 + H_{23}p_3}{H_{31}p_1 + H_{32}p_2 + H_{33}p_3} = y_a'.$$

Demostrando finalmente que la matriz H genera la misma proyección independiente de la escala. Esto permite que se pueda definir esta matriz con solo 8 grados de libertad.

$$H = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{pmatrix}.$$

Ya definida la matriz de homografía y sus propiedades más importantes, junto al uso que puede tener en la generación del ortomosaico, es necesario entender cómo se puede estimar. Considerando que la H posee 8 grados de libertad, son 8 las incógnitas a resolver por pares de imágenes. Con todo esto en consideración, se procede a desarrollar las ecuaciones según pares de emparejamientos (X_i, Y_i) y (X_i', Y_i') a partir de la relación $P' = HP$, donde $P_i' = (x_i', y_i', w_i')$, $P_i = (x_i, y_i, w_i)$, $X_i' = \frac{x_i'}{w_i'}$, $Y_i' = \frac{y_i'}{w_i'}$, $X_i = \frac{x_i}{w_i}$, e $Y_i = \frac{y_i}{w_i}$. Con esto se representan los vectores P y P' de la siguiente manera.

$$P_i' = \begin{pmatrix} X_i' \cdot w_i' \\ Y_i' \cdot w_i' \\ w_i' \end{pmatrix}.$$

$$P_i = \begin{pmatrix} X_i \cdot w_i \\ Y_i \cdot w_i \\ w_i \end{pmatrix}.$$

Reemplazando en la relación $P_i' = HP_i$, se tiene:

$$\begin{pmatrix} X_i' \cdot w_i' \\ Y_i' \cdot w_i' \\ w_i' \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{pmatrix} \begin{pmatrix} X_i \cdot w_i \\ Y_i \cdot w_i \\ w_i \end{pmatrix}.$$

Luego, desarrollando la multiplicación:

$$\begin{pmatrix} X_i' \cdot w_i' \\ Y_i' \cdot w_i' \\ w_i' \end{pmatrix} = \begin{pmatrix} X_i \cdot w_i \cdot H_{11} + Y_i \cdot w_i \cdot H_{12} + w_i \cdot H_{13} \\ X_i \cdot w_i \cdot H_{21} + Y_i \cdot w_i \cdot H_{22} + w_i \cdot H_{23} \\ X_i \cdot w_i \cdot H_{31} + Y_i \cdot w_i \cdot H_{32} + w_i \end{pmatrix}.$$

Finalmente:

$$\begin{cases} X_i' = \dfrac{X_i' \cdot w_i'}{w_i'} = \dfrac{X_i \cdot w_i \cdot H_{11} + Y_i \cdot w_i \cdot H_{12} + w_i \cdot H_{13}}{X_i \cdot w_i \cdot H_{31} + Y_i \cdot w_i \cdot H_{32} + w_i} = \dfrac{X_i \cdot H_{11} + Y_i \cdot H_{12} + H_{13}}{X_i \cdot H_{31} + Y_i \cdot H_{32} + 1} \\ Y_i' = \dfrac{Y_i' \cdot w_i'}{w_i'} = \dfrac{X_i \cdot w_i \cdot H_{21} + Y_i \cdot w_i \cdot H_{22} + w_i \cdot H_{23}}{X_i \cdot w_i \cdot H_{31} + Y_i \cdot w_i \cdot H_{32} + w_i} = \dfrac{X_i \cdot H_{21} + Y_i \cdot H_{22} + H_{23}}{X_i \cdot H_{31} + Y_i \cdot H_{32} + 1} \end{cases}.$$

$$\Rightarrow \begin{cases} X_i' \cdot (X_i \cdot H_{31} + Y_i \cdot H_{32} + 1) = X_i \cdot H_{11} + Y_i \cdot H_{12} + H_{13} \\ Y_i' \cdot (X_i \cdot H_{31} + Y_i \cdot H_{32} + 1) = X_i \cdot H_{21} + Y_i \cdot H_{22} + H_{23} \end{cases}.$$

$$\Rightarrow \begin{cases} X_i \cdot H_{11} + Y_i \cdot H_{12} + H_{13} - X_i' X_i \cdot H_{31} - X_i' Y_i \cdot H_{32} = X_i' \\ X_i \cdot H_{21} + Y_i \cdot H_{22} + H_{23} - Y_i' X_i \cdot H_{31} - Y_i' Y_i \cdot H_{32} = Y_i' \end{cases}.$$

Obteniendo dos ecuaciones lineales con las coordenadas originales de las imágenes. Si esto se aplica para cuatro emparejamientos, se tendrá un conjunto de 8 ecuaciones, lo que permite resolver el sistema y obtener así la matriz de homografía H que proyecta una imagen en el plano de la otra. Con esto se puede armar el siguiente sistema de ecuaciones lineales de forma matricial, el que se puede resolver fácilmente (Gómez Ortega, 2018).

$$\begin{pmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -X_1X_1' & -X_1'Y_1 \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -X_1Y_1' & -Y_1Y_1' \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -X_2X_2' & -X_2'Y_2 \\ 0 & 0 & 0 & X_2 & Y_2 & 1 & -X_2Y_2' & -Y_2Y_2' \\ X_3 & Y_3 & 1 & 0 & 0 & 0 & -X_3X_3' & -X_3'Y_3 \\ 0 & 0 & 0 & X_3 & Y_3 & 1 & -X_3Y_3' & -Y_3Y_3' \\ X_4 & Y_4 & 1 & 0 & 0 & 0 & -X_4X_4' & -X_4'Y_4 \\ 0 & 0 & 0 & X_4 & Y_4 & 1 & -X_4Y_4' & -Y_4Y_4' \end{pmatrix} \begin{pmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \end{pmatrix} = \begin{pmatrix} X_1' \\ Y_1' \\ X_2' \\ Y_2' \\ X_3' \\ Y_3' \\ X_4' \\ Y_4' \end{pmatrix}.$$

4.6.2. Algoritmo Random Sample Consensus (RANSAC) para la eliminación de emparejamientos erróneos.

El algoritmo RANSAC es un enfoque de estimación de parámetros generales diseñado para encontrar y eliminar valores atípicos en una serie de datos explicados por un modelo, para el tema de este trabajo, la matriz de homografía. RANSAC es una técnica que genera candidatos de solución utilizando la menor cantidad de datos (emparejamientos) posibles, los necesarios para estimar el modelo que subyace en el problema. (Gómez Ortega, 2018). El algoritmo tiene la siguiente estructura:

1. Se seleccionan aleatoriamente los puntos requeridos para determinar los parámetros del modelo.

2. Se obtiene el modelo con los puntos seleccionados.

3. Se calcula cuántos puntos del conjunto completo calzan bien en el modelo con cierto nivel de tolerancia.

4. Si la fracción de inliers sobre el total de puntos supera cierto umbral, se reestima el modelo con estos puntos y se eliminan los outliers. De lo contrario, se repite desde el paso 1 al 4 un número determinado de veces, quedándose con el mejor modelo.

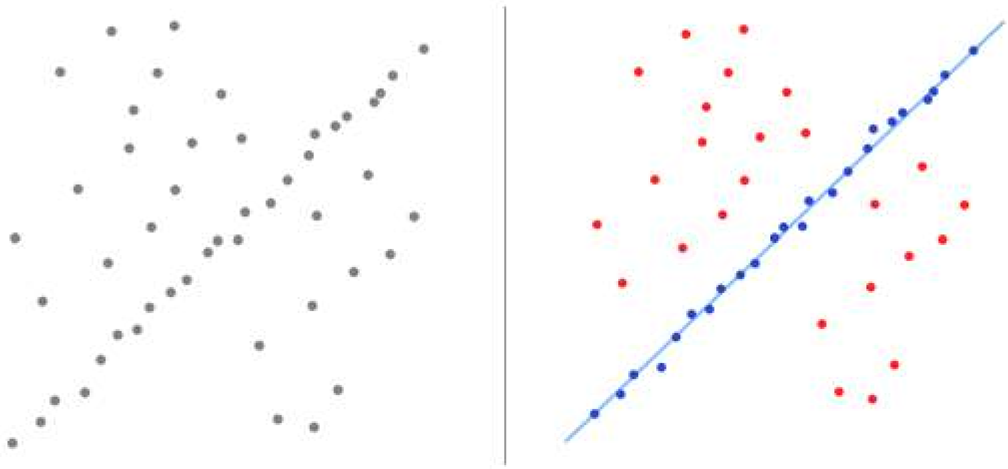


Figura 29. Representación del algoritmo RANSAC.

A la izquierda se tiene un conjunto de datos con distintos valores atípicos. En la derecha el algoritmo RANSAC encuentra un modelo que explica la mayoría de los datos y permite identificar cuáles son los valores atípicos.

5. Resultados.

A continuación se explicará los resultados obtenidos teniendo en consideración el número de fotografías necesarias para determinar cuáles son las fotografías que deberían poseer solapamiento para generar el ortomosaico de la Universidad Central del Ecuador.

Se comienza trabajando con las imágenes en bruto y posteriormente redimensionadas, luego, utilizando los métodos y algoritmos anteriormente descritos, se buscan sobre estas imágenes cuáles son las que comparten información para poder modificar la imagen ya sea rotándola, modificando su escala, o estirándola, para que al superponerlas se empiece a crear el ortomosaico. Después se realiza una adecuación de las imágenes para que los bordes de éstas al superponerse sobre otra no se noten y se vea una fotografía más homogénea, obteniendo así, el ortomosaico.

5.1. Fotografías.

Como componente principal para la obtención del ortomosaico se utilizó 30 fotografías, con dimensiones de 4000×2250 pixeles:


f01

f02

f03

f04

f05

f06

f07

f08

f09

f10

f11

f12

Figura 30. Fotografías utilizadas.

5.2. Aplicación de los algoritmos.

Los algoritmos realizados en la sección anterior, además de permitir que las imágenes estén bien adecuadas para la generación del ortomosaico, están pensados para reducir la cantidad de procesamiento que se tendría que hacer en los pasos posteriores. El paso siguiente consiste en encontrar la relación que poseen las imágenes entre sí de manera más detallada, interesando cuáles son las que se encuentran solapadas. Para ello se detectan los puntos característicos de cada imagen; se emparejan estos entre pares de imágenes; y finalmente se realiza una verificación para eliminar aquellos emparejamientos erróneos.

El algoritmo SIFT es aquel que presenta mejor rendimiento en la búsqueda de puntos comunes en imágenes aéreas sobrepuestas. Los resultados positivos que este tiene se asocian a su invarianza tanto de rotación como de escala. Además de las ubicaciones de los puntos clave, SIFT proporciona un descriptor local para cada punto clave, haciendo que una imagen contenga varios miles de puntos clave SIFT como se explicó en el marco teórico.

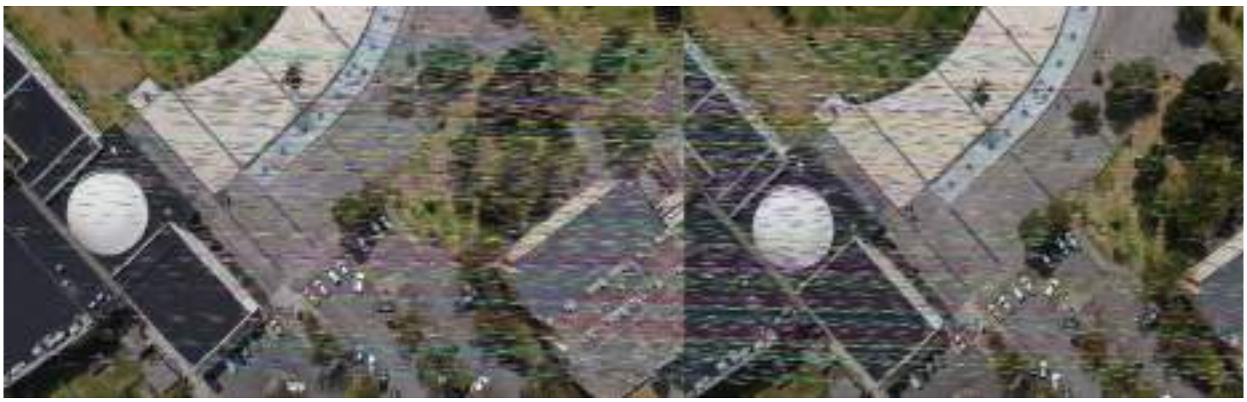
A continuación se indican los pasos realizados para obtener algunos resultados de aplicar este algoritmo junto con algunas fotografías de la Universidad Central del Ecuador.

1. Cargar 2 fotografías.
2. Elegir dimensión de fotografías.
3. Aplicar algoritmo SIFT (encuentra puntos característicos y los une con líneas con diferente color).
    a. Detección de puntos extremos en el espacio-escala.
    b. Localización de puntos clave.
    c. Asignación de orientación.
    d. Descriptor de puntos clave.
4. Aplicar método Panorama Stitching.
5. Resultado (ortomosaico).

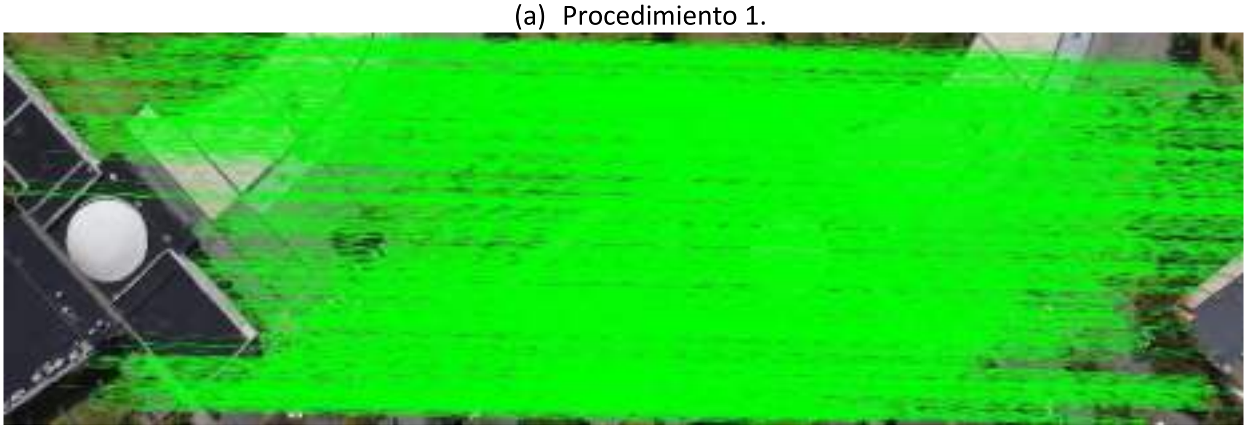
Se utilizaron las fotografías f01, f02 de la Figura 30, con dimensiones originales de 4000×2250 pixeles.

(a) (b)

Figura 31. (a) f01. (b) f02.



(a) Procedimiento 1.



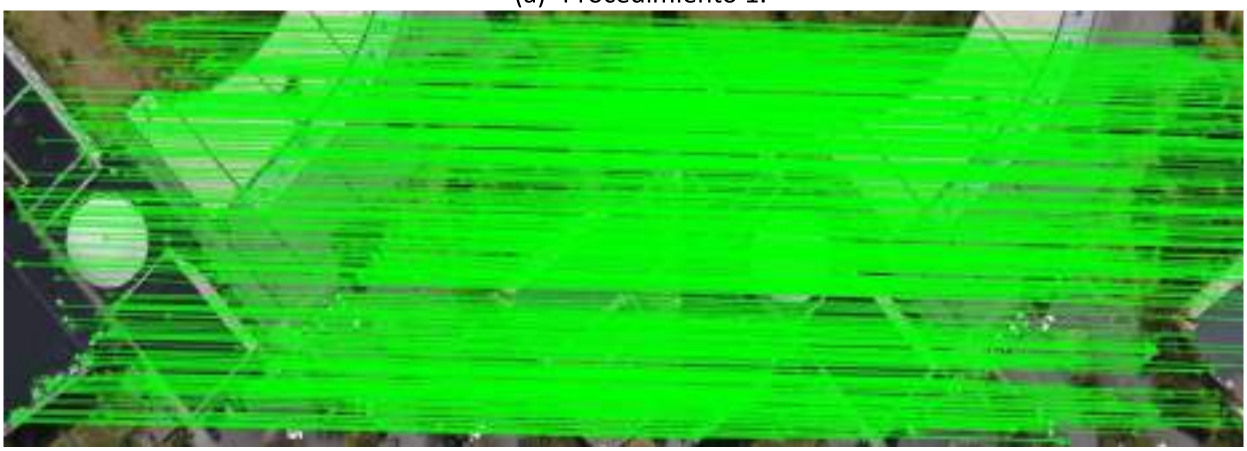
(b) Procedimiento 2

Figura 32. Puntos característicos comunes. Fotografías a 4000x2250 pixeles.

En la Figura 30, podemos observar que utilizando dos procesos diferentes y fotografías con 4000×2250 pixeles, los puntos característicos comunes son en gran cantidad y cada una de las líneas que se observa corresponden a una coincidencia excelente.

Por otro lado, se utilizaron las mismas fotografías f01, f02 de la Figura 30, pero redimensionadas a 800×500 pixeles.

(a) Procedimiento 1.



(b) Procedimiento 2

Figura 33. Puntos característicos comunes. Fotografías a 800 x 500 pixeles.

En la Figura 33 podemos observar que utilizando dos procesos diferentes y fotografías redimensionadas a 800×500 pixeles, los puntos característicos comunes difieren en la cantidad, es decir, el número de puntos característicos comunes es menor en comparación con la Figura 32, debido al redimensionamiento de la fotografía. Además, se observa una coincidencia satisfactoria ya que hay líneas diagonales que se unen con puntos diferentes.

Si elegimos dos puntos P1 y P2 correlacionados de la Figura 33. (a), representados de color rojo como se indica en la figura siguiente, podemos observar que son los mismos puntos en cada fotografía.

Figura 34. Puntos correlacionados de la Figura 33. (a).

Pero, se observa en la figura anterior, que existe un desplazamiento del punto P1 al punto P2, debido al movimiento del dron para realizar la fotografía es por ello que el valor de cada pixel varía, como se representa en la figura siguiente.

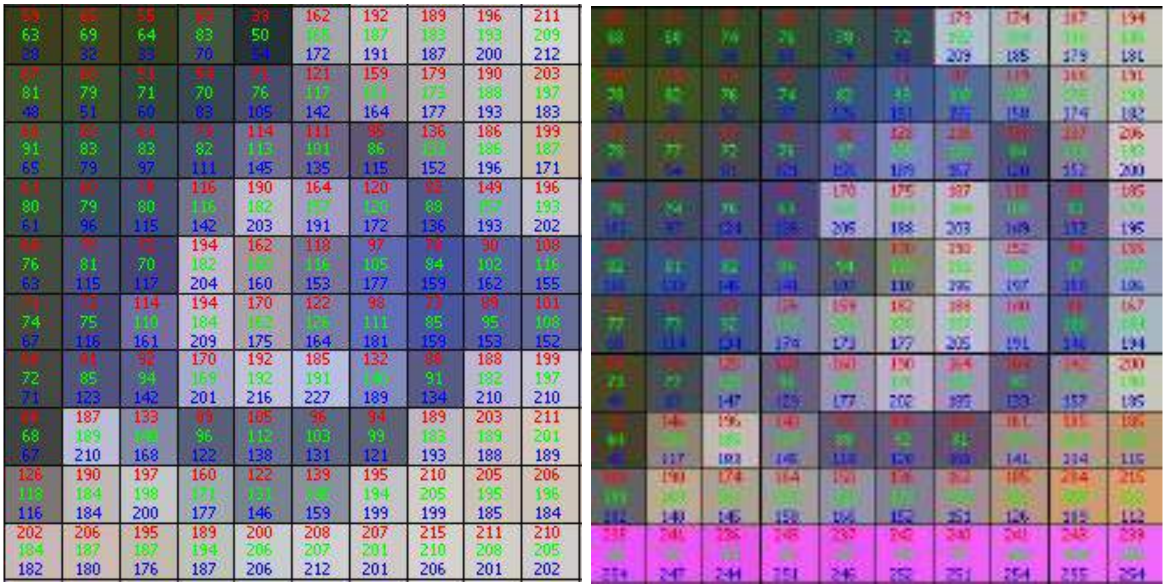
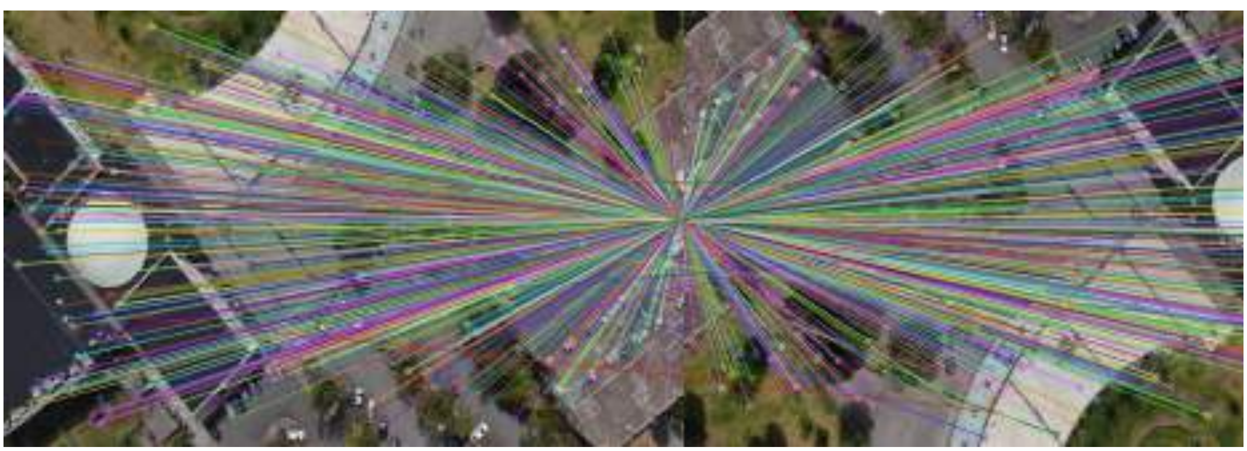


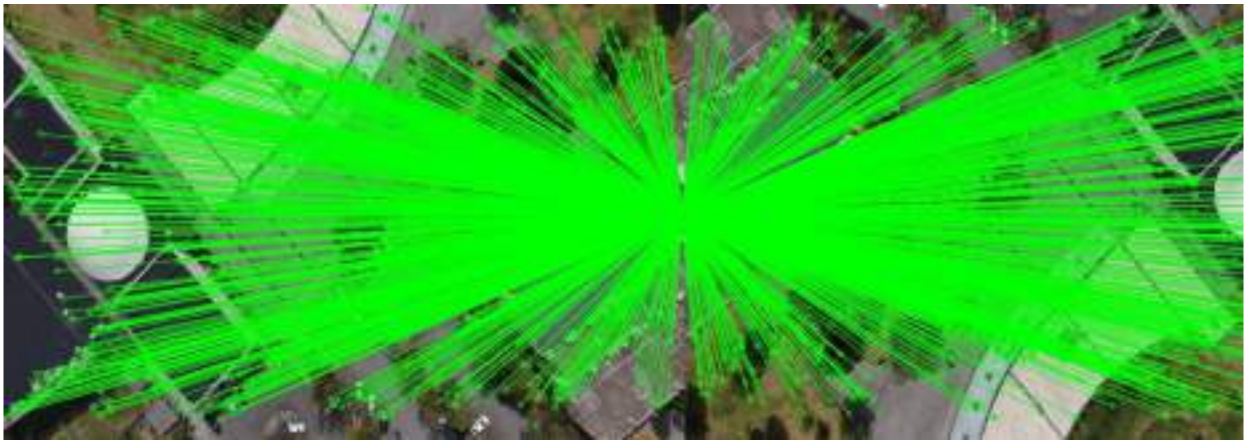
Figura 35. Pixeles de puntos correlacionados de la Figura 33. (a).

Lo que nos indica que el valor de cada pixel es diferente en cada punto característico, esto puede ser a casusa del cambio de posición del dron conforme avanza para capturar la fotografía, al cambio de iluminación, brillo, contraste y/o cambio en las condiciones del ambiente.

Usando las fotografías f01, f02 de la Figura 30 y redimensionadas a 800×500 pixeles y con la segunda fotografía f02 rotada 180 grados hacia la izquierda obtenemos las siguientes coincidencias.



(a) Procedimiento 1.



(b) Procedimiento 2

Figura 36. Puntos característicos comunes.

Realizando esta modificación en los dos procedimientos utilizados, podemos observar que se encuentran buenas coincidencias entre los puntos característicos comunes entre las dos imágenes.

Luego de realizar este algoritmo se aplicó en lo que se conoce como Panorama Stitching (Puntadas panorámicas) que consiste en unir imágenes para crear una imagen más grande o un panorama encontrando los puntos de interés o las características de SIFT y luego usa los descriptores para hacer coincidir estas características. En función de estas características coincidentes o correspondencias, básicamente tomará una imagen, la deformará y la alineará con la otra imagen. (Vision First Principles of Computer, 2021).

Si tenemos una gran colección de imágenes, simplemente se aplica el algoritmo SIFT a cada una de ellas, luego se combina las características entre estas diferentes imágenes y se puede juntar estas imágenes para obtener un ortomosaico en 2 dimensiones.

Antes de indicar los resultados del algoritmo SIFT y su aplicación para la obtención del ortomosaico, se explica de una manera matricial como es la coincidencia entre imágenes para obtener una imagen combinada.

5.3. Matricialmente.

Como ya hemos visto, una imagen se puede considerar como una matriz formada por los valores RGB de cada pixel, así, supongamos que se tiene tres imágenes de dimensiones 6×8 pixeles, con las cuales encontramos los puntos característicos comunes entre ellas y su traslape longitudinal y transversal.

Representamos como $I_1, I_2\ y\ I_3$ a las matrices formadas por los canales (r, g, b) para cada una de las imágenes.

$$I_1 = \begin{pmatrix} i_{11} & i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} \\ i_{21} & i_{22} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} \\ i_{31} & i_{32} & i_{33} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} \\ i_{41} & i_{42} & i_{43} & i_{44} & i_{45} & i_{46} & i_{47} & i_{48} \\ i_{51} & i_{52} & i_{53} & i_{54} & i_{55} & i_{56} & i_{57} & i_{58} \\ i_{61} & i_{62} & i_{63} & i_{64} & i_{65} & i_{66} & i_{67} & i_{68} \end{pmatrix}.$$

$$I_2 = \begin{pmatrix} j_{11} & j_{12} & j_{13} & j_{14} & j_{15} & j_{16} & j_{17} & j_{18} \\ j_{21} & j_{22} & j_{23} & j_{24} & j_{25} & j_{26} & j_{27} & j_{28} \\ j_{31} & j_{32} & j_{33} & j_{34} & j_{35} & j_{36} & j_{37} & j_{38} \\ j_{41} & j_{42} & j_{43} & j_{44} & j_{45} & j_{46} & j_{47} & j_{48} \\ j_{51} & j_{52} & j_{53} & j_{54} & j_{55} & j_{56} & j_{57} & j_{58} \\ j_{61} & j_{62} & j_{63} & j_{64} & j_{65} & j_{66} & j_{67} & j_{68} \end{pmatrix}.$$

$$I_3 = \begin{pmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} & k_{17} & k_{18} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} & k_{27} & k_{28} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} & k_{37} & k_{38} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} & k_{47} & k_{48} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} & k_{57} & k_{58} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} & k_{67} & k_{68} \end{pmatrix}.$$

Si suponemos que en la matriz I_2 los elementos de la primera y segunda fila (color azul) son iguales a los elementos de la penúltima y última fila (color azul) de la matriz I_1, se tiene un traslape frontal o longitudinal representando una nueva matriz de 10×8 pixeles de la siguiente forma.

$$I_{TF} = \begin{pmatrix} i_{11} & i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17} & i_{18} \\ i_{21} & i_{22} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27} & i_{28} \\ i_{31} & i_{32} & i_{33} & i_{34} & i_{35} & i_{36} & i_{37} & i_{38} \\ i_{41} & i_{42} & i_{43} & i_{44} & i_{45} & i_{46} & i_{47} & i_{48} \\ i_{51}\ o\ j_{11} & i_{52}\ o\ j_{12} & i_{53}\ o\ j_{13} & i_{54}\ o\ j_{14} & i_{55}\ o\ j_{15} & i_{56}\ o\ j_{16} & i_{57}\ o\ j_{17} & i_{58}\ o\ j_{18} \\ i_{61}\ o\ j_{21} & i_{62}\ o\ j_{22} & i_{63}\ o\ j_{23} & i_{64}\ o\ j_{24} & i_{65}\ o\ j_{25} & i_{66}\ o\ j_{26} & i_{67}\ o\ j_{27} & i_{68}\ o\ j_{28} \\ j_{31} & j_{32} & j_{33} & j_{34} & j_{35} & j_{36} & j_{37} & j_{38} \\ j_{41} & j_{42} & j_{43} & j_{44} & j_{45} & j_{46} & j_{47} & j_{48} \\ j_{51} & j_{52} & j_{53} & j_{54} & j_{55} & j_{56} & j_{57} & j_{58} \\ j_{61} & j_{62} & j_{63} & j_{64} & j_{65} & j_{66} & j_{67} & j_{68} \end{pmatrix}.$$

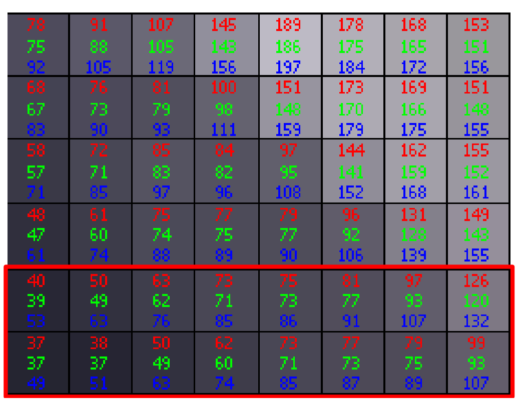
Por otro lado, si en la matriz I_3 los elementos de la primera y segunda columna (color rojo) son iguales a los elementos de la penúltima y última columna (color rojo) de la matriz I_1, se tiene un traslape lateral o transversal, representando una nueva matriz de 6 x 14 pixeles de la siguiente forma.

$$
I_{TL} = \begin{pmatrix}
i_{11} & i_{12} & i_{13} & i_{14} & i_{15} & i_{16} & i_{17}\ o\ k_{11} & i_{18}\ o\ k_{12} & k_{13} & k_{14} & k_{15} & k_{16} & k_{17} & k_{18} \\
i_{21} & i_{22} & i_{23} & i_{24} & i_{25} & i_{26} & i_{27}\ o\ k_{21} & i_{28}\ o\ k_{22} & k_{23} & k_{24} & k_{25} & k_{26} & k_{27} & k_{28} \\
i_{31} & i_{32} & i_{33} & i_{34} & i_{35} & i_{36} & i_{37}\ o\ k_{31} & i_{38}\ o\ k_{32} & k_{33} & k_{34} & k_{35} & k_{36} & k_{37} & k_{38} \\
i_{41} & i_{42} & i_{43} & i_{44} & i_{45} & i_{46} & i_{47}\ o\ k_{41} & i_{48}\ o\ k_{42} & k_{43} & k_{44} & k_{45} & k_{46} & k_{47} & k_{48} \\
i_{51} & i_{52} & i_{53} & i_{54} & i_{55} & i_{56} & i_{57}\ o\ k_{51} & i_{58}\ o\ k_{52} & k_{53} & k_{54} & k_{55} & k_{56} & k_{57} & k_{58} \\
i_{61} & i_{62} & i_{63} & i_{64} & i_{65} & i_{66} & i_{67}\ o\ k_{61} & i_{68}\ o\ k_{62} & k_{63} & k_{64} & k_{65} & k_{66} & k_{67} & k_{68}
\end{pmatrix}.
$$

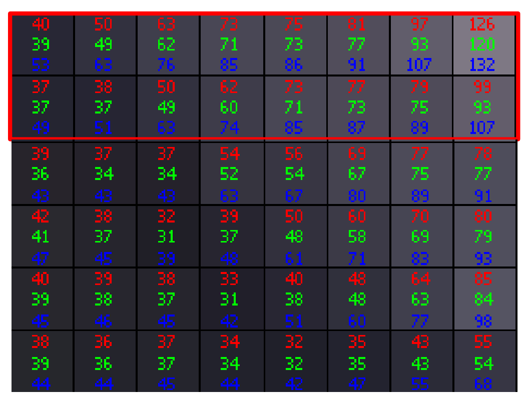
Finalmente, agrupando las matrices de traslape frontal y traslape lateral I_{TF} y I_{TL}, obtenemos una matriz final de la siguiente manera, donde los elementos de color verde representan los pixeles comunes entre las tres imágenes.

$$
I_{TF-TL} = \begin{pmatrix}
i_{11} & i_{12} & \cdots & i_{16} & i_{17}\ o\ k_{11} & i_{18}\ o\ k_{12} & k_{13} & \cdots & k_{18} \\
i_{21} & i_{22} & \cdots & i_{26} & i_{27}\ o\ k_{21} & i_{28}\ o\ k_{22} & k_{23} & \cdots & k_{28} \\
i_{31} & i_{32} & \cdots & i_{36} & i_{37}\ o\ k_{31} & i_{38}\ o\ k_{32} & k_{33} & \cdots & k_{38} \\
i_{41} & i_{42} & \cdots & i_{46} & i_{47}\ o\ k_{41} & i_{48}\ o\ k_{42} & k_{43} & \cdots & k_{48} \\
i_{51}\ o\ j_{11} & i_{52}\ o\ j_{12} & \cdots & i_{56}\ o\ j_{16} & i_{57}\ o\ j_{17}\ o\ k_{51} & i_{58}\ o\ j_{18}\ o\ k_{52} & k_{53} & \cdots & k_{58} \\
i_{61}\ o\ j_{21} & i_{62}\ o\ j_{22} & \cdots & i_{66}\ o\ j_{26} & i_{67}\ o\ j_{27}\ o\ k_{61} & i_{68}\ o\ j_{28}\ o\ k_{62} & k_{63} & \cdots & k_{68} \\
j_{31} & j_{32} & \cdots & j_{36} & j_{37} & j_{38} & \cdots & \cdots & \cdots \\
j_{41} & j_{42} & \cdots & j_{46} & j_{47} & j_{48} & \cdots & \cdots & \cdots \\
j_{51} & j_{52} & \cdots & j_{56} & j_{57} & j_{58} & \cdots & \cdots & \cdots \\
j_{61} & j_{62} & \cdots & j_{66} & j_{67} & j_{68} & \cdots & \cdots & \cdots
\end{pmatrix}.
$$

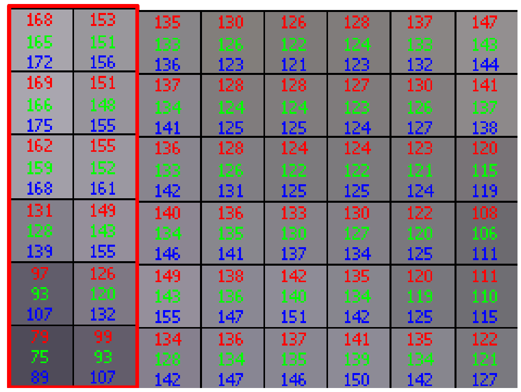
Para visualizar gráficamente la coincidencia de puntos comunes, consideramos tres imágenes de dimensión 6×8 pixeles, de la siguiente manera:



(a)



(b)



(c)

Figura 37. Representación de las matrices (a). I_1, (b). I_2 y (c). I_3.

En cada una de las imágenes anteriores podemos visualizar cada uno de los valores de cada pixel, con esto nos podemos dar cuenta que entre las tres imágenes existen puntos en común (marcados con color rojo) que a partir de estos valores podemos obtener una sola imagen como resultado de la unión de las tres imágenes anteriores, como se indica a continuación:

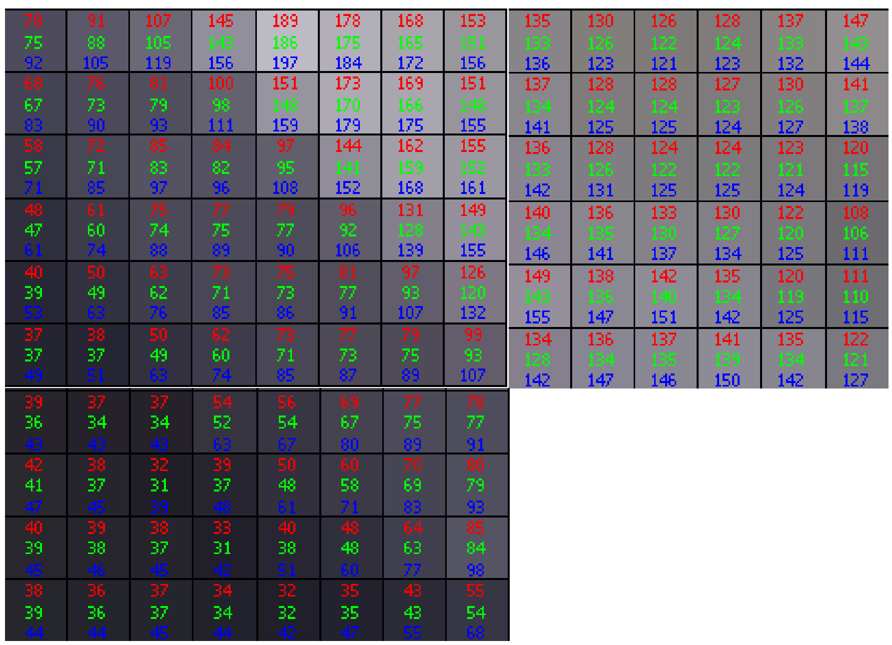


Figura 38. Imagen resultante.

5.4. Ortomosaico.

Un ortomosaico es el proceso de combinar varias imágenes o fotografías creando una única composición. El ortomosaico constituye la proyección orto-rectificada en dos dimensiones de la zona de estudio. Para construir el ortomosaico se debe estimar la calidad de las imágenes, luego se alinean las fotos, se estima la posición de cada fotografía, se genera una nube de puntos densa de alta calidad y finalmente se eligen los parámetros para construcción de la ortofoto: modo de mezcla mosaico, tamaño del píxel predeterminado dado que es el que brinda la máxima resolución efectiva, división de bloques de 1024×1024. (Instituto Geofísico del Perú, 2020).

En cuanto a la calidad visual que tiene un ortomosaico, esto está relacionado directamente con el solapamiento, traslape o la sobre posición de imágenes, la altura de vuelo y la luminosidad con la que se cuenta al momento de realizar la captura de imágenes aéreas. (Jarrín, 2020).

5.4.1. Caso 0.

Usando las 30 fotografías de la Figura 30.

Figura 39. Ortomosaico Caso 0.

5.4.2. Caso 1.

Usando las fotografías f01, f03, f05, f07, f09, f11, f13, f15, f17, f19, f21, f23, f25, f27, f29; 15 fotografías de la Figura 30, es decir, saltando una fotografía.

Figura 40. Ortomosaico Caso 1.

5.4.3. Caso 2.

Usando las fotografías f01, f04, f07, f10, f13, f16, f19, f22, f25, f28; 10 fotografías de la Figura 30, es decir, saltando dos fotografías.

Figura 41. Ortomosaico Caso 2.

5.4.4. Caso 3.

Usando las fotografías f01, f05, f09, f13, f17, f21, f25, f29; 8 fotografías de la Figura 30, es decir, saltando tres fotografías.

Figura 42. Ortomosaico Caso 3.

5.4.5. Caso 4.

Usando las fotografías f01, f06, f11, f16, f21, f26; 6 fotografías de la Figura 30, es decir, saltando cuatro fotografías.

Figura 43. Ortomosaico Caso 4.

5.4.6. Caso 5.

Usando las fotografías f01, f07, f13, f19, f25; 5 fotografías de la Figura 30, es decir, saltando cinco fotografías.

Figura 44. Ortomosaico Caso 5.

5.4.7. Caso 6.

Usando las fotografías f01, f08, f15, f22, f29; 5 fotografías de la Figura 30, es decir, saltando seis fotografías.

Figura 45. Ortomosaico Caso 6.

5.4.8. Caso 7.

Usando las fotografías f01, f09, f17, f25; 4 fotografías de la Figura 30, es decir, saltando siete fotografías.

Figura 46. Ortomosaico Caso 7.

5.4.9. Caso 8.

Usando las fotografías f01, f10, f19, f28; 4 fotografías de la Figura 30, es decir, saltando ocho fotografías.

Figura 47. Ortomosaico Caso 8.

5.4.10. Caso 9.

Usando las fotografías f01, f11, f21; 3 fotografías de la Figura 30, es decir, saltando nueve fotografías.

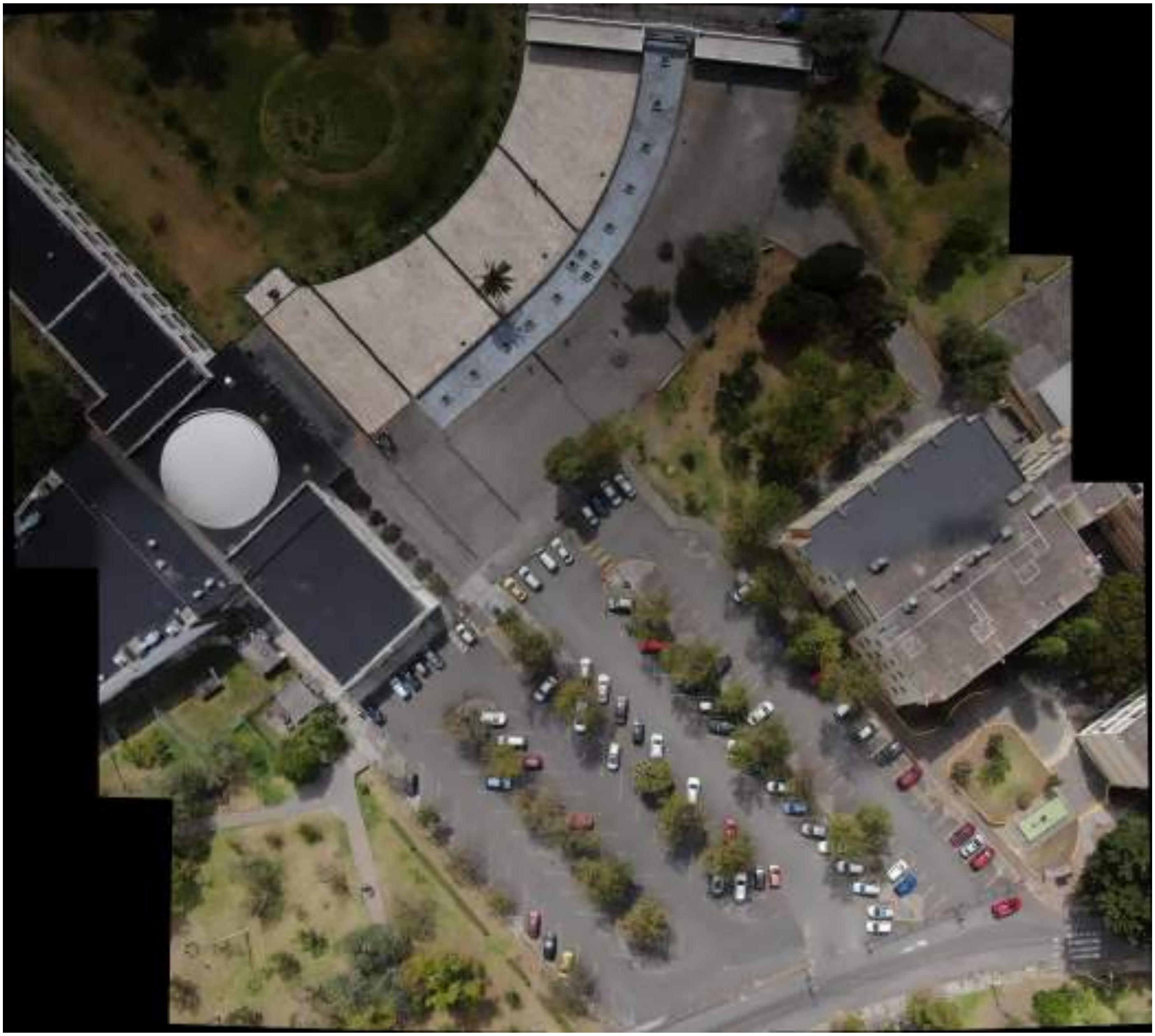


Figura 48. Ortomosaico Caso 9.

5.4.11. Caso 10.

Usando las fotografías f01, f12, f23; 3 fotografías de la Figura 30, es decir, saltando diez fotografías.

Figura 49. Ortomosaico Caso 10.

Observación: La fotografía f23 no se observa en el ortomosaico resultante.

5.4.12. Caso 11.

Usando las fotografías f01, f13, f25; 3 fotografías de la Figura 30, es decir, saltando once fotografías, se produce error en el algoritmo y no devuelve ningún resultado.

5.4.13. Caso 12.

Usando las fotografías f01, f14, f27; 3 fotografías de la Figura 30, es decir, saltando doce fotografías.

Figura 50. Ortomosaico. Caso 12.

Observación: La fotografía f27 no se observa en el ortomosaico resultante.

5.4.14. Caso 13.

Usando las fotografías f01, f15, f29; 3 fotografías de la Figura 30, es decir, saltando trece fotografías, el algoritmo se ejecuta pero no despliega ningún resultado.

5.4.15. Caso 14.

Usando las fotografías f01, f16: 2 fotografías de la Figura 30, es decir, saltando catorce fotografías, el resultado es igual que en el Caso 13.

Tabla de resultados.

Siguiendo el mismo procedimiento para los casos donde se redimensiona las fotografías al 50% (2000×1125 pixeles) y 10% (400×225 pixeles) de la dimensión original, se tiene los resultados siguientes:

	Casos														
Dimensiones	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
100%	C	C	C	C	C	C	C	C	C	C	OB	E	OB	NR	NR
50%	C	C	C	C	C	C	C	C	C	C	C	OB	NR	NR	NR
10%	C	C	C	C	C	C	C	NR	OB	OB	OB	NR	NR	OB	NR

Donde: **C:** Correcto, **E:** Error, **OB:** Observación (faltan algunas fotografías en el ortomosaico), **NR:** Ningún resultado (ortomosaico).

Con la anterior tabla de resultados podemos observar que conforme se redimensiona la fotografía y el número de fotografías a no tener en cuenta aumenta, el ortomosaico va a tener fallas. Por ejemplo si tenemos en cuenta las dimensiones originales de las fotografías y conforme hacemos los saltos de fotografías, el ortomosaico óptimo se obtendrá en el Caso 9 (salto de 9 fotografías), por otro lado, si redimensionamos las fotografías a un 10% y de igual manera realizamos los saltos de fotografías, el ortomosaico óptimo se obtendrá en el Caso 6 (salto de 6 fotografías).

Si observamos a detalle las fotografías de la Figura 30, se puede observar un cambio en la iluminación en cada una de ellas, esta variación de iluminación pudo deberse a algún cambio ambiental, por ejemplo, una nubosidad presentada al momento de tomar la fotografía. Este suceso se puede considerar como un elemento fundamental que afectó en algunos casos al reconocimiento de los puntos característicos comunes y por ende al resultado del algoritmo utilizado para la obtención del ortomosaico.

Conclusiones.

En este trabajo se ha tratado de manera general algunos conceptos matemáticos y métodos utilizados para poder recrear un ortomosaico en 2 dimensiones de la Universidad Central del Ecuador.

Para un buen resultado del ortomosaico se necesita que la fotografía sea nítida y con buena iluminación, pues de lo contrario, en una foto borrosa, movida o de poca resolución o iluminación, no se podrá identificar con precisión los puntos característicos comunes y se obtendrá un ortomosaico con imágenes faltantes o no se observara ningún resultado como se evidenció en los casos realizados.

Además, como se trata de una investigación no tan detallada no se ha tomado en consideración características importantes como estimar la mejor calidad de las imágenes con las cuales obtener el mejor rendimiento, el modo de mezcla del ortomosaico, el tamaño del píxel predeterminado dado que es el que brinda la máxima resolución efectiva, por ello, es necesario ampliar la investigación a un tratado de imágenes previo.

Al realizar este trabajo y al ser un estudio inicial, algunas características para la toma de datos (fotografías) no fueron estudiadas previamente, como es el caso del porcentaje de traslape y solape óptimo, ya que este procedimiento hubiese colaborado en mejorar el rendimiento del equipo computacional y tiempo empleado, pero, se logró un resultado satisfactorio en cada uno de los casos realizados.

El algoritmo desarrollado en Python no es en su totalidad un algoritmo eficiente, ya que necesita resumirse o mejorarse con otros métodos o funciones que optimicen tiempo y recursos.

Con el desarrollo de este trabajo se espera demostrar que la matemática interviene en cualquier campo de estudio, en este caso se evidenció su importancia en la fotogrametría y en particular en el desarrollo de un modelo u ortomosaico en 2 dimensiones.

Referencias.

Drone, C. (2019). *Para que sirve el traslape en Fotogrametría con RPAS | by Coatza Drone | Medium*. https://medium.com/@coatzadroneoficial/para-que-sirve-el-traslape-en-fotogrametría-con-rpas-2949b2ddf21b

Duarte Jiménez, K. T. (2018). *Evaluación de desempeño de distintos software para la generación de ortofotomosaicos a partir de imagenes tomadas con un vehículo aéreo no tripulado*.

Figueiras, S. (2018). *Integración de nubes de puntos generadas a partir de técnicas de fotogrametría aérea por multicorrelación en zonas urbanizadas.* http://oa.upm.es/53250/1/TESIS_MASTER_SERGIO_FIGUEIRAS_SANCHEZ.pdf

Gómez Ortega, A. R. (2018). *ALGORITMO PARA LA GENERACIÓN DE ORTOMOSAICOS A PARTIR DE IMÁGENES AÉREAS TOMADAS POR DRONES EN LA AGRICULTURA*.

He, X., Zemel, R. S., & Mnih, and V. (2006). *Topological Map Learning from Outdoor Image Sequences*.

Hernández, S. L. (2017). *Generación de Ortoimagenes usando Vehículos Aéreos no Tripulados aplicado a la Agricultura*.

Instituto Geofísico del Perú. (2020). *Levantamiento topográfico mediante fotogrametría aérea con dron y mediciones GPS de Alto Larán y Río Chico*. *511*. www.igp.gob.pe

Instituto Geográfico Agustín Codazzi. (2002). *¿Qué es la fotogrametría? | Instituto Geográfico Agustín Codazzi*. https://www.igac.gov.co/es/contenido/que-es-la-fotogrametria

Jarrín, A. (2020). *ANÁLISIS DE MODELOS DIGITALES DE TERRENO PARA LA OBTENCIÓN DE UN MAPA DE DENSIDAD DE DRENAJE MEDIANTE SOFTWARES FOTOGRAMÉTRICOS Y SISTEMAS DE INFORMACIÓN GEOGRÁFICA PARA IMÁGENES OBTENIDAS CON UAV, RESULTADOS APLICADOS A ESTUDIOS MORFOMÉTRICOS EN LADERAS.* Escuela Politécnica Nacional.

Li, D., Sun, H., & Wang, H. (2015). An improved SIFT algorithm for image stereo matching. *Xinan Jiaotong Daxue Xuebao/Journal of Southwest Jiaotong University*, *50*(3), 490–496. https://doi.org/10.3969/j.issn.0258-2724.2015.03.017

Lowe, D. G. (2011). *Object Recognition from Local Scale-Invariant Features*.

Min, Z., Jiguo, Z., & Xusheng, X. (2012). Panorama Stitching Based on SIFT Algorithm and Levenberg-Marquardt Optimization. *Physics Procedia*, *33*, 811–818. https://doi.org/10.1016/J.PHPRO.2012.05.139

Olvera, P., David, R., Manuel, D. R., Ortega, P., Carlos, J., Antonio, F. M., & Saúl, T. A. (2013). *Procesamiento y análisis de imágenes mediante el algoritmo SIFT en OpenCV .* 195–200.

Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W. C., Bismpigiannis, T., Grzeszczuk, R., Pulli, K., & Girod, B. (2008). Outdoors augmented reality on mobile phone using Loxel-based visual feature organization. *Proceedings of the 1st International ACM Conference on Multimedia Information Retrieval, MIR2008, Co-located with the 2008 ACM International Conference on Multimedia, MM'08*, *October*, 427–434. https://doi.org/10.1145/1460096.1460165

Tatay, T. (2003). *5 Programas Interesantes para Crear Fotos Panorámicas*. https://www.dzoom.org.es/programas-para-hacer-fotos-panoramicas/

Universidad Miguel Hernández de Elche. (2021). *umh1782 2020-21 Lección 006-5 - Detección de Bordes en Imágenes: Comparación funciones - YouTube*. https://www.youtube.com/watch?v=XmdTp1K-2A0&ab_channel=UniversidadMiguelHernándezdeElche

Vision First Principles of Computer. (2021). *SIFT Descriptor | SIFT Detector - YouTube*. https://www.youtube.com/watch?v=IBcsS8_gPzE&t=13s&ab_channel=FirstPrinciplesofComputerVision

Zhang, W., & Košecká, J. (2005). Localization based on building recognition. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, *2005-Septe*. https://doi.org/10.1109/CVPR.2005.489