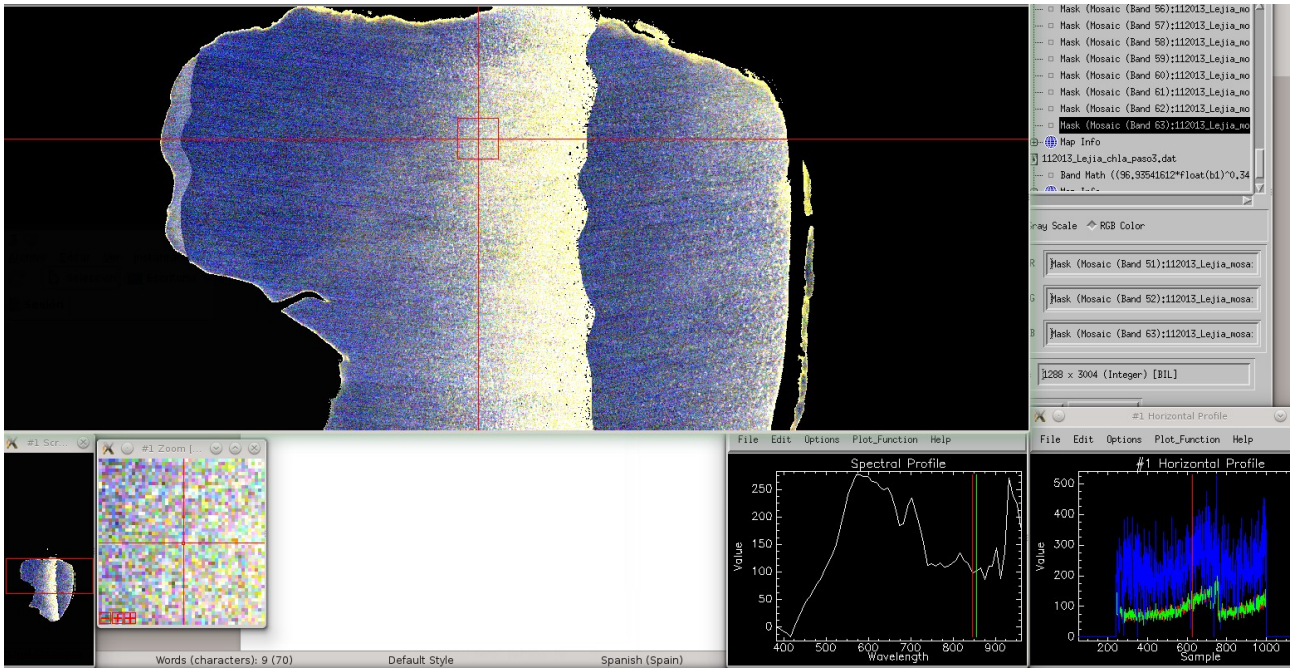
Corrección de imágenes ENVI con errores mediante interpolación matricial difusa.

La toma de imágenes hiperespectrales, a veces, conlleva algunos errores debido a las condiciones ambientales, como nubes, o problemas de enfoques o reflejos incorrectos.

El resultado es obtener imágenes que se salen de los resultados deseados, como la siguiente:



Este documento se trata de buscar un modo de corrección bajo la hipótesis siguiente:

Hipótesis:

La zona a corregir corresponde a un espacio de continuidad espacio-temporal en medio de las zonas correctas, por lo tanto es plausible que su corrección corresponda a una relación de continuidad que se puede representar mediante una interpolación entre los bordes de cada lado adyacentes a la zona a corregir.

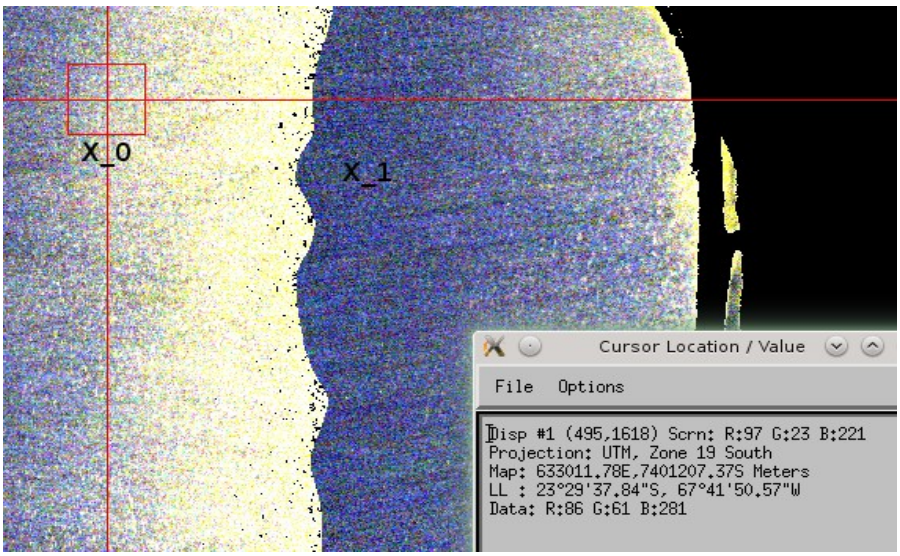
Metodología:

Ahora, para resolver tal propuesta será necesario implementar y seguir una serie de pasos para lograr una corrección aproximada a la realidad, esto es conseguir la modificación de la imagen con algún método que permita extraer su información y devolverla sin modificar las otras variables.

Paso 1. Detección de bordes.

Un set de imágenes ENVI hiperespectrales consiste en una serie de imágenes mapa de bits relacionadas por el lugar geográfico, el mismo, y las diferentes longitud de onda en que son tomadas cada una, permitiendo obtener información del lugar a distintos niveles.

Por lo tanto, al ser mapa de bits, es posible conocer y manipular el valor del pixel y su posición dentro de la matriz imágen. De aquí que se puede obtener el valor de los puntos de corte en el eje horizontal: x_0 a la izquierda y x_1 a la derecha.



$X_0 = 630$ y $X_1 = 770$ representan las posiciones horizontales X de pixeles en la matriz.

Paso 2. Definir ecuación de interpolación correctiva mediante Interpolación Lineal (IL).

Se propone una función de interpolación del tipo lineal $y(x) = m * x + c$

Cada punto de la imágen, pixel, tiene un valor natural $(x, y) \in \mathbb{N}^2$ que varía en 8 bits.

$V(x, y) = (\{0,..,2^8 - 1\}; \{0,..,2^8 - 1\})$ Haciendo ceteris paribus, dejando constante, en eje vertical 'y', definimos la pendiente 'm' y 'c' en:

$$m = \frac{V(X_1) - V(X_0)}{X_1 - X_0}$$

$$c = \frac{V(X_0) X_1 - V(X_1) X_0}{X_1 - X_0}$$

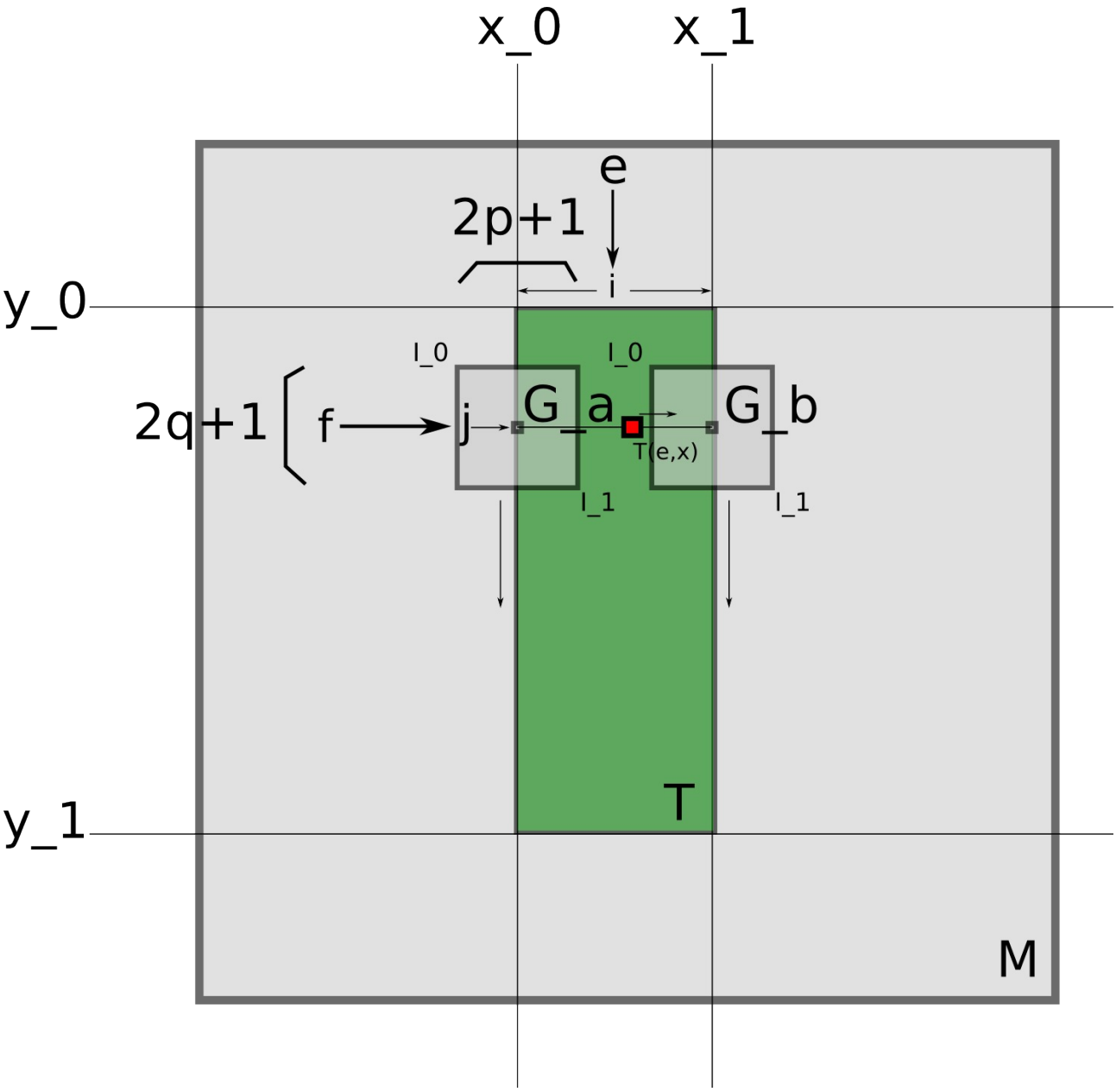
Por lo que la función de interpolación lineal correctiva quedaría en:

$$V(X) = mX + c$$

Paso 3. Definir ecuación de interpolación correctiva mediante Interpolación Difusa (ID).

Esta es una forma de crear la zona de reemplazo de manera difusa, utilizando varios parámetros.

Primero observar la siguiente imágen y su composición, nos referiremos a ella. Esta imágen representa una matriz de una bi-dimensional a la cual se le desea corregir la zona en verde.

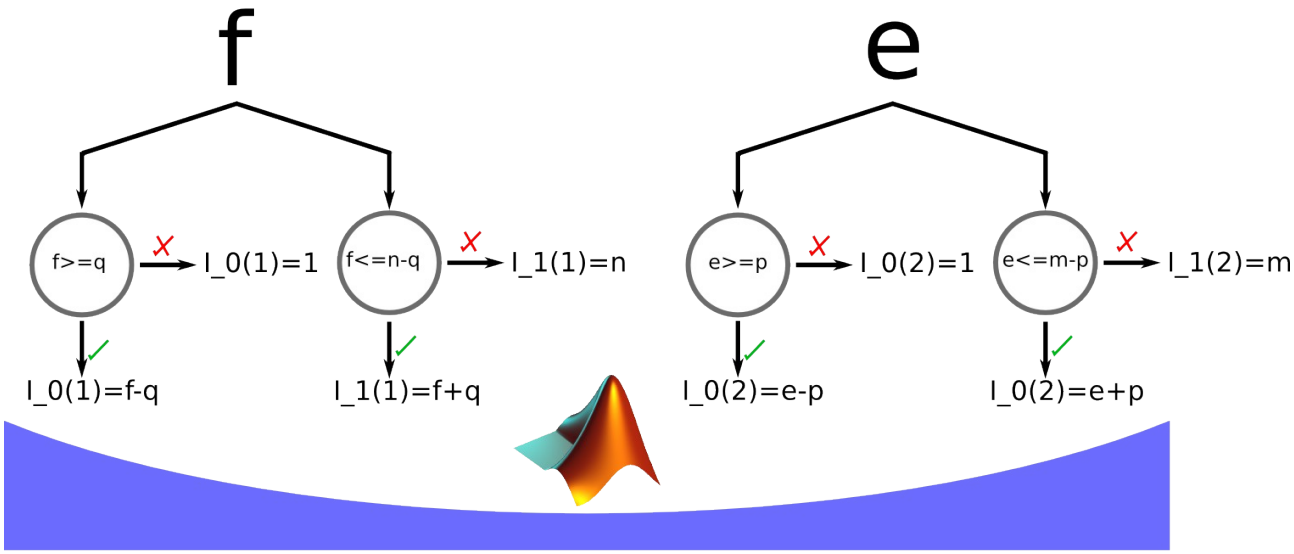


Nos interesa obtener los valores de T que formarán parte de la nueva zona corregida de la imagen.

Para eso es necesario seguir cuidadosamente lo siguiente:

1. Establecer los límites de la zona a corregir, asignar los valores de estos límites a (x_0, y_0) y (x_1,y_1).
2. Determinar la orientación de la corrección, si es en sentido vertical u horizontal.
3. Determinar la zona de adyacencia en la cantidad de pixeles a considerar adyacentes al pixel de borde analizado, p en la horizontal, q en la vertical. Lo que resultan **dos grupos** de matrices **G_a** y **G_b.** Esto se define en base a la operación lógica, comparativa, que depende

de los tamaños y posición de los pixeles borde.



$$I\_0=[\max\{f-q,1\},\max\{e-p,1\}]$$
$$I\_1=[\min\{f+q,n\},\min\{e+p,m\}]$$

4. Definir un factor de peso (0 a 1) que se le da a la interpolación lineal, de manera que: $Q = Factor * Interpolación(lineal) + (1 - Factor) * Interpolación(difusa)$
5. Obtener el valor de la serie de pixeles entre los pixeles borde, a partir de una selección aleatoria de un valor de cada uno, considerando además un factor de distancia entre ambos. Es el vector de interpolación difusa.
   - $\beta = [1 : distancia(V_0, V_1)]$ Y $\alpha = 1 - \beta$
   - $ga = G_a(aleatorio(filas(G_a)), aleatorio(columnas(G_a)))$
   - $gb = G_b(aleatorio(filas(G_b)), aleatorio(columnas(G_b)))$
     $T(e,:) = \beta * ga + \alpha * gb$
6. Luego, se recorta una matriz que corresponde a la zona a corregir original.
   - $M_o = M(X, Y)$
7. Se reemplaza la zona a corregir de la matriz original por una operación lineal entre un factor de reserva (de 0 a 1) y la matriz T.
   - $M(X, Y) = Factor_{reserva} * M_o + (1 - Factor_{reserva}) * T$
8. Repetir por cada capa de color (R,G,B).
9. Se guarda la matriz como imagen, en el formato original.

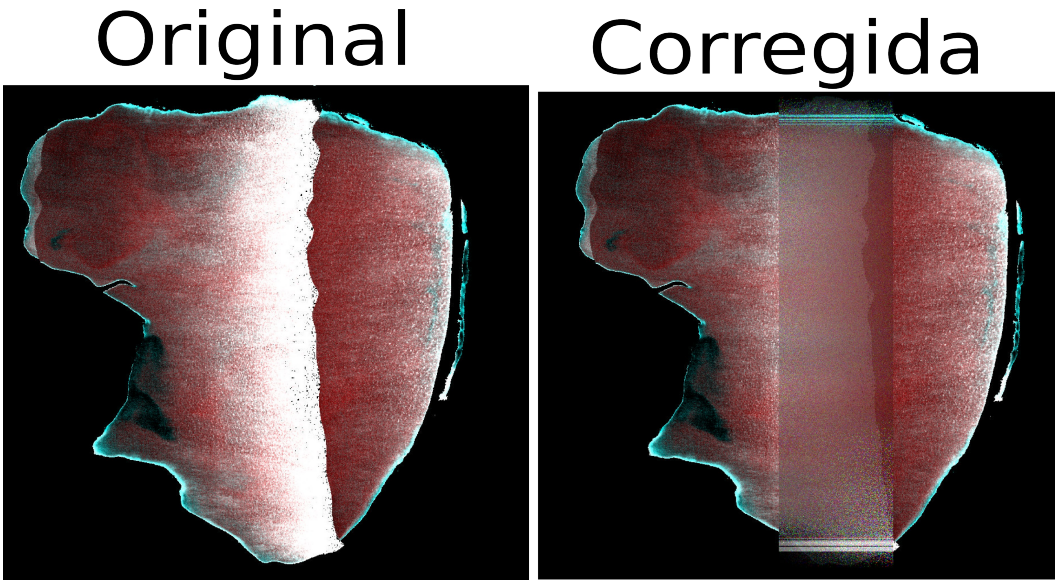
Resultados.

Este método permite corregir imperfecciones de la toma de imágenes bajo ciertos supuestos, como resultado entrega una aproximación bastante más acertada a la realidad. Sin embargo no es precisa.

Para encontrar precisión se requerirá de un estudio estadístico y comparativo entre imágenes bien tomadas a las que se les aplique una corrección, imágenes bien tomadas que no se le aplique corrección con imágenes de la misma zona corregidas, etcétera.

Además será necesario determinar los parámetros correspondientes a los ratios (q,p) y los factores

con que se opera en el método, de manera que se permita determinar los valores más adecuados según algún caso determinado.



Consideraciones.

El diseño de este algoritmo se puede implementar bajo algún lenguaje de programación matemático que permita o tenga en sus librerías la posibilidad de leer imágenes como matrices y un set de funciones que permitan interpretarlas.

En este caso se ha utilizado matlab para definir las funciones y operar las matrices.

interpolacion_matrix

```
%M=interpolacion_matrix(V,x,y, b,c,p,q,factor)
%interpola una zona de la matriz
%V: matriz de valores
%[x_0 x_1]: valores de posición horizontal
%[y_0 y_1]: valores de posición vertical
%b: orientación 1: horizontal, 0: vertical
%tener cuidado en definir bordes horizontales y verticales c=0 todo el
%registro vertical, c=1 todo registro horizontal, c=2 parcial, se toman
%ambos vectores es una interpolación difusa, basada en la asgincación de un valor obtenido
a partir
% de las zonas contiguas de cada pixel frontera, seleccionado al azar
%la zona contigua es criterio del usuario, basada un un valor p: cantidad
%de pixeles en torno a la vertical, y q cantidad de pixeles en torno a la
%horizontal
%p= es el ratio horizontal que recoge a la izq y derecha del punto elegido
%q= es el ratio vertical que recoge hacia arriba y abajo del punto elegido
%ambos determinan una matriz de dimensión 2p+1 x 2q+1 con centro el punto
%escogido
%factor: es un par de valores que le asigna un peso la interpolación lineal
%difusa, tiene valor de 0 a 1.
%$factor_intercambio: es el factor que se aplica al intercambio de la zona, primero a
%la misma zona y luego a la matriz de reemplazo
function M=interpolacion_matrix(V,x,y, b,c,p,q,factor, factor_intercambio)
%verificar que sean mayores que 0 y dentro de V
[n_v, m_v] = size(V);
M=double(V);
%%se evalua si corresponde operar la función
if length(x) <=2 || length(y) <=2  || factor <=1 && factor >=0 || factor_intercambio <=1 &&
factor_intercambio >=0
%%se construyen los vectores de factor y factor_intercambio
factor=[factor,1-factor];
```

```
factor_intercambio=[factor_intercambio,1-factor_intercambio];
x_0=x(1);
x_1=x(2);
y_0=y(1);
y_1=y(2);

if c==0
   y_0=1;
   y_1=n_v;
elseif c==1
   x_0=1;
   x_1=m_v;
elseif c==2

else
        msgbox('Error al ingresar zona de registro')
end

if x_0>=1 && x_1>x_0  && x_1<=m_v && y_0>=1 && y_1>y_0 && y_1<=n_v
   if b==1
      X=[x_0:x_1];
      Y=[y_0:y_1];
      lx=length(X);
      ly=length(Y);
      for j=y_0:y_1
      m(j,1)=(V(j,x_1)-V(j,x_0))/(x_1-x_0);
      c(j,1)=(V(j,x_0)*x_1-V(j,x_1)*x_0)/(x_1-x_0);
      %%matriz grupo A-varian las columnas
      I_0(1,:)=[max(j-q,1), max(x_0-p,1)];
      I_1(1,:)=[min(j+q,n_v),min(x_0+p,m_v)];
      %%matriz grupo B
      I_0(2,:)=[max(j-q,1), max(x_1-p,1)];
      I_1(2,:)=[min(j+q,n_v),min(x_1+p,m_v)];
      %%Grupos de matrices de adyacencia en torno a pixel frontera
      G_a=V(I_0(1,1):I_1(1,1),I_0(1,2):I_1(1,2));
      G_b=V(I_0(2,1):I_1(2,1),I_0(2,2):I_1(2,2));
      %%tamaÃ±o de matrices de adyacencia
      [n_a, m_a]=size(G_a);
      [n_b, m_b]=size(G_b);
      %%se determina aletoriamente que valor de matrices de adyacencia
      %%obtener
      for k=1:lx
      r_a_n=randi(n_a);
      r_a_m=randi(m_a);
      r_b_n=randi(n_b);
      r_b_m=randi(m_b);
      ga(k)=G_a(r_a_n,r_a_m);
      gb(k)=G_b(r_b_n,r_b_m);
      end
      %vector factor de posición del cada pixel
      beta=[1:lx]/lx;
      alpha=1-beta;
      %%Se obtienen los vectores de interpolacion lineal y difusa pura.
      vectorlineal=round(double(m(j,1)))*X+c(j,1);
      vectordifuso=alpha.*round(double(ga))+beta.*round(double(gb));
      %#se suman linealmente
      vector=round((factor(1)*vectorlineal+factor(2)*vectordifuso));
      %%se asocia la línea correspondiente de T con el vector.
      T(j,:)=vector;
      end
      %%se rescata la matriz a corregir original
      M_o=double(M(Y,X));
```

```
        %%se determina la nueva matriz
        M(Y,X)=factor_intercambio(1)*M_o+factor_intercambio(2)*T;
    elseif b==0
        X=[x_0:x_1];
        Y=[y_0:y_1];
        lx=length(X);
        ly=length(Y);
        for i=x_0:x_1
        m(1,i)=(V(y_1,i)-V(y_0,i))/(y_1-y_0);
        c(1,i)=(V(y_0,i)*y_1-V(y_1,i)*y_0)/(y_1-y_0);
        %matriz grupo A-varian las filas
        I_0(1,:)=[max(y_0-q,1), max(i-p,1)];
        I_1(1,:)=[min(y_0+q,n_v),min(i+p,m_v)];
        %matriz grupo B
        I_0(2,:)=[max(y_1-q,1), max(i-p,1)];
        I_1(2,:)=[min(y_1+q,n_v),min(i+p,m_v)];
        %
        G_a=V(I_0(1,1):I_1(1,1),I_0(1,2):I_1(1,2));
        G_b=V(I_0(2,1):I_1(2,1),I_0(2,2):I_1(2,2));
        %
        [n_a, m_a]=size(G_a);
        [n_b, m_b]=size(G_b);
        %
        for k=1:ly
        r_a_n=randi(n_a);
        r_a_m=randi(m_a);
        r_b_n=randi(n_b);
        r_b_m=randi(m_b);
        ga(k)=G_a(r_a_n,r_a_m);
        gb(k)=G_b(r_b_n,r_b_m);
        end
        %
        beta=[1:ly]/ly;
        alpha=1-beta;
        %
        vectorlineal=m(j,1)*X+c(j,1);
        vectordifuso=alpha.*round(double(ga))+beta.*round(double(gb));
        %
        vector=round((vectorlineal*factor(1)+vectordifuso*factor(2)));
        %

        T(:,i)=vector;
        end
        M_o=double(M(Y,X));
        M(Y,X)=factor_intercambio(1)*M_o+factor_intercambio(2)*T;
    else
      msgbox('Error al ingresar orientación 0 para vertical o bien 1 para horizontal')
    end
else
    msgbox('Error al ingresar tus datos de interpolación')
end
else
    msgbox('Error en largo de vetores x o y o factor')
end

end
```

Luego, se invoca la función para operar las imágenes de prueba, bajo factores determinados por el

usuario.

Lector TIFF

```
DIRECTORIO=pwd;
cd(DIRECTORIO);
lista={'banda674.tif'; 'banda702.tif'; 'banda711.tif'; 'banda749.tif'};
[n m] = size(lista); %n indica posicion lista(n,1)
i=1;
file=char(strcat(DIRECTORIO,'/imagenes/',lista(i,1)));
[X,R] = geotiffread(file);
INFO = geotiffinfo(file);
UTM.geokey=INFO.GeoTIFFTags.GeoKeyDirectoryTag;
UTM.tipo=INFO.Projection;
factor=.2;
factor_intercambio=0.1;
p=25;
q=25;
b=1;
c=0;
x=[600,800];
y=[1 3];
V1=X(:,:,1);
V2=X(:,:,2);
V3=X(:,:,3);
color={'R';'G';'B'};
bits=INFO.BitDepth;
for i=1:n
    figure
    for j=1:3
M(:,:,j)=interpolacion_matrix(X(:,:,j),x,y, b,c,p,q,factor,factor_intercambio);
Im(:,:,j)=mat2gray(M(:,:,j));
[M_16b(:,:,j),Map_16b]=gray2ind(Im(:,:,j),2^bits-1);
subplot(1,5,j)
imshow(M(:,:,j))
titulo=strcat(lista(i,1),' en :',color{j});
title(titulo);
    end
subplot(1,5,4)
imshow(M)
title([lista(i,1),',corregida.'])
subplot(1,5,5)
imshow(X)
title(lista(i,1))
size(M_16b)
%guardar imagenes corregidas:
  file_rev=char(strcat(DIRECTORIO,'/imagenes/rev_',lista(i,1)));
  geotiffwrite(file_rev,M_16b,R,'GeoKeyDirectoryTag',UTM.geokey);
end
```