Tarea_1_proc_imagenes

May 31, 2021

0.0.1 Tarea 1: Procesamiento de imágenes.

- Profesor: Miguel Carrasco.
- Alumno: Héctor Henríquez Leighton.

[315]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

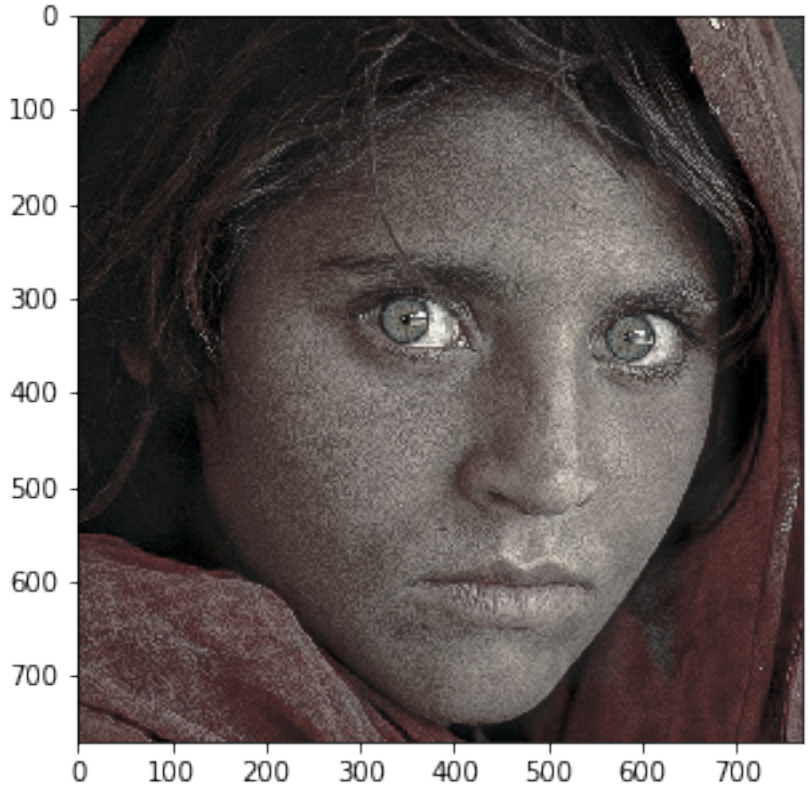
[316]:
```
## Carga de imagen
img = cv2.imread('natgeo_afgirl.png')
img = cv2. cvtColor(img, cv2.COLOR_BGR2RGB) ## Se lee imagen con canales en␣
 ↪orden RGB

print("Height:{},\nWidth:{}".format(img.shape[0], img.shape[1]))
plt.figure(figsize=(10,10))
plt.imshow(img)
plt.show()
```

```
Height:1800,
Width:1200
```

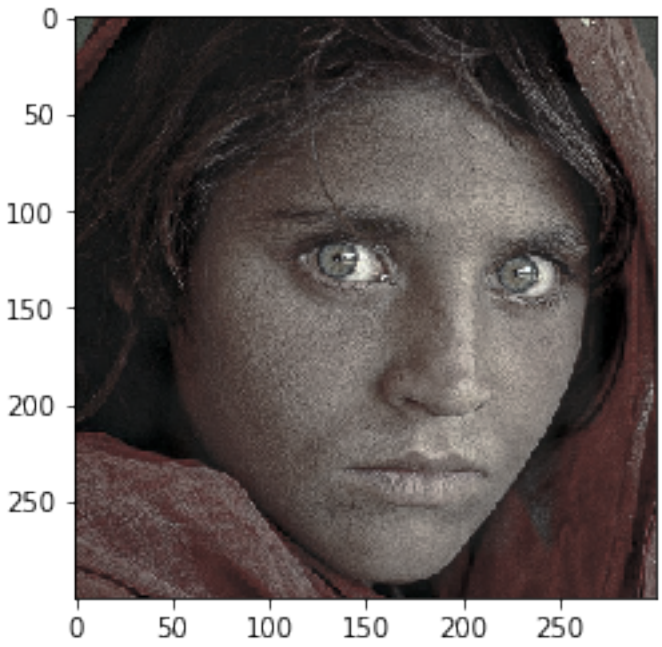

0
200
400
600
800
1000
1200
1400
1600
0
200
400
600
800
1000

[317]:
```
## ROI
face_roi = img[440:1210,240:1010]

print("resolución ROI:", face_roi.shape)
plt.figure(figsize=(5,5))
plt.imshow(face_roi)
plt.show()
```

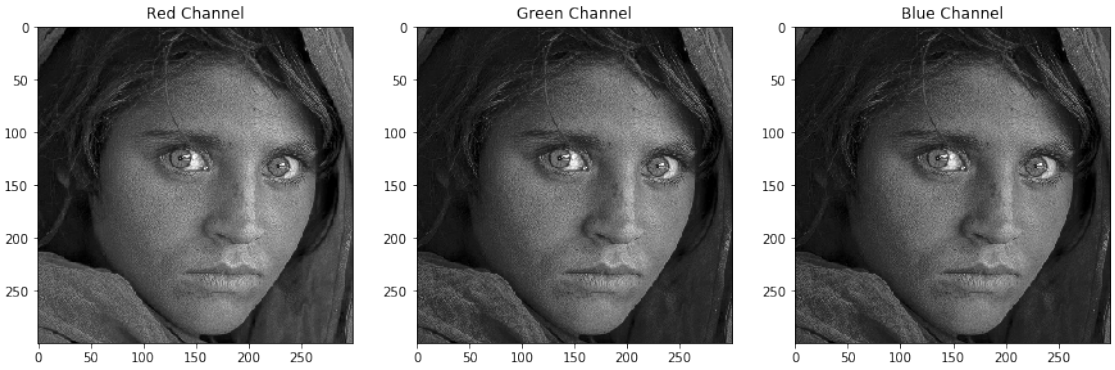
```
resolución ROI: (770, 770, 3)
```



[318]:
```
width = 300
height = 300
dim = width,height
imgResize = cv2.resize(face_roi,dim)

plt.figure()
plt.imshow(imgResize, cmap='gray')
plt.show()
```



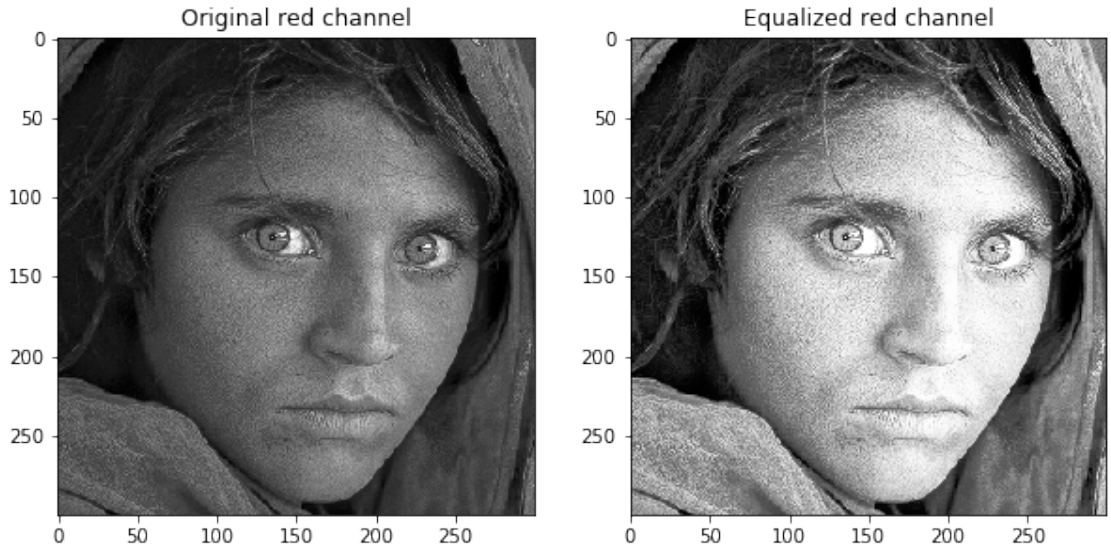
[319]:
```
red, green, blue = cv2.split(imgResize)

f = plt.figure(figsize=(15,15))

f.add_subplot(1, 3, 1)
plt.title("Red Channel")
plt.imshow(red, cmap = 'gray')

f.add_subplot(1, 3, 2)
plt.title("Green Channel")
plt.imshow(green, cmap = 'gray')

f.add_subplot(1, 3, 3)
plt.title("Blue Channel")
plt.imshow(blue, cmap = 'gray')

plt.show()
```



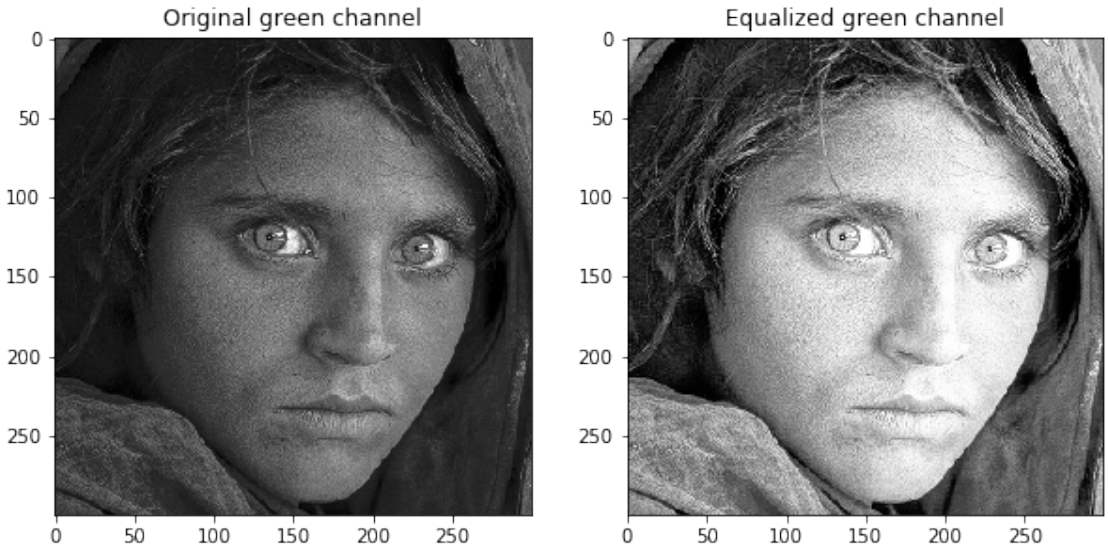
- **La diferencia mas notoria es la intensidad del pañuelo, que en el canal rojo tiene valores mas altos, representados en esta imagen como un gris, en comparación con los canales verde y azul en que el pañuelo se ve mas oscuro, debido a la menor cantidad de esos colores.**

0.0.2 Ecualización de histogramas para cada canal

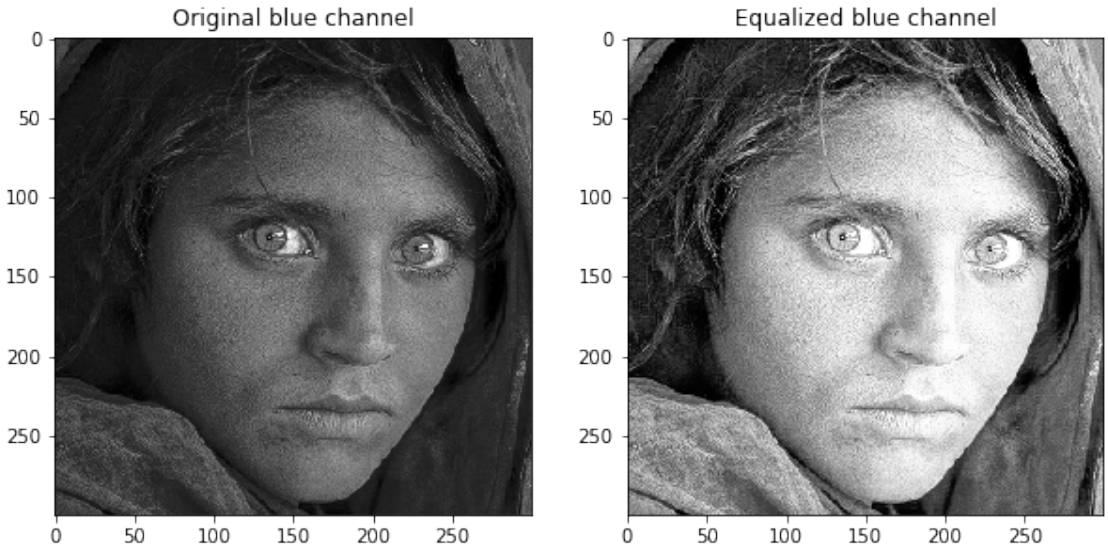
[320]:
```
## Histograma original red
histRed = cv2.calcHist([imgResize], [0], None, [256], [0,255])

## Imagen canal rojo ecualizada
redEq = cv2.equalizeHist(red)

## Imagen canal rojo original
f = plt.figure(figsize=(10,5))
f.add_subplot(1, 2, 1)
plt.imshow(red, cmap="gray")
plt.title('Original red channel')

## Imagen canal rojo ecualizado
f.add_subplot(1, 2, 2)
plt.imshow(redEq, cmap="gray") ## Imagen ecualizada
plt.title('Equalized red channel')
plt.show()
```



[321]:
```
## Ecualizacion canal verde
greenEq = cv2.equalizeHist(green)

## Imagen canal verde original.
f = plt.figure(figsize=(10,5))
f.add_subplot(1, 2, 1)
plt.imshow(green, cmap="gray")
plt.title('Original green channel')

## Imagen canal verde ecualizado.
f.add_subplot(1, 2, 2)
plt.imshow(greenEq, cmap="gray") ## Imagen ecualizada
plt.title('Equalized green channel')
plt.show()
```



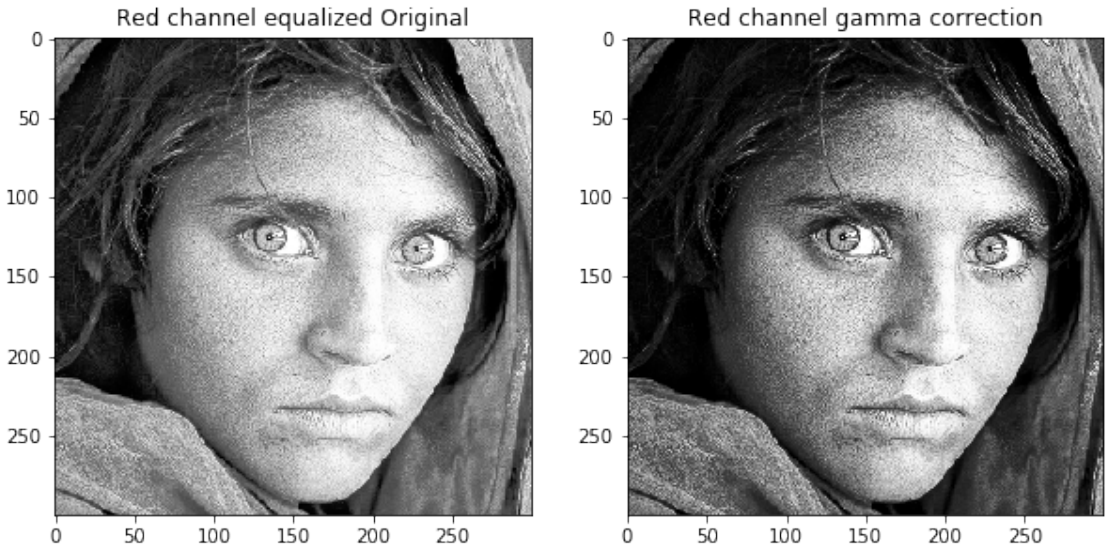
[322]:
```
## Ecualizacion canal azul
blueEq = cv2.equalizeHist(blue)

# newHistRed = cv2.calcHist([redEq], [0], None, [256], [0,255])

## Imagen Canal azul original
f = plt.figure(figsize=(10,5))
f.add_subplot(1, 2, 1)
plt.imshow(blue, cmap="gray")
plt.title('Original blue channel')

## Imagen canal azul ecualizado
f.add_subplot(1, 2, 2)
plt.imshow(blueEq, cmap="gray") ## Imagen ecualizada
plt.title('Equalized blue channel')
plt.show()
```



0.1 Corrección Gama

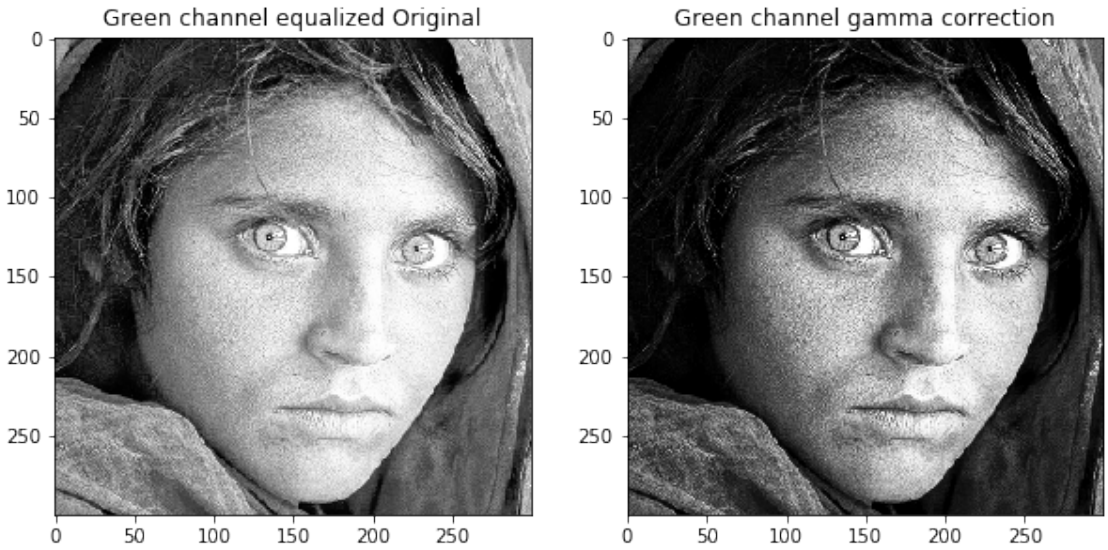
[323]:
```
## Se define función de correción de gama
def correccion_gama(img,factor):
    img = img/255.0
    img = cv2.pow(img,factor)
    return np.uint8(img*255)

factorGama = 1.6
gamaRed = correccion_gama(redEq,factorGama)

print("Factor de correción gama: {}".format(factorGama))
f = plt.figure(figsize=(10,10))
f.add_subplot(1, 2, 1)
plt.imshow(redEq, cmap="gray")
plt.title('Red channel equalized Original')

f.add_subplot(1, 2, 2)
plt.imshow(gamaRed, cmap="gray") ## Imagen ecualizada
plt.title('Red channel gamma correction')
plt.show()
```

```
Factor de correción gama: 1.6
```



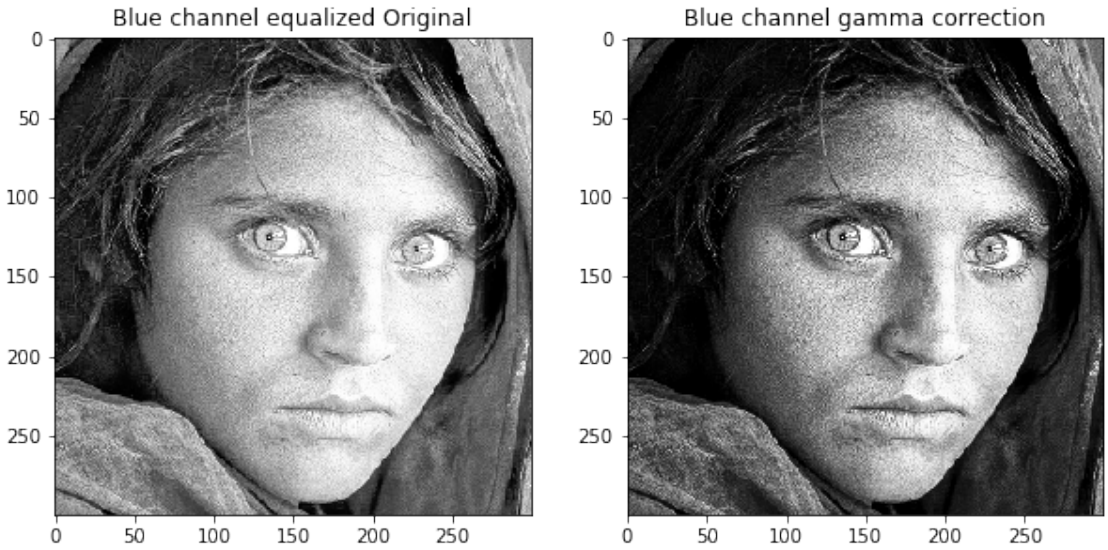
[324]:
```
##Canal verde
factorGama = 1.9
gamaGreen = correccion_gama(greenEq,factorGama)

print("Factor de correción gama: {}".format(factorGama))

f = plt.figure(figsize=(10,10))
f.add_subplot(1, 2, 1)
plt.imshow(greenEq, cmap="gray")
plt.title('Green channel equalized Original')

f.add_subplot(1, 2, 2)
plt.imshow(gamaGreen, cmap="gray") ## Imagen ecualizada
plt.title('Green channel gamma correction')
plt.show()
```

```
Factor de correción gama: 1.9
```



[325]:
```
# Canal azul
factorGama = 1.8
gamaBlue = correccion_gama(blueEq,factorGama)

print("Factor de correción gama: {}".format(factorGama))

f = plt.figure(figsize=(10,10))
f.add_subplot(1, 2, 1)
plt.imshow(blueEq, cmap="gray")
plt.title('Blue channel equalized Original')

f.add_subplot(1, 2, 2)
plt.imshow(gamaBlue, cmap="gray") ## Imagen ecualizada
plt.title('Blue channel gamma correction')
plt.show()
```

```
Factor de correción gama: 1.8
```

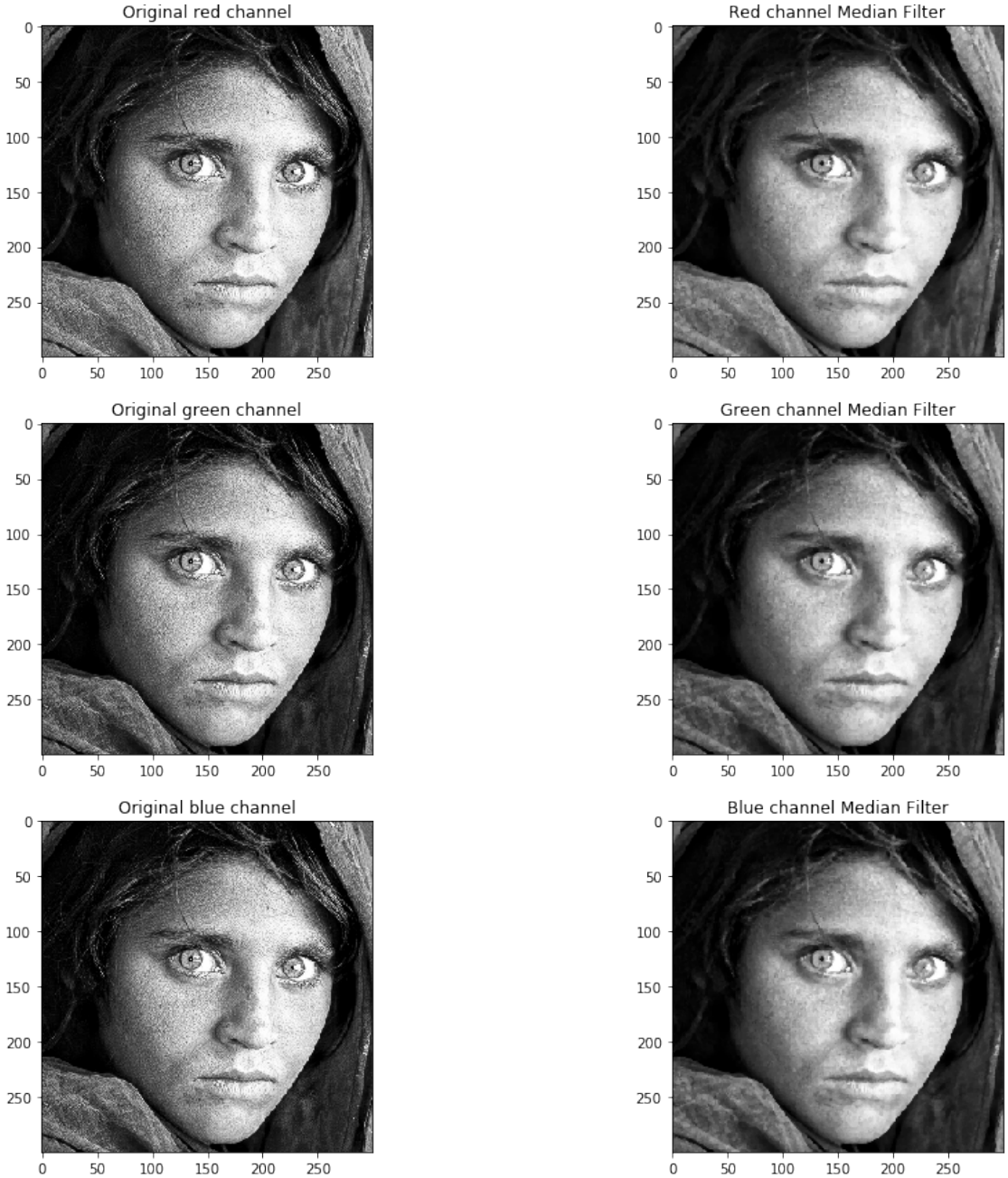


0.2 Filtro de mediana

[328]:
```
#filtro mediana sobre las imagenes postprocesadas con corrección gama
window = 3
redMedian = cv2.medianBlur(gamaRed,window)
greenMedian = cv2.medianBlur(gamaGreen,window)
blueMedian =  cv2.medianBlur(gamaBlue,window)

print("Ventana de 3x3")
f = plt.figure(figsize=(15,15))
f.add_subplot(3, 2, 1)
plt.imshow(gamaRed, cmap="gray")
plt.title('Original red channel')
f.add_subplot(3, 2, 2)
plt.imshow(redMedian, cmap="gray") ## Imagen ecualizada
plt.title('Red channel Median Filter')

f.add_subplot(3, 2, 3)
plt.imshow(gamaGreen, cmap="gray")
plt.title('Original green channel')
f.add_subplot(3, 2, 4)
plt.imshow(greenMedian, cmap="gray") ## Imagen ecualizada
plt.title('Green channel Median Filter')

f.add_subplot(3, 2, 5)
plt.imshow(gamaBlue, cmap="gray")
plt.title('Original blue channel')
f.add_subplot(3, 2, 6)
```

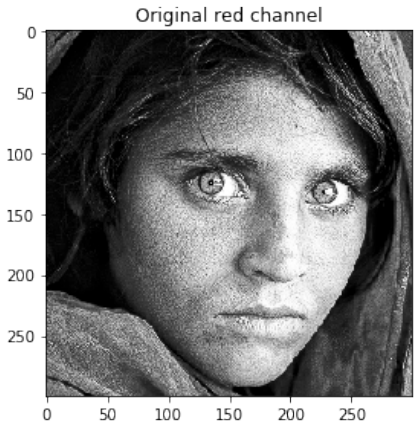
```
plt.imshow(blueMedian, cmap="gray") ## Imagen ecualizada
plt.title('Blue channel Median Filter')
plt.show()
```

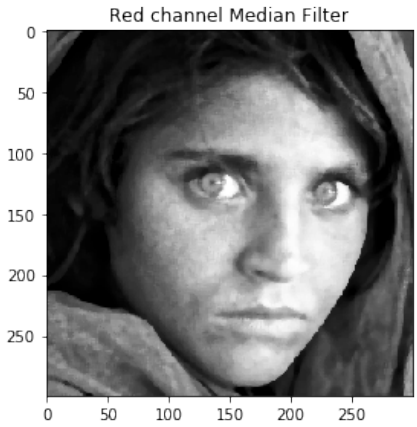
Ventana de 3x3

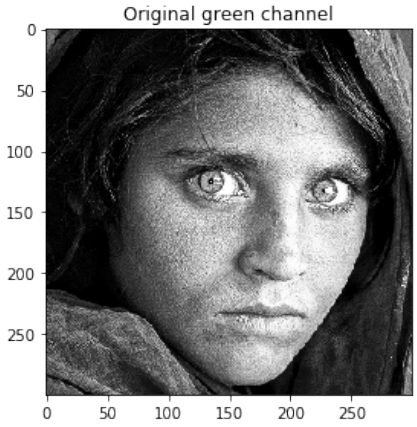


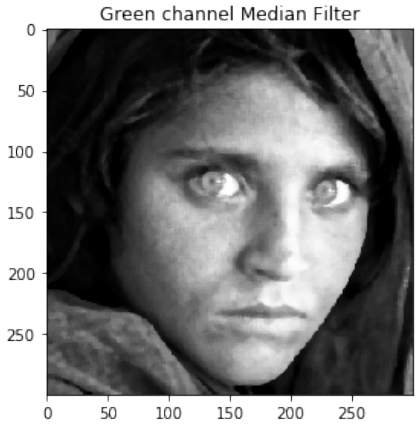
[327]:
```
#filtro no lineal (mediana)
window = 5
redMedian = cv2.medianBlur(gamaRed,window)
greenMedian = cv2.medianBlur(gamaGreen,window)
blueMedian =  cv2.medianBlur(gamaBlue,window)

print("Ventana de 5x5")
f = plt.figure(figsize=(15,15))
f.add_subplot(3, 2, 1)
plt.imshow(gamaRed, cmap="gray")
plt.title('Original red channel')
f.add_subplot(3, 2, 2)
plt.imshow(redMedian, cmap="gray") ## Imagen ecualizada
plt.title('Red channel Median Filter')

f.add_subplot(3, 2, 3)
plt.imshow(gamaGreen, cmap="gray")
plt.title('Original green channel')
f.add_subplot(3, 2, 4)
plt.imshow(greenMedian, cmap="gray") ## Imagen ecualizada
plt.title('Green channel Median Filter')

f.add_subplot(3, 2, 5)
plt.imshow(gamaBlue, cmap="gray")
plt.title('Original blue channel')
f.add_subplot(3, 2, 6)
plt.imshow(blueMedian, cmap="gray") ## Imagen ecualizada
plt.title('Blue channel Median Filter')
plt.show()
```

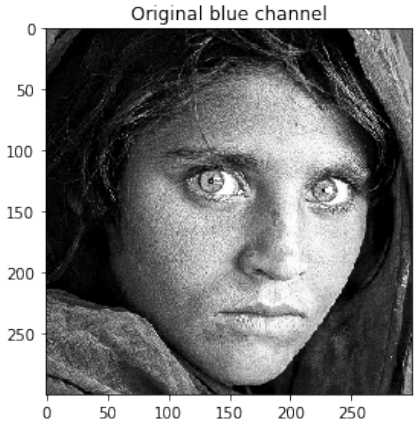
```
Ventana de 5x5
```

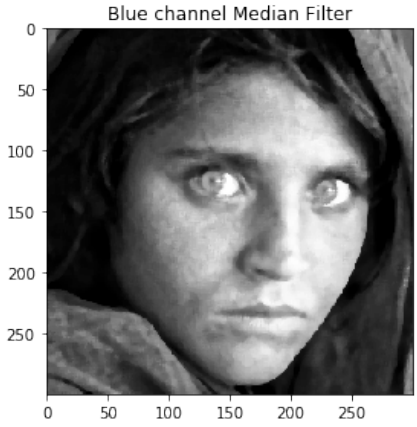








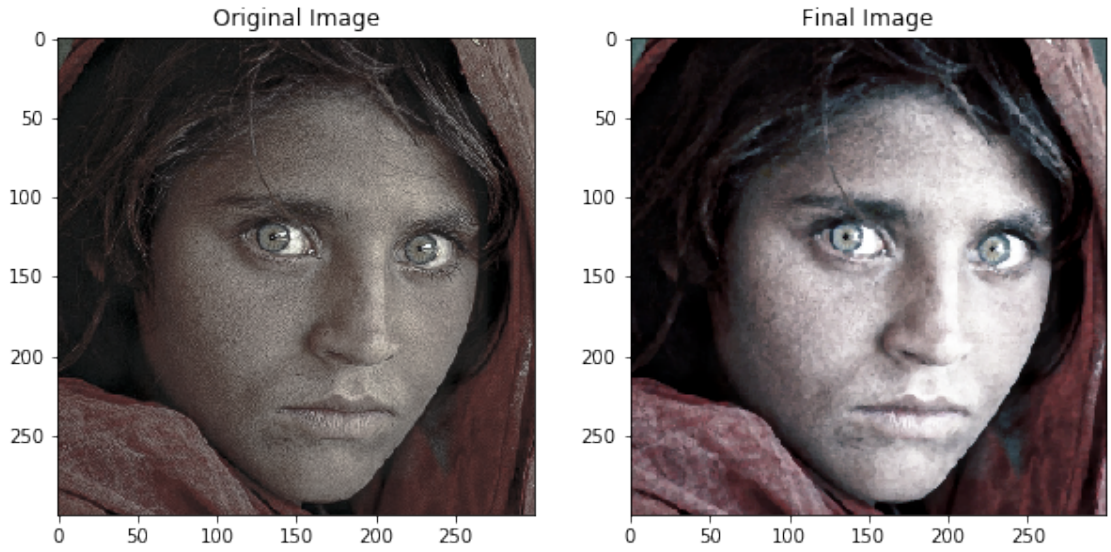




[329]:
```
## Imagen en color
R = redMedian
G = greenMedian
B = blueMedian

fullColor = cv2.merge([R,G,B])
```

```
f = plt.figure(figsize=(10,10))
f.add_subplot(1, 2, 1)
plt.imshow(imgResize, cmap="gray")
plt.title('Original Image')
f.add_subplot(1, 2, 2)
plt.imshow(fullColor, cmap="gray") ## Imagen ecualizada
plt.title('Final Image')
```

[329]: `Text(0.5, 1.0, 'Final Image')`



0.3 1. ASPECTOS TEÓRICOS DEL PROBLEMA

- El objetivo del procesamiento de esta imagen es lograr una mejora en la imagen original, ya sea por mejorar la intensidad de la luminancia, mejor ecualización de los canales de color o reducción del ruido.
- Para esto se debe procesar la imagen en sus distintos componentes.

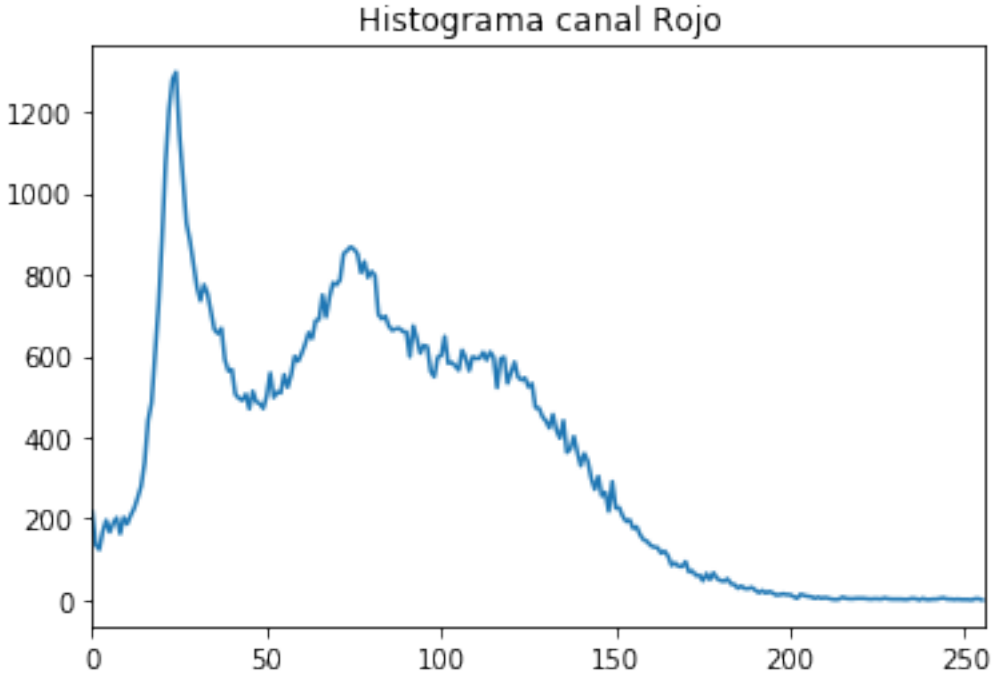
Estructura de la imagen

- Una imagen es una matriz bidimensional de pixeles que toman valores en 8 bit(entre 0 y 255). Además presenta un tercera dimensión que corresponde a la profundidad o canales de color.
- El orden de los canales de color corresponde a RGB (red, green and blue), pudiendo contener otros canales como luminancia u otros componentes del espectro electromágnético (UV, IR, rayos X, o emisión angosta de algunas moléculas como por ejemplo Hidrógeno).
- Esta estructura de arreglo multidimensional permite realizar operaciones matemáticas sobre los valores de los pixeles, lo que determina cambios en la forma que visualizamos la imagen.

Teoría asociada a las transformaciones

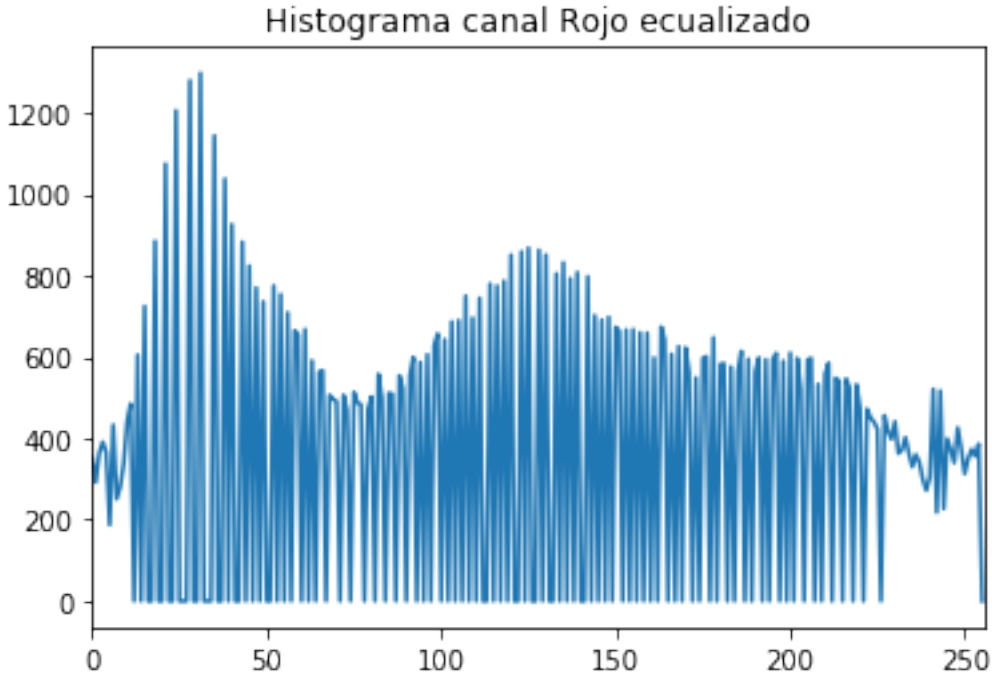
- En primer lugar se focaliza el procesamiento en la región de interés (ROI) definida por coordenadas(i,j) que delimitan el área donde se encuentra el rostro.
- Se define arbitrariamente una resolución de 300 x 300 pixeles para la imagen del rostro, lo cual se realiza mediante interpolación de los valores de los pixeles para llevar la matriz al tamaño esperado de 300 x 300. Esta conversión puede aumentar o disminuir la resolución dependiendo de la resolución original.
- Es necesario separar los canales de color de la imagen para realizar el procesamiento en forma individual.
- **Histograma de la imagen:**

[330]:
```
hist_red = cv2.calcHist([imgResize], [0], None, [256], [0,255])
plt.plot(hist_red)
plt.xlim([0,256])
plt.title("Histograma canal Rojo")
plt.show()
```



- Se calcula el histograma para cada canal de color.
- El histograma corresponde a la representación de frecuencias en la imagen para cada valor de pixeles, en el rango de la escala de la imagen (0 a 255 para 8 bit).
- Arriba podemos ver el ejemplo del histograma para el canal de color rojo en el que la mayoría de los pixeles se encuentran dentro del rango de valores entre 50 y 170, con un peak en un rango estrecho de aproximadamente 25.
- Nótese la ausencia de valores entre 180 y 250.

[331]:
```
redEq = cv2.equalizeHist(red)
NewhistRed = cv2.calcHist([redEq], [0], None, [256], [0,255])
plt.plot(NewhistRed)
plt.xlim([0,256])
plt.title("Histograma canal Rojo ecualizado")
plt.show()
```

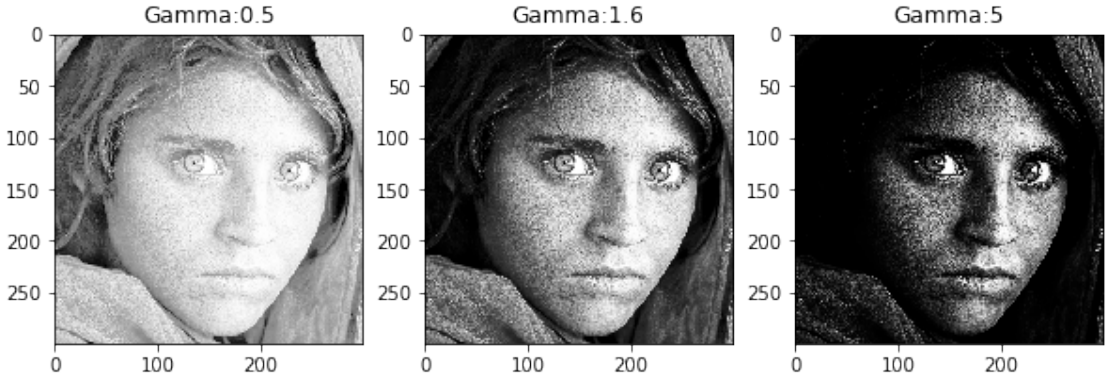


- Arriba podemos ver el histograma luego de la ecualización.
- Podemos notar que ahora todos los valores de los pixeles se encuentran mejor distribuidos dentro del rango de la imagen. Antes no habia pixeles con valores entre 200 y 250.
- Esta ecualización del histograma permite una mejora en el contraste general de la imagen.

- **Corrección gamma:**
- Para mejorar el brillo de la imagen se utiliza una técnica de mejoramiento en el espacio, adaptandola a la forma de percepción visual humana de las imágenes.
- De esta manera se aplica una función no lineal al brillo de la imagen, de manera de optimizar la luminosidad.

[332]:
```
factorGama1 = 0.5
exampleRed1= correccion_gama(redEq,factorGama1)

factorGama2 = 1.6
exampleRed2= correccion_gama(redEq,factorGama2)

factorGama3 = 5
```

```
exampleRed3= correccion_gama(redEq,factorGama3)

f = plt.figure(figsize=(10,10))
f.add_subplot(1, 3, 1)
plt.imshow(exampleRed1, cmap="gray")
plt.title('Gamma:{}'.format(factorGama1))

f.add_subplot(1, 3, 2)
plt.imshow(exampleRed2, cmap="gray") ## Imagen ecualizada
plt.title('Gamma:{}'.format(factorGama2))

f.add_subplot(1, 3, 3)
plt.imshow(exampleRed3, cmap="gray")
plt.title('Gamma:{}'.format(factorGama3))
plt.show()
```

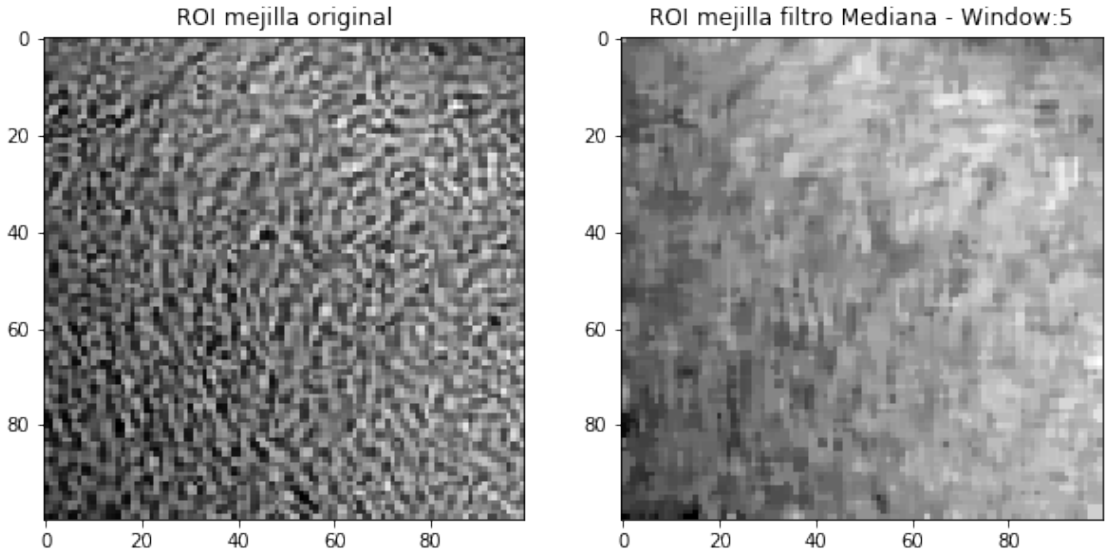


- **Ejemplo con 3 valores diferentes de Gamma.**

- **Reducción en el ruido de la imagen:**
- Luego de ecualizar y ajustar el brillo de cada canal de la imagen, se aplican técnicas de reducción del ruido.
- Existen diferentes filtros que consiguen reducir la cantidad de ruido, en particular se utilizó el filtro de mediana.
- El Filtro de mediana al usar la mediana de los valores de los pixeles contenidos en la ventana (3x3 o 5x5), elimina los outliers y esto lleva a una reducción en el ruido de la imagen.
- La percepción visual de este efecto es la reducción en el granulado presente en la imagen, con una visualización mas "suave" de la imagen. Esto también puede llevar a disminuir la nitidez de algunos detalles de la imagen, lo que puede servir por ejemplo para reducir las imperfecciones. Esto es frecuentemente utilizado en filtros de cara, para disminuir la irregularidad cutánea.

[333]:
```
## ROI ejemplo
example_roi = img[820:920,480:580]
rExample, gExample, bExample = cv2.split(example_roi)
```

```
window = 5
noiseReduction =  cv2.medianBlur(rExample,window)

f = plt.figure(figsize=(10,10))
f.add_subplot(1, 2, 1)
plt.imshow(rExample, cmap="gray")
plt.title("ROI mejilla original")

f.add_subplot(1, 2, 2)
plt.imshow(noiseReduction, cmap="gray")
plt.title("ROI mejilla filtro Mediana - Window:{}".format(window))
plt.show()
```



- **Podemos notar como el filtro de mediana suaviza la imagen y disminuye el ruido caracterizado por "granulado" de la imagen, con pixeles que tienes diferentes valores, algunos con valores extremos (cercanos al blanco 255 y otros cercanos al negro 0).**
- **Composición de imagen en color:**
- Dado que el procesamiento se realiza en forma independiente en cada canal de color, para generar una imagen completa a color es necesario unir nuevamente los 3 canales, conservando el orden RGB para guardar la imagen final.

0.4 Descripción de la Solución

- **Lectura de la imagen:**
- Se utilizó una imagen en color con formato PNG de 1800 x 1200 pixeles.
- Esta imagen está conformada por 3 canales en orden RGB (red, green and blue).
- Dado que la librería openCV lee las imagenes en orden BGR, se debe explicitar el orden de

los canales o bien modificarlos con la función **cvt.Color().**

- **ROI y Resize de la imagen:**
- En nuestra imagen, la resolución de la imagen del rostro (ROI) es de 810 x 770 pixeles, lo que implica una **reducción en la resolución original** al realizar la interpolación con el comando **cv2.resize().**
- **Separación de canales de color:**
- Luego de tener la imagen con área de interés seleccionada y la resolución requerida, se dividen los tres canales RGB para procesamiento individual con la función **cv.split()**.
- **Ecualización del histograma para cada canal:**
- Posteriormente se realiza una ecualización del histograma para cada canal de color, lo que mejora el contraste general de la imagen para cada uno de sus canales. Esto se realiza con la función **cv2.equalizeHist().**
- El proceso de ecualización del histograma lleva a una imagen con los valores de intensidad de los pixeles que estadisticamente se encuentra mejor distribuida en el rango de la imagen. Sin embargo, esto no garantiza que desde el punto de vista de la percepción visual, sea una imagen óptima.
- **Corrección gamma:**
- En esta etapa se realiza un ajuste manual y subjetivo del valor de corrección gamma, para cada uno de los colores.
- Considerando el monitor usado y preferencias personales de la percepción visual de cada canal, se optaron por los siguientes valores de factor gamma:
- R: 1.6
- G: 1.9
- B: 1.8

- **Aplicación de filtro de Mediana:**
- El objetivo de esta etapa es disminuir el ruido de la imagen y para esto se prueban filtros para cada canal de color con dos tamaños de ventanas distintos: 3x3 y 5x5.
- Al eliminar valores de outliers de los pixeles, reduce el ruido y suaviza la imagen. Sin embargo, esto tiene como consecuencia una pérdida en la nitidez de la imagen.
- Este efecto de disminución de la nitidez es mas notorio con el filtro aplicado con el mayor tamaño de ventana.
- Por esto se prefiere el tamaño de 3x3, ya que reduce el ruido pero no determina una disminución tan alta del detalle de la imagen.

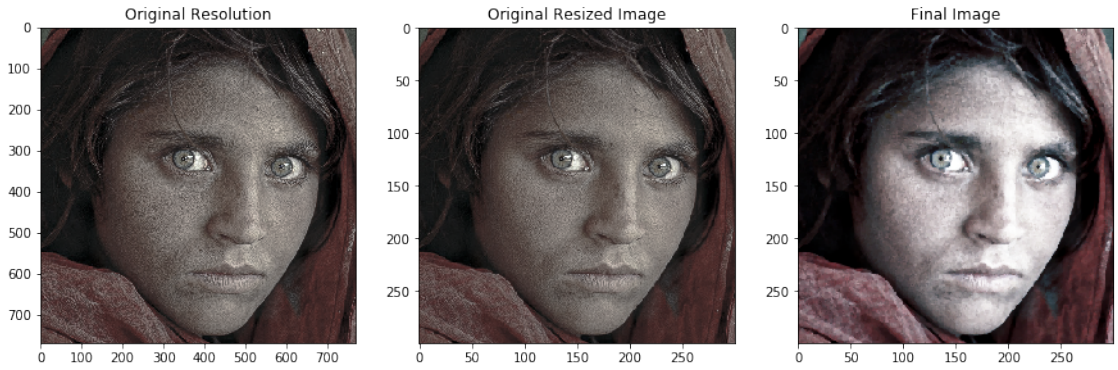
- **Generación de nueva imagen en color:**
- Finalmente los tres canales (RGB), ecualizados, balanceados con corrección gama y con el filtro de mediana aplicado, se fusionan nuevamente con la función **cv2.merge()** para formar una imagen en color.

0.5 Resultados y discusión:

[334]:
```
f = plt.figure(figsize=(15,15))
f.add_subplot(1, 3, 1)
plt.imshow(face_roi, cmap="gray")
plt.title('Original Resolution')
f.add_subplot(1, 3, 2)
plt.imshow(imgResize, cmap="gray")
```

```
plt.title('Original Resized Image')
f.add_subplot(1, 3, 3)
plt.imshow(fullColor, cmap="gray") ## Imagen ecualizada
plt.title('Final Image')
```

[334]: `Text(0.5, 1.0, 'Final Image')`



- En primer lugar podemos ver que la imagen original tenía una resolución mayor que la definida posterior para el ROI, lo cual lleva a una leve degradación de la calidad de la imagen.
- Desde el punto de vista de la intensidad de color y contraste, podemos ver que la imagen original contenia alta cantidad de color rojo por el pañuelo que cubre la cabeza de la niña, con un nivel de contraste no tan alto.
- Por otro lado, se puede apreciar que la imagen original contenia ruido representado por el granulado prominente existente en la fotografía.

- La ecualización mejoró bastante el contraste general de la imagen y especialmente evidente en los colores rojo y verde, haciendose mucho mas intensos el color del pañuelo y de los ojos.
- Al incrementar el contraste, se perdió detalle en cabello.
- Durante la corrección gamma, fue particularmente dificil equilibrar las zonas mas oscuras con las de mayor intensidad de brillo. En este caso la región central de la cara en relación la nariz y boca tienen alta intensidad, lo cual al bajar el gamma para favorecer el brillo de las regiones oscuras(cabello), se generaba una sobresaturación de la intensidad de la cara, lo cual no es posible de resolver al aplicar una transformación que actúe completamente sobre la imagen. Para poder resolver este problema, quizas hubiera sido útil una transformación selectiva sobre alguna área de la cara, por ejemplo aplicando una máscara sobre la zona de mayor brillo y que la transformación ocurra en la zona que no queda cubierta por la máscara.

- El filtro de mediana fue efectivo en reducir el ruido de la imagen original, generando una imagen mucho mas suave, lo que es evidente en la piel de la cara y en algunas zonas de la tela del pañuelo.
- Sin embargo, el filtro de mediana con ventana de 5x5 generó una pérdida significativa de la nitidez de la imagen.

- Finalmente, creo que el proceso consiguió mejorar el contraste de color y reducir el ruido de

la imagen.

[ ]: