Histogram equalization

Histogram equalization is the process of modifying the intensities of the image pixels to enhance the contrast. The human eye likes contrast! This is the reason that almost all camera systems use histogram equalization to make images look nice. The interesting thing is that the histogram equalization process is different for grayscale and color images. There's a catch when dealing with color images, and we'll see it in this recipe. Let's see how to do it.

How to do it...

1. Create a new Python file, and import the following packages:

```
import sys

import cv2
import numpy as np
```

2. Load the input image. We will use the image, `sunrise.jpg`:

```
# Load input image -- 'sunrise.jpg'
input_file = sys.argv[1]
img = cv2.imread(input_file)
```

3. Convert the image to grayscale and display it:

```
# Convert it to grayscale
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imshow('Input grayscale image', img_gray)
```

4. Equalize the histogram of the grayscale image and display it:

```
# Equalize the histogram
img_gray_histeq = cv2.equalizeHist(img_gray)
cv2.imshow('Histogram equalized - grayscale', img_gray_histeq)
```

5. In order to equalize the histogram of the color images, we need to follow a different procedure. Histogram equalization only applies to the intensity channel. An RGB image consists of three color channels, and we cannot apply the histogram equalization process on these channels separately. We need to separate the intensity information from the color information before we do anything. So, we convert it to YUV colorspace first, equalize the Y channel, and then convert it back to RGB to get the output. You can learn more about YUV colorspace at **http://softpixel.com/~cwright/programming/colorspace/yuv**. OpenCV loads images in the BGR format by default, so let's convert it from BGR to YUV first:

```
# Histogram equalization of color images
img_yuv = cv2.cvtColor(img, cv2.COLOR_BGR2YUV)
```

6. Equalize the Y channel, as follows:

```
img_yuv[:,:,0] = cv2.equalizeHist(img_yuv[:,:,0])
```

7. Convert it back to BGR:

```
img_histeq = cv2.cvtColor(img_yuv, cv2.COLOR_YUV2BGR)
```

8. Display the input and output images:

```
cv2.imshow('Input color image', img)
cv2.imshow('Histogram equalized - color', img_histeq)

cv2.waitKey()
```

9. The full code is given in the `histogram_equalizer.py` file that is already provided to you. The input image is shown, as follows:

10. The histogram equalized image looks like the following: