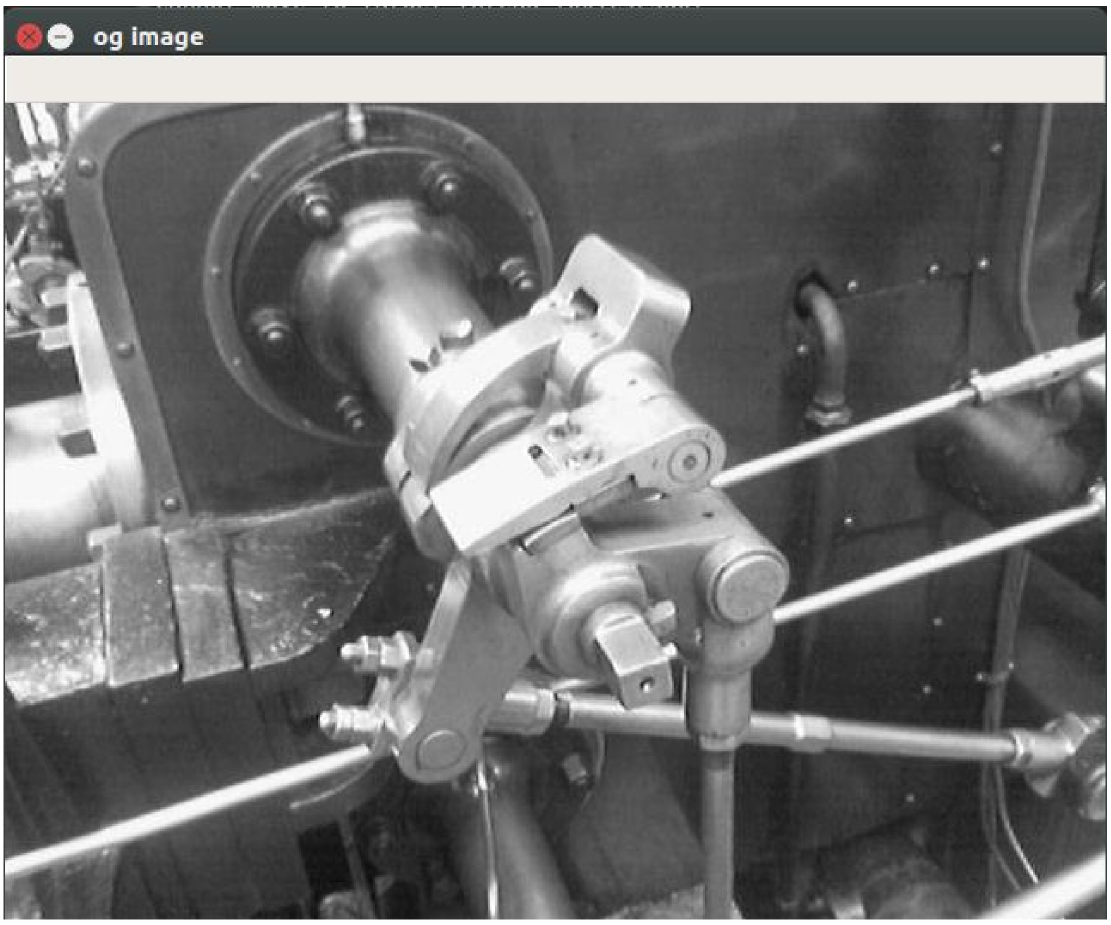
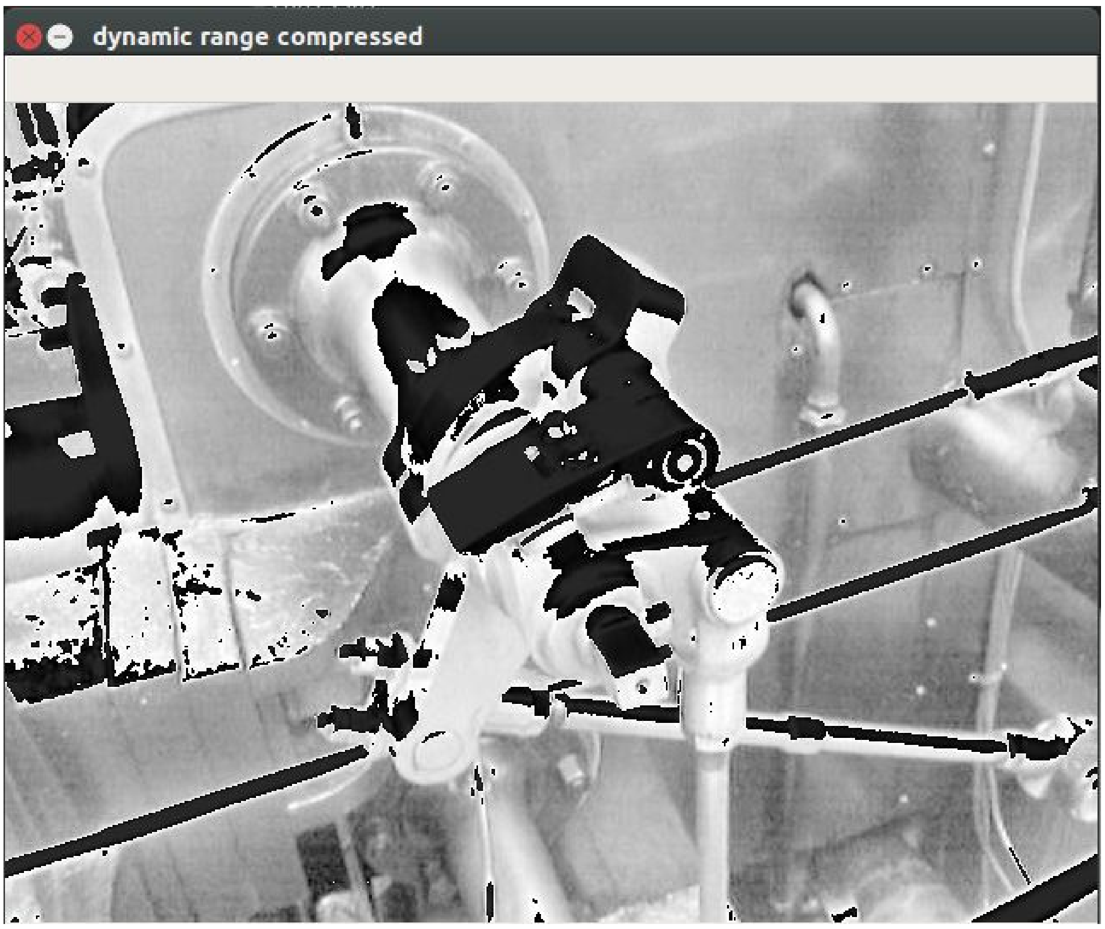
CODE:

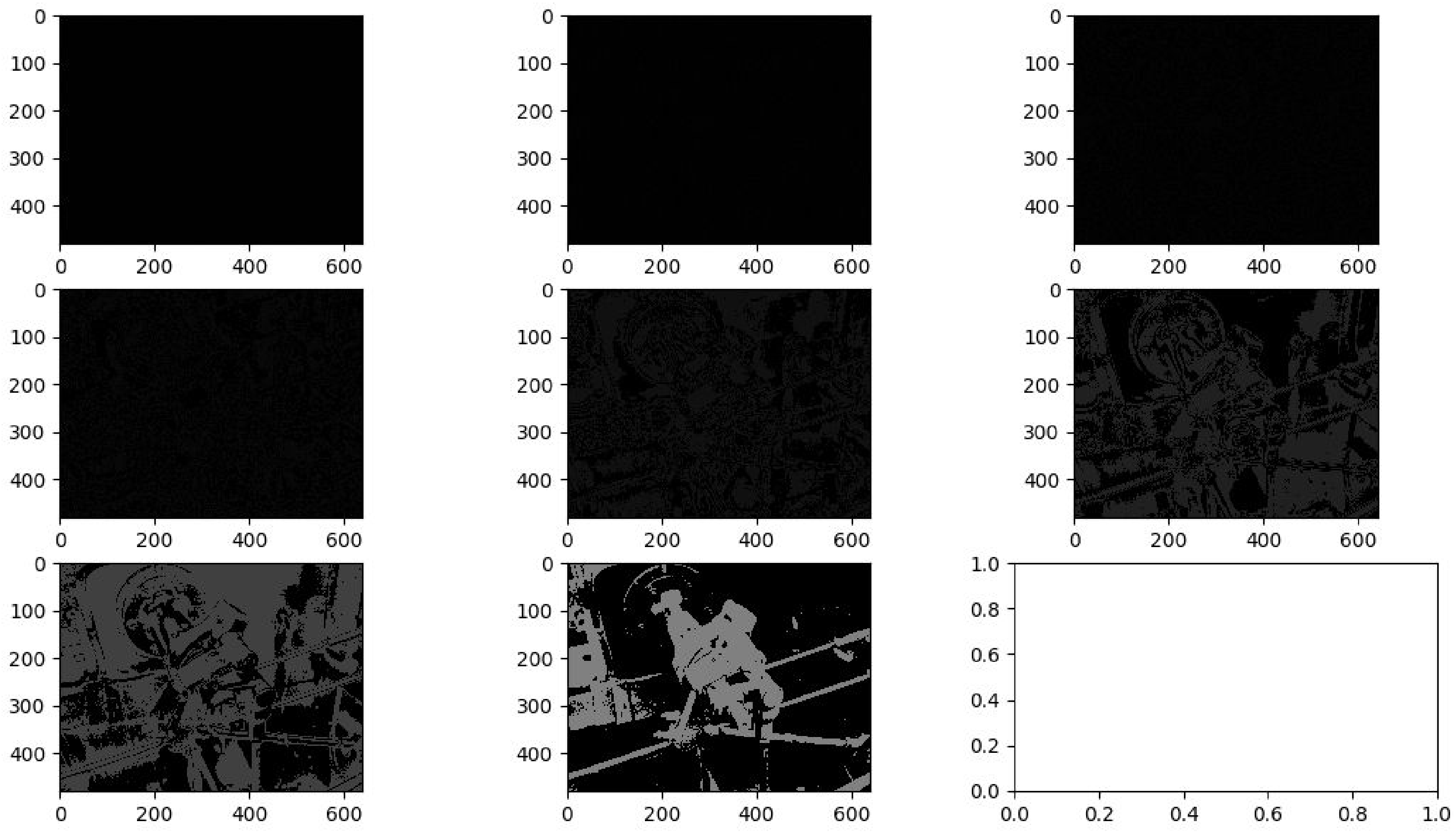
```
import cv2 as cv
import math


def contrast_stretching(img1, rmin, rmax, smin, smax, pix_max=255):
    img = img1.copy()
    l = smin / rmin
    m = (smax - smin) / (rmax - rmin)
    n = (pix_max - 1 - smax) / (pix_max - 1 - rmax)
    u, v = img.shape
    for i in range(u):
        for j in range(v):
            val = img[i][j]
            if val <= rmin:
                img[i][j] = int(l * val)
            elif val >= rmax:
                img[i][j] = int(n * (val - rmax) + smax)
            else:
                img[i][j] = int(m * (val - rmin) + smin)
    return img


def dynamic_range_compression(img1, c):
    img = img1.copy()
    m, n = img.shape
    for i in range(m):
        for j in range(n):
            img[i][j] = c * math.log(img[i][j] + 1)
    return img


def bit_plane_slicing(img, n_bits=8):
    return [img & (1 << nb) for nb in range(n_bits)]


def display_img(title, img, delay=1000):
    cv.imshow(title, img)
    cv.waitKey(delay)


if __name__ == '__main__':
    img = cv.imread('img.png', 0)
    stretched_img = contrast_stretching(img, 30, 200, 80, 210)
    comressed = dynamic_range_compression(img, 10)
    plane_sliced_imgs = bit_plane_slicing(img)


    display_img('og image', img)
    display_img('stretched image', stretched_img)
    cv.waitKey(10000)
    display_img('dynamic range compressed', comressed)
    [display_img('plane' + str(1 << nb), photo) for nb, photo in
enumerate(plane_sliced_imgs)]
```



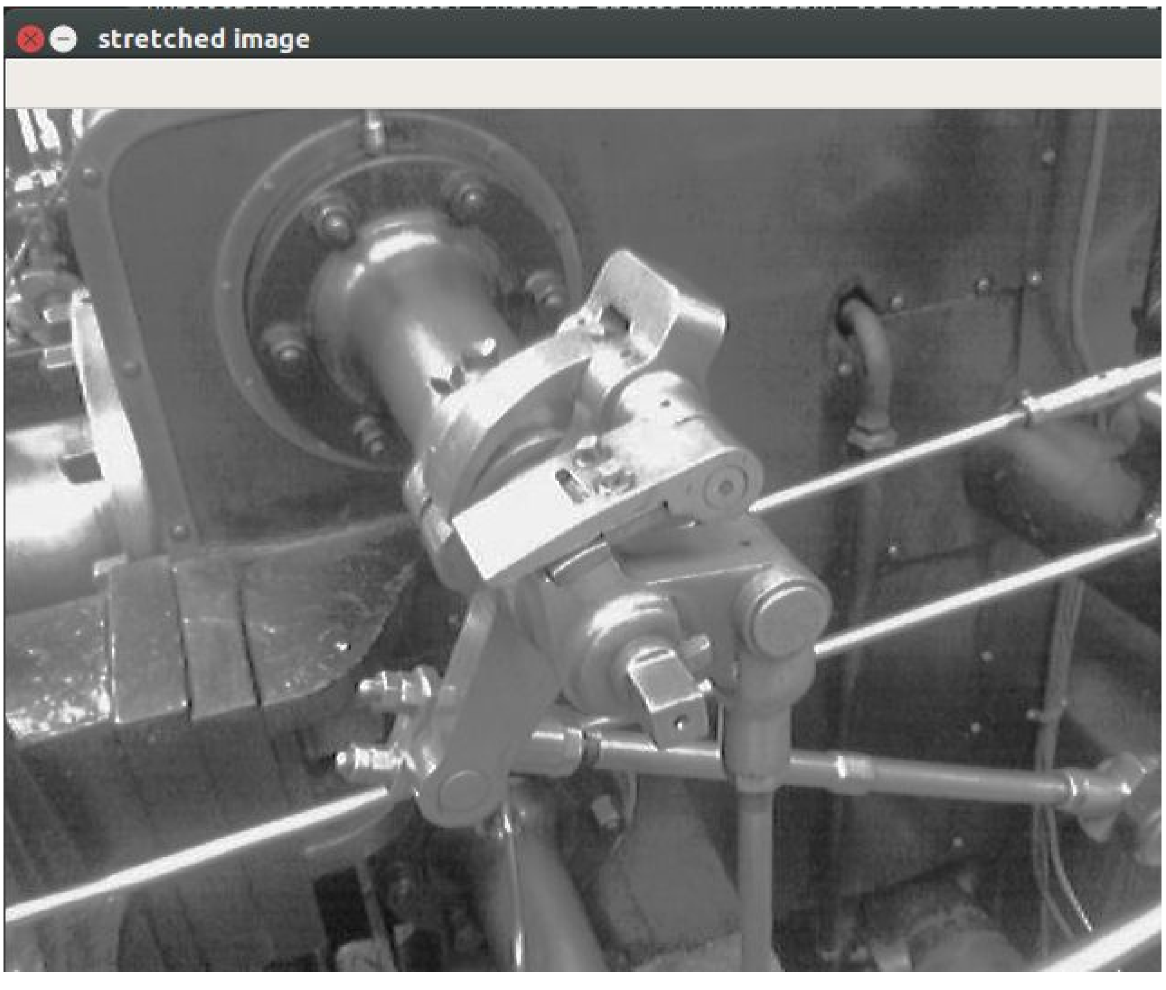
Original Image



Dynamic Range Compressed Image



Bit Plane Sliced Array of Images



Contrast Stretched Image