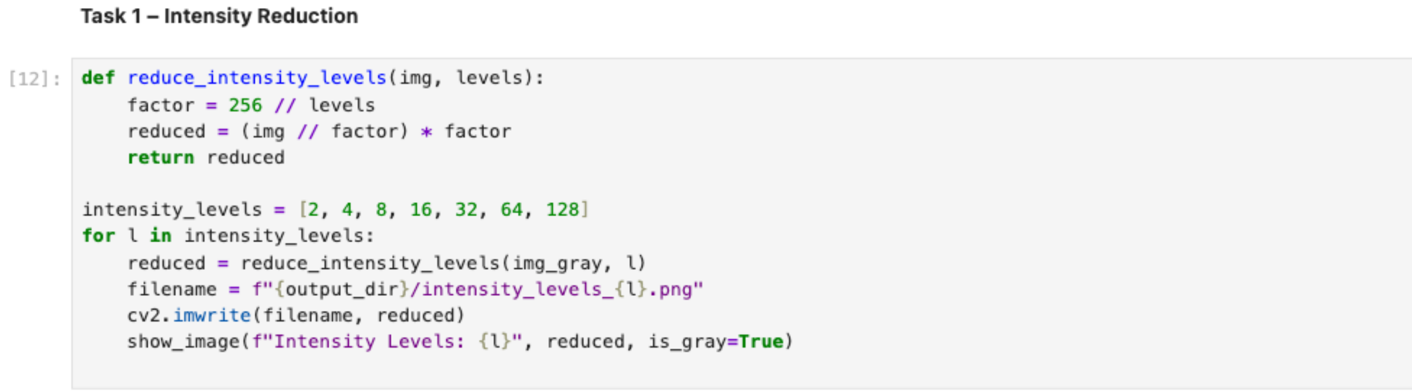
EC7212 – Computer Vision and Image Processing

Take Home Assignment 1

EG/2020/3817 – Amarasinghe A.U

GitHub Repo Link - https://github.com/Amayuru1999/EC7212-CVIP-Assignment-3817.git

1. To reduce the number of intensity levels in an image from 256 to 2, in integer powers of 2. The desired number of intensity levels needs to be a variable input to your program.



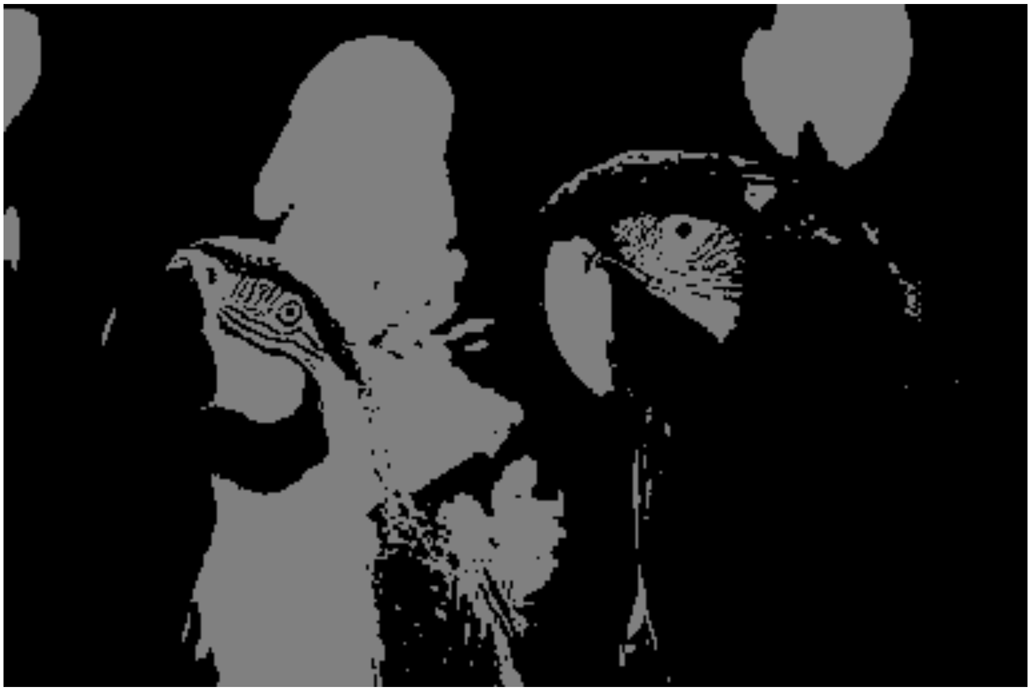
Task 1 – Intensity Reduction

[12]:

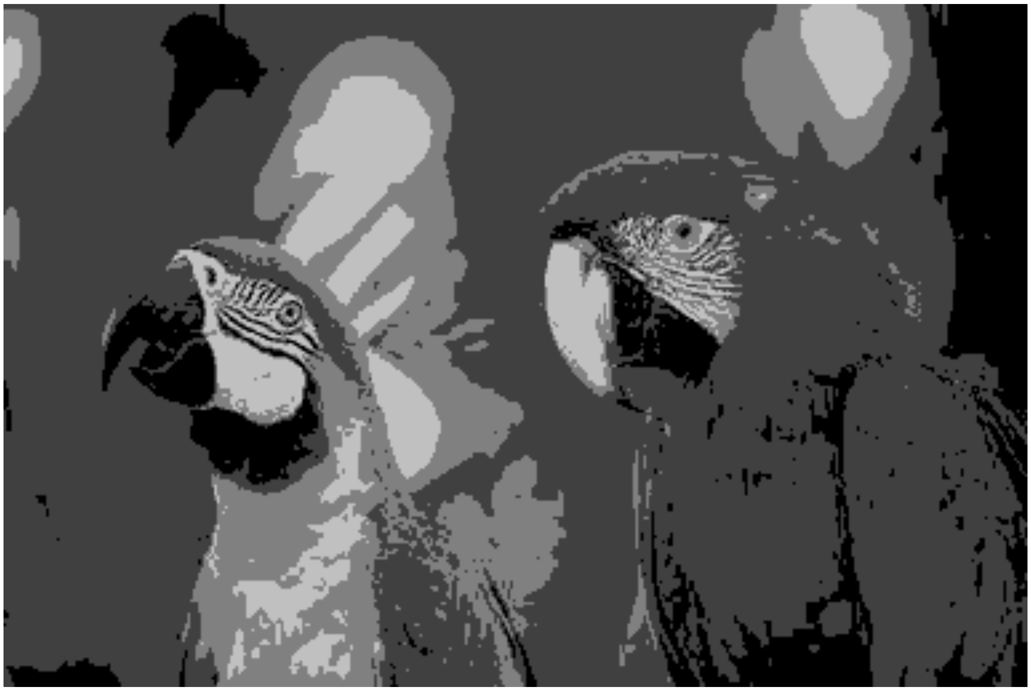
```
def reduce_intensity_levels(img, levels):
    factor = 256 // levels
    reduced = (img // factor) * factor
    return reduced

intensity_levels = [2, 4, 8, 16, 32, 64, 128]
for l in intensity_levels:
    reduced = reduce_intensity_levels(img_gray, l)
    filename = f"{output_dir}/intensity_levels_{l}.png"
    cv2.imwrite(filename, reduced)
    show_image(f"Intensity Levels: {l}", reduced, is_gray=True)
```

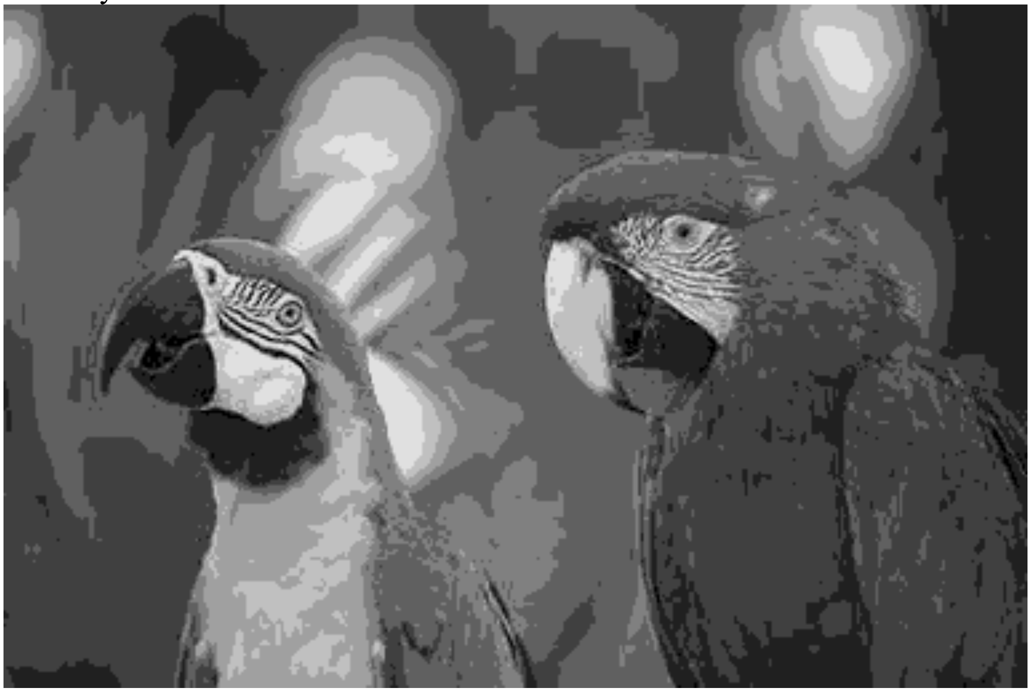
Intensity Levels: 2



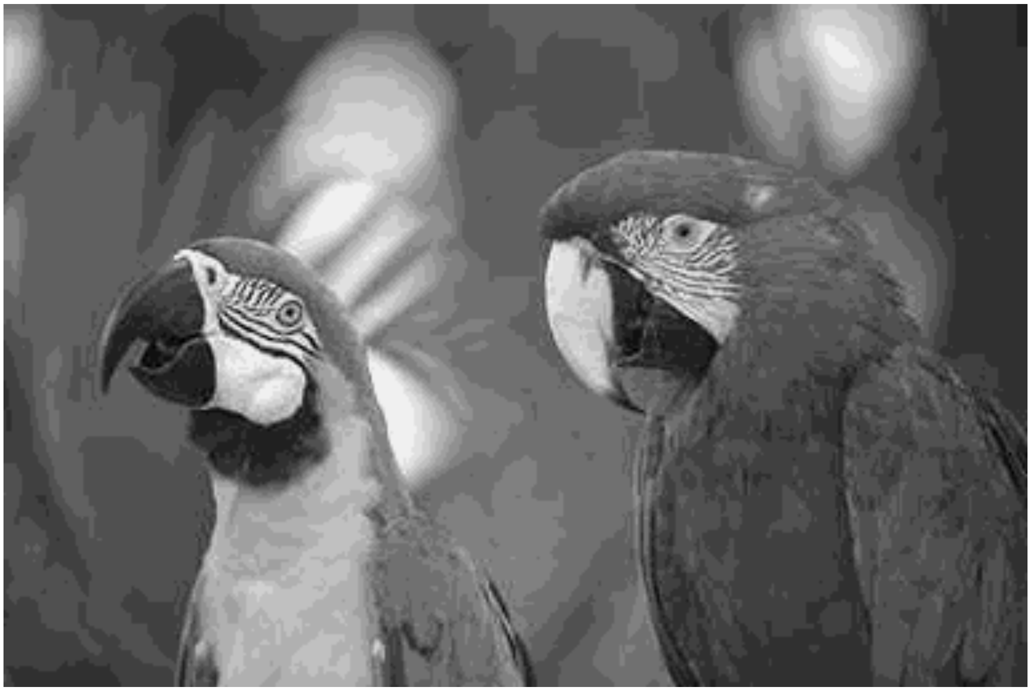
Intensity Levels: 4



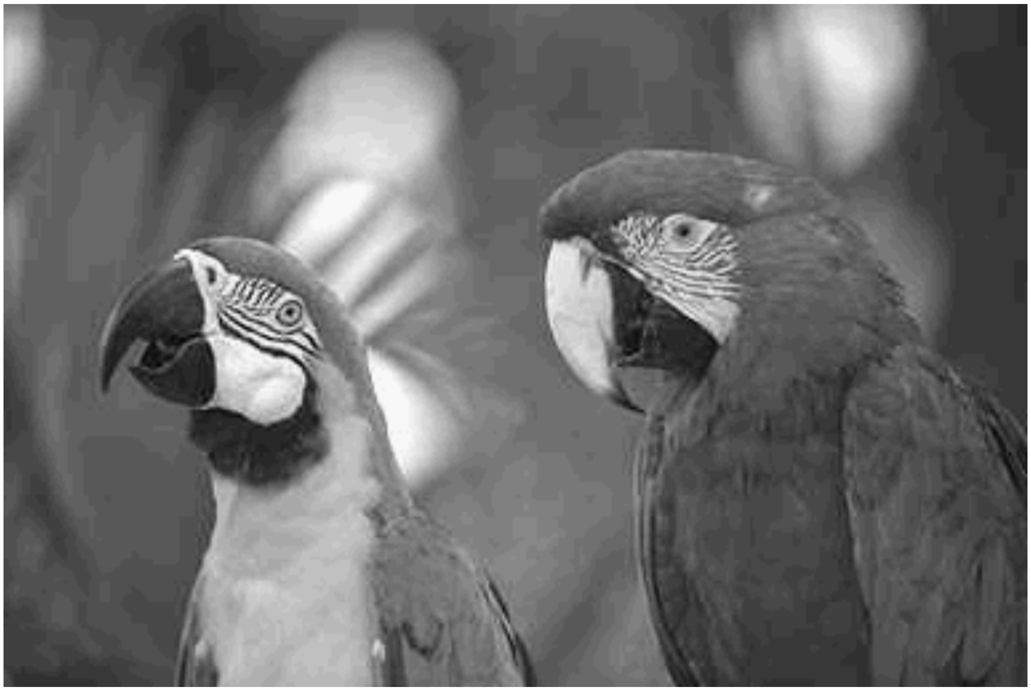
Intensity Levels: 8



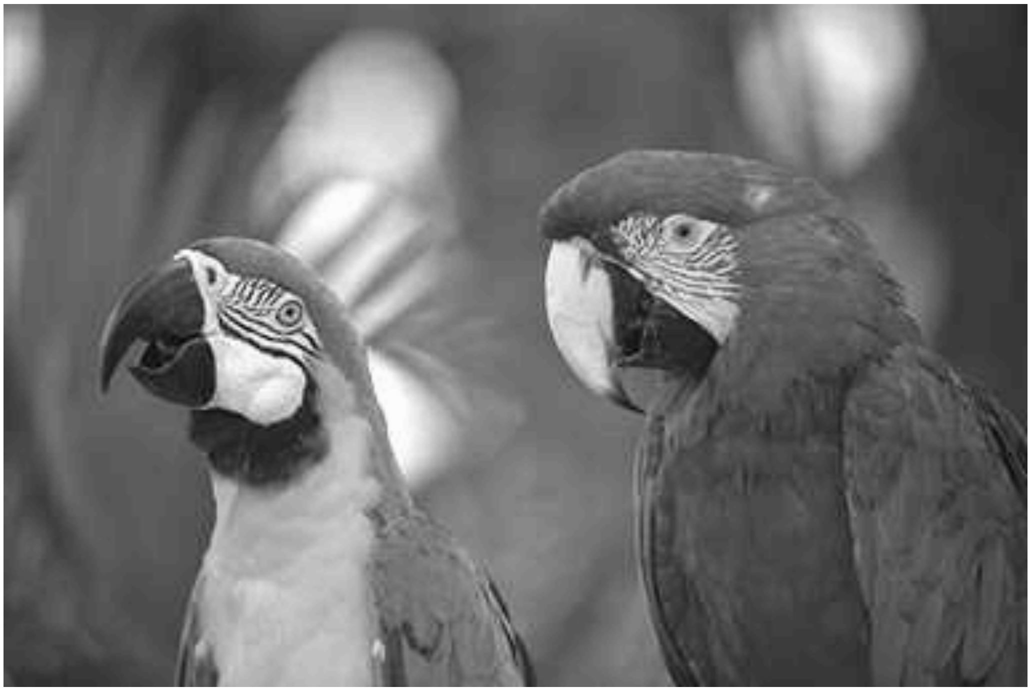
Intensity Levels: 16



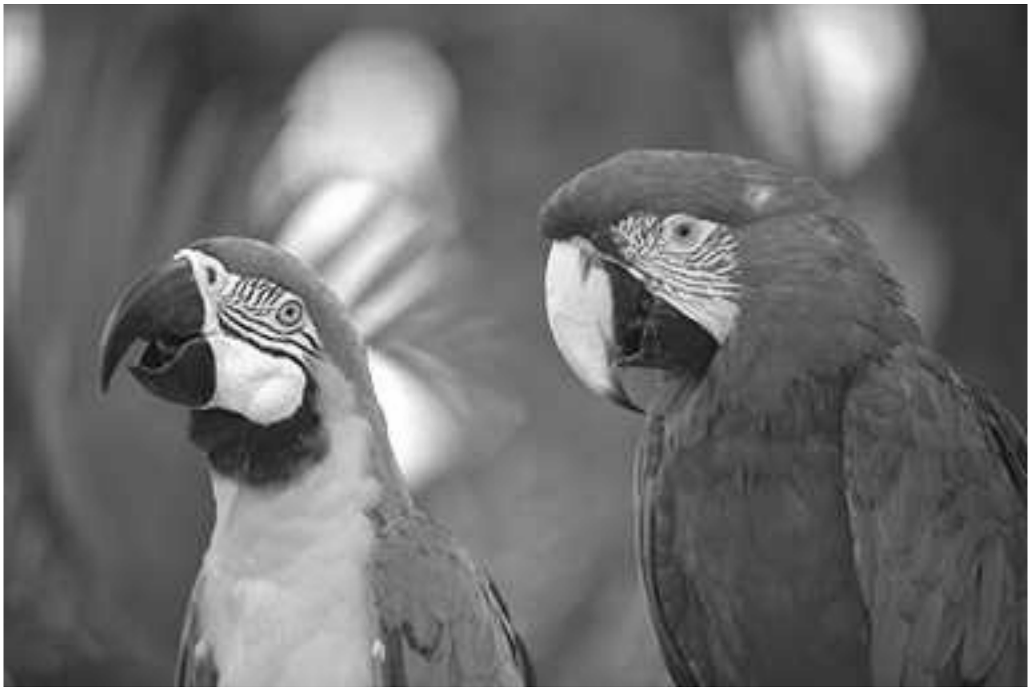
Intensity Levels: 32



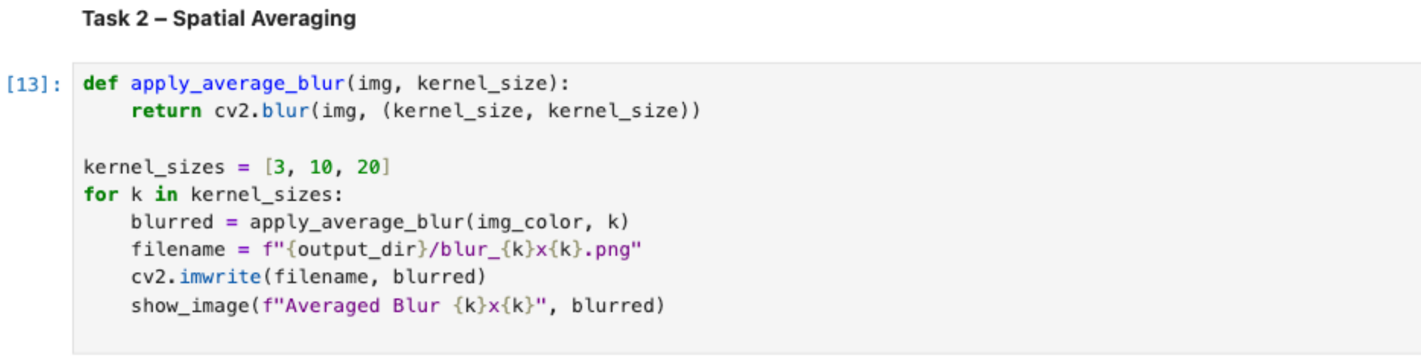
Intensity Levels: 64



Intensity Levels: 128



2. Load an image and then perform a simple spatial 3x3 average of image pixels. Repeat the process for a 10x10 neighborhood and again for a 20x20 neighborhood.



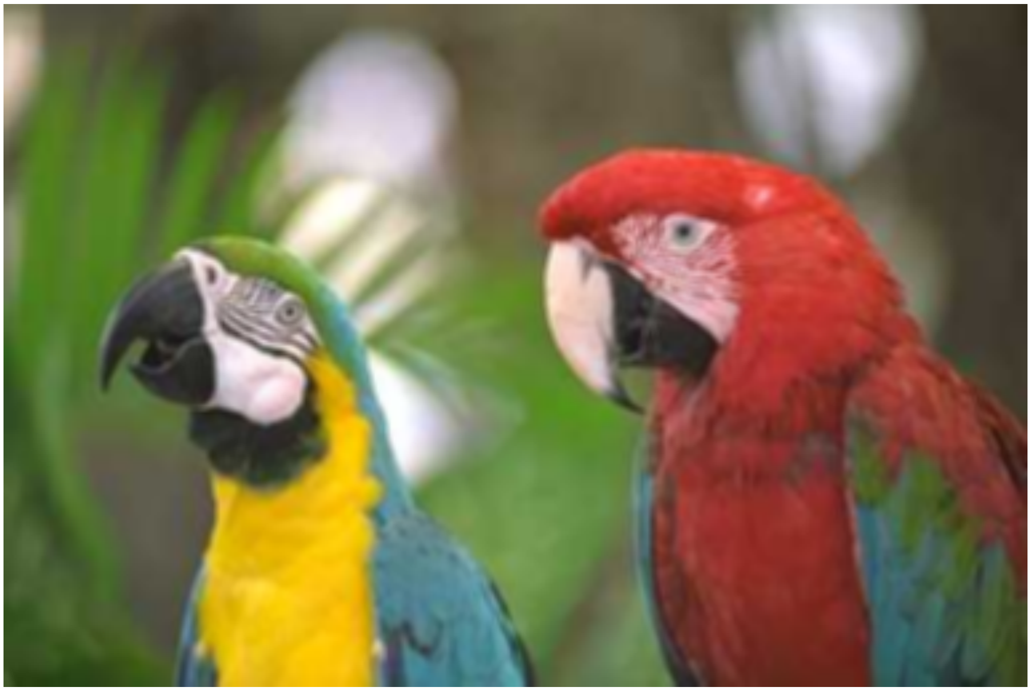
Task 2 – Spatial Averaging

[13]:

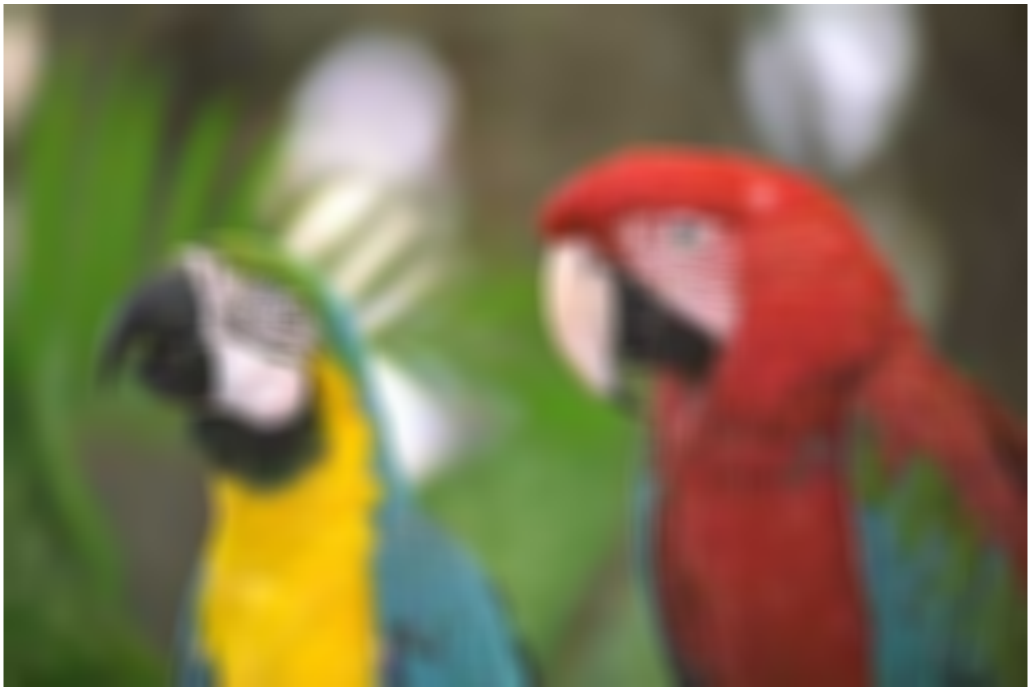
```
def apply_average_blur(img, kernel_size):
    return cv2.blur(img, (kernel_size, kernel_size))

kernel_sizes = [3, 10, 20]
for k in kernel_sizes:
    blurred = apply_average_blur(img_color, k)
    filename = f"{output_dir}/blur_{k}x{k}.png"
    cv2.imwrite(filename, blurred)
    show_image(f"Averaged Blur {k}x{k}", blurred)
```

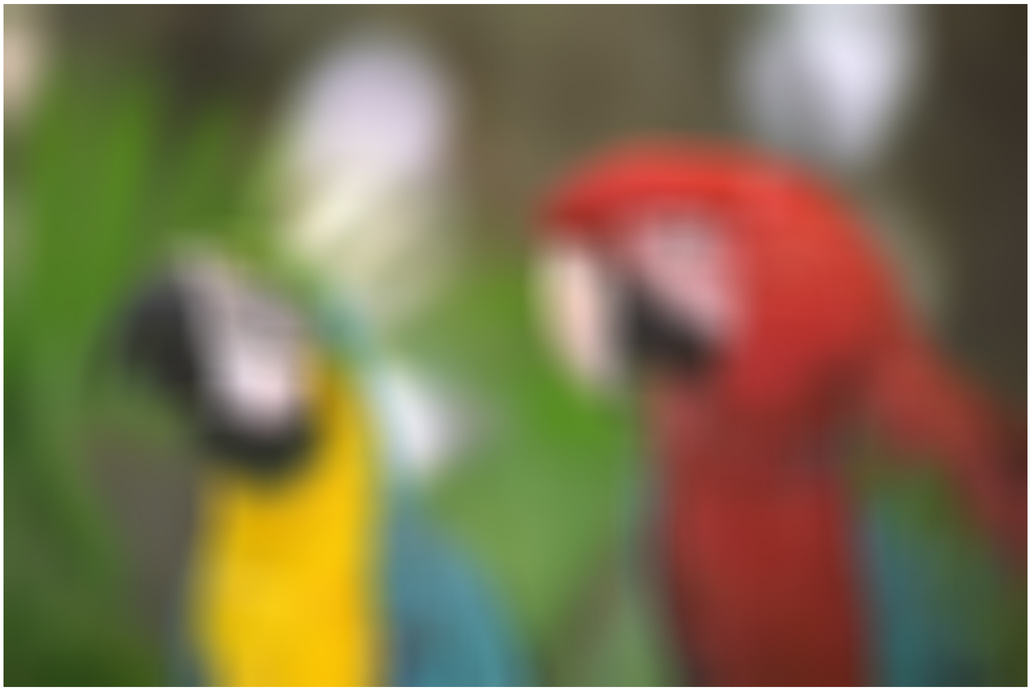
Averaged Blur 3x3



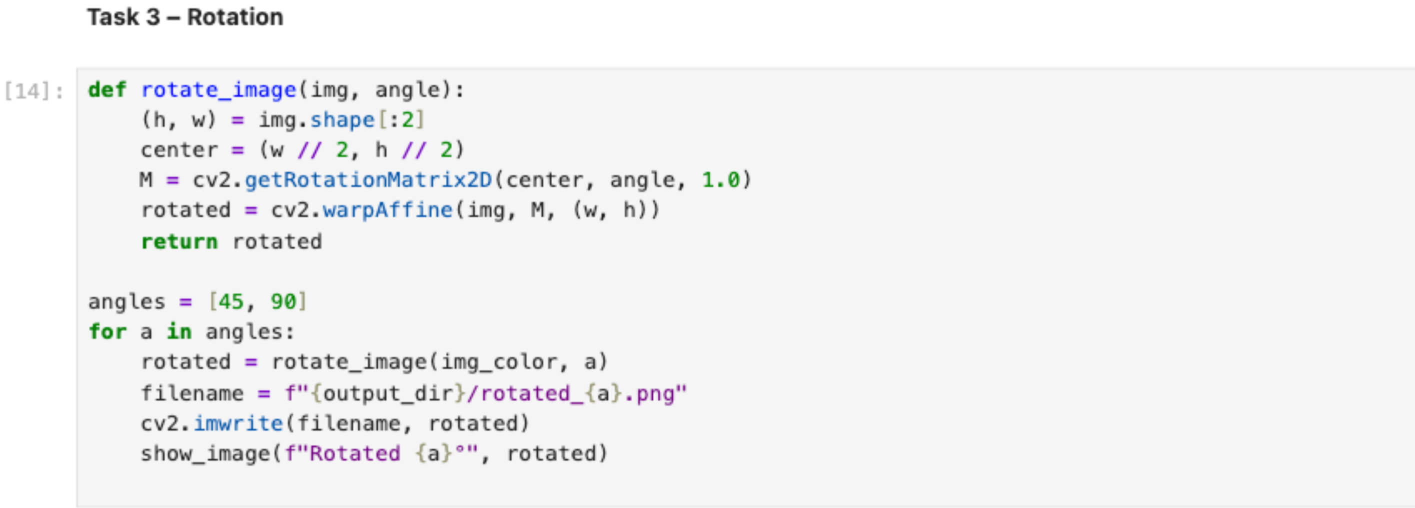
Averaged Blur 10x10



Averaged Blur 20x20



3. Rotate an image by 45 and 90 degrees.



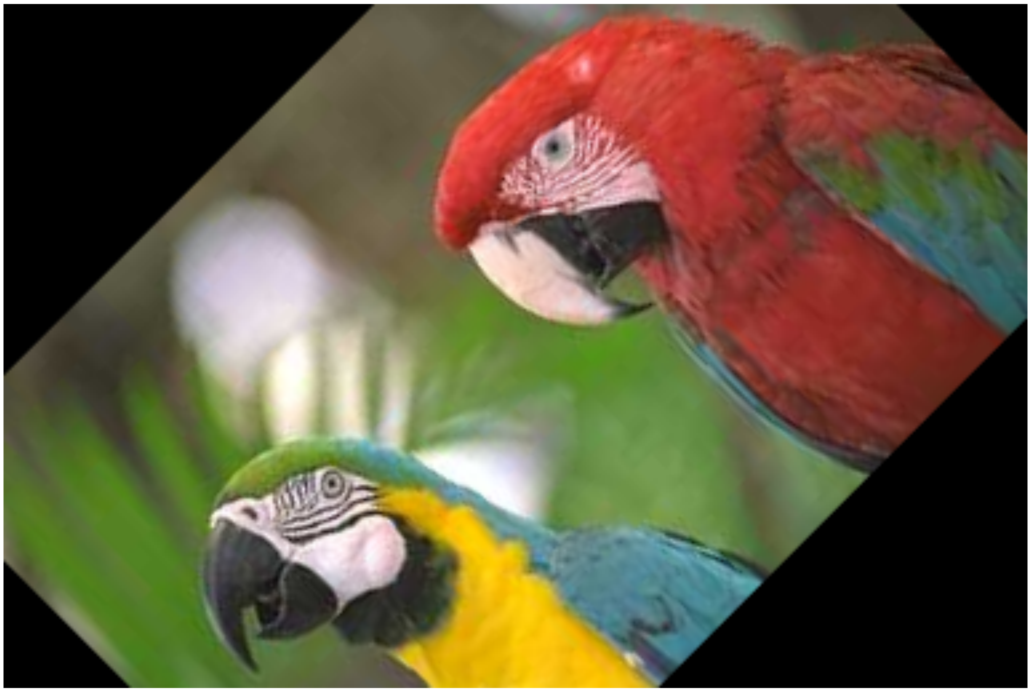
Task 3 – Rotation

[14]:

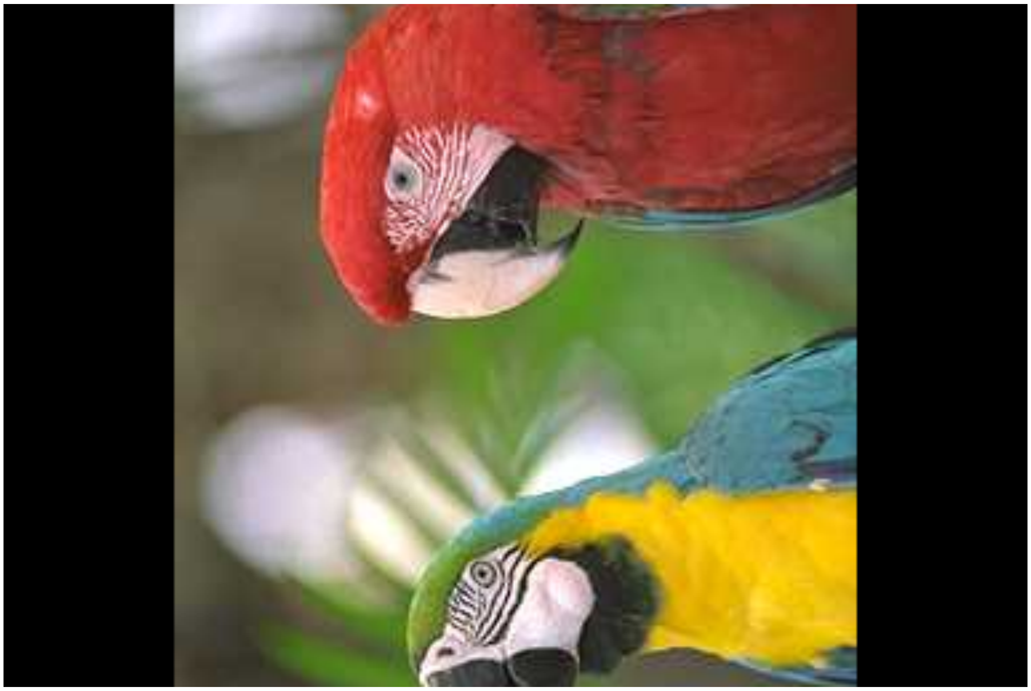
```
def rotate_image(img, angle):
    (h, w) = img.shape[:2]
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, angle, 1.0)
    rotated = cv2.warpAffine(img, M, (w, h))
    return rotated

angles = [45, 90]
for a in angles:
    rotated = rotate_image(img_color, a)
    filename = f"{output_dir}/rotated_{a}.png"
    cv2.imwrite(filename, rotated)
    show_image(f"Rotated {a}°", rotated)
```

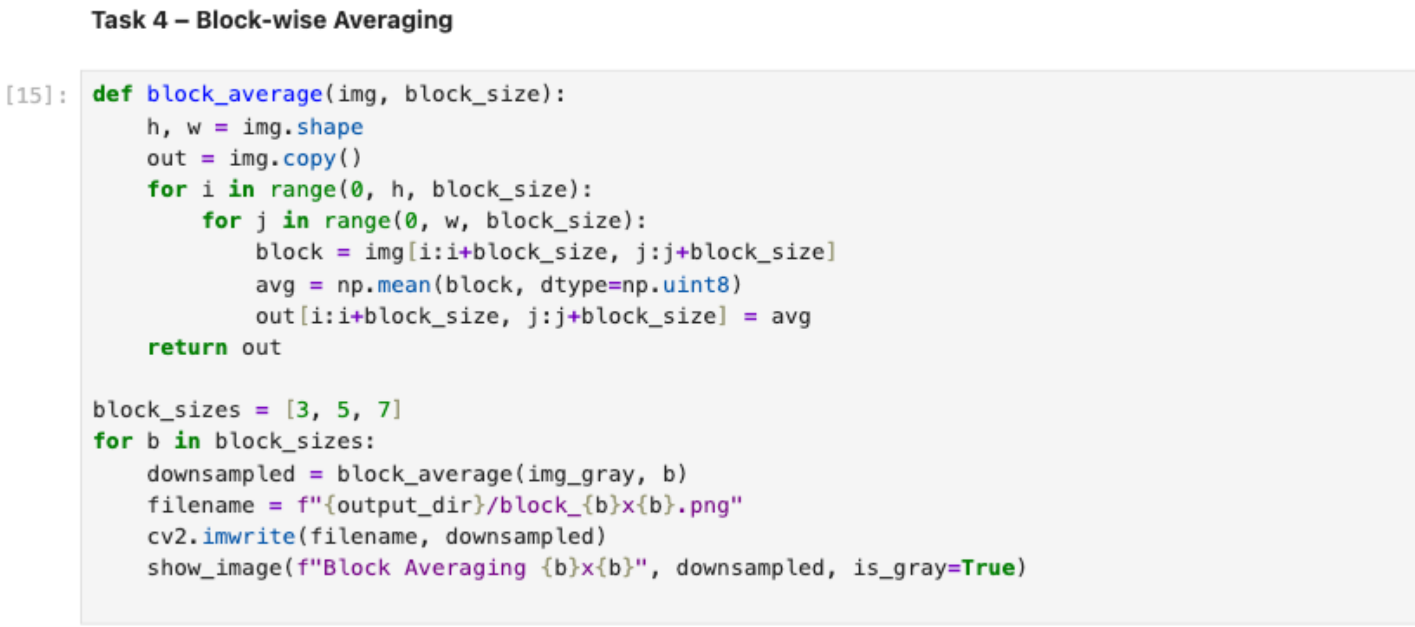
Rotate 45°



Rotate 90°



4. For every 3×3 block of the image (without overlapping), replace all the corresponding 9 pixels by their average. This operation simulates reducing the image spatial resolution. Repeat this for 5×5 blocks and 7×7 blocks.



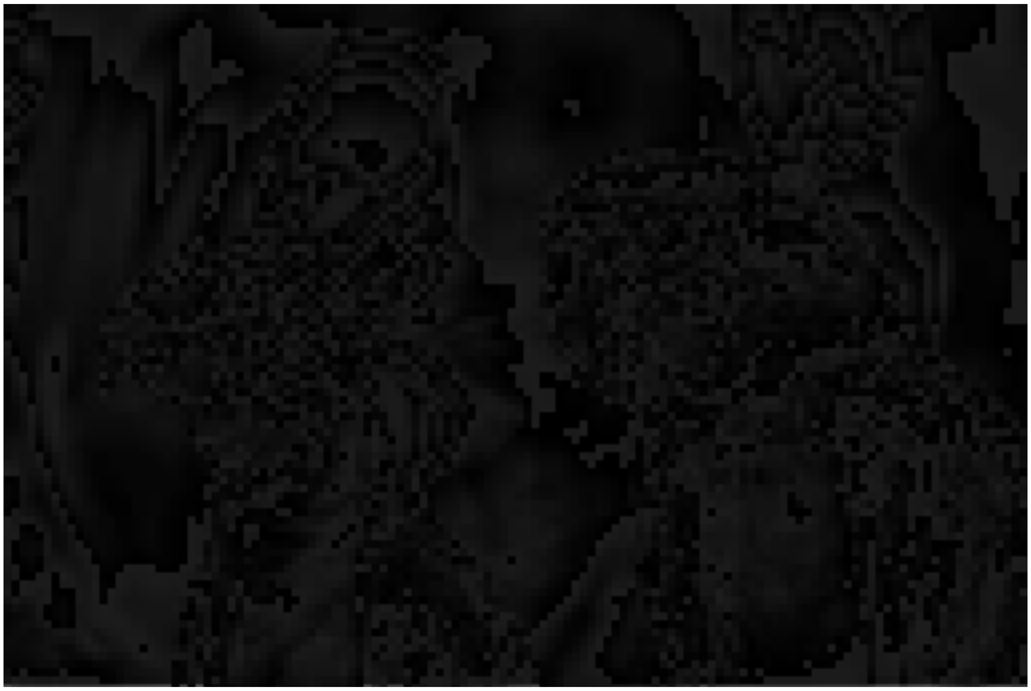
Task 4 – Block-wise Averaging

[15]:

```
def block_average(img, block_size):
    h, w = img.shape
    out = img.copy()
    for i in range(0, h, block_size):
        for j in range(0, w, block_size):
            block = img[i:i+block_size, j:j+block_size]
            avg = np.mean(block, dtype=np.uint8)
            out[i:i+block_size, j:j+block_size] = avg
    return out

block_sizes = [3, 5, 7]
for b in block_sizes:
    downsampled = block_average(img_gray, b)
    filename = f"{output_dir}/block_{b}x{b}.png"
    cv2.imwrite(filename, downsampled)
    show_image(f"Block Averaging {b}x{b}", downsampled, is_gray=True)
```

Block Averaging 3×3



Block Averaging 5×5

Block Averaging 7×7