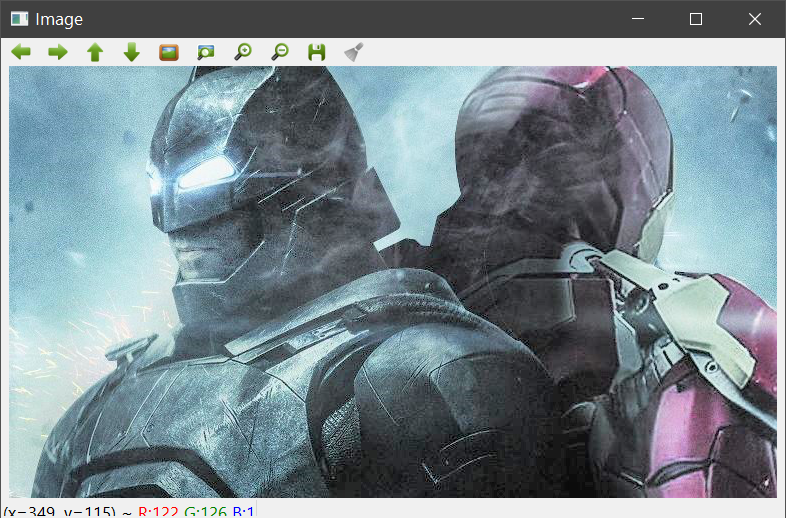
CARTOONIZE AN IMAGE

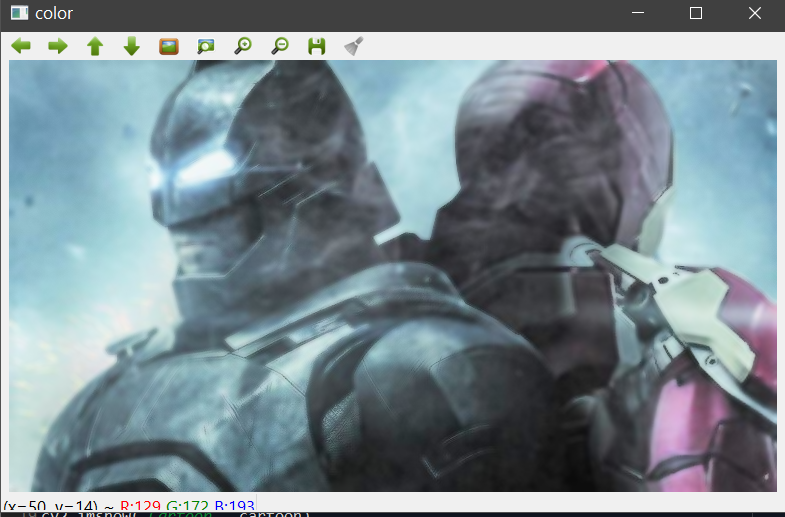
Using OpenCV and Python, an RGB color image can be converted into a cartoon in 5 steps:

1. Apply a [bilateral filter](https://en.wikipedia.org/wiki/Bilateral_filter) to reduce the color palette of the image.
2. Convert the original color image to grayscale.
3. Apply a [median blur](https://en.wikipedia.org/wiki/Median_filter) to reduce image noise in the grayscale image.
4. Create an edge mask from the grayscale image using [adaptive thresholding](https://en.wikipedia.org/wiki/Thresholding_(image_processing)).
5. Combine the color image from step 1 with the edge mask from step 4.



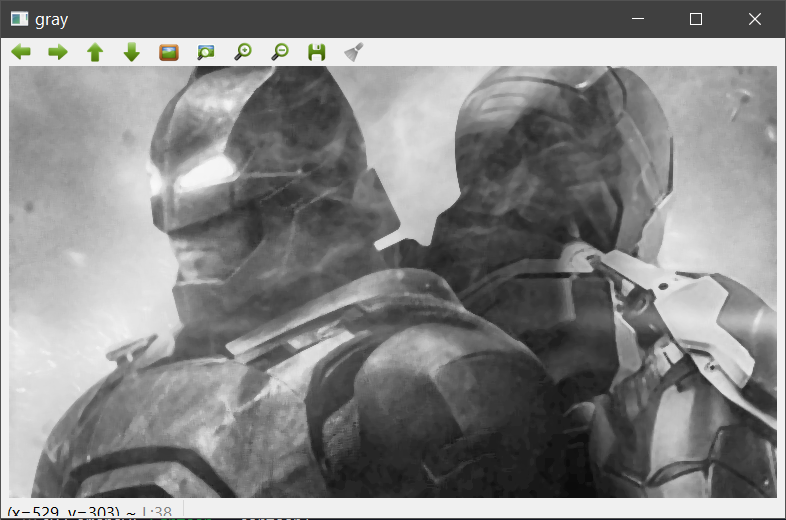
**Step 1: Edge-aware smoothing using a bilateral filter**

Because a bilateral filter smooths flat regions while keeping edges sharp, it is ideally suited to convert an RGB image into a cartoon. Unfortunately, bilateral filters are orders of magnitudes slower than other smoothing operators (e.g., Gaussian blur). Thus, if speed is important, it might be a good idea to operate on a down-scaled version of the original image.



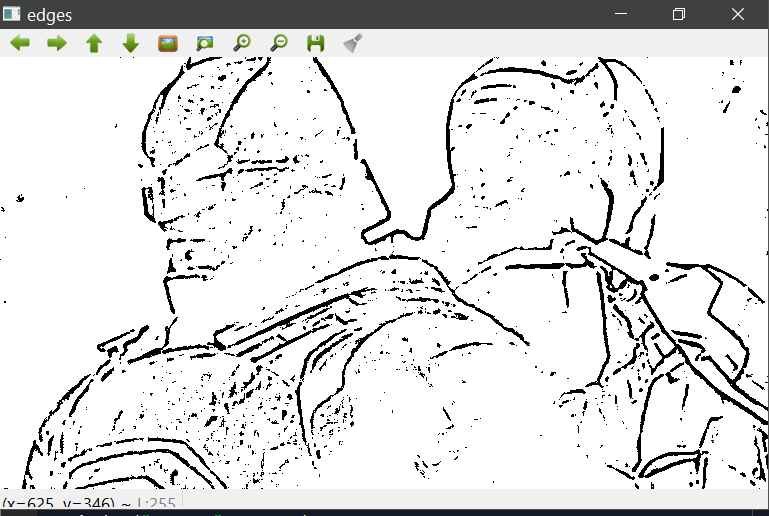
**Steps 2-3: Reduce noise using a median filter**

OpenCV offers a variety of choices when it comes to edge detection. The beauty of adaptive thresholding is that it detects the most salient features in each (small) neighborhood of an image, independent of the overall properties of the image, which is exactly what we want when we seek to draw bold, black outlines around objects and people in a cartoon. However, this property also makes adaptive thresholding susceptible to noise. It is therefor a good idea to pre-process the image with a median filter, which replaces each pixel value with the median value of all the pixels in a small (e.g., 5 pixel) neighborhood:



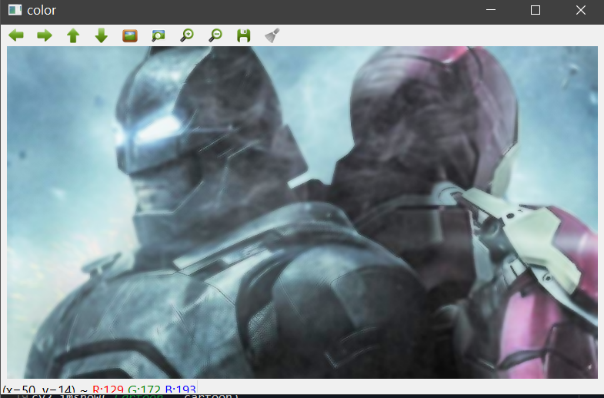
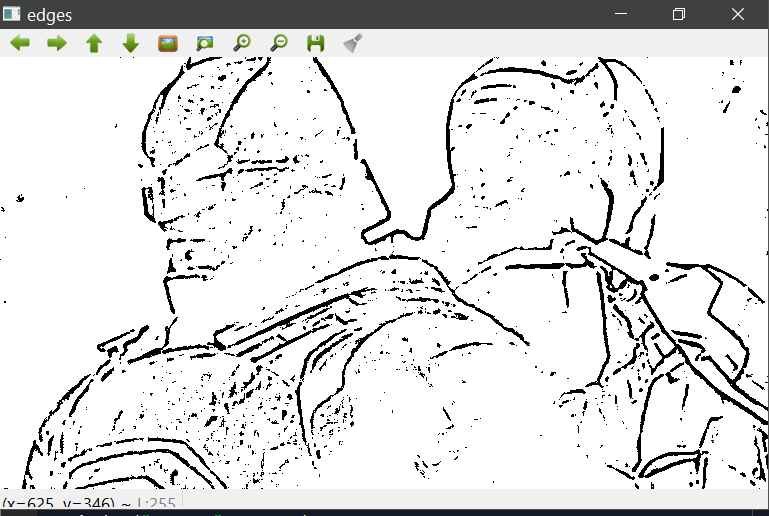
**Step 4: Create an edge mask using adaptive thresholding**

After noise reduction it is safe to apply adaptive thresholding. Even if there is some image noise left, the cv2.ADAPTIVE\_THRESH\_MEAN\_C algorithm with blockSize=9 will ensure that the threshold is applied to the mean of a 9x9 neighborhood minus C=2:



**Step 5: Combine color image with edge mask**

The last step is to combine the processed color image (color) with the edge mask (edges):

 **+** 

* 