SIFT:

Motivation: Object recognition is one the hottest area in computer vision. It is fundamental to facial recognition, object tracking in videos, object matching and many more applications. To achieve object recognition(from images), one first must be able to extract distinct features of the image so that the image(or its segments) could be compared to a set of references. Moreover, such features should be invariant to changes in scale, rotation and translation of the image and must be resistant to noise. SIFT(Scale Invariant Feature Transform) is one of the approaches to obtain such features.

Approach:

According to Prof. Dowe’s paper on Distinctive Image Features from Scale-Invariant Keypoints(2004), there are four stages to SIFT:

1. “Scale-space extrema detection”

Searches the entire image for candidate interest points

1. “Keypoint localization”

Calculate the location and scale for each candidate, remove candidates that are not stable

1. “Orientation assignment”

Assign each key point with one or more orientations that are calculated based on the gradient direction at that key point location in the image

1. “Keypoint descriptor”

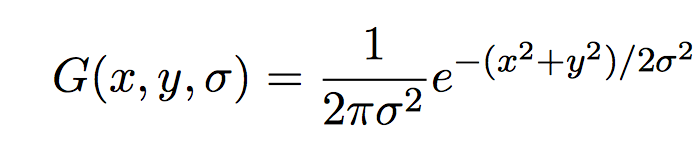
For each key point, calculate the gradient in its surrounding area. This allows the transform to be distortion resistant.

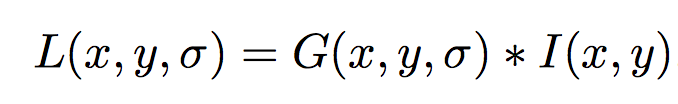
The above approach “transforms image data into scale-invariant coordinates relative to local features”.

Our project seeks to achieve scale, rotation and translation resistance, due to time-constraint, we would not be implementing stage 4 “Keypoint descriptor” of SIFT.

Implementation:

Stage one:

1. Apply Gaussian filters of different scales to the image. By using different scales the Gaussian filters would have different variances. Due to the inherent properties of Gaussian filters, this would “smooth” out the images, removing finer details of the image. At different scales, the details of the image that are insignificant compared to the standard deviation of the Gaussian filter applied would be removed. The Gaussians are generated using the following formula:

and the image, represented as an array of digits, is convolved with the Gaussian.

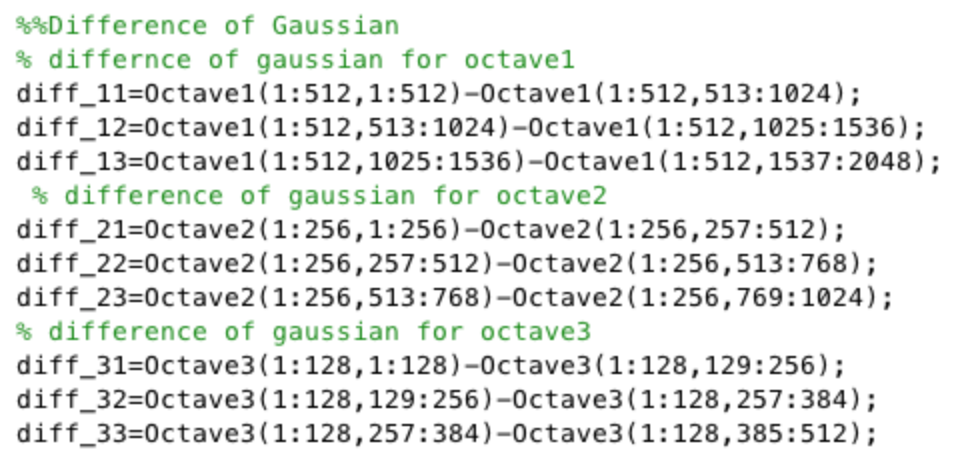
where L(x,y,sigma) standards for the value of the resulting image at location (x,y) under the Gaussian filter with standard deviation sigma. I stands for the original image.

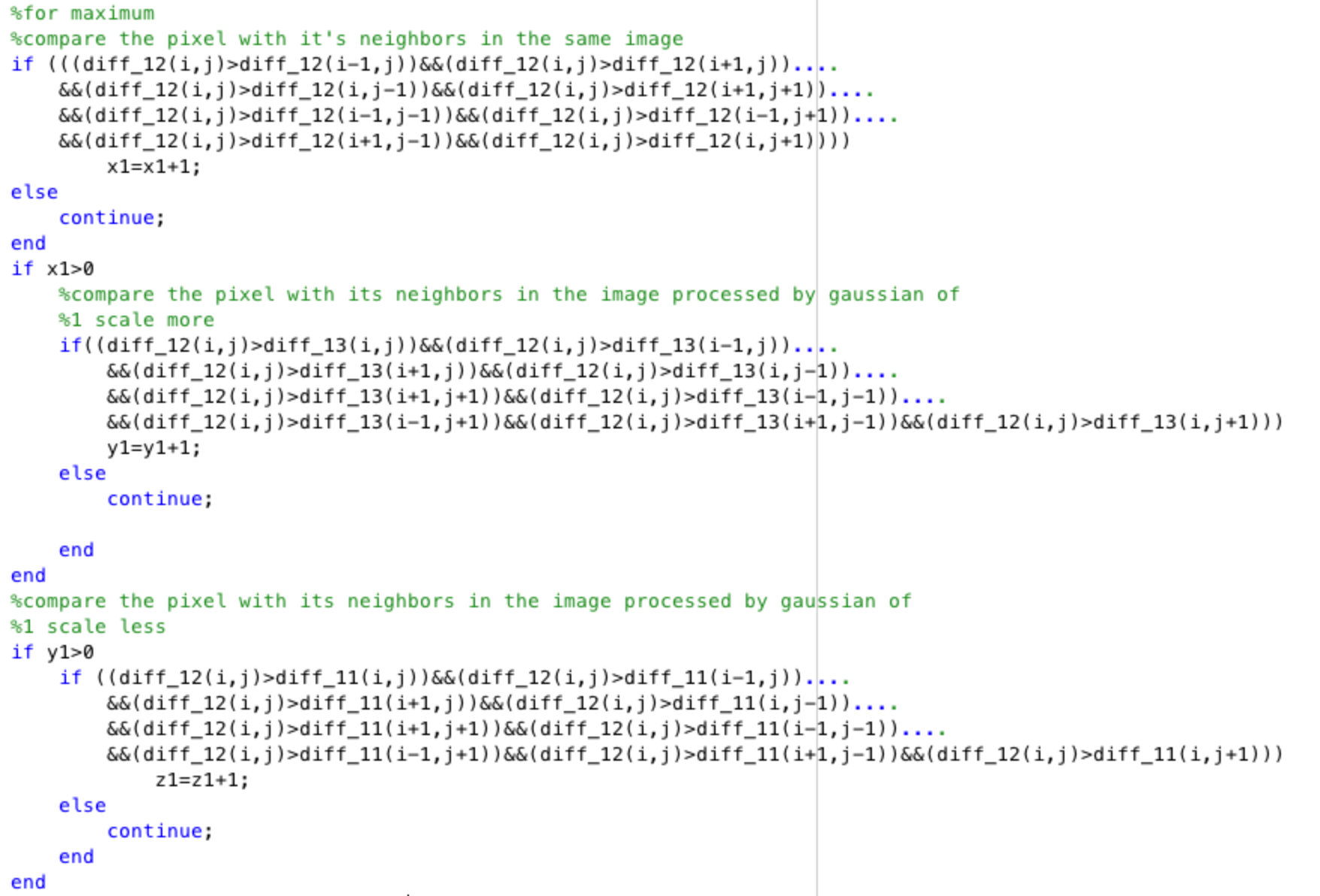
We applied Gaussians with scale 0, 1, and 2 to the image. At scale 0, we are essentially preserving the original image, at scales 1 and 2 we are “smoothing out” the image to an increasing extend. We have 3 octaves of resulting images, each octave consists of images resulting from repeated applying the gaussian filter of the same scale to the original image. After each octave, the image is down-sampled by two.



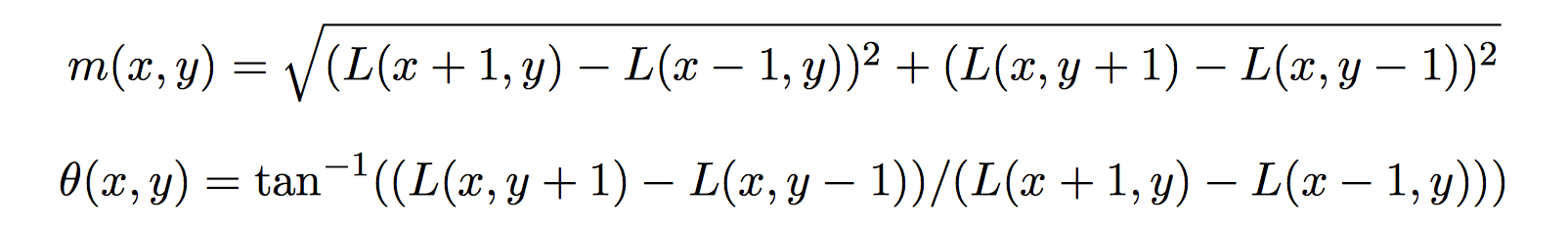
Stage two:

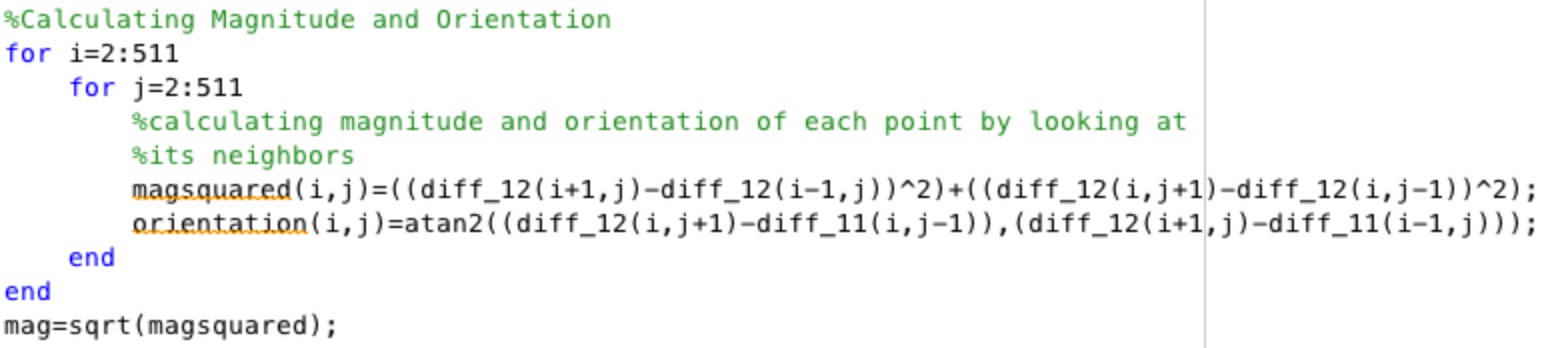
Now we have the image smoothed to different extends, with variant amount of fine detailed preserved in the resulting images. Within each octave, we use Difference of Gaussian, which is basically subtracting neighboring images from each other. Difference of Gaussian is proven to be a close approximation of scale-normalized Laplacian of Gaussian, which is shown to give “ produce the most stable image features compared to a range of other possible image functions, such as the gradient, Hessian, or Harris corner function”. Moreover, Difference of Gaussian is efficient to compute since it’s just subtracting images.

Then for each pixel in a resulting image, we compare it to its eight neighboring pixels in the same image and nine neighboring pixels in the images processed by adjacent scales. It’s selected if it’s greater or smaller than all its neighbors. The result is a candidate key point.



Stage three:

To calculate the magnitude and orientation of each key point, we look at all it’s neighboring pixels in the image that is processed with the same scale.

m(x,y) stands for the gradient magnitude of the point and theta(x,y) stands for the orientation of the point.

Next we would create 36 bins coving 360 degrees of angle, a key point is added into the bin if it’s orientation is within the angles covered by that bin. Then for each point we would take the maximum of all bins for the sum of magnitude of that bin and that bin would be the orientation of the point.

