(/) / Tutorials (/tutorials/) / Computer Vision (/category/computer-vision/) / SIFT: Getting rid of low contrast keypoints (/tutorials/sift-scale-invariant-feature-transform-eliminate-low-contrast/)

SIFT: Getting rid of low contrast keypoints

Key points generated in the previous step (/tutorials/sift-scale-invariant-feature-transform-keypoints/) produce a lot of key points. Some of them lie along an edge, or they don't have enough contrast. In both cases, they are not useful as features. So we get rid of them. The approach is similar to the one used in the Harris Corner Detector (/tutorials/harris-corner-detector/) for removing edge features. For low contrast features, we simply check their intensities.

Removing low contrast features

This is simple. If the magnitude of the intensity (i.e., without sign) at the current pixel in the DoG image (that is being checked for minima/maxima) is less than a certain value, it is rejected.

Because we have subpixel keypoints (we used the Taylor expansion to refine keypoints), we again need to use the taylor expansion to get the intensity value at subpixel locations. If it's magnitude is less than a certain value, we reject the keypoint.

Removing edges

The idea is to calculate two gradients at the keypoint. Both perpendicular to each other. Based on the image around the keypoint, three possibilities exist. The image around the keypoint can be:

```
* **A flat region:** If this is the case, both gradients will be small.
* **An edge:** Here, one gradient will be big (perpendicular to the edge) and the other will be small (a
long the edge)
* **A "corner":** Here, both gradients will be big.
```

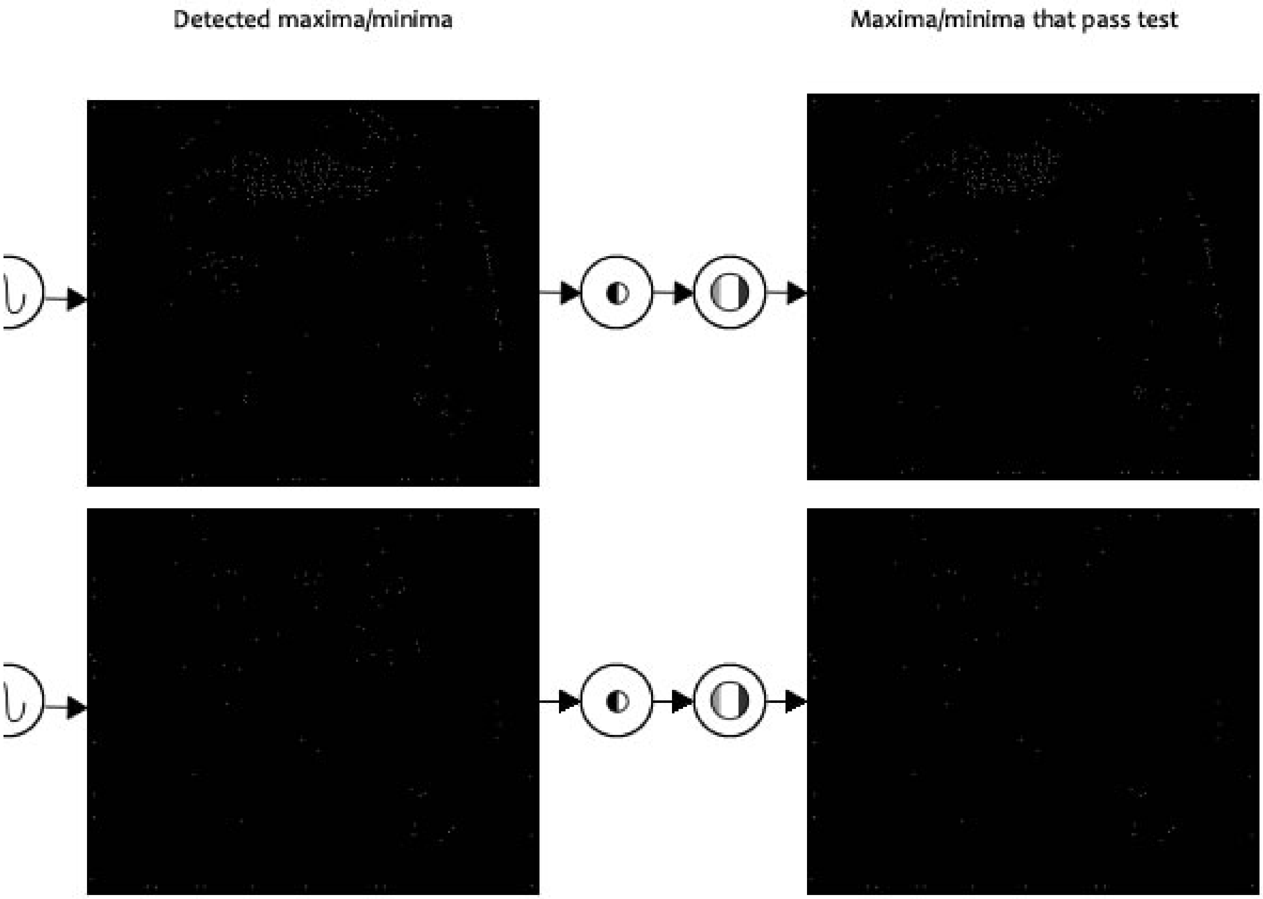
Corners are great keypoints. So we want just corners. If both gradients are big enough, we let it pass as a key point. Otherwise, it is rejected.

Mathematically, this is achieved by the Hessian Matrix. Using this matrix, you can easily check if a point is a corner or not.

If you're interested in the math, first check the posts on the Harris corner detector (/tutorials/harris-corner-detector/). A lot of the same math used used in SIFT. In the Harris Corner Detector, two eigenvalues are calculated. In SIFT, efficiency is increased by just calculating the ratio of these two eigenvalues. You never need to calculate the actual eigenvalues.

Example

Here's a visual example of what happens in this step:



Both extrema images go through the two tests: the contrast test and the edge test. They reject a few keypoints (sometimes a lot) and thus, we're left with a lower number of keypoints to deal with.

Summary

In this step, the number of keypoints was reduced. This helps increase efficiency and also the robustness of the algorithm. Keypoints are rejected if they had a low contrast or if they were located on an edge.

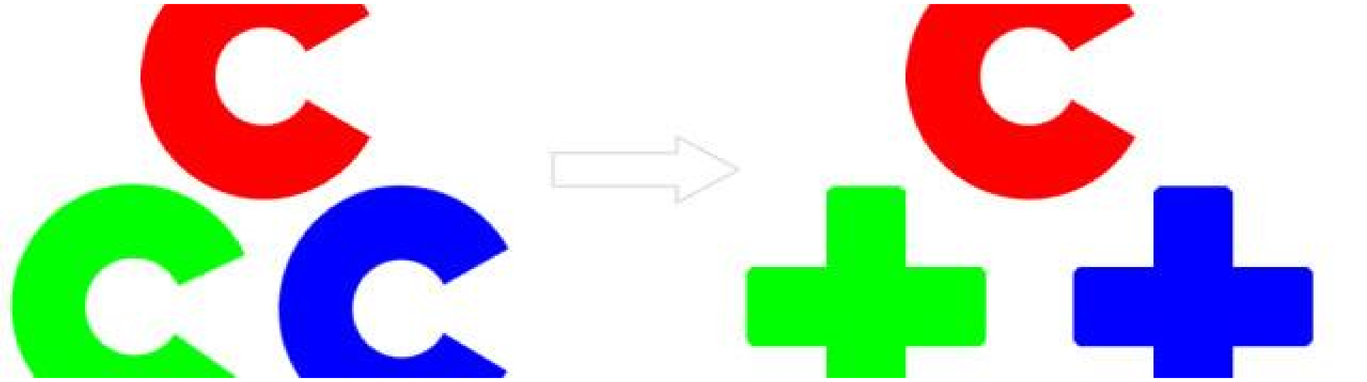
In the next step we'll assign an orientation to all the keypoints that passed both tests.

This tutorial is part of a series called **SIFT: Theory and Practice**:

1. SIFT: Introduction (/tutorials/sift-scale-invariant-feature-transform-introduction/)
2. SIFT: The scale space (/tutorials/sift-scale-invariant-feature-transform-scale-space/)
3. SIFT: LoG approximations (/tutorials/sift-scale-invariant-feature-transform-log-approximation/)
4. SIFT: Finding keypoints (/tutorials/sift-scale-invariant-feature-transform-keypoints/)
5. **SIFT: Getting rid of low contrast keypoints**
6. SIFT: Keypoint orientations (/tutorials/sift-scale-invariant-feature-transform-keypoint-orientation/)
7. SIFT: Generating a feature (/tutorials/sift-scale-invariant-feature-transform-features/)

Related posts

Timing macros in C/C++ (/tutorials/timing-macros/)



OpenCV's C++ interface (/tutorials/opencvs-c-interface/)

Obstacle avoidance with the Bug-1 algorithm (/tutorials/obstacle-avoidance-with-the-bug-algorithm/)

(http://utkarshsinha.com/)

Utkarsh Sinha created AI Shack in 2010 and has since been working on computer vision and related fields as a hobby! Currently he works as a writer of software in Bangalore, India.

The bottom sidebar

AI Shack on Facebook

AI Shack

Like

2,756 people like AI Shack.

Facebook social plugin

Contact

You can contact me through the internet over the following channels:

Facebook (http://facebook.com/aishack/)
Twitter (http://twitter.com/aishack)
Github (http://github.com/aishack)
Email (mailto:utkarsh@utkarshsinha.com)

Where to start?

So you want to get started with AI but you're not sure where. Here are some links to get you started:

- Get started with OpenCV (/tracks/opencv-basics/)

- Track a specific color on video (/tutorials/tracking-colored-objects-in-opencv/)
- Learn basic image processing algorithms (/tracks/image-processing-algorithms-level-1/)
- How to build artificial neurons? (/tutorials/a-single-neuron-dictomizer/)
- Look at some source code (http://github.com/aishack/)

(/)

Created by Utkarsh Sinha (http://utkarshsinha.com/)