

Chris McCormick [About](#) [Tutorials](#) [Archive](#)

Gradient Vectors

07 May 2013

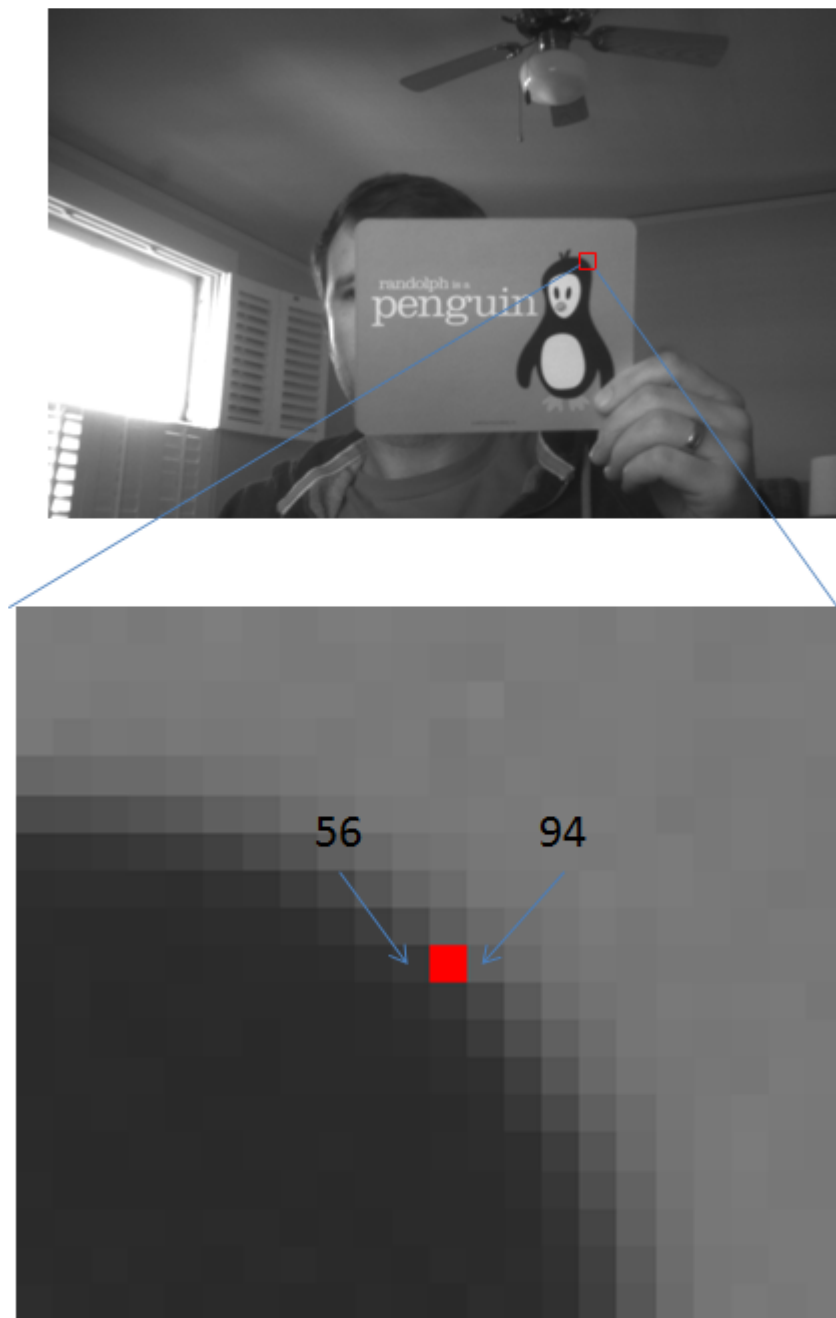
Gradient vectors (or “image gradients”) are one of the most fundamental concepts in computer vision; many vision algorithms involve computing gradient vectors for each pixel in an image.

After a quick introduction to how gradient vectors are computed, I’ll discuss some of its properties which make it so useful.

Computing The Gradient Image

A gradient vector can be computed for every pixel an image. It’s simply a measure of the change in pixel values along the x-direction and the y-direction around each pixel.

Let’s look at a simple example; let’s say we want to compute the gradient vector at the pixel highlighted in red below.

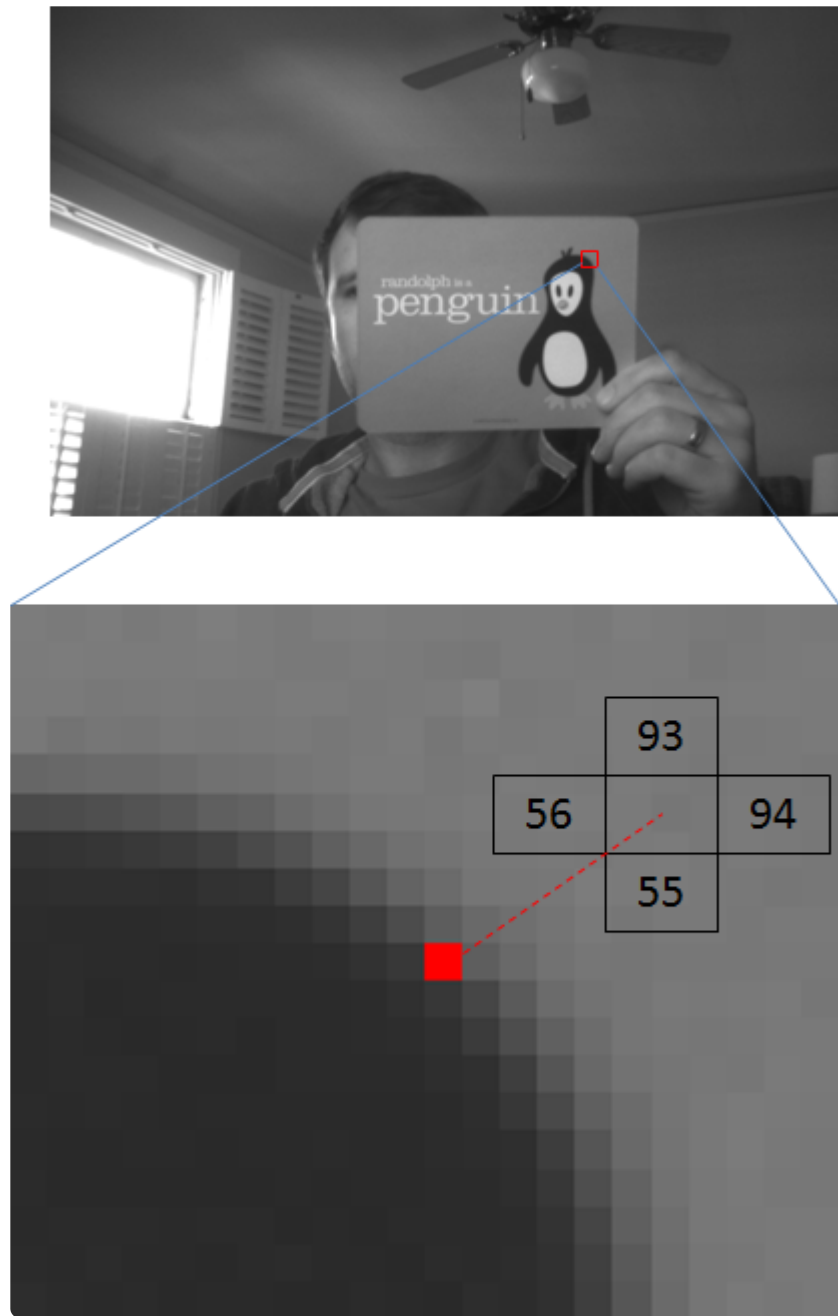


This is a grayscale image, so the pixel values just range from 0 - 255 (0 is black, 255 is white). The pixel values to the left and right of our pixel are marked in

the image: 56 and 94. We just take the right value minus the left value and say that the rate of change in the x direction is 38 ($94 - 56 = 38$).

Note: At this pixel, the pixels from dark to light as we move left to right. If we looked at the same spot on the left side of the penguin's head where the pixels instead change from light to dark, we'd get a negative value for the change. You can compute the gradient by subtracting left from right or right from left, you just have to be consistent across the image.

We can do the same for the pixels above and below to get the change in the y-direction:



$93 - 55 = 38$ in the y-direction.

Putting these two values together, we now have our gradient vector.

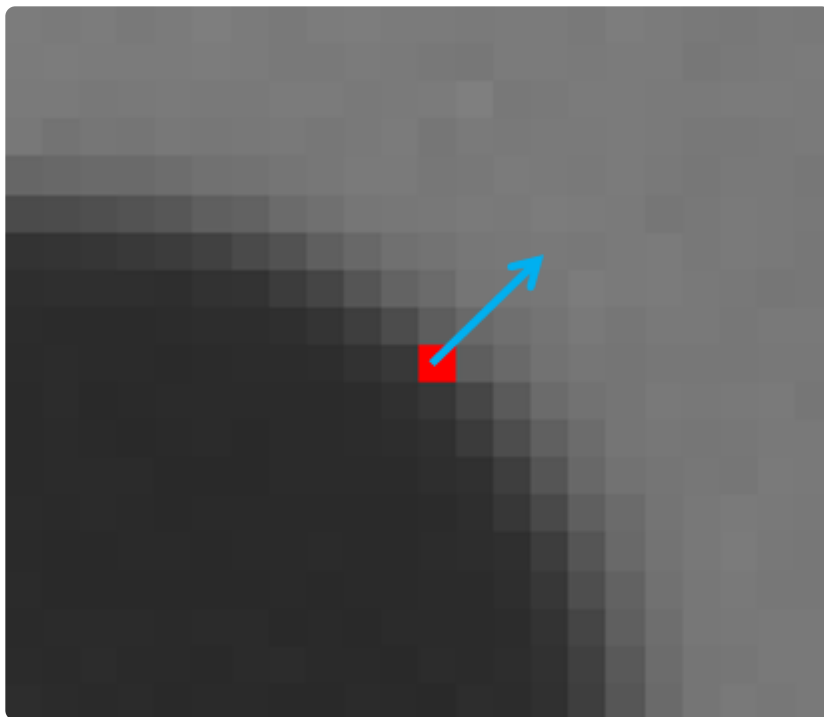
$$\begin{bmatrix} 38 \\ 38 \end{bmatrix}$$

We can also use the equations for the magnitude and angle of a vector to compute those values.

$$\text{Magnitude} = \sqrt{(38)^2 + (38)^2} = 53.74$$

$$\begin{aligned} \text{Angle} &= \arctan\left(\frac{38}{38}\right) = 0.785 \text{ rads} \\ &= 45 \text{ degrees} \end{aligned}$$

We can now draw the gradient vector as an arrow on the image. Notice how the direction of the gradient vector is perpendicular to the edge of the penguin's head—this is an important property of gradient vectors.



Let's see what it looks like to compute the change in the x and y direction at every pixel for the image. Note that the difference in pixel values can range from -255 to 255. This is too many values to store in a byte, so we have to map the values to the range 0 - 255. After performing this mapping, pixels with a large negative change will be black, pixels with a large positive change will be white, and pixels with little or no change will be gray.

Change in x-direction



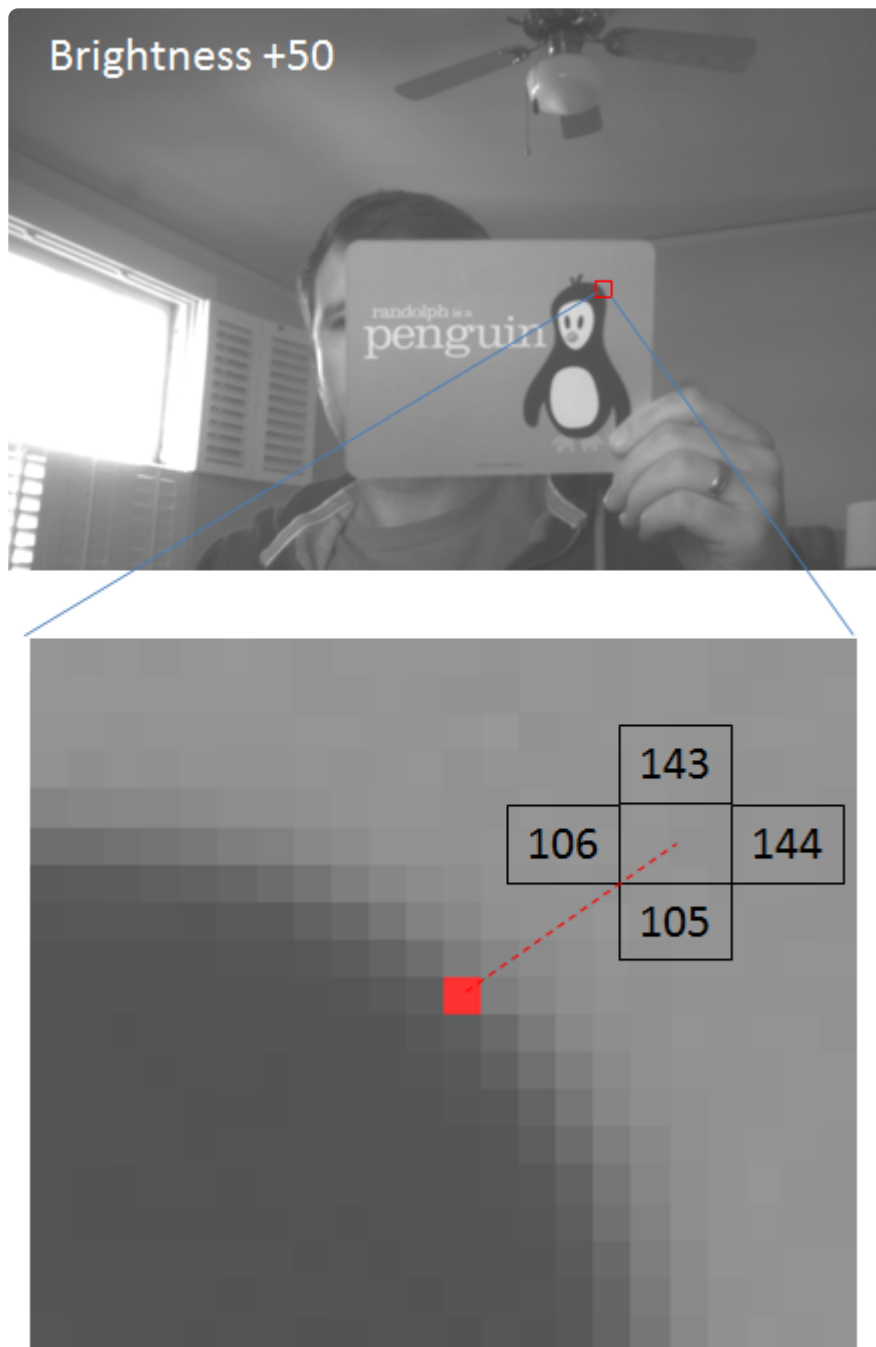
Change in y-direction

Gradient Vector Applications

The first and most obvious application of gradient vectors is to edge detection. You can see in the gradient images how large gradient values correspond to

strong edges in the image.

The other less obvious application is to feature extraction. Look at what happens to the gradient vector when I increase the brightness of the image by adding 50 to all of the pixel values.



In this brighter image, the rate of change in the x-direction is still $144 - 106 = 38$, and the rate of change in the y-direction is still $143 - 105 = 38$, the same as in our original image. So even though the pixel values are all completely different, we still get the same gradient vector at this pixel!

When we base our feature descriptors on gradient vectors instead of just the raw pixel values, we gain some “lighting invariance”. We’ll compute the same descriptor (or at least closer to the same descriptor) for an object under different lighting conditions, making it easier to recognize the object despite changes in lighting.

Mathematics

This is a brief introduction to gradient vectors without much use of the mathematical terms or expressions for what we’re doing. In another post, titled [Image Derivatives](#), I approach the same topic from a more mathematical perspective. The Image Derivatives post is actually my notes on a computer vision lecture given by Professor Shah which is freely available online.

39 Comments

mccormickml.com

 Login ▾

 Recommend 22

 Tweet

 Share

Sort by Best ▾



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS ?

**llgxvi** • a month ago

greate article, saves people a lot of time

^ | v • Reply • Share ›

**Ramses Alexander Coraspe Valde** • 3 months ago

Hi, Does anyone have any sample python code?

^ | v • Reply • Share ›

**llgxvi** → Ramses Alexander Coraspe Valde • a month ago

took me several hours with minimum python knowledge,

<https://github.com/llgxvi/i...>

^ | v • Reply • Share ›

**Irfan Iqbal** • 6 months ago

Nice Stuff Sir:-

Sir Kindy send me this code at irfan29641@gmail.com

Thanks in advance!!!

^ | v • Reply • Share ›

**llgxvi** → Irfan Iqbal • a month ago

more like Send me the code, peasant,

forgive me for posting this link again,

<https://github.com/llgxvi/i...>

^ | v • Reply • Share ›

**rishabh** • a year ago

Thanks for your explanation :)

^ | v • Reply • Share ›

**Vishesh Boin** • a year ago



Vishesh Dreja · a year ago

Thankyou for this explanation. I just have one question . why 85 was distributed between 70 and 90 and why not between 80 and 90 .. i mean 85 is closer to 80 then 70 and we also have 80 on the quantization scale?

^ | v · Reply · Share ›



moon · a year ago

Hi. Why subtract the neighboring pixels and not from the original pixel itself?
If we are moving from left to right, and our chose pixel is (red one) is 50. Why not 94-50 instead of 94 -56 which is to the left of red pixel(50). Thank you

^ | v · Reply · Share ›



llgxvi → moon · a month ago

check the link at the bottom of the article,
 $f(x+1) - f(x)$ is one of 3 possibilities,
it's called "forward difference"
the other two:
backward difference
central difference (the way the author used)

^ | v · Reply · Share ›



Adnan Gujjar · 2 years ago

hi **@@Chris McCormick** Please can you share code of this article. I want to draw histogram of image in which x-axis should be magnitude and and y axis should be Angle. I use sobel to find Magnitude and Angle in opencv with c++. Next how can i draw the histogram by using Angle and Magnitude?? Thanks in advance

^ | v · Reply · Share ›



Chris McCormick Mod → Adnan Gujjar · 2 years ago

Interesting! This code was written in Matlab, I believe.

To do what you're describing, I think you would need to use color or a grayscale value in order to represent the 'count' of the number of pixels with each magnitude and angle. You would also need to define bin ranges.

What I did here was different than a histogram - my images show the results of applying a

what I did here was different than a histogram--my images show the results of applying a filter to the image.

^ | v · Reply · Share ›



Adnan Gujjar → Chris McCormick · 2 years ago

ok thank you [@Chris McCormick](#) . If you feel free please email the code (ad.gujjar@gmail.com) .Later i also want to do the same thing you do. Thanks

^ | v · Reply · Share ›



Danil Lebediev · 2 years ago

thank you for your explanation! it's very helpful.

^ | v · Reply · Share ›



Chris McCormick Mod → Danil Lebediev · 2 years ago

Thanks!

^ | v · Reply · Share ›



Nittin Singh · 2 years ago

Wonderful stuff!!!!

^ | v · Reply · Share ›



Chris McCormick Mod → Nittin Singh · 2 years ago

Thank you!

^ | v · Reply · Share ›



Tan Yong Soon · 2 years ago

This is absolutely spot on. Thanks for the great explanation.

^ | v · Reply · Share ›



Chris McCormick Mod → Tan Yong Soon · 2 years ago

Thank you!

^ | v · Reply · Share ›



kuddai · 2 years ago



Concise and short explanation. Love it!

^ | v · Reply · Share ›



Chris McCormick Mod → kuddai · 2 years ago

Thanks!

^ | v · Reply · Share ›



Tanya Boone · 2 years ago

Thanks to your explanation I'm finally able to fully understand this method. I have one question for you: If I have a 32x32 image and obviously the 64x128 window won't work. Do you think if I resize everything proportionally will work?. Thanks in advance.

^ | v · Reply · Share ›



Chris McCormick Mod → Tanya Boone · 2 years ago

Hi Tanya, glad it was helpful. You're talking about the HOG detector window, correct? You could cut down the number of blocks in the detector to create a 32 x 32 pixel window.

^ | v · Reply · Share ›



preetam · 2 years ago

awesome explanation

^ | v · Reply · Share ›



Chris McCormick Mod → preetam · 2 years ago

Thanks!

^ | v · Reply · Share ›



Bhakti · 2 years ago

Using HOG Feature descriptor which type of feature I can extracted from the person image or Car Image. and can i match the features between two same images? and if is it Possible then How can we match? Thank You So much

^ | v · Reply · Share ›



Chris McCormick Mod → Bhakti · 2 years ago



Hi Bhatki - I didn't quite follow your question--are you just looking for some code for implementing the HOG descriptor?

And on the matching question, are you asking about testing whether two similar images are actually the same? If so, check out [pHash](#), I believe that might be simpler than HOG.

If you're talking about identifying people or cars in an image, then yes, the HOG detector is often used for those tasks.

^ | v · Reply · Share ›



magnetar · 2 years ago

Thanks Mr Chris,
Simple and clear

^ | v · Reply · Share ›



Chris McCormick Mod ➔ magnetar · 2 years ago

Thanks, glad it helped!

^ | v · Reply · Share ›



Bhakti · 3 years ago

Nice Explanation...Thank you Very much...I do appreciate with u and please always give more and more explanation in detail n ..this description always help us..

^ | v · Reply · Share ›



Chris McCormick Mod ➔ Bhakti · 2 years ago

Thanks for the kind words! Glad it was helpful.

^ | v · Reply · Share ›



Tiago D'Agostini · 3 years ago

Very good explanation, but can I add a suggestion? On your example you should have given an example where the X and Y components were different (not both 38), makes easier to follow in the following statements if you are reusing the same component or you are using both and which one is which on that case.

^ | v · Reply · Share ›



Chris McCormick Mod → Tiago D'Agostini • 3 years ago

Yeah, fair point. I used the actual gray-scale values from the image, but it probably wouldn't hurt to cheat and tweak one of them a bit for clarity :)

^ | v • Reply • Share ›



Tapas • 3 years ago

Thanks Mr Chris

Very simple explanation. I do appreciate your efforts.

^ | v • Reply • Share ›



Chris McCormick Mod → Tapas • 3 years ago

Thanks, glad it was helpful!

^ | v • Reply • Share ›



Tapas → Chris McCormick • 3 years ago

BTW did you really draw arrow(sky blue color) having both mag and direction ?
Or the arrow was having gray color.Thanks for your time.

^ | v • Reply • Share ›

[Show more replies](#)



Savio Pereira • 3 years ago

Hi Chris,

This is a really neat and intuitive explanation! Thanks a lot.

^ | v • Reply • Share ›



Chris McCormick Mod → Savio Pereira • 3 years ago

Thank you, glad it was helpful!

^ | v • Reply • Share ›



sagar • 3 years ago

Hi,

This is very nice and neat explanation. Thank You very much. Also the way you covered change

in brightness topic is also nice. As it is need to have algorithm which will give same output in different lighting condition.

But, as per my study when illumination changes. Not all the pixel are uniformly updated with new intensity values. I have observed that change in intensity values depends on reflection/absorption property of particular object in frame.

So, Is there any good method to tackle such issue. Do, we have method which will give correct result for illumination changing environment. ?

Your inputs will be valuable...

Thanks Again.

^ | v · Reply · Share ›



Cooper · 3 years ago


This post is awesome. Thanks.

^ | v · Reply · Share ›

ALSO ON MCCORMICKML.COM


AdaBoost Tutorial

17 comments · 3 years ago

 **disqus_OCqj7YZHey** — Good morning, Thanks for your early reply Chris! If you now some sources of information ...

The Gaussian Kernel

3 comments · 3 years ago

 **Игор Яловецкий** — Ok, i like your explanation about Gaussian curve, but what about the kernel?


Gradient Descent Derivation




65 comments · 3 years ago

 **Tushar Sarde** — Thanks!

Image Derivative

16 comments · 3 years ago

 **Aya al-bitar** — I've read your blogs about HOG , Gradient vector and image derivative and they were extremely helpful, Thanks ..

 **Subscribe**  **Add Disqus to your site** **Add Disqus**  **Disqus' Privacy Policy** **Privacy Policy** **Privacy Policy**

Related posts

[The Inner Workings of word2vec](#) 12 Mar 2019

[Applying word2vec to Recommenders and Advertising](#) 15 Jun 2018

[Product Quantizers for k-NN Tutorial Part 2](#) 22 Oct 2017

© 2019. All rights reserved.