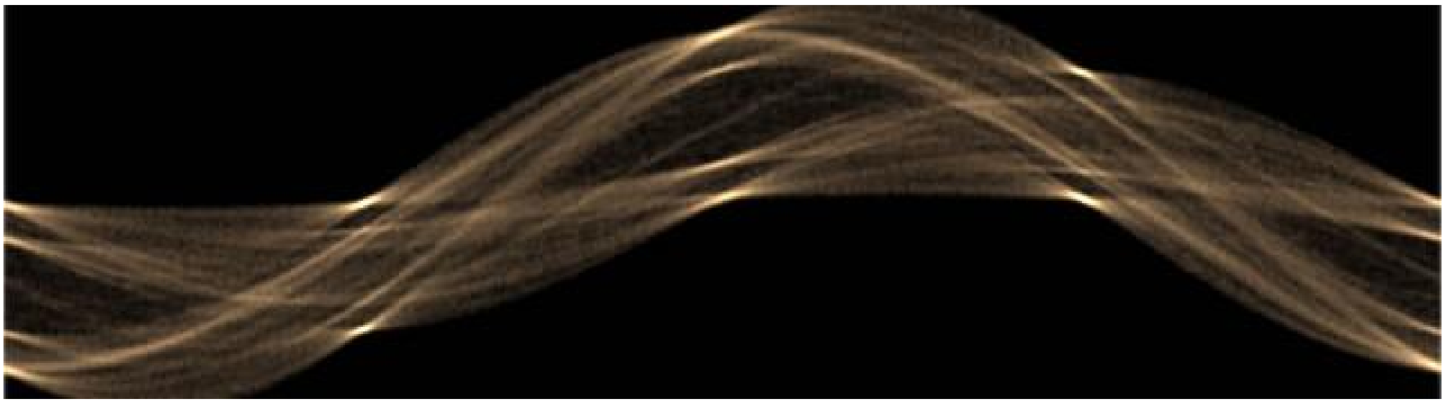


The Hough Transform

By Utkarsh | *Published: March 6, 2010*

The Hough transform is an incredible tool that lets you identify lines. Not just lines, but other shapes as well. In this article, I'll talk about the mechanics behind the Hough transform. It will involve a bit of math, but just elementary concepts you learned in school.

In this article, we'll work with lines only, though the technique can be easily extended to other shapes.

Why the Hough Transform?

Lets say you take the snapshot of pole. You figure out edge pixels (using the Canny edge detector, the Sobel edge detector, or any other thing). Now you want a geometrical representation of the pole's edge.You want to know its slope, its intercept, etc. But right now the "edge" is just a sequence of pixels.

You can loop through all pixels, and some how figure out the slope and intercept. But that is one difficult task. Images are never perfect.

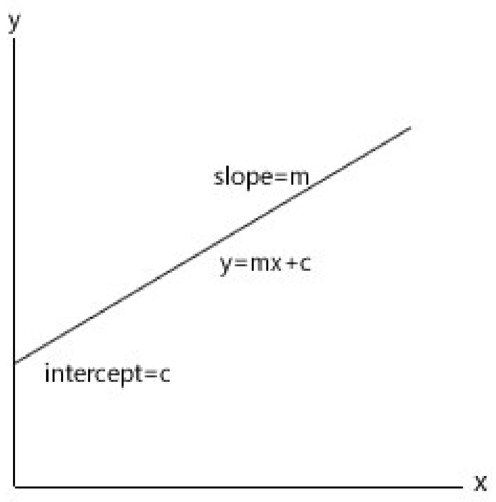
So you want some mechanism that give more weightage to pixels that are already in a line. This is exactly what the Hough Transform does.

It lets each point on the image "vote". And because of the mathematical properties of the transform, this "voting" allows us to figure out prominent lines in the image.

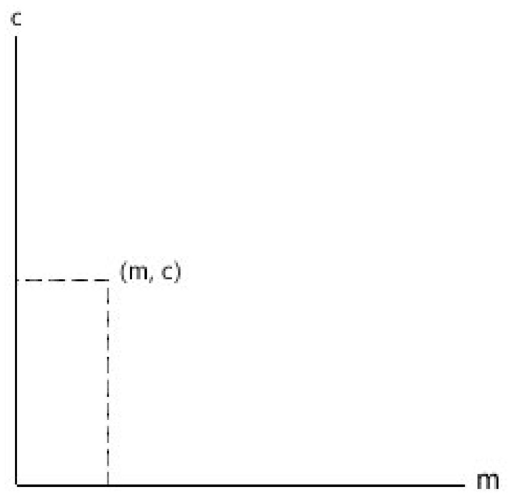
From lines to points

A lines is a collection of points. And managing a collection of points is tougher than managing a single point. Obviously. So the first thing we learn is how to represent a line as a single point, without losing any information about it.

This is done through the **m-c space**.



xy space



mc space

As shown in the above picture, every line has two quantities associated with it, the slope and the intercept. With these two numbers, you can describe a line completely.

So the parameter space, or the mc space was devised. Every line in the xy space is equivalent to a single point in the mc space. Neat eh?

From points to lines

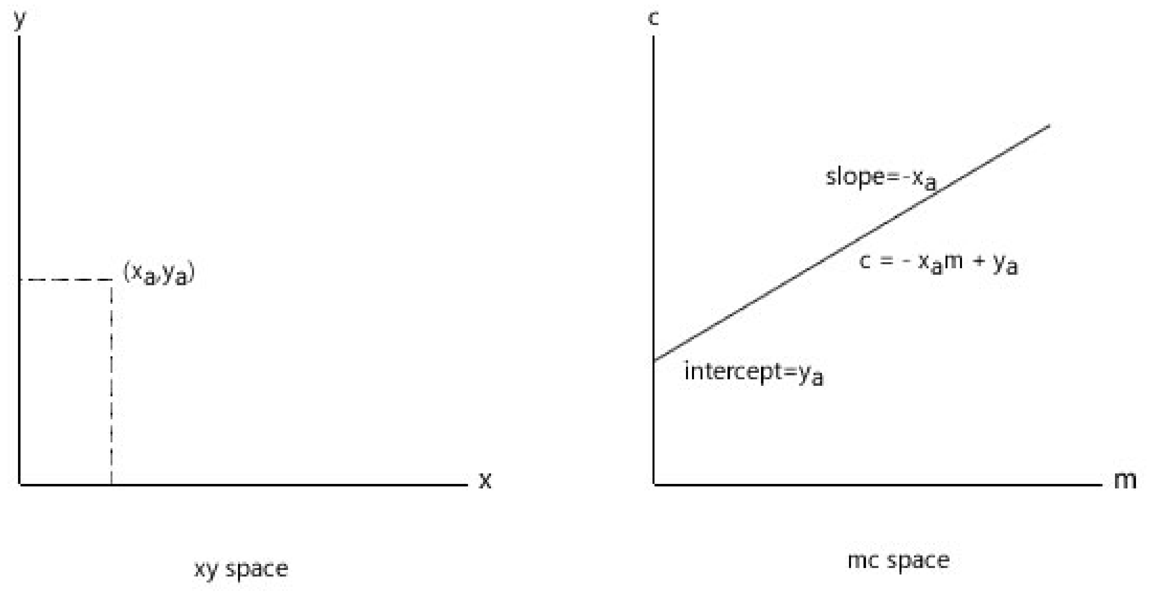
Now onto the next step. Consider a point (say, (x_a, y_a))in the xy space. What would its representation in the mc space be?

For starters, you could guess that infinite lines pass through a point. So, for every line passing through (x_a, y_a), there would be a point in the mc space.

And you’re correct there, because of the following:

1. Any line passing through (x_a, y_a): $y_a = mx_a + c$
2. Rearranging: $c = - x_a m + y_a$
3. The above is the equation of a line in the mc space.

So, a point in the xy space is equivalent to a line in the mc space.

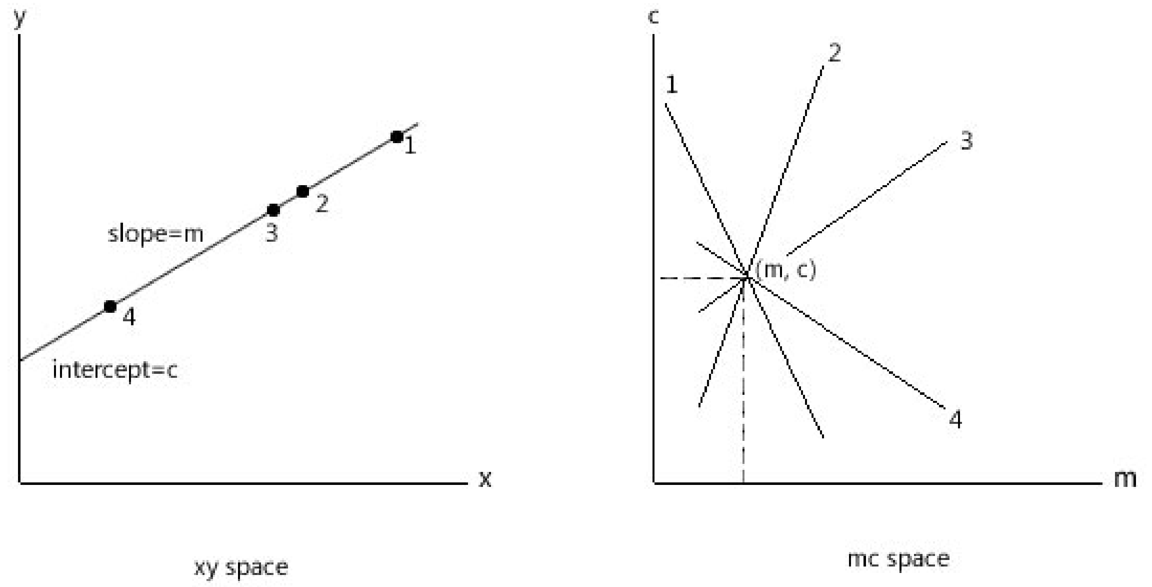


How does this help us?

The Hough transform is all about doing what we just learned: converting points in the xy space to lines in the mc space.

You taken an edge detected image, and for every point that is non black, you draw lines in the mc place. Obviously, some lines will intersect. These intersections mark are the parameters of the line.

The following picture will clarify the idea:



The points 1, 2, 3, and 4 are represented as various lines in the mc space. And the intersection of these lines is equivalent to the original line.

Great!

Now you know the basic idea of how the Hough transform works. But there's one big flaw in the We'll discuss how to resolve this issue in the next article.

More in this series

1. The basics
2. Solving the problem

Issues? Suggestions? Visit the Github issue tracker for AI Shack

Back to top

Suka 2.575 orang menyukai ini.

Downloads

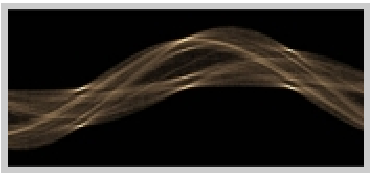
Download the VC2008 code

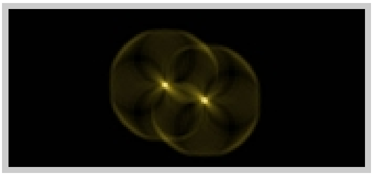
Download the patent filed by Hough

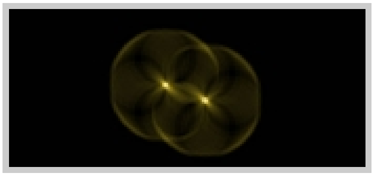
Summary

This is a test summary

- First
- Second
- Third


Hough transform in OpenCV


Circle Hough Transform


Hough circles in OpenCV

11 Comments

aythin
Posted January 25, 2011 at 3:14 pm | *Permalink*

hi,
i gone through your materials…it was useful

i need the database of images used in this project since i am doing a project on object recognition.Also i need codings for image matching and finally recognition

Reply

Utkarsh
Posted January 25, 2011 at 6:10 pm | *Permalink*

What database? What code?

Reply

Sivadev
Posted March 10, 2011 at 3:23 pm | *Permalink*

Hello,

I have one question, can be use houghlines or houghcircles to detect object of vision/video. The example give is for a “goal_arena.jpg” picture / still image. Is it possible?

Reply

Utkarsh
Posted March 11, 2011 at 4:11 pm | *Permalink*

Yes it’s possible. You need to first detect edges in that still image (like canny, sobel, etc). Then you pass it to the hough transform. Then you’ll get the lines.

Reply

snehal
Posted April 22, 2011 at 12:24 pm | *Permalink*

This material is very informative.How it will be usefull to determine the Horizon? and what if I have to consider texture information in images?

Reply

surya
Posted June 19, 2011 at 8:54 am | *Permalink*

hi ,your tutorial was useful for me.
i wanted to try line following robot using image processing.
as i go through this hough transform tutorial i came to know that i can detect lines using this

transform.and my question is how can we detect squares, and how to send data to robot after detecting line for following path? i am using zigbee communication between robot and PC.

thank you.

Reply

Utkarsh
Posted June 19, 2011 at 11:04 am | *Permalink*

If you just want to move the robot, you can send out motion commands to the robot – move forward, turn left, etc. If you want to send the lines, you could send a byte for rho and another byte for theta. You could put a blank byte in between to keep everything in sync (like 0xFF). Simple as that.

Reply

pyal
Posted July 22, 2011 at 9:54 am | *Permalink*

It's simple to understand and useful for learning. Great-UTkarsh

Reply

pyal
Posted July 22, 2011 at 10:12 am | *Permalink*

Hi UTkarsh,

Thanks for your nice tutorial.
Could you please tell me something about the stuffs like Hough Transform (for 3 D), RANSAC, Voroni diagram, delaunay triangulation. Where I can get some thing easy tutorial as you posted. I need to these since I'm working on point cloud- 3D data processing.

wish you a nice time,
pyal

Reply

Gaurav Mittal
Posted November 21, 2013 at 2:23 am | *Permalink*

Awesome work dude, I am so glad I found your website. Planning to read all your post 😃

Reply

Utkarsh
Posted April 27, 2014 at 11:05 pm | *Permalink*

I'm glad *you* found me!

Reply

One Trackback

- By Referências em visão computacional | onox on September 8, 2011 at 9:11 am

Post a Comment

Your email is *never* published nor shared. Required fields are marked *

Name *

Email *

Website

Comment

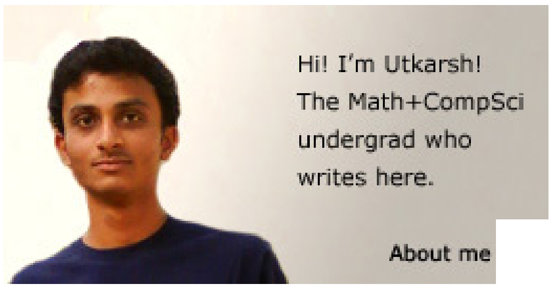
You may use these HTML tags and attributes `<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <strike> <strong>`

Post Comment

Free updates

Email Address

More info



Popular Posts

- Scanning QR Codes
- The Canny Edge Detector
- Implementing Canny Edges from scratch
- Image Moments
- Predator: Tracking + Learning

More on AI SHack

Popular **Beginners** **Utkarsh's favs**

- Scanning QR Codes
- The Canny Edge Detector
- Implementing Canny Edges from scratch
- Image Moments
- Predator: Tracking + Learning

About me

My name is Utkarsh Sinha, and I'm an undergraduate student, pursuing B.E. Computer Science + M.Sc. Mathematics. Here, I help you understand ideas in Artificial Intelligence, using a not so techy and mathematical language. And in the process, learn more about Artificial Intelligence myself.

Read more at the about page