Hough Transform for Directional Orientation

By

Mr. Sirapop Nuannimnoi ID. 56090500421

Mr. Thanawat Tosakul ID. 57090500417

Presented to

Dr. Peeraya Sripian

This project is a part of study

in the course of CSS 331 Computer Vision.

Academic Year 2016. Semester 1.

King Mongkut's University of Technology Thonburi

(This work is based on the original work by Matthew Ho and Ivan Papusha of Stanford University)

Project Title	Hough Transform for Directional Orientation
Candidates	Mr. Sirapop Nuannimnoi
	Mr. Thanawat Tosakul
Project Advisor	Dr. Peeraya Sripian
Program	Bachelor of Science (Applied Computer Science)
Department	Mathematics
Faculty	Science
Course	CSS 331 Computer Vision
Academic Year	2016

Abstract

Based on the original work by Matthew Ho and Ivan Papusha, the main purpose of this project is to implement a Hough transform for directional orientation. Unlike the original work, we develop a small computer program using Visual C++ and OpenCV with the algorithms of the original work and our software is limited only for finding vanishing points of still images, which indicates direction, with least-squares approximation method for one vanishing point detection, and for multiple vanishing points, pairwise intersections k-means clustering method.

Keywords : Directional Orientation, Hough Transform, Peak Detection, Vanishing point

INTRODUCTION

Vanishing points are points in a picture which are the intersections of a set of parallel lines. Vanishing points correspond to the eye point from which the image should be viewed for correct perspective. For example, a vanishing point in a hallway can be an indicator for which “direction” the hallway is pointing in. The detection for these points is beneficial in many fields including field robotics, computer science, 3D image reconstruction and so on. There are several methods for vanishing point detection. One of the well-known methods makes use of the line segments detected in images which can be detected with Hough transform. We expect that after performing Hough transform on the sample still images, we might be able to detect one vanishing point using least-square L2-norm approximation on Hough peaks, and detect multiple vanishing points using k-means clustering on the pairwise intersections of the detected line segments in our sample images.

RELATED WORKS

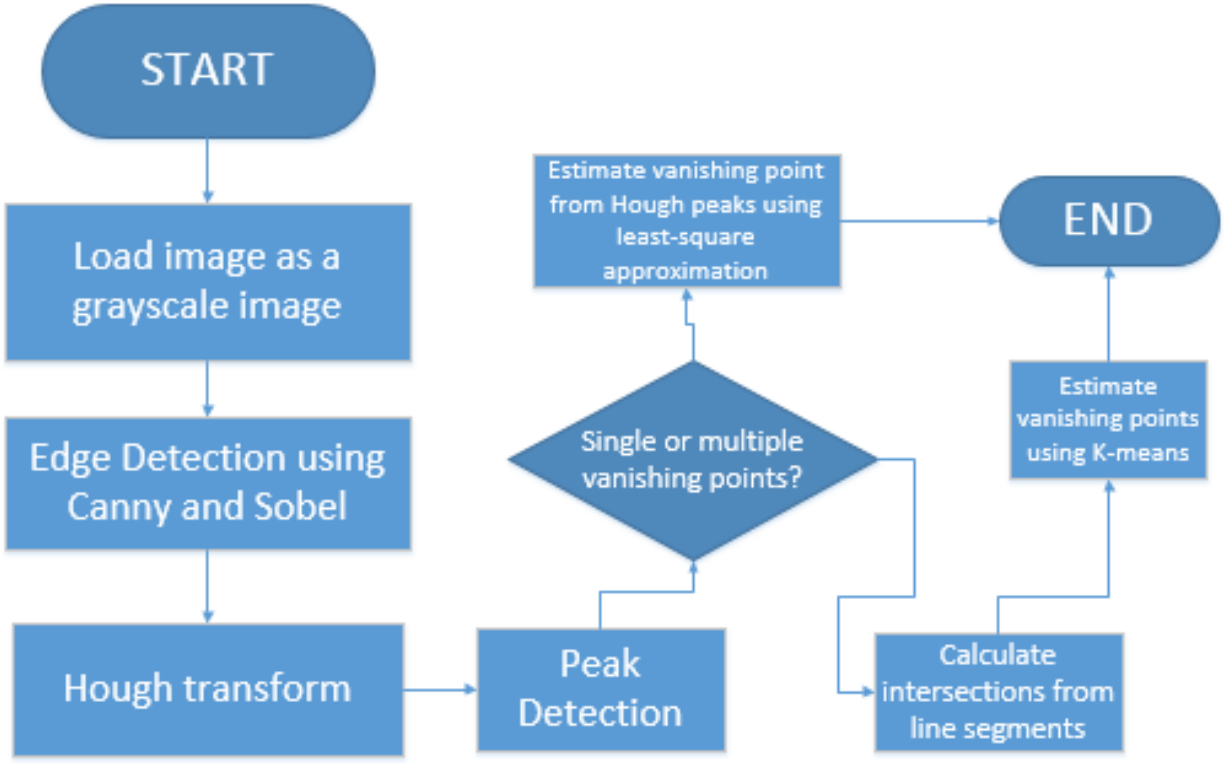
This work is based on the original work by Matthew Ho and Ivan Papusha of Stanford University, published in Spring 2010. **[1]** We studied from both their published paper and codes provided on the website of Stanford University. For multiple vanishing points detection, we implemented our own K-means from scratches.

Moreover, we looked through OpenCV documentation site to learn how to use some OpenCV functions. We listed the OpenCV functions that we used in the table below.

Functions	Link to description
HoughLines(dst, lines, 1, CV_PI/180, 100, 0, 0);	http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html
Canny(detected_edges, detected_edges, lowThreshold, lowThreshold*ratio, kernel_size);	http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html
Sobel(src_gray, grad_x, ddepth, 1, 0, 3, scale, delta, BORDER_DEFAULT);	http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/sobel_derivatives/sobel_derivatives.html

IMPLEMENTATION

Flowchart



For the formula of the least-square approximation, we used the same formula that was described in the paper.

$$\begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) \\ \cos(\theta_2) & \sin(\theta_2) \\ \vdots & \vdots \\ \cos(\theta_n) & \sin(\theta_n) \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \approx \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_n \end{bmatrix}$$

From the equation above, we get the left most matrix as A. The right most is ρ. We need to find the one in the middle that make the difference of A – ρ as small as possible, which in the original paper, the least-square approximation was formed as:

$$\begin{aligned} x &= (A^T A)^{-1} A^T \rho \\ &= \frac{1}{\det(A^T A)} \begin{bmatrix} a_2^T a_2 & -a_1^T a_2 \\ -a_2^T a_1 & a_1^T a_1 \end{bmatrix} \begin{bmatrix} a_1^T \rho \\ a_2^T \rho \end{bmatrix} \end{aligned}$$

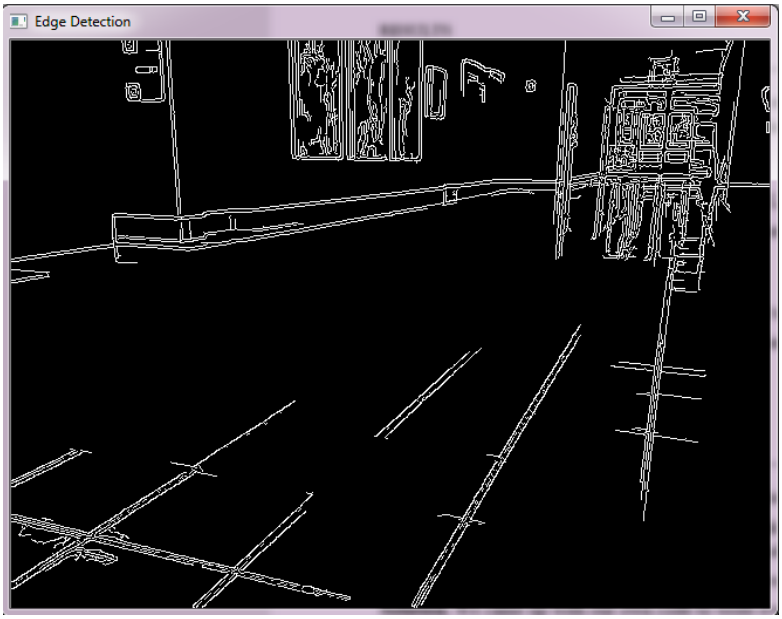
Roles and Responsibilities

Thanawat Tosakul (57090500414) is responsible for coding Edge Detection and Hough transform, and debugging the software. Sirapop Nuannimnoi (56090500421) is

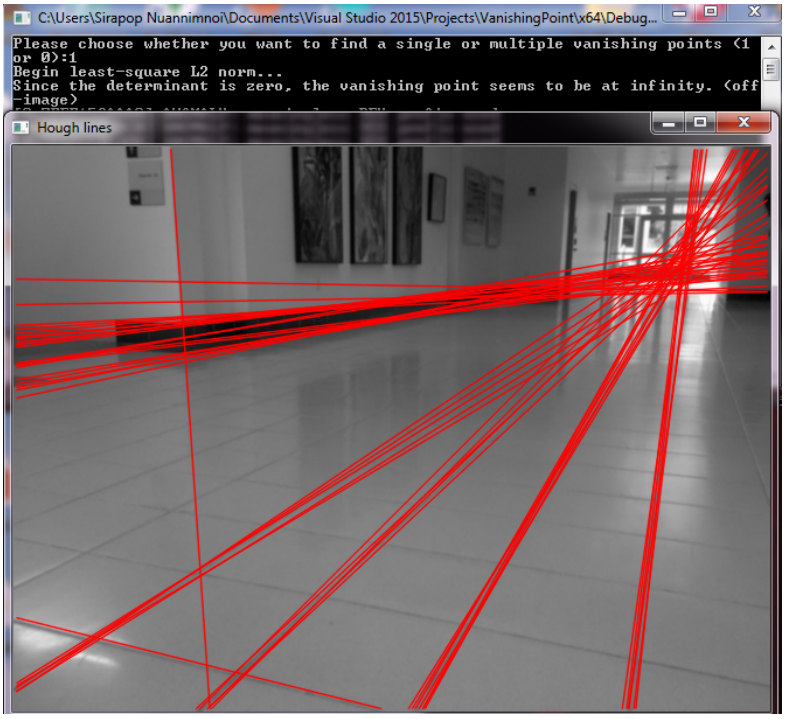
responsible for designing and coding least-square approximation and K-means based on original algorithms from the paper.

RESULTS

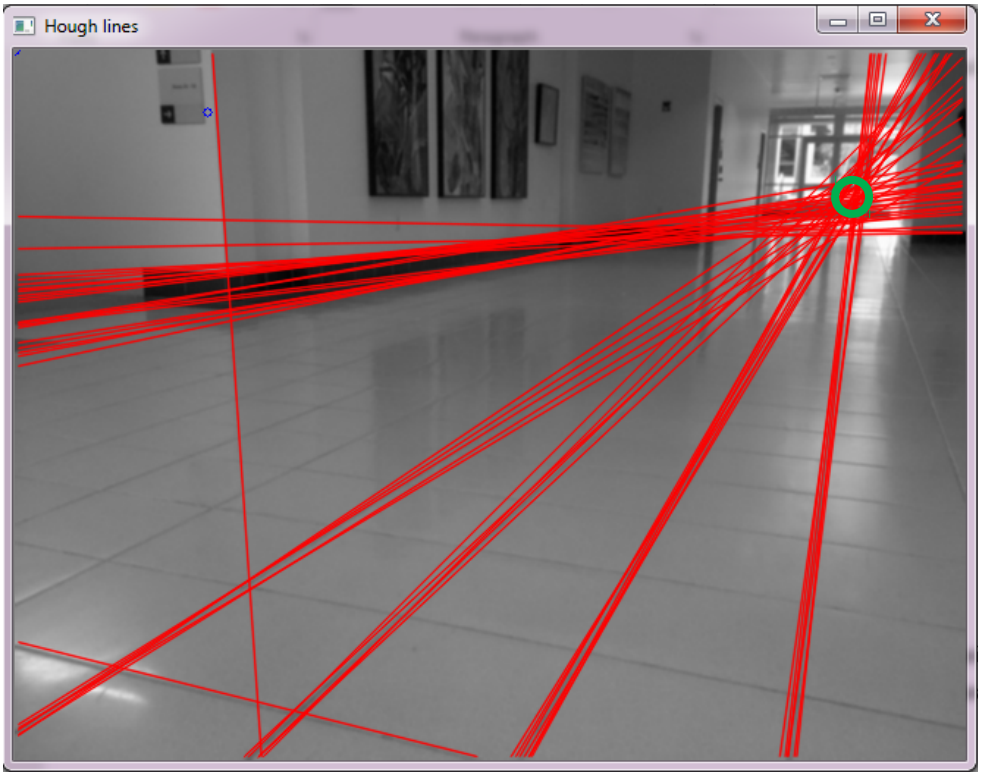
First, this is our result for edge detection. In our work, we changed the methods for edge detection a little. Instead of using simple thresholding mentioned in the original work, we use Sobel and Canny methods.



After we followed a formula of least-square L2-norm approximation in the original paper, our result shows that the single vanishing point in the sample is off-image, meaning that it is invisible.



Later, we implemented the K-means method to find multiple vanishing points with K equal to 3. However, the results were not as we expected. As seen in the picture below, the blue dots show the locations of three vanishing points but the expected results should be around green circle.



DISCUSSION

Problems: The project owners use Matlab codes which are quite difficult for us to understand especially when we looked through the code for peak detection, single vanishing point detection, intersection calculation and K-means. Besides, the project of similar work we found on the Internet is not up to date, some parts of the program cannot be compiled.
Solution: We came up with our own code to write every needed part for finishing this project. However, this solution is not effective since we are also not sure whether our results were correct or we were just lucky that they looked surprisingly correct.

FUTURE WORK

If we had more time, like two or maybe three months, we would try to implement our program to support video files. This program can be developed to the level that we can use it in real world with real time data through cameras. In the future, vanishing point detection might help a lot in automated driving and robot navigation.

CONCLUSION

Our small program can detect single vanishing point by using least-squares approximation on the detected Hough peaks, and can also detect multiple vanishing points from the pairwise intersections of detected line segments by using K-means clustering.

REFERENCE

---

[1] Papusha, I., & Ho, M. (n.d.). Hough Transform for Directional Orientation. Retrieved November 8, 2016, from https://stacks.stanford.edu/file/druid:rz261ds9725/Ho_Papusha_RealTimeHoughTransform.pdf