SDU International Journal of Technologic Sciences
Vol. 3, No 1/Special Issue, February 2011
pp. 11-22

Computer Technology

DRIVABLE FLOOR DETECTION WITH MONOCULAR VISUAL PERCEPTION

Onur ŞENCAN[*], Evangelos SARIYANİDİ [**], Hakan TEMELTAŞ [***]

Abstract

In this study, vision-based scene analysis techniques are studied for the purpose of improved navigation abilities for the mobile robots with a single camera. In this approach, first the camera searches the scenery in order to detect major vertices. In the next step, using geometrical properties of the drivable floor regions in which the mobile robot can without collusion, are defined. Experimental works are implemented on a differential drive robot with monocular camera and the success of the approach is demonstrated.

Keywords: robot vision, scene analysis, drivable floor regions detection, autonomous mobile robots, monocular vision perception.

1. Introduction

Nowadays, maintaining the faultless performance of the mobile robots in unmapped environments especially with the existence of dynamical objects is a major problem to overcome in order to realize fully autonomous navigation techniques. Obtaining the data of the fixed reference points (*landmarks*) to determine localization of both landmarks and observer is a problem treated under SLAM [1] which is used for mapping, correcting the odometer error and collision avoidance with a guidance of a path planned beforehand [2]. In addition to this fact, it is obvious that there is a demand on a camera system to construct the autonomous abilities in the literal sense and the awarement of the robots in a dynamic environment.

In spite of the analogy between human eye and camera, there are a lot of task that can be solved by human perception system quite easily, could be a challenging problem for robot vision system design. Most particularly, becoming different from ideal situations at indoor navigation under the influence of nonhomogeneous illumination, uncertainties in neighborhood and measurements with addition of nonmodelled disturbances cause difficulties in vision-based solutions.

Within this study, we propose methods for identification of drivable floor regions to support autonomous navigation capabilities of robots in indoor applications.

[*] Istanbul Technical University, Department of Control Engineering, 34469 Maslak Istanbul/Turkey, E-mail: osencan@itu.edu.tr

[**] Istanbul Technical University, Department of Control Engineering, 34469 Maslak Istanbul/Turkey, E-mail: sariyanidi@itu.edu.tr

[***] Istanbul Technical University, Department of Control Engineering, 34469 Maslak Istanbul/Turkey, E-mail: hakan.temeltas@itu.edu.tr

2. Methodology

Frames that are obtained through a single camera are processed one by one in order to detect drivable floor regions. The detection is based both on color and shape similarities. These similarities are tested in different ways so that the detection is not limited to ideal circumstances such as ideal color regions or ideal illumination. Furthermore, the corrupted corridor lines are aimed to be corrected. Two color modes have been used: YUV and RGB. Both modes have several advantages and disadvantages, so each mode has been substituted into premises which cover the disadvantages of the other.

In the first place the detection of corridor lines is accomplished. In advance to achieve this, every frame is blurred in order to eliminate misleading noises. The edge regions on the blurred image are detected and the lines from the edge image are acquired with Probabilistic Hough Transformation (PHT) [2]. The problem with PHT is that even clear and long lines on the image are represented as small line segments which are unpractical and cause larger computational cost. To overcome this problem, an algorithm which clusters similar small lines as bigger lines (w.r.t slope, distance and color similarities) is developed. Another algorithm is developed in order to filter the clustered lines as presumptive corridor edges through perspective knowledge. These edges are evaluated and a large region is marked as floor, which will be used as a mask for other processes.

Secondly, "Flood filling" is applied on several (usually two or three) points, which are priori known as floor points. Briefly, flood filling is the clustering of similar-valued points around a certain point. The intersection of flood fill result and the possible floor mask explained in the paragraph above, results more precise region of the drivable areas. Overall procedure of this study can be seen at Figure 1. A raw corridor image with poor illumination is chosen for the applications.

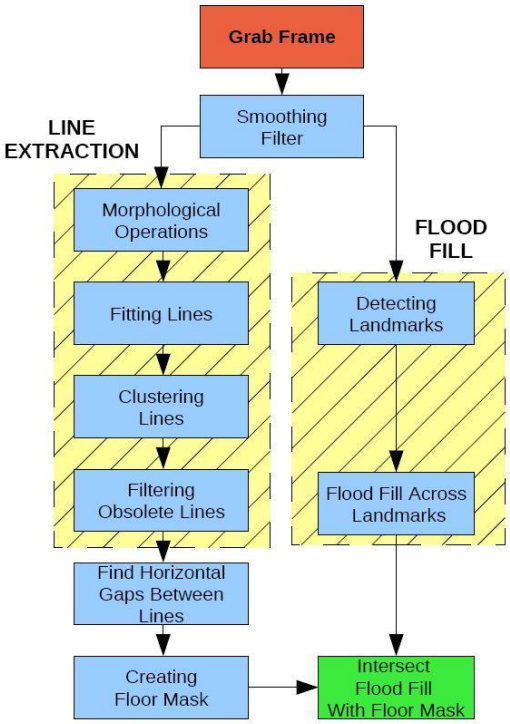

Fig.1. Overview: Drivable regions detection procedure

3. Image Processing

3.1 Smoothing

Smoothing is an image processing operation which is mostly applied in order to reduce the noise. Especially line detection outcomes can be very noisy without smoothing. There are several types of smoothing; four types of smoothing have been tested through this project: Median Filter, Gaussian Filter, Bilateral Filter and Mean-shift Filter. As an edge-preserving smoothing filter, Bilateral Filter has given the best solution for accurate line extraction.

Median Filter. Median Filter [3] is an elementary filter type in which the values of pixels within a square neighborhood are replaced with the value of center (median) pixel. Although Median Filter is a fast filter, it is not useful for line extraction applications.

Gaussian Filter. In comparison to Median Filter, Gaussian Filter is a more useful but slower type of filter. In the filtering process values of the pixels within a square neighborhood are distributed according to a Gaussian distribution around the center pixel.

Bilateral Filter. Although Bilateral Filter is similar with Gaussian Filter, it differs from Gaussian Filter with its tendency to preserve the edges within an image. [4] This feature qualifies the filter as the most useful one for line extraction applications. There are several parameters which contribute to the application of Bilateral Filter. These have been determined for this project with experimental results. It is possible to develop a function that sets these parameters to provide stability under different illumination effects, image size, etc.

Mean-shift Filter. Mean-shift Filter is a filter type which finds color peaks in image and emphasizes these peaks while blurring the pixels of similar color values. Although Mean-shift Filter can be very useful for several applications such as landmark extraction (assuming that landmarks have a specific color), it has no specific importance in line extraction [5]. The regions of the image with the extreme color values have been isolated while the similar-valued regions are blurred.

3.2 Morphological Operations

One of the most significant points in this project is line extraction. Therefore, in this case the accurate edge detection is very crucial. The logic under edge detection is computing the derivative of the image in a specific direction or specific directions. High-derivative points that correspond to the rapid change of pixel values, mostly means that a discontinuity exists in that region of the image. There are several edge detection algorithms and three of these have been tested for this project: Laplace, Sobel and Canny Transformation. Canny Transformation has been the preferred one in this project for the reasons explained under each subtitle below.

Sobel Transformation. Sobel Transformation [6] is a transformation algorithm which works best in a single direction, in x or y, which means that horizontal or vertical lines could be easily and accurately detected through this algorithm. But this project involves the line detection of corridor lines which are never vertical and rarely horizontal. Although this algorithm is not specifically useful for this project, it can still be useful and more efficient for other robotic vision processes such as wall detection, which requires the detection of vertical lines.

Laplace Transformation. Similar to Sobel Transformation, Laplace Transformation involves the computation of derivative of the pixel values, but the derivative is computed in two directions: x and y.

Canny Transformation. Laplace Transformation forms the basis of this transformation. As a very widely used edge detection algorithm Canny Transformation varies from Laplace Transformation in two ways: The first difference is the computation of the derivative of image data in four directions instead of two, and the second difference is the algorithm tries to form closed regions from the derivation results. Result of the Canny Transformation on a corridor image can be seen at figure 2.

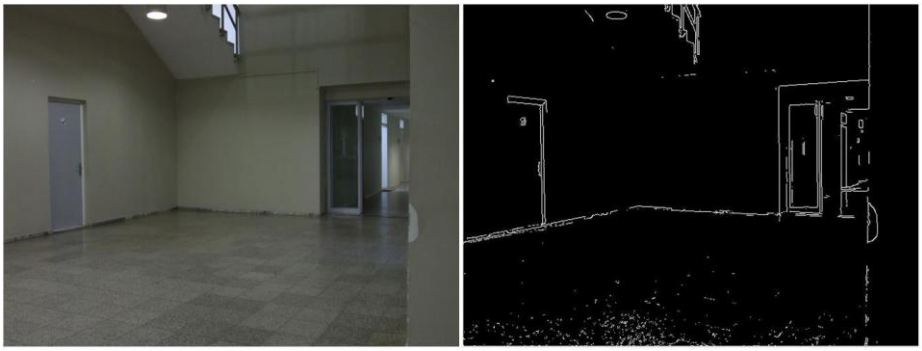

Fig. 2. Results of Canny Filter applied on the raw corridor image

4. Line Extraction Methods

The step that follows the edge detection is line detection. Hough Transformation [7] is the most widely used algorithm for finding lines in an image. There are two main types of Hough transformation: Standard Hough Transformation and Probabilistic based Hough Transformations.

4.1 Detecting Lines

4.1.1. Detecting Lines with Standard Hough Transformation

Standard Hough Transformation (SHT) [7] is the basic line finding algorithm in the literature. It is applied on to binary images which are the edge detection outcomes in this case. In SHT, lines are represented with the line slope α and a point on the line p1. This representation corresponds to an infinite line instead of a line segment. Therefore, this indication has some disadvantages.

First of all, even small line segments are represented as infinite lines. This results into a very noisy line image which is not useful. Secondly, the actual position of a line remains unknown, even if the line on the edge image is very clear. Assume that there is a noiseless horizontal line of 100 pixels on the middle of the edge image with size of 640 pixels width and 480 pixels height. The line found through SHT will be an infinite horizontal line on the middle of the image with the correct height value. But the vertical position of the line will remain totally unknown; it can be on the left side or on the right side of the image. It can be even impossible to say whether a line belongs to the corridor or the ceiling.

4.1.2. Progressive Probabilistic Hough Transformation(PPHT)

Progressive Probabilistic Hough Transformation [8] is also applied on a binary image, specifically to the edge detection results. The greatest difference between SHT and PPHT is

the PPHT finds the line segments on an image instead of fitting infinite lines, which is much more useful in this application. The disadvantage of this method is that even clear and long lines are represented as small sequential line pieces. This is impractical and takes more computational time comparing to SHT. An algorithm is developed for this project to eliminate these disadvantages of PPHT [8]. This algorithm is named Line Clustering and it is discussed in the next title in detail.

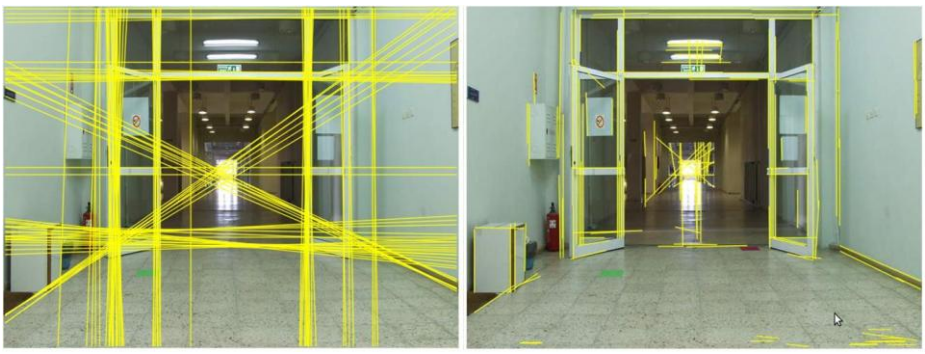

Fig. 3. Standard Hough Transformation (left) and Progressive Probabilistic Hough Transformation (right)

5. Line Clustering Methods

5.1 The Need for Line Clustering

As stated in the title above, PPHT is the line finding method used in this project which has a disadvantage: representing even clear and long lines as unpractical small line pieces. In order to eliminate this disadvantage, an algorithm is developed to merge small line pieces which actually represent a bigger line. Comparison of the SHT and the PPHT methods can be seen in figure 3. Each corridor line in the edge image in figure 3 (right) has been represented as small line pieces with PPHT. In another corridor view the unmerged line pieces are shown on the left side of the figure 4. The output of line clustering is shown in the right side of the same figure.

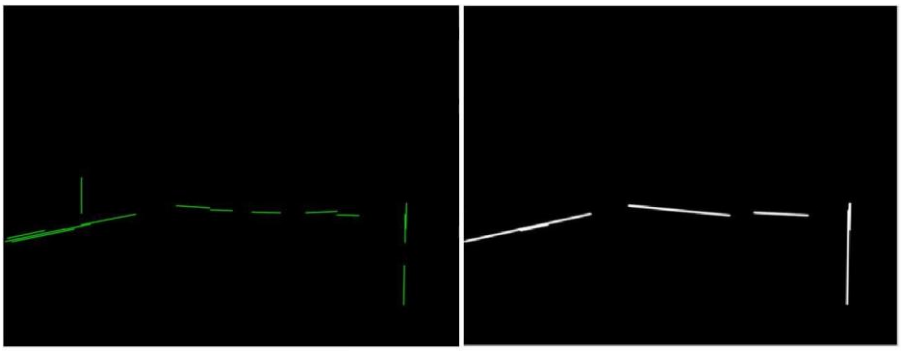

Fig. 4. PPHT results on the edge image are small line pieces (left), these pieces are merged through line clustering algorithm (right)

5.2 The Logic under Line Clustering

The goal is detecting the lines which are spatially close and have a similar slope. In other words; clustering the line pieces separately which represent an actual line on the image. A recursive clustering algorithm is used for clustering the lines.

Each line cluster has two basic data of its own: a slope and a points array. The slope is the average slope of the line pieces which forms the line cluster. The points array is the array in which the initial and final points of each line piece of the cluster is stored.

The first line piece of the loop forms also the first cluster. This means that the slope of the first cluster is equal to the slope of this first line piece, and the members of the points array of this cluster is the initial and final line of this line piece.

The second line piece of the loop does not directly form a new cluster. It is checked whether the slope of this line piece is similar with the slope of the line cluster which was just created. If the slopes are similar and the line piece is spatially close to this cluster, the line piece is added to the first cluster. The slope of this cluster is updated and the initial and final points of this line piece are added to the points array of the cluster. But if the slopes are not similar or the line piece is not spatially close to the cluster, a new cluster is formed for this second line piece. The process goes like this and the general flow of the algorithm can be seen in figure 5.

6. Floor Mask Creation through Perspective Information

6.1 Basic Perspective Definitions

Perspective rules are the principles which are applied to represent 3D objects or images as a 2D image. The loss of an axis entails certain deformations on the 2D image. These deformations follow certain rules and these rules can be used to gather information to reconstruct the third axis. Some definitions about perspective are necessary to understand how corridor lines are filtered through these rules.

Cartesian Scene. Cartesian scene is a scene constituted mainly by linear elements. An indoor environment can be considered as a Cartesian scene and the elements of this scene are corridor lines, wall lines, doors, windows, paintings on the wall etc. All of these elements are usually placed in an order. Through this order, the distortions caused by the projection of an indoor scene on a 2D image can be analyzed systematically.

One-point Perspective. In this project one-point perspective exists for camera rotations of $0, \pi, \ldots, n\frac{\pi}{2},\ n \in \square$ degrees in regards to construction the floor plan that robot moves in. In other words, one-point perspective exists when the image plate of the camera is parallel to the indoor scene in two axes [10]. In this case, vertical lines (walls, vertical edges of a door etc.) are represented as parallel, non-converging vertical lines. The horizontal lines of the scene which are parallel to the image plate are represented as parallel, non-converging horizontal lines. All other lines which have no parallelism between indoor scene and image plate are represented as lines which converge to one point, the single vanishing point (see figure 6 for illustration).

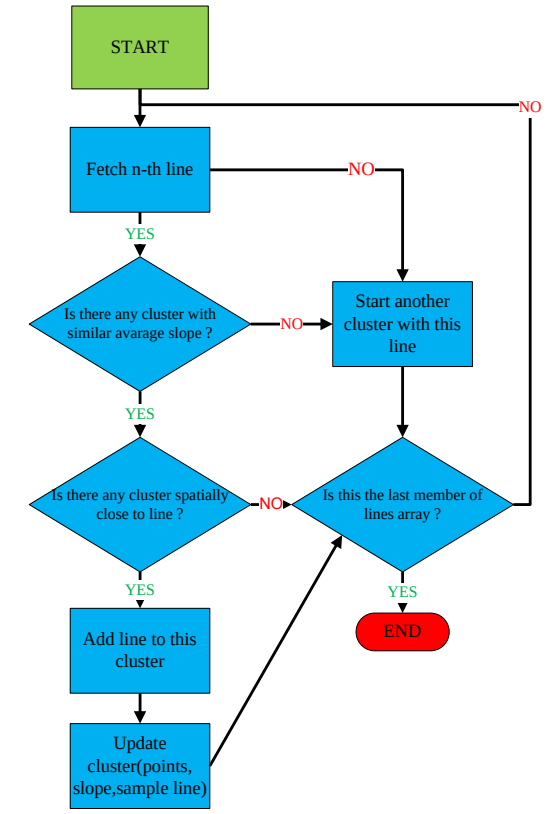

Fig. 5. Overview: The procedure of line clustering

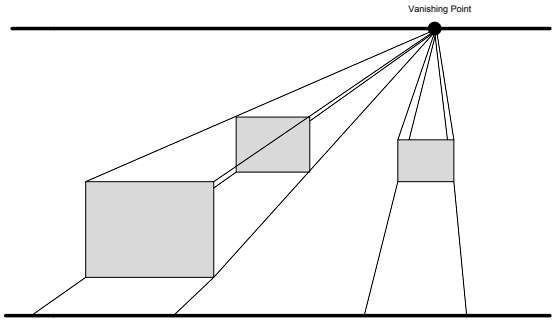

Fig. 6. One point perspective – image plate is parallel to Cartesian scene in two axes

Two-point Perspective. In this project two-point perspective exists for camera rotations which are different from floor plan like in the one-point perspective as already mentioned. In other words, two-point perspective exists when the image plate of the camera is parallel to the indoor scene in only one axis [11]. Vertical lines of the scene are represented as vertical lines similar to one-point perspective rules. All other lines converge to two separate points: Left vanishing point and right vanishing point (see figure 7)

Two-point perspective has a special emphasis for indoor applications. This project involves mobile robots with an integrated camera. The camera will always be mounted perpendicular to the floor, which means that one axis of the image plate will always be parallel to the scene. In this case, two-point perspective type is the standard of this project. One-point perspective can also exist, but one-point perspective can be considered as a special case of two-point

perspective in which the second point tends to go infinity. Briefly, one-point perspective is covered by two-point perspective and three-point perspective which will never be necessary for indoor navigation.

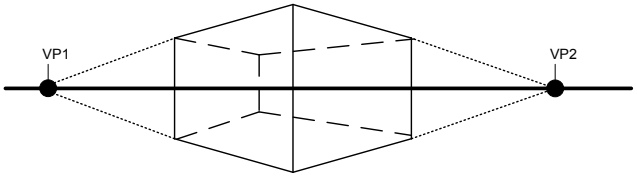

Fig. 7. Two-point perspective – The image plate is parallel to Cartesian scene in only one axis

Horizon Line. Horizon line is a line which does not practically exist. Horizontal line is a theoretically existing line which separates the earth from the sky. As long as there are no stairs and ascents on the floor, any point which is seen on the image and belongs to the floor, has to be below the horizon line.

7. Filtering Corridor Lines

7.1 Application of Two-Point Perspective Rules

This project involves filtering lines corridor lines through an indoor image. As mentioned before, two-point perspective comprehends all possible situations in this project.

In two-point perspective, vertical lines can never be corridor lines. Vertical lines can represent walls, columns etc. This information provides the elimination of so many candidate corridor edge lines. Additionally, identifying the position of the vanishing points can lead to much helpful information for indoor robotics.

7.2. Application of Horizon Line Rule

The definition of horizon line gives a clue about where the corridor lines must be looked for. Any point which is on the floor has to be below the horizon line. This means that a corridor line can only below the horizon line. According to this information, many lines can be eliminated with very low computational cost. An example of lines filtered through this rule is shown in figure 8 (left). It is worthwhile to give some information about how to locate the horizon line on an image. As mentioned before, the camera used for this project is mounted on a robot perpendicular to the floor which means that the image plate is always perpendicular to the actual scene. In this case the horizontal line is an infinite horizontal line on the image. The vertical position of the horizon line changes with the camera angle.

Fig. 8. Application of Horizon Line Rule (left) and Application of Landmark Rule (right)

7.2. Filtering Lines below Landmark Paths

A corridor line is the line which separates a wall or a column from the floor. Anything above a corridor line does not belong to the floor, and anything below a corridor line does not below to a wall or a column.

In this project a landmark is a point on the floor. According to the information above, a landmark cannot be above a corridor line. This means that any line which is under a landmark can be eliminated from being a corridor line. An example is shown in figure 8 (right).

8. Detection of Horizontal Gaps between Probable Corridor Lines

There are several reasons which cause gaps between filtered lines. Some of these reasons are:

1. The existence of walls or columns in the input image.
2. The existence of open doors in the input image.
3. The gap does not actually exist in the input image, but for some reason the line is not detected (technical reasons and imperfections mostly).

These are some basic reasons but not the only ones; there can be other reasons such as existence of objects or obstacles. Gaps caused by the three reasons listed above can be seen in figure 9 (left). The first two gaps (from left to right) are not actual gaps between corridor lines, the corresponding corridor line can be seen on the raw input image but it was not detected by line finding algorithms. The third gap is caused by an open door. The fourth gap is caused by a wall.

Fig. 9. Extracted corridor lines and certain horizontal gaps between the lines (left) and Treatment of Horizontal Gaps (right)

9. Construction of Floor Mask

The corridor lines are detected in this project in order to construct a floor mask from the input image. The horizontal gaps between probable corridor lines are detected in order to add these gaps to the floor mask. This floor mask is not the final opinion about drivable floor regions. The goal in creating the floor mask is including all drivable regions to the mask (figure 9/right). In other words, the portion of the image from which the floor mask is subtracted, should not include any drivable regions. So adding certain undrivable regions to the floor mask is an acceptable error. That is why the gaps between lines are included to the floor mask (figure 10).

Fig.10. **Floor mask created from corridor lines and horizontal gaps – the height of the gaps added to mask begin from the horizon line**

10. Intersection of Flood Filling and Floor Mask

10.1. Flood Fill Method

The Flood Fill Algorithm [12] is a point clustering method of points which have the similar color values between defined thresholds. In order to perform the Flood Fill algorithm on a specific image to find out the floor region one should determine a point on the floor surface. However, finding a point on the floor surface is not simple as it seems. Therefore, robot can retrieve the points by means of image processing techniques or information of a point on the floor should be given to robot somehow. In this case locating recognizable landmarks on the floor surface to feature remarkable regions is a technique that has been used generally. Flood Fill method can be applied around these landmarks.

Disadvantages of Flood Fill Method. **Regardless of the fact that flood fill is a reliable method for finding regions in an image by reason of its dependence on color similarities, method could come up with incorrect results. If the color space of the walls and the floor is close enough than the algorithm comes with most erroneous result.**

Intersecting Floor Mask Parts Using Flood Fill Method. In the used line filtering method, after constituting a wide floor mask and flood fill around landmarks, outcomes of these two processes are combined with each other. Importance of this combination is to overcome weaknesses of these two different methods. Disadvantage of one method could be turned into an advantage for other method. A flood fill output of an indoor scene is shown at the left side of figure 11 and the floor mask application is shown at the right side of the same figure. Clearly, outputs of these two methods are not sufficient enough. But the combination of these two methods is necessary to detect floor surface efficiently.

10.2. Estimating the vanishing points through filtered lines

After filtering the lines with using the methods which are mentioned in this paper, intersection point of the edge lines with vanishing line/skyline can be calculated. First approach should be clustering the intersection points on horizontal line. Then, first two clusters are chosen which have denser distribution than others. If the covariance between these two clusters is high enough to distinguish them from others, mean values of the clusters can be used as vanishing points in the perspective view.

The logic behind this process is as follows. Corridor lines are pointed to true vanishing points according to coherence with perspective lines. Additionally, furnitures that are located in perspectively appropriate way also provide the auxiliary directions to vanishing points. But perspectively inappropriate located lines will point to inaccurate vanishing points. However, in a corridor scene, number of perspectively correct lines will excess the wrong ones. Furthermore, probability of the inaccurate vanishing point clusters is extremely low due to randomness of inaccurate located lines.

Fig.11. Flood fill results (left) and floor mask (right) on corridor image

Advantages of Estimating the Vanishing Points. Some of the prominent advantages of estimating the vanishing points which are suitable for perspective view of the camera are:

1. Estimating the angle of the position through corridor
2. Estimating the angle of the position through walls, obstacles (relative position w.r.t. floor) etc.
3. Stating the mobile robot position
4. Possible corridor edge lines can be arranged with the consideration of filtering the points that do not converge to vanishing points to obtain more accurate floor maps.

11. Concluding Remarks

In this paper a drivable floor regions detection method for indoor robotics applications was presented. This technique is mostly based on corridor line extraction which is achieved by monocular visual perception. The lines are extracted with several techniques instead of a single one, and the circumstances are not considered as ideal. The key of this method is applying perspective rules on indoor scenes in order to gather information about the corridor lines. Usually an image processing of a single frame involves certain operations on every single pixel of the image, which are approximately three hundred thousand in total for an image of 640 pixels width and 480 pixels height. The key algorithms of this method involve operations on lines which are constituted by only two pixels and these lines are at most a few hundred in number. This means that these algorithms do not cause too much excess in computation time and the method is applicable for real-time. The results of this method can be improved with further filtering on corridor lines by the estimation of vanishing points.

Acknowledgements

This work was achieved in Robotics Laboratory at Istanbul Technical University which set sight on to develop mobile robot systems at present and supported by TUBITAK (Turkish Scientific and Technical Council) under Grant number: 107E007.

References

[1] S. Borthwick and H. Durrant-Whyte, "Simultaneous localization and map building for autonomous guided vehicles.", IROS, pp. 761–768, 1994.

[2] J.-S. Gutmann and C. Schlegel, "AMOS: comparison of scan matching approaches for self-localization in indoor env.," Proc. of Euromicro Workshop, pp. 61-67, 1996.

[3] J. J. Bardyn et al., "Une architecture VLSI pour un operateur de filtrage median," Congres reconnaissance des formes et intelligence artificielle, vol.1, pp. 557–566, Paris, 25–27 January 1984.

[4] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," In Proceedings of the Sixth Internatinal Conference on Computer Vision, pp. 839–846, 1998.

[5] D. Comaniciu and P. Meer, "Mean shift analysis and applications," IEEE International Conference on Computer Vision, vol. 2, p. 1197, 1999.

[6] I. Sobel and G. Feldman, "A 3 × 3 Isotropic Gradient Operator for Image Processing," In Pattern Classification and Scene Analysis, pp. 271–272, Wiley, 1973.

[7] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures, Communications of the Association for Computing Machinery 15,"(pp. 11–15, 1972.

[8] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic Hough transform," Computer Vision Image Understanding 78, pp. 119–137, 2000.

[9] H. Kalviainen et al., "Probabilistic and non-probabilistic Hough Transforms: overview and comparisons," Image and Vision Computing 13, pp. 239-252, 1995.

[10] Wikipedia article: "One Point perspective on Perspective (Graphical)",web site: http://en.wikipedia.org/wiki/One-point_perspective#One-point_perspective, last visited:06/07/2009.

[11] Wikipedia article: "Two Point perspective on Perspective (Graphical)",web site: http://en.wikipedia.org/wiki/Two-point_perspective#Two-point_perspective, last visited:06/07/2009.

[12] P. Heckbert, A Seed Fill Algorithm, Academic Press, New York, 1990.