Machine Vision and Applications (1996) 9:9–13

Machine Vision and Applications
© Springer-Verlag 1996

Matching free stereovision for detecting obstacles on a ground plane

Ming Xie

School of Mechanical & Production Engineering, Nanyang Technological University, Singapore 639798
e-mail: mmxie@ntuix.ntu.ac.sg

Abstract. This paper presents a method for detecting obstacles on a ground plane from a stereo pair of images. Although we use stereovision, the obstacle detection algorithm relies neither on stereo matching nor 3D reconstruction. The principle here is to apply the projective transformation constraining the left and right images to obtain a frame of superimposed features (e.g. edges). By analysing feature superimposition after the projective transformation, a free moving space or space occupied by obstacles/occluded features can be determined.

Key words: Stereovision – Mobile robot guidance – Obstacle detection – Projective transformation

1 Introduction

Obstacle detection on a ground plane is an important problem in computer vision and robotics (Masaki 1992; Sawhney and Hanson 1993; Slack 1993; Xie 1994). The efficient detection of ground-plane obstacles enables the practical guidance of a mobile robot or vehicle in an indoor environment with a flat floor (e.g., a factory) or an outdoor environment with man-made road (e.g. a motorway). In this paper, we present a solution for detecting an obstacle on a ground plane from a pair of stereo images. The basic idea of the solution is as follows. According to the geometric principle of 2D vision, there is a unique and invertible projective transformation matrix that relates the image plane to a 2D plane in the camera's 3D space. Suppose that we have stereo cameras observing a common plane (the ground plane). Then we can immediately derive an unique and invertible projective transformation matrix that relates the left image plane to the right image plane of the stereo cameras. By applying this projective transformation matrix, one can transform the geometric primitives (e.g. contour points) from the right image into the left image. If the geometric primitives are all inside the common plane observed by the stereo cameras, there will be no apparent disparity on the left image after the projective transformation. If geometric primitives are located either above and/or below the common plane, then there will be obvious apparent disparity in the left image after the projective transformation. By locating the region containing the apparent disparity in the left image, one can determine the space occupied by the obstacles. In our implementation, we use edge primitives (contour points). Hence, our method only involves computing the edge detection and the superimposed edge-map segmentation (i.e. we segment the edge maps in the left image after the projective transformation of the right edge map). Compared to the standard stereovision processing steps, it is obvious that our method is both fast and reliable, since stereo matching is not necessary and not included here.

2 Problem statement

Our application context is the development of a visual guidance system for a mobile robot or vehicle that moves on a ground plane. The two major tasks of a visual guidance system are: (a) the detection of road marks on the ground plane for the task of following the road and (b) the detection of obstacles above the ground plane for the task of avoiding the obstacles. Our concern is how to use stereovision efficiently to detect the presence of obstacles above the ground plane so as to provide useful information to the obstacle-avoidance module in our visual guidance system.

Figure 1a shows the stereo cameras mounted on a mobile robot. At each instant, the stereovision system outputs a stereo pair of images. Figure 1b and c shows one pair of stereo images that contain one obstacle seen by the stereo cameras. Now, the question is how to determine the free moving space of the mobile robot or the space occupied by the obstacles from a stereo pair of images.

To answer this question, we propose an efficient solution that involves neither stereo matching nor 3D reconstruction. The basic idea of the solution is to make use of the geometric principle of 2D vision. The major steps of the method are:

1. We determine the projective transformation matrix between the two image planes of the stereo cameras.
2. We extract the edge maps from the pair of stereo images.
3. We project the edge map of the right image onto the left image.

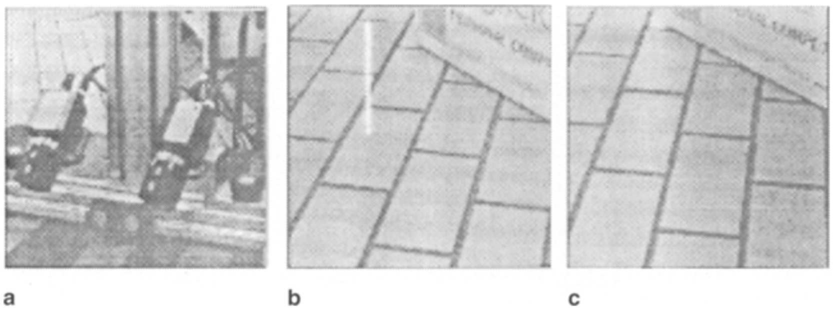


Fig. 1a–c. Detecting obstacles on a ground plane with stereovision: **a** a pair of stereo cameras mounted on a mobile robot; **b**, **c** one pair of stereo images

4. We segment the superimposed edge maps on the left image into two regions: (a) the region without apparent disparity and (b) the region with apparent disparity. The region without apparent disparity indicates the free moving space in which the mobile robot can travel. The region with apparent disparity indicates the space that contains obstacles or corresponds to the occluded space.

3 Projective transformation in stereo images

We model the image formation with a perspective projection. It is commonly known that there is a 3×4 projective transformation matrix that maps 3D object coordinates (X, Y, Z) in a world coordinate system $OXYZ$ onto the index coordinates (u, v) in the image plane; that is:

$$\begin{pmatrix} \rho u \\ \rho v \\ \rho \end{pmatrix} = M_{3\times4} \bullet \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \tag{1}$$

where $M_{3\times4}$ is the projective transformation matrix and ρ is a scaling factor (or homogeneous coordinate).

The definition of the world coordinate system $OXYZ$ is independent of the definition of the coordinate system ouv associated with the image plane. Suppose that there is a 2D plane $\Re^2$ in the camera's 3D space. We can define the world coordinate system $OXYZ$ in such a way that OXY is on the 2D plane $\Re^2$. Therefore, we know that the Z coordinates of the points inside the 2D plane $\Re^2$ are all equal to zero. From Eq. 1, the relationship between a point inside a 2D plane $\Re^2$ in OXY and its image in ouv can be described by:

$$\begin{pmatrix} \rho u \\ \rho v \\ \rho \end{pmatrix} = H_{3\times3} \bullet \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}, \tag{2}$$

where $H_{3\times3}$ is the 2D projective transformation matrix that relates the image plane to a 2D plane $\Re^2$ in the camera's 3D space. This 2D projective transformation is invertible because an image point can only be back-projected into one point on the 2D plane $\Re^2$ (if this plane is not perpendicular to the image plane at the image point).

Now, consider that we have two cameras (L, R) in two positions P_1 and P_2. In the position P_1, we have the 2D projective transformation matrix $L_{3\times3}$ which relates the image plane of the camera L to the 2D plane $\Re^2$. In position P_2, we have another 2D projective transformation matrix $R_{3\times3}$ that relates the image plane of the camera R to the same 2D plane $\Re^2$. Since the matrices $(L_{3\times3}, R_{3\times3})$ are invertible, we can determine a third projective transformation matrix $RL_{3\times3}$ as follows:

$$RL_{3\times3} = L_{3\times3} \bullet R_{3\times3}^{-1}. \tag{3}$$

In fact, the projective transformation matrix $RL_{3\times3}$ relates the image plane of the camera R to the image plane of the camera L, constrained by the 2D plane $\Re^2$ (in a 3D world); that is:

$$\begin{pmatrix} \rho u_l \\ \rho v_l \\ \rho \end{pmatrix} = RL_{3\times3} \bullet \begin{pmatrix} u_r \\ v_r \\ 1 \end{pmatrix}, \tag{4}$$

where (u_l, v_l) are image coordinates in the camera L and (u_r, v_r) are image coordinates in the camera R.

So far, we can highlight the following interesting points of the projective transformation matrix $RL_{3\times3}$:

- The matrix $RL_{3\times3}$ can be determined from four pairs of matched points. This opens the possibility of automatic on-line calibration. However, on-line calibration relies on the stereo matching of at least four pairs of points. In our study, we do off-line calibration to obtain the projective transformation matrix $RL_{3\times3}$.
- The matrix $RL_{3\times3}$ can be a very useful geometric constraint for the verification of correctness of region-based stereo (or motion) matching. Usually, an image region corresponds to a planar surface patch in a 3D space. If two regions (from two different images) match, then a projective transformation matrix $RL_{3\times3}$ must exist, and it projects one region onto the other. The projection must form a perfect superposition.
- If one point is inside a 2D plane in the camera's 3D space, and this point is visible to the pair of stereo cameras, then the right image of this point can be projected onto the left image by the projective transformation matrix $RL_{3\times3}$. After the projection, the two images must be superposed perfectly. In contrast, if one point is located outside the 2D plane, then the projective transformation by $RL_{3\times3}$ will absolutely not superimpose the right image on the left image of the point.

The last point constitutes a necessary condition for detecting an osbtacle on a ground plane:

Necessary condition. If a geometric primitive is outside a 2D plane and is observed by a pair of CCD cameras (left camera and right camera), the geometric primitive will not be superimposed in the left camera after transforming the geometric primitive from right camera into left camera by a 2D projective transformation matrix.

4 Implementation

The implementation of our algorithm for detecting obstacles on a ground plane is simple due to the absence of a stereo matching process. The major steps of our algorithm are:

1. Obtain the projective transformation matrix $RL_{3\times 3}$ by off-line calibration.
2. Extract the edge maps from the pair of stereo images by using the Canny-Deriche edge detector (Deriche 1987). (This detector is one of the most reliable edge detectors. We have not encountered any problem in using it for real images).
3. Project the edge map of the right image onto the left image by using the matrix $RL_{3\times 3}$.
4. Segment the two superimposed edge maps into two sets: (a) the subset of superimposed contour points and (b) the subset of nonsuperimposed contour points.
5. Delete false superimposed contour points.
6. Determine the common boundary between the two subsets of contour points.

In the following, we explain steps 4–6 in detail. By applying the projective transformation $RL_{3\times 3}$, we project the edge map of the right image onto the left image. Then, we use a neighborhood window of the size of 5×5 (if the calibration is very precise and there is no optical distorsion of the lenses, one may choose a size of 3×3. The selection of the size 5×5 tolerates error due to calibration, optical distorsion, vibration of a mobile robot, etc.), and we can separate the edge map into two sets: (1) set A, the subset of superimposed contour points and (2) set B, the subset of nonsuperimposed contour points. A contour point in the left edge map is considered superimposed if there is at least one contour point inside the neighborhood window and this contour point belongs to the right edge map. Set A indicates the free moving space (i.e. obstacle free space). Set B indicates the space occupied by the visible obstacles or the occluded space. Here, by "occluded space" we mean the space that can be seen by one camera, but is not visible to the other camera.

Due to the complexity of the scene and the fact that Eq. 3 provides only a necessary condition, the contour points in set A may not all correspond to the geometric primitives inside the ground plane. For example, in Fig. 2d, the printed characters on the box introduce some isolated and superimposed contour points to set A, which we must eliminate. Fortunately, they can easily be eliminated by contour tracing. Contour tracing means grouping the contour points in set A into a set of linked contour chains. Then, we delete all the contour chains shorter than 15 pixels. All the deleted contour points in set A are added to set B.

It must be noted that the deletion of false superimposed contour points in set A can be further enhanced by taking into account the intensity information on the contour points. If the intensity values on two superimposed contour points are not similar, this must be a false superimposition.

As for the determination of the boundary separating set A and set B, one straightforward solution is first to determine the polygonal envelopes of set A and set B. If we denote as EA the polygonal envelope of set A, and as EB, the polygonal envelope of set B, we can safely declare that (EA–EB) is the free moving space, and the rest is the space occupied by the obstacles or the occluded space. The drawback of this solution is the high cost in computing time needed for the determination of the polygonal envelopes. We propose a simple and fast solution to determine the apparent (not the exact) boundary between set A and set B. This solution is valid in the sense that the information about the presence or absence of obstacles will be used to guide a mobile robot or vehicle, which is a dynamic process. At any one instant, it is not necessary to know the precise boundary of the free moving space. The apparent boundary between set A and set B is precise enough for the steering control module of a mobile robot or vehicle. The proposed solution is:

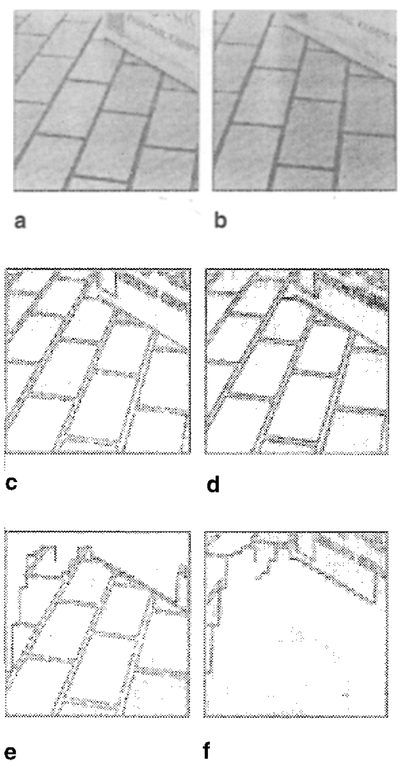
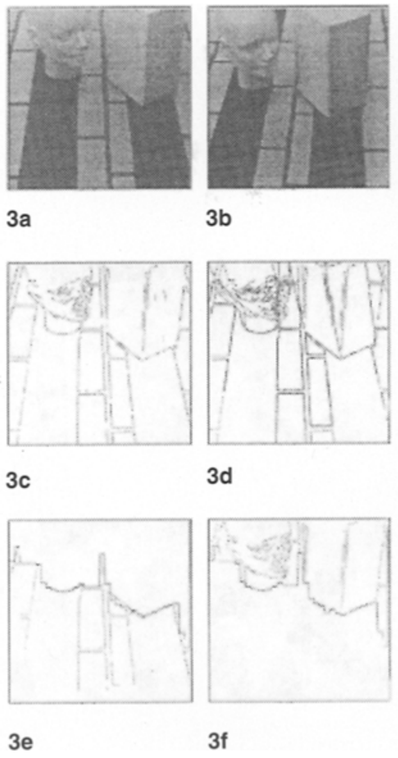
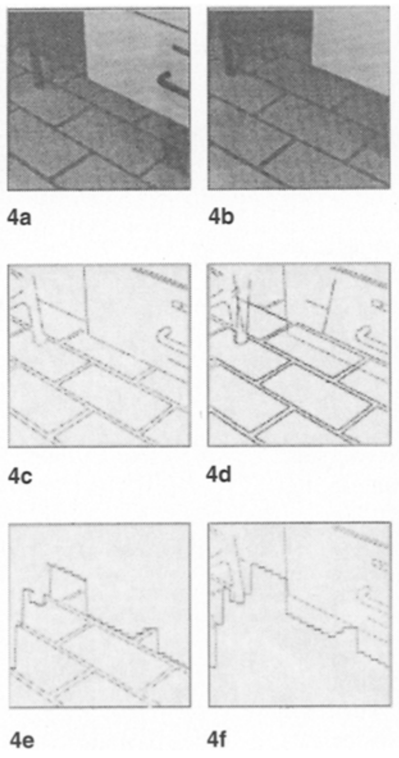


Fig. 2a–f. Example 1: **a** the left image; **b** the right image; **c** the left edge map; **d** the left edge map superposed with the right edge map projected onto the left image; **e**, **f** show the boundary separating the free space and the space occupied by obstacles (or the occluded space)

- Determine the (top) staircase boundary (SBA) of set A. Consider that the edge map is a 2D discrete space. The positive X direction is from left to right and the positive Y direction from top to bottom. If (rx, ry) are the edge map's resolutions in the X and Y directions, we divide the edge map space into n regions $\{R_i, i = 1, ..., n\}$. Each region is of the size of $lx \times ry$ where $n * lx = rx$ (in our implementation, $lx = 16$). For each region R_i, we associate a horizontal line L_i of the length of lx to it. The initial position of the line L_i is at the top edge of the region R_i. Then, we move down the line L_i along the vertical edges of the region R_i until it encounters a contour point in set A and stops the line at the position of the contour point. If there is no contour point in the region R_i, the final position of the line L_i is at the top edge of the region R_i. After moving all the lines in





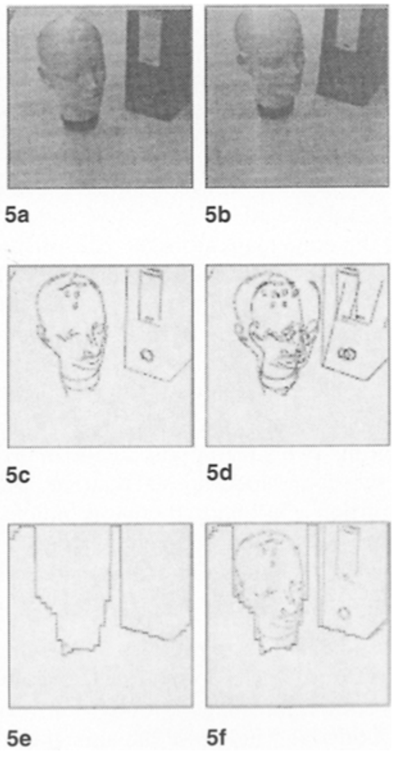


Fig. 3a–f. Example 2: **a** the left image; **b** the right image; **c** the left edge map; **d** the left edge map superposed with the right edge map projected onto the left image; **e, f** show the boundary separating the free space and the space occupied by obstacles (or the occluded space)

Fig. 4a–f. Example 3: **a** the left image; **b** the right image; **c** the left edge map; **d** the left edge map superposed with the right edge map projected onto the left image; **e, f** show the boundary separating the free space and the space occupied by obstacles (or the occluded space)

Fig. 5a–f. Example 4: **a** the left image; **b** the right image; **c** the left edge map; **d** the left edge map superposed with the right edge map projected onto the left image; **e, f** show the boundary separating the free space and the space occupied by obstacles (or the occluded space)

$\{L_i, i = 1, ..., n\}$ to their respective final positions, we create the staircase boundary (SBA) of set A; that is:

SBA = $\{L_i, i = 1, ..., n\}$.

- Determine the (bottom) staircase boundary (SBB) of set B.
  Similarly, for each region R_i, we associate another horizontal line S_i of the length of lx to it. The initial position of the line S_i is at the bottom edge of the region R_i. Then, we move up the line S_i along the vertical edges of the region R_i until it encounters a contour point in set B and stops the line at the position of the contour point. If there is no contour point in the region R_i, the final position of the line S_i is at the top edge of the region R_i. After moving all the lines in $\{S_i, i = 1, ..., n\}$ to their respective final positions, we create the staircase boundary (SBB) of set B; that is:

  SBB = $\{S_i, i = 1, ..., n\}$.

From SBA and SBB, the apparent boundary (AB) between set A and set B is calculated by taking the maximum part (in terms of Y coordinate values) of SBA and SBB, that is:

$$AB = \{\ max_y(L_i, S_i),\ i = 1,\ ...,\ n\ \}.$$

5 Experimental results

We have tested our algorithm with real images of various situations. We show four examples of experimental results. Now we give brief comments on these four examples.

- Figure 2 presents the first example in which there is only one obstacle. The box is a well-structured object. However, the printed characters may create complications for image-based obstacle-detection algorithms. Our algorithm works well in the presence of this kind of irrelevant information. From Fig. 2f, one can see that the algorithm can also detect occluded space (the left part in Fig. 2f).
- Figure 3 presents the second example in which there are two obstacles. The head model simulates a natural and nonstructured object. We can see that the algorithm can determine well the free moving space and the space occupied by the obstacles.
- Figure 4 presents the third example. This example has also two obstacles. One is small and on the ground (it is a leg of a chair). The other is large and does not touch the ground (it is the partial view of a drawer). This proves that the algorithm is also capable of detecting the presence of visible obstacles that may not contact the ground plane.
- Figure 5 is similar to example 2, except that the ground plane has no texture information. This was done to an-

swer the possible question about whether the algorithm still works in the absence of the texture information on the ground plane. We can say that the algorithm works in this case. This is because we do not make any assumptions about the texture information on a ground plane.

In these experiments, the lenses of the stereo cameras are focused on an area of about 0.8 m$\times$1.4 m on the ground. This area is about 1.5 m away from the front side of the robot. The image size is 512×512. The head models in examples 2 and 4 are about 0.1 m $\times$ 0.1 m $\times$ 0.3 m. For the questions of what is the smallest detectable obstacle and what is the maximum range of detection, one indirect answer is that it mainly depends on the parameters of optical lens (i.e. open angles in X and Y directions), the viewing direction of cameras and the resolution of the image. All these factors are under the control of user.

6 Conclusions

A simple, fast and reliable algorithm for detecting obstacles on a ground plane has been presented in this paper. This algorithm makes use of the geometric principle of 2D vision and takes input from a stereovision system. This method does relies neither on stereo matching nor 3D reconstruction. Instead, the problem of detecting an obstacle on a ground plane is solved by segmenting superposed edge maps after a 2D projective transformation of the right edge map. Experiments with real images prove the validity and efficiency of the method.

Acknowledgements. This work is supported by the Ministry of Education (Singapore) under the grant 4-25003 (ARC 1/94).

References

Deriche R (1987) Using Canny's criteria to derive a recursively implemented optimal edge detector. Int J Comput Vision 1:167–187

Masaki I (1992) Vision-based vehicle guidance. Springer, Berlin

Sawhney HS, Hanson AR (1993) Trackability as a cue for potential obstacle identification in 3-D description. Int J Comput Vision 11:237–265

Slack MG (1993) Navigation templates: mediating qualitative guidance and quantitative control in mobile robots. IEEE Trans Syst Man Cybernetics 23:452–466

Xie M, Trassoudaine L, Alizon J, Gallice J (1994) Road obstacle detection and tracking by an active and intelligent sensing strategy. Machine Vision Appl 7:165–177

Ming Xie received his BEng degree in Process Automation in 1984 from the University of China Textile, China. Subsequently, he obtained his DEA (Equiv. MEng) degree in Industrial Automation in 1986 from the University of Valenciennes, France and his PhD degree in Computer Science in 1989 from INRIA Rennes and University I of Rennes, France. In February 1993, he joined Nanyang Technological University, Singapore, and is now a lecturer in the school of Mechanical & Production Engineering. His research interests include visual inspection, intelligent machine vision, vehicle/robot guidance, object recognition, parallel computing and computer graphics. He is a member of IEEE