Stereo Image to Graphics Conversion for Indoor Environments

Vishwmitra S. Bhadouria, Disha Prakash, and Venkatesh K. Subramanian
Indian Institute of Technology, Kanpur, India
Email: {vsb2210, dishaprakashk}@gmail.com, venkats@iitk.ac.in

***Abstract*—The paper presents an efficient algorithm for approximating a real world 3D scene captured by a camera by a computer generated 3D graphics environment. First, to extract the 3D edge from an available 2D stereo image pair, straight edges are detected from both the images and a correspondence is established between them through the fundamental matrix based matching scheme. In the next step, we triangulate the end points of the common region of this matched edge pair to find the end points of a 3D straight edge. Using these, we locate the existence of planes between coplanar 3D lines, devising a coplanarity score for pairs of 3D lines and finally eliminating non-physical 3D planes by SIFT based analysis on the basis of discrepancies in depth of SIFT features in those 3D planes.**

***Index Terms*—image to graphics conversion, 3D edges, pseudo-plane, epipolar**

I. Introduction

The background of our work is the field of 3D computer graphics which concerns with approximating a real 3D scene captured by a camera by computer generated 3D models. Most prevalent methods use depth maps for evaluation of 3D planes from a stereo image pair. They rely on extracting corresponding image points from the given stereo image pair and triangulating them for estimation of depth (as well as the other two coordinates) of 3D points. We have used straight edge feature matching instead of point-wise matching and establish point correspondences based on edge correspondences for depth estimation. The reason for our choice is that straight edges are easily available in man-made environments, more robust against geometric distortions, incomplete matching and occlusion, and their identification is easier due to contrast change and hence offers robust matching.

Our approach to straight line detection is a slightly extended version of fast straight line detection [1] which finds the presence of all possible straight lines between pairs of corners (found by Rostens fast feature detector) and accepts its presence if more than ninety five percent of the pixels of these lines lie on the edges. Now matching straight lines across a stereo image pair with significant base line difference between two cameras is difficult because of inherent deficiencies in the straight line extraction process, which result in unreliable endpoints, edge lengths as well as edge centers. Prevailing line matching approaches (based on number of line segments they consider at a time) can be grouped in two categories. The first approach [2], [3] matches straight lines in a stereo image pair by considering one pair of lines at a time and finding match based on similarities in their geometric characteristic such as orientation, length etc. Employing minimum distance, orientation and length together is a very well suited approach for very small base line stereo pairs or for image tracking purposes where a single camera is employed and extracted images (at consecutive time instants) are almost similar. The second approach matches a bunch of straight lines to a similar bunch in the other image and offers an obvious advantage of better disambiguating capabilities by exploiting topological relationships such as left of, collinear with etc at the cost of increased complexity [4], [5]. Paper [6] introduces a fundamental matrix based line matching technique which exploits the epipolar constraint between two views and calculates a correlation based matching score for finding the straight line correspondences. This technique is also applicable to images taken from a stereo camera pair with a significant base line. We found this as the most suitable approach for our work. For 3D reconstruction using straight lines [7] describes the geometric limitations of line based 3D reconstruction algorithms. [8] introduces a multiple high-resolution image based model. Another method, [9] introduces a plane-sweep based 3D reconstruction based on identifying planar surfaces using single 3D lines and inter-image homographies.

However, correlation score calculation over multiple views for identification of valid planes from the set of all possible planes through a particular 3D line is quite expensive in terms of computation time. We have used only two views for evaluation of the 3D structure of the scene and for disambiguating physically existing planes from pseudo planes, we use 3D SIFT points (calculated by triangulation of matched 2D SIFT point pairs in a stereo image pair).

II. Line Detection and Matching

A. Straight Edge Detection

We first perform Canny edge detection to extract binary edge images from both the input stereo images. Now to detect straight edges from these binary edge images we create a visited point image (initially

Manuscript received February 15, 2014; revised May 9, 2014.

doi: 10.12720/joig.2.1.83-88

containing only zeroes as all pixels are as yet unvisited: 1 represents a visited point and 0 represents non-visited point). If a particular pixel of the Canny image is found to be non-zero as well as unvisited (found by checking visited point image at this particular pixel), select a point on the perimeter of a $(2L+1)\times(2L+1)$ square centered around the pixel under test.

Thus there are $8L$ perimeter pixels. Consider this pixel under test and any chosen point on the perimeter as two end points of a hypothetical line. Whether or not this hypothesis is valid is now determined by counting the number of unvisited Canny edge pixels in it. If the total number of such pixels is greater than or equal to a threshold (set as $0.9L$ in our work), line hypothesis is considered as valid, and the corresponding end points are stored and the points on this line are set as visited in the visited point image. The above steps are repeated for all the points on the perimeter of the square. Thus all hypothetical lines joining the perimeter pixels of the $(2L+1)\times(2L+1)$ square (see Fig. 1) with the pixel under consideration (the centre of the square) are checked. The above procedure is repeated for all the pixels of the Canny image from top left to bottom right in a sequential manner. Detecting edge segments in this way by keeping a threshold takes care of possible missing and/or misplaced pixels in an edge segment and thus helps in dealing with noisy edges. At this point, we have small straight edge segments of L pixels (L was set to 8).

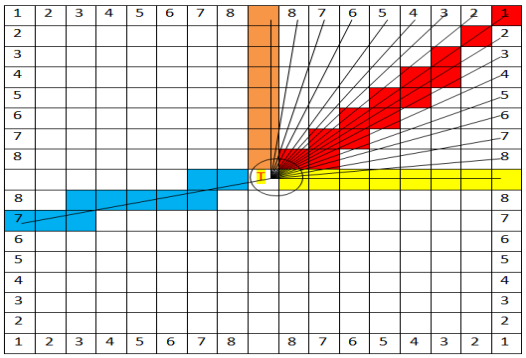


Figure 1. 17×17 square search region around test pixel ($L=8$)

Now by looking for neighboring collinear segments, we merge these edge segments to get longer edges. Each straight edge segment is tested with all the other edge segments for similarity of slope. There may never be exact equality of slope, so a threshold is again used. Consider an edge segment with end points P_1, P_3 and another edge segment with end points P_2, P_4. Then slopes m_1 and m_2 are given as

$$m_1 = \frac{P_{3y}-P_{1y}}{P_{3x}-P_{1x}} \tag{1}$$

And

$$m_2 = \frac{P_{4y}-P_{2y}}{P_{4x}-P_{2x}}. \tag{2}$$

Thresholding criteria used is

$$abs\left(\tan^{-1}\frac{m_1-m_2}{1+m_1m_2}\right) < 2\ deg \tag{3}$$

Find the nearest and farthest of these four points. Let P_1, P_2 be the nearest and P_3, P_4 be the farthest points.

If distance between nearest points is less than specified threshold (set as 3 pixels in our work) a hypothetical line (see Fig. 2) is assumed between farthest points (i.e. between P_3, P_4) and minimum distances of both nearest points with this hypothetical segment (say d_1, d_2) is calculated.

$$d_1 = \left|\frac{(P_{4x}-P_{3x})(P_{3y}-P_{1y})-(P_{3x}-P_{1x})(P_{4y}-P_{3y})}{\sqrt{(P_{4x}-P_{3x})^2+(P_{4y}-P_{3y})^2}}\right| \tag{4}$$

$$d_2 = \left|\frac{(P_{4x}-P_{3x})(P_{3y}-P_{2y})-(P_{3x}-P_{2x})(P_{4y}-P_{3y})}{\sqrt{(P_{4x}-P_{3x})^2+(P_{4y}-P_{3y})^2}}\right| \tag{5}$$

If d_1 and d_2 both are below a certain threshold, (set as 2 pixels in our work) then the hypothetical line segment is considered as a valid line segment and both smaller segments are replaced by it.

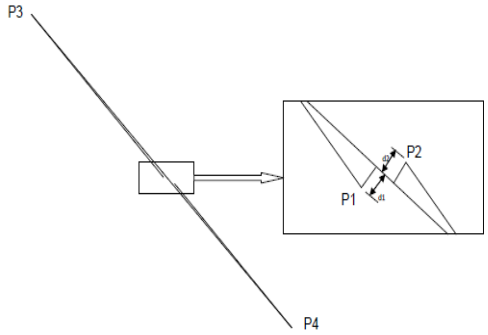


Figure 2. Checking the possibility of merging.

B. *Fundamental Matrix Based Feature Matching*

Calculation of fundamental matrix: Let the camera projection matrices of the left and right cameras be given by

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} \tag{6}$$

And

$$Q = \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} & Q_{14} \\ Q_{21} & Q_{22} & Q_{23} & Q_{24} \\ Q_{31} & Q_{32} & Q_{33} & Q_{34} \end{bmatrix} \tag{7}$$

Then the fundamental matrix F (calculated from camera projection matrices as in [10]) is given by

$$F = \begin{bmatrix} |M_{11}| & |M_{12}| & |M_{13}| \\ |M_{21}| & |M_{22}| & |M_{23}| \\ |M_{31}| & |M_{32}| & |M_{33}| \end{bmatrix} \tag{8}$$

Epipolar constraint: Let M_P and M_Q be the homogeneous 3×1 column vector representations of 2D points m_P and m_Q, which in turn are projections of a 3D point M on the first camera image plane and second camera image plane respectively. m_Q is then constrained to lie on a certain line in the image plane of the second

camera. If the equation of such a line is $ax + by + c = 0$ then a, b and c are given by

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = FM_P \quad (9)$$

This line is called the epipolar line of m_P where F is 3×3 fundamental matrix. Since m_Q lies on this line,

$$[M_Q]^T[F][M_P] = 0 \quad (10)$$

The above equation is called the epipolar constraint [11] and is vital in solving correspondence problems.

Matching scheme: Here, we establish a one to one correspondence between the members of two sets of straight line segments detected earlier. As we know, the images under consideration (say Image ***L*** and Image ***R***) are un-calibrated stereo images with possibly a considerable baseline difference between the two cameras, end points of detected straight lines, their lengths, and even centers of edges are unreliable. This is why, we choose the fundamental matrix based matching scheme for line matching. Thus, we have to begin with, two images containing many as yet, unmatched straight lines and a fundamental matrix using which a point in one image can be mapped to its corresponding epipolar line in the other image. That epipolar line is, in turn, bound to contain the corresponding point of the original point selected in the first image.

Let $\{L_1, L_2, L_3, \dots, L_m\}$ be the m straight edges found in first image and let $\{R_1, R_2, R_3, \dots, R_n\}$ be the n straight edges found in second image.

- First, we select the first point $m_1 = (L_{1x1}, L_{1y1})$ lying on straight line L_1 (the first subscript is used for particular edge number, second subscript describes the coordinate and third one is used for describing a particular point index).
- For the selection of a particular point on line, we check if the line slope is more towards *x*-axis or *y*-axis (i.e. we see whether the line forms a smaller angle with the x-axis or the y-axis).
- For a line inclined more towards *x*-axis, *x* is incremented (or decremented depending on whether or not the *x* coordinate of the end point is greater than that of the start point of the line segment under consideration) in steps of one pixel at a time (starting from the start point of the edge segment) and the corresponding *y* coordinate is computed by using

$$y = \frac{y_2 - y_1}{x_2 - x_1}(i) + y_1 \quad (11)$$

where i is for i^{th} point and $(x_1, y_1), (x_2, y_2)$ are end points of line. i ranges from x_1 to x_2.

- For a line inclined more towards the *y* axis, *y* is incremented in steps of one pixel at a time and the corresponding *x* coordinate is computed by using

$$x = \frac{x_2 - x_1}{y_2 - y_1}(i) + x_1 \quad (12)$$

where i ranges from y_1 to y_2.

Now after selecting the point on the line, we perform the following steps for matching the straight lines.

- We find the corresponding epipolar line parameters (a, b *and* c in equation $ax + by + c = 0$) lying in Image ***R*** by using (9). Denote this line by E_1^1.
- Find the intersection point (say (R_x, R_y) of line E_1^1 with line R_1 (first edge in second image).
- If $(\mathrm{R_x}, \mathrm{R_y})$ lies within edge segment R_1, compute the matching score S for points $m_1 = (L_{1x1}, L_{1y1})$ and $m_2 = (R_x, R_y)$ in the $o \times p$ neighbourhood (with $o = p = 7$ in our work) using [12] following equation:

$$S(m_1, m_2) = \frac{\sum_{i=-p}^{p}\sum_{j=-o}^{o}[I_1(L_{1x1}+i, L_{1y1}+j) - \mu_1(I_1)][I_2(R_x+i, R_y+j) - \mu_2(I_2)]}{(2p+1)(2o+1)\sqrt{\sigma_1^2(I_1) \times \sigma_2^2(I_2)}} \quad (13)$$

where

$$\mu_1(I_1) = \frac{\sum_{i=-p}^{p}\sum_{j=-o}^{o} I_1(L_{1x1}+i, L_{1y1}+j)}{(2p+1)(2o+1)} \quad (14)$$

$$\mu_2(I_2) = \frac{\sum_{i=-p}^{p}\sum_{j=-o}^{o} I_2(R_x+i, R_y+j)}{(2p+1)(2o+1)} \quad (15)$$

$$\sigma_1(I_1) = \sqrt{\frac{\sum_{i=-p}^{p}\sum_{j=-o}^{o} I_1^2(L_{1x1}+i, L_{1y1}+j)}{(2p+1)(2o+1)} - \mu_1^2(I_1)} \quad (16)$$

$$\sigma_2(I_2) = \sqrt{\frac{\sum_{i=-p}^{p}\sum_{j=-o}^{o} I_2^2(R_x+i, R_y+j)}{(2p+1)(2o+1)} - \mu_2^2(I_2)} \quad (17)$$

- If the matching score S is greater than some pre-specified threshold (set as 0.8 in our work) increment the score counter corresponding to edge R_1 by one. If the score is not greater than the threshold, the score is not considered (treated as zero) and the counter is not incremented.
- Repeat the above steps for all the other right image edges (i.e.$\{R_2, R_3, \dots R_n\}$). At this point we will have n score counters.
- Repeat the above steps for all the other points on edge L_1. Find the maximum value in array of score counters.
- If the maximum value is greater than the threshold, store the right image edge corresponding to the score counter holding the maximum value as the matched edge. Repeat all the above steps for $\{L_2, L_3, \dots . L_m\}$.

Thus we have accomplished left to right image edge matching. For better accuracy, right to left image edge matching is also performed (but now, fundamental matrix will be replaced by its transpose) and only edges with mutual agreement both ways are considered as matched edge pairs.

III. 3D Entities from Matched 2D Line Pairs

We first describe the algorithm for finding 3D edges from matched 2D line pairs, followed by the algorithm for identifying the planes in the 3D scene.

A. 3D Lines from Matched 2D Line Pairs

We need to establish an endpoint wise correspondence between the two corresponding 2D line pairs for calculating the endpoints of a 3D line.

First, a 2D straight edge (L) lying in first image (Image $\boldsymbol{L}$) is selected from a particular matched edge pair. End points of the selected 2D straight edge are converted into homogeneous form by adding a third coordinate "1" to them. Then these homogeneous end points are pre-multiplied by the fundamental matrix F to estimate the parameters of the corresponding epipolar lines in second image. Now, since we have the epipolar lines (corresponding to both the end points) in Image $\boldsymbol{R}$, we find their intersection with its matched line R. These respective intersection points (lying in the second image) are corresponding points of the end points of line L. Matching scores of the first end point with its corresponding point and that of second end point with its corresponding point are now calculated. Since we are going to triangulate these end point pairs, which are extremely sensitive to pixel matching error, we evaluate the matching scores of point pairs, and if the matching score is less than 0.95, we select the neighboring point of the previously chosen endpoint and start over. If the matching score is found greater than 0.95, we triangulate end points pairs to calculate end points of the 3D edge in space that is represented by this pair of matched image lines L and R.

In an ideal scenario, when both the image points and both the camera matrices are measured accurately, we could have used the inhomogeneous method of triangulation [13] (which presumes the scale factor as 1). However due to errors in the measurement process, back-projected rays do not intersect each other and this method fails to give an accurate estimate of the 3D point, or even may give no point at all. However, to find the best solution for the 3D point in this practical scenario (when measured image points are inaccurate, camera matrices P and Q are assumed to be calculated accurately), we use minimization of geometric re-projection error [14] as a preprocessing block and its output serves as the input to the inhomogeneous triangulation algorithm.

3D edge descriptor: The descriptor (see Fig. 3) used to describe a 3D edge, stores all the details of 3D edge as follows:

- First end point of 3D edge, which in turn is a structure containing the corresponding points from which it is calculated.
- Second end point of 3D edge, which has similar construction as first end point.
- Pointer to array containing co-planarity scores with all the other 3D edges.

Removal of very small edges: Edges formed by too small 2D lines are likely to give wrong results, because they are more susceptible to re-projection error caused due to pixel mismatch. Moreover, the effect of quantization error generated due to analog to digital conversion of image is likely to be more dominant for these edges. Hence, 3D edges whose 2D endpoints are less than 15 pixels apart are rejected in further processing.

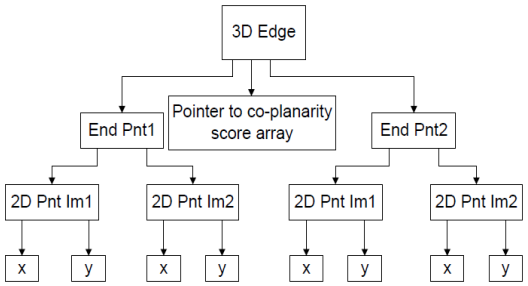


Figure 3. 3D edge descriptor (where Im 1 and Im 2 refers to Image $\boldsymbol{L}$ and Image $\boldsymbol{R}$ respectively)

B. Calculation of Coplanarity Scores

For identification of valid planes between any 3D edge pair, their endpoints can be checked for planarity. In an ideal scenario, the equation of the plane passing through any three endpoints can be formed, and planarity of 3D edge pair can be accepted or rejected depending upon whether or not the fourth endpoint satisfies this plane equation. However, end points of 3D edges obtained by triangulation of endpoints of the common portion of a pair of 2D edges have some noise added during analog to digital conversion. Besides, this there are inherent slight inaccuracies involved in the determination of the projection matrices, so we suggest the following method:

- Find the least square error plane for the four endpoints of the edge pair under investigation.
- Find the minimum distances of this plane from all the four endpoints.
- Find the largest value of these four minimum distances. This is termed as the co-planarity score in our work.
- If the co-planarity score for an edge pair is less than the threshold, it is considered as co-planar.

C. Pseudo Plane Removal

At this point, we have a (Number of 3D edges) × (Number of 3D edges) binary "Planarity matrix", with a '1' at location (i, j) indicating the presence of a plane between the 3D edge pair (i, j) and a 0 at location (i, j) representing absence of such a plane between the 3D edge pair (i, j). But this set might still contain pseudo planes, which do not exist physically in space, such as a doorway instead of a door. We collect matched point pairs using SIFT for separating valid planes from these pseudo planes.

D. Obtaining Matching Points Using SIFT

Application of SIFT for finding matching points in both the images is as follows:

- SIFT is applied on both images to get two sets of point features.
- These feature sets are matched using a FLANN based matcher which finds the minimally distant feature pairs corresponding to two images. Overall minimum distance of a feature pair is also calculated.
- This overall minimum distance is multiplied by a fixed constant (>1) to compute the threshold distance.

- Now distances of all the feature pairs are compared with threshold distance, and key points of those feature pairs whose distance is less than the threshold distance are considered as matched point pairs and stored in an array for further use.

E. SIFT Matching Points for Pseudo Plane Removal

At this stage, we have a few matched points in both the images and we can triangulate them to get the corresponding 3D points. These 3D points can then be used for validation of a plane that is believed to exist between a pair of 3D edges.

The algorithm for removing pseudo planes from the set of detected planes is as follows:

- Select a 3D edge pair (i, j) having a possible plane between them taken from a '1' at (i, j) in the Planarity matrix.
- Find the first image SIFT points (from the set of matched 2D SIFT point pairs) lying inside quadrilateral formed by endpoints of projections of 3D edge pair (i, j) into first image plane (i.e. corresponding lines of edge pair (i, j) in first image).
- Triangulate these first image SIFT points with corresponding SIFT points of second image to get a set of 3D points. Store these 3D points in an array (call it checkpoint array).
- Check for other 3D edges whose projections lie in the quadrilateral formed by endpoints of projections of 3D edge pair (i, j) into first image plane.
- Store endpoints of these 3D edges also in checkpoint array. This is done because we may not get SIFT points for few edge pairs, and moreover these 3D endpoints are calculated on the basis of more than 95% matching score and hence are supposed to be accurate
- Obtain the equation of the plane passing through the endpoints of edge pair (i, j).
- Calculate the number of elements of checkpoint array (which contains the 3D SIFT points as well as endpoints of edges lying between edge pair (i, j)) which agree with the equation of plane. Here also due to digitization of the image, points do not exactly satisfy the equation of plane so a threshold is used.
- If more than 50% of the total number of elements of the checkpoint array do not satisfy the plane equation, the plane hypothesis is rejected by overwriting the location (i, j)of intersection array with a 0.
- The above steps are repeated for all the edge pairs scoring a '1' at (i, j) in the Planarity matrix.

IV. Results and Discussion

We present here two datasets. The input stereo image pair for both the datasets has been shown first, followed by the outputs at various stages during straight line detection and matching. For the Canny edge detector, we select upper and lower thresholds at 100.0 and 10.0 respectively and an aperture size of 3. The minimum length threshold for straight lines is set to 14.

A. Dataset 1

Fig. 4 shows the input stereo image pair. This stereo pair is then subjected to Canny edge detection (see Fig. 5(a)). Fig. 5(b) shows 186 straight lines detected in Image 1 and 198 lines in Image 2 and the matched lines after performing fundamental matrix based line matching have been shown in Fig. 5(c).

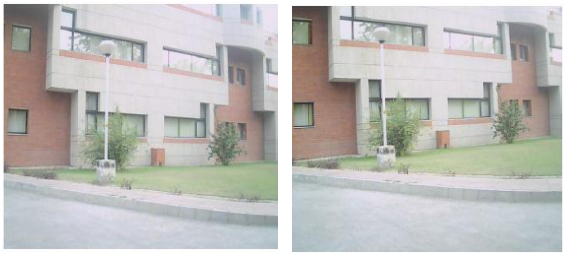
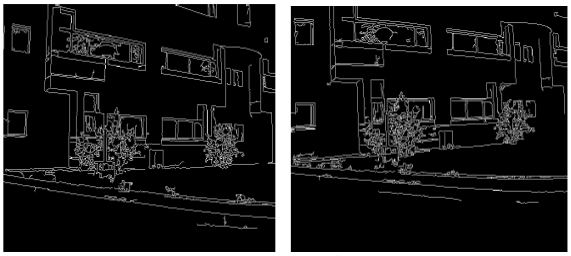
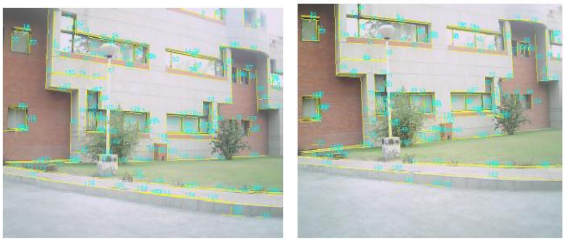


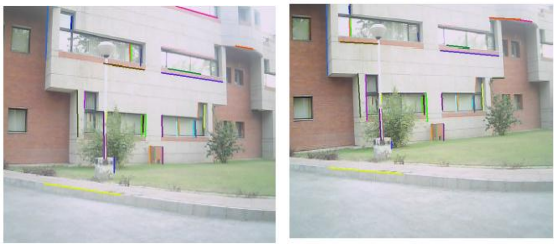
Figure 4. Stereo image pair (Image 1 and Image 2)



(a) Canny edge detector output for stereo image pair



(b) Detected straight lines



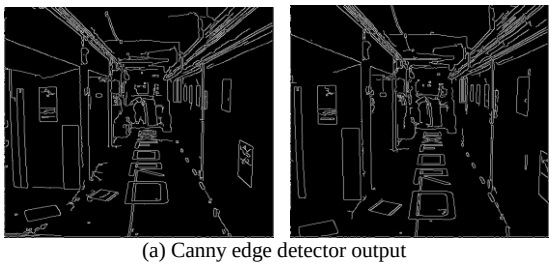
(c) Matched lines between Image 1 and Image 2

Figure 5. Output of various stages during straight line detection and matching (dataset 1).

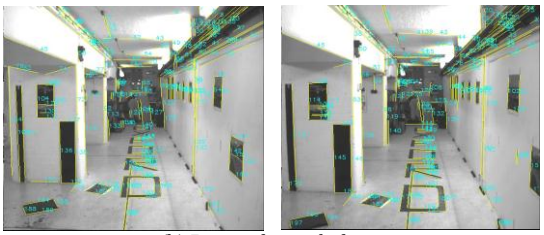
B. Dataset 2

Fig. 6 shows the input stereo image pair (taken from the University of Oxford website). The Canny edge detector output has been shown in Fig. 7(a). Fig. 7(b) shows 190 straight lines detected in Image 1 and 198 lines in Image 2 and the matched lines after performing fundamental matrix based line matching have been shown in Fig. 7(c).

Figure 6. Stereo image pair (Image 1 and Image 2)



(a) Canny edge detector output



(b) Detected straight lines

(c) Matched lines between Image 1 and Image 2

Figure 7. Output of various stages during straight line detection and matching (dataset 2).

V. Conclusion

We have developed and implemented a novel approach for 3D plane estimation from a stereo image pair. In this approach, we have used bounded straight line-segments as features and have emphasized the use of epipolar geometry of two stereo views for matching these line features. The straight line extraction method introduced here is robust against missing pixels. Fundamental matrix based matching is performed here as the end points, lengths and centers of the detected straight lines are susceptible to possible occlusion, fragmentation and/or missed edges. A descriptor storing the location as well as co-planarity information of a set of 3D edges is also developed. Valid 3D planes are finally separated from pseudo planes through a method based upon 3D scale invariant feature points which are calculated by triangulating matched 2D scale invariant feature points of both the stereo images. Due to the inherent robustness and sub-pixel level accuracy of 2D SIFT points, and the calculated 3D SIFT points, the pseudo plane rejection algorithm is accurate and robust to geometric re-projection error.

References

[1] I. R. P. Smith and A. Davison, "Real-time monocular slam with straight lines," in *Proc. 17th British Machine Vision Conference*, Sept. 2006, pp. 17-26.

[2] N. Ayache, "Stereovision and sensor fusion," *MIT-Press*, 1990.

[3] G. Medioni and R. Nevatia, "Segment-based stereo matching," in *Proc. CVGIP*, July 1985, pp. 2-18.

[4] N. Ayache and B. Faverjon, "Efficient registration of stereo images by matching graph descriptions of edge segments," in *Proc. IJCV*, 1987, pp. 107-131.

[5] P. Gros, L. Imag, and I. Rhone-alpes, "Matching and clustering: Two steps towards automatic object modelling in computer vision," in *Proc. AAAI Fall Symposium Series: Machine Learning in Computer Vision*, 1993, pp. 40-44.

[6] C. Schmid and A. Zisserman, "Automatic line matching across views," in *Proc. CVPR*, San Juan, 1997, pp. 666-671.

[7] T. Buchanan, "Critical sets for 3D reconstruction using lines," in *Proc. ECCV*, 1992, pp. 730-738.

[8] P. F. F. Bignone, O. Henricsson, and M. Stricker, *Automatic Extraction of Generic House Roofs from High Resolution Aerial Imagery*, Springer Verilag, 1996, pp. 85-96.

[9] Baillard and A. Zisserman, "A plane-sweep strategy for the 3d reconstruction of buildings from multiple images," in *Proc. ISPRS Journal of Photogrammetry and Remote Sensing*, 2000, pp. 56-62.

[10] R. Hartley and A. Zisserman, "Bilinear relations," in *Proc. Multiple View Geometry in Computer Vision*, Apr. 2004, pp. 411-413.

[11] R. Hartley and A. Zisserman, "Epipolar geometry and the fundamental matrix," in *Proc. Multiple View Geometry in Computer Vision*," Apr. 2004, pp. 239-246.

[12] Z. Zhang, R. Deriche, O. Faugeras, and Q. T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial Intelligence*, vol. 78, pp. 87-119, Oct. 1995.

[13] R. Hartley and A. Zisserman, "Camera models," *in Multiple View Geometry in Computer Vision*, Apr. 2004, pp. 312-313.

[14] R. Hartley and A. Zisserman, "Camera models," in *Multiple View Geometry in Computer Vision*, Apr. 2004, pp. 315-320.

Vishwmitra S. Bhadouria received his B.Tech. degree from Uttar Pradesh Technical University, Uttar Pradesh, India, in 2007. He joined Bharat Electronics limited, Bangalore, India, in 2007 and worked as radar signal processing engineer till 2011. He received his M.Tech. Degree in electrical engineering from the Indian Institute of Technology (IIT) Kanpur, Uttar Pradesh, India, in 2013. He joined IBM, Bangalore, India, in 2013, where he is currently working as Associate hardware engineer. His main areas of research are computer vision, and signal processing.

Disha Prakash received her B.Tech Degree in Electronics and Instrumentation from M.I.P.S, Kanpur, India, in 2012. Currently, she is working as a project associate in the department of electrical engineering and A.C.E.S, Indian Institute of Technology, Kanpur. Her research interests include image Processing and computer vision.

Venkatesh K. Subramanian received his B.E degree in Electronics from Bangalore University, M.Tech degree in Communication and PhD in Signal Processing from Indian Institute of Technology, Kanpur. Currently, he is working as a professor at the electrical engineering department in Indian Institute of Technology, Kanpur. His research interests include image processing, video processing, computer vision and vision applications in navigation and robotics.