Stereo vision for Robotics

Jyothi Digge
SGB Amravati University
Amravati,India

Yashraj Digge
SIES GST
Navi Mumbai, India

ABSTRACT
The human eye-brain system is responsible for the highest percentage of sensory data from the environment. A major part of this system which is unique to all predator species (including man) is the ability to perceive distance of various objects using their two eyes. This is one of the primary reasons that man and other predators have both their eyes on the same side of their head (i.e. front) unlike those in herbivore species like cows, goats, rabbits etc. The phenomenon of calculation of distance using only optical aids (namely the 2 eyes) is known as Binocular Depth Perception or stereo vision

The importance of judging distance is apparent in comprehending and manipulating a 3-D environment, which is why it is so crucial in modern robotics. We intend to replicate the effect in software using 2 web cameras for image acquisition (instead of the eyes) using conventional/new methods in this paper. We have tried to find solution to two problems 1) Point correspondence problem 2) Stereo Triangulation Problem After solving these two problems, all that is necessary is to combine the outputs of these two to generate a 3-D depth map enumerated in any standard unit of length or distance. MATLAB was considered as the primary development tool.

Categories and Subject Descriptors

1.4 Image Processing and Computer Vision

General Terms

Theory, algorithm and Experimentation

Keywords
Robotics, Binocular, Depth Perception, Optical aid and Sensory data

1. INTRODUCTION
Depth perception has been commonly used in many applications like quarrying, construction, instrumentation, for level measurement, proximity sensors for robots etc.[1] But mostly other tools like ultrasound sensors; infrared proximity sensors etc have been used. Perceiving depth using purely optical means, as is found among higher predators in nature have been tested and applied in many robotic applications, but there is still a need to augment them with other devices such as those mentioned above[2]. The main reason for this seems to be the lack of knowledge of the visual cortex and by extension the higher brain in the task of perceiving depth. The use of optical sensors to measure distances is also called stereoscopy[3.]
The basic model for human stereoscopy is quite simple i.e. to capture 2 images of the same object from slightly different angles, measuring the deflection and then calculating the depth using one of the many parallax formulae. The apparatus for such a system would be 2 web-cameras mounted on a stand, with a single degree of freedom (i.e. rotation). Both cameras are connected to a computer using standard ports (typically USB). The computer will run processes for image acquisition, deflection measurement as well as 3-d rendering, whose output on the screen will be given as a 3d graph which can be viewed from different viewpoints. This 3-d map will be equivalent representation for objects' co-ordinate relations in the scene being pictured.The most vexing problem encountered in creating such a system is that of finding point correspondence in both images. This problem arises as a subpart of deflection measurement. In this paper we will explain 3 point correspondence algorithms. The image acquisition and the rendering process. Although important were not covered as a part of literature due to widespread availability of API s and toolboxes for the task. Stereo matching by itself is responsible for finding the pair of matching points in the left and the right image. Calculation of depth can be done by using simple 3-d trigonometry or using the parallax formula. This section mentions 3 stereo matching algorithms, all using different principles and all of them varying in their matching accuracy and performance. It must be noted that the algorithms presented here are not the only ones which are "on the table" during the final implementation phase.

2. BASIC SYSTEM ARCHITECTURE

A typical stereovision system will consist of 5 steps. The simplified chart is shown in (Figure1).

1. Image acquisition from the 2 cameras.
2. Establishing point correspondence.
3. Listing point correspondence
4. Calculating depth for each list entry.
5. Rendering depth onto a 3d map

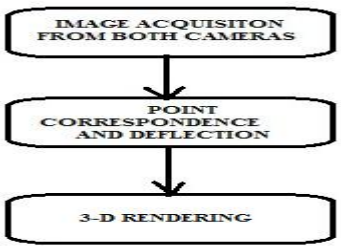


Fig 1: A 3 step general flowchart for a depth perception system

3.POINT CORRESPONDANCE CALCULATION

This step is probably the most important and the most difficult step in designing the entire system. Point correspondence calculation, involves the finding a point in the second image representing a given point in the first image. The pair formed by a point and its corresponding point is called a correspondence pair. It is between this pair that the deflection is calculated. To generate a 3-d representation of the environment from two 2-d

images the point correspondence calculation must be carried out for every sub region or every pixel. Several algorithms exist for point correspondence. We have used the following algorithms 1) Gabor wavelets [2], Multi Channel Graph Matching [3]-[4] and Use of Hopfield neural networks [5].

3.1 Use of Gabor wavelets

Gabor functions are Gaussian-modulated sinusoids, which are well-known for their optimum joint resolution in both spatial and frequency domains. It is also widely used in psychophysical models for simple cells, as well as theories for computer vision. The complex 2-D Gabor functions ***h (z,* y)** have the general form
Where

$$h(x,y) = g(x,y)\exp[2\pi j(u_C x + v_C y)] \quad (1)$$

$$g(x,y) = \frac{1}{2\pi\lambda\sigma^2}\exp[-\frac{(x/\lambda)^2 + y^2}{2\sigma^2}] \quad (2)$$

h (z,* y)** is a 2-D Gaussian function modulated by a complex sinusoid with aspect ratio **X** and scale parameter ***0. The spatial frequency response of the Gabor function (2) is:

$$H(u,v) = \exp\{-2\pi^2\sigma^2[(u-u_C)^2\lambda^2 + (v-v_C)^2]\} \quad (3)$$

This algorithm belongs to area based stereoscopy class of algorithms. The output of this algorithm is a 2-d disparity map which represents the deflection of the corresponding point in the other image for each pixel. This algorithm uses mathematical modeling of disparity cells which are found in the visual cortex. Each cell can be modeled as:

$$h(x,d_1;w_C) = g(x_1)\exp(jx_1{}^T w_C) \quad (4)$$

Where **x1**=(x-d_1(y)), g(x_1) is the Gaussian shape in eq 3. The cell responses for each image can be approximated for each image using a linear Gabor filter:

$$y_l(x,0;w_C) = l(x) * h(x,0;w_C) \quad (5)$$

$$y_r(x,0;w_C) = r(x) * h(x,0;w_C) \quad (6)$$

Where y_l and y_r are phasors that are represented by magnitude and direction. Lastly the disparity channel is modeled as:

$$B(x,\Delta\theta;w_C) = T[y_l(x,0;w_C) + y_r(x,\Delta\theta;w_C)] \quad (7)$$

$$= T[m_l(x)\angle\psi_l(x) + m_r(x)\angle\psi_r(x)] \quad (8)$$

The disparity map generated for a pentagon image using this algorithm is shown in (Figure 2).

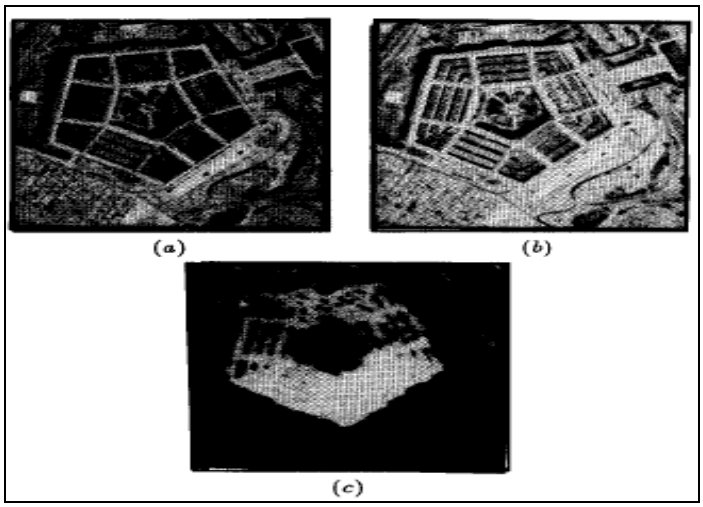


Fig 2: (a) and (b) are left and right images of the pentagon and (c) is the final disparity map.

3.2 Multi Channel Graph Matching

This algorithm belongs to the feature based class of correspondence algorithms. The basic primitive feature used in this algorithm is curve segments in an image. The curve segments are extracted using zero crossings (similar to edge detection). The centroids of these curve segments are then extracted and a graph is formed connecting these centroids. Similar graph is created for the other image as well. Then a graph isomorphism is performed to find the corresponding points. There are 5 steps in this method:

3.2.1 Zero crossing curve segment extraction.

The zero-crossings in the left image and the right image of a stereo pair are detected by using a two-dimensional Laplacian of a Gaussian operator given by:

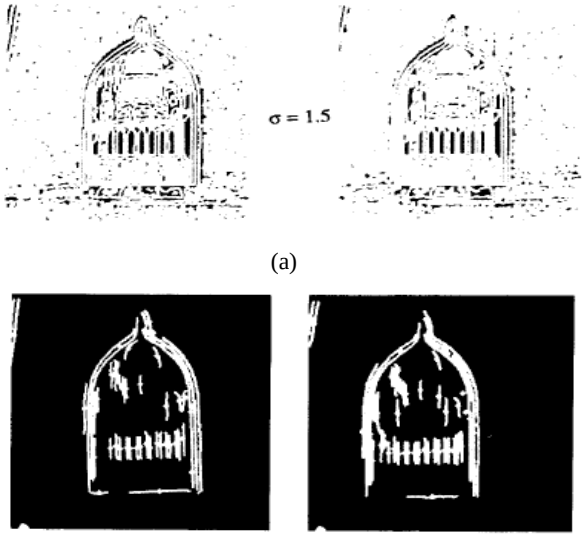
$$\nabla^2(i,j) = \left[\frac{(i^2+j^2)}{\sigma^2} - 2\right]\exp\left\{\frac{-(i^2+j^2)}{2\sigma^2}\right\} \quad (9)$$

Zero crossings are obtained by scanning along the line and the column of each processed image, and locating pairs of adjacent elements of opposite signs. Figure 3 shows 0-crossing extraction output of an image:

3.2.2 Curve tracking

A tracking algorithm is used to segment the boundaries of the objects in the scene into short curve-segments. Each curve-segment is labeled by a number, the locations of all the edge points forming the curve-segment are stored, and the centroids of each curve-segment are calculated. The starting points for tracked curve-segments are the edge pixels whose gradient magnitudes are above a pre-defined high threshold. From each starting point the curve is tracked recursively in both directions by considering the magnitude and the orientation of the neighboring points. At any point on the curve each of its &neighborhood points is included in the tracked curve if its gradient magnitude is above a predefined low threshold, and its

orientation does not differ by more than a pre-defined threshold, say 18, from the previously tracked point. If the difference in the orientation between two tracked points exceeds the threshold, or if the magnitude of the new point dips below the low threshold, and then the curve is broken off. (Figure 3) shows the result of curve tracking.



(a)

(b)

Fig 3: (a) image shows 0 crossing extraction and (b)shows the result of curve tracking for left and right image respectively.

To identify an instance of each left curve-segment in the right image we must extract and use some distinctive features about the curve. We use the Generalized Hough Transform of the curve as a distinct property of the curve-segment. The Generalized Hough Transform is used because it groups all the information about the individual zero-crossings on the curve into a single table which uniquely represents that curve-segment. Pixels are noisy and point-to-point matching is not possible. A table called the R-table is constructed for each curve-segment in the left and the right image. This is done by calculating the orientation ***Oxy*** of each edge pixel.

$$Diagonal\ displacement\ (DD-45^{\circ}) = \sum_{x=0}^{3}\sum_{y=0}^{3}[f(x,y) - f(x+1,y-1)] \quad (10)$$

$$Horizontal\ displacement(HD) = \sum_{x=0}^{4}\sum_{y=0}^{3}[f(x,y) - f(x,y+1)] \quad (11)$$

$$Vertical\ displacement(VD) = \sum_{x=0}^{3}\sum_{y=0}^{4}[f(x,y) - f(x+1,y)] \quad (12)$$

$$Diagonal\ displacement(DD-135^{0}) = \sum_{x=0}^{3}\sum_{y=0}^{4}[f(x,y) - f(x+1,y+1)\ (\quad 13)$$

3.3 Use of Hopfield Networks

When using the neural network technique the matching problem can be formulated as minimization of a cost function (constrained optimization) where all the constraints on the solution can explicitly be included in the cost function. Minimization of the cost function can then be performed by a Hopfield network. By using a neural network to carry out the feature matching, a global solution can be obtained because all the neurons are interconnected in a feedback loop, and the output of one affects the input of all the others as shown through the mathematical summation of each neurons output although each neuron is working independently. General Hopfield neural network is shown in (Figure 4).

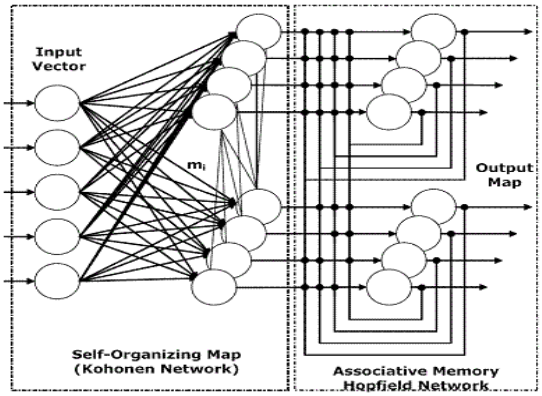


Fig 4: A general Hopfield neural network

3.3.1 Feature extraction using interest operator

The first step is to find points of high interest in both images. The points of high interest are distinct points in the two images which can be more easily matched than any other point in the image. These points generally occur at the corners and edges of the object in the image. These points are called the interesting points which are mostly 'corners" in the image. The 'corners' are defined as points in the image that have high variance in all directions. High variance occurs when there is a radical change in the grey level of the image. The operator used is called the Interest Operator which finds the variance in the horizontal, vertical, and both diagonal directions. To obtain the directional variances, a 5x5 window is opened around each image point.

The function f(x,y) refers to the grey level value at that point in the image. The smallest value of VD, *HD, DD-45",* and DD-135", called the interest operator value, is taken as the variance for that point. The output of this stage is shown in (Figure 5).

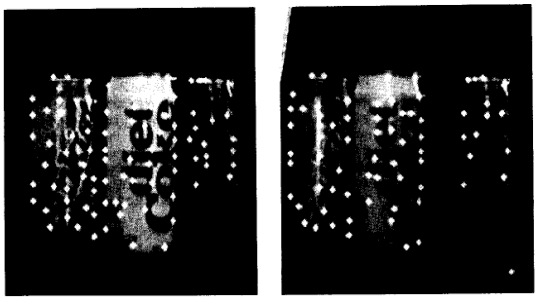


Fig 5: Images (left and right) and their interesting points superimposed.

3.3.2 Energy minimization using Hopfield networks

A two dimensional binary Hopfield network is used to find the correspondence between the interesting points in the left and right images. The network is represented as an $N_i * N_j$, array of neurons where N_i and N_j, are the total number of interesting points in the left and right images respectively. The state of each neuron in the network represents a possible match between an interesting point in the left image with one in the right image. The Liapunov energy equation for a two-state Hopfield network is given by

$$E = -(1/2)\sum_{i \neq j} T_{ij} V_i V_j - \sum_i V_i I_i \qquad (14)$$

Where V_i and V_j, represent the binary state of ***i*** and ***j*** neurons respectively, which can be either 1 (active) or 0 (inactive), $T_{ij}=T_{ji}$ is the interconnection strength between the two neurons and Zi is the initial input to each neuron. A change in the state of neuron ***i*** by ΔV_i will cause an energy change of ΔE.

$$\Delta E = -\sum_{i \neq j} [T_{ij} Vj + I_i] \Delta V_i \qquad (15)$$

The values of $T_{ij}$is computed for every neuron by minimizing the energy function. High value of T_{ij} indicates correspondence between i^{th} pixel in the left image and j^{th} pixel in the right image. The o/p of this process is shown in the (figure 6).

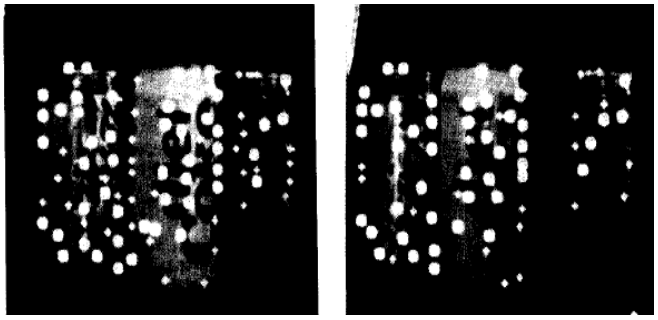


Fig 6: The experimental result is represented as the large circles represent matched points and small circles representing points with ambiguous matches

Running the Hopfield network resulted in 49 interesting points having a single match and **5** interesting points having multiple matches. To find a correct match for each interesting point with multiple matches, the vertical disparity (Y-disparity) of each multiple match is compared with an average Y-disparity obtained from all the single matched points. The candidate whose Y-disparity is within a threshold of the average Y-disparity is chosen as the correct match and other matches are deleted. Our assumption is that the Y-disparity should be the same for all the correctly matched points. The interesting points outside this window will never be considered as a possible match because of the way the stereo images are formed. The window size chosen was large enough to include the largest horizontal shift (disparity) of the three cans as well as any vertical displacement of the interesting points caused by noise or illumination due to different locations of the camera when pictures were taken. Also a window of size 40x40 is opened around the point *i* and all the other points. The updating procedure is iterated until the network reaches a stable state. The network is said to be at its stable state (minimum energy) when there is no change in the states of neurons which is then stopped after another 5000 extra iterations.

The results of all the outputs were decomposed into disparity values for individual pixels. Then intersection of these three sets of points yielded the "valied members" for which the disparity values were averaged.

The algorithms used for stereo matching encompass fields of digital signal processing in case of Gabor wavelets to artificial neural networks in case 2-d Hopfield networks are used. If required, any preprocessing steps such as contrast enhancement, noise filtering, image rectification will automatically tie in the methods of image processing. In the rendering system the rendering itself and the camera transform operations like roll and pitch transforms, for example require 3-d perspective projection and 3-d coordinate transforms which are a part of computer graphics.

4. 3D RENDERING

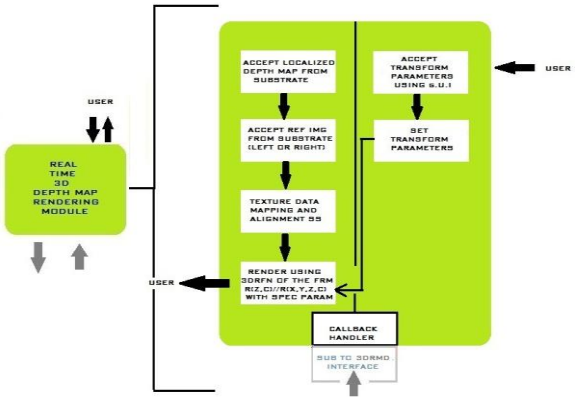


Fig 7: **Architecture of the 3-D Rendering Module.**

Like the image acquisition module the 3-D Rendering module shown in (Figure 7) contains 2 partially independent "logics". The "left logic" is for rendering the image and the "right logic" is for accepting the graph rendering and transform parameters.

The "left logic" works as follows:

1. Accept the localized depth map from the stereo matching module (through substrate).
2. Accept reference image from the image acquisition module (through substrate).
3. Set the texture mapping input data to the reference image matrix.
4. Perform alignment (i.e. un-padding/unpacking for X,Y,Z,C format).
5. Render using the modified mesh() or surf() function.

The "right logic" is responsible for transforming the rendered graph transform using a G.U.I for accepting parameters for roll, pitch and yaw as well as vertical, lateral and vectored camera movements.

Unlike the first module however, where the two logics are independent of the fact that the "communication" takes place through camera drivers and not internally, the camera parameters in this case need to be explicitly passed to the rendering subsystem. Moreover the interleaving of these "logics" and the parameter handoff dependency will be handled by the sequencer. Like other modules the 3DRM uses the sub-to-3d-mod interface to interact with the substrate

5. USER INTERFACE DESIGN

The (figure 8) is a visual conception of the software as it will appear to the user. This figure was generated keeping in mind the design goals specified during the requirements stage

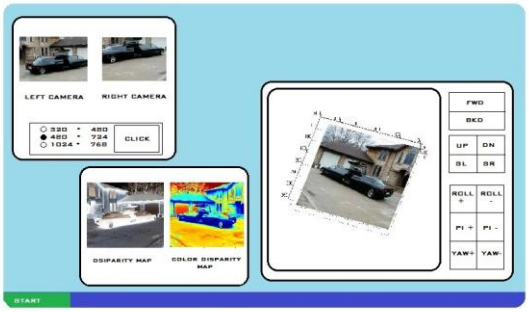


Fig 8 : System Visualization.

It can be seen that the system's graphical output contains 3 windows. The window on the top left is the screen generated by the image acquisition subsystem featuring displays for the left and right images as well as a control panel for resolution control. The click button which captures the frames from the cameras will only be used only in the prototype, as the real time functionalities are not supported initially.

The window on the bottom left is the output of the stereo vision subsystem, displaying the disparity map on the left and the non localized color depth map on the right.

The largest window is the output screen generated by the 3-d rendering subsystem, with a screen that shows the depth map in 3-d as defined by the viewing parameters. These viewing parameters are controlled by the 3 control panels where: -

a) Topmost (fwd\bkd) controls translation along y axis b) Middle (up/dn and sl/sr) control translation along z and x axis respectively. c) Bottom controls camera's roll, pitch and yaw motion.

6. CAMERA CALIBRATION PROCESS

Print out a checkerboard calibration sheet and secure it on a rigid flat surface e.g. a hard cardboard or a tray as shown in (figure 9). This will be used as a reference object during calibration

1. Hold the tray in front of the camera
2. Capture the scene using the camera rig such that the entire tray is visible in both the left and right image.
3. Save these 2 images on disk in any standard format
4. Start the calibration tool
5. Load the left image into the Matlab Workspace in any of the mode.
6. Click on the "extract grid corners" button to start the grid extraction process.
7. Mark the corners on the image such that it properly encompasses the squares.
8. Enter the window size for auto-corner extraction this and the physical dimension of the squares

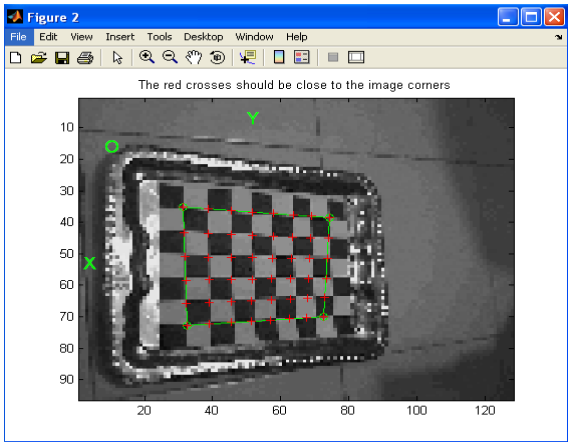


Fig 9: Manual corner marking process (origin, x-axis and y-axis are shown).

7. TESTING & RESULTS

The Black Box testing was performed for the Stereo Triangulation and the triple channel disparity routine as shown in (Figure 10).

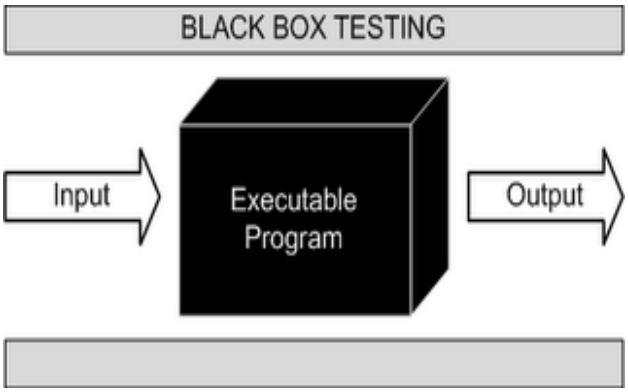


Fig 10: Schematic Diagram depicting Black Box Testing process

7.1 Black Box Testing for the Triple Disparity Segment

1. Run the Triple disparity routine for any general scene.
2. Use the MATLAB command window to view the 'dispx' and 'dispy' matrices which indicate the 'x' and 'y' disparity respectively.
3. Since the x and y co-ordinates of the principle points are already known, Black Box testing involves visual verification of the principle and the matched points.
4. To perform step 3, use the MATLAB's 'subplot' and 'imshow' functions to view the left and right images.
5. Mark any principle point on the left image using the data cursor tool.
6. Make a note of the corresponding point's x and y co-

ordinates in the 'dispx' and 'dispy' matrices.

7. Mark these values on the right image using the data cursor tool.
8. If the sub-image in step 4 and step 7 are identical, then matching has occurred properly, else it indicates incorrect matching or ambiguity.
9. Repeat step 5 till 8 for all principle points in the left image.
10. Repeat step 1 to 9 for at least 5 different scenes.

It was observed that after following the above procedure, the routine performs near perfect stereo matching in case of scenes with very high variance. On the other hand, scenes involving homogenous objects like walls or any other large flat surfaces result in matching ambiguity.

7.2 Black Box Testing for Stereo Triangulation Segment

1. Generate N 4-tuples of the form [x, y, x', y'] from the test results of the stereo correspondence segment. Here (x, y) indicate the co-ordinates of the principle points and (x', y') indicate the corresponding stereo matched points.
2. Load all the calibration parameters into the MATLAB workspace.
3. Write (x, y) n its transpose form as 'x_L' array and do the same for (x', y') as 'x_R' array.
4. Call the stereo triangulation function giving the xL and xR arrays as arguments.
5. Store the returned results in 'left depth' and 'right depth' arrays, each of size 3 x 1.
6. Discard the right depth array and store only the third value of the left depth array in a variable 'distance'.
7. Measure the perpendicular physical distance from the left camera to the actual object appearing at the principle point in the left image and note the value in variable 'tape'.
8. If the difference between 'tape' and 'distance' is within acceptable limits then it can be concluded that the stereo triangulation segment provides the correct output in this instance.
9. Repeat the procedure for several principle points in the same image and calculate the average error.
10. Repeat steps 1 to 9 for at least 5 different scenes.

7.3 System Testing Procedure and Result

Fig 11: Front view of the stereo Rig

Using the Stereo Rig shown in (Figure 11). A general scene with high 'image variance' was selected as shown in (Figure.12).

Fig 12:The Chosen Scene for System Testing

The results obtained after running the script is as shown in (Figure 13).

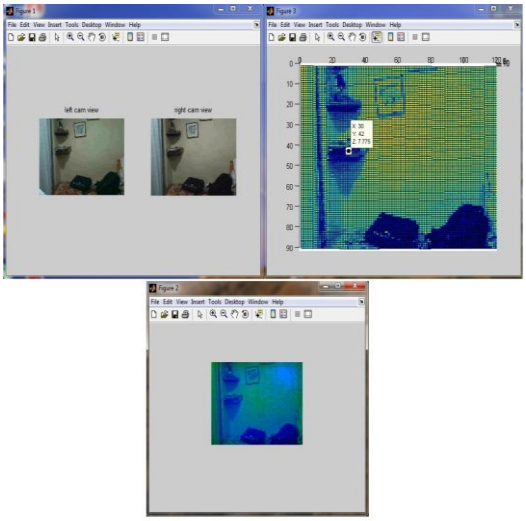


Fig 13: The Output for the stereo image with x,y and Z coordinates

The result is verified by measuring the actual frame orthogonal distance using the measuring tape. Comparing the actual measured distance and the third z-axis value displayed we calculate absolute and relative error. This is given in Table.1.

Average Error = 0 ft

Root Mean Square Error = 0.16 ft

Table .1.Indicating the triangulated and measured distance

Serial No.	Triangulated Distance(ft)	Measured Distance(ft)
1	7.7	7.6
2	6.3	6.5
3	5.6	5.5
4	7.7	7.5
5	5.3	5.3

8. CONCLUSION

In this paper we have presented the new method of combining 3 algorithms to calculate the stereo vision, where the averge error is zero. Stereo vision is highly important in fields such as robotics, to extract information about the relative position of 3D objects in the vicinity of autonomous systems . We explored the basics of binocular depth perception systems. In the process we looked at several methods of finding corresponding points in the image pairs as well as different ways for rendering the calculated depth data. We used these techniques in our own implementation of a binocular depth perception system(stereo vision). In addition we used a 3-D rendering system to give the output a new level of realism.

9. ACKNOWLEDGMENTS

Our thanks to Mrs. Aparna Bannore,HOD ,Department of Computer Engineering ,SIES GST for technical discussion and guidance

10. REFERNCES

[1] BHAT DIKAR AND NAYAR SHREE .1998.Ordinal Measures for Image Correpondence. - IEEE Transactions on Pattern Analysis and Machine Intelligence: 4 : Vol. 20 (April1998).pp - 415-423

[2] WILLIAM AND CHEN TIEN YUH BOVIK ALLEN, KLARQUIST.1994Stereo Vision Using Gabor Wavelets" Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation IEEE : Vol. 12(Dec1994) .pp. 13-17

[3] FIELDING GABRIEL AND KAM MOSHE. 1997.Applying the Hungarian Method to Stereo Matching" IEEE Conference on Decision & Control. - San Diego, California USA ,(June19970. pp. 1-6.

[4] NASRABADI NASEER, LIU YI AND CHINAG JEN LAI 1988.Stereo Vision Correspondence Using a Mutlichannel Graph Matching Technique" . - IEEE International Conference on Robotics and Automation, 1988. Vol. 3(Dec 1988).pp-1804-1809

[5] HEIKKILÄ JANNE AND SILVÉN OLLI .(1997),A Four-step Camera Calibration Procedure with Implicit Image Correction IEEE Conference on Computer Vision and Pattern Recognition. - San Juan, Puerto Rico (October1997.) Vol. 1.pp-1-5

[6]GONZALEZ RAFAEL C. AND E.WOODS RICHARD.2003.Digital Image Processing. Pearson Education, 2003. - Vol. 2.