Visual Odometric Navigation: the Generalized Feature Vector way

J.Bhattacharya
CMERI
Mahatma Gandhi Avenue
India 713209,Durgapur,West Bengal
bjhilik@cmeri.res.in

S.Majumder
CMERI
Mahatma Gandhi Avenue
India 713209,Durgapur,West Bengal
sjm@cmeri.res.in

ABSTRACT

One of the main challenges faced by object tracking and environment-modeling techniques is the frame-to-frame correspondence of the object of interest. False detections may lead to the tracking of wrong object thus misrepresenting information about the object location and its track. The tracking algorithm of the detected object should also be computationally inexpensive and suitable for real time applications. This paper discusses how GFV, a multidimensional entity encapsulating multiple feature parameters, can uniquely identify dominant features of an object, and increase the detection reliability due to its potential to function consistently in any kind of environment, uninfluenced by view point invariance or extrinsic factors, thus generating minimal false alarms. Further a method to determine the 3D position of the object is presented which works on uncalibrated camera images and can be successfully applied to online processes. Experimental analysis using a outdoor mobile robot have been carried out to establish the competence of the algorithm. A statistical approach to reject outlier data, if any, is applied while generating the trajectory of the mobile robot used for experiments

Keywords

Feature detection, trajectory identification, object tracking.

1. INTRODUCTION

The implementation of vision based automated systems in various fields like security, surveillance, robot navigation, remote environment sensing and medical diagnosis is in the continuous evolvement of research in tandem with the field of object tracking and environment modeling. The task of tracking encapsulates within it primary operations like image segmentation, object detection and extraction, depth estimation and finally, object trajectory estimation. The main challenge faced by the detection techniques lies in the frame-to-frame correspondence of the regions of interest; which becomes difficult for non-rigid objects exhibiting complex motion, or in frames where the object is occluded, or when the scene illumination is extremely influenced by environmental conditions. Mainly two approaches are taken in the vision based correspondence problem solving. These are area based and feature based techniques. Detection of feature from exteroceptive sensors has remained an important area of research for several reasons. Firstly it provides the unique opportunity to abstract and encapsulate the dominant and distinguishable characteristics of the environment or scene from the sensory data. Secondly it is a process of reducing the resource requirement and the associated complexity of handling large data sets in real-time. Often features are defined as geometric primitives such as point, line, arc segments or some form of derived entities from the amplitude return history such as color and texture for example. In general, features segregate “objects of interest” from the raw sensory data. Various algorithms have been proposed by different researchers for object detection and depth recovery.

The progress of research in the field of feature detection using vision can be mainly categorized into four distinguishable classes. In the initial stage researchers mainly concentrated on detecting geometric features like edge and corners of the image to identify objects of interest. A large amount of work has been undergone in this area [1, 2]. The problems of most of these algorithms lie in the fact that they are not invariant to affine transformations and are also viewpoint dependent. The second class of algorithms uses primitive geometrical shapes as

features. General methods for shape recognition are moment based, structure based and Fourier descriptor based [3]. The next class of algorithms models the object as probabilistic distributions. These distributions represent features such as color [4], texture [5] etc. Another variation to color and texture detection of a region is background modeling which is beneficial in detecting only moving objects [6]. For the fourth class of algorithms, the object shape and appearance are generated simultaneously. These models also encode different views of an object, removing the shortcoming of viewpoint dependencies of the previous methods [7]. Inspite of many uniqueness and advantages, most of these methods require large computational power and hence unsuitable for real-time navigation and tracking application. Another limitation is that many of these methods have been developed for a specific sensor suite and well-structured indoor environment for specific applications and consider camera calibration as a prerequisite rendering them unsuitable for outdoor and unstructured environment where extrinsic parameters are dominant rather than intrinsic. Present work extracts features by detecting and representing them in a generalized feature vector (GFV), which can be used to uniquely identify each of the dominant objects in an image. Once features are detected using GFV, the next important task lies in estimating the depth measurement of the object of interest. In the past few years several techniques for depth recovery and construction of depth maps have been developed. This area is still an active research area and development in this field is in continuous progress. The issue has been investigated by different researchers from different viewpoints but can be categorized mainly into six main classes. The first class includes all methods, which are based on depth measurement from two cameras. Finding corresponding points between the two images precedes depth calculation while using stereo [8]. The second class comprises of methods that use simple geometry to recover depth information [9]. The next class of algorithms are those that derive depth information of the targets from the velocity estimation of the targets [10]. The fourth class of algorithms consider calculation of depth from optical blur, defocusing techniques [11] .The next class uses interpolation functions for depth estimation [12] .The last class comprises of those methods which use auxiliary devices such as laser range finders or ultrasonic sensors to measure depth.

As a significant departure, the work reported here uses the image magnification to estimate the depth and thereby compute the trajectory. The main interesting issue of this algorithm is that it "does not require" explicit camera calibration "for depth recovery". The paper is organized in the following manner. Section 1 provides basic background of the problem. This section also includes an outline of various significant work carried out for consistent feature detection and depth estimation. Section 2 defines the GFV framework and its comparison with other conventional approaches briefly, whereas Section 3 includes the position determination of the detected object for trajectory development. Section 4 deals with results and performance analysis of experimental findings. Finally, discussion and conclusion of this work is presented in Section 5.

2. THE GFV FRAMEWORK

The basic idea of using GFV as a scene descriptor stems out of the fact that point features often require a secondary level of corroboration such as color and texture to make it invariant. The generalized feature vector (GFV) is considered to be a multidimensional entity, which can include multiple parameters like color, shape, energy, entropy, size ratios and many more. Some of these parameters may be orthogonal to the other. In principle GFV can include as many parameters as desired. Another uniqueness of GFV is that it can also accommodate "feature parameters obtained from other co-located sensors". There is no limit on how many feature parameters can be included in GFV. Although inclusion of multiple parameters can improve the detection reliability it however may increase the computation cost. Therefore for optimal performance not more than three parameters should be used. However the actual number of parameters will depend on the application requirements and available computational resources. Figures 1 & 2 shown in the appendix at the end of the paper, further demonstrates the algorithmic flow of the GFV briefly using a sample image. The method mainly consists of two steps: - During first step a reference model of GFV is created which is then applied to the actual data in the second step. The details of the algorithm and its establishment have been discussed in the reference [13] and are beyond the scope of this paper.

The suitability of GFV lies in the fact that even when any information about the environment of the object to be detected or presence of other objects in its surrounding is not known, the method will provide reasonably accurate results instantly without false alarms. The user need not have to decide which features are to be matched or in which order they are to be matched in order to get the best matching. Thus GFV is self-deciding and can operate independently in any environment without any prior knowledge about it. Failures of many object detection algorithms, mainly due to view point invariance; occlusion and influence of other extrinsic factors can

be successfully resolved by the GFV. GFV also responds very well even in outdoor environment. Similar objects can be identified from a sequence of images taken at different time in different environmental conditions. Since GFV is essentially a multi parametric matching method, it is more robust compared to any other step-by-step matching algorithm.

3. POSITION CALCULATION

Any camera image is a 2D projection of the 3D world using perspective transformation. As a result, estimation or recovery of object distance from the camera requires elaborate mathematical procedure. Various depth detection algorithms using monocular camera and their relative merits have been already mentioned in section 1. In this section an alternative technique using the thin lens equation and the image magnification factor (shown in equation 1 below) is used to calculate the depth. This technique is suitable for online processes and doesn't require large computational overhead. For computing the object depth for every image frame, the magnification ratio is estimated for each image frame from its object and image dimension ratio. This method has only one limitation i.e. object shape; size and approximate dimensions should be predefined. The image dimensions like area, perimeter, shape and size ratios are already computed while detecting the object as seen in section 2. Any of the above mentioned dimensions may be used but the choice should be kept fixed for all the image frames. The depth estimation procedure is further illustrated below:

The thin lens equation gives

$$\frac{1}{u}+\frac{1}{v}=\frac{1}{f} \tag{1}$$

where u is the image distance, v is the object distance required to be calculated and f is the focal length of the lens. The magnification ratio m, is given by

$$m=\frac{u}{v}=\frac{I}{O} \tag{2}$$

Here, I and O give the image size and object size respectively. Substituting u as mv in equation 1, v can be written as

$$v=\frac{(1+m)f}{m} \tag{3}$$

While computing the depth d_n for each of the camera frames n using equation 3, there are two factors that should be resolved. Firstly obtaining the image size for computing the magnification factor should not consider the total surface of the extracted image. The part of the image to be considered for a particular frame is variable and depends on the viewpoint of the camera for that image frame. This fact is further explained using figure 3 and 4 and subsequently elaborated in the discussion. Secondly, the depth dimension is obtained relative to the camera frame and need not be considered as the actual object distance relative to a fixed world coordinate system. Reason behind this approach is that the camera may be positioned and maneuvered using pan and tilt angle hence making the camera plane rotated with respect to the world frame. Further this depth cannot be associated with the depth dimensions of the other camera frames for trajectory identification as each frame may have a different orientation i.e. different pan and tilt angles of the camera and hence each of the depth dimensions refers relative distance measurement with respect to different camera planes. To obtain the actual depth in the world frame, the calculated depth in each frame needs to be transformed to the world coordinate system. In order to carry out this transformation, the knowledge of the extrinsic camera parameters is necessary for each image frame, which can be obtained through camera calibration. However, for real time applications the procedure becomes complex and time taking. The following paragraphs explain how the problems mentioned above are addressed.

Initially an example is used to illustrate how the appearance of an image of any particular object changes along with the viewpoint or rotation angles of the camera. This further helps to point out how the calculation for the image magnification depends on the viewpoint of the camera. Figure 3 below depicts a rectangular box viewed by the camera from three positions identified as 1, 2 and 3 respectively.

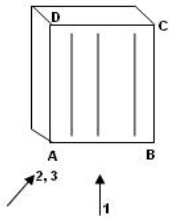

Fig 3: A rectangular object seen from three different camera positions 1, 2 and 3 are shown in the figure above

Position 1 assumes the camera to be perfectly aligned with the world frame therefore no rotation is considered. Positions (2) and (3) denote the same camera position however the camera angles are different. Position 2 considers a pan angle whereas position 3 assumes the camera frame to be rotated by both pan and tilt angles. The resulting image appearance for each of the camera positions is depicted in figure 4. For calculating the magnification ratio for figure 4a, the total surface of the image is to be considered; however for the

remaining two images (4b and 4c) of the figure or for any similar case where more than one of the side faces are visible, such a step would provide wrong results. Thus for correct magnification determination the surfaces needs to be separately identified in order to select the desired one among them. The GFV method discussed previously can easily separate out the region of interest of the object as it can detect all the outer corners of the image from which the boundary edges can be calculated. The selection of the corners to calculate the edge, which will denote the image size, depends on the shape of the object and will vary accordingly. In this particular case for different camera positions two bottom edges (bottom edges are used here just as an example, top or side edges can be used as well) may be detected. For example if figure 4c is considered, the two bottom edges detected are *E'A'* and *A'B'*. As the object dimensions are known, one of the detected edges can now be selected depending on their length, i.e. if the matching is to be done with object side *AB*, then the longer among the two detected edges (considering side *AB* > side *EA*) will be chosen or vice versa. This part is conferred in details while discussing trajectory identification case studies later in the paper.

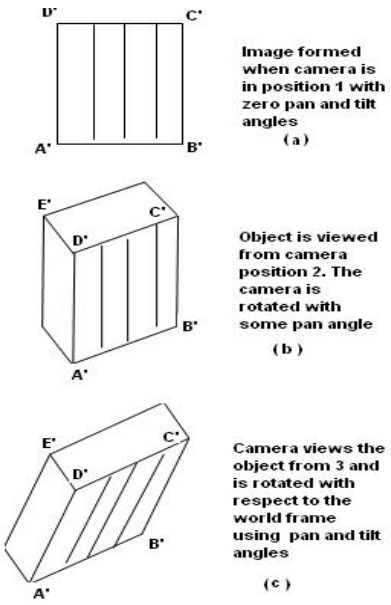

Figure 4: The image of the rectangular object formed for the three camera viewpoints depicted in figure 3 is seen above

The next task is to compute the camera rotation angles relative to the first frame so that the trajectory can be identified. Before getting into the details of the rotation angle computation process, Figure 5 depicts the rotation of the camera plane with respect to the world frame for camera position 3 of figure 3. This figure is used to establish the impact of the camera rotations on its corresponding image frames. The relation between the rotation angles of the camera plane and its resultant image frame is discussed in subsequent paragraphs.

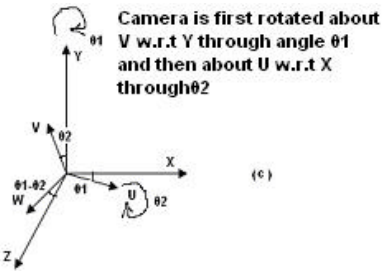

Figure 5: The camera plane orientation for the camera position 3.

In figure 5 the world frame is depicted by *XYZ* plane and *UVW* depicts the camera plane. The corresponding image frame for the camera orientation in figure 5 is depicted in Figure 6. $(PQRS)_1$ here depicts image frame 1 and $(PQRS)_n$ depicts the n^{th} frame. The first image frame is considered as the reference; hence it is assumed that the camera plane of the first frame is aligned with the world frame and all the other camera plane rotations are with respect to this reference frame. The consecutive camera plane rotations of figure 5, by angles θ_1 and θ_2, effects the *x-axis* and *y-axis* of its image frame (nth frame) to make θ_1 and θ_2 angles with the *x* and *y* axis of the first image plane respectively as shown in figure 6.

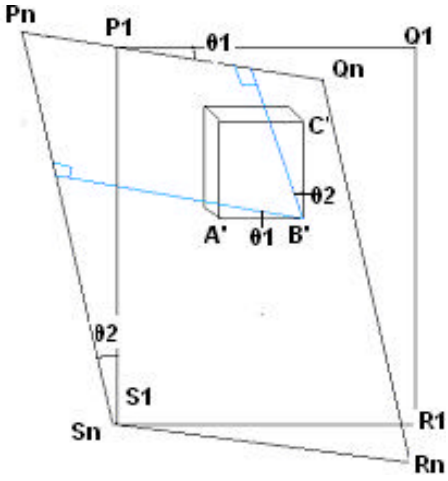

Figure 6: Camera frames $PQRS_1$ and $PQRS_n$ and their alignment is shown above in order to compute the rotation angles

Once these rotation angles are computed, the transformation from the nth frame to the first frame can be undergone. As the camera is aligned with the world frame in the first image frame i.e. the pan and tilt angles of the camera is zero hence the actual depth can be obtained after this transformation. The

trajectory can still be generated even if the first frame is not aligned with the world plane as the relative rotations of all the frames with respect to the first one is computed and the transformation is carried out accordingly. But for such cases the actual depth cannot be determined.

From figure 6, using parallel line properties, it can be seen that angle between side *A'B'* and the perpendicular from point *B'*on P_nS_n equals θ_1 and the angle between side *B'C'* and the perpendicular from point *B'* on P_nQ_n equals θ_2. If the coordinates of *A'*, *B'* and *C'* are given by (x_na, y_na), (x_nb, y_nb) and (x_nc, y_nc) respectively the angles can be calculated from the figure using the following relations.

$$\theta_1 = \tan^{-1}\left(\frac{abs(y_n a - y_n b)}{abs(x_n a - x_n b)}\right) \quad (4)$$

$$\theta_2 = \tan^{-1}\left(\frac{abs(x_n c - x_n b)}{abs(y_n c - y_n b)}\right) \quad (5)$$

Thus the camera angles can be determined for every image plane from the above relations once the points a_n, b_n *and* c_n for every frame is determined. This concept is further used to identify the object trajectory. Two different cases of trajectory identification are discussed:

a) path generated by an object in a situation where the camera is fixed throughout,

b) path generated by a moving camera while tracking a fixed object.

A detailed discussion of the two cases is further presented below.

(a) Camera stationary, object moving

The following steps are executed while identifying the object trajectory:

1. The object of interest is extracted using the GFV algorithm discussed in section 2. The object is denoted in the image plane by its centoid position ${}^{I}O$ in every frame n

$${}^{I}O = (x_n , y_n) \quad (6)$$

2. Five image corners (x_nlb, y_nlb) , (x_nrb, y_nrb) , $(x_nrt, y_n rt)$, (x_nbl,y_nbl) , (x_nbr,y_nbr) are determined. These points are the leftmost bottom, rightmost bottom, rightmost top, bottom leftmost and bottom rightmost pixels coordinates of the detected object and are used to determine image size. An algorithm below presents how corners can be selected for calculating image size of a three dimensional rectangular box when camera rotations are unknown.

```
if (xbl > xlb) && (xbl < xrb)
corners[ ] = {xlb, xbl, xrb}
else if (xbr > xlb) && (xbr < xrb)
corners[ ] = {xlb, xbr, xrb}
else  corners[ ] = {xlb, xrb}
end
if size(corners) > 2
if(length(corners[1],corners[2]))> (length(corners[2],corners[3]))
corners[3 ] = []
else
corners[1 ] = []
end
end
```

From the algorithm, two corners (x_n1, y_n1) *and* (x_n2, y_n2) are selected based on the fact that the larger edge is used to calculate the magnification. The reverse can also be done if desired. The length of the edge formed by these corners can be used as the image size ${}^{I}SZ_nx$ for determining the magnification ratio, 3D position of the object (discussed in step 4) and rotation angles of the camera. (Rotation angles will not be required for this case as the camera is fixed for all the frames). The equation 7 is used to calculate the image sizes ${}^{I}SZ_nx$ *and* ${}^{I}SZ_ny$ in pixels along the x and y axes respectively.

$$\begin{aligned} {}^{I}\mathrm{SZ_n x} &= \sqrt{(y_n 1 - y_n 2)^2 + (x_n 1 - x_n 2)^2} \\ {}^{I}\mathrm{SZ_n y} &= \sqrt{(y_n rt - y_n rb)^2 + (x_n rt - x_n rb)^2} \end{aligned} \quad (7)$$

Magnification ratio can be calculated using the metric coordinates of the image size and the corresponding object side dimension.

3. Depth d_n is computed using equation 3.

4. The 3D coordinates of the image point ${}^{I}O$ are given by the following equation:

$$[X(n)\ Y(n)\ Z(n)] = [x_n.SZx/{}^{I}SZ_nx \quad y_n.SZy/{}^{I}SZ_ny \quad d_n] \quad (8)$$

where *SZx and SZy* are the object sizes along the x and y dimension respectively. The 3D point calculated lies in the camera plane.

5.For graphical representation, the X and Z coordinates are used to denote the horizontal displacement and depth of the object respectively, the vertical displacement of the object i.e. the Y coordinate is not taken into consideration at present. Its utility will be later understood while discussing case (b). As the camera is fixed for all the image frames, all the n points *(X (n), Z (n))* lie on the same XZ plane and thus can be plotted to identify the trajectory generated by the object.

(b) Camera moving, object stationary

When a moving camera captures a video of a fixed object then the displacements (change in centroid position) of the object observed in the image frames is due to the movement of the camera from frame to frame. This camera movement is calculated from these centroid displacements for identifying the trajectory generated by the camera. Initially the object is detected and the image sizes, depth and 3D Coordinates are calculated using equations 7, 3 and 8 respectively. The 3D points calculated for each frame lies on a different camera plane as the camera is in constant motion. The camera is considered to be positioned at the origin of a fixed reference frame for

the first image frame. The 3D coordinates of the first and nth frame can be related by rotation and translation matrices as shown in equation 10, where the rotation matrix denotes the camera rotation of the nth frame relative to the first frame and the translation matrix denotes the translation of the camera from the fixed origin. The experimental results given later in this paper use a set-up where the camera is fixed on a tripod mounted on a trolley. Thus only pan angle change is considered in the calculations. Using the selected corners (x_n1, y_n1) and (x_n2, y_n2) calculated in Step 2, the pan angle θ can be calculated using equation 9.

$$\theta = \tan^{-1}\left(abs(y_n2 - y_n1) / abs(x_n2 - x_n1)\right) \quad (9)$$

The affine transformation of the camera from the fixed frame to the nth frame is given by:

$$\begin{bmatrix} X(n) \\ Y(n) \\ Z(n) \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} X(1) \\ Y(1) \\ Z(1) \end{bmatrix} - \begin{bmatrix} x'(n) \\ y'(n) \\ z'(n) \end{bmatrix} \quad (10)$$

If $y_nlb < y_nrb$ then θ is positive else it is negative. It is clear from equation 10 above that (x_n', y_n', z_n') is the translated origin of the camera plane in the n^{th} frame and hence the required camera displacement. Equation 10 can be written as

$$x_n' = X(1)\cos\theta - Z(1)\sin\theta - X(n),$$
$$z_n' = X(1)\sin\theta + Z(1)\cos\theta - X(n) \quad (11)$$

Once (x_n', z_n') is calculated using equation 11, plotting it for all the n image frames gives the calculated camera trajectory.

4. RESULTS AND PERFORMANCE ANALYSIS

The trajectory generation for moving objects or moving camera has been accomplished using the proposed method and some of the results are shown.

(i) Experiment 1:

Figure 7 represents a scene where the object is fixed and the camera is in motion.

Figure 7: White mark in the center shows the path followed by the camera mounted on a trolley. The track line was created using a white marker while pushing the trolley at an approximate constant speed.

The generated plot shown in figure 8 is a smoothed plot using the best polynomial fit. The best fits of the plot are estimated using the norm of residuals of the fits and are again crosschecked by determining the R-Square values for each fit. The observed values are shown in Tables 1 and 2 respectively.

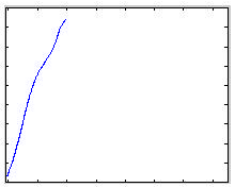

Figure 8: The calculated trajectory of the path shown in figure 7 using the present approach

TYPE	ORDER	NORM	StD
Poly	2	35.3552	2.2405
Poly	4	27.24	1.72
Poly	6	25.43	1.61
Poly	8	21.2	1.34
Poly	10	21.2	1.34

TABLE 1: Norm of Residuals.

It is seen that the norm of residuals converges after the eighth order fit. Coefficient calculation with 95% confidence bound and normalization by a mean of 5.833 and StD of 5.753 gives the corresponding R-square values.

TYPE	ORDER	SSE	R-SQUARE
Poly	2	1249.9	0.9921
Poly	4	741.81	0.9953
Poly	6	646.75	0.9959
Poly	8	452.557	0.9972
Poly	10	449.553	0.9972

TABLE 2: R-Squares values

Though the SSE values and the standard deviation decreases as the order of the fit increase, but the goodness of the fit (judged by the R-Square value) remains same after the 8^{th} order fit. Hence for both the best-fit estimation techniques the eighth order fit is the optimal fit for the curve.

(ii) Experiment 2:

The next case shows generated object trajectories when the camera tracks a moving object from a fixed position.

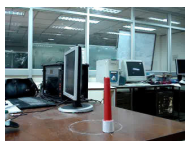

Figure 9: White circular path depicts the path followed by the red object

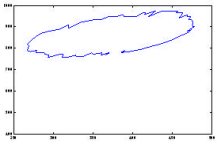

Figure 10: The generated trajectory of the path in fig 9

Similar to the previous case, a moving average filter is used to smooth the plot. It is seen that the three point averaging filter gives the best fit. The fact is further demonstrated in table 3 .

TYPE	NORM	StD	SSE	R-SQUARE
3 point	50.73	4.15	2574.5	0.9971
5 point	64.92	5.31	4215	0.9952

TABLE 3: Best-Fit Estimation

It is observed that the residuals diverge from 3-5 point averaging.

(iii) Experiment 3:

The following experiment was carried out with the All Terrain Robot developed at CMERI Durgapur during its testing on the grounds of the institute. The figures depict the identified trajectory (depicted by the blue colored plot) of the path traversed by the ATR.

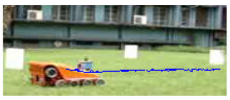

Figure 11a: Trajectory identified after rejecting outlier data using averaging window of 4σ gate

Figure 11b: Trajectory identified after rejecting outlier data using 6σ gate

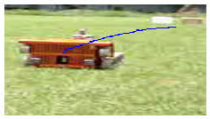

Figure 11c: Trajectory identified after rejecting outlier data using averaging window of 4σ gate

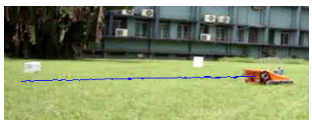

Figure 11d: No outlier detected

The statistical cut-off values were selected after estimating the rejection percentage for a gate of 3σ, 4σ, 5σ and 6σ for figures 11a,b and c. Table 4 shows the rejection rates for the figures. A 7% rejection was considered to be the maximum allowable rejection rate and choice of the cut-off was made accordingly.

Figures/Gates	3 σ	4 σ	5 σ	6 σ
11a	8.25	6.5	6	5.5
11b	84.14	59.14	7.85	2.1
11c	23.625	1.125	1	1

TABLE 4: Rejection rates for different statistical cut-off gates

5. DISCUSSIONS AND CONCLUSION

This paper presents an odometric navigation using uncalibrated camera images. The proposed methodology relies on a simple but elegant approach for consistent feature detection using GFV method. These features are then used for generation of visual odometry of any mobile robot. The indoor and outdoor field experiments show that this is a more resilient and computationally efficient approach which can be used to resolve navigation problems. Work is in progress for online implementation of this methodology for autonomous navigation of an unmanned aerial robot project currently pursued by CMERI.

6. REFERENCES

1. H. Moravec , Visual Mapping By A Robot Rover, In Proceedings Of The International Joint Conference On Artificial Intelligence (Ijcai) 1979, 598–600.
2. Canny, A Computational Approach To Edge Detector, IEEE Transactions On Pami, 1986, Pp679-698,
3. D. Cremers, and C. Schnorr, Statistical Shape Knowledge In Variational Motion Segmentation,,Israel Nent. Cap. J, 2003, 21, 77–86.
4. Withagen, Klamer Schutte1and Frans Groen, Likelihood Based Object Detection and Object Tracking Using Color Histograms, Proc. Icip2002, September 22-25, Rochester, New York
5. M. J. Nassiri, A. Vafaei, and A. Monadjemi, Pwaset , Texture Feature Extraction Using Slant-Hadamard Transform ,Volume 17 December 2006 Issn 1307-6884
6. C. Wren, A. Azarbayejani, and A. Pentland, 1997. Real-Time Tracking Of The Human Body, Pfinder IEEE Trans. Patt. Analy. Mach. Intell. 19, 7, 780–785
7. S.Park, and J.K Aggarwal, A Hierarchical Bayesian Network For Event Recognition Of Human Actions And Interactions. Multimedia System,2004, Volume-10,Issue- 2,Pages 164–179
8. Stan Birchfield and Carlo Tomasi, Depth Discontinuities by Pixel-to-Pixel Stereo, International Journal of Computer Vision, Volume 35 , Issue 3, December 1999,Pages: 269 – 293,ISSN:0920-5691.
9. Y. L. Murphey, J. Chen, J. Crossman, J. Zhang, P. Richardson,and L. Sieh, .Depth_nder, A Real-time Depth Detection System for Aided Driving, IEEE

Intelligent Vehicles Symposium Proceedings,October 2000.
10. Wietske I. Meyerind, Marco A. Gutierrez, Skrgio S. Furui, Marina S. Rebelo, Chdido P. Melo, Spatiotemporal-Frequency Analysis Applied to Motion Detection Proceedings of the 22"d Annual EMBS International Conference, July 23-28,2000, 1720-1723.
11. Murali Subbarao,Tai Choi,Arman Nikzad , Focusing Techniques, OE/Technology SPIE Conference
12. Mirzabaki Mahdi, A New Method for Depth Detection Using Interpolation Functions, WSCG posters proceedings, February 2-6, 2004, Plzen, Czech Republic
13. .J. Bhattacharya and S. Majumder ,The Generalized Feature Vector (GFV): A New Approach for Vision Based Navigation of Outdoor Mobile Robot, 14th National conference on Machines and Mechanisms(NaCoMM-2009), NIT Durgapur ,India, December 17-18, 2009

APPENDIX

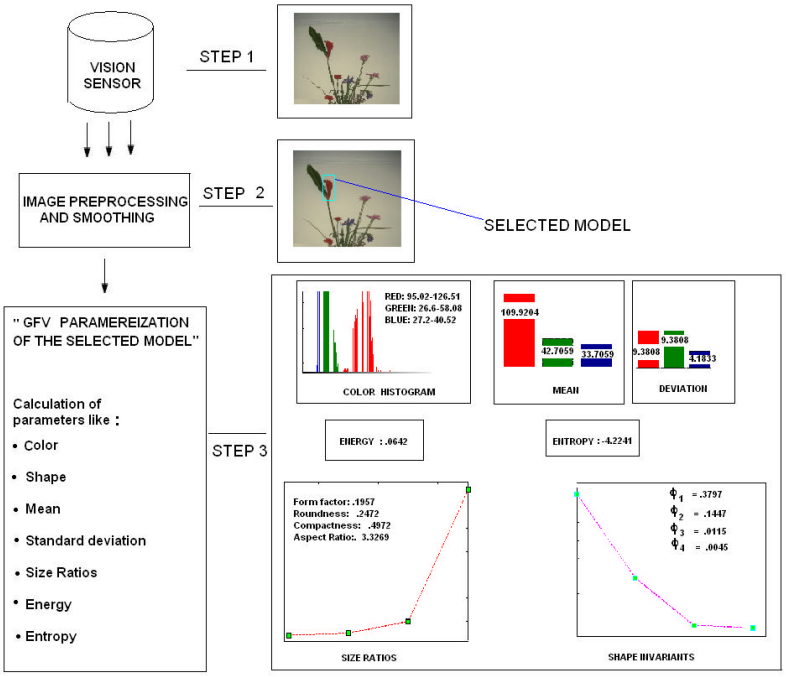

Figure 1: Reference model creation

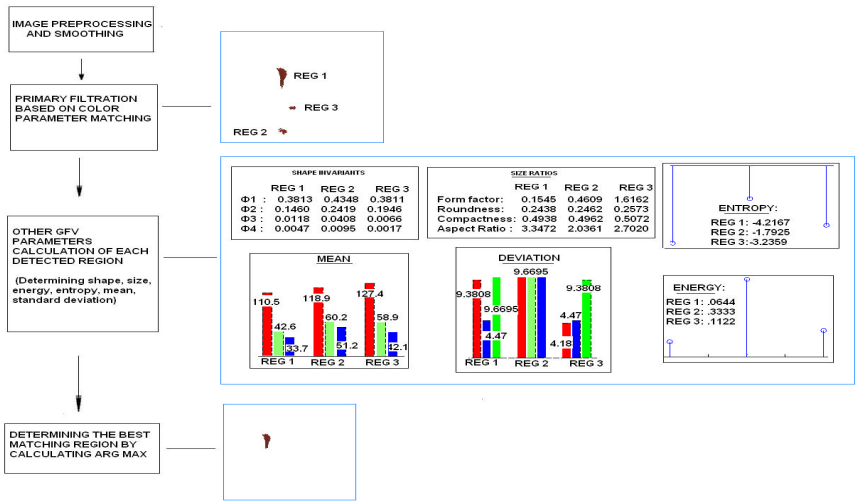

Figure 2: Application of GFV to experimental dataset.