2010 Second International conference on Computing, Communication and Networking Technologies

Real Time Vision Based Object tracking using CAMSHIFT Algorithm with enhanced Color Image Segmentation

Mrs M Mahalakshmi
PG Student, M.E., Embedded System Technologies, College of Engineering,
Anna University, Guindy,Chennai,INDIA
Email: msubastrie@yahoo.com

Abstract— In this paper we implement a vision based moving Object Tracking system with Wireless Surveillance Camera which uses a color image segmentation and color histogram with background subtraction for tracking any objects in non-ideal environment. The implementation of the moving video objects based on the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm is presented by optimizing the kernel variants by adjusting the HSV value for various environmental conditions. The object occlusions are also removed by calculating the minimal distance between the two objects using Bhattacharya coefficients and it is robust to changes in shape with complete occlusion. Based on the selection of the users Region of Interest (ROI) the HSV value of the object being tracked by means of CAMSHIFT algorithm which uses an Adaptive block based approach for continuous object tracking. A software approach for real time implementation of moving object tracking is done through MATLAB.

Keywords - Color Image segmentation,Bhattarcharya coefficient, CAMSHIFT Tracking and Wireless Link

I. INTRODUCTION

Tracking of humans is an important computer vision building block. It is needed for many applications, ranging from surveillance and military through computer-aided driving and advanced human-machine interfaces. The main challenges in human tracking are: (1) differentiating between background and foreground areas; (2) differentiating between the tracked object and other human objects on the same scene; (3) changes in lighting, which causes the appearance of the tracked object to change; (4) two-dimensional scaling of the tracked object, for scenes where the objects change their distance from the camera; and (5) occlusions of the tracked object by other objects.

The final tracker we devised consisted of a color-histogram continuously adaptive mean-shift tracker [1] based on the OpenCV [2] implementation, where the initial histogram accuracy has been improved by using a masked histogram instead of a box histogram. The mask was calculated by interactively allowing the user to place a predefined template on the initial human being tracked, and separating foreground from background using a gray-scale region growing [3] technique on both background and foreground areas. The tracker has been compared with the standard mean-shift tracker, where the initial histogram is based on calculating a box histogram for a selected region.

Related works are discussed in Section II and Section III describes the Image Segmentation and Background Subtraction Section IV describes about continuously adaptive Mean shift Tracking (CAMSHIFT) algorithm and Mean Square Difference. Section V describes about the CAMSHIFT system model Implementation details and its results are discussed in the in section V.

II. RELATED WORKS

Comparative assessment of segmentation algorithms is often based upon subjective judgment, which is qualitative and time consuming. Therefore, there is need for automatic, objective spatiotemporal measures, not only for comparison of overall algorithmic performance, but also as a tool to monitor spatiotemporal consistency of individual objects.

Recently, a number of video segmentation measures have been proposed in the presence of ground-truth. The main contribution of this work is to develop quantitative performance measures for video object tracking and segmentation, which do not require ground-truth segmentation maps. The proposed measures exploit color and motion features in the vicinity of the segmented video object. One of the features is the spatial color contrast along the boundary of each object plane. The second is color histogram differences across video object planes, which evaluates the goodness of segmentation along a spatiotemporal trajectory. The third feature is based on motion vector differences along the object plane boundary.

Often a single numerical figure does not suffice to evaluate the goodness of a segmentation/tracking for a whole video sequence. Since the spatial segmentation quality can change from frame to frame and/or, depending upon the scene content the temporal segmentation stability may deteriorate over subsequences, we propose additional measures to localize in time or in space the unsuccessful segmentation outcomes. An overall, in this paper we aim at formulating color-based deformable models to segment and track objects in video robust against noisy data and varying illumination. To achieve this, computational methods are presented to measure color constant gradients. Further, a model is proposed for the estimation of noise through these color constant gradients. As a result, the associated uncertainty is known for each color constant gradient value. The associated uncertainty is subsequently used to weight the color constant gradient during the deformation process. As a result, noisy and unstable gradient information will contribute less to the deformation process than reliable gradient information yielding robust object segmentation and tracking.

III. IMAGE SEGMENTATION

Image segmentation is a fundamental task in computer vision, and the application of segmentation to color images is used in a wide range of tasks, including content-based image retrieval for multimedia libraries [4], skin detection [5], object recognition [6], and robot control [7]. A variety of approaches to this problem have been adopted in the past, which can be divided into four groups: pixel-based techniques, such as clustering [4]; area-based techniques, such as split-and-merge algorithms [7]; edge-detection, including the use of color-invariant snakes [8]; and physics-based segmentation [9].

A review of the methods and applications of color segmentation is given in [7]. The approach to image segmentation adopted in this work relies on clustering of pixels in feature space using on-parametric density estimation. The subject of clustering, or unsupervised learning, has received considerable attention in the past and the clustering technique used here is not original [4]. However, much of the work in this area has focused on the determination of suitable criteria for defining the "correct" clustering. The method follows the formation of self-generating representations using knowledge of data accuracy [10], defining the size required for a peak in feature space to be considered an independent cluster in terms of the noise in the underlying image.

A. Colour image segmentation

The segmentation process maps pixels from an arbitrary number n grey-scale images into an n-dimensional grey-level space, and calculates a density function in that space. A color image can be represented as three grey-scale images, showing for instance the red, green and blue components of the image, although many alternative three-dimensional schemes have been proposed [11]. Therefore a color image will generate a 3D space.

The aim of the color segmentation routine described here was to identify distinctly colored regions in an image that corresponded to physical objects present in the scene. Some color spaces (e.g. HSI, YIQ) separate the achromatic (I, Y) and chromatic (HS, IQ) information onto different axes. Therefore, the achromatic information can be discarded and the segmentation performed on the remaining two chromatic dimensions. This confers the additional advantage of reducing the dimensionality of the problem, and so reducing the processor time required. This approach was tried with limited success [12].

A more effective way of removing the intensity information was found to be normalizing the RGB values prior to any color space conversions, using r = R/(R+G+B), g = G/(R+G+B), and b = B/(R+G+B). This is equivalent to finding the intersection of the color vectors in RGB space with the plane of constant intensity passing through (1,0,0), (0,1,0) and (0,0,1). It also retains the advantage of reducing the dimensionality of the color space from three to two, since r +g +b = 1and so any two of these components is sufficient to describe the normalized color vector. It is therefore desirable to use the color space that has the simplest possible error propagation from RGB. Therefore a new color space referred to as IJK was developed, a simple rotation of the RGB color space with no scaling, such that one axis lay along the vector R = G = B, and so represented the intensity I. The second axis J lay along the projection of the R axis onto the plane normal to the intensity axis, and the third axis K was perpendicular to the others. The conversion from RGB was performed using the rotation matrix. When this rotation was applied to the normalized RGB space, the values for the intensity axis I was uniform across the image as expected. In practice, any arbitrary set of perpendicular axes in the normalized color space can be used in the segmentation.

$$\begin{pmatrix} I \\ J \\ K \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{2}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{1}$$

The algorithm was tested with all combinations of pairs of r, g and b; with the hue and saturation fields from HSI; with the I and Q fields from YIQ; and with the J and K fields from the new color space. Ignoring differences due to error propagation, no significant advantage of one of these choices over the others was found.

The first step in the segmentation was to map the pixels $X_i = 1, N_p$ from the original images into color space, producing an n-dimensional scatter gram $S(g_1, n)$ where grey levels g_1, N_p of pixels at the same positions in the images are used as coordinates in feature space. To map out the whole space would be prohibitively expensive in terms of memory and processor time in high dimensional problems, and so the data itself was used to define a list of N_k knot points $G_i = 1, N_k \langle N_p$ that span the space. The traditional problem of clustering algorithm has been the definition of the "correct" clustering i.e. the correct scale at which to define peaks in the color space. If clustering is performed at too small a scale, then artificial peaks due to noise will be generated. Conversely, if the scale is too large then small but salient peaks will be absorbed into nearby, larger peaks. One approach to this problem has been to perform the segmentation at a range of scales, and then select the correct scale by applying some measure of the quality of clustering [4] to the resultant images.

B. Background subtraction

We have initially tried to use background subtraction combined with the segmentation approach suggested by [13] to determine the vector of movement in the scene. The method described in [13] suggests accumulating foreground pixels over time, where the pixel's age is represented by its grayscale value; then, calculating the vector of movement in the scene is done by calculating the gradient on the accumulated history image.

In order to segment foreground and background, we subtracted from every pixel the weighted average of its history as shown in the, and labeled the pixel according to some threshold: if the subtraction yielded an absolute value higher than the threshold it was labeled foreground. Once foreground and background were segmented, we applied the method from [13] segmentation

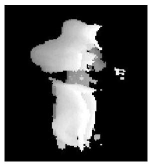

Figure 1. Output frame from history algorithm. As the grayscale gradient suggests, objects is moving to the right.

$$F_{i,j}^{curr} = \begin{cases} label\ (i,j) = fg \Rightarrow white \\ label\ (i,j) = bg \Rightarrow F_{i,j}^{prev} - c \end{cases} \quad (2)$$

with motion history gradients determine the vector of movement in the scene by preserving the short-term history of foreground pixels in any given frame: every foreground-labeled pixel was colored white (in the single-channel output frame) and for every pixel labeled background; we subtracted its previous grayscale value by a constant c as the equation in figure 1 suggests. Thus, every nonzero pixel in each single-channel output frame describes a foreground pixel at some point in "history". The gradient of the grayscale image is then equivalent to the vector of movement in the last c frames (see figure 1).This method can yield an understanding of the vector of movement in the last c frames of the scene, but it lacked the ability to differentiate between multiple objects in the scene. As soon as we introduced more than one object (or moving background elements), there was no way of telling which one was the object being tracked. This method can be used as an auxiliary method to some other tracking mechanism (to determine vector of movement and facilitate in path initialization for example), or in applications where a single object is being viewed in the scene (e.g. the application from [13]-a music synthesis program). Therefore, we have decided not to use it for our current implementation.

IV. CAMSHIFT TRACKING ALGORITHM

CAMSHIFT stands for the "continuously adaptive mean-shift" algorithm. Figure summarizes this algorithm. For each video frame, the raw image is converted to a color probability distribution image via a color histogram model of the color being tracked, e.g., flesh color in the case of face tracking. The center and size of the color object are found via the CAMSHIFT algorithm operating on the color probability image. The current size and location of the tracked object are reported and used to set the size and location of the search window in the next video image. The process is then repeated for continuous tracking. The algorithm is a generalization of the Mean shift algorithm, highlighted in gray in figure.

A. Continuously Adaptive Mean Shift tracking Algorithm

When using real cameras with discrete pixel values, a problem can occur when using HSV space as can be seen in Figure 3. When brightness is low (V near 0), saturation is also low (S near 0). Hue then becomes quite noisy, since in such a small Hexcone, the small number of discrete hue pixels cannot adequately represent slight changes in RGB. This then leads to wild swings in hue values. To overcome this problem, we simply ignore hue pixels that have very low corresponding brightness values. This means that for very dim scenes, the camera must auto-adjust or be adjusted for more brightness or else it simply cannot track. With sunlight, bright white colors can take on a flesh hue so we also use an upper threshold to ignore flesh hue pixels with corresponding high brightness. At very low saturation, hue is not defined so we also ignore hue pixels that have very low corresponding saturation. The CAMSHIFT part of the Algorithm is as follows which is shown in Figure 2 also.

1. Choose a search window size.
2. Choose the initial location of the search window.
3. Compute the mean location in the search window.
4. Center the search window at the mean location

   computed in Step 3.
5. Repeat Steps 3 and 4 until convergence (or until the mean location moves less than a preset threshold).

We are using the scheme implemented in OpenCV in which the mean-shift search is performed using a predefined number of iterations, where every iteration. OpenCV uses spatial moments to calculate a new center of mass on the back projection.

B. Color probability Distributions

When using real cameras with discrete pixel values, a problem can occur when using HSV space as can be seen in Figure 3. When brightness is low (V near 0), saturation is also low (S near 0). Hue then becomes quite noisy, since in such a small hexcone, the small number of discrete hue pixels cannot adequately represent slight changes in RGB. This then leads to

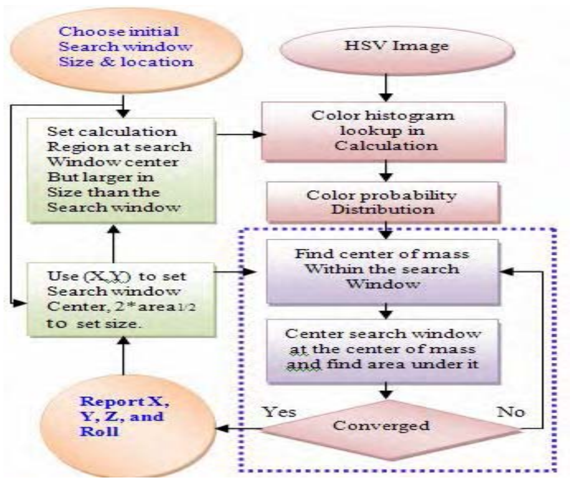
Choose initial Search window Size & location
HSV Image
Set calculation Region at search Window center But larger in Size than the Search window
Color histogram lookup in Calculation
Color probability Distribution
Find center of mass Within the search Window
Use (X,Y) to set Search window Center, 2*area1/2 to set size.
Center search window at the center of mass and find area under it
Report X, Y, Z, and Roll
Yes
Converged
No

Figure 2. CAMSHIFT Tracking Algorithm

wild swings in hue values. To overcome this problem, we simply ignore hue pixels that have very low corresponding brightness values. This means that for very dim scenes, the camera must auto-adjust or be adjusted for more brightness or else it simply cannot track. With sunlight, bright white colors can take on a flesh hue so we also use an upper threshold to ignore flesh hue pixels with corresponding high brightness. At very low saturation, hue is not defined so we also ignore hue pixels that have very low corresponding saturation.

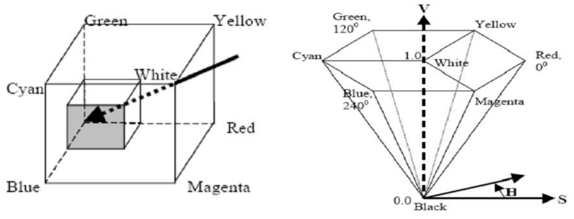

Figure 3. RGB Color and Hex cone based HSV

C. *Bhattacharyya Coefficient*

Bhattacharyya coefficient between '*c*' and '*r*' is defined as

$$\hat{\rho}(y) \equiv \rho[\hat{c}_u(y), \hat{r}_u] = \sum_{u=1}^{m} \sqrt{\hat{c}_u(y)\hat{r}_u} \tag{3}$$

The similarity function inherits the properties of the kernel profile when the target and candidate histograms are represented according to 'p' and 'q'. A differentiable kernel profile yields a smooth differentiable similarity function. It is expected that the maximum of this function should be at the position of the moved object or the similar object in the subsequent frame or image. Smoothness of the function makes it possible to search the maximum using any gradient based search algorithm, but here we are not concerned with the methodology or efficiency of automatic search. We are in fact interested in the accuracy of finding the position of the object.

D. *Object Representation*

To characterize the object, first a feature space is chosen. The object is represented by its probability density function (pdf). The pdf can be estimated by m-bin histogram of object, where m is the number of colors. The histogram is not the best nonparametric density estimate [14], but it is good enough for most pattern recognition applications. Other discrete density estimates can also be employed. The reference object is the one to be searched in the same image or may be in next image of a video sequence or in any image where a similar object may be found. The candidate objects are tested against the reference object to check the similarity between them. Both the reference and the candidate objects are represented by m-bin histograms as an estimate to their pdf's. Both the pdf's are to be estimated from the data.

$$\hat{r} = \{\hat{r}_u\}_{u=1\ldots m} \qquad \hat{c} = \{\hat{c}_u(y)\}_{u=1\ldots m} \tag{4}$$

Where $\hat{r}$ and $\hat{c}$ represent the m-bin histograms of reference object and the candidate object at location y, respectively.

E. *Mean Square Difference (MSD)*

MSD is an accurate matching criterion because of its spatial nature. Its problem is the lack of robustness due to various reasons; a brief account of which follows. MSD may not give good results with significant changes in illumination of the object. It also experiences difficulties if the size or orientation of the object is rapidly changing. Finally, MSD may completely breakdown under occlusions. Due to these reasons, MSD is not a good practical solution. Its narrow peak and numerous local maxima make it difficult for gradient based search methods to be used to find the maximum. However, here we are not concerned with the efficient automatic search, so full exhaustive search may be used. Nevertheless, we can still use it to assess the performance of other criteria because the maximum of this function indicates high similarity based on the gray level of pixel intensities. In doing so, we will have to make sure that we avoid the cases that are not handled well with MSD. The expression for the MSD is given as

$$MSD = \frac{1}{n^2}\sum_{i=1}^{n}(X_i - Y_i)^2 \tag{5}$$

Where X_i and Y_i are the corresponding pixels of the adjacent object windows.

F. *Comparison of MSD and Bhattacharyya Coefficient*

The sharp peak of MSD gives exact coordinates of slightly moved or transformed object. Sharpness of the peak is not adequate for the application of gradient based optimization methods. Bhattacharyya coefficient, through a differentiable kernel, yields a fairly smooth function, but target localization by this curve is problematic due to its biased nature. In the experiments we compare the peaks of Mean Square Difference and Bhattacharyya coefficient functions and observe that there is a fairly large difference between the two

G. *Distance minimization*

Based on the fact that the probability of classification error is directly related to the similarity of the two distributions, the choice of the similarity measure in [14] was such that it was supposed to maximize the Bayes error arising from the comparison of target and candidate pdf's. Being a closely related entity to the Bayes error, a Bhattacharyya coefficient was chosen and its maximum searched for to estimate the target localization. Bhattacharyya coefficient of two statistical distributions is defined as

$$\rho[p(y),q] = \int \sqrt{p_z(y) q_z} dz \qquad (6)$$

V. CAMSHIFT- System model Implementation

In order to track the object using CAMSHIFT algorithm we have implemented an Wireless vision interface model for acquiring the real time images from remote place through JK wireless Surveillance camera which is interfaced with Matlab for tracking the objects based user defined region of Interest. To grab the image we have using Zebronics image grabber TV tuner for acquiring the image from wireless camera through 2.5GHz video receiver module interfaced with PC. Complete module shown below in the Figure 4.To interface the Image grabber and Matlab we have used a Dynamic Link Library files vcapg2.dll for acquiring the real time images from remote location using MATLAB.

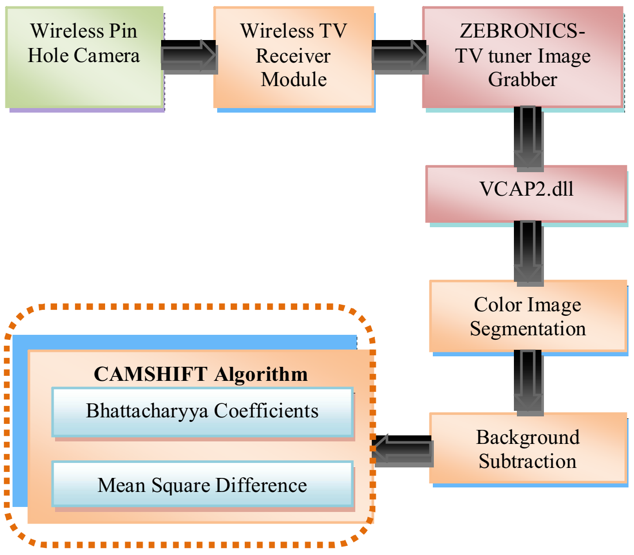

Figure 4. CAMSHIFT System Model

A. *Initial Window Size and Placement*

In practice, we work with digital video images so our distributions are discrete. Since CAMSHIFT is an algorithm that climbs the gradient of a distribution, the minimum search window size must be greater than one in order to detect a gradient. Also, in order to center the window, it should be of odd size. Thus for discrete distributions, the minimum window size is set at three. For this reason too, as CAMSHIFT adapts its search window size, the size of the search window is rounded up to the current or next greatest odd number. In practice, at start up, we calculate the color probability of the whole scene and use the zeroth moment to set the window size and the centroid to set the window center.

B. *Setting Adaptive Window Size Function*

Deciding what function of the zeroth moment to set the search window size of the CAMSHIFT algorithm depends on an understanding of the distribution that one wants to track and the goal that one wants to achieve. The first consideration is to translate the zeroth moment information into units that make sense for setting window size. Our goal is then to track the whole color object so we need an expansive window. Thus, we further multiply the result by two so that the window grows to encompass the connected distribution area. We then round to the next greatest odd search window size so that the window has a center.

For 2D color probability distributions where the maximum pixel value is 255, we set window size *s* to

$$s = 2 * \sqrt{\frac{M_{00}}{256}} \qquad (7)$$

We divide by 256 for the same reason stated above, but to convert the resulting 2D region to a 1D length, we need to take the square root. In practice, for tracking faces, we set window width to *s* and window length to *1.2s* since faces are somewhat elliptical.

C. *Comments on Software Calibration*

Much of CAMSHIFT's robustness to noise, transient occlusions, and distracters depends on the search window matching the size of the object being tracked—it is better to err on the side of the search window being a little too small. The search window size depends on the function of the zeroth moment M_{00} chosen above. To indirectly control the search window size, we adjust the color histogram up or down by a constant, truncating at zero or saturating at the maximum pixel value. This adjustment affects the pixel values in the color probability distribution image which affects M_{00} and hence window size. For 8-bit hue, we adjust the histogram down by 20 to 80 (out of a maximum of 255), which tends to shrink the CAMSHIFT window to just within the object being tracked and also reduces image noise. HSV brightness and saturation thresholds are employed since hue is not well defined for very low or high brightness or low saturation. Low and high thresholds are set off 10% of the maximum pixel value.

D. *Comments on Hardware Calibration*

To use CAMSHIFT as a video color object tracker, the camera's field of view (zoom) must be set so that it covers the space that one intends to track in. Turn off automatic balance if possible to avoid sudden color shifts. Try to set (or auto-adjust) AGC, shutter speed, iris or CCD integration time so that image brightness is neither too dim nor saturating. The camera need not be in focus to track colors. CAMSHIFT will work well with cheap cameras and does not need calibrated lenses

E. *Target localization*

By target localization we mean finding the spatial coordinates of the object in the image or frame of interest.

These coordinates can be found using some similarity measure. The estimate of target location is the maximum value of this similarity measure.

VI. RESULT

In order to have better object tracking results we use software based threshold values. For better results GAMMA & THRESHOLD parameter is implemented along with the CAMSHIFT algorithm for better tracking and removal of noise particle. The performance analysis of these two parameters is discussed below with various images acquired from the wireless camera shown in Figure 5, 6, 7 consecutively. In fact the tracking of an object also depends on the distance and positioning of the wireless camera. In this paper we have tested the tracking on Objects such as Compact Disc and human tracking. For testing purpose we have tested the algorithm with real time with an Image Size of 320x240. CAMSHIFT tracking purely depends on the size of the images and center of mass.

The performance analysis of these two parameters is discussed below in the TABLE I with various images acquired from the wireless camera shown in Figure 5, 6, 7 consecutively and its statistical data for various HSV - Value and Threshold values are shown in Figure 5, 6 and 7 respectively.

TABLE I. PERFORMANCE ANALYSIS

Threshold	**HSV V -Parameter**	**Result**
0.1	12.725	Performance of the tracking is good
0.1	6.45	Noise due to Gamma factor tracking performance is fair
0.40	12.725	Tracking is not Possible
0.40	6.45	Tracking is not possible

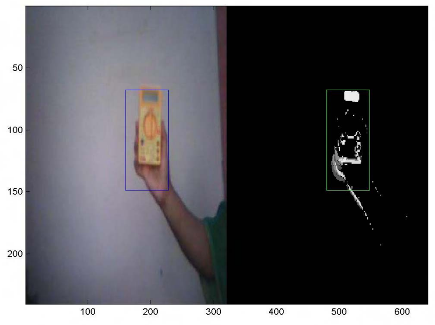

Figure 5. Performance of the tracking is good

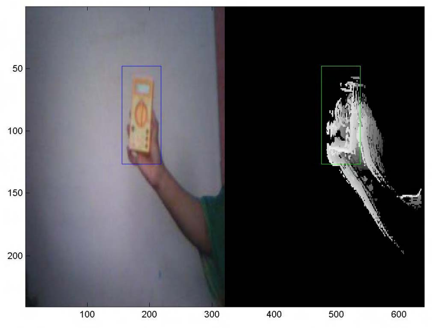

Figure 6. Noise due to Gamma factor tracking performance is fair

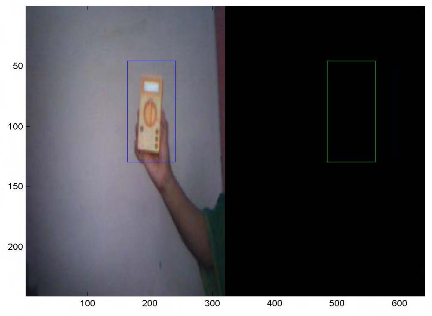

Figure 7. Tracking is not possible

The center of mass is estimated by calculating the size of the region of interest area. Complete tracking is computed by acquiring the new position of the object and its new center of mass values hence by adapting the new position repeatedly the object is tracked continuously. We tested the algorithm on various images CD tracking, Child's Head tracking, and Multimeter Tracking under various lightening conditions based on the user's region of Interest and New positions are object displace from its initial position's obtain the continuous adaptive tracking the new position is now initiated as initial position and so on whose illustrations shown in the Figure 8, 9, 10 respectively tracking positions based on region of interest.

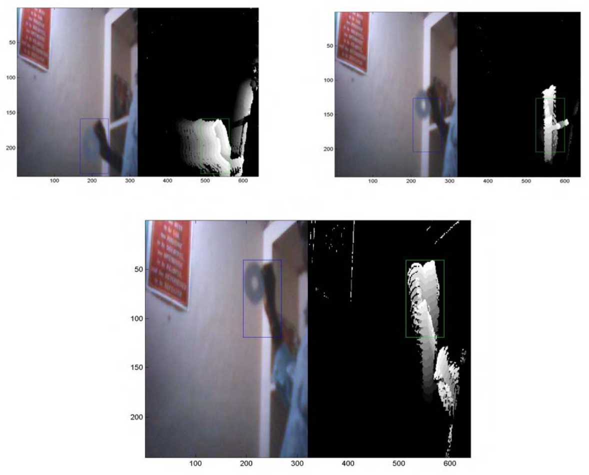

Figure 8. Illustration of Compact Disc Tracking

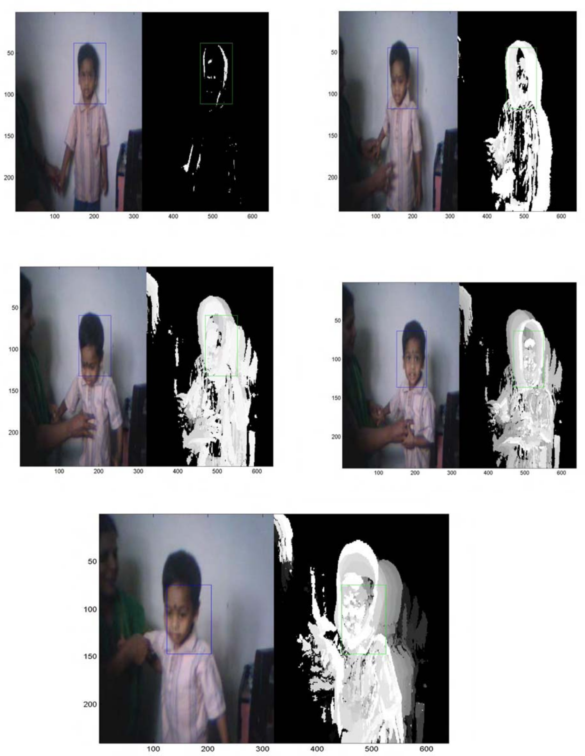

Figure 9. Illustration of Tracking Head of a Child

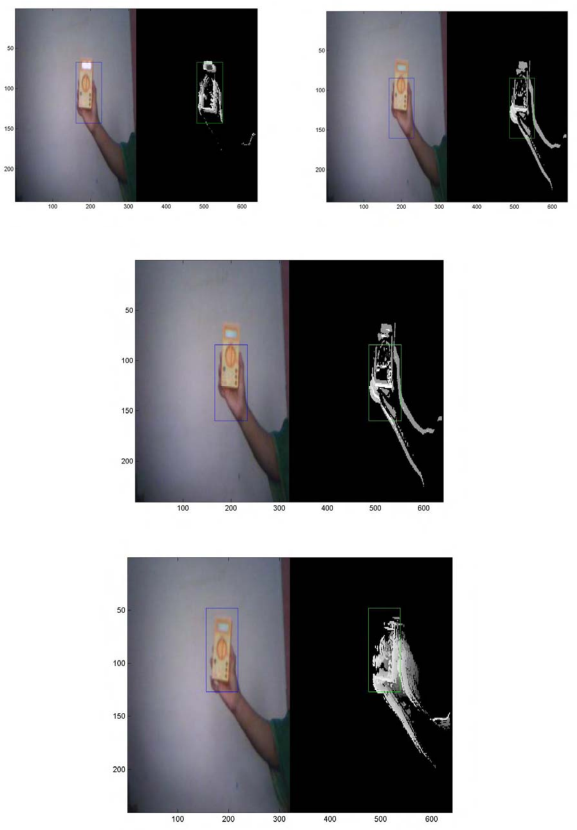

Figure 10. Illustration of Multimeter tracking

All the results have been computed by varying the HSV value particularly the V-paramters which plays an vital role during tracking any objects.This particular parameter has to be adjusted depending upon the ligtening conditions under various environment.

VII. CONCLUSION

In this paper we have explored the use of variable kernels to enhance color image segmentation and back ground subtraction methods for removing the noise under various environmental conditions. HSV value of the object to be tracked is tuned for fine values of the images acquired from wireless camera. Experimental results show the improved tracking capability and versatility of our implementation of mean-shift object tracking algorithms when compared with results using the standard kernel. The CAMSHFIT tracker along with color image segmentation would be a very effective and efficient solution for video tracking

VIII. FUTURE WORK

By processing real-time images and communicating wirelessly in outdoor environments, we can track moving objects against complex, cluttered backgrounds. Further work is currently underway to extend for tracking multiple objects at same time by enhancing the Bhattacharyya coefficients. Sometimes the objects would be lost when the object is out of frames to track the object from lost we can implement a pan tilt mechanisms for tracking continuously.

ACKNOWLEDGMENT

The authors gratefully acknowledge the following individuals for their support: Prof. Mr. Mohan Head of the department, Department of Electrical and Electronics Engineering, Anna University, Ms. K. Deepa, Lecturer, Department of Electrical and Electronics Engineering, Anna University and Mr. M. Manigandan, Senior Lecturer, Velammal Engineering College, and to my parents and friends for their valuable support for devoting their precious time, sharing their knowledge and co-operation.

REFERENCES

[1] D. Comaniciu and V. Ramesh, Real Time Tracking of Non-Rigid Objects using Mean Shift, Computer vision and Pattern Recognition, 2000 proceedings IEEE conference.

[2] OpenCV Library, Intel Corporation.

[3] D.Marshall, Region Growing, Vision Systems,1994.

[4] E.Pauwles and G.Frederix. Non-Parametric Clustering for Image Segmentation and Grouping. Computer Vision and Image Understanding, 75 nos. 1/2, pp. 73-85, 1999.

[5] Y.Fang and T.Tan, A Novel Adaptive Colour Segmentation Algorithm and its Application to Skin Detection. In Proc.BMVC 2000, pp.23-31. BMVA, 2000.

[6] G.D.Finlayson and G.Y.Tian. Colour Normalisation for Colour Object Recognition. International Journal of Pattern Recognition and Artificial Intelligence, pp. 1271-1285, 1999.

[7] W.Skarbek and A.Koschan. Colour Image Segmentation : A Survey. Technischer Bericht 94-32, Technical University of Berlin, 1994.

[8] T.Gevers, S. Ghebreab, and A.W.M. Smeulders. Colour Invariant Snakes. In Proc. BMVC 1998, pp. 578-588. BMVA, 1998.

[9] G.J.Klinker, A. Shafer, and T. Kanada. A Physical Approach to Colour Image Understanding. International Journal of Computer Vision, 4, pp. 7-38, 1990.

[10] N.A. Thacker, I. Abraham and P. Courtney. Supervised Learning Extensions to the CLAM Network. Neural Networks, 10 no. 2, pp 315-326, 1997.

[11] G. Wyszecki and W.S. Stiles. Color Science: Concepts and Methods, Quantitative Data and Formulae (2nd Edition) John Wiley and Sons, New York, 1982.

[12] P.A. Bromiley, P. Courtney and N.A. Thacker. A Case Study in the use of ROC Curves for Algorithm Design. In Proc. BMVC 2001, BMVA, 2001.

[13] G. R. Bradski and J. Davis, Motion Segmentation and Pose Recognition with Motion History Gradients, Machine.

[14] D. Comaniciu, V. Ramesh, P. Meer, “Kernel-Based Object Tracking”, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.25, No. 5, 2003.