Human Computer Interaction (HCI) using Object as Gesture Input

Saket Joshi, Shounak Gujarathi, Abhishek Mirge
BE Computer, K.K.W.I.E.E.R, Pune University, Nasik, India

***Abstract*— Human Computer Interaction (HCI) has become one of the most active research areas in the past few years. Many existing techniques are based on hand gesture and usage of dedicated hardware. However, these methods have limited response because they either are costly or have low efficiency. In this paper, we propose handling basic Windows OS operations using gesture input in which the object is decided by the user and based on its trajectory, we detect gestures. These gestures are used as a real time input to the windows operating system based on which the assigned tasks are performed. The choice of assigning a specific gesture to a specific operation is customizable to the user. The proposed algorithms are CAMshift algorithm and SURF algorithm. CAMshift is a fast and effective algorithm for real-time object tracking based on its color. The event of object loss is detected using Bhattacharya distance method and to re-detect the object we use the SURF algorithm which is gray scale based and transformation invariant. This proposed method aims at achieving a low cost and more efficient secondary input based solution for HCI as compared to existing technologies.**

***Keywords*—CAMshift, gesture, object detection, SURF, tracking**

I. INTRODUCTION

Target tracking is attracting increasing attention in fields of video and image processing due to the prospect of wide applications and significant role it plays in industrial applications. Object tracking can be used for Human Computer Interaction. Object tracking can be divided into series of steps, such as object representation, feature selection for tracking, object detection, background subtraction, and object segmentation. Object trackers can be categorized into three different categories such as point trackers, kernel trackers and silhouette trackers. Point tracking methods can be further sub-categorized into deterministic methods such as [2] and statistical methods [3]. Kernel based object tracking methods are Mean Shift tracking [4], and Continuously Adaptive Mean Shift tracking (CAMshift) [1,5]. Eigen-tracking [6] and Support Vector Machines [7] are kernel based methods for multi-view appearance models. Silhouette tracking methods are divided into contour evolution methods [8] and shape matching methods [9]. CAMshift (Continuously Adaptive Meanshift) is a target tracking algorithm proposed by Bradski [10] that uses color histogram as its target model. This algorithm may not be easily influenced by the changes in the shape of the target as the histogram of a target picture is the recording of color probability and hence can effectively solve the problem that the target is moving or partly sheltered. Besides, this algorithm can achieve the purpose of target tracking fast, since it can reduce computation time by using the gradient descent method. CAMshift solely relies on back projected probabilities therefore it can fail in cases when the object's appearance changes (e.g., due to object or camera movement, or due to illumination and some other environmental changes), when similarly colored objects have to be re-detected or the target and background are alike in color or the target might be lost in dynamic background

In this paper, we propose an idea of using Object detection and Tracking algorithm for Human Computer Interaction. For the purpose of object detection and tracking we are using Improved CAMshift and SURF algorithms. The traditional CAMshift algorithm only uses the color information, while other factors, such as motion information and texture features, are not considered at all. Previous studies [11,12] all use SIFT (Scale-invariant feature transform) as matching algorithm which is slow in computation. We propose a method to judge whether the target in CAMshift is lost or not. Once the target is lost, we can re-match the target using SURF (the Speeded Up Robust Features) algorithm. SURF is a feature-matching algorithm which has the characteristic that it can remain invariant to target rotation, scale zoom and brightness variations. The proposed method is able to accurately locate and track targets. This method is significantly superior to traditional CAMshift algorithm, and has good robustness as well. We describe the CAMshift algorithm along with the improvement in section II. The SURF algorithm is explained in section III. Section IV explains how to integrate CAMshift and SURF algorithms. Section V describes the application which is being developed that will use these algorithms for Human Computer Interaction.

II. CAMSHIFT

CAMshift (Continuously Adaptive Meanshift) is based on a robust non-parametric technique for climbing gradient to find the mode (peak) of probability distributions called Meanshift. Meanshift algorithm is based on the characteristics of the distribution of all kinds as well as the color information searches centre point of the object. Meanshift algorithm doesn't update the size of search window and prone to converge to local maximum. These demerits are corrected in CAMshift algorithm.

A. CAMshift Algorithm

CAMshift uses color information to track the moving target and is mainly based on Back Projection calculation and Mean Shift algorithm [13, 14]. The color histogram is used to show the color distribution (information) in the image. CAMshift uses color histogram as its target model. Suppose an image contains n number of pixels and each pixel has associated with a set of values of Hue (H), Saturation (S) and Value (V). Let us consider the histogram to be an m-bin histogram meaning every partition in the histogram contains m number of samples. The H value of each pixel is counted into the histogram only if the S and V values of that pixel are under the initial thresholds; else the H is counted directly as 0. We define the H, S and V values of each pixel as $\{h_i\}_{i=1...n}$,$\{s_i\}_{i=1...n}$ and $\{v_i\}_{i=1...n}$, the histogram is $\{q_k\}_{k=1...m}$, and define a function c: $R^2 \rightarrow \{1...m\}$ that associates to the pixel at location x^*_i and the histogram bin index $c(x^*_i)$.Then we compute the color histogram as : q_k=

$$\sum_{i=1}^{n} \delta[c(h_i) - k]\, u\,[s_i - s_{min}]\, u\,[v_i - v_{min}]\, u\,[v_{max} - v_i]. \quad (1)$$

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}. \quad (2)$$

$$\delta(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}. \quad (3)$$

The color probability distribution of a part of an image represents the probability of it being the target. For better observation, every value in the function is multiplied by 255 to make up a gray image. This is known as Back Projection. Back projection is generated by the color histogram. Let the back projection value of each pixel be $\{b_i\}_{i=1...n}$, then the formula of computing back projection is

$$b_i = \frac{q_c(h_i)}{n} * 255. \quad (4)$$

CAMshift uses an elliptical search window and the calculation of window's location is a converging process. Each time the new window is computed according to the window of the last frame, and the new window is located when the difference of the two windows is below a pre-set threshold. The specific procedure in mathematical form is shown below, where I(x, y) is each pixel's value in back projection.

0-order moment: $M_{00} = \sum_x \sum_y I(x,y).$ (5)

1-order moment: $M_{01} = \sum_x \sum_y yI(x,y).$ (6)

$M_{10} = \sum_x \sum_y xI(x,y).$ (7)

2-order moment: $M_{20} = \sum_x \sum_y x^2 I(x,y).$ (8)

$M_{11} = \sum_x \sum_y xyI(x,y).$ (9)

$M_{02} = \sum_x \sum_y y^2 I(x,y).$ (10)

Now the centre's coordinate of the search window is

$$x_c = \frac{M10}{M00} \qquad y_c = \frac{M01}{M00} \quad (11)$$

As the search window is ellipse-sized we need the length l and the width w which are calculated as

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}}$$

$$w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}}$$

where $a = \frac{M20}{M00} - x_c^2$, $b = 2(\frac{M11}{M00} - x_c y_c)$, $c = \frac{M02}{M00} - y_c^2$.

This procedure continues until the window converges.

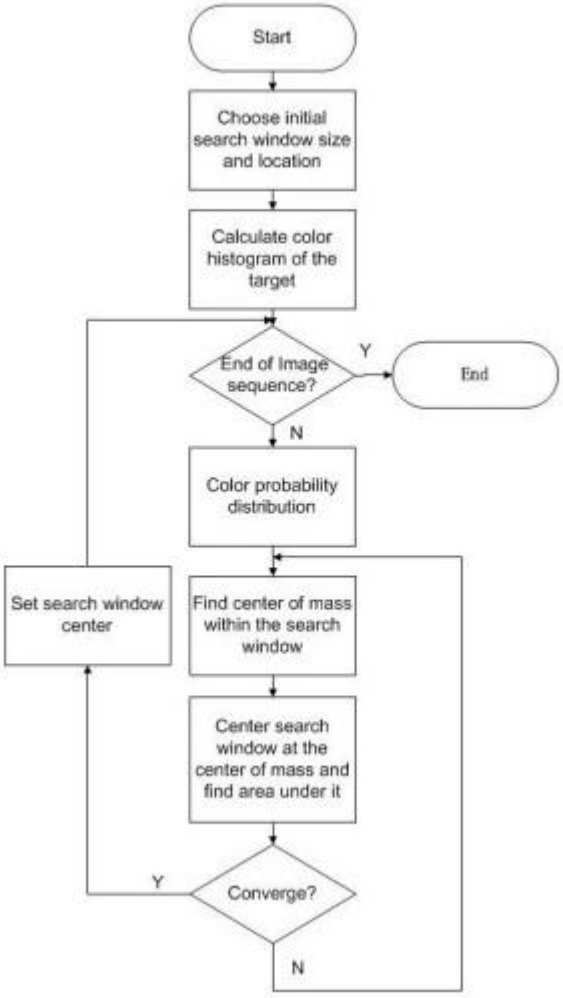

Fig. 1 Block Diagram of CAMshift algorithm

B. Improvement in CAMshift Algorithm

CAMshift is not robust against the background's noise i.e. when illumination changes during the tracking. For different illumination we need to artificially set the thresholds of S and V in the initializing period in order to maximize the effective pixels. In this paper we propose an improved CAMshift Algorithm to solve the above problem. The new algorithm is able to adjust the thresholds of S and V adaptively against the environment changes. The algorithm ignores those pixels with high S values when under a strong illumination and also those with low S are ignored when under weak illumination. Considering the above case we respectively change the thresholds of S and V after the search window is initialized. When the ratio of back projection value between the whole image and the search window reaches the minimum, the adjustment finishes. Sometimes only part of the object is recognized with this method because both the pixels inside and outside the search window would decrease when the thresholds of S and V change, then there are too few pixels to represent

the whole object. This problem can be solved by calculating the ratio of the effective pixels to all the pixels in the search window. We set its minimum value to 0.3 so as to ensure that there are enough pixels in the search window to represent the whole object. When either S or V is below 30, there is usually much noise and when either of them is larger than 120, the object is usually incomplete. So we set both S and V between 30 and 120. And our algorithm can find the best value to reach the best tracking performance.

III. SURF ALGORITHM

SIFT [15] is an image feature proposed by Lowe that is invariant to rotation and scale. SURF [16, 17] is the speeded up and robust SIFT, which can offer a better performance than SIFT. SURF is based on keypoint detection, descriptor generation and key points matching.

A. Key points Detection

SURF is characterized by the use of integral images. Here, the calculations of area of an upright rectangular region are reduced to four operations, and the calculation of first-order Haar wavelet response will be six operations.

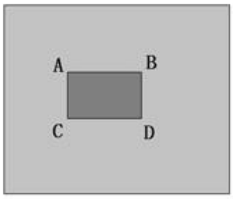

Fig. 2 Area computation using integral images

The integral image of image I(x, y) ($0 \le x \le M$, $0 \le y \le N$) can be defined by the formula:

$$I_{\sum}(x, y) = \sum_{i=0}^{i \le x} \sum_{j=0}^{j \le y} I(x, y).$$

Fig.2 shows how to perform fast pixel intensities, which can be calculated by:

$$\sum_{ABCD} = I_{\sum}(A) + I_{\sum}(D) - [I_{\sum}(B) + I_{\sum}(C)]$$

Gaussian pyramid, i.e., the image scale space is mainly used to find interest points in different scales. Here, Gaussian kernels can be changed in size to create the Gaussian pyramid. As Fig.3 shows, Laplacian of Gaussian is approximated to the box filter.

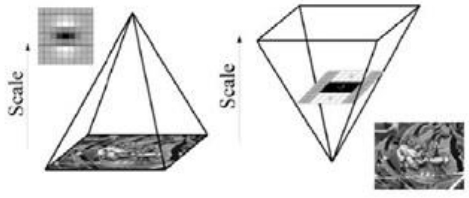

Fig. 3 Gaussian pyramid

Using this technique, multiple layers of the scale-space pyramid can be processed simultaneously and it negates the need to subsample the image, thus having better performance. To determine whether a point is extremum, the determinant of Hessian is used at the interest points localization. Suppose f(x, y) is a continuous function with two variables, then the Hessian matrix is:

$$H(f(x, y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad (12)$$

Its determinant is:

$$H(f(x, y)) = \det H = \frac{\partial^2 f}{\partial x^2} * \frac{\partial^2 f}{\partial y^2} - \frac{\partial^2 f}{\partial x \partial y} * \frac{\partial^2 f}{\partial x \partial y}. \quad (13)$$

If det H <0, which means the Eigen values of H have different signs, and then the point is not a local extremum. Otherwise it is an extremum. Replacing f(x, y) with I(x, y), then the Hessian matrix of the image is:

$$H(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix}. \quad (14)$$

Here $L_{xx}(x, \sigma) = \frac{\partial^2 g}{\partial x^2} * I(x, y)$, $L_{xy}(x, \sigma) = \frac{\partial^2 g}{\partial x \partial y} * I(x, y)$ and $L_{yy}(x, \sigma) = \frac{\partial^2 g}{\partial y^2} * I(x, y)$

After they are picked out, each of the extremums is compared with the 26 points adjacent to it (shown in Fig.4). The extremum point becomes an interest point if the value of any extremum point is above or below all of the 26 adjacent points.

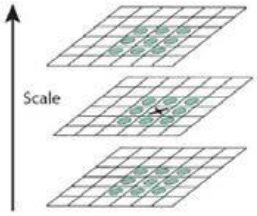

Fig. 4 Comparing the value of an extremum point with the 26 adjacent points

B. SURF Interest Point Descriptor

SURF interest point descriptor calculates the Haar responses in both X and Y coordinates in the circle region centred at interest points with a radius of 6σ. It relies on the dominant orientations of all the interest points. The size of Haar wavelet is 4σ, and the sum of vectors is calculated in every 60 degrees in the circle. Finally, the orientation with the largest sum of vectors is the dominant orientation. The process is shown in Fig.5.

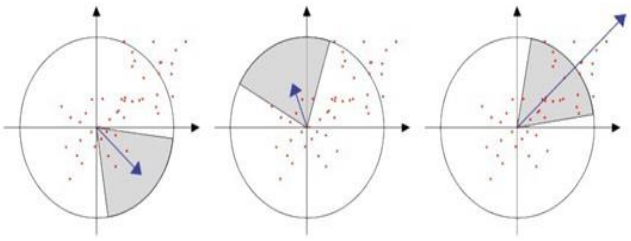

Fig. 5 Orientation Assignment

After the determination of dominant orientation, a square window is constructed which is centered at each interest point with a side length of 20σ. Then it is divided into 4×4 sub-region and the wavelet response is calculated in both the dominant orientation and the orientation vertical to it. If we define the wavelet of x and y as dx and dy, then there will be 4 values ∑dx, ∑dy, ∑|dx|, ∑|dy|, and totally it will be a 64-

length vector for each interest point. Thus we obtain the descriptor component by normalizing it.

IV. Improved CAMshift with SURF Algorithm

Though Improved CAMshift is able to automatically change its parameters to fit different illumination cases, some drawbacks still remain to be overcome. It is observed that the tracking performance is not ideal when the background is very similar to an object. Another drawback is that an object is easy to be lost under dynamic background because CAMshift uses Hue as the only feature in tracking. SURF feature is much more robust. When it is added to CAMshift, the accuracy of the whole tracking system is highly improved. Though, as SURF has much higher computational complexity its use to limited only to redetecting the object.

A. Determination of the lost Object

The lose object can occur in two different kinds of scenarios. On one hand, the confusion of an object with background when they are much similar in Hue; On the other, when the object moves too fast, the object region of 2 continuous frames do not overlap so that the search window may not converge to catch up the movement. To solve such problems, we consider evaluating the similarity between the target and the candidate object by Bhattacharyya distance. Consider that we use m-bin color diagram. The target's normalized color diagram is p (u)(u=1...m), and the candidate object's normalized color diagram is q(u)(u=1...m), then the Bhattacharyya distance is:

$$l(p, q)=\sum_{u=1}^{m}\sqrt{p(u)q(u)}. \qquad (15)$$

Ideally when the candidate object is all the same with the target, l(p, q)reaches maximum, but actually it can hardly be reached. We set 0.8 as the threshold. When l(p, q) is below 0.8, we think the object lost.

B. Improved CAMshift with SURF Method

Algorithm: Improved CAMshift with SURF method
Input: Template image
Output: Location of the target

1. Begin
2. Read the Template image and the first frame of the Image sequence, calculate the location of the target using SURF method;
3. Calculate the color histogram of the Template image, calculate the thresholds of S and V;
4. while next frame do
5. Update the size and the location of the search 5window using CAMshift;
6. Calculate Bhattacharyya distance according to Equation (15);
7. IF Bhattacharyya distance < 0.8 then
8. Target lost, recalculate the location using SURF
9. End IF
10. End While
11. End

Algorithm 1 shows the process of our algorithm. When the object is lost, SURF is used to search in the whole image to match the target. When matching points are found, the centre of the points is calculated and the search window is located around it. Then CAMshift continues to work. Under such a method, the system can gain good real time characteristic as well as robustness.

V. Application based on this method

Depending on the available environmental circumstances it is not always feasible to use a primitive input device like Mouse or Keyboard. In addition to this if we use mouse or keyboard as an input then it is static to that particular position location. Hence to all above problems we provide a solution that is an Object based Gesture Input which will be useful to the user to perform his basic windows OS operations from a distance without using primitive input devices. So the application being developed is handling of basic windows operations such as switching between the tabs of windows, minimize, maximize and closing a window, opening the start menu, task manager, command prompt, etc. using Object based Gesture Input. The front end of this system will be implemented using Visual Studio. The APIs required for video and image processing are provided by a library named Open CV (Open Computer Vision). The system will be able to perform all the functionalities as specified in real time and hence provide a suitable alternative to primitive input devices. The user will register the Object which he will be using for performing operations. Then the user will move the Object in front of the camera to perform a Gesture. The application will then perform the operation based on the Gesture. Whenever the user moves the Object in front of the Camera, the trajectory path followed by the Object is detected using improved CAMshift algorithm. This trajectory will be classified into Vectors i.e.it will have direction and slope. Accordingly, we have considered a total of 8 distinct vectors as shown in fig. 6 and fig. 7. A gesture will be a combination of these vectors and thus, ideally we have 8! i.e.40320 different possible gestures. The event of object loss is judged by the Bhattacharya Distance. After it has been judged that the object is lost, we will redetect the object using the SURF method and resume tracking using CAMshift Algorithm.

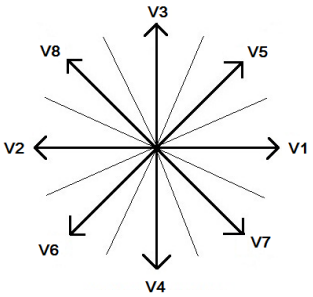

Fig 6. Vector Representation

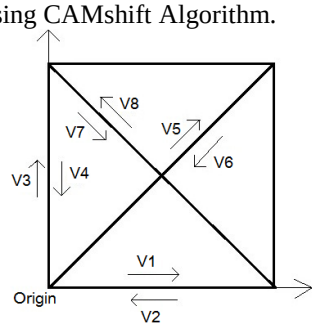

Fig 7. On Screen Representation

VI. CONCLUSION

In this paper, we propose an idea of using Object detection and Tracking algorithm for Human Computer Interaction. For the purpose of object detection and tracking we are using Improved CAMshift and SURF algorithms. The event of object been lost is checked using Bhattacharya distance method. This strategy proves to be very efficient and robust with a low computational complexity. Our HCI application proves that this method eliminates the need for cumbersome dedicated hardware and delivers good results with standard webcams (15fps, 640*480 resolution) and processors (dual-core 2.6 Ghz) available in the market. Our future work could focus on developing a framework that will allow HCI on many different platforms.

REFERENCES

[1] Object tracking using improved Camshift with SURF method, Jianhong Li, Ji Zhang, Zhenhuan Zhou, Wei Guo, Bo Wang and Qingjie Zhao, School of Computer Science, Beijing Institute of Technology, Beijing, P.R. China, OSSC-2011.

[2] SALARI, V. AND SETHI, I.K. 1990. Feature point correspondence in the presence of occlusion. IEEE Trans.Patt. Analy. Mach. Intell. 12, 1, 87–91.

[3] BROIDA, T. AND CHELLAPPA, R. 1986. Estimation of object motion parameters from noisy images. IEEE Trans.Patt. Analy. Mach. Intell. 8, 1, 90–99.

[4] COMANICIU, D., RAMESH, V., ANDMEER, P. 2003. Kernel-based object tracking. IEEE Trans. Patt. Analy. Mach. Intel. 25, 564–575.

[5] Bradski, G. R. 1998. Real Time Face and Object Tracking as a Component of a Perceptual User Interface, Wacv'98 (October 19 - 21, 1998). WACV. IEEE Computer Society, Washington, DC, 214.

[6] BLACK, M. AND JEPSON, A. 1998. Eigen tracking: Robust matching and tracking of articulated objects using a view-based representation. Int. J. Computer Vision 26, 1, 63–84.

[7] AVIDAN, S. 2001. Support vector tracking. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 184–191.

[8] ISARD, M. AND BLAKE, A.1998. Condensation - conditional density propagation for visual tracking. Int. J. Computer Vision 29, 1, 5–28.

[9] HUTTENLOCHER, D., NOH, J., AND RUCKLIDGE, W. 1993. Tracking non rigid objects in complex scenes, (ICCV). 93–101

[10] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, 2nd Quarter, 1998.

[11] XuenaQiu, Shirong Liu, Fei Liu, "Kernel-based Target Tracking with Multiple Features Fusion," 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, pp.3112-3117, 2009

[12] HuiyuZhou, Yuan Yuan, Chunmei Shi, "Object tracking using SIFT features and mean shift," Computer Vision and Image Understanding, pp.345-352, 2009.

[13] W. SHAO, A. HUANG, and Q. WEI, "Research on Object Tracking," Image Technology, p. 01, 2006.

[14] K. Xu, Y. Y. He, and W. Y. Wang, "Object tracking algorithm with adaptive color space based on CamShift," Jisuanji Yingyong/ Journal of Computer Applications, vol. 29, pp. 757-760, 2009.

[15] D.G. LOWE, "Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision," vol.60, No.2, pp.91-110, 2004

[16] H.Bay, T. Tuytelaars and L. Van Gool, "SURF: Speeded Up Robust Features," Computer Vision–ECCV 2006, pp. 404-417, 2006.

[17] C.Evans," Notes on OpenSURF library," University of Bristol, Tech.Rep. CSTR-09-001, January 2009