Optimized Tracking using VAIT

for Moving Objects

D karmaker[†], M S U Miah[*], M A Imran[*], M H Rahman[*] and M Z Alam[†]
Computer Vision Lab
Department of Computer Science
American International University-Bangladesh
{d.karmaker, saef, alimran, hafizur@aiub.edu} and zubaer190@yahoo.com

***Abstract*—This Automated object tracking and recognition system is very essential in terms of surveillance system and many more applications. Nevertheless correctness of these applications still remains the prior challenge. In this paper a simple procedure is proposed to develop a system that monitors a video and detect the objects that are moving and then recognize the objects comparing with our proposed method. Which is primarily targeted to stop stealing of vehicles from parking lot. This particular method can be further extended for many other applications. We use optical flow to detect moving objects and construct a data model of that particular object from different viewpoints. Applying VAIT on those certain regions that has got optical flow we try to figure out the matched object from our data models. Which boosts the performance and reduces the computation significantly. A moving vehicle can be in different angle at a given time. The tangible challenge for this paper is to recognition those objects.**

Keywords*— VAIT; tracker; DoG; SURF; SIFT; Optical Flow; Blob Analysis; Vehicle Detection; Vehicle Tracking.

I. Introduction

Developing object Tracking and recognition system is still one of the most challenging works for computer vision applications today. In many security places, we need this very accurately interpretation any input video. Objects may vary in shape during movement due to rotation. So it is very educated to recognize the object very accurately. We are trying to detect an object and recognize it in a way so that we can detect them in diverse angles. This papers main aims to procedure a system which can detect moving object in a video and recognize that object. For detection we use optical flow, Blob analysis and for recognition of the object we use our own VAIT features. Precise recognition and minimization of running time of object tracking is kept as a goal while designing the system. Our main target is here to apply VAIT features in video frames not in full area of the video frame for matching points but in detected areas only to recognize the detected object. In the following part of the paper we will be discussing about the reason of using VAIT feature for object detection and then next two sections it is written the whole procedure of our developed system.

Detecting and Tracking Moving Objects for Video Surveillance [1], they proposed an approach which relies on a graph representation of detected moving regions for deriving a robust tracker. The detection phase performed after the compensation of the image flow induced by the motion of observation platform produces a large number of regions. Another paper Automatic Detection and Tracking of Pedestrians in Videos with Various Crowd Densities [3], which is related video tracking. They worked on detecting human and draw their trajectory in varies crowd density. For videos with low density of pedestrians, first they detect individuals in each video frame using a part-based human detector that efficiently handles occlusion. Later, we employ a global data association method based on Generalized Minimum Clique Graphs for tracking each person over the course of the whole video. There is also a book which discussed on Moving Object Detection and Tracking in Videos [2] in chapter 2.

Comparing to other recent works on Object tracking and recognition techniques, we here try to simplify the tracking system for a fixed positioned camera records to track only moving objects using optical flow and focus on faster tracking and more accurate recognition with various angles using VAIT. We try to give a standard procedure of tracking moving object in a video and recognizing them.

II. System Architecture

We planned to take 8 different views of each object model data. Indeed 5 views are enough to generate all 8 views. Because all different views are shifted by 45 degrees, thus for a car except for view 0 and view 4 all are mirror image which can be generated by simple transformation. From figure 1 shows that if we can collect views from o to 4 shown with white color, rest of the views 5, 6 and 7 show with red color can be reconstructed by applying geometrical transform. After converting all the images to gray scale Images we made Matrix by horizontal concatenation. This conveys all properties of all 8-side views, which is a big matrix. To perform feature-matching form this big matrix is computationally very costly. So we need to come up with some elimination method.

A. Finding Interest Points

In this stage, the virtual image is done convolution with Gaussian filters at different scales, and then it is taken the difference of successive Gaussian-blurred images. Interest Points are then chosen from maxima/minima of the Difference of Gaussians (DoG) that occur at multiple scales. Specifically, a DoG image $D(x,y,\sigma)$ is given by

$$D(x,y,\sigma) = L(x,y,k_i\sigma) - L(x,y,k_j\sigma)$$

Where $L(x, y, k\sigma)$ is the convolution of the original image $I(x,y)$ with the Gaussian blur $G(x, y, k\sigma)$ at scale $k\sigma$ i.e.

$$L(x, y, k\sigma) = G(x, y, k\sigma) * I(x,y)$$

Hence a DoG image between scales $k_i\sigma$ and $k_j\sigma$ is just the difference of the Gaussian-blurred images at scales $k_i\sigma$ and $k_j\sigma$.

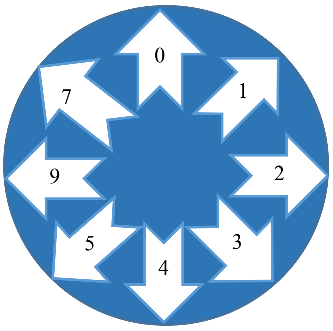


Fig 1: Different viewpoints of object data model creation.

Once DoG images have been got, interest points are identified as local minima/maxima of the DoG images across scales. This is done by comparing each pixel in the DoG images to its eight neighbors at the same scale and nine corresponding neighboring pixels in each of the neighboring scales. If the pixel value is the maximum or minimum among all compared pixels, it is selected as a candidate interest point [11].

B. *Interest Point Descriptor*

To describe a point neighboring pixels are considered. Histograms are formed of dimension 4 x 4 with 8 bins each considering the magnitude and orientation of 16 x 16 area nearest of the interest point. Thus each histogram contains 4 x 4 sub-sets of interest region [13]. Magnitudes are further weighted by Gaussian function resulting a vector containing all values of these histograms. 4 x 4 = 16 histograms each having 8 bins the vector has total count of 128 elements. After that it is normalized to invariance affine changes [12].

C. *Data Model Creation*

We take all the views of an object and prepare a data model Matrix. This Model Matrix is like an image of all views. But it is actually different matrix's under one array. If we try to detect objects from optical flow, the entire image needs to be processed. Which will generate many more interest points that are of no interest to this particular project. Figure 4 shows that almost 80% interest points can be ignored as they are static and computation will be significantly reduced if we apply matching only in moving areas which can be calculated from optical flow.

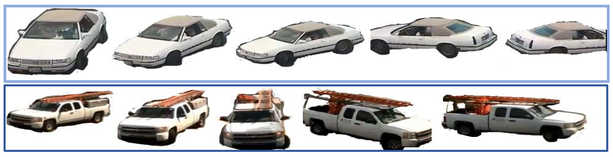


Fig 2: data model containing 5 different angles.

III. Work Flow Of Our System

Our system is mainly in two major parts. Object detection and recognition. The next sections will describe step by step procedure of our system. In the work flow, first we prepare our Models for recognition. Then we read video input and run loop taking its each frame at each iteration. Then we apply optical flow [4] with its previous frame. From the results of optical flow we filter our mask, which actually represents our moving objects. We apply morphological operations [10] on mask to remove noises there. After removing noise and we convert the mask into binary matrix. Then we apply blob analysis [5] on this to detect the objects. In each iteration, we take every object's locations and sizes from blob analysis result and crop the parts of the frame. We apply SURF Features on each part of image that we got. Then we compare the VAIT features with our pre prepared SURF Models. The Maximum possible Model is considered the detected object. Then we display our results drawing on the frame of the video.

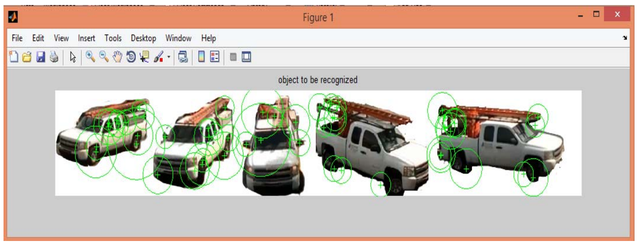


Fig 3: interest point extraction using SURF

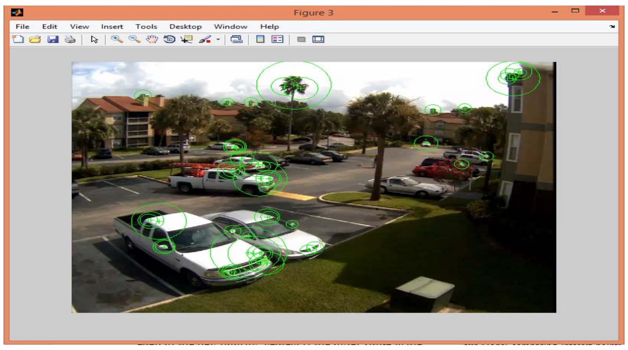


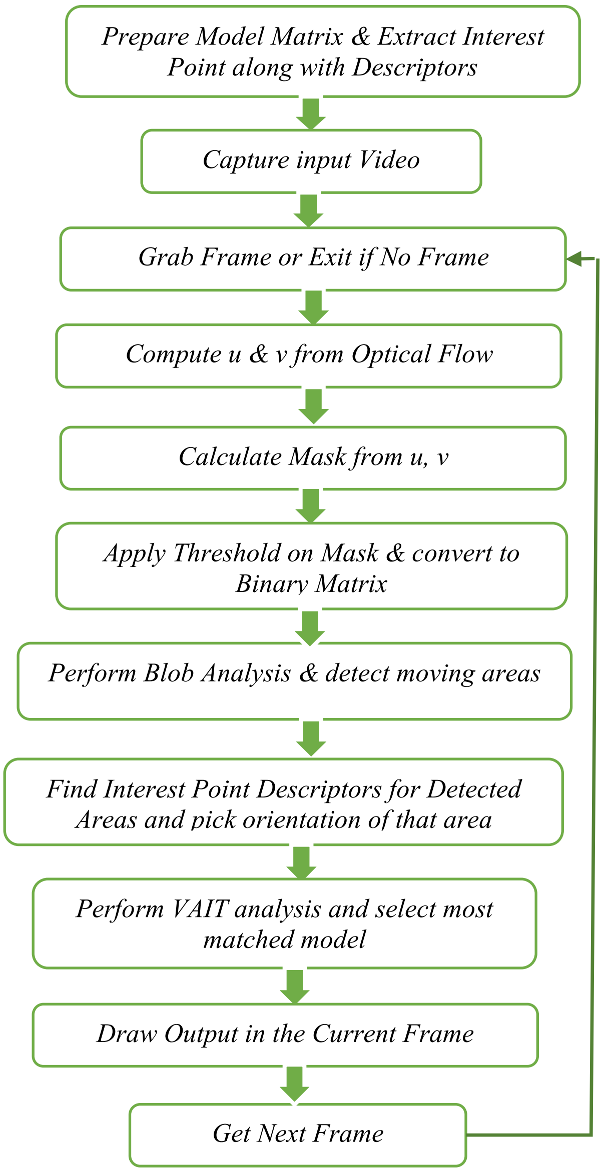
Fig 4: interest points detected

IV. Reading Dataset and Video

A. *Learning Dataset and Preparing Models*

In this very first stage, our plan is to make ready our Models for recognizing the detected objects. We have Fixed Number of views in our dataset. This Number will be fixed. More View will give more perfect result in recognizing object. But it will also slow down our programs runtime. So we take 5 views for cars reproducing total 8 samples to recognize a car. We read all the views of each object in dataset and covert in gray matrix. After merging them we get our Model Matrix. This Model actually represents all possible views of rotation of an object. Then we extract corresponding SURF features for Model Matrix. They are actually our Models for recognition. Finally, we insert them in a Set. Pseudo code is given bellow.

```
ImageFiles = Read All files from Dataset Directory
Number_of_Files = count (ImageFiles)
BEGIN LOOP i = 1 to Number_of_Files
    ModelMatrix = empty
    BEGIN LOOP j=i to i+Number_of_views
        CurFile = Take ith Image
        CurImg = ReadImage(CurFile)
        U = Conv2Gray(CurImg)
        ModelMatrix = Concat(ModelMatrix, U)
    END LOOP
    Model = Find_Interest_Points(ModelMatrix)
    Models[counter]= Model
END LOOP
```



Prepare Model Matrix & Extract Interest Point along with Descriptors
Capture input Video
Grab Frame or Exit if No Frame
Compute u & v from Optical Flow
Calculate Mask from u, v
Apply Threshold on Mask & convert to Binary Matrix
Perform Blob Analysis & detect moving areas
Find Interest Point Descriptors for Detected Areas and pick orientation of that area
Perform VAIT analysis and select most matched model
Draw Output in the Current Frame
Get Next Frame

Fig 5: System Architecture

Mentioned earlier benefits of using surf features, Surf detection technique can match the points of two images with different angles. We made a Model which actually contains SURF features of all possible views of an object.

B. Reading Input Videos

As our main goal is detect objects and recognize them from a video, our one of the main task will be reading a video input. It may be a file for process. Or it may be live automatic monitoring from surveillance cameras. In this stage, we will just take input a video and read it. For applying optical flow on video to detect moving objects, we need two consecutive frames of the input video. So we take consecutive frames of it in a loop for object detection [13] and recognition purpose.

C. Moving Object Detection

This is a loop which will take each two consecutive video sequences or frames. This loop will be continued until total video is finished or exit for surveillance camera for tracking. Inside this loop operation we perform two main operations for our system. One, Moving Object detection using optical flow [14] and Second, Object recognition by VAIT features.

D. Object Detection using Optical Flow

In this most important part, we take a frame of video and perform optical flow with the previous frame. In our program we used Lucas-Kanade for optical flow operation. There are other algorithms for calculating optical flow from two consecutive image with small changes of objects. By optical flow operation we will get the two matrix U and V. U is movement in X direction and V is movement in Y direction. Then we made a matrix which will carry movements in both directions (scalar one). By applying vector summation theory and then thresholding [8] the matrix we get a Matrix. We call it Mask. Because, this matrix is used to detect moving objects in our video. And in this matrix we have cells with non-zero values which are moved in video and other cells are filled with zeros. For Mask calculation, we use vector theory. Then we convert Mask matrix into Binary matrix for Blob Analysis to detect the object. Blob Analysis [5] is the one of the most popular algorithm which takes the Binary matrix as an input returns the location and size of 1's in a group in Binary Matrix. We call it bboxes which contains all matrix of 1's in group which actually represents the moving objects in a frame. This is how we can detect object by Blob Analysis. Here is the steps for this procedure:

```
Video_Input = ReadVideo(File_Name)
BEGIN LOOP foreach Frame
    CurFrame = generateFrame(VideoInput, index)
    [dy, dx, dt] = Find_Gradients (CurFrame)
    [U, V] = Get_OpticalFlow (dy, dx, dt)
    Mask = Calculate_By_Vector_SUM (U,V)
    Mask = Convert_Binary_Matrix(Mask,Thresshold)
    Binary_Mask = Morphological_operations(Mask)
    B = Blob_Analysis (Binary_Mask)
    Darw(CurFrame,‘rectangles’,B.Loations,B.Sizes)
END LOOP
```

E. Recognition using VAIT

VAIT stands for “Vehicle Angle Invariant Transform”. Now we have bboxes and also the models (set of surf features) and now we are ready for recognition. For this we take every rectangle from bboxes and crop the image from current frame. This cropped image is an object’s image. We call these the sceneImages. Then we calculate the surf feature of sceneImage and Then we compare the features with some indexes of our data Models. Selection of these indexes comes from computing the orientation. Which means we need to compute orientation of each pixel of the sceneImage and find out the most prominent direction. For example if the most prominent direction is 34 degree, which will correspond to index 4 from figure 1 and in figure 6 shows that will make it 0 as center. Also take a note from rest of the indexes, as we are diverting from the actual direction weights need to be reduced and red color directional indexes can be ignored. Thus reducing a whole lot of computation dynamically. We insert the match result in variable box Pairs. When the value of box Pairs or matched key points [7] is maximum. Then we can call the object is very similar the selected dataset Image.

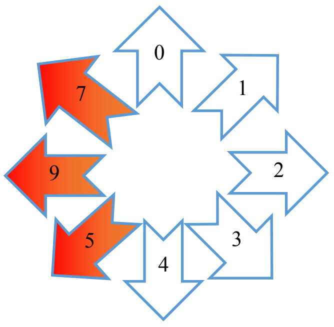


Fig 6: VAIT Index selection.

This is how we recognize the object. We only cropped the objects which is very small in size in our algorithm. So when we matching the SURF feature calculation time is very low. To compute the orientation we find the tangent of each pixel of the selected area. Next we need to convert all the orientations into histogram of eight bins each containing all the directions amongst 45 degree each. Here is step by step procedure. This is necessary to reduce the matching computation as if we get the orientation then model indexes can be reduced by the flowing algorithm.

```
BEGIN LOOP i = 1 to Count(Detect_Boxes)
     SceneImage = generate_sub_Image(CurFrame,
     B[i].size,B[i].Location)
     Ip = Find_Interest_Points(SceneImage)
     Previous_Matched = 0
     Index = 0;
     Flag = 0;
     Limit = findOrentation(SceneImage)
     BEGIN LOOP j= (limit – 2) to (limit +2)
     mp = Comapre_Match_Points(Models[i], ip)
     IF Match_Point >= Target_Points
        IF Match_Point > Previous_Matched
           Previous_Matched  = Match_Point
           Index = j
        END IF
        Flag = 1
     END IF
     IF FLAG == 1
        Darw_DISPLAY(Recognized With Index Model)
     END IF
END LOOP
```

V. Result

In our system, we take every frame, we show the detected moving objects by yellow rectangles and after recognition of an object we draw red color rectangles around the detected objects with the file name of Dataset, with which it matches maximum.

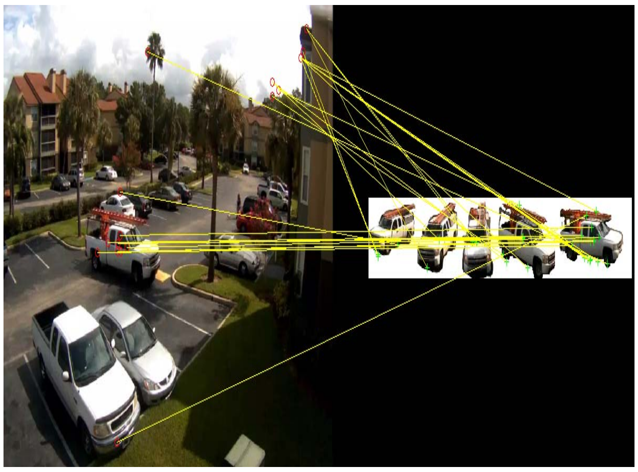


Fig 7: Match points in the entire frame

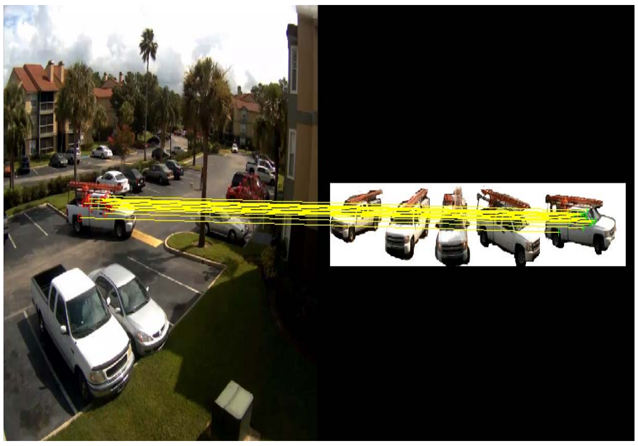


Fig 8: VAIT fracture matching

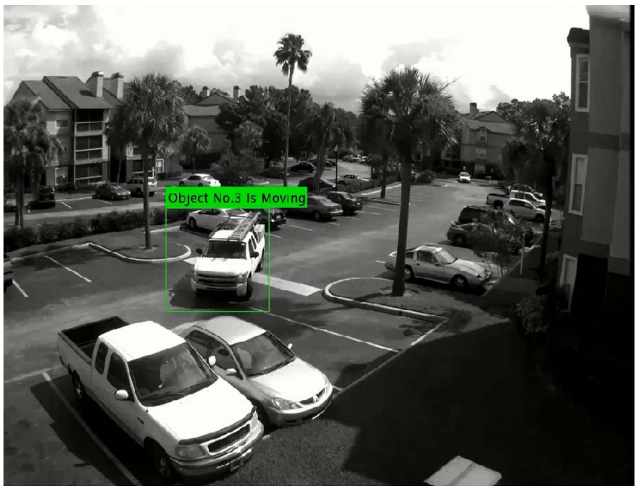


Fig 9: Object detection and recognition

The percentage of the result accuracy depends on video quality and quality of the dataset Images. The more noise free image and video will give us more accurate results. Our primary concern was in this paper to detect moving objects (Like cars) and recognize them in very secured place (Parking Areas).There video cameras are very close from the car. So it can be proper tracking for those vehicles.

Accuracy rate is depends on the quality of the video. Videos of High regulation give us more accurate recognition results. But the problem is with the increase the quality of the video, it also increase the run time of the program. So, it makes slower to get results than video. Based on the experimental results we got histogram like below:

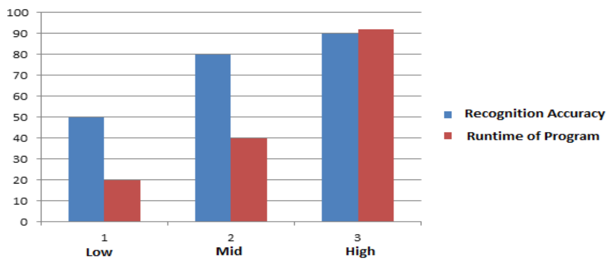


Fig 10: Accuracy Vs Runtime

This Actually shows us with the increase of the quality of the video Results is becoming more accurate. But the problem here is Runtime also is increasing, because the program need to check more pixels to detect the objects.

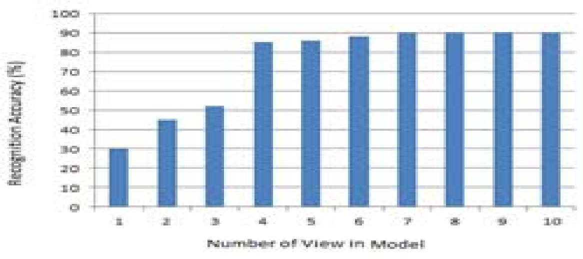


Fig 11: Accuracy Vs Number of views in model

From the experimental results we can say that, this will good well work with Mid quality videos, which give the 80% accurate results, because total processing time need to be very close the video time. Otherwise, the purpose of the system fails.

The results can be made more accurate with the increase of the number of views as well. From our testing results we experienced that for car recognition we used 4 views and got 80% above accuracy.

But after the 6 views we got same amounts of accuracy. So it will be useless adding more views after 4 for a car. It will increase the size of the model image that will also slow down our program. So for good overall performance of our system, we should choose the moderate quality of the video and at most 4-6 views of in each Model Image.

VI. Conclusion

Our system can be used in different security places for tracking car, vehicles or any other objects. It detect only which object are moved in the video and try to match that detect object with our dataset. Thus this system is very good for the place for anti-theft security. We will not say our system is working without error and giving proper results. But our system is capable enough to detect and recognize the object of different angles. Still there are some drawbacks in our system. Firstly, we used optical flow for our moving object detection. So, our video camera for recording video must be stable or in fixed place. Never It can be moveable. Secondly, we try to remove noise from video. But still, for better performance video must be clear and have proper lights. Noise free video and motionless video camera will give us perfect object tracking and recognition result.

References

[1] Isaac Cohen, G´erard Medioni, “Detecting and Tracking Moving Objects for Video Surveillance”, University of Southern California

[2] Karasulu, Bahadir, Korukoglu, Serdar, “Moving Object Detection and Tracking in Videos”, Chapter 2, Performace Evolution Software, 2013

[3] Afshin Dehghan, Haroon Idrees, Amir Roshan Zamir, and Mubarak Shah, “Automatic Detection and Tracking of Pedestrians in Videos with Various Crowd Densities”, Springer International Publishing Switzerland 2014

[4] Yair Weiss David J. Fleet. “Optical flow estimation”, page 29, 1985.

[5] Sarah Sookman, “Blob analysis and edge detection in the real world”, Matrox Imaging, page 29, 2006.

[6] Herbert, B., A. Ess, T. Tuytelaars, and L. Van Gool, SURF: "Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008.

[7] Lowe, David G. "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, Volume 60, Number 2, Pages 91–110.

[8] Gonzalez, Rafael C. & Woods, Richard E. (2002). Thresholding. In Digital Image Processing, pp. 595–611. Pearson Education.

[9] Nick Efford. Digital Image Processing: A Practical Introduction Using JavaTM. Pearson Education, 2000.

[10] Soille, P., Morphological Image Analysis: Principles and Applications, Springer-Verlag, 1999, pp. 173-174.

[11] (16 May 2014 at 05:48) Gaussian blur, Available at: http://en.wikipedia.org/wiki/Gaussian_blur (Accessed: 21 September 2014).

[12] Lowe, D. G., “Distinctive Image Features from Scale-Invariant Keypoints”, International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[13] (2 September 2014) Scale-invariant feature transform, Available at: http://en.wikipedia.org/wiki/Scale-invariant_feature_transform#cite_note-Lowe2004-3 (Accessed: 21 September 2014).

[14] M A U Rahman, M S U Miah, M A Fahad and D Karmaker, “ SHIMPG: Simple human interaction with machine using Physical Gesture” Control Automation Robotics & Vision (ICARCV), 2014 IEEE. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7064322.

[15] D karmaker, H Rahman, M S Rahaman, M S Rahman, “Global Motion tracking with six parameter model”, ARPN Journal of Systems and Software,Volume 1 No. 5, 2011. [Online]. Available: http://scientific-journals.org/journalofsystemsandsoftware/archive/vol1no5/vol1no5_3.pdf