A REAL-TIME MOTION DETECTION WITH DIFFERENTIAL IMAGES AND TRACKING WITH MEAN-SHIFT AND KALMAN FILTER

A Thesis

by

KOOJIN SUNG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Zixiang Xiong
Committee Members,	Byungjun Yoon
	Ulisses Braga Neto
	Bruce Gooch
Head of Department,	Miroslav M. Begovic

August 2017

Major Subject: Electrical Engineering

ABSTRACT

The main techniques of computer vision that can be helpful to surveillance system are detection and tracking. This thesis proposes a real-time motion detection and tracking system based on a single camera as a cost-effective solution for reducing human labor on surveillance. The detection algorithm deals with the change in pixels between sequential frames. Arithmetic operations on these pixel values provide position information of motion. Tracking process is more complicated. In this project, the tracking system requires selection of ROI (region of interest) as preprocessor. Then, mean-shift algorithm examines the distinct pattern of ROI and track the pattern every frame. To prevent a failure of mean-shift tracking, the tracking system is equipped with mathematical tool, Kalman filter. Kalman filter estimates and predicts the desirable route of mean-shift tracking, using its position and velocity information. The filter corrects unacceptable deviations from the route and helps a tracking window keep functional. This project separately developed detection algorithm and tracking algorithm and combined them at the final stage. The redundant imaging techniques are excluded in the proposed system in order to minimize the computation time, which ultimately shorten the delay for a real-time implementation. This system will promote low delay but high performance real-time surveillance system.

DEDICATION

To my father Ki Yul Sung, my mother Hye Seon Park and my brother Yong jin Sung who have supported me.

ACKNOWLEDGEMENTS

I would like to record my application to Dr. Z. Xiong for his continuous guidance and supervision all through the thesis research. Based on Dr. Xiong’s inputs, I could develop the ideas on my thesis and I acheived academic acomplishment though the research.

My appreciation also goes to the committee members, Dr. B. Yoon, Dr. U. Brage-Neto and Dr. B. Gooch for being my committee members and giving feedbacks on my thesis.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a chair advisor, Professor Z. Xiong of the Department of Electrical and Computer engineering. And, this work was reviewed by a thesis committee consisting of Professor U. Braga Neto and Professor B. Yoon of the Department of Electrical and Computer engineering and Professor B. Gooch of the Department of Computer Science and Engineering.

All work for the thesis was completed independently by Kookjin Sung.

Funding Sources

My graduate study was partly supported by a department scholarship from the Department of Electrical and Computer engineering at Texas A&M University.

NOMENCLATURE

AI	Artificial Intelligence
CV	Computer vision
DC	Direct current
DST	Destination
FOV	Field of view
HSV	Hue saturation value
PDF	Probability density function
PTZ	Pan tile zoom
ROI	Region of interest
RGB	Red green blue
SRC	Source
WMV	Window media video

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

Page

1. INTRODUCTION

It is evident that current security systems do not work perfectly without machine-aided surveillance. A traditional surveillance system demands manpower-intensive and time-consuming investigative techniques: therefore positive results are not always guaranteed(1). Computer vision techniques can play a pivotal role in compensating for the limitations of human surveillance thanks to its powerful performance but effectively less taxing property.

1.1 Problem statement

As the public safety sector confronts more contemporary challenges, the security concerns have grown significantly. Classical security systems no longer cope effectively with drastically increasing number of crimes and threats of terrorism. Relying on labor-intensive security is becoming less prevalent, and automatic monitoring has been suggested to replace it.

Of course, video surveillance systems are already prevalent in commercial establishments involving human supervision or external materials. For example, motion detection by infrared, ultrasound or body heat are well-known cutting-edge technologies adopted by many video surveillance systems currently. However, the advancement of image processing technology allows us to reach a stage where use of expensive media is not indispensable and more importantly, unstaffed surveillance cameras are ubiquitously available. In addition, a camera command system based on AI (artificial intelligence) is no longer passively works by human input. Computer vision-based video supervision provides more accurate information about what it watches than human eyes do. Therefore, a well-designed automatic camera can decrease hardware cost and reach range of which a human is not capable. This advantage, powerful but relatively inexpensive, triggered developers to graft automatic techniques to camera. The deployment of cameras has opened a wide application field for camera industry. Increasing commercial demand for automated video surveillance systems, it has become a fast-emerging research topic in the field of computer vision.

1.2 Proposed Approach

Motion detection and tracking algorithms have been primarily important topics in computer vision. Much research deals with detection and tracking systems separately. In this project, however, a whole system, which includes both detection and tracking is proposed. Theoretically, the most precise motion tracking is guaranteed when the monitoring system extracts entire pixel-scale changes in field of view (FOV). This is an ordinary detection principle for single images and applying this approach to every video sequence would offer a real-time motion tracking system. However, this approach requires a tremendous number of computations, which causes an unacceptable delay in real-time detection. This thesis presents here a comprehensive model of a video surveillance system by introducing an efficient tracking algorithm, equipped with state-of-the-art computer vision technologies but with fewer computations. Once, the motion is detected by camera, the system launches a tracking algorithm. The reason that tracking algorithm is more efficient and faster than continuous detection is very simple and clear. Unlike involving all pixels into computation that detection algorithm does, tracking algorithm examines the adjacent pixels after ROI (region of interest) is determined. Of course, detection algorithm is still necessary at the initial state to decide what to track.

1.3 Review of past literature

1.3.1 Surveillance system with computer vision

Some psychophysical studies have emphasized limitations of human capability to monitor moving object(2-4); not only does employing human sources require costs (5) but the chance of missing targets also increases with growing number of monitor displays(3). Additionally, non-automated system limits the area under surveillance, and consistent performance is hard to be achieved due to losing attention of human as monitoring time elapses(6).

1.3.2 Mean-Shift algorithm

Mean shift algorithm was present in 1975 by Fukunaga and Hostetler. The Mean Shift is a non-parametric procedure that find the peak of the probability distribution by recursive convergence. Mean shift has been extended to application of computer vision and clustering technique (7). Cheng first used mean-shift algorithm in 1995 as a solution of mode seeking problem(8). Cheng generalized mean shift algorithm in several ways; he allowed non-flat kernel, used weighed points in data and allowed a shift to be performed on any subset of n-dimensional Euclidean space. Later, Comaniciu(9) proposed Kernel-based object tracking using adaptive scale and background-weighted histogram extensions. However, it is pointed out that mean shift algorithm has some drawbacks. First, mean shift algorithm fails to track spatial information of color. Second, similarity measures are not very discriminative. Third, mean shift is prone to choose local minima instead of global minima, causing misclassification(10). Continuously adaptive mean-shift(Cam-shift) algorithm was introduced as a solution in that it can handle dynamic probability distributions by adjusting the size of window size(11). In this thesis, however, mean shift tracking is employed in order to simplify computation and its weakness is compensated by Kalman filtering.

1.3.3 Kalman Filter

Kalman filter was introduced by Rudolf E. Kalman in 1960 as a tool for estimation of theoretically meaningful value from noisy measurements with high accuracy(12). The paper described how to derive a recursive solution to the linear discrete data filtering problem by eliminating uncertainty and an error of measurements(13). Kalman filter was applied to navigation for the Apollo Project, which required estimates of the trajectories of manned spacecraft going to the Moon and back(14). Kalman filter is extremely powerful method that it has been employed in diverse real world applications including robotics, communication systems, GPS, weather prediction and aircraft control(15). In real world, most of systems are nonlinear rather than Gaussian. The application of

Kalman filter was extended to nonlinear system. For example, batch filter and particle filter are powerful nonlinear filter beyond Kalman filter.

1.4 Thesis structure overview

The proposed system includes two methods: motion detection and tracking the detected motion. In the introduction part, the reason that computer vision technologies are helpful to video surveillance is discussed and works related to mean-shift algorithm and Kalman filter are introduced.

The second chapter begins with comprehensive description of the proposed system. In this section, the thesis briefly explains how the proposed system is structured and what result is expected to be generated. The thesis states the preliminaries of the system including specifications of hardware and software. After that, the thesis introduces core algorithms and their applications for the proposed methods are discussed in detail. This section is divided into two parts: detection and tracking algorithm. For a detection section, a single motion detection algorithm is introduced. The thesis illustrates the detection process and its result with simple examples. For tracking, the thesis at first briefly describes the entire tracking system with a schematic. And then mean-shift algorithm and Kalman filter, the methods used in the tracking system, are intimately covered in separate section. In mean-shift part, color space conversion and histogram back-projection are also explained as a preprocessing of mean-shift algorithm. Kalman filter is simplified with some assumptions. The thesis discusses how the Kalman filter recursively infers the real tracking path. Demonstrations of Kalman filter tracking is also present in this chapter. At the last part of this chapter, the figure demonstration of mean-shift tracking aided by Kalman filter is shown.

The last chapter discusses academically meaningful aspects of the thesis and limitations found. Also, possible future works to advance the proposed system are introduced.

2. PROPOSED SYSTEM CONFIGURATION

2.1 Overview

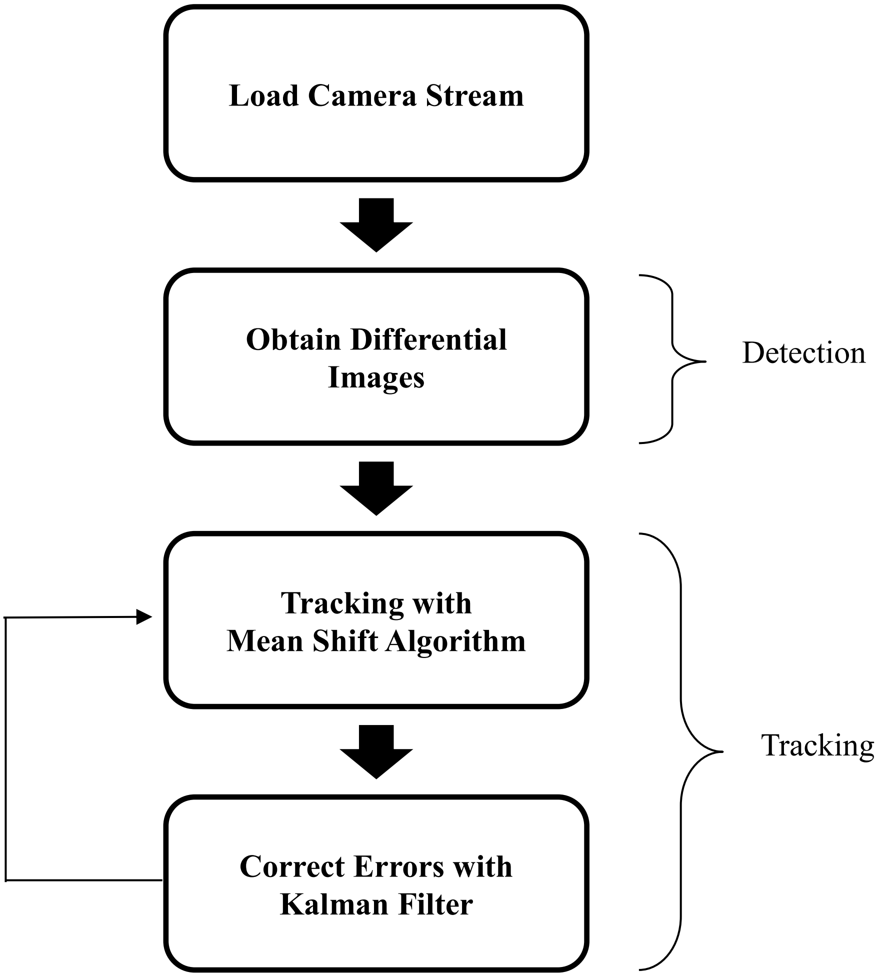

Load Camera Stream
Obtain Differential Images
Detection
Tracking with Mean Shift Algorithm
Tracking
Correct Errors with Kalman Filter

Figure 1. Overview of the proposed system

This thesis proposes a computer vision-based surveillance system. The system combines detection and tracking method that image processing techniques are applied. The figure 1 shows the overview of the entire procedure. Detection is conducted by a differential image technique which compares pixel values over the frames. When video sequences are loaded to the system, detection algorithm converts the original images into binarized images that motion is denoted with white dots. Once the system completes detection and a target is determined, the tracking algorithm is initiated to track the selected target. A mean-shift is a tracking algorithm that iteratively obtains the mean value of pixels in the ROI. Practically, however, mean-shift tracking itself does not produce robust results because its performance is highly corrupted when similar color pattern is present at target's environment. Therefore, Kalman filter is introduced to compensate the weakness of mean shift. Kalman filter is a mathematically method which is well-known for handling noisy measurements by estimating the previous measurements and then predicting the next measurement. Mean shift algorithm accompanied with Kalman filter results in accurate tracking even under noisy environment.

2.2 Requirement

The proposed system initially tested preliminaries to confirm the functionality of a real-time motion detection and tracking system. The profile of hardware and software are focused on universal standard to ensure operation on moderate level. The results proved that the proposed system with the given equipment can yield the level of theoretically expected performance. In addition, it is tested that the designed system is compatible with other types of camera such as PTZ(pan-tile-zoom) camera and external webcam.

2.2.1 Hardware requirement

In this project, the proposed system is conducted by a webcam built in a laptop. The camera and computing system have specifications in table 1. The camera provides the resolution of 640 x 480 and 1280 x 720. The former resolution is adopted for this project because the latter resolution arises unacceptable delay. The video streaming is delivered to the system as a WMV(window media video) format. WMV is a compressed video compression format for several proprietary codecs developed by Microsoft

Table 1. Camera specification

Camera Resolution	640 x 480 pixels
Frame Rate	24 frames per second
Video Capture Format	WMV (Window Media Video)
Computer Profile	CPU: Intel Core i5-4200M at 2.50GHZ Memory: 8 GB Graphic Card: NVIDIA GeForce GT 740M Operating System: Window 8.1

2.2.2 Software requirement

The software application of the proposed system is written in C++ with Microsoft Visual Studio 2015. Additionally, OpenCV C++ API is used to simplify and expedite programming work. OpenCV is an open source software library for computer vision and machine learning, which is developed by Intel. The main advantage of OpenCV is to facilitate real-time implementation of computer vision with high accuracy. OpenCV provides a collection of software algorithms put together in a library to be used by industry and academic field for computer vision applications and research. OpenCV supports from simple image processing practices to complicated machine learning algorithms. In this thesis, a substantial portion of computation for tracking algorithm is operated with OpenCV function commands. Although MATLAB also provides many pre-coded functions for image processing and has a strengthen in simplicity, the benefits of speed offered by OpenCV with C++ far outweigh convenience of MATLAB.

2.3 The method

2.3.1 Detection system

2.3.1.1 Overview

The proposed system takes advantages of a simple motion detection technique called differential images(16). Figure 2 shows the process of motion detection. For every frame, the loaded image goes through the pre-processing step that converts RGB format images to gray-scaled images for simple computation. Then, arithmetic logic operations on three sequential frames lessen the sensitivity of detection and extract true detections.

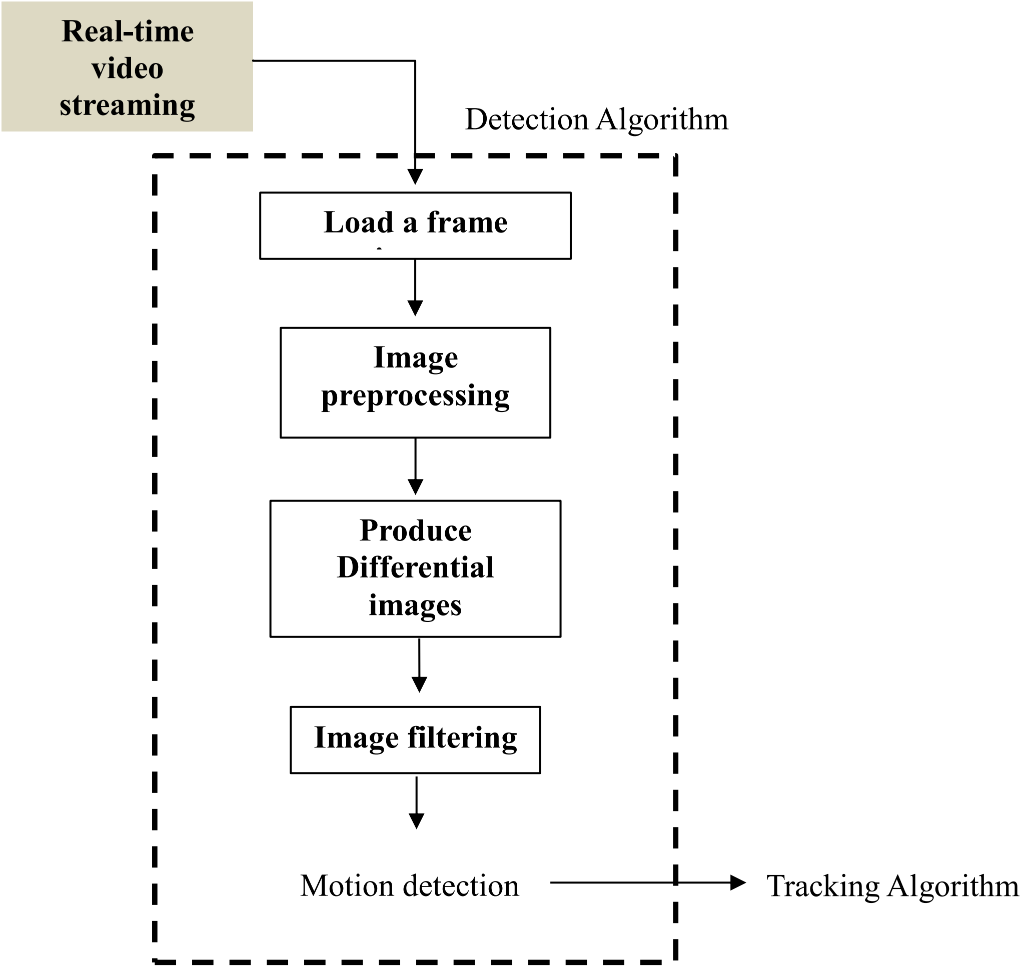


Figure 2. Detection process

2.3.1.2 Preprocessing

The OpenCV function ‘VideoCapture’ imports image frames of real-time video streaming from a connected camera. In this project, each input frame has 640 x 480 pixels with 3-layer (red, green, blue) RGB color format. The system converts the frame image from RGB color format to gray-scaled space that has only single sample per each pixel. This operation shortens the amount of computation and therefore enables real-time implementation. Figure 3 shows the comparison between RGB color space and grayscale space of image.

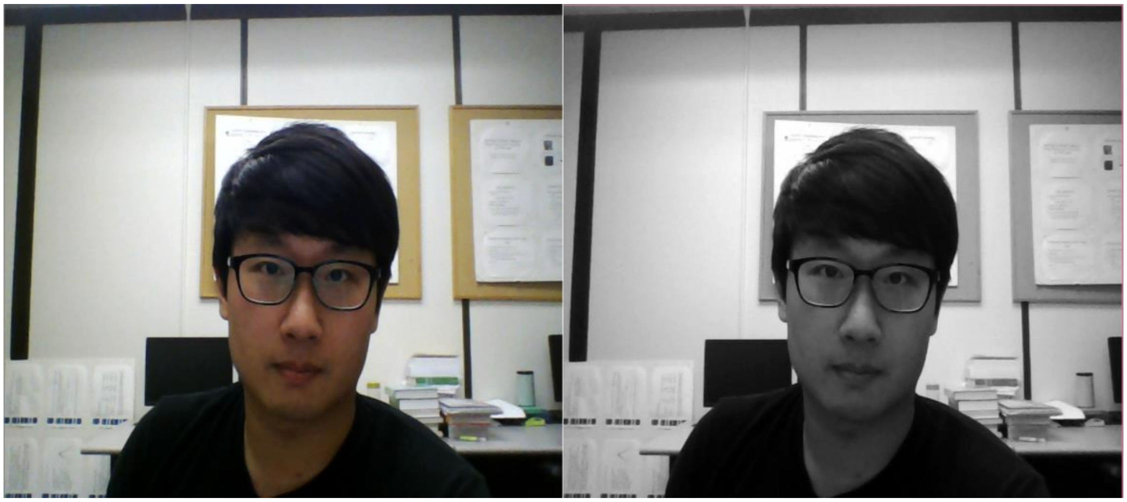


Figure 3. Original image (left) and gray-scaled image (right)

2.3.1.3 Differential image

When the new motion present in the camera view, there is a change in array of pixel values. That is, pixels within the motion region experience changes in their intensity value if a motion occludes the existing background. The real-time video streaming can be divided into numerous sequential images and the pixel value (gray-scale intensity) at same positions are measured over time. The non-zero pixels resulting from subtraction between sequential images indicates motion detection. This single motion detection technique is called differential images(16). The absolute difference of two consecutive frame images is used for detecting changes.

$$I_{dif}\ (x, y) = \left|\ I_{current}(x, y) - \mathrm{I}_{previous}(x, y)\ \right|$$

where *(x, y)* is the coordinates of the pixel.

The absolute difference means that the resulting value of two corresponding pixels is nonzero if two pixels have different value, otherwise the resulting value is zero. However, the simple subtraction is not enough to result in robust detection because it performs too sensitively. This method captures even any single pixel that has an undesirable nonzero value caused by light illumination and shadows. These unnecessary nonzero pixels are considered noise. The proposed system, therefore, employs the more advanced technique for robust detection. The detection system extracts only meaningful changes that represent the true motion changes over time. After that, image processing based noise elimination is also performed to enhance the robustness. Figure 4 shows the entire process of differential image algorithm using logic arithmetic between sequences.

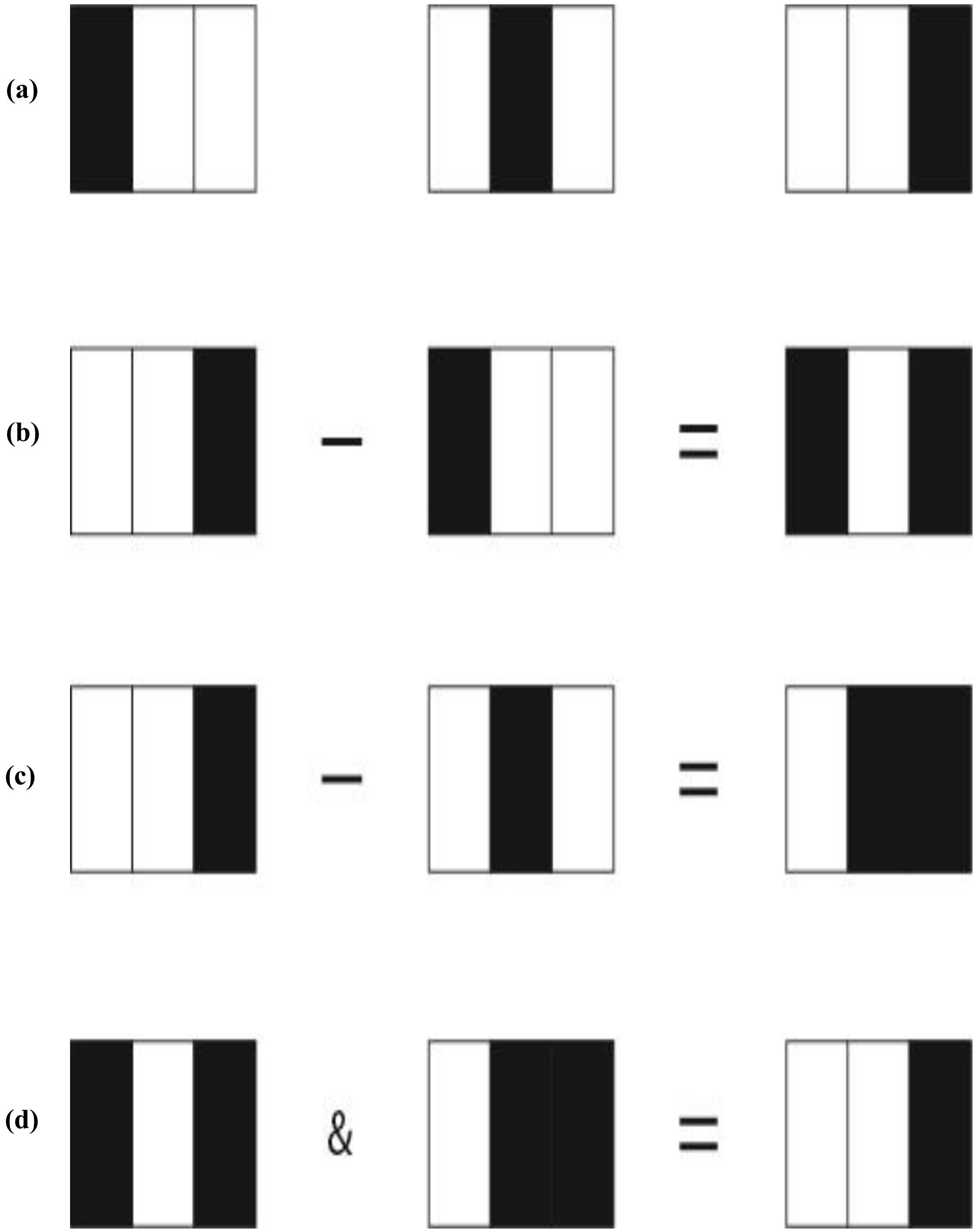


Figure 4. Motion detection process (Differential images)

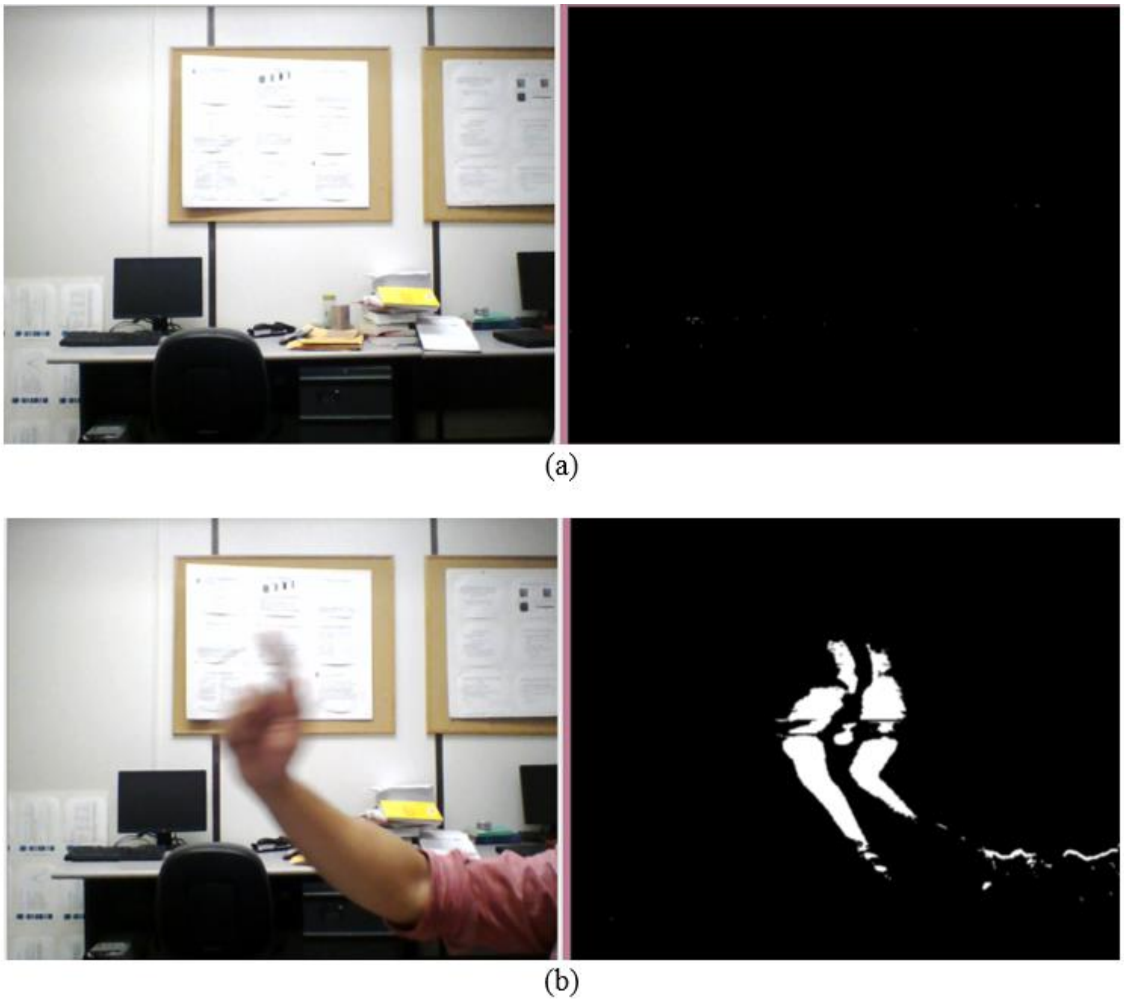
The effective motion detection is performed with three successive frames as shown figure 4(a). The arithmetic operations on differential images generates effective single motion detection. Three consecutive frames $I_1(x, y),\ I_2(x, y)$ and $I_3(x, y)$ are given, and the black region represents nonzero part. Figure 4(a) implies that the object move in the right direction. The resulting frames g1 and g2 represent the absolute differences where g1=$|I_3(x, y) - I_1(x, y)|$and g2=$|I_3(x, y) - I_2(x, y)|$.

In figure 4(b), the absolute difference between I_3 and I_1 results in g1. Similarly, the resulting image, g2, is gained by absolute difference between I_3 and I_2 as shown in figure 4(c). Then, the movement of an object becomes visible by AND operation between resulting images g1 and g2. Figure 4(d) shows that a logic AND operation extracts a changed part from sequential frames. The advantage of this operation is that the uninteresting background is removed from the result, thus the robust detection is possible (17).

When the differential image algorithm is applied to the frames, it is expected that motion detection results from pure difference between pixel. However, the environmental factors such as light shining, shadows and camera calibration cause some deviations in pixel values. Thus, the previous background image is not completed removed from the current images and remains nonzero as scattering over the entire image. These undesirable pixels are noise pixels. Gaussian denoising or morphological erosion is implemented for eliminating noise. However, the resultant image does not display clear detection because each pixel has different intensity value and some nonzero pixels remain with small intensity value although they must be removed by denoising implementation. To solve this problem, binarization is performed. The output image from denoising filter is segmented by thresholding. Each input pixel src (x, y) is examined whether its intensity value is greater than the constant threshold. If so, the output pixel dst (x, y) becomes maximum intensity, and otherwise it becomes zero.

$$\mathrm{dst(x,y)}=\begin{cases} 255 & \text{if } \mathrm{src}(x,y) > \text{threshold} \\ 0 & \text{if } \mathrm{src}(x,y) < \text{threshold} \end{cases}$$

The advantage of thresholding is that it gives a brighter and clearer detection. Figure 5 shows that the result after thresholding have the white dots that indicates the detected motion. The better result is available with adjusting threshold constant.


(a)

(b)

Figure 5. Demonstration of motion detection. (a) Background Image. (b) Image of motion detected

2.3.2 Tracking system

2.3.2.1 Overview

The mean-shift algorithm and Kalman filtering are two independent parts in the tracking module. After ROI is selected, the color histogram of ROI is applied to the frame image to detect features which are highly relevant to the ROI. For each pixel in the image, its relevance to the target is determined and computed using the histogram back-projection technique(18). The proposed tracking system perform this technique every frame as a spadework for tracking the ROI. When the ROI moves, the mean-shift window repeats track mean value of the ROI and build a new ROI. For each loaded frame, mean-shift tracking continues until the number of iteration satisfies the maximum number or the window no longer moves. Validation of tracking result is performed by comparing the histogram of the resultant window and that of the initial ROI using Bhattacharyya distance. If they are in similar pattern, it means that tracking is successful and the system terminates the iteration until the next frame comes. Otherwise, the proposed tracking system launches Kalman filter to optimize a mean-shift tracking. For every frame, Kalman filter updates its estimation with the new position and velocity information of center of mass. The entire tracking process is schematically shown in figure 6.

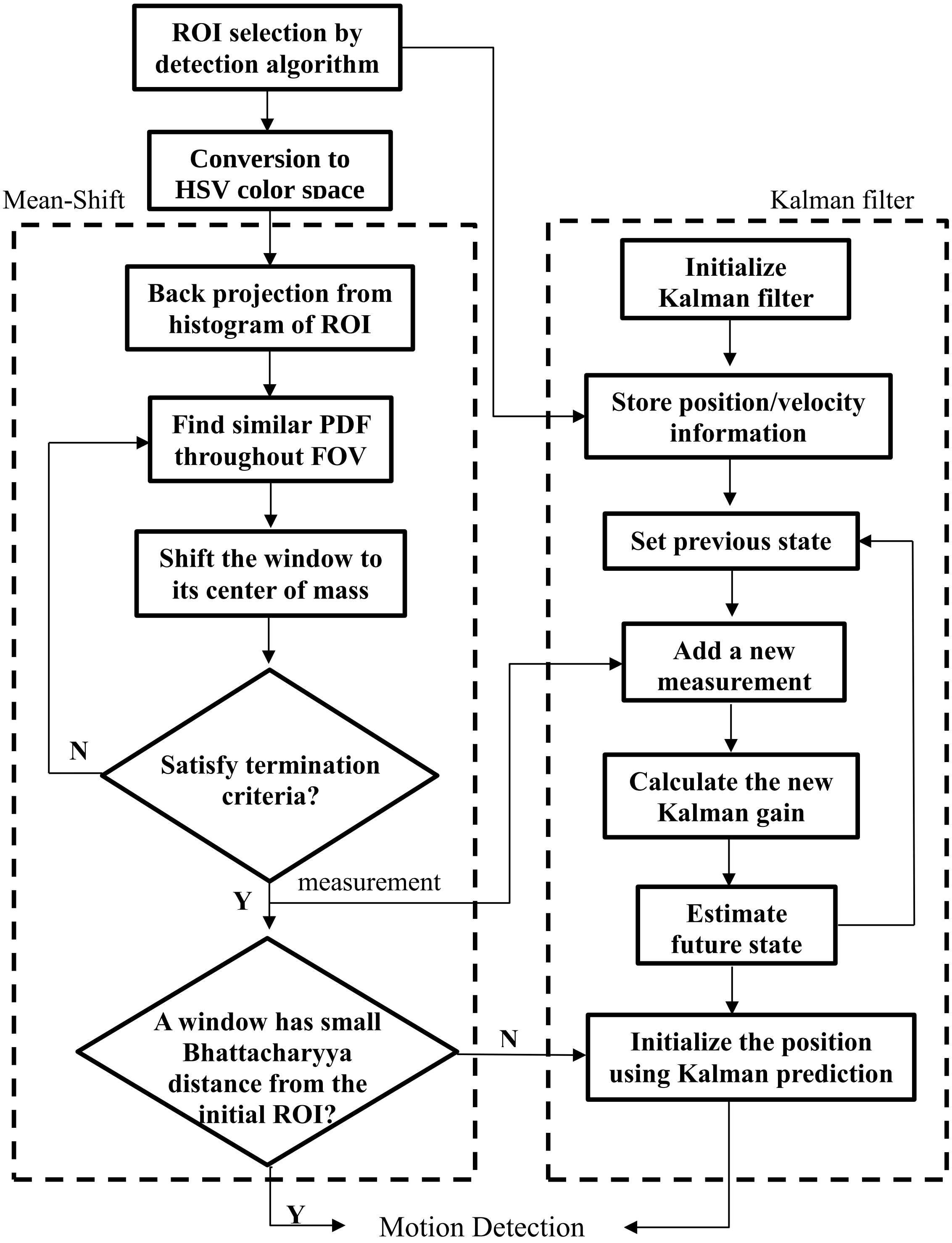

ROI selection by detection algorithm
Conversion to HSV color space
Mean-Shift
Kalman filter
Back projection from histogram of ROI
Find similar PDF throughout FOV
Shift the window to its center of mass
Satisfy termination criteria?
N
Y
measurement
A window has small Bhattacharyya distance from the initial ROI?
N
Y
Initialize Kalman filter
Store position/velocity information
Set previous state
Add a new measurement
Calculate the new Kalman gain
Estimate future state
Initialize the position using Kalman prediction
Motion Detection

Figure 6. The schematics of tracking process

2.3.2.2 Preprocessing

Each frame is acquired in RGB color format which is a basic additive color space based on red, green and blue. RGB format cause some problems in operating mean shift algorithm because it is highly sensitive to environment. First, RGB is sensitive to lighting variation that may misclassify the pixel pattern. Moreover, in RGB space, object shape is falsified by shadows and all the measured geometrical properties are affected by an error (19). Shadows magnify problems such as merging objects, distortion of color histogram and false. In this project, conversion to HSV (hue saturation value) color system is proposed as many approaches do. Unlike RGB, HSV separates the image intensity from the color information. Therefore, the system conduct histogram matching only on the intensity component and leave the color components alone. In RGB color space, shadows are most likely to have very different characteristics than the part without shadows even though they are in same color. In HSV color space, the hue component of both patches is similar.

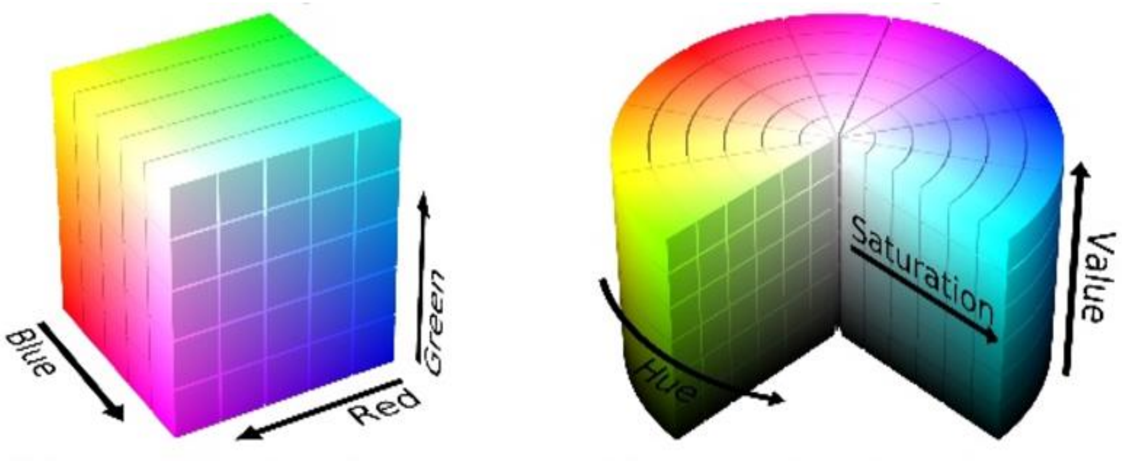


Figure 7. RGB structure (Left) and HSV structure (Right)

In the proposed system, RGB value of pixel is represented with 8-bit depth (0 to 255). Conversion to HSV color format can be calculated as follows:

$$\mathrm{R}' = \frac{\mathrm{R}}{255} \qquad G' = \frac{G}{255} \qquad B' = \frac{B}{255}$$

$$C_{max} = \max\ (\mathrm{R}', \mathrm{G}', \mathrm{B}'),\ \ C_{min} = \min\ (\mathrm{R}', \mathrm{G}', \mathrm{B}')$$

$$\Delta = C_{max} - C_{min}$$

$$\mathrm{H} = \begin{cases} 0 \quad if \quad \Delta = 0 \\ 60 \times \left(\frac{G'-B'}{\Delta} mod6\right)\ if\ C_{max} = R' \\ 60 \times \left(\frac{B'-R'}{\Delta} + 2\right)\ if\ C_{max} = G' \\ 60 \times \left(\frac{R'-G'}{\Delta} + 4\right)\ if\ C_{max} = B' \end{cases} \qquad \mathrm{S} = \begin{cases} 0 \quad if\ C_{max} = 0 \\ \frac{\Delta}{C_{max}}\ \ if\ C_{max} \neq 0 \end{cases} \qquad V = C_{max}$$

Table 2. Color values in HSV and RGB

Color	Color name	(H, S, V)	(R, G, B)
⬛	Black	(0°, 0%, 0%)	(0, 0, 0)
⬜	White	(0°, 0%, 100%)	(255, 255, 255)
🟥	Red	(0°, 100%, 100%)	(255, 0, 0)
🟦	Blue	(240°, 100%, 100%)	(0, 0, 255)
🟨	Yellow	(60°, 100%, 100%)	(255, 255, 0)
🟩	Gray	(0°, 0%, 50%)	(128, 128, 128)
🟫	Marron	(0°, 100%, 50%)	(128, 0, 0)
🟩	Green	(120°, 100%, 50%)	(0, 128, 0)
🟪	Purple	(300°, 100%, 50%)	(128, 0, 128)

As shown Table 2, hue is expressed in degree out of 180° and saturation and value are expressed in percentage. The demonstration of RGB to HSV color space conversion is as shown in figure 8.

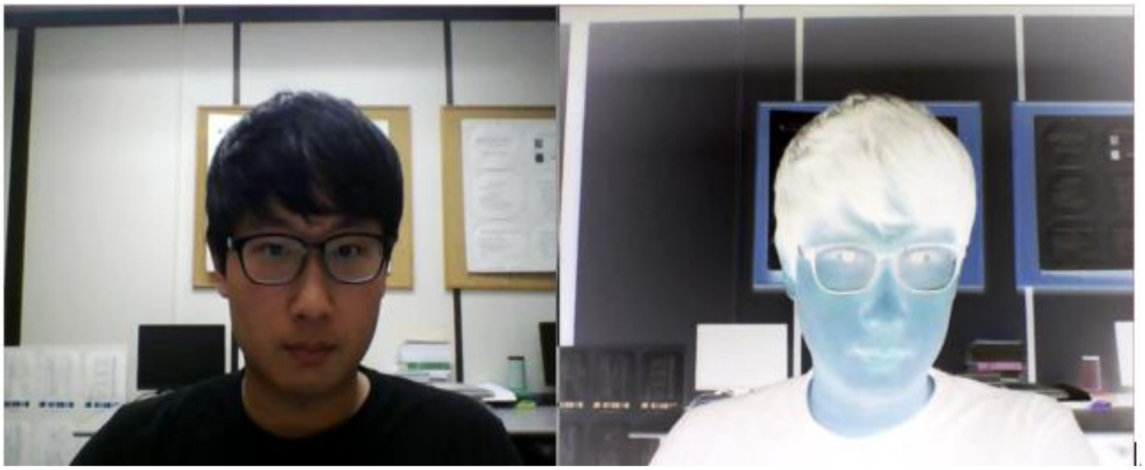
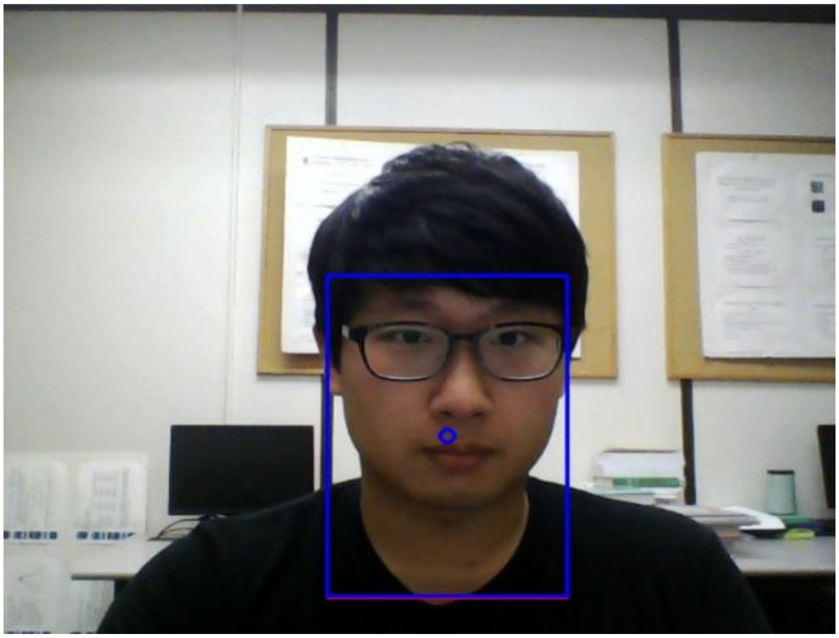


Figure 8. Demonstration of RGB to HSV conversion: RGB image (left). HSV image(right)

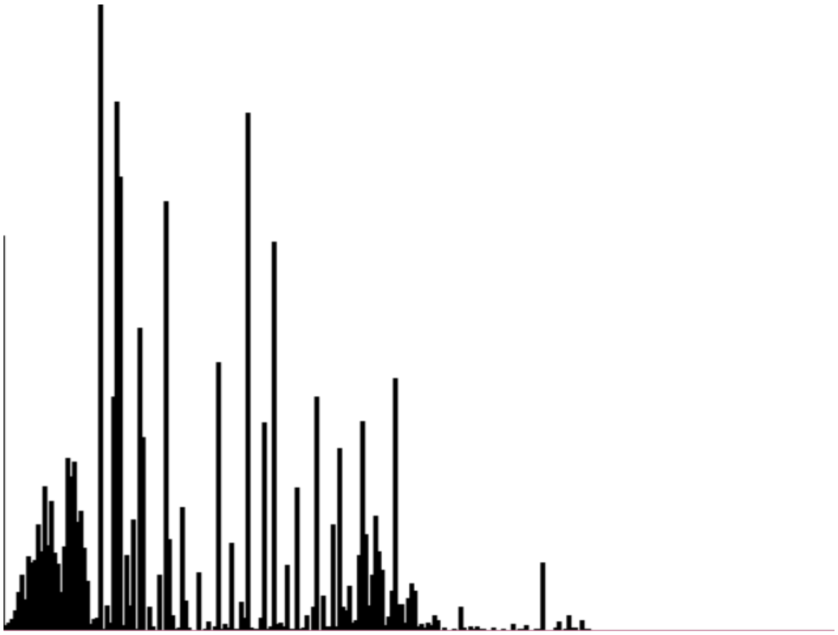
2.3.2.3 Histogram back-projection and mean-shift tracking

After converting to HSV color format, histogram back-projection is performed as a preprocessing of mean-shift algorithm. The histogram back-projection technique helps the tracking window find the feature of ROI while tracking. At first, the hue histogram of ROI is calculated and dominant pixel values over the ROI are recorded. Normalizing the acquired histogram yields the probability density function (PDF) of the ROI, making its norm equal to one. The next step is to allocate grayscale value (0 to 255) to each pixel according to its corresponding PDF value. In other words, if a pixel has very high probability value which denotes dominant pixel value in the ROI, the pixel is allocated with a high grayscale value close to 255.

This is the first step of tracking process when the input frame is loaded. Figure 9(a) shows the input frame and selection of the ROI (blue box). Once the ROI is selected as a target of tracking, the system calculates the histogram of hue component (color information) of the ROI. Figure 9 (b) illustrates the histogram of ROI where the horizontal axis of histogram denotes hue value and the vertical axis denotes the frequency of correcponding hue value.



(a)



(b)

Figure 9. Example of (a) selection of ROI and (b) its histogram (x-axis: hue intensity from 0 to 255, y-axis: corresponding probability)

Returning to the frame image, the tracking window examines each pixel to see how it relates to hue values in the histogram. Then, the probability of the corresponding value is allocated to the pixel as its intensity. This operation provides the new intensity distribution of image that emphasizes the features of the ROI in the image. Filtering the result with an appropriate threshold gives binarized image. Figure 10 shows how back-projection image is created from the original image in figure 9. The ROI in figure 9 mostly contains face that has skin color. Thus, the skin color pixels are given high intensity value close to white color. The white dots act as interesting points that a mean-shift window employs them as sample data for tracking algorithm.

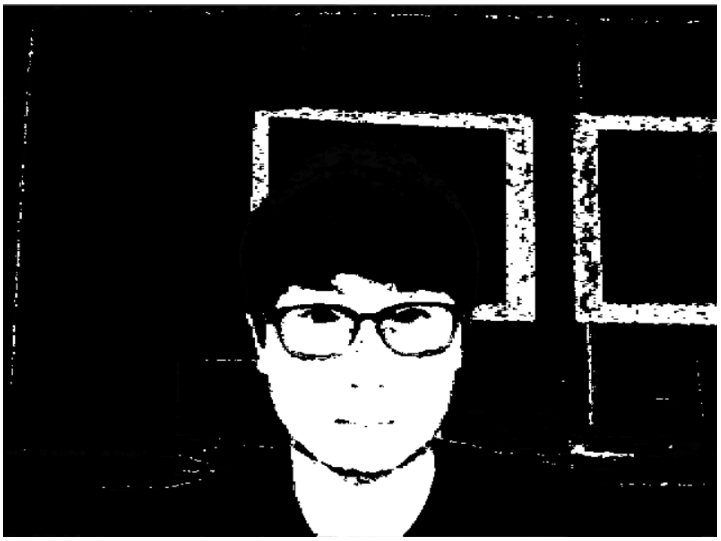


Figure 10. The example of back-projection Image

2.3.2.4 Mean-Shift

Mean-shift is an iterative tracking algorithm that employs a non-parametric gradient estimation for moving objects whose characteristic is defined by histograms (7). Literally, this algorithm continuously finds the mean of data and shifts to the point. Specifically, mean shift is a tool for finding modes (most frequently present pixel value) in a set of data samples, considering feature space as an empirical probability density function. If ROI is selected, mean shift regards it as a sample from PDF. Figure 7 shows the process of mean-shift algorithm. For each data point, the algorithm characterizes a window around it and computes the mean of the data point. After that, it shifts the center of the window to the mean until it converges. It can be considered that the window moves to denser region of the dataset each iteration. In short, when the denser area is detected in the feature space, they correspond to the mode of the probability density function.

If the unlimited iteration is available, it would guarantee highly accurate performance of tracking. However, this approach causes tremendous computation time and disables the real-time tracking system. Therefore, there should be a trade-off between the amount of computation and accuracy. Many types of termination criteria have been established in similar research such as limiting the number of shift, a threshold for shift distance or fixed operation time. The concern of real-time tracking is that computations time should be minimized as possible, but simultaneously it should retain the high-quality tracking (20). The tracking window does not need to experience the same number of iterations for each video sequence because the hue distributions are different every sequence. For each frame, the same level of tracking is obtained by different number of iterations. Some sequences require the relatively large number of mean-shift performance if they encounter complicated and hardly discriminable hue patterns. Therefore, termination criteria with the insufficient number of shift does not guarantee robust tracking.

The appropriate number of iterations or an acceptable error is empirically determined. In this thesis, it is designed that the tracking system completes mean-shift algorithm if the maximum number of shifts or insubstantial shift happens. the window cannot avoid redundant back projection and mean-shift computations even if qualified mean-shift is achieved at early stage. The OpenCV function ‘TermCriteria’ allows user to input the maximum number of iterations (maxCOUNT) and the desired accuracy or change in parameters at which the iterative algorithm stops (epsilon). The proposed system establishes 10 iterations for the maximum count and 0.5 for epsilon. Figure 11 shows how mean-shift algorithm is processed.

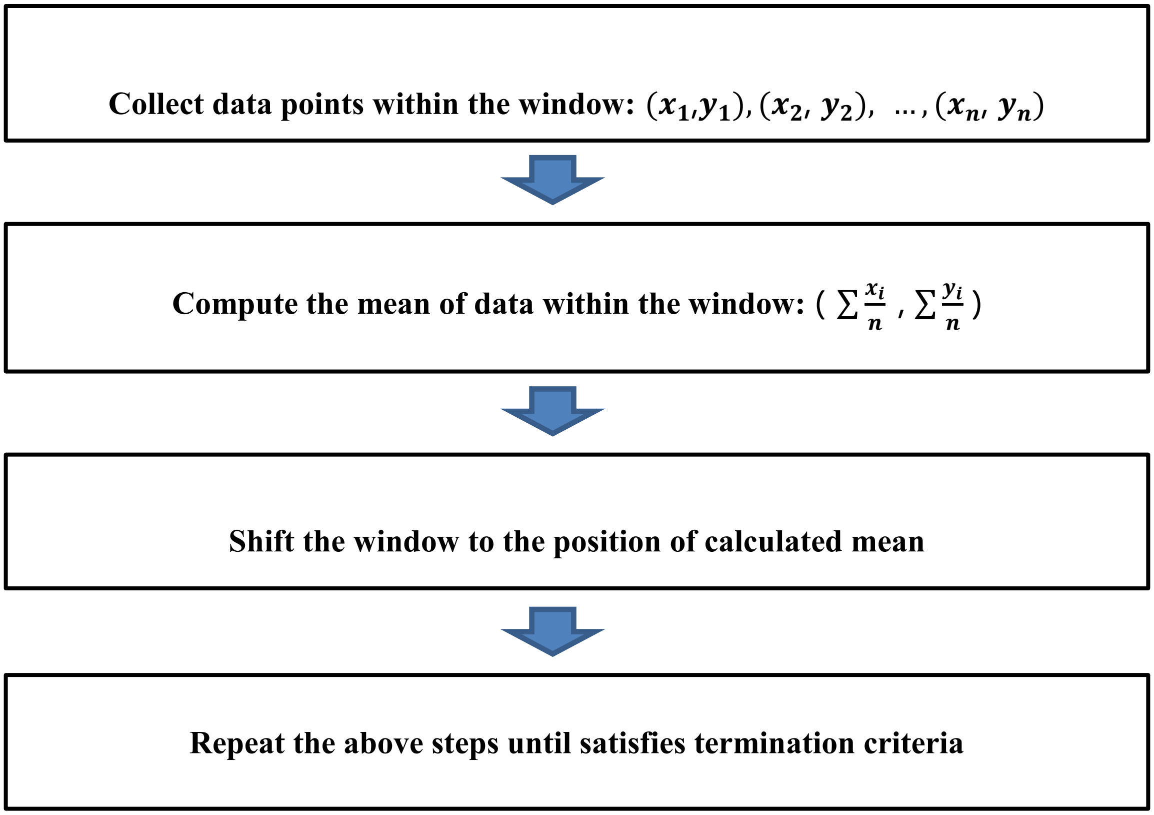

Collect data points within the window: $(x_1,y_1),(x_2,\ y_2),\ \ldots,(x_n,\ y_n)$
Compute the mean of data within the window: $(\ \sum\frac{x_i}{n}\ ,\sum\frac{y_i}{n}\ )$
Shift the window to the position of calculated mean
Repeat the above steps until satisfies termination criteria

Figure 11. Mean-Shift Algorithm Process

More intuitive understanding for mean-shift is available through figure 12. As shown in figure 12, the tracking window repeatedly performs a shift according to minima of a density function. Then, the kernel climbs the slope of PDF to reach the peak which denotes the densest point in data space. For this reason, mean-shift is also called mode seeking algorithm or hill climbing algorithm.

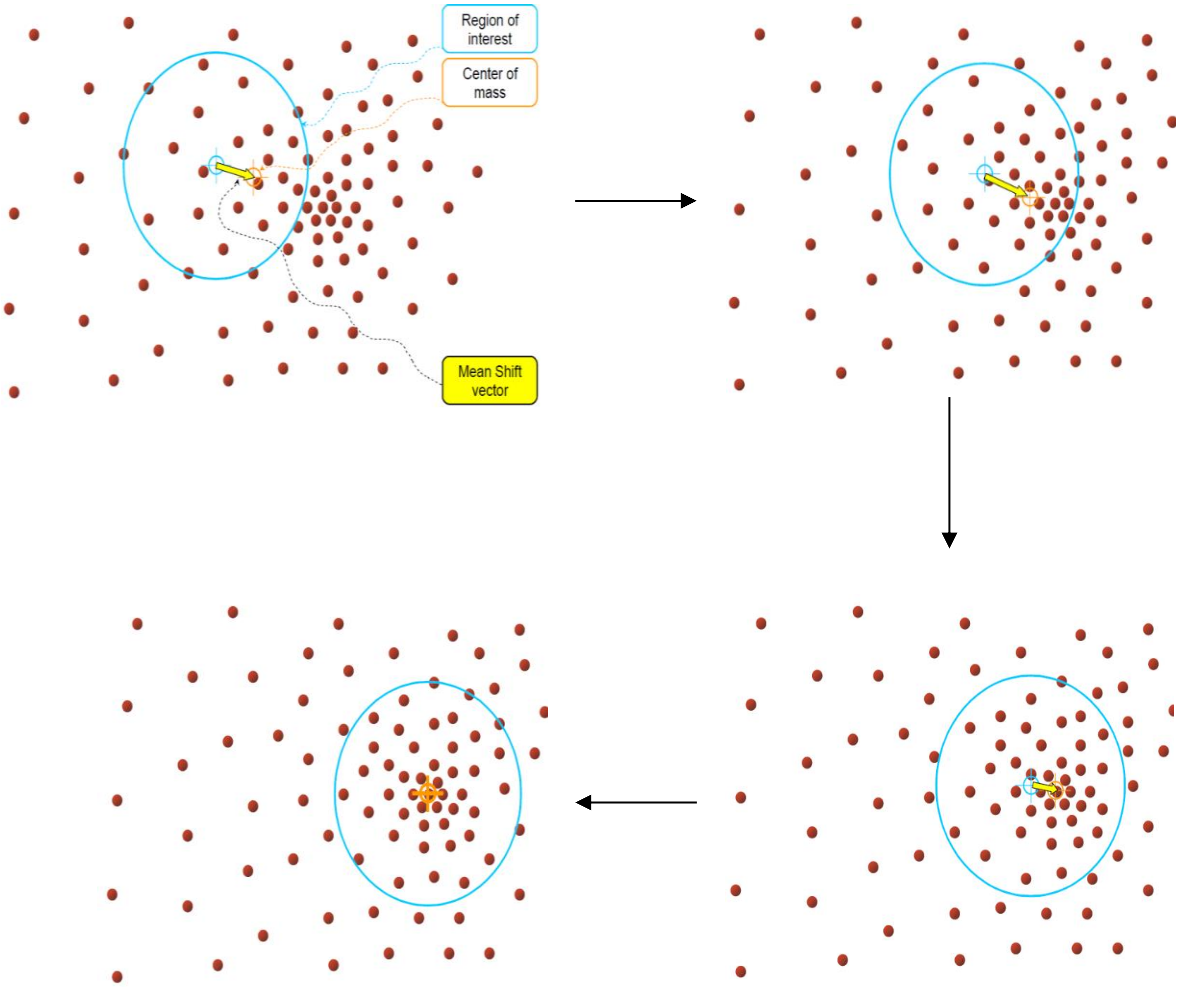


Figure 12. Principle of Mean-Shift Analysis

2.3.2.5 Error determination

The tracking window keeps exploring the mode until it satisfies the criteria for termination of mean-shift. However, mean-shift tracking algorithm has a weakness that it gets stuck in some regions and stop tracking if the certain region has higher PDF value than that of adjacent areas. In order word, mean-shift algorithm is highly prone to regard local minima of distance point as destination even though it is not a global minimum. For example, figure 13 shows that the tracking window stays at the center of the current region even though the denser region is obviously present at the below. This phenomenon also takes place if occlusion happens over the ROI. In addition, mean-shift is irresponsive to rapid motion because fast movement of the ROI causes distant separation between its previous position and next position over video sequences, and thus the window is more likely to conduct false seeking during navigation.

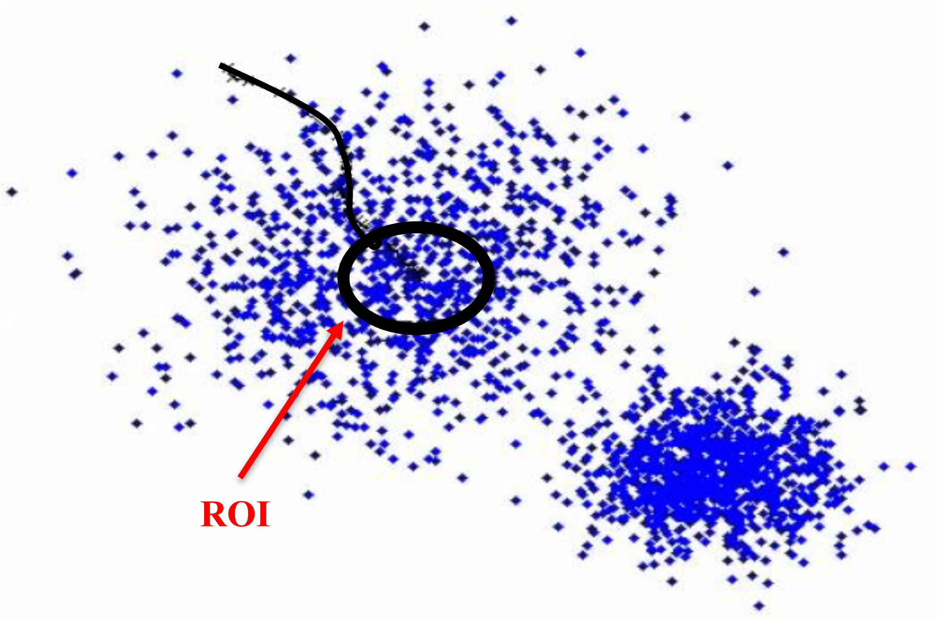


Figure 13. Error occurrence in mean-shift tracking

The proposed system employs Bhattacharyya distance to determine incorrect tracking. The Bhattacharyya distance is a renowned method for measuring divergence of probability distribution, which provides quantitative similarity between histograms. A metric d (H_1, H_2) compares two histograms H_1 and H_2 using Bhattacharyya distance. The equation for d (H_1, H_2) is as follow.

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2}} \sum_{I} \sqrt{H_1(I) \cdot H_2(I)}}$$

where $\overline{H_k} = \frac{1}{N} \sum_J H_k(J)$ (mean of H_k) and N is the total number of histogram bins.

The lower d (H_1, H_2) is, the more similar H_1 and H_2 are. That is, the two histograms are more likely to be correlated with higher d (H_1, H_2). OpenCV 3.0 provides the function ‘compareHist’ to perform a comparison. For Bhattacharyya distance, the OpenCV command ‘CV_COMP_BHATTACHARYYA’ should be inserted. Once, the system detects the discrepancy between two histograms, which means tracking the ROI fails, the position tracking by Kalman filter initiates the position of the ROI. The detail process of Kalman filter will be discussed next chapters.

2.3.2.6 Error correction with Kalman filter

The proposed system employs a Kalman filter as a solution of malfunctioning of mean-shift tracking. Kalman filter is very powerful method of correcting errors in the sense that it provides estimations of future states of dynamic system based on previous measurements even if the true state of the system is unknown (21). It is an iterative mathematical process that uses a set of equations and consecutive input measurements to estimate the true value quickly such as position and velocity of the object. In other word, true value of dynamics can be inferred when the measured values contain uncertainty or variation. Kalman Filter is a sufficient statistic to compute the minimum variance prediction of the process at time k over any arbitrary finite subset of the space.

In this project, Kalman filter can be used to track the true mean of moving object when the mean shift filter fails to track and shows unacceptable variations. Every calculated mean value goes into Kalman filter as measurements. Since Kalman filter only relies on mathematical analysis dynamic pattern, it is very robust to change of motion shape or speed while sole mean-shift tracking is vulnerable to these factors

.

When the ROI is chosen, the first mean-shift operation finds the center of mass of the ROI and it is regarded as the initial state of Kalman filter. To follow a trace of the window, the tracking system needs to know where it is located and how fast it is so that it can navigate throughout the FOV. Therefore, the state matrix is formed with position and velocity variables. Generally, the state information at time k is denoted by x_k.

$$x_k = [\,\mathrm{x1}, \mathrm{x2}, \ldots, \mathrm{v1}, \mathrm{v2}, \ldots]$$

As shown in table 3, the process of Kalman filter starts from the two equations: process equation and measurement equation. The process equation use pre-defined matrices to predict the position and velocity at the next moment in the future. Since this variance of measurement should be known for Kalman filter implementation, it is assumed to be a zero-mean Gaussian noise and measurement equation of the system can also be performed according to table 3.

Table 3. Equations and variables for state and measurement

<table>
<tr><th>Process Equation</th><th>Measurement Equation</th></tr>
<tr><td>$x_{k+1} = A \cdot x_k + B \cdot u_k + w_k$</td><td>$z_k = H \cdot x_k + v_k$</td></tr>
<tr><td colspan="2">w_k : process noise ~ N (0, Q)
v_k : measurement noise ~ N (0, R)
x_k : system state at k (signal value)
z_k : measurement at k
u_k : external control at k
A : state transition model, n x n matrix
B : optional control – input model, n x 1 matrix
H : observation model, m x n matrix
Q : process noise covariance matrix
R : measurement covariance matrix</td></tr>
</table>

w_k is a vector that its elements are process noise terms for each parameter in the state vector. As mentioned above, the noise is assumed to be Gaussian. It is drawn from a zero-mean multivariate normal distribution with covariance given by the covariance matrix Q.

These two equations can be interpreted that the new optimal estimate is a prediction made from previous optimal estimate. And the new uncertainty is also predicted from the previous uncertainty with some additional uncertainty from the environment. In real world, pure Gaussian is almost impossible. In this thesis, however, noise functions (w_k, v_k) are assumed to be Gaussian because Kalman Filter effectively yields correct estimations even if noise parameters are poorly estimated.

Correlation between velocity and position arise if the new position is estimated based on the previous one. For example, if the current velocity becomes higher, the position moves farther, so the position will be more distant. This correlation can be denoted by covariance matrix ($\sum_{\mathrm{ij}}$) that its each element is the degree of correlation between the i$^{\mathrm{th}}$ state variable and the j$^{\mathrm{th}}$ state variable. This covariance matrix is denoted by P_k called the error covariance.

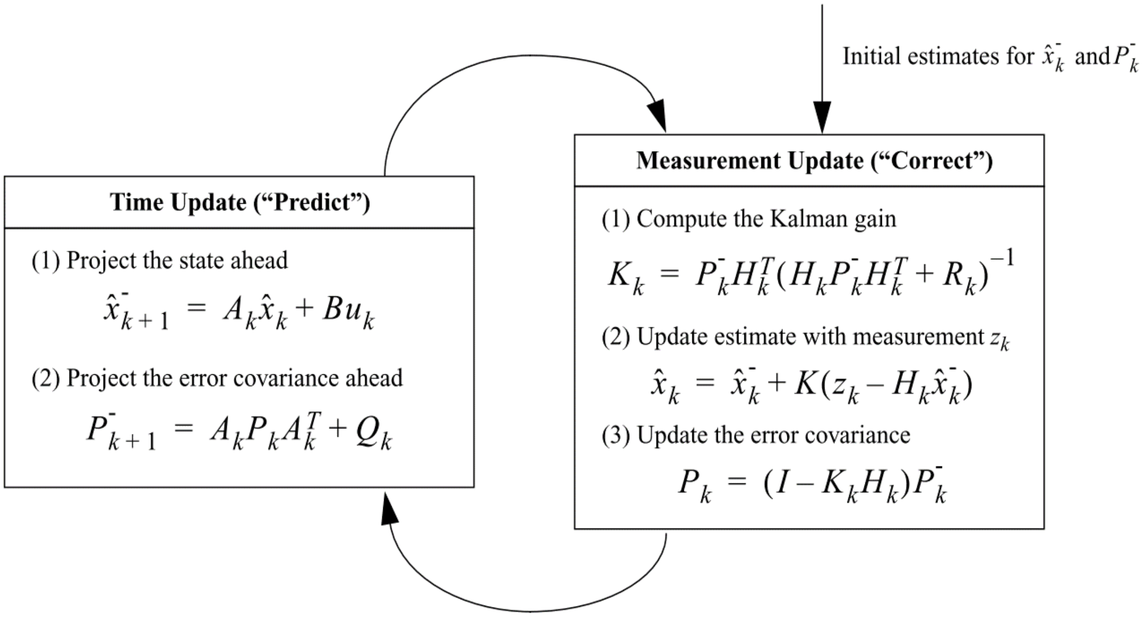

Initial estimates for $\hat{x}_k^-$ and P_k^-
Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_{k+1}^- = A_k \hat{x}_k + B u_k$
(2) Project the error covariance ahead
$P_{k+1}^- = A_k P_k A_k^T + Q_k$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K(z_k - H_k \hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H_k) P_k^-$

Figure 14. The process of Kalman Filter

Kalman filter proceeds several calculation steps for estimation and prediction at every iteration. Kalman gain (K_k) is calculated with the previous estimations x_{k-1} and P_{k-1}. It allows an estimation to be narrowed into the true value. With the updated Kalman gain and the output of measurement equation, the current state is estimated. The last step of measurement is to calculate a new error in estimate. It predicts the amount of error or uncertainty in that estimate.

The process should start from k=0 and assume some initial state such as $x_0 = 0$, and P_0 = 0. Even though the initial assumptions are far different from the true value, Kalman filter drastically approaches to true value and shows improvement every iteration.

For a simple example of Kalman filter, let us imagine the case of DC voltage measurement at the certain point of circuit. The voltage at the point, theoretically, should have a constant value, but its real measurements may vary due to many types of noise. If Kalman filter is used here over some time, the estimation value would recursively converge to true voltage. The plot of this example is shown as below.

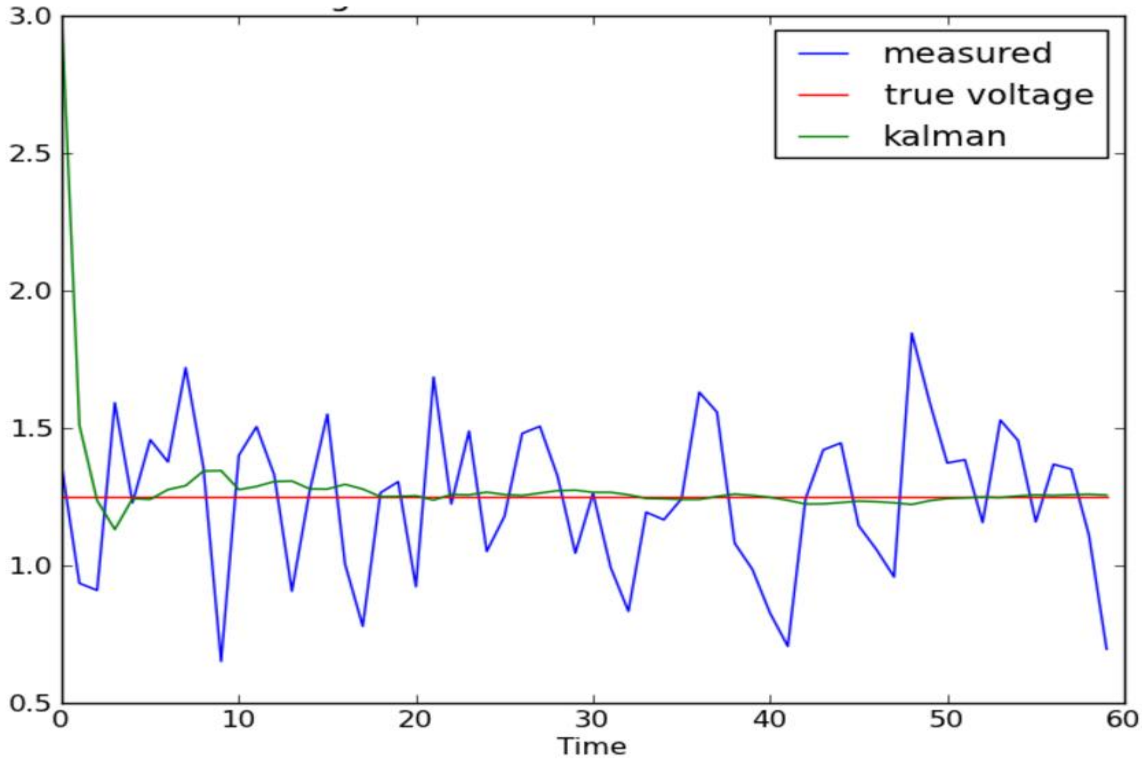


Figure 15. The example of Kalman filtering

Unlike the above example that each state has only one-dimensional component, this thesis needs to treat two-dimensional vector space. There are 4 elements for state input. The state input is a vector form X= [x_k y_k v_x v_y] where (x_k y_k) are position information of centroid of moving object being tracked by mean-shift and (v_x v_y) are their velocity. The parametric matrix A and H that describe model parameter are as follows

$$A=\begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad H=\begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \end{bmatrix}$$

where Δt is a period of measurement and $\Delta t = 0.5$ is used in this project.

Matrix A is derived by newton’s law. Matrix B is for external control, but it is not used in this project because there is no external force on motions. Matrix H extracts position information from the state input.

Above parameters setting simplifies Kalman filter implementation in the proposed system. The simplified computation enables robust real-time tracking. Moreover, the advantage of the proposed system is that it is not bound to operate at certain features such as face or circular object. The figures below demonstrate that the performance of mean shift tracking with Kalman filter is successful.

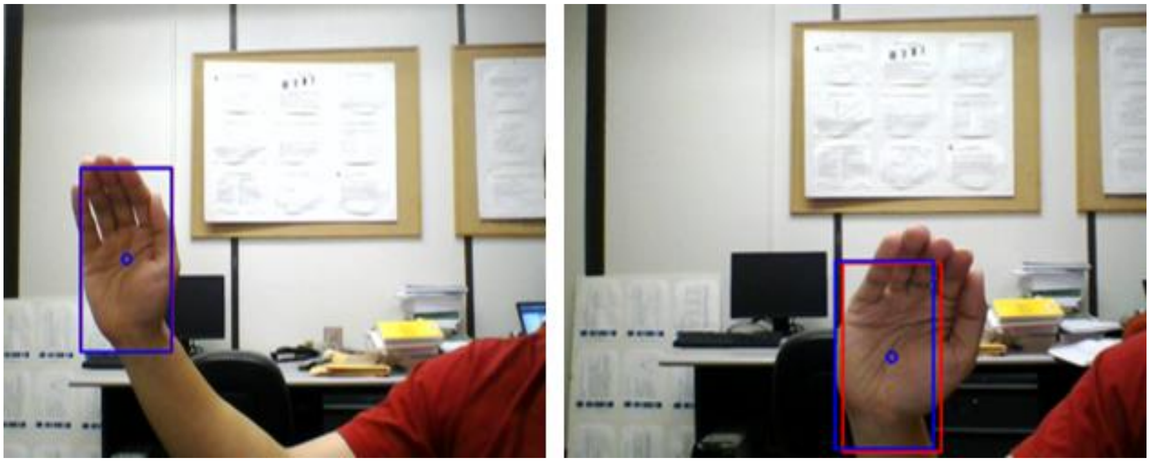


Figure 16. Captures of tracking implementation on the movement of a hand. Kalman filter (red box) continuously records the correct position of the center of window

As mentioned above, the tracking window only analyzes pattern of ROI, therefore, it can track any type of motion or object.

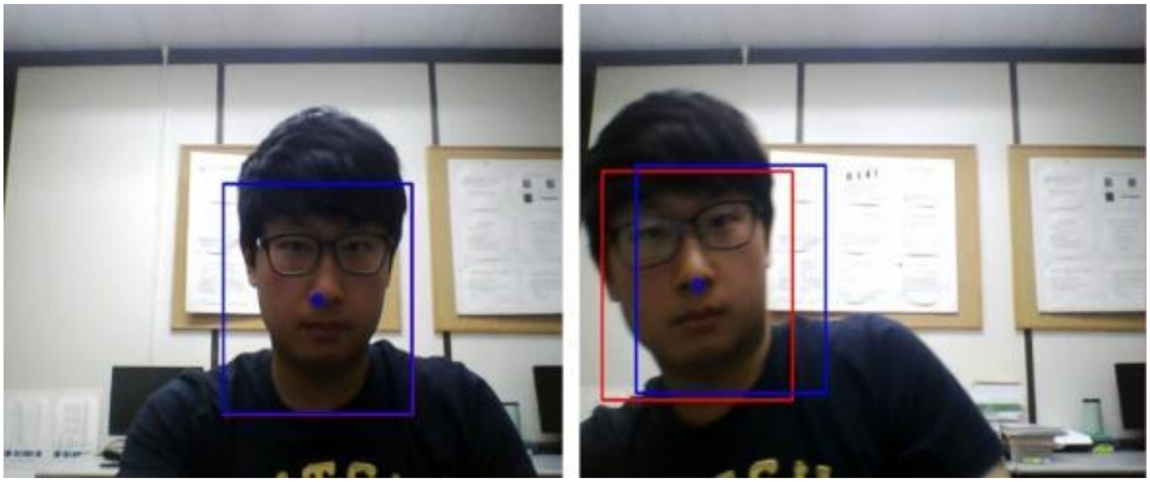


Figure 17. Captures of tracking the head movement

2.4 Discussion

In the previous chapter, theoretical basis of the proposed method was discussed. However, appropriate parameters and input variables were obtained by experiments. For example, setting termination criteria for optimal mean-shift is shown as follows

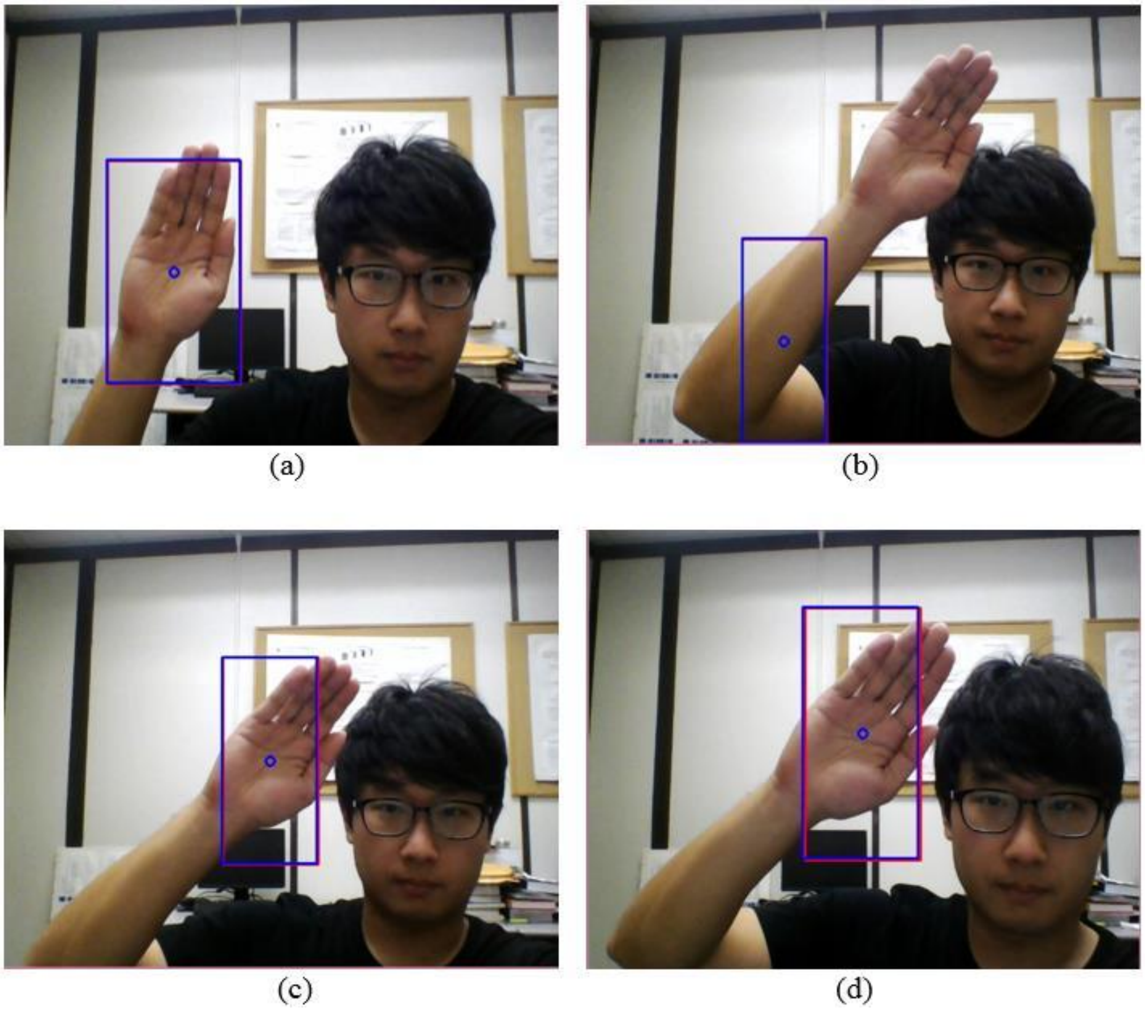
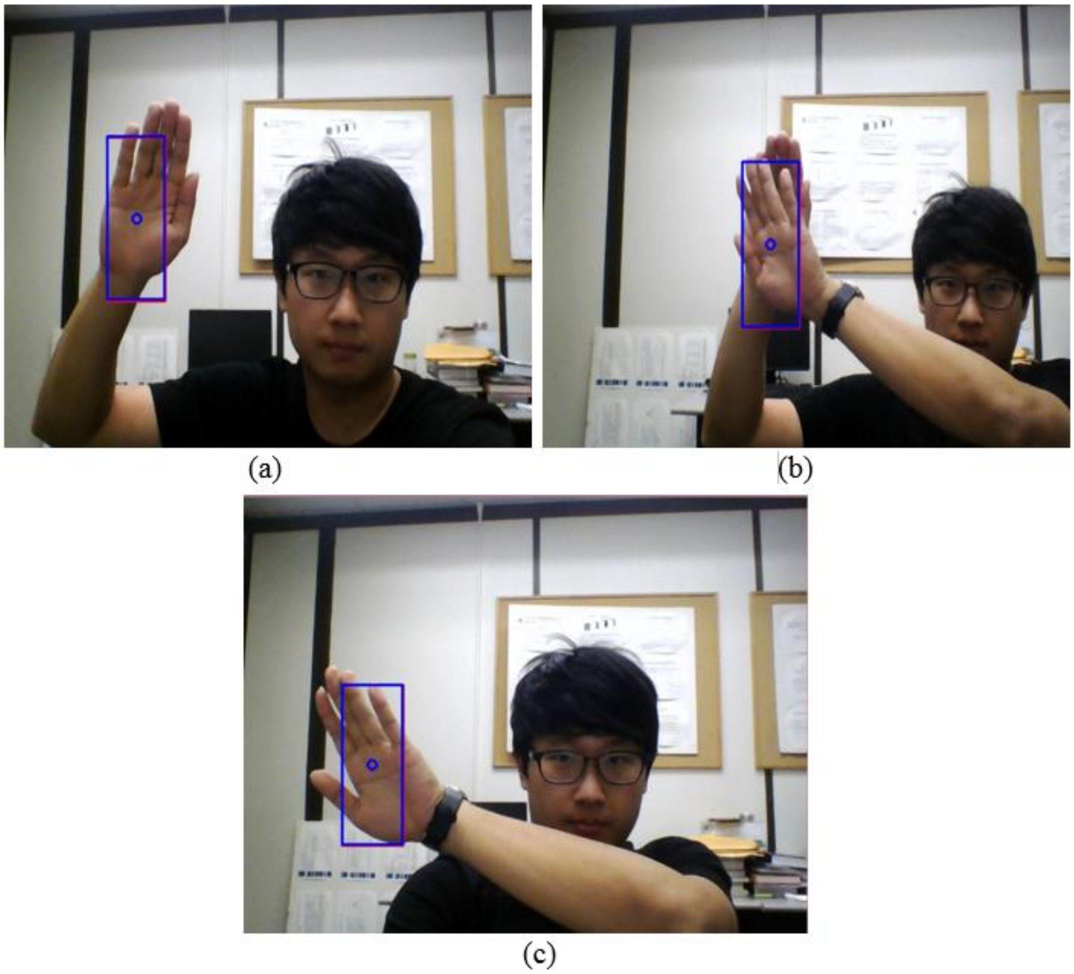


Figure 18. Mean-shift tracking with different iterations termination: (a) Original image (b) 3 iterations (c) 10 iterations (d) 50 iterations

Figure 18(b) shows that the window fails to track the original information. This is because three iterations are insufficient opportunities to converge to true information. It is obvious that the larger number of iteration the criteria has, the more precise tracking the system achieves. However, it seems that tracking with 10 iterations results in sufficiently remarkable motion capturing. To establish real-time system, the amount of computation should be small as possible.

An occlusion in tracking means that the target is covered by outliers for some period. Occlusions commonly happens while tracking with mean-shift and they significantly undermine the performance of motion tracking. To improve the tracking performance, the proposed system amends the occlusion-interfered erroneous target location by employing Kalman filter estimation. Thus, correct target location can be obtained. However, the potential problem is that similarity measures are not discriminative if an occlusion is caused by the object similar with the target. The figure 19 shows the example of problematic occlusion. The figure 19(a) illustrates that the right hand is chosen as the ROI and the tracking window holds the target in good shape. But, in figure 19(b) the target is overlapped by the left hand which has completely same feature information with that of the target. As a result, the window loses the original target (right hand) and tracks the left hand.



(a) (b)

(c)

Figure 19. The example of occlusion

The solution of this occlusion may be achieved by effectively analyzing the occlusion situation to generate proper template mask. Moreover, since tracking based on color histogram precludes the application of more elaborate motion models, employing other types of similarity measures could also be good solution.(10)

3. CONCLUSION AND FUTURE WORK

3.1 Conclusion

A surveillance system with real-time motion detection and tracking system is proposed in this paper. The system employ a webcam as hardware preliminary and C++ API OpenCV as a software application. Differential image technique is a very simple algorithm but extremely effective in extracting changes in pixel value which denotes motion detection. Mean shift technique tracking the center of selected object is used for tracking algorithm. Although mean shift algorithm is prone to be affected by noise, the system continuously correct errors using a Kalman filter that iteratively predicts true value.

The entire system consists of low-energy and less complexity image processing techniques. Therefore, the proposed system in this paper is considered as a cost-effective solution for an unmanned surveillance platform which performs competitively compared to costly conventional systems.

To complete the system, the way of selecting ROI should be devised. Then, the selected ROI will be tracked by mean shift algorithm with Kalman filter. The proposed method is that the added window navigates throughout the FOV until it finds the region that includes largest number of nonzero elements. The optimal size of window will be determined by experiments, considering characteristics of environment.

3.2 Future work

As shown above results, once a motion is detected as the ROI, the proposed system successfully tracks it thanks to robust functionality of mean-shift algorithm and Kalman filter. Therefore, one of most important thing in this project is to clarify what to detect and track. Drawing a bounding box around the ROI is a most frequently used method that denotes the detection. The easiest way of creating a bounding box is to use tracking API in OpenCV. There is a function named ‘selectROI’ in the version of OpenCV 3.0. More classical way is as follows

1. Calculate the convex polygon which has a smallest contour that contains all input point set.
2. Obtain two points on the polygon: the most upper right and the most lower left. (or upper left of lower right).
3. Then, draw a rectangle having the above two points as its diagonal points.

Furthermore, the detection system can be improved by introducing an automatic decision criteria for detection while the current system manually draws a ROI. For example, setting a threshold for time delay or an area of polygon can be solution.

Mean-shift algorithm has a drawback. The tracking window fails to track the true ROI if it encounters the region that has a similar PDF with ROI’s one, but it is not the ROI. As discussed in the introduction part, cam-shift algorithm can result in more accurate tracking because it adaptively resizes a tracking window while mean-shift algorithm uses a fixed window. The real-time tracking equipped with cam-shift can be a great research topic.

This thesis suggests a real-time motion detection and tracking algorithm for single stationary camera while the most of commercial and industrial fields demand all-directional surveillance system. To broaden the range of angle, use of multiple cameras or PTZ(pan-tilt-zoom) camera is recommended. The challenge of multi-angle camera research is that geometrical distortions should be considered between frames due to increased space dimensionality and relative movement between camera and object. Thus, future work should be directed toward robust interpretation of sequential position information.

Nowadays, the research trend of video surveillance is focused on embedded system camera. Rapidly growing machine learning technologies enables the device to train its observations to gather and analyze information like a human brain. For example, feature selection or extraction using machine learning is used for automatic detection of a human face. Clustering patterns of the ROI is becoming an emerging state-of-the-art tracking methods and Kalman filter tracking in collaboration with machine learning is also actively studied. Thus, video surveillance controlled by smarter AI would no longer be imaginary technology.

REFERENCES

1. Dyson WE. Terrorism: An investigator's handbook: Routledge; 2011.

2. Sears CR, Pylyshyn ZW. Multiple object tracking and attentional processing. Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale. 2000;54(1):1.

3. Sulman N, Sanocki T, Goldgof D, Kasturi R, editors. How effective is human video surveillance performance? Pattern Recognition, 2008 ICPR 2008 19th International Conference on; 2008: IEEE.

4. Leslie AM, Xu F, Tremoulet PD, Scholl BJ. Indexing and the object concept: developingwhat'andwhere'systems. Trends in cognitive sciences. 1998;2(1):10-8.

5. Collins RT, Lipton AJ, Kanade T, Fujiyoshi H, Duggins D, Tsin Y, et al. A system for video surveillance and monitoring. 2000.

6. Javed O, Shah M. Automated video surveillance. Automated Multi-Camera Surveillance: Algorithms and Practice. 2008:1-9.

7. Fukunaga K, Hostetler L. The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Transactions on information theory. 1975;21(1):32-40.

8. Cheng Y. Mean shift, mode seeking, and clustering. IEEE transactions on pattern analysis and machine intelligence. 1995;17(8):790-9.

9. Comaniciu D, Ramesh V, Meer P. Kernel-based object tracking. IEEE Transactions on pattern analysis and machine intelligence. 2003;25(5):564-77.

10. Yang C, Duraiswami R, Davis L, editors. Efficient mean-shift tracking via a new similarity measure. Computer Vision and Pattern Recognition, 2005 CVPR 2005 IEEE computer society conference on; 2005: IEEE.

11. Bradski GR. Computer vision face tracking for use in a perceptual user interface. 1998.

12. Kalman RE. A new approach to linear filtering and prediction problems. Journal of basic Engineering. 1960;82(1):35-45.

13. Welch G, Bishop G. An introduction to the Kalman filter. 1995.

14. Grewal MS, Andrews AP. Applications of Kalman filtering in aerospace 1960 to the present [historical perspectives]. IEEE Control Systems. 2010;30(3):69-78.

15. Daum F. Nonlinear filters: beyond the Kalman filter. IEEE Aerospace and Electronic Systems Magazine. 2005;20(8):57-69.

16. Verstraeten C. OpenCV Simple Motion Detection [Available from: https://blog.cedric.ws/opencv-simple-motion-detection.

17. Stein M. [cited 2017]. Available from: http://www.steinm.com/blog/motion-detection-webcam-python-opencv-differential-images/.

18. Shen Y, Li S, Zhu C, Chang H, editors. Task Specific Top-Down Visual Attention Based on Local Pattern Analysis and Histogram Backprojection for Fast Object Localization. Computer and Information Technology (CIT), 2014 IEEE International Conference on; 2014: IEEE.

19. Surkutlawar S, Kulkarni RK. Shadow suppression using RGB and HSV color space in moving object detection. International Journal of Advanced Computer Science and Applications. 2013;4(1).

20. Dubuisson S, editor The computation of the Bhattacharyya distance between histograms without histograms. Image Processing Theory Tools and Applications (IPTA), 2010 2nd International Conference on; 2010: IEEE.

21. Ali NH, Hassan GM. Kalman filter tracking. International Journal of Computer Applications. 2014;89(9).