Speech, Image, and Language Processing for Human Computer Interaction:

Multi-Modal Advancements

Uma Shanker Tiwary
Indian Institute of Information Technology Allahabad, India

Tanveer J. Siddiqui
University of Allahabad, India

Information Science
REFERENCE

Managing Director:	Lindsay Johnston
Senior Editorial Director:	Heather A. Probst
Book Production Manager:	Sean Woznicki
Development Manager:	Joel Gamon
Development Editor:	Myla Harty
Acquisitions Editor:	Erika Gallagher
Typesetter:	Jennifer Romanchak
Cover Design:	Nick Newcomer, Lisandro Gonzalez

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Speech, image, and language processing for human computer interaction : multi-modal advancements / Uma Shanker Tiwary and Tanveer J. Siddiqui, editors.
p. cm.
Includes bibliographical references and index.
Summary: "This book identifies the emerging research areas in Human Computer Interaction and discusses the current state of the art in these areas"--Provided by publisher.
ISBN 978-1-4666-0954-9 (hardcover) -- ISBN 978-1-4666-0955-6 (ebook) -- ISBN 978-1-4666-0956-3 (print & perpetual access) 1. Human-computer interaction. I. Tiwary, Uma Shanker, 1960- II. Siddiqui, Tanveer.
QA76.9.H85S654 2012
004.01'9--dc23
2011047364

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

Chapter 10

Multi Finger Gesture Recognition and Classification in Dynamic Environment under Varying Illumination upon Arbitrary Background

Armin Mustafa
Samsung India Software Operations, India

K.S. Venkatesh
Indian Institute of Technology Kanpur, India

ABSTRACT

This chapter aims to develop an 'accessory-free' or 'minimum accessory' interface used for communication and computation without the requirement of any specified gadgets such as finger markers, colored gloves, wrist bands, or touch screens. The authors detect various types of gestures, by finding fingertip point locations in a dynamic changing foreground projection with varying illumination on an arbitrary background using visual segmentation by reflectance modeling as opposite to recent approaches which use IR (invisible) channel to do so. The overall performance of the system was found to be adequately fast, accurate, and reliable. The objective is to facilitate in the future, a direct graphical interaction with mobile computing devices equipped with mini projectors instead of conventional displays. The authors term this a dynamic illumination environment as the projected light is liable to change continuously both in time and space and also varies with the content displayed on colored or white surface.

1. INTRODUCTION

With an ever increasing role of computerized machines in society, the need for more ergonomic and faster Human Computer Interaction (HCI) systems has become an imperative. Here, we aim to develop an 'accessory-free' or 'minimum accessory' interface suitable for communication and computation without the requirement of often used gadgets such as finger markers, colored gloves, wrist bands, or touch screens. We describe here a robust method to detect various types of gestures.

DOI: 10.4018/978-1-4666-0954-9.ch010

Our approach works by locating different salient parts, specifically fingertips, in a spatio-temporally dynamic foreground projection. This projection itself constitutes the varying illumination which the gestures have to be detected: moreover, we allow this projection to fall upon a nearly arbitrary background surface. The overall performance of the system was found to be adequate for real-time use, in terms of speed, and accurate and reliable enough in a practical setting. The long term objective is to eventually facilitate a direct graphical interaction with mobile computing devices equipped with mini projectors instead of conventional displays. It must be noted that unlike the conventional setting in which intrusions are detected as regions of major change in a 'learned' static background, our 'background' is in fact the instantaneous displayed output of the computing device, and is therefore generally liable to vary in space and time with the content displayed. Furthermore, keeping in mind the exigencies of anywhere-anytime computing, the system we propose does not require a plain white surface to be available to display upon: instead, it only requires that the surface should have at all points, a certain minimum non-specular reflectance and also be planar, even if not strictly normal to the projector-camera axis. According to most currently reported approaches, such an unconstrained problem specification would necessitate the use of an IR (invisible) channel for the finger intrusion detection. Our approach operates exclusively with visual detection and applies the principle of reflectance modeling on the scene where intrusion needs to be detected and also on intrusion which in our case is hand to achieve this. Briefly, it consists of the following two steps:

1. A process we call Dynamic Background Subtraction, under varying illumination upon an arbitrary background using a reflectance modeling technique that carries out visual detection of the shape of intrusion on the front side projected background. This particular process in patented by us in India (Application No: 974/DEL/2010)
2. Detecting the gestures and quantifying them: this is achieved by specially tuned light algorithms for the detection of the contour trajectory of the intruding hand through time, and tracking multiple salient points of this intrusion contour. Gestures can then be classified and subsequently quantified in terms of the extracted multi trajectory parameters such as position, velocity, acceleration, curvature, direction, etc.

A special, simplified, case of the above general approach is the demonstrated Paper Touchpad which functions as a virtual mouse for a computer, operating under conditions of stable (non-dynamic) illumination on arbitrary backgrounds, with the requirement of a single webcam and a piece of paper upon which the 'touchpad' is printed. This is an interactive device easy to use anywhere, anytime and employs a homographic mapping between screen and piece of paper. The paper touchpad, however, does not obviate the display.

In the end, we aim to design a robust real time system which can be embedded into a mini-projector and camera equipped mobile device that can be used without accessories anywhere a flat surface and some shade (from excessively bright light such as direct sunlight) is available. The single unit would substitute for the computer or communicator, the display, keyboard, mouse, a piano, a calculator etc.

2. RELATED WORK

Most reported techniques have usually been using some gadgets/accessories or other sort of assistive tools. For example visual ways to interact with the computer using hand gestures involved the use of a rather unique and quite ingenious Omni-directional sensor (Hofer, Naeff, & Kunz, 2009), which adds to the system cost and assumes the availability of the Omni-directional sensor. Many other researchers have studied and used glove-based or wrist band based devices (Oka, Sato, & Koike, 2002) to measure hand postures, shape and location with high accuracy and speed, especially for virtual reality. But they aren't suitable for several applications because the cables connected to the gloves restrict unfettered hand motion. Besides, the apparatus is difficult to carry and use anywhere and anytime. Some have also proposed hand gesture recognition in which the camera was placed a few meters away (Lockton, 2009), but this can't be used for direct interaction with a computer system in the most common modes of computer use. The method in Kim, Kim, and Lee (2007) detects hand and finger gestures in projected light but they require a color camera, an infrared camera, front and rear projection and are used specifically for an augmented desk interface as opposed to our system, which just requires one color camera and one projector and can be used nearly anywhere.

At the same time, single- and multitouch technologies, essentially touch (contact detection) based, were used for human computer interaction, employing devices like a touch screen (e.g., computer display, table and wall) or touchpad, as well as software that recognizes multiple simultaneous touch points. But this requires the use of specifically multi touch hardware surfaces and specific systems interfaced with it as observed in Grossman, Balakrishnan, Kurtenbach, Fitzmaurice, Khan, and Buxton (2001) and Fukumoto, Suenaga, and Mase (1994). The techniques used were mostly amongst the following: Frustrated Total Internal Reflection (FTIR), Rear Diffused Illumination (Rear DI) such as Microsoft's Surface Table, Laser Light Plan (LLP), LED-Light Plane (LED-LP) and finally Diffused Surface Illumination (DSI) to be found in Segan and Kumar (1999). Such specialized surfaces hardly qualify for anywhere-anytime application.

Certain optical or light sensing (camera) based solutions were attempted sometime later. The scalability, low cost and ease of setup are suggestive reasoning for the popularity of optical solutions. Each of these techniques consists of an optical sensor (typically a camera), infrared light source, and visual feedback in the form of projection or LCD as seen in Wu, Shah, and Lobo (2000) and Eldridge and Rudolph (2008). Monocular camera views were also used for 3D pose estimation of hand gestures like in Shimada, Shirai, Kuno, and Miura (1998) which was very computationally expensive

Infrared imagings for building an interface as in Lin and Chang (2007) and augmented desktops as in Han (2005) and Oka, Sato, and Koike (2002) also appear in the literature. Such techniques employ infrared cameras, infrared light source, IR LED's with few inches of acrylic sheets, baffles, compliant surfaces etc. for proper operation. Similarly, Westerman, Elias, and Hedge (2001) requires capacitive sensor array, keyboard and pointing device. All these types of Multi touch devices used for HCI require complicated setups and sophisticated devices which make the system much more costly and difficult to manage. Similarly, Kim, Kim, and Lee (2007) used infrared cameras to segment skin regions from background pixels in order to track two hands for interaction on a 2D tabletop display. Their method then used a template matching approach in order to recognize a small set of gestures that could be interpreted as interface commands. However, no precise fingertip position information was obtained using their technique.

After some time, techniques using Stereo Vision came into existence but didn't gain much popularity because of certain drawbacks like the need for some complex calibration and the subject having to adjust according to the needs of the camera, which makes it difficult to use for real-life situations (Mitra & Acharya, 2007) as well as diminishing user friendliness. Some have used simple CRT/LCD displays but the capture was done with two cameras placed at two different accurately maintained angles (Thomas, 1994) which were again unsuitable for day to day applications.

Many approaches use markers attached to a user's hands or fingertips to facilitate their detection in the video stream captured by the camera as seen in Mistry, Maes, and Chang (2007). While markers help in more reliably detecting hands and fingers, they present obstacles to natural interaction similar to glove-based devices, though perhaps less cumbersome than the former. Besides, the user has to remember to carry the markers around without fail. A few works provide a comprehensive survey of hand tracking methods and gesture analysis algorithms (Jones & Rehg, 1999). But these are meant for whole-body gestures which are unsuitable for acting as a direct interface with the computer or any system for a seated subject 'before a desk'.

Depth information obtained using 2 cameras were used for classification into background and foreground in Gordon, Darrell, Harville, and Woodfill (1999). Another approach is to extract image regions corresponding to human skin by either color segmentation or background image subtraction or both. Because human skin isn't uniformly colored and changes significantly under different lighting conditions, such methods often produce unreliable segmentation of human skin regions and are user (skin color) dependent. Methods based on background image modeling followed by subtraction also prove unreliable when applied to images with a complex background and time varying illumination conditions as in Thomas (1994). For an effective and practical solution, we need that the system, even with a dynamic background must give good results. Only this will allow our proposed approach to operate directly under fore-projected illumination to come to fruition.

After a system identifies image regions in input images, it can analyze the regions to estimate hand posture. Researchers in Pavlovic, Sharma, and Huang (2001) and Vladimir, Rajeev, and Thomas (1993) have developed several techniques to estimate pointing directions of one or multiple fingertips based on 2D hand or fingertip geometrical features. Wu, Lin, and Huang (2001) uses a 3D human hand model for hand gesture analysis. To determine the model's posture, this approach matches the model to a hand image obtained by one or more cameras as seen in Yuan and Zhang (2010) and Wren, Azarbayejani, Darrell, and Pentland (1997). Using a 3D human hand model solves the problem of self-occlusion, but these methods don't work well for natural or intuitive interactions because they're too computationally expensive for real-time processing and require controlled environments with a relatively simple background.

Moreover, all these approaches either assume the background to be static or use infrared light as a fourth invisible channel for visual segmentation. In Hofer, Naeff, and Kunz (2009) detection of hand gestures for the replacement of mouse is shown but this works only for a static background, and we also need a separate monitor for operation. Our formulation is generic and aims to replace monitor, keyboard, piano, mouse etc. Motonorihi, Ueda, and Akiyama (2003) and Helman, Juan, Leone, and Aderito (2007) detect hand gestures but not in dynamic background and highly changing lighting conditions and does not eliminate the use of monitor, keyboard etc. Also in Utsumi and Ohya (1999), hand gestures are detected to interact with sound/music system which is a rather specific application, whereas our system is much more general. Similarly, in Jenab and Reiterer (2008) finger movement is used to interact with the mobile screen but only a few gestures are supported, suitable for a static background only. Apart from all this, Xing, Wang, Zhao, and Huang (2009) gives a survey of almost all existing algorithms in gesture recognition for various applications and most of them used Hidden Markov Models, Finite State Machines, Artificial Neural Networks, wavelets etc. but none of the existing algorithm satisfies all the criteria and conditions of our problem.

In an era where people dislike carrying large gadgets, or complex setups and assistive tools or accessories with them, we need to rework our paradigm. It isn't enough to simply make the devices smaller and better. We need a 'minimum accessory interface' which uses visual segmentation techniques for foreground segmentation on dynamic backgrounds, in fact, even operates under projector front-illumination conditions, and forgoes an infrared channel. With reducing prices of cameras and projectors, our proposed system should eventually become cost-competitive and replace hardware like items mouse, keyboard, monitor etc.

3. OUR APPROACH: PRINCIPLES, ASSUMPTIONS AND CALIBRATION

3.1. The Projector Camera System

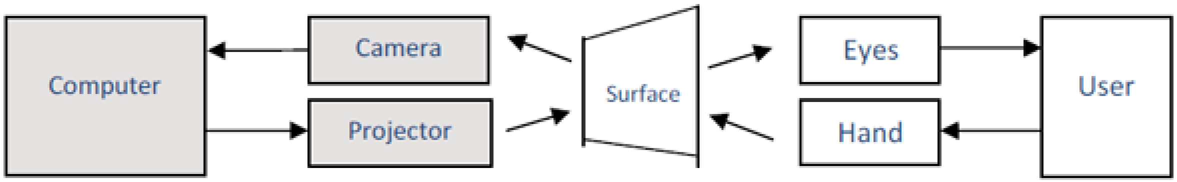
Projection systems are used for various esoteric purposes such as to implement augmented reality, as well as to simply create both displays and interfaces on ordinary surfaces. Ordinary surfaces have varying reflectance, color (texture), and geometry. These variations can be accounted for by integrating a camera into the projection system and applying methods from computer vision. Projector-camera systems became popular in these years, and one of the popular purposes of them is 3D measurement. The only difference between camera and projector is the direction of flow of optical information. 3D scene is projected onto the 2D image plane in camera; and 2D pattern in projector is projected onto 3D scene. Hence, a straightforward solution for projector calibration is using camera calibration methods, which generally require 3D-2D projection maps (homographies).

In most such applications, the projector serves, that is to say, interacts with, only the camera. Our system uses a projector-camera pair in a subtly different and more powerful role: both to provide device output to the user and to simultaneously provide user input to the system. Both the camera and the user utilize the projector's output, and both the user and the projector provide necessary inputs to the camera. In short, the projection surface serves as the medium of communication between user and machine. The set-up is shown in the Figure 1.

3.2. Surface Types and Reflectance

For our purpose in the following discussion, we can afford to use the terms *reflectivity* and *reflectance* interchangeably, as our concern is with thick surfaces whose reflection properties are not contributed to by thin reflection (multi layered surface) phenomena. Reflectivity is a directional property, and on the basis of this directionality, most surfaces can be grossly divided into those that are specular reflectors and those that are diffuse reflectors, though we keep in mind that the terms specular and diffuse are relative. For specular surfaces, such as glass or polished metal, reflectivity will be nearly zero at all angles except at the appropriate reflected angle. For diffuse surfaces,

Figure 1. Projector camera system



such as matte white paint, reflectivity is uniform; radiation is reflected in all angles equally or near-equally. Such surfaces are said to be Lambertian. Most real objects have some mixture of diffuse and specular reflective properties.

Besides, the output of a light-sensitive device depends on the color of the light incident upon it. Radiant sensitivity is considered as a function of wavelength i.e., the response of a device or material to monochromatic light is a function of wavelength, also known as spectral response. It is a means of relating the physical nature of change in light to the changes in image and color spaces. Recent computational models of color vision demonstrate that it is possible to achieve exact color constancy over a limited range of lights and surfaces described by linear models. The success of these computational models hinges on whether any sizable range of surface spectral reflectances can be described by a linear model with about three parameters. A visual system exhibits perfect or exact color constancy if its estimates of object color are unaffected by changes in ambient lighting. Human color vision is approximately color constant across certain ranges of illumination, although the degree of color constancy exhibits changes with the range of lighting examined.

As long as the lighting and surface spectral reflectances in the scene approximately lie within limited ranges, the color estimates are approximately correct. Spectral response on the plane upon which projection takes place will differ with the spectral response of the intruding object, thus giving evidence of intrusion. We have used the concept of reflectance modeling in our work. The reflectances of various objects like hand, arbitrary background etc to create different models which are in turn used for intrusion detection under varying illumination. Since it is not the appearance of the surface that is being modeled, but its reflectance, intrusion detection becomes possible over a wide range of spatially and temporally varying illumination conditions. Using these concepts we develop an algorithm which first models, and then uses, reflectance properties of the projection surface to detect intrusion.

3.3. Assumptions

We expect the surface, the user and the system (consisting of the computing device, its projector and its camera) to meet some general criteria.

1. The surface must be near-flat, with a Lambertian (non-specular) reflectivity uniform over the projection surface.
2. The reflectance coefficient is not too low either at any wavelength in the visible band, or at any point or region upon the surface. This is the first part of a *singularity-avoidance* requirement.
3. We allow the surface to possess some space-varying texture, subject to meeting the criteria set down above at each point individually.
4. Surface reflectance spectrum should differ sufficiently from that of human skin at all points.
5. We allow ambient illumination to be present, and to have any spectral bias, so long as its intensity allows the projector output to sufficiently dominate and so long as the ambient illumination is constant over time during the entirety of a user session.
6. Ambient illumination must preferably be non-specular (diffuse) to avoid shadow casting.
7. There should not be instances of regions or times where both ambient and projector illumination are zero, resulting in very dark regions on the surface. This is the second part of the *singularity-avoidance* requirement

8. The camera-projector combination is assumed to be fixed in space relative to each other (by sharing a common chassis, for example) as well as fixed in space relative to the projection surface during an interaction session.
9. Bounded depth: While capturing the videos, the light intensity reflected by the fingers should be nearly constant to avoid abrupt intensity changes due to intrusions occurring too close to the camera/projector. This is ensured by keeping the hand and fingers close to the projection surface at all times. In other words, the depth variation across the projection surface during the gesturing action should be a reasonably small fraction of the camera/projector distance.
10. Each finger gesture video should be brief and last for no more than about 3-4 seconds. Longer gesture times will delay the identification of the gesture as identification and appropriate consequent action is only possible after each gesture performed is completed.
11. The optics of projector and camera are kept as nearly co-axial and coincident as possible to reduce the shadow and parallax effects.
12. We confine ourselves to an image size of 640 x 480 pixels because larger sizes, while improving spatial resolution of the gestures, would increase the computational burden, and adversely affect real time performance.
13. At a maximum, 2 fingers were used to make a proper sign. This choice varies from signer to signer and programmer to programmer. More the skin region, more is the complexity of the coding for tracking the motion of the fingers.
14. Good computing power.
15. Camera, preferably without AGC and white balance adaptation

3.4. Experimental Setup and Calibration

Under the abovementioned set of assumptions, the system's operation during a session is initiated with a session calibration phase which process consists of the following steps in sequence.

1. Calibration to ambient illumination.
2. Calibration to skin color under ambient illumination.
3. Surface texture calibration under projector illumination.
4. Skin color calibration under projector illumination.
5. Camera-projector co-calibration for white balance.

Apart from all these session-specific parameter settings, the system has to be one-time factory-calibrated to map camera and projector spatial and temporal resolutions to one other.

Gestures are captured through the use of a single web camera facing towards the hand of the user and the projection surface. The details of calibration will be discussed in Section 4.

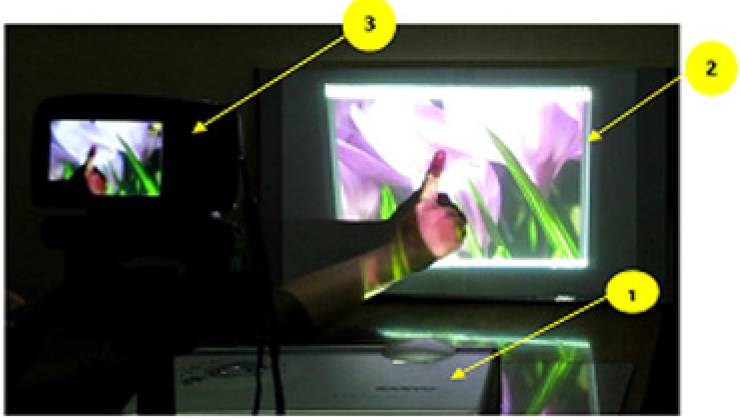
The experimental setup of our system is shown in the Figure 2. Dynamic data is projected on a surface and gesture is performed on respective surface which is captured by camera and the video stream obtained is processed to define the gestures.

4. RELIABLE INTRUSION DETECTION UNDER PROJECTOR ILLUMINATION

4.1. The Statistical Gaussian Model Based Approach

What we present in the following is, to the best of our knowledge, original. There are hardly any reports we could find in the literature we could find dealing with intrusion detection in a

Figure 2. Experimental setup of the invention: 1-Projector,2-Screen on which random scenes are being projected and hand is inserted as an intrusion and 3-Camera recording the screen



dynamically illuminated environment. In Nadia and Cooperstock (2004) the authors deal with intrusion detection in camera projector system handles geometry and color compensation, but does not gives any compensation for factors like Luminance, Camera parameters etc. The methods outlined in this subsection were actually developed and implemented at our lab chronologically prior to the reflectance modeling approach we present next. This approach is more pedagogically intuitive, and while original, represents less radical innovation than reflectance modeling.

In the process of arriving at a method that effectively achieves our goals, we first describe an approach to change/intrusion detection that is more preliminary and is in common use: it makes more assumptions about the environment such as that no ambient illumination is present (an unrealistic assumption). Further it reality does not constitute what may properly be termed reflectance modeling in the rigorous sense, as surface and skin reflectance models are not estimated. Thus the performance of the preliminary approach we present in this section is markedly inferior under even compliant conditions, and places more restrictions upon the environment. On the other hand, we do choose to present it in some detail because this method was actually first implemented before our more refined final approach was conceived. It also has some pedagogic value, as it directly addresses many of the most important challenges of the problem of intrusion detection. Extracting intrusion based on color image segmentation or background subtraction often fails when the scene has a complicated background and dynamic lighting. In the case of intrusion monitoring, simple motion detection, or an approach based on color modeling, may be sufficient. But variations in lighting conditions, constantly changing background, as well as camera hardware settings and behavior complicate the intrusion detection problem. It is often necessary to cope with the phenomenon of illumination variations as it can falsely trigger the change detection module. Further, motion detection as a means of intrusion detection may also fail in the scenario we plan to work in, where the background can be dynamic, with moving entities flying across the screen at times giving rise to what may be termed spurious flow artifacts. The information in each band of the RGB color space of the video sequences activates our pixel wise change detection algorithm in the observed input frame in spite of a continuously changing background. This is achieved by recursively updating the background on the basis of the known projected information and seeking conformance in each camera captured frame to the current reference frame. Ordinary surfaces can have space varying reflectance, color, and geometry. These variations can be accounted for by integrating a camera into the projection system and applying methods from computer vision. The methods currently in use are fundamentally limited since they assume the camera, illumination, and scene as static. Steps involved in the method are as follows:

Step 1: Before we start our learning phase we need to assume that the projector screen surface has complete uniformity.
Step 2: Matching frames of the projected and captured videos

In the experiment conducted we fixed the no of frames in both captured and projected video and hence calibrated and matched the captured and projected videos.

- Projected video has 100 frames between two black frames
- Captured video has 500 frames between two nearest black frames
- Result: 1 frame of projector was temporally equal to 5 frames of captured video

Step 3: Calibration of colors for projector camera system:

Pure red, green and blue colors are sent via the projector and captured by the camera for a set of *'n'* frames. The camera output is not pure red, green or blue. Here, every pure input has all its corresponding response RGB components non zero. This is on account of an imperfect color balance match between projector and camera.

Since the color values which are projected and the ones which are captured from the camera don't match we carry out color calibration over *'n'* frames.

Considering the red input only:

a. Find the mean red, mean green and mean blue of the output for the *'n'* frames
b. Find the maximum and minimum for each red, green and blue output from the *'n'* frames.
c. Find the difference between maximum and the mean value for every RGB output component for the red input which gives the deviation.
d. Follow the same procedure for green as well as blue input for *'n'* frames.

The projected RGB values are represented by R_P, G_P and B_P and the corresponding camera captured colour values are represented by R_C, G_C and B_C. With the projector output held constant, we in fact capture '*n*' frames with the camera for the purpose of statistical averaging over camera noise, and denote these time indexed camera capture components as $R_C(t)$, $G_C(t)$ and $B_C(t)$.

An imperfect match between the white balances of projector and camera results in a certain amount of crosstalk between different colour channels, so that $R_P = [255\ 0\ 0]^T$; $G_P = [0\ 255\ 0]^T$; $B_P = [0\ 0\ 255]^T$, whereas $R_C = [R^r_C\ R^g_C\ R^b_C]^T$; $G_C = [G^r_C\ G^g_C\ G^b_C]^T$; $B_C = [B^r_C\ B^g_C\ B^b_C]^T$. Each component X^y_C ; X = R, G, B and y = r, g, b in each output vector of these equations is the time and space average of the *n* resp. captures:

$$X_C^{\,y} = \frac{\sum_{k,l=1,1}^{M,N} \sum_{t=1}^{n} X_C^{\,y}(t)}{n \times M \times N} \tag{1}$$

where *M, N* are the numbers of pixel rows and columns in the captured image, and *k,l* are the row and column indices.

Step 4: Now for detecting the intrusion blob we need to calculate the mean and maximum deviation for each input RGB component. Ideally, the mean as well as the deviation for each RGB output component for every individual pure input is determined. For simplicity we take the mean and maximum values only. x_r, x_g, x_b are respective maximum deviations from mean value for red, green and blue components

$$\boldsymbol{x_r = R^r_{max} - R^r_{C},\ x_g = R^g_{max} - R^g_{C},\ x_b = R^b_{max} - R^b_{C}} \quad (2)$$

where R^r_{max}, R^g_{max}, R^b_{max} are the extreme red green and blue components under the red input. Similarly, we define and compute:

$$\boldsymbol{y_r = G^r_{max} - G^r_{C},\ y_g = G^g_{max} - G^g_{C},\ y_b = G^b_{max} - G^b_{C}} \quad (3)$$

$$\boldsymbol{z_r = B^r_{max} - B^r_{C},\ z_g = B^g_{max} - B^g_{C},\ z_b = B^b_{max} - B^b_{C}} \quad (4)$$

Step 5: Formation of color bias matrix: This matrix is formed by the mean values and maximum deviations in each of the red, green and blue inputs and outputs. The color bias matrix is as shown in Equation (5).This matrix is used to calculate the expected values by performing matrix multiplication with known input

$$\begin{pmatrix} R^r{}_C & G^r{}_C & B^r{}_C \\ R^g{}_C & G^g{}_C & B^g{}_C \\ R^b{}_C & G^b{}_C & B^b{}_C \end{pmatrix} \quad (5)$$

Step 6: Calculating the total maximum deviations in RGB

The total deviation for each component is the sum of deviation or variance at each input. To find these values we need to follow the equation given below:

Dev(R) =deviation due to red input + deviation due to green input+ deviation due to blue input (6)

$$dev(R) = \sigma(\mathrm{R}) = x_r + y_r + z_r \quad (7)$$

Similarly,

$$dev(G) = \sigma(\mathrm{G}) = x_g + y_g + z_g \quad (8)$$

$$dev(B) = \sigma(\mathrm{B}) = x_b + y_b + z_b \quad (9)$$

Step 7: Finding expected values

- Project the video
- Convert it into number of frames
- Every pixel of every single frame is now decomposed into its RGB components
- These RGB values are then normalized by dividing each by 255
- Now we multiply this normalized RGB with the color bias matrix to get the expected values

For any single pixel p(i,j) of the projected video, let the value of RGB components be given by $[R,G,B]^T$.To calculate the expected value in the absence of intrusion, we need to do matrix multiplication of the pixels RGB values and the color bias matrix. Let the final expected (normalized) values for the red, green and blue be R_e, G_e and B_e, calculated as shown in Equation (10).

$$\begin{pmatrix} R^r{}_C & G^r{}_C & B^r{}_C \\ R^g{}_C & G^g{}_C & B^g{}_C \\ R^b{}_C & G^b{}_C & B^b{}_C \end{pmatrix} * \begin{pmatrix} R \div 255 \\ G \div 255 \\ B \div 255 \end{pmatrix} = \begin{pmatrix} R_e \\ G_e \\ B_e \end{pmatrix} \quad (10)$$

Step 8: Steps for finding the observed values

- Interpolate and resize the captured video to projected video for pixel matching.
- Convert the captured videos to frames
- Every pixel of every single frame is now decomposed into its RGB components
- Intrusion detection is done according to the equations below
- Equations are derived which relate the image coordinates in the camera to the external coordinate system.

Step 9: Each red green and blue will have their individual Gaussian models. According to the Statistical Gaussian models obtained above we can do background subtraction by defining a range of around '$2\sigma_x$' around the mean which constitutes the background and the values obtained outside that range is considered to be intrusion. Here subscript *'x'* represents the colors Red*(r)*, Green*(g)* or Blue*(b)*

Now we take each red (R_v), green (G_v) and blue (B_v) component of the observed value V of each pixel, which can generically be represented as X_v, where X represents R, G or B and apply following equations on it to detect the intrusion. 'σ_x' is the variance and expected values are R_e, G_e and B_e of the respective RGB components which can be represented as X_e , where X represents R, G or B.

The RGB values of every pixel of the captured frames are now taken and compared with the expected values as given before. Here, k is chosen to be 0.73 based upon the empirical tests and is used for thresholding. After the detection of intrusions a binary image is created with 1s at pixels where intrusion has occurred and 0s elsewhere. The decision equations are written below:

$\boldsymbol{X_e \square (k *\sigma_x) < X_v < X_e} + \boldsymbol{(k *\sigma_x)}$; Then it is Background (11)

Else it is intrusion

The flowchart of the statistical approach to find intrusion is shown in Figure 3

4.2. Compensations

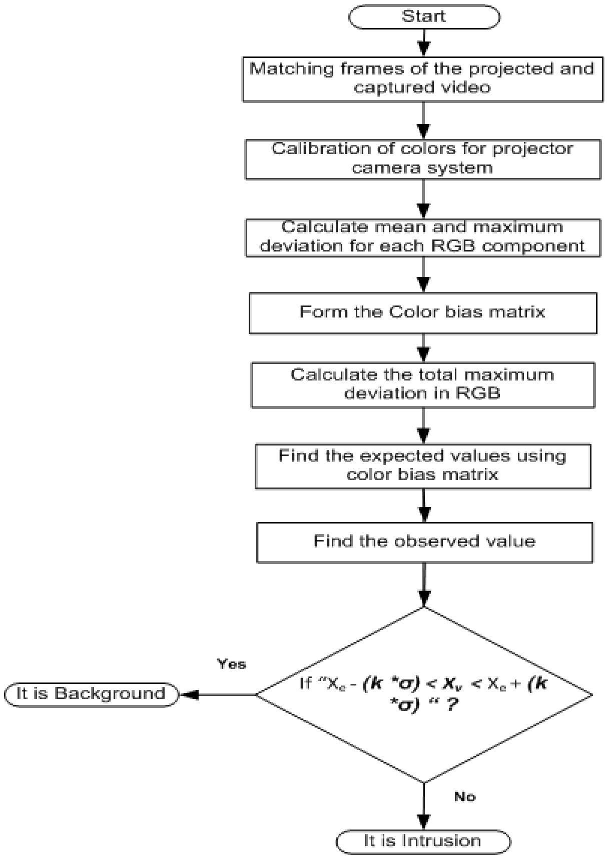
4.2.1. Correction for Camera Auto Gain and Auto Colour Balance

The approach assumes that the camera does not implement automatic gain control and automatic white balance. If this is not the case additional measures are required for proper detection. This subsection deals with those additional measures. When a camera implements automatic gain control, and the feature is not optional (cannot be turned off), changes in the content of the projected scene *D[n]* will result in global luminance adjustments in the camera output, affecting, essentially, every pixel. By our decision rule, this will affect the results of the detector, which it should not. We now outline briefly a method to undo this effect, by remapping the luminance function suitably to defeat the auto gain effect. The equations that follow invariably contain some empirical constants that could change with the camera in use.

The method estimates the illumination conditions of the observed image and normalizes the brightness before carrying out background subtraction. The first step towards this is a color space transformation to transform the image into YC_bC_r colour space. Our subsequent interest is confined to the Y component.

The RGB color space does not provide sufficient information about the illumination conditions and effect of such conditions on any surface. So we transform to YC_bC_r space and then apply threshold to Y component to enhance the segmentation by using the intensity properties of the image. Threshold segmentation was implemented as the first step to decrease the details in the image set greatly for efficient processing. Hence we calculate luminance at each pixel and then calculate

Figure 3. The statistical approach to detect intrusion


Start
Matching frames of the projected and captured video
Calibration of colors for projector camera system
Calculate mean and maximum deviation for each RGB component
Form the Color bias matrix
Calculate the total maximum deviation in RGB
Find the expected values using color bias matrix
Find the observed value
If "X_e - $(k *\sigma) < X_v < X_e + (k *\sigma)$ " ?
Yes
It is Background
No
It is Intrusion

the new value for 'k' the deflection coefficient at each pixel according to the value of luminance.

This is done by developing a linear relationship between luminance and *'k'*

$$k^y -.82= (slope*(Y - Y_{min}) \quad (12)$$

$$k^y = (slope*Y)+(.82)-(slope*Y_{min}) \quad (13)$$

where, k^y - The factor by which the old value of *'k'* must be multiplied

$$slope = (0.06/(Y_{max} \square Y_{min})); \quad (14)$$

L_{min}, L_{max} -Minimum and Maximum Luminance for pixels in the frame respectively.

4.2.2. Dominant Color Compensation

This compensates for possible inbuilt white balance adaptation by the camera. Automatic white balance adaptation in the camera tends to suppress the dominate color. We therefore artificially increase sensitivity to the dominant color to compensate for the adaptation. The value of *'k'* is set as follows:

$$k^c =[(R+G+B) \div (3 * Dom\_color)]+0.9 \quad (15)$$

where, *Dom_color* is the dominant color for that particular pixel (either R or G or B) and *'k^c'* is a new constant for modulating *'k'*. Hence, the final value of constant *'k'* is given by:

$$k_{final} = k^* \, k^y / k^c \quad (16)$$

After this dominant color and luminance compensation, we replace *k* with k_{final} in the detection Equation.(11).

4.3. Intrusion Detection using Reflectance Model

The methods outlined in this subsection were actually developed and implemented at our lab. It is more general and all encompassing formulation of the problem; it introduces and uses the method of reflectance modeling. It gives additional advantages to the user by allowing use of non white and textured surfaces for projection which was not permissible in the Gaussian model approach. What follows in this subsection constitutes the main and most significant innovative part of our work.

Reflectance modeling represents the more refined approach to the problem of intrusion detection in highly varying and dynamic illumination in the presence of near-constant non-dominant ambient illumination. We now launch into a discussion of this method in a systematic manner. The main aim of the problem was detection of events that differ from what is considered normal. The normal in this case, is, arguably, the possibly highly dynamic scene projected on the user specified surface by the computer through the mini projector. We aim to detect the intrusion through a novel process of reflectance modeling. The session begins with a few seconds of calibration which itself includes generating models of the hand, the surface, and the ambient illumination. Subsequently, we proceed to detect the hand in constantly changing background caused by the mixture of unchanging ambient illumination and the highly varying projector illumination under front projection. This kind of detection requires carefully recording the camera output with certain constraints followed by the learning phase and projector-camera co-calibration to match the no of frames per second and number of pixels per frame. This is executed with the steps explained below:

4.3.1. Calculation of Expected RGB Values and Detecting Intrusion at Initial Stages under Controlled Projector Illumination

1. Recording and modeling surface under ambient lighting (ambient lighting is on and projector is off). This defines a model say S_A, which is surface under ambient lighting and is true for any sort of arbitrary texture plane surface.
2. Now, the hand is introduced on the surface illuminated by the ambient lighting and a model for hand is obtained, say H_A, which is hand/skin under ambient light. This is done through the following steps: first the region occupied by the hand is segmented by subtraction and a common Gaussian model for all the sample pixels of the hand available over the space of the foreground and over all the frames of the exposure.
3. Hand is removed from the visibility of camera and the projector is switched on with three lights one by one, Red, Green and Blue. This is followed by observing and modeling of the surface in ambient light in addition to the colored light of projector, which can be represented by S_{AP}^R, S_{AP}^G and S_{AP}^B respectively. It is assumed that we cannot switch off the ambient light as we wish. Each $S_{AP}^Y = [\, R_S^Y, G_S^Y, B_S^Y \,]$, where *Y* represents the projection of lighting and may take values as R, G or B
4. Now the surface in colored(R, G, B) projector light (S_P^Y) is determined by differencing S_{AP}^Y and S_A at each pixel. The relationship of the session parameters is as shown in Equation (18). This specifies the green, red and blue component of the surface under projection. The subtraction should be done component wise i.e., for each red, green and blue color

$$S_P^Y = [R_P^Y, G_P^Y, B_P^Y]^T \qquad (17)$$

$$S_P^Y = S_{AP}^Y - S_A; \qquad (18)$$

5. The hand is introduced under a scenario when the ambient light is on, and the projector is displaying three lights one by one, Red, Green and Blue. We get new models of hand which are H_{AP}^R, H_{AP}^G and H_{AP}^B for red, green and blue light respectively captured under combination of ambient light and projector white light. Each $H_{AP}^Y = [\ R_H^Y,\ G_H^Y,\ B_H^Y\ ],$ where Y represents the projection of lighting and may take values as R, G or B
6. Hence, the model of the hand in projected white light is obtained, H_P^Y which is obtained in the same way as S_P^Y.

$$H_P^Y = [R_{PH}^Y, G_{PH}^Y, B_{PH}^Y]^{T\,(19)}$$

$$H_P^Y = H_{AP}^Y - H_A; \qquad (20)$$

7. Color bias matrix is constructed for both models of hand(M_H) and surface(M_S) like we construct in the Gaussian model
8. Now project the known changing data on the surface under observation by camera. Let us assume the data is *D[n]* where *'n'* is the frame number. But the camera receives a sum of the reflections of the ambient lighting from the surface.
9. Normalization of the models H_P^Y and S_P^Y is done to obtain values which are less than or equal to one by dividing each component by 255, which is the maximum value that each component can reach.
10. Now the expected values of the dynamic background when projected on the surface(S_e) is obtained which is ought to be seen through the camera by performing a matrix multiplication of the *D[n]* and the M_S followed by addition of S_A

$$S_e = (M_S \times D[n]) + S_A;\ S_e = [S_e^R, S_e^G, S_e^B]^T \qquad (21)$$

11. Next we calculate the expected values of the hand pixels when dynamic background is projected on the hand (H_e) which is ought to be seen through the camera by performing matrix multiplication of the *D[n]* and the M_H followed by addition of the H_A image

$$H_e = (M_H \times D[n]) + H_A\ ;\ H_e = [H_e^R, H_e^G, H_e^B]^T \quad (22)$$

The average result of the net outcome of the above calculation and the Gaussian model method is the values expected in the region of the hand skin pixels during intrusion in the combination of ambient lighting and foreground projection on the hand. Now these values can be used to detect the blobs for the fingers of the hand entering the frames by detecting skin regions manipulated by the models obtained earlier.

12. Now consider the *'nth'* of the time and space normalized camera output. The value of the observation at any given pixel be *'V'* while H_e and S_e are the expected values for the same pixel in that frame. Evaluate the ratio:

$$\eta = \frac{\left\| V - H_e \right\|}{\left\| V - S_e \right\|} \qquad (23)$$

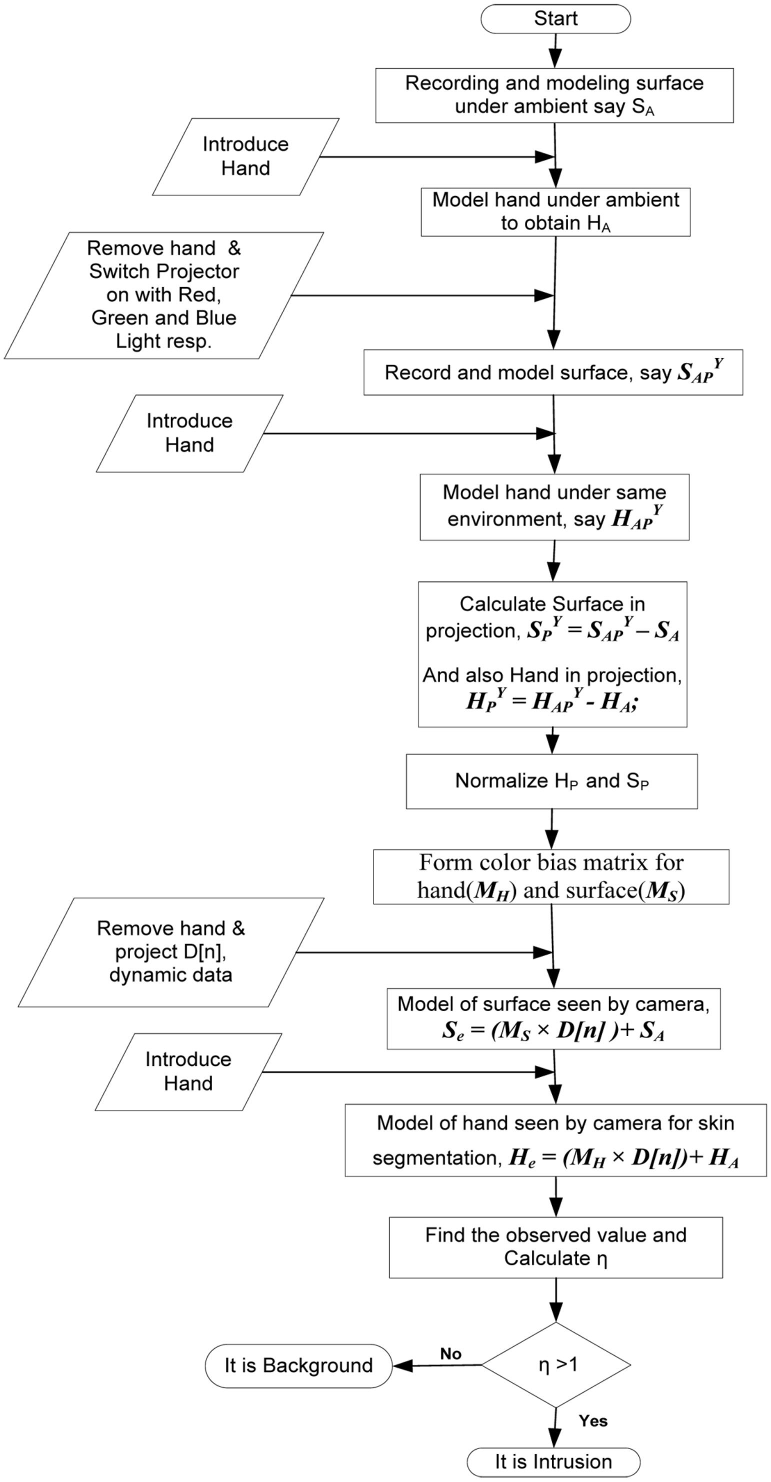
Decision rule is as follows: If η > 1 then it is intrusion, else it is background. The majority decision of all the three RGB components is taken as final decision.

The flowchart of the use of reflectance modeling method to detect intrusion is shown in Figure 4.

4.3.2. Shadow Removal and other Post Processing

Shadows are often a problem in background subtraction because they can show up as a foreground object under segmentation by change detection. A shadow may be viewed as a geometrically distorted version of the pattern that together with

Figure 4. Flowchart representation of reflectance modelling method


Start
Recording and modeling surface under ambient say S_A
Introduce Hand
Model hand under ambient to obtain H_A
Remove hand & Switch Projector on with Red, Green and Blue Light resp.
Record and model surface, say S_{AP}^{Y}
Introduce Hand
Model hand under same environment, say H_{AP}^{Y}
Calculate Surface in projection, $S_P^Y = S_{AP}^Y - S_A$
And also Hand in projection, $H_P^Y = H_{AP}^Y - H_A;$
Normalize H_P and S_P
Form color bias matrix for hand(M_H) and surface(M_S)
Remove hand & project D[n], dynamic data
Model of surface seen by camera, $S_e = (M_S \times D[n]) + S_A$
Introduce Hand
Model of hand seen by camera for skin segmentation, $H_e = (M_H \times D[n]) + H_A$
Find the observed value and Calculate η
η >1
No
It is Background
Yes
It is Intrusion

the pattern produces an overall distorted figure. Shadows can greatly hinder the performance level of pattern detection and classification systems.

There are a number of possible methods for the detection and removal of image shadows. In our method we employ the concept that the point where shadows are cast has the same ratio between the RGB components expected in the absence of intrusion to those observed in its presence. Hence the red, green and blue component ratios are calculated at each point in the area where intrusion is detected and this ratio is used to determine shadow regions where these ratios is consistent across R, G, B. After removing the shadow, Noise removal algorithm is applied on the image to remove both salt and pepper and Gaussian noise using a 4×4 median filter and Gaussian filter respectively. This is then followed by application of connected component analysis by performing foreground cleanup in a raw segmented image. This form of analysis returns the required contour of hand removing the other disturbances and extra contours.

5. GESTURE DETECTION

In this section, we present the essential gesture detection and quantification methods to build a complete gesture based visual interface. While all the techniques outlined below were most certainly independently developed in our lab literally from first principles, we ourselves acknowledge that despite being original, this part of our work applies relatively straightforward and well known elementary image operations and cannot be said to constitute major innovation.

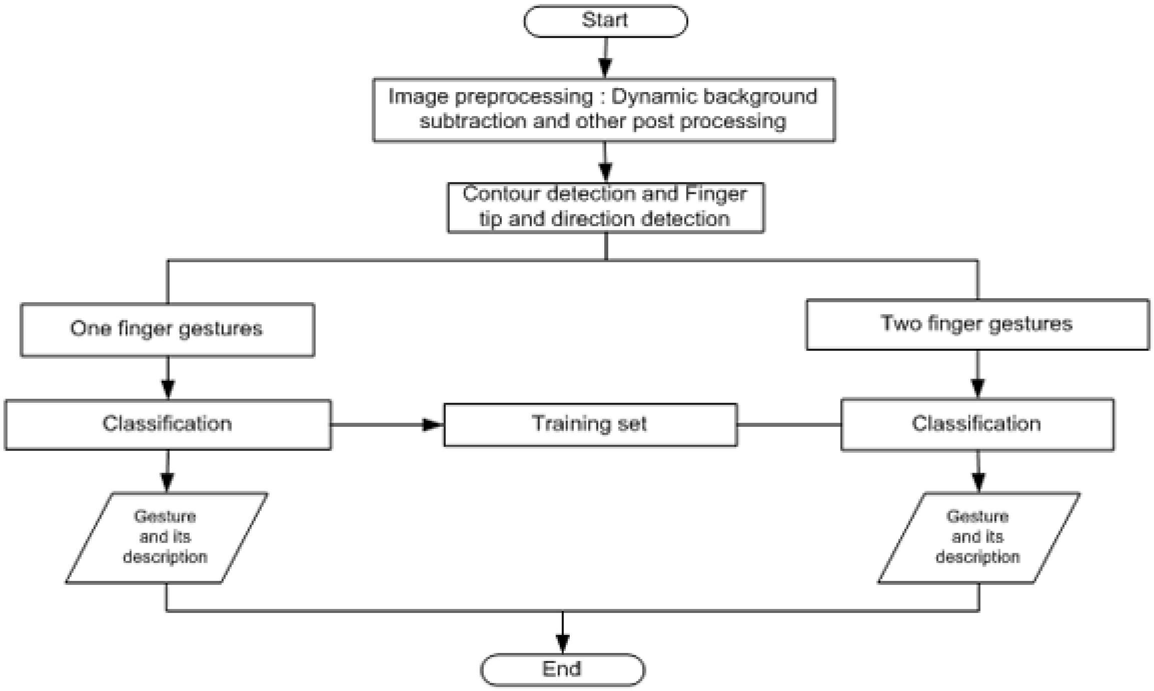
After detection of the binary images by techniques outlined in the previous sections, we need to detect the finger tips and the type and attributes of the gestures. The aim of this project is to propose a video based approach to recognize gestures (one or more fingers). The algorithm includes the following steps and is shown in Figure 5.

1. Contour detection of hand which is represented by a chain sequence in which every entry in the sequence encodes information about the location of the next point on the curve
2. **Curvature mapping:** The curvature of a smooth curve is defined as the curvature of its osculating circle at each point. Curvature may either be negative or positive. Calculation of curvature at each point in the contour by applying the usual formula, along with detection of corner points by computing second derivatives using Sobel operators and finding eigenvalues from the autocorrelation function obtained. Using the first method of curvature, we apply the usual formula for signed curvature *k*:

$$k = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}} \tag{24}$$

   where $x^{'}$ and $y^{'}$ gives the first derivative in horizontal and vertical direction. $y^{''}$ and $x^{''}$ are the second derivatives in the horizontal and vertical direction
3. **Positive curvature extrema extraction on contour** (determining the highest positive corner points) **This is done by two methods:** One method finds out the maximum positive peaks of the signed curvature calculated in the step above and other method finds the corner points by computing second derivatives. In case of more than one positive curvature points of almost equivalent magnitude of curvature, we classify the gesture to be multiple fingers. The two methods are applied jointly upon each frame, because it was found that corner detection alone produced many false positives.
4. **Segregating the gesture into single or multiple finger:** Single finger gestures: Click, Rotate (Clockwise and Anticlockwise), Move arbitrary and Pan Multiple finger

Figure 5. Flowchart representation of gesture detection from intrusion


Start
Image preprocessing : Dynamic background subtraction and other post processing
Contour detection and Finger tip and direction detection
One finger gestures
Two finger gestures
Classification
Training set
Classification
Gesture and its description
Gesture and its description
End

gestures: Zoom (Zoom-In and Zoom-Out), Drag.

5. **Frame to frame fingertip tracking by using motion model estimation:** The trajectory, direction evolution, starts and end points of each finger in the gesture performed is traced through the frames. This is done by applying motion model upon the high curvature point in every frame on the retrieved contour and verifying if the detected point lies in the vicinity of the prediction made using the preceding frames. Tracking motion feedback is used to handle momentary errors or occlusions.
6. **Gesture classification and quantification:** The final classification and subsequent gesture quantification is performed using the information represented diagrammatically in Figure 6.

The gestures shown in Figure 6 are described as follows:

Single finger gestures:

Click: When there is no significant movement in the finger tip except for a vibration.

Pan: When the comparative thickness of the contour is above some threshold.

Move: When there is significant movement in the finger tip in any direction.

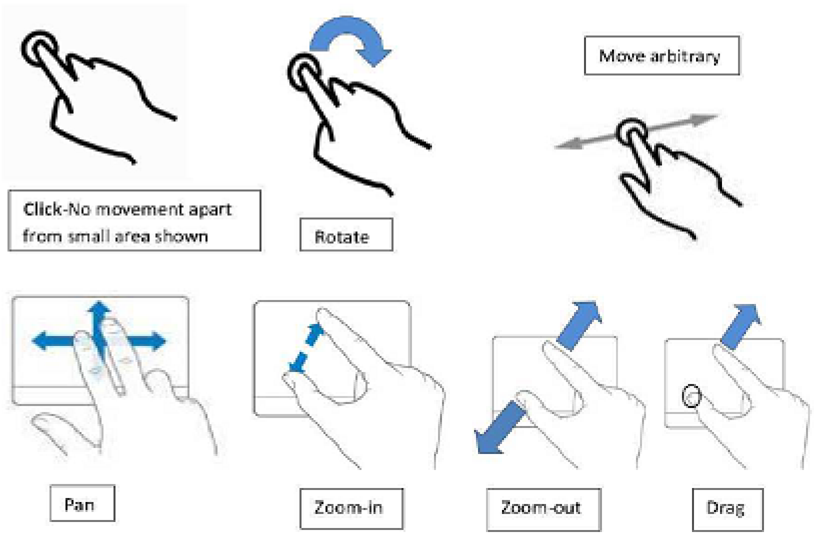
Rotate: For this slope is calculated at each point along the trajectory and the and following equations are implemented: Let at time '*t'* the coordinates of finger tip are *(x, y)* and at some time *'t + k'* the fingertip coordinates are *(x',y')*

$$a = \frac{y' - y}{x' - x}.\ b = \frac{x' - x}{y' - y} \tag{29}$$

where 'a' and 'b' represents the slope and inverse slope respectively

When the gesture ends, we find out how many times both '*a'* and '*b'* becomes zero and what is

Figure 6. Gesture classification criteria



their sum. The times *'a'* becomes zero represents that the line is horizontal and the times *'b'* becomes zero represents that the line is vertical line. The presence of line represents absence of curve and thereby helps us to find out whether our gesture is rotate or not.

Two finger gestures:

Drag: When one of the finger tip stays constant and other finger tip moves.

Zoom out: When the Euclidean distance between the two finger tips decrease gradually.

Zoom-in: When the Euclidean distance between two finger tips increase gradually

Tables 1 and 2 describe each gesture that is how it is performed and what it represents.

6. RESULTS AND CONCLUSION

First a clean binary image of the hand is obtained using the method of reflectance modeling, and then gesture detection can be achieved by applying the algorithms explained above. Specifically, the

Table 1. For single finger gestures

No	Gesture	Meaning	Signing Mode
1	Click	It is derived from normal clicking action as we do on mouse of PC's or laptop so as to open something	Tapping index finger on the surface. The position specifies the action location
2	Move Arbitrary	Move in random directions from current position	Move index finger in arbitrary direction on the surface
3	Rotate		Complete or incomplete circle is drawn with index finger in Clockwise and Anti clockwise direction
	(a)Anti-Clockwise	Rotating object in Anti-clockwise direction like taking turn	
	(b)Clockwise	Rotating an object in clockwise direction	
4	Pan	Movement of object or window from one place to another	Index and middle finger stay and move together moving in arbitrary direction

Table 2. For multi-finger gestures

No.	Gesture	Meaning	Signing mode
1	Drag	It signifies movement of window or object in one direction	Enacted by fixed thumb and arbitrary movement of index finger
2	Zoom		
	(a)Zoom-in	Increase in size of window or object	Move index finger and thumb away from each other
	(b)Zoom-out	Decrease in size of window or object	Move index finger and thumb away from each other

system can track the tip positions and motions of the fingers to classify the gestures and find out their attributes. The figure shows the detection of contour of hand and tip of finger(s) in dynamic projection on arbitrary background, followed by tracking the trajectories, velocities and direction of the movement thereby classifying the gestures. These positions depict the commonly held positions of hand, common to all gestures (Figure 7).

By application of our algorithms for both plain and arbitrary backgrounds, we detect the intrusion successfully. This method is accurate and robust and works over a wide range of ambient lighting and varying illumination conditions.

The performance analysis of the reflectance modeling method is as follows:

1. The algorithm was run on:
    a. Three kinds of skin samples- Fair, Dusky and Black.
    b. Three kinds of background surfaces on which projection was made
2. Scale limitation: This represents the area occupied by the hand in the surface area which is being captured by the camera. The minimum value came out be 10% of the screen area approximately and maximum value came out to be 80% of screen area. Outside this range performance is negatively affected
3. Gesture should not contain more than two fingers.
4. 1% error in fingertip detection in the gesture performed i.e., 1 frame missed out of 80 frames approximately in a video
5. Pixel level accuracy of tip is in 10 pixel diameter circle

7. APPLICATIONS

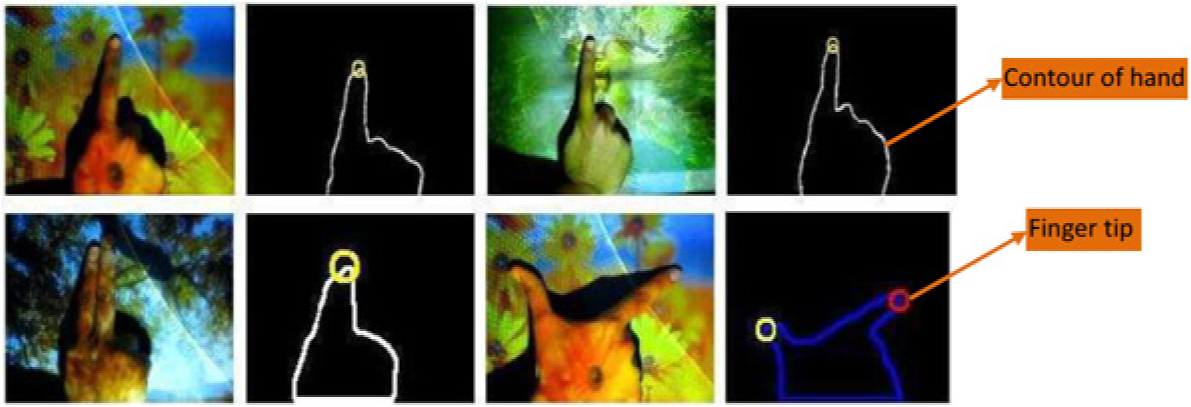
This work finds many applications for new era systems which can act such as both mobile and computers. The best application is in the making of a human computer interface (HCI) where the interfacing devices like keyboard, mouse, calculator, piano etc would become obsolete. It will help in creating a new era system consisting of a projector-camera combined with a processor which can be used as a computing device much smaller than any of the existing systems.

There are several factors that make creating applications in HCI difficult. They can be listed as:

- The information is very complex and variable
- Intrusion detection techniques should be highly robust
- Developers must understand exactly what it is that the end user of their computer system will be doing.

Certain conditions may be relaxed to get attractive applications:

Figure 7. Shows detection of contour and finger tip for single and multiple finger gesture on arbitrary background



- When the front projection is absent i.e., when no dynamic or white light is being projected on to the screen, we can design systems like paper touchpad, virtual keyboard, virtual piano etc. These applications just have a static arbitrary background. Equations given for the dynamic reflectance model simplify accordingly
- Considering a case of back lit projection where dynamic data is being projected at the back allows us to design a system where we can directly interact with the monitor or screen. Here again, the general equations we have given will simplify appropriately. We omit the details here.

Figure 8. Paper touch pad setup on left and the printed touchpad on shhet of paper on right. 1.Paper touchpad, 2. Webcamera just above the paper touchpad and 3. Monitor which is mapped to the touchpad

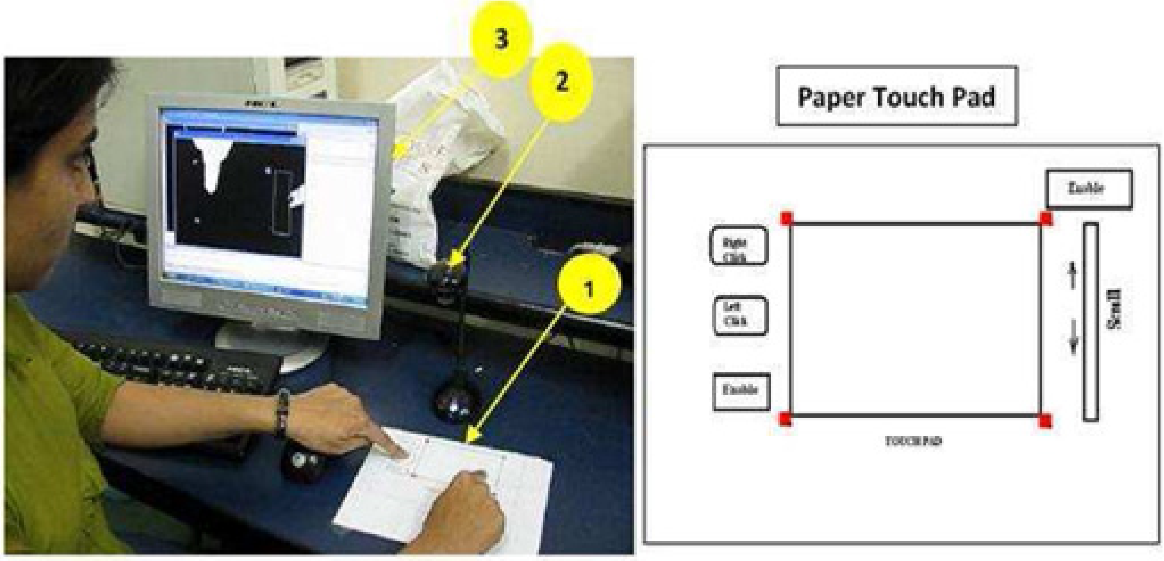
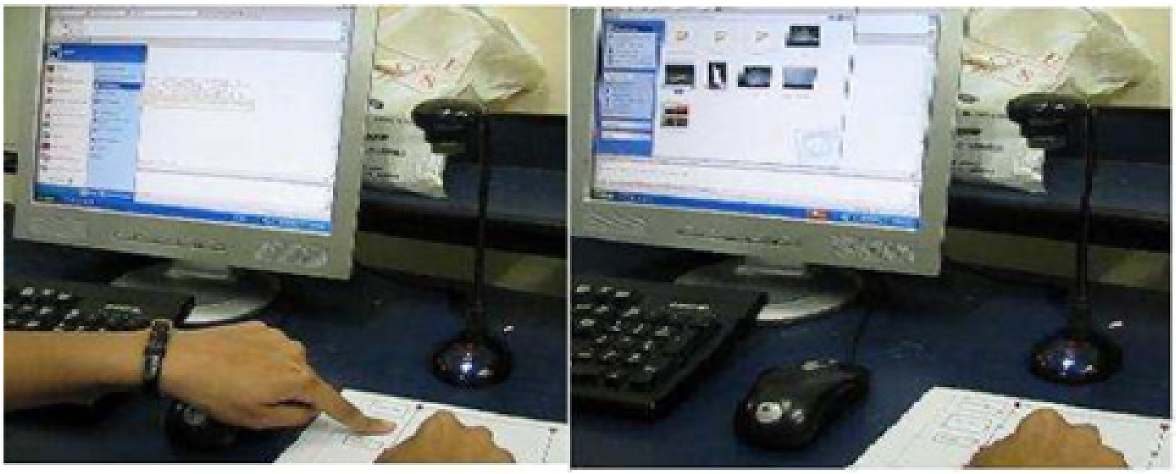


Figure 9. Paper touch pad showing operation of left click



One of the key applications is Paper Touch-pad. The paper touchpad is a kind of a virtual mouse used for providing mouse cursor and its functions in any computer system using an ordinary sheet of paper with a few fixed markings on it for calibration. The setup and the layout of paper touch pad is shown in Figure 8. The red dots on the corner of the printout of the touchpad are used for homographic mapping. The figures show the movement of the cursor and left click operation of the mouse. In Figure 9, the first picture shows left click operation on ‘My Pictures’ icon in start menu and the picture besides it shows the window of My Pictures opened on the display screen as a result. Along similar lines, we can design application specific keyboards/keypads, and use our techniques to enable ‘paper keyboards.’

9. FUTURE POSSIBILITIES

1. This may be further extended for whole body gestures which may be used for sign language recognition or for robotic and other applications
2. We may also use an infra-red laser or flood illumination as an invisible 4th channel for detecting more details of gestures performed and to further eliminate the effects of the visible band varying illumination.
3. Extract more information like speed and acceleration from the gesture performed and allowing the user to communicate through these parameters as well.
4. As the end result, we aim to design a robust real time system which can be embedded into a mobile device that can be used without accessories anywhere a flat surface and some shade is available. The single unit would substitute for the computer/communicator, the display, keyboard and pointing device which may require a projector, camera, processor and memory.
5. We can move on to develop vision techniques to recognize gesture sequences, instead of just individual gestures as well as more complicated finger gestures, which can be a great help in faster communication.

REFERENCES

Eldridge, R., & Rudolph, H. (2008). *Stereo vision for unrestricted human computer interaction*. Rijeka, Croatia: InTech.

Fukumoto, M., Suenaga, Y., & Mase, K. (1994). Finger-pointer: Pointing interface by image processing. *Computers & Graphics*, *18*(5), 633–642. doi:10.1016/0097-8493(94)90157-0

Gordon, G., Darrell, T., Harville, M., & Woodfill, J. (1999). Background estimation and removal based on range and color. In *Proceedings of the International IEEE Conference on Computer Vision and Pattern Recognition.*

Grossman, T., Balakrishnan, R., Kurtenbach, G., Fitzmaurice, G., Khan, A., & Buxton, B. (2001). Interaction techniques for 3D modeling on large displays. In *Proceedings of the Symposium on Interactive 3D Graphics* (p. 1723).

Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18thAnnual ACM Symposium on User Interface Software and Technology.*

Helman, S., Juan, W., Leonel, V., & Aderito, M. (Eds.). (2007, May 23-25). *Proceedings of the GW 7th International Workshop on Gesture in Human-Computer Interaction and Simulation*, Lisbon, Portugal.

Hofer, R., Naeff, D., & Kunz, A. (2009). FLATIR: FTIR multi touch detection on a discrete distributed sensor array. In *Proceedings of the International Conference on Tangible and Embedded Interaction* (pp. 317-322).

Jenabi, M., & Reiterer, H. (2008, October). *Fintteraction: Finger interaction with mobile phones*. Paper presented at the Future Mobile Experiences Workshop, Lund, Sweden.

Jones, M., & Rehg, J. (1999). Statistical color models with application to skin detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Vol. 1).

Kim, S. G., Kim, J. W., & Lee, C. W. (2007). Implementation of multi-touch tabletop display for HCI (human computer interaction). In *Proceedings of the 12th International Conference on Human-Computer Interaction: Interaction Platforms and Techniques* (pp. 854-863).

Lin, H. H., & Chang, T. W. (2007). A camera based multitouch interface builder for designers. In J. A. Jacko (Ed.), *Proceedings of the 12th International Conference on Human-Computer Interaction: Applications and Services* (LNCS 4553, pp. 1102-1109).

Lockton, R. (2009). *Hand gesture recognition using special glove and wrist*. Oxford, UK: Oxford University.

Mistry, P., Maes, P., & Chang, L. (2007). WUW - Wear Ur World - A wearable gestural interface. In *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems* (pp. 4111-4116).

Mitra, S., & Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, *37*(3), 311–324. doi:10.1109/TSMCC.2007.893280

Motonorihi, S., Ueda, S., & Akiyama, K. (2003). Human interface based on finger gesture recognition using omni-directional image sensor. In *Proceedings of the IEEE International Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems* (pp. 68-72).

Nadia, M., & Cooperstock, J. (2004). Occlusion detection for front projected interactive displays. In *Proceedings of Pervasive Computing and Advances in Pervasive Computing*.

Oka, K., Sato, Y., & Koike, H. (2002). Real-time fingertip tracking and gesture recognition tracking. *IEEE Computer Graphics and Applications*, *22*(6), 64–71. doi:10.1109/MCG.2002.1046630

Pavlovic, V., Sharma, R., & Huang, T. (2001). Visual interpretation of hand gestures for HCI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7), 677–695. doi:10.1109/34.598226

Segan, J., & Kumar, S. (1999). Shadow gestures: 3D hand pose estimation using a single camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 479-485).

Shimada, N., Shirai, Y., Kuno, Y., & Miura, J. (1998) Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. In *Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 268-273).

Thomas, M. (1994). *Finger Mouse: A freehand computer pointing interface* (Unpublished doctoral dissertation). The University of Illinois, Chicago, IL.

Utsumi, A., & Ohya, J. (1999). Multiple-hand-gesture tracking using multiple cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 473-478).

Vladimir, I., Rajeev, S., & Thomas, S. (1993). *Visual interpretation of hand gestures for HCI: A review*. Chicago, IL: The University of Illinois.

Westerman, W., Elias, J. G., & Hedge, A. (2001). A multi touch: A new tactile 2-D gesture interface for HCI. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 45, pp. 632-636).

Wren, C., Azarbayejani, A., Darrell, T., & Pentland, A. P. (1997). Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7), 780–785. doi:10.1109/34.598236

Wu, A., Shah, M., & Lobo, N. (2000). A virtual 3D blackboard: 3D finger tracking using single camera. In *Proceedings of the Fourth International IEEE Conference on Automatic Face and Gesture Recognition*, Grenoble, France (pp. 536-543).

Wu, Y., Lin, J. Y., & Huang, T. S. (2001). Capturing natural hand articulation. In *Proceedings of the IEEE International Conference on Computer Vision* (Vol. 2, pp. 426-432).

Xing, J., Wang, W., Zhao, W., & Huang, J. (2009). A novel multi-touch human-computer-interface based on binocular stereo vision. In *Proceedings of the International Symposium on Intelligent Ubiquitous Computing and Education* (pp. 319-323).

Yuan, W., & Zhang, W. (2010). A novel hand-gesture recognition method based on finger state projection for control of robotic hands. In H. Liu, H. Ding, Z. Xiong, & X. Zhu (Eds.), *Proceedings of the Third International Conference on Intelligent Robotics and Applications* (LNCS 6425, pp. 671-682).