Image Processing Based Touch-Sensing System

Synopsis of Project report submitted to
Indian Institute of Technology Kharagpur
for the award of the degree

of

Bachelor of Technology
in Instrumentation Engineering

by

Tushar Kant Deo

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR
MAY 2018

DECLARATION

I certify that

a. the work contained in this Report is original and has been done by me under the guidance of my supervisor(s).
b. the work has not been submitted to any other Institute for any degree or diploma.
c. I have followed the guidelines provided by the Institute in preparing the Report.
d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
e. whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the Report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Signature of the Student

CERTIFICATE

This is to certify that the Report entitled **Image processing based Touch sensing system**, submitted by **Tushar Kant Deo** to Indian Institute of Technology Kharagpur, is a record of bona fide research work under my (our) supervision and is worthy of consideration for the award of the degree of Bachelor of Technology of the Institute.

_____

Superviser

Date:

CONTENTS

Chapter 1 Introduction

In recent years, touch screens have been widely used in many commercial devices such as mobile phones, tablets, game machines and control panels. From an interaction point of view, touch screens have two main advantages: one is that they enable the user to interact directly with what is displayed rather than indirectly doing it with a pointer, which significantly improves the users' experience; the other is that they allow one to do so without requiring any intermediate devices that would need to be held in the hand. Given that, touch screens are used in situations where the keyboard and the mouse systems do not offer a suitably intuitive or rapid interaction. The only drawback of the currently commercialized touch screens is their high cost which is due to the usage of costly technologies (resistive or capacitive) .

A variety of touch screen technologies have been utilized to sense touch, and commonly used touch screens include resistive, capacitive, infrared-based, and webcambased touch screens. Compared with other ones, webcambased touch screen is less accurate, but the loss in accuracy is highly compensated by the low cost of the computer vision technologies used, which provides a big advantage over conventional touch screens.

1.1 Overview

Main focus of this work is to achieve human computer interaction without using any extra hardware and with the help of computer vision.To achieve this I have worked on two aspects ,First one is to convert any screen into touch screen using single camera and Second work is to control the mouse from distance using hand motion.

In this paper, I present an inexpensive webcam-based touch screen system based on a simple and generic touch detection technique. Only a single webcam and a regular screen are used to achieve this goal. This work has focused on the finger touch position detection methods for Image processing-based touch sensing systems. By using the scene geometry and the camera model, we can determine the position of the finger on the screen.Image processing-based touch sensing system is used to large touch screen for TV weather forecast, Digital Signage. Image processing-based touch sensing systems has

advantages: (1) the surface or bottom sides of touch screen need not be instrumented like resistive membrane-based systems and capacitive-based sensing systems. (2) this approach can convert non-touch screen into touch screen. (3) this approach enables various operations not only by touch but also by gesture operations.

In second part I present an robust algorithm to detect and track a marker which can be used to control mouse from distance.This can give us a new way to interact with computer even when we cannot reach to it.It can be helpful especially in works which involve operating a computer from distance like presentation.

1.2 Related Work

A variety of approaches have been developed to realize touch screens. A resistive touchscreen panel consists of several layers, and among those layers, the two most important ones are thin, transparent electrically-resistive layers separated by a thin space. When an object or a finger presses down on the outer surface, the two layers touch and is connected at that point, and the touch position will be calculated from the obtained data . A capacitive touch screen panel consists of an insulator such as glass, coated with a transparent conductor. Since human body is an electrical conductor, touching the surface of the screen will lead to a distortion of the screen's electrostatic field, measurable as a change in capacitance . An infrared touchscreen uses an array of X-Y infrared LED and photodetector pairs laid around the edges of the screen to detect a disruption in the pattern of LED beams. Xu illustrates one kind of systems for detecting touch events by an optical touch screen where transmitters and receivers are positioned according to an alternating scheme on each side of the touchscreen. Many webcam-based approaches have been proposed to overcome the high cost of the existing touch screen technologies. The principle of these methods is to detect the hand/finger in the video feed during the interaction and estimate its position when touching a regular screen, and as a result simulate a touch screen.

Chapter 2 Theoretical studies

2.1 Hardware Setup

The Physical setup of the system is represented in Fig. 1.

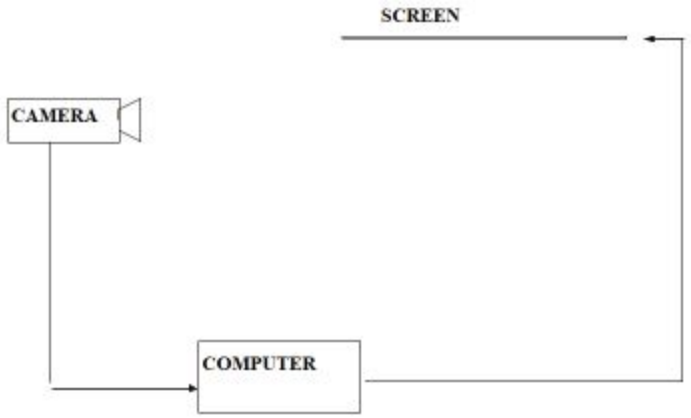


Fig. 1. System Setup

The setup is simple and involves placing of the Camera in an appropriate position, so that the whole of the screen lies within the image frame. The camera can be placed at any angle, provided the screen is fully visible. This is followed by a user friendly calibration technique and at the end of the calibration the setup is ready for the user to interact with the screen.

A. Camera

A camera is used as the image source for the proposed algorithm. This is used to capture the position of finger.Cameras' CCD sensors can detect the rays and produces a bright spot in the image.

B.Webcam Position

In our proposed method, we require a single webcam. In order to calculate the touch point between the finger and the monitor, we need to detect the finger from the image captured by the webcam. To make this complex task easier, we opt to position our webcam in a way to have the monitor's complete view when placed aside of monitor.

2.2 Inverse Perspective Transformation

The camera captures the 3D image and is translated into a 2D image plane. During this process, the image undergoes a distortion called perspective distortion. As a result, the real world coordinates (Ri) pointed by the finger becomes a nonlinear function of the coordinates (Pi) obtained from the raw image captured by the camera. In order to determine the real world coordinates, this distortion must be compensated. This is achieved by performing inverse perspective transformation on the raw image. Mapping the points on the distorted image to a 2D rectified image, establishes a linear relationship with real world coordinates and rectified image coordinates .

Fingertip transformation algorithm

For perspective transformation, we need a 3x3 transformation matrix. Straight lines will remain straight even after the perspective transformation. To find this transformation matrix, we need 4 points on the input image and corresponding points on the output image. For our application, the four output points are the four corner points of the screen frame coordinate and the four input points are provided by the user. Among these 4 points, 3 of them should not be collinear. Then transformation matrix can be found by the function **cv2.getPerspectiveTransform**.The function calculates the 3×3 matrix of a perspective transform so that:

$$\begin{bmatrix} t_i x'_i \\ t_i y'_i \\ t_i \end{bmatrix} = \texttt{map\_matrix} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Where

$$\texttt{dst}(i) = (x'_i, y'_i), \texttt{src}(i) = (x_i, y_i), i = 0, 1, 2, 3$$

Then apply **cv2.warpPerspective** with this 3x3 transformation matrix. The function warpPerspective transforms the source image using the specified matrix:

$$\texttt{dst}(x, y) = \texttt{src}\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right)$$

Color space transformation and Filtering

The rectified image from the perspective calibration consists of only 2D rectified image. Hence further processing is carried out on the rectified image, by extracting and transforming the RGB color space into single channel Grayscale and HSV scale. This grayscale and HSV channel image is smoothed using Gaussian filter having a 3x3 window.

2.3 Video tracking

The objective of video tracking is to associate target objects in consecutive video frames. The association can be especially difficult when the objects are moving fast relative to the frame rate. Another situation that increases the complexity of the problem is when the tracked object changes orientation over time. For these situations video tracking systems usually employ a motion model which describes how the image of the target might change for different possible motions of the object.

Examples of simple motion models are:

- When tracking planar objects, the motion model is a 2D transformation (affine transformation or homography) of an image of the object (e.g. the initial frame).
- When the target is a rigid 3D object, the motion model defines its aspect depending on its 3D position and orientation.

To perform video tracking an algorithm analyzes sequential video frames and outputs the movement of targets between the frames.

These are major components of a visual tracking system: target representation and localization, as well as filtering and data association.

The following are some common target representation and localization algorithms:

- Kernel-based tracking (mean-shift tracking): an iterative localization procedure based on the maximization of a similarity measure (Bhattacharyya coefficient).
- Contour tracking: detection of object boundary (e.g. active contours or Condensation algorithm). Contour tracking methods iteratively evolve an initial contour initialized from the previous frame to its new position in the current frame. This approach to contour tracking directly evolves the contour by minimizing the contour energy using gradient descent.

The following are some common filtering algorithms:

- Kalman filter: an optimal recursive Bayesian filter for linear functions subjected to Gaussian noise.It is an algorithm that uses a series of measurements observed over time, containing noise (random variations) and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone.
- Particle filter: useful for sampling the underlying state-space distribution of nonlinear and non-Gaussian processes.

Proposed object tracking algorithm:

We have used HSV image and created a mask to extract finger from each frame.Mask is created using dynamic value interactively from user in the initialization step using interactive window

- Take each frame of the video
- Convert from BGR to HSV color-space
- We threshold the HSV image for a range of blue color
- Now extract the blue object alone, we can do whatever on that image we want.

2.4 Feature Detection

Image feature is a simple image pattern, based on which we can describe what we see on the image. For example cat eye will be a feature on a image of a cat. The main role of features in computer vision(and not only) is to transform visual information into the vector space. This give us possibility to perform mathematical operations on them, for example finding similar vector(which lead us to similar image or object on the image)

For example, take below image:

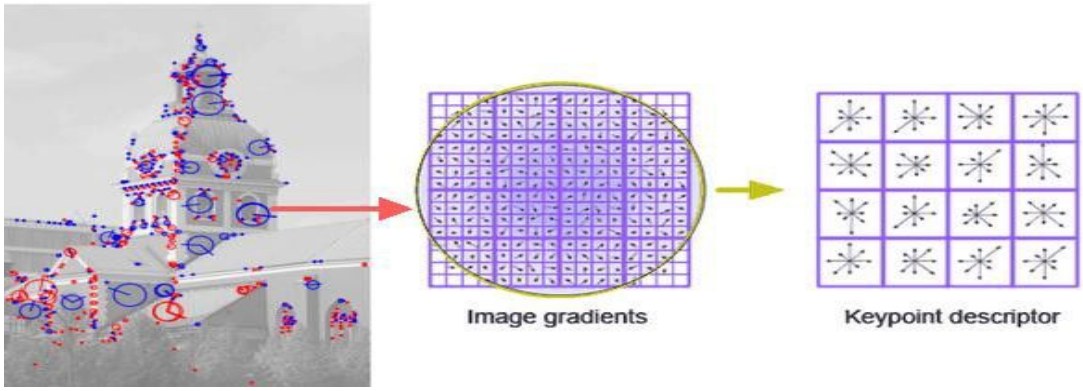
Understanding features:Image is very simple. At the top of image, six small image patches are given. Question for you is to find the exact location of these patches in the original image. How many correct results you can find ?

A and B are flat surfaces, and they are spread in a lot of area. It is difficult to find the exact location of these patches.

C and D are much more simpler. They are edges of the building. You can find an approximate location, but exact location is still difficult. It is because, along the edge, it is same everywhere. Normal to the edge, it is different. So edge is a much better feature compared to flat area, but not good enough (It is good in jigsaw puzzle for comparing continuity of edges).

Finally, E and F are some corners of the building. And they can be easily found out. Because at corners, wherever you move this patch, it will look different. So they can be considered as a good feature. So now we move into more simpler (and widely used image) for better understanding.

There are two ways of getting features from image, first is an image descriptors(white box algorithms), second is a neural nets(black box algorithms).There are many algorithms for feature extraction, most popular of them are SURF, ORB, SIFT, BRIEF. Most of this algorithms based on image gradient.



2.5 Morphological Transformations

Morphological transformations are some simple operations based on the image shape. It is normally performed on binary images. It needs two inputs, one is our original image, second one is called structuring element or kernel which decides the nature of operation. Two basic morphological operators are Erosion and Dilation. Then its variant forms like Opening, Closing, Gradient etc also comes into play. We will see them one-by-one with help of following image:

Erosion

The basic idea of erosion is just like soil erosion only, it erodes away the boundaries of foreground object (Always try to keep foreground in white). So what does it do? The kernel slides through the image (as in 2D convolution). A pixel in the original image (either 1 or 0) will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero).

So what happens is that, all the pixels near boundary will be discarded depending upon the size of kernel. So the thickness or size of the foreground object decreases or simply white region decreases in the image. It is useful for removing small white noises (as we have seen in colorspace chapter), detach two connected objects etc.

2.6 Gaussian Filtering

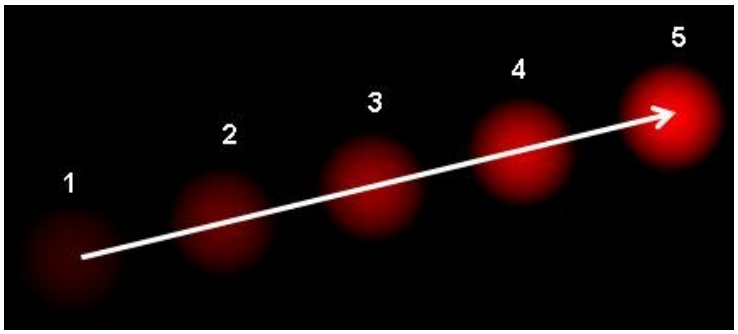
In this approach, instead of a box filter consisting of equal filter coefficients, a Gaussian kernel is used. It is done with the function, cv2.GaussianBlur(). We should specify the width and height of the kernel which should be positive and odd. We also should specify the standard deviation in the X and Y directions, sigmaX and sigmaY respectively. If only sigmaX is specified, sigmaY is taken as equal to sigmaX. If both are given as zeros, they are calculated from the kernel size. Gaussian filtering is highly effective in removing Gaussian noise from the image.

2.7 What are contours?

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

2.8 Optical Flow:

Optical flow is the pattern of apparent motion of objects in a visual scene caused by the relative motion between an observer and a scene. It is the pattern of apparent motion of image objects between two consecutive frames. It is 2D vector field where each vector is a displacement vector showing the movement of points from first frame to second.



Assumptions of optical flow:

1. The pixel intensities of an object do not change between consecutive frames.
2. Neighbouring pixels have similar motion.

Consider a pixel I(x,y,t) in first frame. It moves by distance (dx,dy) in next frame taken after dt time. So since those pixels are the same and intensity does not change, we can say, I(x,y,t)=I(x+dx,y+dy,t+dt)

Then take taylor series approximation of right-hand side, remove common terms and divide by dt to get the following equation:

$$f_x u + f_y v + f_t = 0$$

Where:

$$f_x = \frac{\partial f}{\partial x} \; ; \; f_y = \frac{\partial f}{\partial y}$$

$$u = \frac{dx}{dt} \; ; \; v = \frac{dy}{dt}$$

Above equation is called Optical Flow equation. In it, we can find fx and fy, they are image gradients. Similarly ft is the gradient along time. But (u,v) is unknown. We cannot solve this one equation with two unknown variables. So several methods are provided to solve this problem and one of them is Lucas-Kanade.

Lucas-Kanade method

We have seen an assumption before, that all the neighbouring pixels will have similar motion. Lucas-Kanade method takes a 3x3 patch around the point. So all the 9 points have the same motion. We can find (fx,fy,ft) for these 9 points. So now our problem becomes solving 9 equations with two unknown variables which is over-determined. A better solution is obtained with least square fit method. Below is the final solution which is two equation-two unknown problem and solve to get the solution.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i f_{x_i}{}^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & \sum_i f_{y_i}{}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix}$$

So from user point of view, idea is simple, we give some points to track, we receive the optical flow vectors of those points. But again there are some problems. Until now, we were dealing with small motions. So it fails when there is large motion. So again we go for pyramids. When we go up in the pyramid, small motions are removed and large motions becomes small motions. So applying Lucas-Kanade there, we get optical flow along with the scale.

2.9 Histogram backprojection:

What is backprojection?

Backprojection is a method to find how well pexel of given image fit the model of histogram model. The method is used to calculate the model histogram of feature and then use it to find the feature in an image.

Algorithm:

1. Create a model histogram of feature to track.
2. Load the image.
3. In each pixel of our Test Image (i.e. p(i,j)), collect the data and find the correspondent bin location for that pixel (i.e. (h_{i,j}, s_{i,j})).
4. Lookup the model histogram in the correspondent bin - (h_{i,j}, s_{i,j}) - and read the bin value.
5. Store this bin value in a new image (BackProjection). Also, we may consider to normalize the model histogram first, so the output for the Test Image can be visible to us.

6. In terms of statistics, the values stored in BackProjection represent the probability that a pixel in Test Image belongs to a region of interest.

Histogram Normalization:

The normalization is a process of changing the range of value of pixel intensity.The histogram normalization is also known as histogram stretching. It is also known as dynamics range expansion.The dynamic range expansion is used to bring the image values to the range which is more familiar to the senses.

The linear normalization of a grayscale digital image is performed according to the formula

$$I_N = (I - \text{Min})\frac{\text{newMax} - \text{newMin}}{\text{Max} - \text{Min}} + \text{newMin}$$

Chapter 3: Results and Discussions

Part 1:Touch Detection

PROPOSED METHOD

System organization

1. Capture a single view image includes fingertip and reflected image from the camera.
2. From the capture image, recognize fingertip and reflected image.
3. Get fingertip and reflected image coordinates.
4. From two obtained coordinates, determine overlap of two coordinates.
5. If coordinates of fingertip and reflected image coordinates overlapping, determine finger has touched the screen.
6. After detecting fingertip, convert the coordinate of fingertip on the camera into the coordinate on the screen.
7. Output the result.

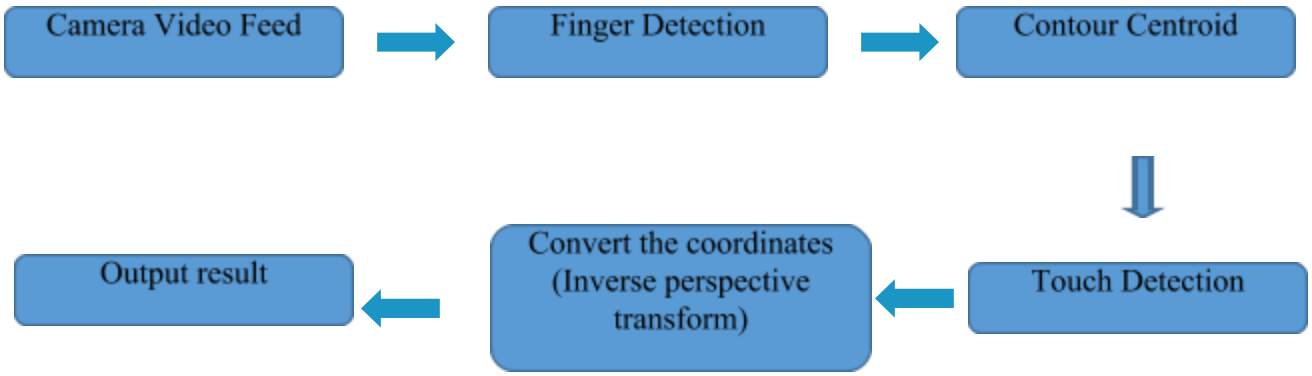


Figure 1: Overall approach of touch sensing algorithm

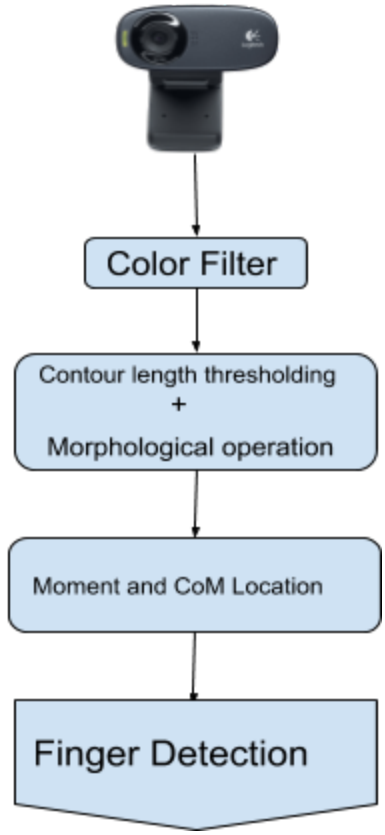


Figure 2: Finger Detection Flow

Touch Detection

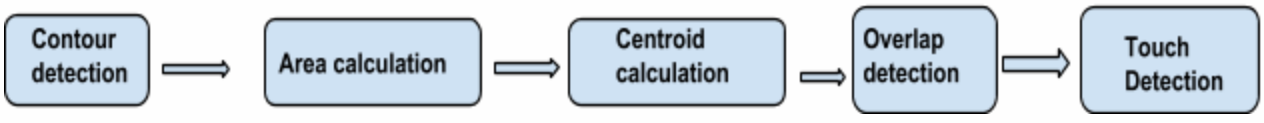


Figure 3 : Touch Detection Flow

Problem With Using Only Image Processing Pipeline

Problem faced:-

1. Algorithms tend to miss frames in between which is not visible through naked Eye because of large FPS of 30.
2. On an average around 30% of frames are missed and no object is detected by image processing timeline.
3. There are false positive on the edge of frames which lead to wrong detection.
4. These misses and false positive makes it impossible to detect finger and touch location.

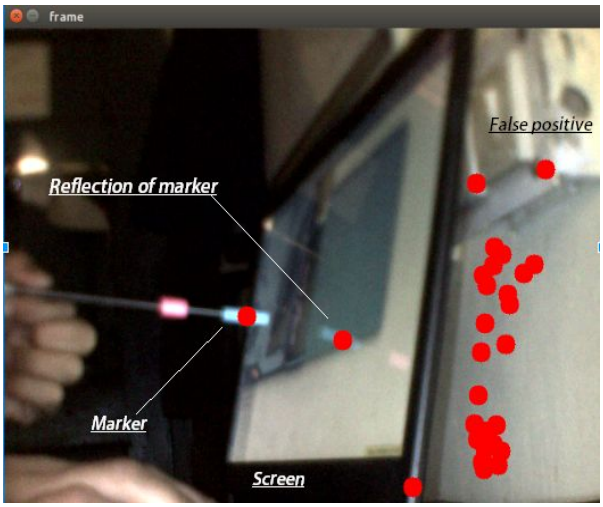


Fig 4: Camera view of normal screen with detection result

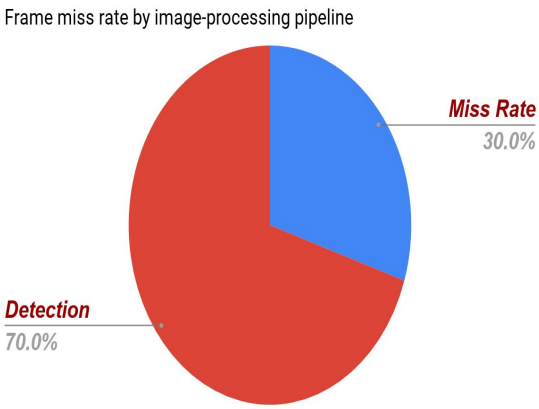


Fig 5: Average miss rate by algorithm

Solution:-

1. I have combined tracking algorithm with image processing pipeline.
2. Overall algorithm is robust of any miss by image processing timeline
3. It also select true positive from noises like false positive
4. It also shows great robustness to occlusion because of its probabilistic nature.

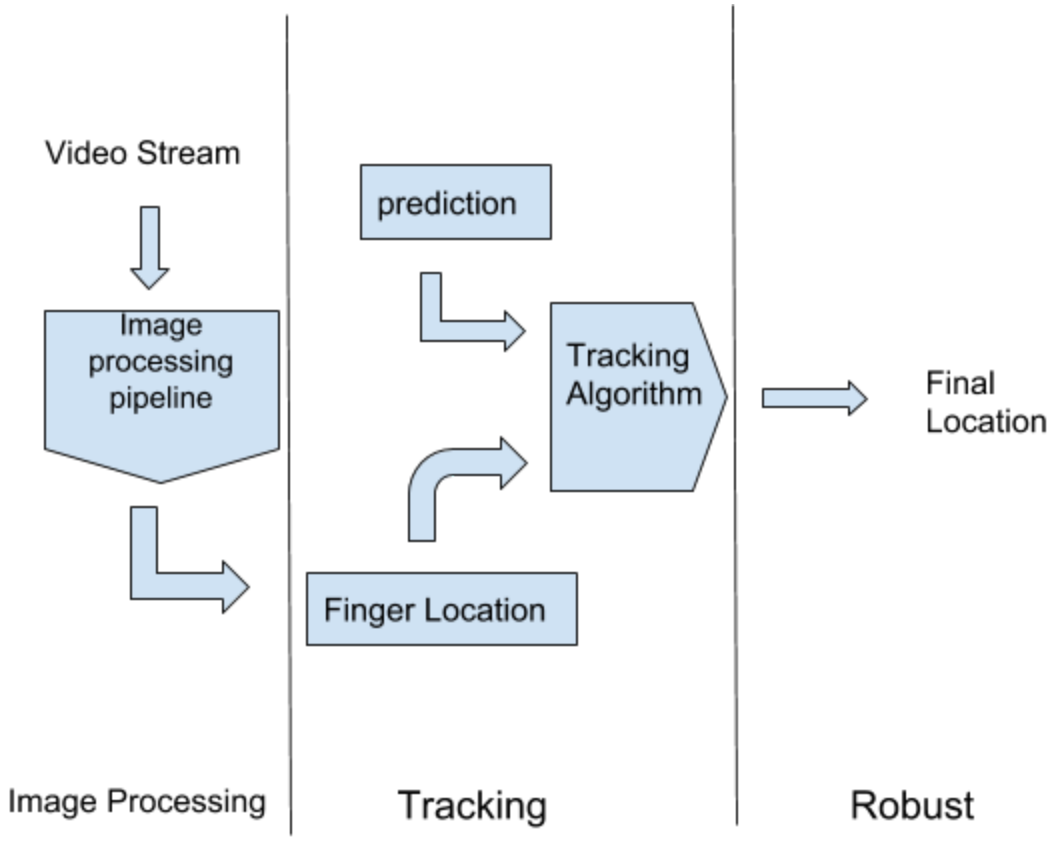


Figure 6 : Tracking algorithm combined with image processing pipeline

General principles behind tracking

The motion model predicts the approximate location of the object. The appearance model fine tunes this estimate to provide a more accurate estimate based on appearance.

What is Object Tracking ?

Simply put, locating an object in successive frames of a video is called tracking.

1. Dense Optical flow: These algorithms help estimate the motion vector of every pixel in a video frame.

2. Sparse optical flow: These algorithms, like the Kanade-Lucas-Tomasi (KLT) feature tracker, track the location of a few feature points in an image.
3. Kalman Filtering: A very popular signal processing algorithm used to predict the location of a moving object based on prior motion information. One of the early applications of this algorithm was missile guidance! Also as mentioned here, "the on-board computer that guided the descent of the Apollo 11 lunar module to the moon had a Kalman filter".
4. Meanshift and Camshift: These are algorithms for locating the maxima of a density function. They are also used for tracking.
5. Single object trackers: In this class of trackers, the first frame is marked using a rectangle to indicate the location of the object we want to track. The object is then tracked in subsequent frames using the tracking algorithm. In most real life applications, these trackers are used in conjunction with an object detector.
6. Multiple object track finding algorithms: In cases when we have a fast object detector, it makes sense to detect multiple objects in each frame and then run a track finding algorithm that identifies which rectangle in one frame corresponds to a rectangle in the next frame.

Tracking vs Detection

1. Tracking is faster than Detection: Usually tracking algorithms are faster than detection algorithms. The reason is simple. When you are tracking an object that was detected in the previous frame, you know a lot about the appearance of the object. You also know the location in the previous frame and the direction and speed of its motion. So in the next frame, you can use all this information to predict the location of the object in the next frame and do a small search around the expected location of the object to accurately locate the object. A good tracking algorithm will use all information it has about the object up to that point while a detection algorithm always starts from scratch. Therefore, while designing an efficient system usually an object detection is run on every nth frame while the

tracking algorithm is employed in the n-1 frames in between. Why don't we simply detect the object in the first frame and track subsequently? It is true that tracking benefits from the extra information it has, but you can also lose track of an object when they go behind an obstacle for an extended period of time or if they move so fast that the tracking algorithm cannot catch up. It is also common for tracking algorithms to accumulate errors and the bounding box tracking the object slowly drifts away from the object it is tracking. To fix these problems with tracking algorithms, a detection algorithm is run every so often. Detection algorithms are trained on a large number of examples of the object. They, therefore, have more knowledge about the general class of the object. On the other hand, tracking algorithms know more about the specific instance of the class they are tracking.

2. Tracking can help when detection fails: If you are running a face detector on a video and the person's face get's occluded by an object, the face detector will most likely fail. A good tracking algorithm, on the other hand, will handle some level of occlusion. In the video below, you can see Dr. Boris Babenko, the author of the MIL tracker, demonstrate how the MIL tracker works under occlusion.
3. Tracking preserves identity: The output of object detection is an array of rectangles that contain the object. However, there is no identity attached to the object. For example, in the video below, a detector that detects red dots will output rectangles corresponding to all the dots it has detected in a frame. In the next frame, it will output another array of rectangles. In the first frame, a particular dot might be represented by the rectangle at location 10 in the array and in the second frame, it could be at location 17. While using detection on a frame we have no idea which rectangle corresponds to which object. On the other hand, tracking provides a way to literally connect the dots!

Part 2: Mouse control using hand motion

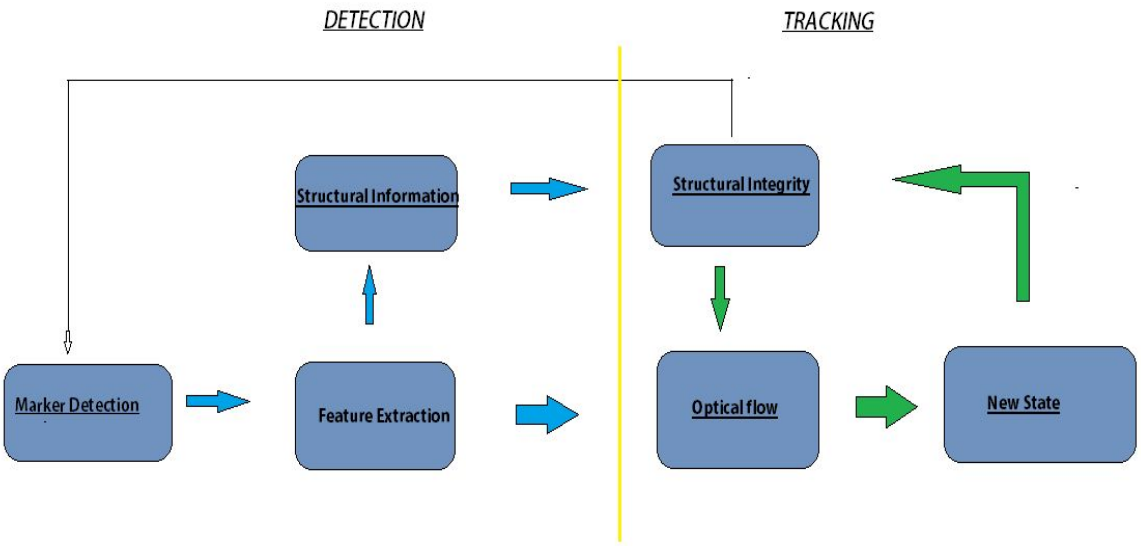


Figure 1: Overall approach of gesture control algorithm

Marker detection:

Detecting the fingertip offers a lot of inherent image processing difficulties which can be bypassed if certain type of marker detection algorithm is used in place of generic skin detection algorithm which has its own computational complexity as well as drawbacks. Here I propose a simplified approach to detect the fingertip using histogram backprojection with a custom marker.

Algorithm:

1. Marker of certain distinct color using cardboard is made to make it cheap and easily available.
2. Histogram model of created marker is generated.
3. The pixel in test image is compared with histogram model.
4. Histogram back projection technique is used to identify the region of interest where the marker can lie.

Feature Extraction:

Features which are easy to detect and are repeatable in different frames make good feature. Corner is one of the feature which is easily detectable and repeatable in different frames. Descriptors are used to define the feature to track or match them in two different frames.SIFT and SURF two famous feature descriptor which are used for matching and whole lot other applications . We use SIFT as descriptor to describe the feature of identified region in previous step of marker detection.

Structural Information:

The spatial and geometric orientation of feature points in space with respect to one another is unique and is in accordance with the information embedded in custom marker while designing the marker. Exploiting this spatial information while tracking gives additional information and reduces the redundancy of the whole process. The unique structural relation between the feature point is calculated and stored in memory while initializing. This information is used whenever tracking fails to track any of the feature point at any advance point in time then this geometrical relationship is utilized to initialize the position of lost feature with respect to known feature point.

Optical Flow:

The optical flow algorithm as defined in theoretical section is utilized here for the purpose of tracking the feature points. The relative motion between the observer and the

object causes the object in the scene to distort or lose its structural property. Hence verification of structural integrity is very important step and is checked in the next step.

Structural integrity:

The information stored in the initializing step is now used and the test for structural integrity is performed on the feature points of marker to check that the deformation in structure is within the tolerable limit. Now there arises two possible cases,one when the complete structure relationship disarray then we need to perform the marker detection and begin the process from step one in another case there is no complete loss in structural integrity but error accumulated in one the feature point is large which can propagate even larger error,to solve this problem the geometrical relationship is utilized to initialize the position of lost feature with respect to known feature points.

Conclusion:

The above mentioned algorithm provides a robust method for the purpose of tracking a marker using the image processing technique. The detected and localized marker now can be used for controlling the mouse pointer and its various other applications. It opens a new way of interaction for projection type displays and new frontier for human-computer interaction.

References:

1. Z. Zhang and Y. Shan, “Visual screen: Transforming an ordinary screen into a touchscreen,” in Proc. IAPR Workshop on Mach. Vision Appl., 2000, pp. 215–218
2. Development of Finger Touch Position Detection Methods Using Single Camera
3. Shenjing Chen , Lifeng Zhangb
4. Development of Single Camera Finger Touch Position Coordinate Transform Algorithm Shenjing Chen , Lifeng Zhang , Seiichi Serikawa
5. Transforming a Regular Screen Into a Touch Screen Using a Single Webcam Longyu Zhang, Jamal Saboune, Member, IEEE, and Abdulmotaleb El Saddik, Fellow, IEEE
6. S. K. Kang, M. Y. Nam, and P. K. Rhee, “Color based hand and finger detection technology for user interaction,” in 2008. Int.. Conf. on Convergence and Hybrid Inform. Technol. (ICHIT ’08), Aug. 2008, pp. 229–236
7. Wikipedia video tracking
8. Wikipedia optical flow
9. Wikipedia histogram normalization
10. Opencv documentation histogram back projection
11. Opencv documentation inverse perspective transformation
12. Opencv documentation structural information in an image