A survey on various Technologies used for Real Time Human Computer Interaction

Nikhil Samuel¹, Jilshna Jose Thanikkal², Fezin P C H³, Deepthi M⁴
¹Assistant Professor, Dept. Of Computer Science & Engineering, Holy Grace Academy of Engineering
²³⁴Dept. Of Computer Science & Engineering, Holy Grace Academy of Engineering

Abstract—This paper aims to present a survey of various unimodal (visual based in particular) Human – Computer interaction technologies that have been used nowadays for helping people with individuals to use computers and computer related activities like gaming. The world we live in has become suffused with computer technologies. They have created change and continue to create change. It is not only on our desktops and in our hands that this is manifest; it is in virtually all aspects of our lives, in our communities, and in the wider society of which we are a part. Utilizing computers had always begged the question of interfacing. The methods by which human has been interacting with computers has travelled a long way. The journey still continues and new designs of technologies and systems appear more and more every day and the research in this area has been growing very fast in the last few decades. The growth in Human-Computer Interaction (HCI) field has not only been in quality of interaction, it has also experienced different branching in its history. Instead of designing regular interfaces, the different research branches have had different focus on the concepts of multimodality rather than unimodality, intelligent adaptive interfaces rather than command/action based ones, and finally active rather than passive interfaces. This paper intends to provide an overview on the state of the art of HCI systems and cover most important branches as mentioned above.

Keywords—HCI, intelligent adaptive interactions, passive interfaces, multimodality and unimodality.

I. INTRODUCTION

Sometimes called as Man-Machine Interaction or Interfacing, concept of Human-Computer Interaction/Interfacing (HCI) was automatically represented with the emerging of computer, or more generally machine, itself. The reason, in fact, is clear: most sophisticated machines are worthless unless they can be used properly by men. This basic argument simply presents the main terms that should be considered in the design of HCI: functionality and usability [1]. Why a system is actually designed can ultimately be defined by what the system can do i.e. how the functions of a system can help towards the achievement of the purpose of the system. Functionality of a system is defined by the set of actions or services that it provides to its users.

However, the value of functionality is visible only when it becomes possible to be efficiently utilised by the user [2]. Usability of a system with a certain functionality is the range and degree by which the system can be used efficiently and adequately to accomplish certain goals for certain users. The actual effectiveness of a system is achieved when there is a proper balance between the functionality and usability of a system [3]. Having these concepts in mind and considering that the terms computer, machine and system are often used interchangeably in this context, HCI is a design that should produce a fit between the user, the machine and the required services in order to achieve a certain performance both in quality and optimality of the services [4]. Determining what makes a certain HCI design good is mostly subjective and context dependant. For example, an aircraft part designing tool should provide high precisions in view and design of the parts while a graphics editing software may not need such a precision. The available technology could also affect how different types of HCI are designed for the same purpose. One example is using commands, menus, graphical user interfaces (GUI), or virtual reality to access functionalities of any given computer.

II. EXISTING HCI TECHNOLOGIES

HCI design should consider many aspects of human behaviours and needs to be useful. The complexity of the degree of the involvement of a human in interaction with a machine is sometimes invisible compared to the simplicity of the interaction method itself. The existing interfaces differ in the degree of complexity both because of degree of functionality/usability and the financial and economical aspect of the machine in market. For instance, an electrical kettle need not to be sophisticated in interface since its only functionality is to heat the water and it would not be cost-effective to have an interface more than a thermostat on and off switch. On the other hand, a simple website that may be limited in functionality should be complex enough in usability to attract and keep customers [1]. Therefore, in design of HCI, the degree of activity that involves a user with a machine should be thoroughly thought.
The user activity has three different levels: physical [5], cognitive [6], and affective [7]. The physical aspect determines the mechanics of interaction between human and computer while the cognitive aspect deals with ways that users can understand the system and interact with it. The affective aspect is a more recent issue and it tries not only to make the interaction a pleasurable experience for the user but also to affect the user in a way that make user continue to use the machine by changing attitudes and emotions toward the user [1].

III. UNIMODAL HCI SYSTEMS

An interface mainly relies on number and diversity of its inputs and outputs which are communication channels that enable users to interact with computer via this interface. Each of the different independent single channels is called a modality [36]. A system that is based on only one modality is called unimodal. Based on the nature of different modalities, they can be divided into three categories:

1. Visual-Based
2. Audio-Based
3. Sensor-Based

IV. VISUAL-BASED HCI

The visual based human computer interaction is probably the most widespread area in HCI research. Considering the extent of applications and variety of open problems and approaches, researchers tried to tackle different aspects of human responses which can be recognized as a visual signal. Some of the main research areas in this section are as follow:

- Facial Expression Analysis
- Body Movement Tracking (Large-scale)
- Gesture Recognition
- Gaze Detection (Eyes Movement Tracking)

While the goal of each area differs due to applications, a general conception of each area can be concluded. Facial expression analysis generally deals with recognition of emotions visually [8] [9] [10]. Body movement tracking [11] [12] and gesture recognition [13] [14] [15] are usually the main focus of this area and can have different purposes but they are mostly used for direct interaction of human and computer in a command and action scenario. Gaze detection [16] is mostly an indirect form of interaction between user and machine which is mostly used for better understanding of user’s attention, intent or focus in context-sensitive situations [17].

The exception is eye tracking systems for helping disabilities in which eye tracking plays a main role in command and action scenario, e.g. pointer movement, blinking for clicking [18]. It is notable that some researchers tried to assist or even replace other types of interactions (audio-, sensor-based) with visual approaches. For example, lip reading or lip movement tracking is known to be used as an influential aid for speech recognition error correction [19].

A. Computer Vision Face Tracking For Use in a Perceptual User Interface

Computer vision algorithms that are intended to form part of a perceptual user interface must be fast and efficient. They must be able to track in real time yet not absorb a major share of computational resources: other tasks must be able to run while the visual interface is being used. This algorithm is based on a robust nonparametric technique for climbing density gradients to find the mode (peak) of probability distributions called the mean shift algorithm. In this, we want to find the mode of a color distribution within a video scene. Therefore, the mean shift algorithm is modified to deal with dynamically changing color probability distributions derived from video frame sequences. The modified algorithm is called the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm.

This algorithm uses a tracker that tracks a given face in the presence of noise, other faces, and hand movements. Moreover, it runs fast and efficiently, such that, objects can be tracked in real time (30 frames per second) while it consumes as few system resources as possible. In other words, this tracker is able to serve as part of a user interface that is in turn part of the computational tasks that a computer might routinely be expected to carry out. This tracker also run on inexpensive consumer cameras and do not require calibrated lenses.

In order to achieve this, this technology focused on color-based tracking [19][20][21][22][23][24], yet even these simpler algorithms are too computationally complex (and therefore slower at any given CPU speed) due to their use of color correlation, blob and region growing. Kalman filter smoothing and prediction, and contour considerations. The complexity of these algorithms derives from their attempts to deal with irregular object motion due to perspective (near objects to the camera seem to move faster than distal objects); image noise; distractors, such as other faces in the scene; facial occlusion by hands or other objects; and lighting variations.
We want a fast, computationally efficient algorithm that handles these problems in the course of its operation. To develop such an algorithm, this technology drew on ideas from robust statistics and probability distributions. Robust statistics are those that tend to ignore outliers in the data (points far away from the region of interest). Thus, robust algorithms help compensate for noise and distractors in the vision data. Therefore it chose to use a robust nonparametric technique for climbing density gradients to find the mode of probability distributions called the mean shift algorithm [25]. (The mean shift algorithm was never intended to be used as a tracking algorithm, but it is quite effective in this role.)

The mean shift algorithm operates on probability distributions. To track colored objects in video frame sequences, the color image data has to be represented as a probability distribution [26]; we use color histograms to accomplish this. Color distributions derived from video image sequences change over time, so the mean shift algorithm has to be modified to adapt dynamically to the probability distribution it is tracking. The new algorithm that meets all these requirements is called CAMSHIFT.

The Continuously Adaptive Mean Shift Algorithm (CamShift) is an adaptation of the Mean Shift algorithm for object tracking that is intended as a step towards head and face tracking for a perceptual user interface. For face tracking, CAMSHIFT tracks the X, Y, and Area of the flesh color probability distribution representing a face. Area is proportional to Z, the distance from the camera. Head roll is also tracked as a further degree of freedom. We then use the X, Y, Z, and Roll derived from CAMSHIFT face tracking as a perceptual user interface for controlling commercial computer games and for exploring 3D graphic virtual worlds.

CamShift is primarily intended to perform efficient head and face tracking in a perceptual user interface (Bradski, 1998). It is based on an adaptation of Mean Shift that, given a probability density image, finds the mean (mode) of the distribution by iterating in the direction of maximum increase in probability density (Intel Corporation, 2001).

The primary difference between CamShift and the Mean Shift algorithm is that CamShift uses continuously adaptive probability distributions (that is, distributions that may be recomputed for each frame) while Mean Shift is based on static distributions, which are not updated unless the target experiences significant changes in shape, size or color. Level-2 Heading: A level-2 heading must be in Italic, left-justified and numbered using an uppercase alphabetic letter followed by a period. For example, see heading “C. Section Headings” above.

B. Modified CAMSHIFT Algorithm

CAMSHIFT is a tracking algorithm that has already been used for face-tracking. It is, however, not being successfully used for people/person tracking. In order make it possible to track people in motion, this modified CAMSHIFT algorithm, incorporates the motion information with CAMSHIFT algorithm, which allows it to track people successfully even in the case of occlusions.

To perform people tracking, modified CAMSHIFT algorithm used Mixture of Gaussian (MoG) method to calculate a good background model. The model assumes that each pixel in the scene is modeled by a mixture of K Gaussian distributions where different Gaussians represent different colors. The weight parameters of the mixture represent the time proportions that those colors stay in the scene. Thus, the probable background colors are the ones which stay longer and are more static. Static single-color objects trend to form tight clusters in the color space while moving ones form widen clusters due to different reflecting surfaces during the movement. The measure of this was called the fitness value [27].

In order to improve the results of the CAMSHIFT algorithm described above, we use an optical flow algorithm (Lucas-Kanade Method) [28] to determine the motion of the tracked persons.

Lucas-Kanade Algorithm

The basic idea of the LK algorithm rests on three assumptions:

1) \textbf{Brightness constancy:} A pixel from the image of an object in the scene does not change in appearance as it (possibly) moves from frame to frame. For grayscale images, this means we assume that the brightness of a pixel does not change as it is tracked from frame to frame.

2) \textbf{Temporal persistence or small movements:} The image motion of a surface patch changes slowly in time. In practice, this means the temporal increments are fast enough relative to the scale of motion in the image that the object does not move much from frame to frame.

3) \textbf{Spatial coherence:} Neighboring points in a scene belong to the same surface, have similar motion, and project to nearby points on the image plane.

4) The OpenCV function implements sparse iterative version of Lucas-Kanade optical flow in pyramids [28]. It calculates coordinates of the feature points on the current video frame given their coordinates on the previous frame. The function finds the coordinates with sub-pixel accuracy.
This algorithm is using color feature similar to classical CAMSHIFT algorithm, motion or optical flow information is also added to it to make it robust against occlusions.

C. Hands-free Interface

Hands-free Interface. is an assistive technology that is intended mainly for the use of the disabled. It helps them to use their voluntary movements, like head movements; to control computers and communicate through customized educational software or expression building programs. People with severe disabilities can also benefit from computer access to partake in recreational activities, use the Internet or play games.

The system uses a USB camera to capture the user’s face motion. The algorithm tracks the motion accurately to control the cursor, thus providing an alternative to the computer mouse or keyboard.

There are several differences in this method compared to other methods we discussed before. The first difference is that, they were computationally expensive, inaccurate or suffered from occlusion [29]. For example, the CAMSHIFT algorithm uses skin color to determine the location and orientation of the head. Although fast and does not suffer from occlusion, this approach lacks precision since it works well for translational movement but not rotational [30]. Other tracking methods use Eigen Space matching techniques, however, they are too computationally expensive to work in real time [30].

Hands-free Interface uses face tracking technique that solves the aforementioned problems and can be used for various real time applications.

Interface used in this algorithm prompts the user to select a region to be tracked, which will be used as an initial feature template. We will use the sum of square difference (SSD) correlation method to locate the feature in subsequent frames. A similarity metric is used to rate the similarity of the template and the image patch. The one with the best score is selected. In order to reduce the number of computations required, search is restricted to a specified region of interest referred to as the search window. The center and size of the search window will automatically adapt to the speed and direction of motion by use of a Kalman estimator.

Hands-free Interface is a promising alternative or extension to pointing devices. It exhibits accuracy and speed, which are sufficient for many real time applications and which allow handicapped users to enjoy many computer activities.

D. Interactive Gaze Controller

This methodology aims to present an application that is able of replacing the traditional mouse with the human face as a new way to interact with the computer. Facial features (nose tip and eyes) are detected and tracked in real-time to use their actions as mouse events. The coordinates and movement of the nose tip in the live video feed are translated to become the coordinates and movement of the mouse pointer on the user’s screen. The left/right eye blinks fire left/right mouse click events. The external devices that the user needs is a webcam that feeds the program with the video stream and a microphone. Instead of using the mouse system uses the facial features of the user. The nose tip was selected as the pointing device; because of the location and shape of the nose; as it is located in the middle of the face it is more comfortable to use it as the feature that moves the mouse pointer and defines its coordinates. It basically does not change its distinctive convex shape which makes it easier to track as the face moves. Eyes were used to simulate mouse clicks, so the user can fire their events as he blinks.

The algorithm is outlined as follows:

Step 1: Track the nose tip.
Step 2: Track the BTE.
Step 3: Detect the eye brows.
Step 4: Detect the eye blinks.
Step 5: Track the eyes.
Step 6: Refine BTE location.
Step 7: Refine nose tip location

Before explaining the algorithm some important steps need to be mentioned. They are:

Setting feature’s ROI

Use the location of the tracked feature in the past two frames (at moments t-1 and t-2) to predict its location in the current frame (at moment t). To do so, we will calculate the shift value that the feature’s template has made between frames t-2 and t- 1, and shift the feature’s ROI in the current frame from the feature’s last place (in frame t-1) with that shift value.

Template Matching

Look for the feature’s new location in the ROI that was set in the previous step. There are two different matching techniques; the first is based on histogram matching, while the second uses the sum of squared differences. The method that is in our tracking process is the sum of squared differences (SSD).
i. A window that has the feature’s template size is scanned over the ROI and the SSD between the template and the current window is calculated.

ii. After scanning the entire ROI the window that has the smallest SSD is chosen as the template’s match, and its location is considered as the feature’s new location.

iii. In order to achieve faster results; while calculating the SSD, if its value is still smaller than the smallest SSD so far, continue its calculation;

iv. Else skip to the next window in the ROI, because we are sure that the current SSD will not be the smallest one.

Selecting the Feature’s Template for Matching

In each frame we apply template matching with the feature’s first template and with the template from the previous frame; this way the matching with the first template will insure that we are tracking the right feature (e.g. if it reappears after an occlusion), as for matching with the template from the previous frame, it ensures that we are still tracking the same feature as its state changes.

Step1: Track the nose tip

Tracking the nose tip will be achieved by template matching inside the ROI

Step2: Track the BTE

The BTE now represents a different concept from the one in the detection mode; it refers to a 15*15 template which is located between the eyes. The BTE is tracked like the nose tip.

Step3: Detect the eye brows

To detect the eyebrow we take a small region above the eye’s expected position and threshold it; since that the region above the eye contains only the eyebrow and the forehead, the thresholding should result in points which represent the eyebrow. To find the eyebrow line from the set of thresholding points we apply the Hough transform.

It is used in our eyebrows detection algorithm. Suppose that we have a set of points, and we need to find the line that passes as many of these points as possible.

In Hough transform the line has two attributes: Θ and τ. To detect the line that passes from the set of points, Hough transform works this way:

For each point in the set:

i. Find the lines that pass from this point (with 8-connectivity net we have 4 lines).

ii. Find the Θ and τ of each line.

iii. For each line: If it already exists (there is a line that has the same Θ and τ, and passes from another point) increase its counter by 1. If it is a new line; create a new counter and assign the value 1 to it.

iv. At the end, the line that has the greatest counter value will be chosen as the detected line, because the maximum counter value means that this line passes from a larger number of points than any other line.

Step4: Detect the eye blink

The blink detection process is run only if the eye is not moving (we consider the eye not moving if it didn’t change its location in the past two frames), because when a person uses the mouse and wants to click, he moves the pointer to the desired location, stops, and then clicks; so basically the same for using the face: the user moves the pointer with the tip of the nose, stops, then blinks. To detect a blink we apply motion detection in the eye’s ROI; if the number of motion pixels in the ROI is larger than a certain threshold we consider that a blink was detected, because if the eye is still, and we are detecting a motion in the eye’s ROI, that means that the eyelid is moving which means a blink. In order to avoid multiple blinks detection while they are a single blink (because motion pixels will appear while the eye is closing and reopening), the user can set the blink’s length, so all blinks which are detected in the period of the first detected blink are omitted. If the left and right blinks are detected at the same time (or in the duration of one of them); discard these blinks and consider them involuntary.

Step5: Track the eyes

To achieve better eyes tracking results use the BTE (a steady feature that is well tracked) as the reference point

i. At each frame after locating the BTE and the eyes, calculate the relative positions of the eyes to the BTE;

ii. In the next frame after locating the BTE assume that the eyes have kept their relative locations to it, so place the eyes’ ROIs at the same relative positions to the new BTE (of the current frame);

iii. To find the eye’s new template in the ROI: first use template matching, the second searched in the ROI for the darkest 5*5 region (because the eye pupil is black), then use the mean between the two found coordinates as the eye’s new location. The problem with the darkest region method was that it picked the eyebrow sometimes as the eye; here comes the eyebrow detection role where the eye’s ROI is placed under the detected eyebrow line to avoid picking it as the eye.
Step6: Refine BTE location

BTE is chosen as a 15*15 template in the middle of the line that connects the eyes, then center the BTE’s ROI on that point and apply template matching to find the final BTE.

Step7: Refine nose tip location

Basically the length of the line that connects the BTE and the nose tip (nose line) should be smaller than the length of the eyes line; if not, nose refinement is applied. If the face plane is parallel to the capturing device; the nose line should be perpendicular on the eyes line. If the face plane was not exactly parallel or the face was rotated, the angle will not be exactly 90 degrees. Calculate the angle between the nose line and the eyes line; if it does not fall inside 90 ± 30 degrees; nose refinement is applied. Center the nose tip ROI at the end of a line that is perpendicular on the eyes line in the BTE location, and which length is half of the eyes line length. Template matching is applied to find the final nose tip location inside the ROI.

V. CONCLUSION

Computer vision-based interaction is an emerging technology that is becoming more useful, effective, and affordable. However, it raises new questions from the HCI viewpoint, for example, which environments are most suitable for interaction by users with disabilities. In general, computer vision interaction systems are very useful; in some cases, these systems are the only ways by which some people can interact with a computer. Computer vision-based interaction systems also give advantages—such as flexibility and lower cost—over other traditional assistive technologies. We have discussed four of the recent advancements in the field of HCI. All of these methods are aiming at the same goal and that is to provide or help people with disabilities to use computers like anybody else. Each of these methods are related to one another as they use the same concept. There are only few differences among these techniques. Interactive Gaze Controller is the latest among all these four methodologies. The speed with which these techniques can be used has to be improved and in the coming years we can expect that this drawback will be removed and thus disabled personalities can also involve in the modern era of digital systems.

REFERENCES


