Hands Free Interface for Human Computer Interaction

S. Kasun Chathuranga*, K.C. Samarawickrama¤, H.M.L. Chandima§, K.G.T.D. Chathuranga‡, A.M. Harsha S. Abeykoon†
Department of Electrical Engineering,
University of Moratuwa, Sri Lanka.
Email: *kasunc@ieee.org, ¤skasunchamara@gmail.com, §lahiruchandima@gmail.com, ‡thilinauom@gmail.com,
†harsha@elect.mrt.ac.lk

***Abstract*— At present, there are no adequate solutions which facilitate the interaction of computers to handicapped people. This research addresses that issue and it provides a comprehensive solution for the problem. The product provides an alternative to the conventional mouse using computer vision and voice recognition technologies as key techniques. The user can use his or her nose's relative movements to move the mouse pointer and use other mouse actions by voice commands. Enhanced template matching image processing technologies have used to track the nose position. Background light changes are rectified with using adoptive thresholds. The voice interaction is achieved by using Speech API calls and overall mouse events are passed to system using system API calls. The only additional hardware components used are, a web cam and a microphone which come by default in most laptop computers in built. And such external microphones and web cameras are available in very affordable price.**

Keywords- hands free interface; nose mouse; mouse for disabled people, nose tracking, ehnaced template maching, adaptive threshold

VI. Introduction

At present, the attention of the society is paid for providing various facilities to disabled people so that they can fulfill their needs conveniently as normal people. [1] However, there is very little attention paid for providing solutions for disabled people to use computers conveniently. [2] To pursuit computers functions properly, the user has to interact with its two most important input devices of a computer [3], which are the keyboard and the mouse which are not designed to aim to be used by disabled people. [4]

Thus, this research is aimed to give a comprehensive solution, which could actually be used by a disabled person in comfortable manner. Special attention is given to solve main difficulties which may arise during the operation of a mouse. However this solution is not limited use of disabled people. This can also used as a hand free interface to a computer in a case of mouse is not available.

The solution consists of two hardware components, a web camera and a microphone. The web camera is expected to be pointed to the face of the person who uses the computer. The software component of the solution accesses the video stream coming from the web camera and moves the mouse pointer accordingly with the nose movements. The microphone captures the voice of the user and detects the voice commands which are used to issue mouse actions such as click, double click and right click. Drag and Scroll events also can be done. And finally mouse events are passed to operating system through System API [5] which is Win32 API in windows.

VII. Methodology

A. Basic Design

Hands Free Interface for Human Computer Interaction is a fusion of two major operations. They are image manipulation based nose tracking and speech recognizing based mouse command issuing part. Image manipulation part can also subdivide in to Face Detection and Nose Tracking.

B. Image Manipulation

1) Face Detection

To move the mouse pointer using face movements, it is necessary to track either the whole face or a part of the face. If the mouse pointer is moved by the movement of the whole face, it is not feasible to move the mouse pointer significantly by rotating the face because the camera does not see a significant difference of the position of the face. But, if some feature of the face is tracked, that feature would move significantly even when the user just rotates the face. Thus, tracking some feature in the face such as the eyes or nose is more effective.

The item selected to track is the nose and it is first necessary to identify the nose. Thus, the face is detected first, and the nose is located in the middle of the detected face by relative dimensions. The method of object detection used for detecting the face is the Haar like features searching algorithm. These features of images are similar to the Harr wavelets introduced by Alfred Haar. These Harr like feature manipulation [6] was introduced by Viola et al, the algorithm is implemented in OpenCV in the function cvHarrDetectObjects.

The cvHarrDetectObjects requires a file called a haar classifier cascade, which includes the details on what type of objects the function should look for. This is some kind of a trainer which trains the function to find the desired objects. What this classifier cascade includes is the haar like features of several hundreds or even thousands of images of the objects which belong to the type of objects in question. The cvHarrDetectObjects looks for the similar haar like features specified by the classifier cascade in the given image and returns the information on the detected objects. The performance of the function in recognizing faces with this

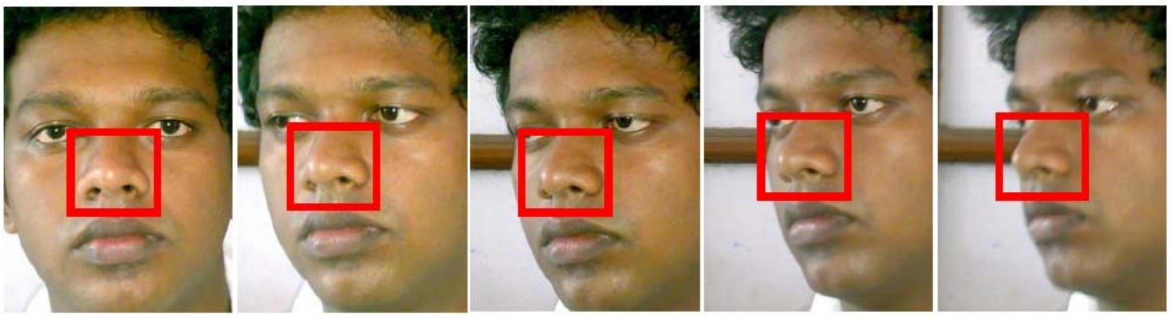

Fig. 1. Human Nose Tracking

classifier cascade is satisfactory and it was not required to build a new classifier cascade for the frontal view of the face.

2) *Nose Tracking*

Tracking is the heart of Hands Free Interface for Computer Interaction. There for implementing a robust effective tracking algorithms are very important. Tracking algorithms should be Not losing tracking point, Tracking the correct point, Lighting error correction and Low CPU consumption

Otherwise, tracking position will be loss and mouse position control will be a real hard task. Lighting correction is also very important that in a dynamic environment. CPU consumption becomes an important issue in case of low end computers. Computational calculations should be less complicated and should be linear as far as possible.

Template matching is a technique which is used here for finding small parts of an image which match a template image. Image template is created from the area which is interested in to track.

Template matching can be subdivided between two approaches as Feature-based matching and Template-based matching.

The template-based, or global, approach uses the entire template, with generally a sum-comparing metric that determines the best location by testing all or a sample of the viable test locations within the search image that the template image may match up to [7], which is used here. Sum of absolute differences (SAD) is used, which is an extremely simple algorithm for finding the correlation between image blocks [8]. It works by taking the absolute difference between each pixel in the original block and the corresponding pixel in the block being used for comparison (1).

$$SAD(x,y) = \sum_{i=0}^{Trows} \sum_{j=0}^{Tcols} Diff(x+i, y+j, i, j) \qquad (1)$$

However calculation of sum of absolute is not adequate. Therefore additional methods have to be implemented to filter the result and correct the errors from it. The two main such schemes which are implemented here are called as 3-D Dual Tracking and Dynamic Threshold.

a) 3-D Dual Tracking

In simple template matching, a sample template is taken in memory which is taken from the initial image from the of the image sequence. Later on the position of the initial template is searched over new images which are on the image sequence from the capturing device.

However this works very well with a creature like Buck (character in Ice Age 3 movie) because he has a brown nose in a white face. Color difference creates significant sum of absolute differences which can easily track that sort of nose. In case of humans after few frames co-relevance to 1st frame and later frames is dramatically reduced due to the change of position of the head and it effects heavily on sum of absolute differences.

If the 1st marked template of 1st image is searched for in entire 5 images in Fig. 1, due to the dissimilarity, tracking may be lost at end. This scenario is also happened to both eyes and mouth. And to avoid this, searching key template is updated frame by frame.

However updating key searching frame by every frame may introduce minor error and that error may increase with the time which may leads to tracking completely other place than the initial place. Hence additional parallel tracking is done with using very 1st template and most suitable position of the image is taken by comparing two concurrent tracking methods.

This method is named as 3-D Dual Tracking because it is used in tracking of 3-D object within a 2-D view which is a completely developed for this research.

a) Dynamic Threshold

Main problem in tacking is changing surrounding lighting and brightness condition. If the tracking is done by previous frame's part, surrounding condition will not affect much because of the time of gap of the two frames is in milliseconds. However it may be a significant issue with comparing to very 1st frame. Hence even the initial position reoccurred in image of an image sequence may not be detect due to this. Therefore threshold for frame by frame tracking can keep as constant. However threshold initial frame tracking should be changed time to time for best performances. Otherwise initial frame tracking will be fail due to exceeding thresh hold.

Open CV cvMatchTemplate with normalized squared difference results have been used in this program for template matching which can be expressed in (2) [9].

$$S(x,y) = \frac{\sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1}[T(x',y') - I(x+x',y+y')]^2}{\sqrt{\sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1}T(x,y)^2\sum_{y'=0}^{h-1}\sum_{x'=0}^{w-1}I(x+x',y+y')^2}} \quad (2)$$

Where I(x, y) is the value of the image pixel in the location (x, y), while T(x, y) is the value of the template pixel in the location (x, y). Function cvMinMaxLoc is used to obtain the least value from the results which gives the best match. This minimum value is used compare with thresholds.

In Dynamic Threshold, average of minimum value of normalized squared difference result of the frame by frame template matching is used for initial frame tracker threshold. This value is taken with calculating last ten results. If this value is too small, preconfigured value is inserted for threshold. This increases the performances of 3-D Dual Tracking significantly.

C. *Speech Recognition and Mouse Event Detection*

Microsoft SAPI (Speech API) was used as the voice command detection engine. SAPI provides functionalities for recognizing and synthesizing human voice.

SAPI comes with the default installation of Microsoft® Windows operating systems from Windows 95 onwards. It is free to be used by any application running on Windows. SAPI basically provides two basic functionalities [10]. They are Voice recognition and Speech synthesis.

Only the voice recognition functionality is necessary for the solution. Voice commands were the key to implement the mouse events. Each mouse event can be given a certain voice command and the corresponding mouse event can be issued at the detection of a valid voice command. This method was implemented to test the suitability and it was finally taken as the method of issuing the voice commands [11].

Five mouse events and two control commands are directed using voice commands. These commands are pre specified and user cannot customize the voice commands at his or her will. This constraint is basically added by the technology used for the speech recognition. Following is a list of voice commands the software solution accepts.

- Click: Issues a left mouse button click
- Double: Issues a left mouse button double click
- Right: Issues a right mouse button click
- Hold: Press and hold left mouse button
- Release: Release the left mouse button held down
- Stop: Stop the mouse emulation
- Start: Resume the stopped mouse emulation

The classes in SAPI cannot be used directly in C++ code. SAPI provides a COM interface which facilitates many programming languages to use SAPI. SAPI class objects are created as COM objects and used inside C++ code.

D. *Top Level Architecture*

Fig. 2 shows the top level architecture of the system, which describes the major blocks which the solution is comprised of.

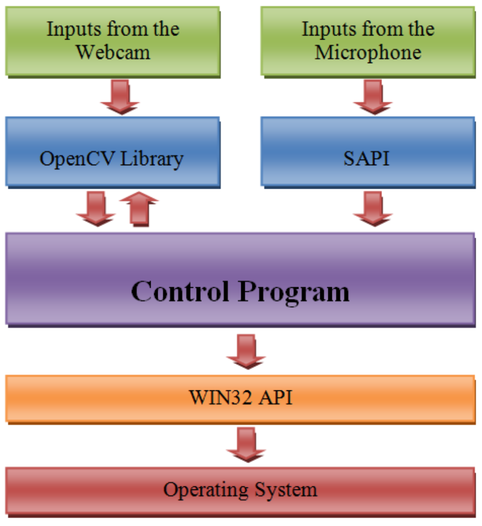

Fig. 2. Top Level Architecture

VIII. Results

Following test were conducted in testing phase to identify defects and to isolate, and subjected to rectification and ensure that the defect is rectified in order to get a quality product at the end.

A. *Success Probability Test for Different Noise Levels*

1) *Description*

Sample was taken ten random students from University of Moratuwa. Each person tested one word in ten times. All values were taken as percentages. Sound level values were measured over 10 minutes (20 samples) and the average is taken (decibel meter is used to measure sound level)

2) *Results*

TABLE I. HEADPHONE MICROPHONE (SOMIC STEREO DYNAMIC HEADPHONES – CD 750)

Voice Command	Success Probability (%)				
	Personal Room (silent) 55 dB	Computer Lab (Occasional Voice) 62 dB	Talking With someone (Average Noise) 67 dB	Public Environment (Noisy) 71 dB	Average Environment
Click	98.0	94.0	84.0	72.0	87.0
Double	98.0	92.0	85.0	74.0	87.3
Right	99.0	96.0	90.0	84.0	92.3
Hold	96.0	90.0	80.0	69.0	83.8

Release	96.0	91.0	81.0	70.0	84.5
Stop	98.0	92.0	83.0	71.0	86.0
Start	97.0	92.0	84.0	72.0	86.3
Average Command	97.4	92.4	83.9	73.1	86.7

TABLE II. DESKTOP MICROPHONE (SOMIC V03 TABLE STAND-ALONE MICROPHONE)

Voice Command	Success Probability (%)				
	Personal Room (silent) 55 dB	Computer Lab (Occasional Voice) 62 dB	Talking With someone (Average Noise) 67 dB	Public Environment (Noisy) 71 dB	Average Environment
Click	94.0	82.0	69.0	46.0	72.8
Double	95.0	81.0	67.0	54.0	74.3
Right	95.0	88.0	83.0	67.0	83.3
Hold	92.0	76.0	60.0	48.0	69.0
Release	93.0	79.0	62.0	39.0	68.3
Stop	94.0	83.0	68.0	47.0	73.0
Start	93.0	82.0	69.0	55.0	74.8
Average Command	93.7	81.6	68.3	50.9	73.6

TABLE III. LAPTOP MICROPHONE (ACER EXTENSA 5630-662G32MN, MODEL NO: MS2231)

Voice Command	Success Probability (%)				
	Personal Room (silent) 55 dB	Computer Lab (Occasional Voice) 62 dB	Talking With someone (Average Noise) 67 dB	Public Environment (Noisy) 71 dB	Average Environment
Click	96.0	88.0	79.0	58.0	80.3
Double	96.0	86.0	77.0	64.0	80.8
Right	97.0	92.0	88.0	74.0	87.8
Hold	94.0	83.0	76.0	57.0	77.5
Release	94.0	85.0	69.0	54.0	75.5
Stop	96.0	87.0	73.0	63.0	79.8
Start	95.0	87.0	74.0	65.0	80.3
Average Command	95.4	86.9	76.6	62.1	80.3

3) Graphical Comparison

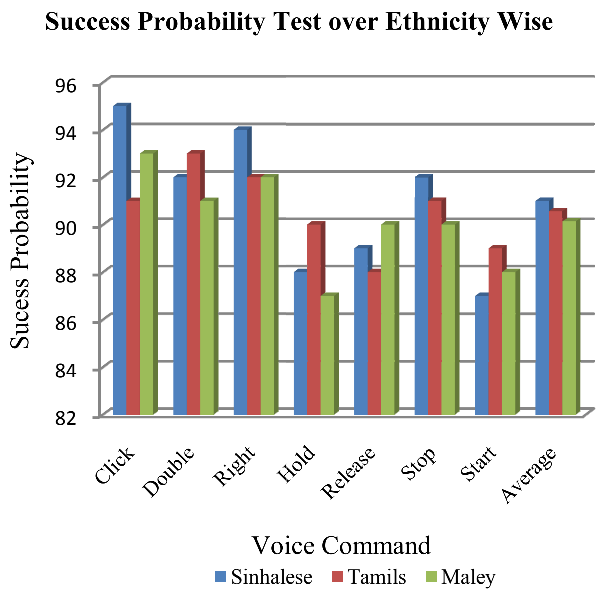

Fig. 3. Success Probability over Difference Voice Commands

4) Analysis

The highest success probability was belongs to command "Right" (Fig. 3) Because of the pronunciation of word "Right" is quite uncommon. So the probability of being including it in noise was very low. The lowest success probability was belongs to command "Hold" The probability of being including it in noise is quite high. The best success probability shows with the use of headphone microphone. Because the noise comes to microphone was relatively low compare with actual command. The worst success probability gives by laptop microphone because it has expose to wide angle of noises and has relatively high noise level compare to actual command.

B. Awkwardness of Using "NoseMouse" with Distance between User and Screen

1) Description

Sample was taken ten random students from University of Moratuwa. Each person tested for each distance in one time. All values were taken as Average time measured in seconds. All time readings were taken by one person by using same stopwatch (Casio Gshock G9000-1V Module 3031). Distances were measure from webcam to top of the user's nose using a standard meter rule.

Model of the laptop was Acer Extensa 5630-662G32Mn Model NO: MS2231. CPU was Intel Core 2 Duo T6600 (2.2 GHz, 2 MB Cache memory). Amount of RAM was 2 GB DDR2 and Hard disk drive has 320 GB capacity. Acer Crystal

Eye Web Cam had 640 x 480 resolution. Screen was a15.4" WXGA LCD screen.

Awkwardness Level has defined as the average time taken to complete a task. The task was designed to move the mouse pointer from start menu position to my computer icon placed on top left corner of the screen and open it by double clicking and go to the hard driver C by double clicking and create a new folder by right click option and finally close the My Computer window.

2) Results

TABLE IV. AWKWARDNESS LEVEL VS. DISTANCE

Distance	Awkwardness
30	34
40	28
50	25
60	25
70	27
80	31
90	33
100	36

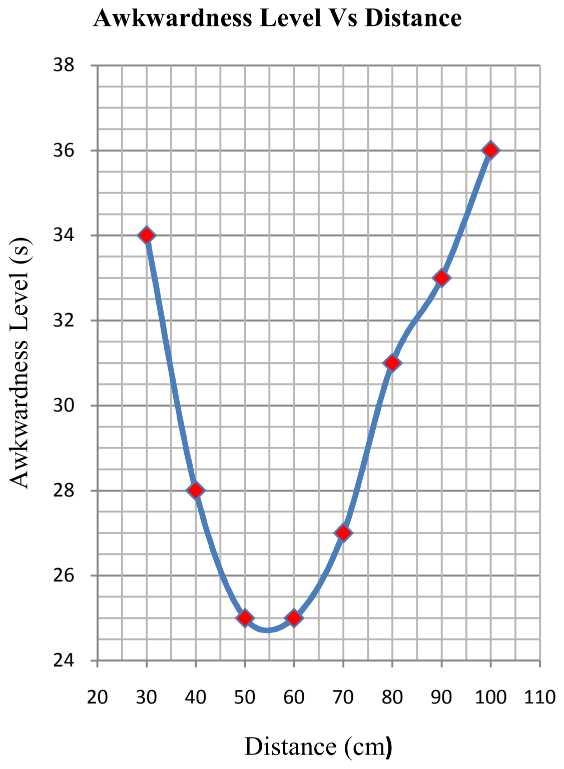

Fig. 4. Awkwardness Level vs. Distance

3) Analysis

By analyzing above graph, it can be identified that awkwardness level got differed with the distance as a minimal. Therefore to achieve the best smoothness level it is necessary to maintain the distance value in between 50cm to 60cm (Fig. 4). In further increase of distance, it made easy to move the mouse pointer but sensitivity was getting low which leads move head in lager angle to move the mouse pointer particular distance along the screen. I.e. higher the distances lower the difficulty and lower the sensitivity. In case of decrease the length, the sensitivity went high, but to move a mouse pointer to a one particular place got more difficult. I.e. Lower the distance higher the difficulty and higher the sensitivity.

C. Initial Nose Identifying Time

1) Description

Sample was taken ten random students from University of Moratuwa. Each person tested one aspect in ten times. All readings were taken by one person (reaction time was almost same). Average values are tabulated in table V. Same watch used to measure every time reading (Casio Gshock G9000-1V Module 3031).

2) Results

TABLE V. INITIAL NOSE IDENTIFYING TIME

Lighting Condition	Nose identifying time (s)	Tracking Retaining Time (min)
Face illuminated with a front light	3	15
Back light in the camera viewport	11	6

3) Analysis

When the face was illuminated with a front light it was pretty easy to recognize it as a human face and can track the nose position easily. That's why it identified the face with the least time of three seconds. When there was a bright outdoor while having low face illumination solution faced difficulties to recognize the human faces, because with the low face illumination, the face may seen as a dark circular type object which doesn't exhibit face's characteristics. When there was a back light in the camera view point though the face was quit bright illuminated, it took more than default time given to identify a face. When the camera lens flashed with an external light sources, it would be little difficult to catch the face correctly.

IV. CONCLUSION

With all 3 tests mentioned above, optimum operation points of the solution can be estimated. To operate the solution with minimum effort, the distance between web camera and face should be maintained in between 50 cm and 60 cm. Even this may little vary with web camera, in most cases 640 x 480 resolution camera would exhibit this range which is most used as computer web cameras.

In case of face recognition, it is better to have the light source behind the computer and not behind the user. This would reduce initial face identifying time and number of tracking lost which might be occurred but which do not make much issue. Using two face samples for identifying people with spectacles can be implemented which also decrease the face identifying time. This option might be required in case of person with very large spectacles is needed to use the solution.

With the analysis of these results, it is very much clear that solution work appropriate as expected. With a combined solution of image processing and voice based command issuing, a comprehensive solution of Hands Free Interface for Human Computer Interaction which facilitates use of computers for handicapped people can be implemented.

REFERENCES

[1] Loiacono, E.T.; McCoy, S.; Chin, W.; , "Federal Web site accessibility for people with disabilities," *IT Professional* , vol.7, no.1, pp. 27- 31, Jan-Feb 2005

[2] Burger, D.; , "Improved access to computers for the visually handicapped: new prospects and principles," *Rehabilitation Engineering, IEEE Transactions on* , vol.2, no.3, pp.111-118, Sep 1994

[3] Noyes, J.; , "QWERTY-the immortal keyboard," *Computing & Control Engineering Journal* , vol.9, no.3, pp.117-122, Jun 1998

[4] Yu-Luen Chen; Chang, W.H.; May-Kuen Wong; Tan-Fuk Tang; Te-Son Kuo; , "An application of infrared technique to control a computer keyboard for handicap," *Consumer Electronics, 1997. ISCE '97., Proceedings of 1997 IEEE International Symposium on* , vol., no., pp.83-86, 2-4 Dec 1997

[5] Roark, C.; , "Operating system application programmer interfaces (API) and their role in open systems for avionics," *Aerospace and Electronics Conference, 1995. NAECON 1995., Proceedings of the IEEE 1995 National* , vol.1, no., pp.271-278 vol.1, 22-26 May 1995

[6] Viola, P.; Jones, M.; , "Rapid object detection using a boosted cascade of simple features," *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* , vol.1, no., pp. I-511- I-518 vol.1, 2001

[7] Wu-Chih Hu; , "Adaptive Template Block-Based Block Matching for Object Tracking," *Intelligent Systems Design and Applications, 2008. ISDA '08. Eighth International Conference on* , vol.1, no., pp.61-64, 26-28 Nov. 2008

[8] Friemel, B.H.; Bohs, L.N.; Trahey, G.E.; , "Relative performance of two-dimensional speckle-tracking techniques: normalized correlation, non-normalized correlation and sum-absolute-difference," *Ultrasonics Symposium, 1995. Proceedings., 1995 IEEE* , vol.2, no., pp.1481-1484 vol.2, 7-10 Nov 1995

[9] Open Source Computer Vision Library Reference Manual, Intel Corporation, 2001

[10] Microsoft Developer Network. Microsoft Speech API: Developing Speech Applications. [Online]. Available: http://www.microsoft.com/speech/developers.aspx

[11] Microsoft Developer Network. Microsoft Speech API (SAPI) 5.3. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms723627%28VS.85%29.aspx