Using head orientation to control a mouse

Zachary Taylor
University of Canterbury
Christchurch, New Zealand
zjt14@uclive.ac.nz

Richard Green (Supervisor)
Computer Science Department
University of Canterbury
Christchurch, New Zealand
Richard.green@canterbury.ac.nz

***Abstract* — An approach for controlling a mouse cursor via a webcam by detecting the position of a face has been developed. The approach uses the Lucas Kanade optical flow algorithm to track the faces location. The Viola-Jones method running in a separate thread continually recalibrates the points that are being tracked. This combines the accuracy of the Viola-Jones method with the speed of the Lucas Kanade method. The sensitivity of the mouse is scaled by a factor that relates to recent mouse movement to allow both large movements and small accurate movements of the mouse to be achieved. A gesture system is used to allow the same number of clicking options as would be found on a regular mouse. Initial testing has shown that the system provides the same functionality as a regular mouse however actions have been found to take approximately four to five times as long to perform.**

I. INTRODUCTION

This paper looks at the problem of allowing a person to use their face to control a mouse via a standard webcam. The purpose of the program is to allow people to interact with a display in a public location without a mouse or keyboard present. The person must simply be able to walk up to the monitor and immediately start moving the mouse.

This application means the program must require no form of setup on receiving a new user before giving control of the mouse. It also means that for the solution to be practical the user cannot be required to wear any type of special markers or equipment. The public location means that the program must be able to handle people moving in front of the camera, in the background or even multiple faces present in the webcams field of view while still functioning correctly.

Several commercial solutions already exist that allow the user to control the mouse pointer via webcam. Four of the most well-known are Camera Mouse [8], Face Mouse [13], Head Mouse [10] and IR Tracker [7]. The main focus of most of these products is to allow those with Cerebral Palsy or other conditions that severely limit a person's ability to move to interact with a computer.

All of the existing solutions have short comings that prevent them being appropriate for the application. Camera mouse was found to only run at around 10 fps on the test computer with a large delay which means that moving the mouse is far too inaccurate and slow. Face mouse requires the user to manual select a position on the face to follow and will lose this position if it is obscured or goes off camera. IR Tracker requires an inferred webcam and the user to wear special head gear. Several other methods are presented in papers however all of these have short comings that prevent them being used in this application. Some of the more common problems were low reliability, low accuracy and an inability to handle faces in the background.

II. BACKGROUND INFORMATION

There are three ways that face movement can be used to control a mouse. These are using the translation of the head, using rotation of the head and using the location of the eyes.

A. *Using translation of the head to move the mouse*

The most widely used method of controlling the mouse by moving the face is to locate a unique point on the users face and initially declare the mouse to be at the center of the screen when this point on the face is in the starting location. In each frame the point is located and its position compared to the original. This difference in position is used to move the mouse pointer.

The method via which the face is located and tracked is where most programs differ. A program developed at Carleton University [1] recognizes and tracks the tip of the nose. The nose tip is used as it is visible during almost any head movement the user would perform. A second approach is to use the Lucas-Kanade optical flow algorithm in combination with an algorithm that detects skin color [4] to find easily tractable points that lie on the face [5]. The average location of these points is then used to move the mouse. A third and by far the simplest approach is to track objects that have been added to the face such as a large cardboard square with a pattern on it [6] or special head gear [7]. While this approach is easier to implement it has the obvious drawback of being less practical then the other approaches. One final approach is to have the user place a box over a unique part of their face. From this point the computer compares each frame for the location on the image that best matches the image in this original box. This approach is used in the Camera Mouse application [8] as is shown in figure 1.

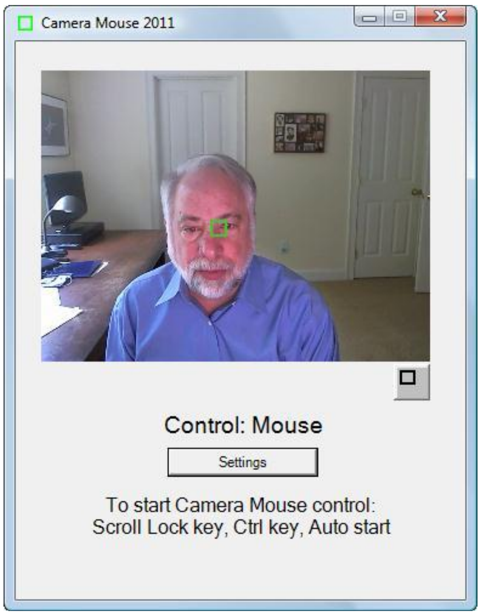
Camera Mouse 2011
Control: Mouse
Settings
To start Camera Mouse control:
Scroll Lock key, Ctrl key, Auto start

Figure 1- Camera Mouse's GUI

B. *Using rotation of the face to move the mouse*

An alternative method to using the faces position to move the mouse is to use the faces orientation with respect to the camera. This approach is much more complex to perform reliably than detecting the faces position though offers the reward of greater accuracy through more intuitive use as the user must simply turn their head to where they wish the mouse to go. This technique is used in a program developed at the University of Moscow [6]. This group used two detectors to determine the angle of the face to the camera. First the Image was analyzed and the head located with a skin detection algorithm. This algorithm gives the head in the same place regardless of rotation. Next the face was located using the Viola-Jones face detection algorithm [9]. Where this algorithm places the face depends on the heads orientation and so a vector between the two locations corresponding to the angle of the face could be created. Some results of this method are shown in figure 2.

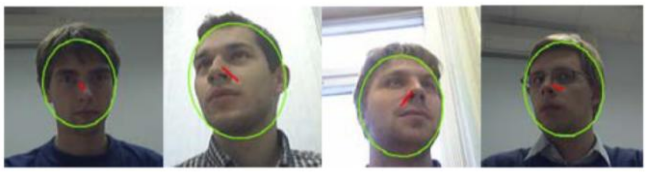

Figure 2- Face orientation detection in operation

A second approach and one that appears to offer the most accurate and robust method of locating the faces position is to compare the face to a pre-existing 3d model of the persons head that the program has using canonical correspondence analysis (CCA) to find the orientation [14][17]. This approach while accurate and robust has the disadvantage of only working for one user.

C. *Using the eyes position to move the mouse*

Detection of the direction the eyes are looking at can also be used to control the mouse. This is often done by illuminating the face with inferred light and then using an inferred camera to detect the light reflected by the pupils [18]. This is used to remove any reflections or lighting conditions that may prevent the pupils from being correctly detected [10]. While the eyes seem a more natural method to use for moving the cursor they have two main draw backs. The first is that detecting the direction of a person's gaze is error prone and computationally intensive, the second is that experiments have shown that a large amount of the eyes movement is controlled subconsciously making exact movements difficult [6].

D. *Relating the face position to the cursor*

All the applications mentioned offer two modes of control of the mouse cursor, position and speed control. In position control the head behaves as an ordinary mouse with position of the head directly corresponding to position of the mouse cursor. This mode allows for very quick movement but has the disadvantage of being far less precise then a normal mouse. This problem with precision at this point is unavoidable as most algorithms are performed on 320 x 240 resolution webcams which must then map the movement onto a display with a resolution of up to 1920x 1200. This means even if the algorithm and user are perfectly accurate the mouse can only be moved to the nearest 6 pixels. In velocity control the head behaves like a joystick with its position controlling the speed the mouse moves in the desired direction. This mode greatly improves precision at the expense of speed of execution. One further disadvantage of this method is that when the mouse is in a screens corner looking at the mouse will cause it to move further into the corner which is unintuitive to a new user. [1] also offered an approach that differed slightly from the above mentioned, it only registered movement away from the origin. This approach mimics the action of lifting up a mouse when the user has run out of drag space on their mouse pad.

E. *Clicking the mouse*

There appears to be no method for clicking the mouse that is in common use. Many methods exist but all of them have clear downfalls and are obviously inferior to a regular mouse. No system is able to give a reliable and easy to use way to perform tasks such as dragging a box to select multiple objects using a webcam alone. Several systems use holding the cursor in place for several seconds to left click [2][10] or having the user close their eyes to click [10]. The most common solution is to use a separate voice recognition system that recognizes words such as click, double and drag to perform the required actions [5] [10] [9].

III. The Proposed Method

From the research performed it was found that most existing methods could be divided into two categories. The first group is methods that require calibration. These methods usually required the user to select a region of the face that is then tracked. When operating these applications run quickly and are highly accurate.

The second group is programs that required no form of setup or calibration. These programs all used either the Viola-Jones method or skin detection methods for face detection. If they use the Viola-Jones method for face detection they perform too slowly on high resolution webcam images to be used for real time applications due to this algorithms large processing time. The methods that used skin detection suffer the large number of false positives and sensitivity to light conditions that are inherent of these methods.

Because of these issues the method proposed aims to combine the strengths of the methods requiring calibration with those that do not. This is done via using two different methods running simultaneously in different threads to locate the faces position. The Viola-Jones method running in one thread places points on the image that are then tracked by the much faster Lucas-Kanade optical flow algorithm in a separate thread. This method effectively increases the speed of the Viola-Jones method while preserving its reliability and accuracy.

A. Detecting the face via the Viola-Jones method

The image is first converted to gray scale and normalized. From this image the face is first located using the Viola-Jones algorithm.

The Viola-Jones algorithm is one of the most accurate and reliable methods for detecting objects in an image. While it can be used to detect any object it is most commonly used for face detection. The algorithm works by overlaying rectangles onto the image [9]. The rectangles used are depicted in figure 3. In these regions the sums of the pixels in the white rectangles is subtracted from those in the gray rectangles. Which rectangles are used and how they are orientated is determined by a classifier that has been trained using a large number of photos of the object to be detected. Due to the large number of possible rectangle locations the classifier is not trained in all of them, instead a cascade of classifiers is used with each detecting only a few features [9]. This greatly increases the algorithms speed. To further increase the algorithms speed it does not work directly on the image but instead works on an integral image. This image has the property that allows rectangular features to be evaluated in constant time. The formulae for the integral image is

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y'),$$

Where ii(x,y) is the coordinates of the point in the integral image and i(x',y') is the coordinates of the point in the original image [9].

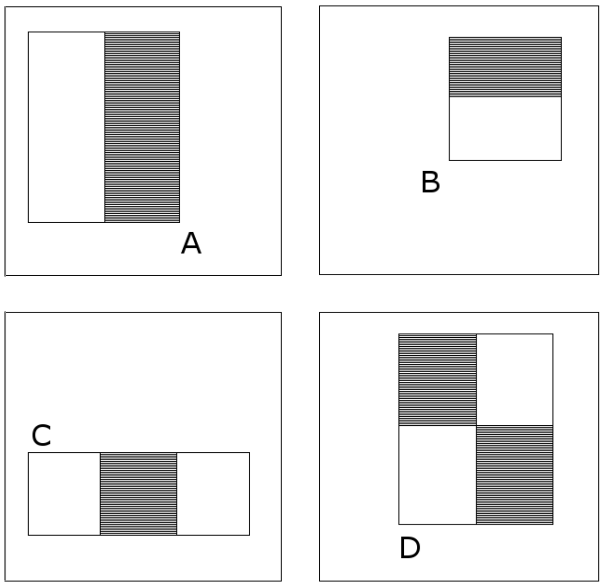

Figure 3- features used by Viola-Jones method

This method has two major drawbacks that prevented it being used directly for mouse movements. The first and most pressing issue is that even on a high end computer the algorithm takes half a second to process a 640x480 pixel image that the webcam provides. The second is that while the rectangle the algorithm places is almost always over the target face its size and exact position will change by a significant amount on each frame when the face is remaining stationary. Both of these problems would combine to give an extremely slow updating mouse that cannot make small movements accurately.

B. Estimating the face location using the Lucas-Kanade method

To overcome the limitations of the Viola-Jones method an intermediate step is used between the face detection and mouse movement. First the Viola-Jones algorithm is set running within its own thread. Every time this thread completes its run it updates a rectangle with the faces location, receives the most recent image from the webcam and begins to run again. The output rectangle is used to place a grid of points on the face as seen in figure 4.

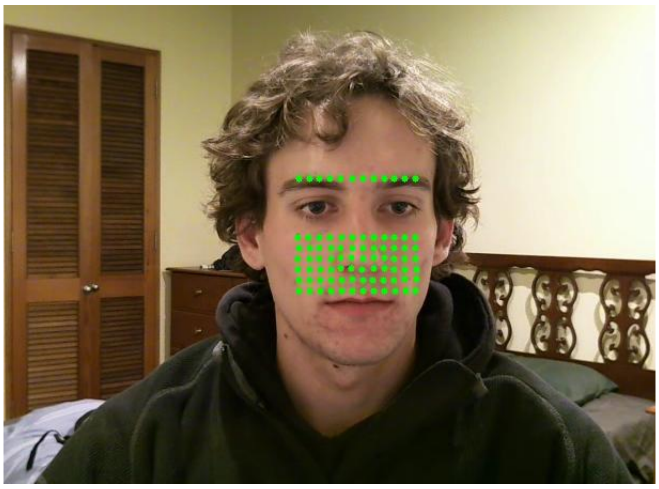

Figure 4- grid of points placed on face

The grid is divided into two sections. Both sections are half the width of the face. The first grid reaches from a quarter of the way down the face to three tenths. The second reaches from two thirds of the way down to three quarters of the way. These regions have been chosen for two reasons. They are almost guaranteed to lie on the face with no points placed on the background and these regions do not lie over the eyes or mouth. The mouth is excluded so that the user can use the mouse accurately and talk at the same time. The eyes were excluded after it was found that while the user was hovering over a button the eyes would unconsciously move to look at other parts of the screen causing the mouse to move. Thus while it appears counter intuitive preventing navigation by using the eyes significantly improved the accuracy that could be obtained with the mouse.

Once these points have been assigned to the face the Lucas Kanade sparse optical flow algorithm is used to estimate their position on successive frames.

The Lucas-Kanade sparse optical flow algorithm is used to track the location of a moving object in successive images [16]. It assumes that the displacement between successive frames is small. If d is the distance a point moves between successive frames then d is the value that minimizes ε as defined in the below equation [15].

$$\varepsilon(\vec{d})=\varepsilon(d_x,d_y)=\sum_{x=u_x-\omega_x}^{u_x+\omega_x}\sum_{y=u_y-\omega_y}^{u_y+\omega_y}[I(x,y)-J(x+d_x,y+d_y)]^2$$

Where
u = original point location
I = image containing point
J = image point must be found for
ωx and ωy = integers

Typical values for ωx and ωy are 1, 2 or 3 pixels

This algorithm works best when tracking corners or other unique features however here it is used to locate the points in the grid which are often placed on featureless areas of skin. Corners are not used as finding them was found to be too slow a process and heavily favored placing points around the mouth and eyes which as already discussed is highly undesirable. As the points are refreshed around every 15 frames while some points do drift this drift usually does not have time to become significant. In cases where it does become large enough to affect the results the point is detected as an outlier and excluded until a new grid is assigned. All the points excluding the outliers' locations are averaged to give the location of the face in the current frame.

To further increase speed only an area twice the size of the head location specified by the Viola-Jones method is searched for the new locations of the points. This method allows for accurate face detection while providing an update rate that is limited by the camera rather then the processing of frames even on high resolution images.

C. From face location to mouse movement

At its heart the algorithm for moving the mouse is extremely simple. The location of the face is compared to an original position and the difference in these positions is set as the offset of the mouse where (0,0) is the center of the screen. All the complications to this method revolve around three problems that had to be overcome for this method to be effective. First is the detecting of these changes amid the substantial amount of noise and false readings produced by the above face location method. The second is allowing the user to quickly and without difficultly change what location is taken to be the original position of the face. The third is an adaptive scaling method to increase the speed of the mouse and greatly increase the precision obtainable with the pointer.

To overcome the problem with noise the position of the reference is adjusted whenever the Viola-Jones method provides a new face location. When this occurs the difference between the previous faces distance from the origin and the new faces distance are compared. The position of the origin is then changed so that these two are equal as shown in figure 5. This prevents the system jumping whenever a new face location is provided. It has the disadvantage that no movement is detected in these frames but as this pause is less than 0.05 seconds in length on the test computer the pauses are not noticeable even if the user is looking for them. Further noise reduction is performed during the optical flow section when if the difference between two frames is above a tolerance the frame is taken as a mistake and disregarded.

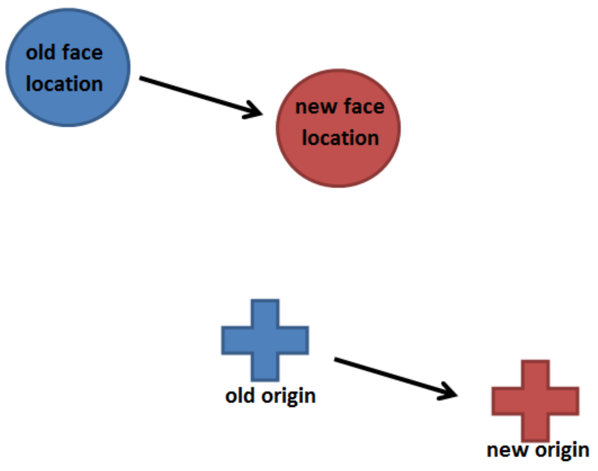

Figure 5- Method for preventing the mouse jumping on receiving a new face

After a while using the system the user will sometimes get into a position where their face must be turned at an uncomfortable angle to move the mouse around the screen or they have run out of travel in their neck while moving somewhere. In this situation the user can quickly move their head back to a center position. The high speed movement is detected and taken as a signal for the system to reset its origin position so that when the user finishes the movement the mouse is in the same place as when they started it. This allows the user to move to a more comfortable position and pan the mouse around the screen. This is similar to how a real mouse can be picked up and brought back to the center of a mouse pad if the edge is reached while the user still wishes to travel in that direction. This process also allows the sensitivity of the mouse to be lower allowing greater precision while still reaching everywhere on the screen.

The largest problem with mouse programs that use face movement is their inability to make small accurate movements of the mouse to click small buttons. Because of this the user is forever hovering all around buttons unable to click on them. Because of this an adaptive sensitivity system was used. This takes the last half second of movements and finds the average number of pixels moved during this time period. This number is then raised to the power of 1.5 and divided by a constant. This scaling factor is then applied to all movements. This means that when the user is moving the mouse across the screen they can pan the entire screen for around 10 degrees of head movement. If the user then slows the mouse pointer down and hovers next to a button for half a second then the same movement will now let them navigate a menu or other similar work where high precision is required. As the change is not instantaneous the user can also see the increasing or decreasing speed as they move allowing control of the mouse to remain intuitive. Taking half a second of movement was used as it was found through trial that this was large enough to allow the user to adjust to the change in sensitivity while still having minimal lag. The power of 1.5 was found through testing as one was found to provide insufficient scaling and a squared relationship caused the mouse to quickly go between effectively stationary and uncontrollably large movements.

D. Clicking the mouse

For this system to be used without a mouse or keyboard it needed the capacity to click the mouse. Many existing solutions use hovering over a location to click. This however only provides the ability to left click and ignores double clicking, right clicking and holding the left mouse button. All these operations are required to operate a normal computer running a windows operating system. The system used by the program builds upon the basic hover concept to allow this required functionality. Once the user has hovered over a location moving less than five pixels for half a second the system plays a sound to indicate the mouse is now in the clicking mode. The user then moves the mouse left to left click, right to right click, up to double click, down to hold the left mouse button or hovers for a further second to resume normal operations without clicking. Upon any of these events occurring the user is prompted with a second sound to show that the instruction has been executed. During the clicking mode resetting of the origin on large movements is disabled to allow the user to flick their head to click as during testing it was found that this was how most users tried to click and they would become confused if this did not register the click.

E. Overcoming limitations of prior research

All the currently existing methods that were examined are constrained in one of three ways.

1) They require calibration before each use. These methods have to be given either the location of the face or a location on the face as well as the tweaking of several variables to set up the enviroment to match the users heads movement.

2) They require a low resolution image. These methods usually make use of the Viola Jones algorithm to find the faces location. With a 640x480 pixel image on a modern cpu this takes roughly 0.5 seconds which means that they either run extremely slowly or use a very low resolution image. This low resolution means that the accuracy of the method is so low they require a custom operating system with extremly large buttons to be usable

3) They are unreliable. These methods use quick face location methods such as looking for skin coloured blobs and assuming the largest is a face. These methods require ideal lighting condtions and often fail to find the face.

The proposed method does not have any of these flaws operating on a 640x480 image at 30 fps (limited by the speed of the camera, 67 fps otherwise), its use of the Viola-Jones algorithm also ensured it is highly reliable.

The proposed method allows left clicking, right clicking, double clicking and holding left click all solely through the webcam. No other method found during research is able to accomplish this. Most methods rely on either keyboard shortcuts, only support left clicks through hover to click or use a separate voice recognition system for clicking.

IV. Results

To generate the results the program was run on a test computer with the following specifications

CPU: i7 875k running at 3.4 GHz
Graphics card: Nvidia GTX470
Screen: Viewsonic 22 inch 1080p
Camera: Logitech Pro 9000 providing 640x480 video at 30 fps
Mouse: Logitech mx510
Operating System: windows 7

The application was written in C++ in Visual studios 2010 making use of the OpenCV computer vision libraries version 2.2 for receiving and processing images. Boost thread libraries[19] were used to set up the threading the program required.

The application gives a frame rate of 30 fps. This is limited by the rate of the camera. If this limit is bypassed using video it runs at 67 fps.

Initially the system was tested for its ability to pick up the users. When using the system by moving the mouse in a diagonal across the screen every 10 seconds for 5 minutes the system was assessed as to if it was picking up the users face or the background. During this test the user was approximately one meter away from the camera. This test was then repeated with a second persons face in the background roughly two meters from the camera. The results are shown in table 1

Test	Correct face	Incorrect	Accuracy
One person	29	1	97%
Background face	24	6	80%

Table 1- correct face detection rates

These results show that when only one person is on screen the algorithm detects and tracks the correct face almost without fail. When people are present in the background however the face detection can occasionally jump to them causing some problems in mouse movements as whenever the mouse jumps person its location is reset to the center of the screen. The false detections almost all occurred while moving the mouse to a corner as this decreased the Viola-Jones methods ability to recognize the face.

To test the speed and accuracy of the program it and several other solutions were used to play a simple flash game to compare their performance. The other solutions tested were a standard computer mouse, a laptop touchpad and the programs camera mouse and head mouse. The game chosen to test the devices on was ellipsis[12] shown in figure 6.

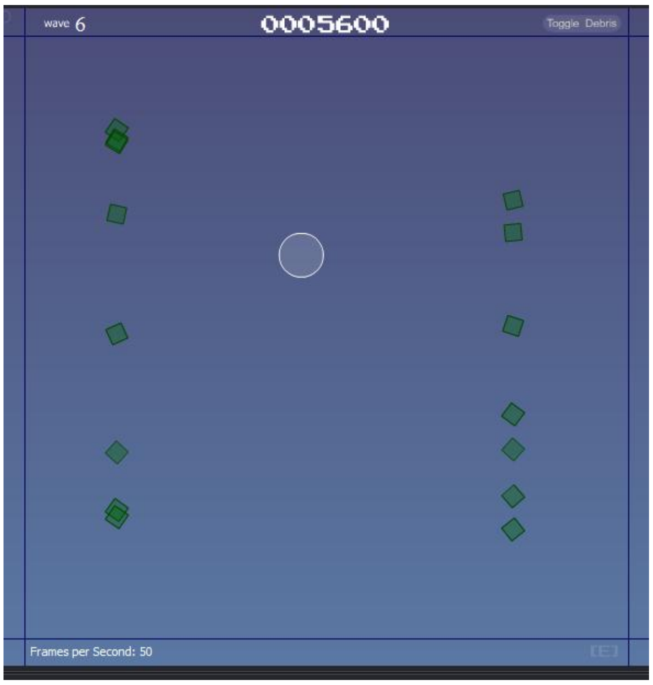

Figure 6- Ellipsis, a game used to evaluate the mouse's performance

In this game waves of squares move from the edge of the screen to a circle located at the center. The player must click on all the squares before they reach this point to move onto the next stage. A score of roughly 250 points is given to the player for each square they click on. This game was selected as the difficulty increased in small steps with each stage meaning that even the worst input method received a non-zero score. The results of this game were also very consistent with most input methods always failing at the same stage as well as providing a clear difference in scores between different inputs. Each input method was tested 3 times and the results are shown in table 2.

Input Method	Run 1	Run 2	Run 3	Average
Mouse	31225	29200	32100	30841
Touchpad	14375	13450	15250	14358
Camera mouse	7475	7475	5875	6941
Head mouse	500	750	500	583
Solution	5225	3975	5300	4833

Table 2- scores from various input methods playing ellipsis

These results shows that while the proposed method is far inferior to a mouse and touchpad its performance is close to that of Camera Mouse a program that requires calibration for each user and by far the most accurate existing solution that could be located. It performance was far superior to Head Mouse a program that also required no calibration.

The test mentioned above only tested the movement of the mouse and not the clicking of the system or its ability to control an operating system. To test this a series of actions were performed using a mouse, a touch pad and the developed solution and the total time taken recorded. The other programs for controlling the mouse could not be tested as they had no means of right clicking or double clicking without additional equipment. The actions were to navigate through twenty folders, open and close five different programs and navigate through a series of ten bookmarked web pages. The results can be seen below in table 3

Input Method	Run 1 (seconds)	Run 2 (seconds)	Run 3 (seconds)	Average
Mouse	73	86	76	78
Touchpad	131	152	185	156
Solution	360	295	341	332

Table 3- time taken to perform series of actions in windows

As can be seen the solution is roughly a quarter the speed of a mouse and half the speed of a touchpad. This shows that while the solution is far slower than a mouse or touchpad it can perform tasks in a reasonable time.

V. Limitations of research

When running the program has a large amount of CPU usage. As the Viola-Jones algorithm is left to run as fast as possible it will use up an entire core on its own regardless of processor. The Lucas Kanade algorithm is limited by the camera but will still use 50% of a core when running on a 3.4 GHz i7 chip. This means that on a dual core system 75% of the CPU time is used up by the program limiting what the computer is capable of.

Currently the mouse always clicks when it hasn't been moved for a set period of time. When using the mouse this quickly becomes frustrating due to having to constantly move the head while reading an article, watching a movie or performing a similar task where extend periods of no mouse movement are required. A solution that was considered for this problem was the development of a more complex gesture such as drawing a square with the mouse to enable or disable clicking.

The mouse has a half second delay in changing sensitivities. This delay means it will usually overshoot whatever someone is wishing to click on as they move to it. Some form of damping may be able to be added to help prevent this issue

The face detection method will lose the face if it is rotated more than around 15 degrees from the center position. This is due to the classifier cascade only being trained on faces looking directly at the camera. This may be overcome by using a classifier cascade that has been trained on faces that are at a large range of angles from the camera.

VI. Conclusion

The design presented in this paper allows accurate movement and clicking of the mouse cursor via movements of the head using a standard webcam. The process of using the Viola-Jones method to calibrate points for the Lucas-Kanade method to track allows the program to operate at a smooth frame rate with a high resolution webcam while still maintaining high accuracy and reliability. This accuracy and speed combined with the adaptive mouse sensitivity implemented in the application offers superior performance to all other tracking methods located that do not require calibration. The gesture system developed for clicking allows full mouse functionality without requiring any additional equipment. The method presented while inferior to a mouse provides a legitimate and practical method for controlling a computer running a windows operating system.

References

1) D. Gorodnichy, S. Malik and G. Roth. Nouse "`Use Your Nose as a Mouse '- a New Technology for Hands-free Games and Interfaces". Proceedings of International Conference on Vision Interface, pp. 354-361, Calgary, 2002.
2) M. Betke, J. Gips, and P. Fleming. "The camera mouse: Visual tracking of body features to provide computer access for people with severe disabilities". In IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 10, No. 1, 2002.
3) R. Feris, T. Campos and R Cesar. "Detection and Tracking of Facial Features in Video Sequences". Lecture Notes in Artificial Intelligence, vol. 1793, pp. 197-206, April 2000
4) V. Vezhnevets. "Method For Localization Of Human Faces In Color-Based Face Detectors And Trackers". In Proc. Third International Conference on Digital Information Processing and Control In Extreme Situations, pp. 51-56, Minsk, Belarus, May 2002.
5) F. Loewenich , F. Maire, "Hands-free mouse-pointer manipulation using motion-tracking and speech recognition", Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces, November 28-30, 2007, Adelaide, Australia
6) V. Vezhnevets, R. Shorgin, A. Vezhnevets. "Mouse Control via User's Head Movements". In Proc. Graphicon-2006, Novosibirsk Akademgorodok, Russia, July 2006 (Translated to English using google docs)
7) IR Tracker http://www.naturalpoint.com/trackir/01-store/store-catalog.html accessed 1/4/2011
8) Camera Mouse, Inc. http://www.cameramouse.com/ accessed 3/4/2011
9) P. Viola, M. Jones, "Robust Real-Time Face Detection", International Journal of Computer Vision, v.57 n.2, p.137-154, May 2004
10) Head Mouse,
http://www.ghacks.net/2009/01/14/computer-mouse-head-control/ accessed 3/4/2011
11) Z. Zhu, K. Fujimura, Q. Ji, "Real-time eye detection and tracking under various light conditions", Proceedings of the 2002 symposium on Eye tracking research & applications, March 25-27, 2002, New Orleans, Louisiana
12) Elipsis
http://www.newgrounds.com/portal/view/339352 accessed 9/4/2011
13) Face mouse http://www.oskaworld.com/category/free-face-mouse.php accessed 3/4/2011
14) W. Yang, D. Yi, Z. Lei, J. Sang, S. Li "2D-3D Face Matching using CCA", In: Proc. IEEE International Conference on Automatic Face and Gesture Recognition, Amsterdam, The Netherlands, September 17-19 2008
15) W. Fernando, L. Udawatta, and P. Pathirana, "Identification of Moving Obstacles with Pyramidal Lucas Kanade Optical Flow and k means Clustering", International Conference on Information and Automation Sustainability, pp111-117, 2007
16) B. Lucas, and T. Kanade. An iterative image registration technique with an application to stereo vision.Proc. DARPA Image Understanding Workshop, pp. 121-130, 1981.
17) L. Terissi and J. G´omez, "3d head pose and facial expression
tracking using a single camera," Journal of Universal Computer Science, vol. 16, no. 6, pp. 903–920, 2010.
18) H. Kubota, M. Takeshi and H. Saito, 3D head pose tracking using a particle filter with nose template matching. Electronics and Communications in Japan, 2011
19) Boost threads
http://www.boost.org/doc/libs/1_46_1/doc/html/thread.html accessed 20/5/2011