Working with a computer hands-free using *N*ouse® Perceptual Vision Interface

Dmitry O. Gorodnichy, Elan Dubrofsky, Mohammad A. Ali

Institute for Information Technology (IIT-ITI)
National Research Council of Canada (NRC-CNRC)
Montreal Rd, M-50, Ottawa, Canada K1A 0R6
http://vrs.iit.nrc.ca/Nouse

Abstract

Normal work with a computer implies being able to perform the following three computer control tasks: 1) pointing , 2) clicking, and 3) typing. Many attempts have been made to make it possible to perform these tasks hands-free using a video image of the user as input. Nevertheless, rehabilitation center practitioners agree that no marketable solution making vision-based hands-free computer control a commonplace reality for disabled users has been produced as of yet. as reported by rehabilitation center practitioners, no marketable solution making vision-based hands-free computer control a commonplace reality for disabled users has been produced as of yet.

Here we present the Nouse Perceptual Vision Interface (Nouse PVI) that is hoped to finally offer a solution to a long-awaited dream of many disabled users. Evolved from the original Nouse "Nose as Mouse" concept and currently under testing with EBRI [1], Nouse PVI has several unique features that make it preferable to other hands-free vision-based computer input alternatives.

First, its original idea of using the nose tip as a single reference point to control a computer has been confirmed to be very convenient for disabled users. For them the nose literally becomes a new "finger" which they can use to write words, move a cursor on screen, click or type. Being able to track the nose tip with subpixel precision within a wide range of head motion, makes performing all control tasks possible.

Its second main feature is a feedback-providing mechanism that is implemented using a concept of Perceptual Nouse Cursor (Nousor) which creates an invisible link between the computer and user and which is very important for control as it allows the user to adjust his/her head motion so that the computer can better interpret them.

Finally, there are a number of design solutions related specifically tailored for vision-based data entry using small range head motion such as motion codes (NouseCode), a motion-based virtual keyboard (NouseBoard and NousePad) and a word-by-word letter drawing tool (NouseChalk).

While presenting the demonstrations of these innovative tools, we also address the issue of the user's ability and readiness to work with a computer in the brand new way – i.e. hands-free. The problem is that a user has to understand that it is not entirely the responsibility of a computer to understand what one wants, but it's also the responsibility of the user to make sure that the computer understands what the user motions mean. – Just as a conventional computer user cannot move the cursor on the screen without first putting his or her hand on the mouse, a perceptual interface user cannot work with a computer until he or she "connects" to it. That is, the computer and user must work as a team for the best control results to be achieved. This presentation therefore is designed to serve both as a guide to those developing vision-based input devices and as a tutorial for those who will be using them.

1 Introduction

In the area of vision based cursor control, several techniques have been proposed to convert face position to a cursor position using a video camera. Several commercial systemsas well as evaluation software have been created for the purpose. Table 1 lists software available for download as of August 2003. Using switch-button-

[1] Elisabeth Bruyere Research Institute of SCO Health Service.

ON-LINE *Confira endereços com informações, depoimentos e softwares para deficientes físicos na rede:*

NOUSE	www.cv.iit.nrc.ca/research/Nouse/download.html	MICROSOFT	www.microsoft.com/enable
DOSVOX	http://intervox.nce.ufrj.br/dosvox	HP	www.hp.com.br/acessibilidade/soft-def.html
MOTRIX	http://intervox.nce.ufrj.br/motrix	APPLE	www.apple.com/disability
REDE SACI	www.saci.org.br	TRACE CENTER	http://trace.wisc.edu/world/computer-access/multi/sharewar.htm
DEFNET	www.defnet.org.br		
ABRAS	www.acessibilidade.org.br/softwares.htm	KOLLER	www.koller.com.br/produtos.html
VIRTUAL VISION	www.micropower.com.br	LENIRA LUNA	http://intervox.nce. ufrj.br/~lenira
JAWS	www.hj.com	INSTITUTO BENJAMIN CONSTANT	www.ibcnet.org.br
IBM	www-3.ibm.com/able	ELIZABET SÁ	www.lerparaver.com/bancodeescola

Table 1. Available vision-based cursor control software (from "Planeta Digital", August 2003)

based and tracking of the eyes with IR cameras for cursor control were also proposed.

Similarly, several techniques and systems have been proposed for vision-based click replacement, most successful of which are IR based. Opening of the mouth was also suggested. Yet still, according to Occupational Therapy practitioners, inside an ordinary rehabilitation center where many disabled residents are in need of such technology, none yet has been found successful. The reason is that, while tackling one or a few data input tasks, none of them offers a complete hands-free solution for all user needs, which includes three main input tasks, and which would adjustable (and ideally automatically adjustable) to all users, all of whom have different motional abilities and constrains.

In the following we present Nouse-PVI technology, which is developed with a feedback from SCO Health Service Occupational Technologists and which is hoped to answers their needs. It is understood however that the concepts, solutions and designs developed for Nouse-PVI will also be applicable to many other vision-based interfaces such as those listed above.

This paper therefore addresses both the needs of Nouse-PVI users and those of PVI designers. While the former may benefit more from Sections 2 and 4, and the latter from Sections 3 and 4, it is our intent to present all sections together to foster mutual collaboration among users and designers.

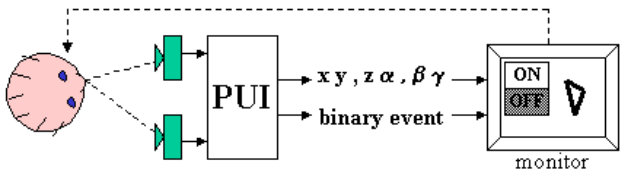

Figure 1. Vision-based control diagram.

2 Critical concepts for users

To efficiently operate Perceptual Vision Interface such Nouse-PVI, one needs to familiarize oneself with several important concepts and results from the area of Video Processing and Recognition.

2.1 What One Needs to Know About Video

The complexity of the video recognition is often under-estimated. Unless cleared out, this under-estimation may create problems for the users of video recognition systems, as it can lead to ungrounded higher expectations, lack of cooperation from the user, and worse performance of the system as result. Therefore it is important complexity issue be clarified and rectified.

Why does this perception exist? - For two reasons. Firstly, while most computer technologies aim to solve problems that people cannot do by themselves (consider 3D reconstruction or data mining), the goal of video recognition technology is to do what humans can do easily. - A complex system is required for a computer to track multiple objects in a scene while even an uneducated child can do so with great ease. This feeling leads to a belief that video recognition is nothing special and should be easy and this can lead to disappointment later on. A similar disappoint and sagging of interest to artificial intelligence in 1960s should be reminded and serve as lesson to those working on or using video recognition technologies.

Secondly, the problem of video recognition is also much more constrained than one would think. When a person looks at a scene, he or she immediately knows where to look and for how long the focus is needed at a specific point. Humans also use extra knowledge such as memory and imagination in order to help recognize objects. A computer cannot easily do this, and to add to the problem, the computers are usually only given 320 x 240 numbers to process at a time. The term for this type of input is "low resolution video". Combine this with the fact that we are not given temporal or spatial knowledge to base any calculations on. When a person walks into a room and surveys the scene in front of them, they are armed with the information of what room they went into in the first place. The computer will most often not be armed with this information. One other constraint is the fact that video recognition technology needs to work in the real world, where lighting and angles are far from perfect. This point can be easily understood by comparing the quality of a home movie to a Hollywood production where people exert

lots of effort to make sure the lighting is perfect.

These obstacles relate to all video processing applications. The application of video recognition for control however has its own complexities and constraints in addition to the ones already mentioned.

2.2 What one needs to know about video-based hands-free computer control

There are many issues that make the programming of video-based hands-free computer control systems harder then one would think. The basic goal of the system is to see user motions and convert them into computer commands so that the user can move the cursor, click and type. While the average user will have approximately 100 millimeters of head motion, which may be seen as 40 pixels in the video image. (Recall that the video image will at most be 320 by 240 pixels). A mapping needs to be established that converts from the world space (Xi, Yi, Zi) head points to the two dimensional cursor space (i,j). This mapping needs to be continuously monotonically increasing in all directions. What this basically means is that whenever the user moves whatever modality is being tracked, the cursor needs to move in the same direction. By modality we refer to feature(s) of the image that is being analyzed (examples are colour, motion, corners, etc.) So if the program is tracking the nose and the user moves it up and to the left, the cursor needs to do the same. While this seems like an obvious observation, this continuous mapping function is not achievable for most of the modalities that are used.

For clicking and typing, the main thing to consider is that the user should not ever perform one of these actions unless they specifically choose to do so. Consider the problem of clicking. Once the user moves the cursor to the intended place, how can he indicate that he wants to click? Of course if the user has a certain mobility then he can press a button or voice a command but if he can only move his face the problem becomes more difficult. Also with clicking, there is the issue of what type of click the user wants to perform. It can be a single click, double click, right click or maybe something else.

As far as typing goes, we need to not only find a way for the user to activate typing mode, but also a way to ensure that all of the letters and characters are easily accessible. Our system would be very inconvenient if it took the user over 20 seconds to find and type each letter.

Yet another constraint is the fact that all users are different. Each one will have different motion ranges and abilities. With some we may be able to recognize

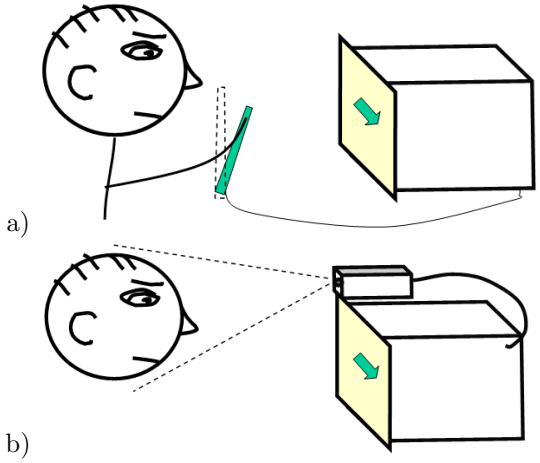

Figure 2. When we hold a mouse/joystick to move the cursor (a), we are constantly aware of where the mouse/joystick is, which makes moving a cursor with a mouse/joystick natural, whereas when we move a cursor remotely, as with a video camera that tracks the person's face position (b), the absence of the constant awareness of how our motion is perceived by the camera makes such a remote cursor control difficult, since the camera may not and often does not perceive our motion the way we think it does, due to light changes, changes to the relative to camera user position, or the deficiency of a vision detection algorithm.

blinks but not with all of them (consider users wearing eye glasses). We need a way to calibrate the system for each user that wants to use it. Once the calibration is done, it would be nice if the system could remember the user the next time he showed up so he wouldn't have to go through it again. This is a face recognition problem and there are many papers devoted to recognize faces in video. Also the system will need to work for various lightings, setups and cameras. Is the user close or far from the camera? Are there shadows? All of these things add to the complexity of vision-based computer control.

3 Feedback-providing Cursor – critical concept for both users and designers

3.1 On importance of feedback for control

Despite the significant advances recently made in vision-based face tracking, of which the techniques developed for automatic detection of faces [19] and eyes[14], 3D-face-model-based tracking [16] and robust

sub-pixel precision nose tracking [12, 13] should be specially acknowledged, the desired hands-free face-tracking-based cursor control has not been achieved yet.

The problem is that, while making it possible to receive user commands remotely, PUIs introduce one major problem — the absence of "touch" (also referred to as feedback connection) with the cursor. This problem, illustrated in Figure 2, has been mentioned back in 2002 in one of the pioneering works on robust vision-based interfaces [13], where we find the following quote: *"One has to realize that, unlike hands-operated interfaces, hands-free interfaces do not have a feedback connection. By holding a mouse, a user not only controls the program, but s/he also keeps the knowledge of where the mouse is. No matter how robust the perceptual user interface is, it can lose the user; it might be even more appropriate to say that a user loses the interface".*

Indeed, when observed by a camera, a user may only guess (and hope) that the camera and computer perceive his or her commands as he or she intents. But, in fact, due to either light changes, changes in relative camera-user position, or simply because of the deficiency of a vision detection algorithm, the user's face motion may and often does get recognized incorrectly. Such a lack of constant awareness of how our motion is perceived by a computer makes remote cursor control difficult. It can be even further shown the following (see Figure 3).

Proposition 1: *What a user needs in order to control a cursor is not a more precise or robust motion detection technique, but a clear real-time closed-loop feedback connection that would allow a user to learn the mapping between the coordinates of the body part controlling the cursor as perceived by the computer and the coordinates of the cursor on a screen.*

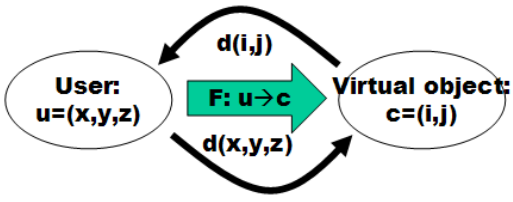

Figure 3. It is the feedback loop that allows a user to learn the motion needed to control a cursor, as the mapping from one coordinate

3.2 Status quo: broken feedback loop

As we examine vision-based cursor control systems available on the market [1, 2, 3, 4, 5] or research literature [18, 8, 9, 10, 7, 17, 11, 13, 16], we note that in order to provide a user with the knowledge on how the face is detected by the camera, these PUIs use a separate window somewhere on a screen in addition to normal cursor, which shows the capture video image of face with the results on vision detection overlaid on top of it. This is how the visual feedback is provided for CameraMouse [1], Quilieye [4] , IBM HeadTracker, which are the examples of the commercially sold vision-based computer control programs that have been tested in SCO Health Center

The drawback of this visual feedback is that the user has to look both at the cursor (to know where/how to move it, e.g. to open a Windows menu) and at the image showing the results captured by the videocamera (to know how to move his head in order to achieve the desired cursor motion). Since a user cannot view two different locations at the same time, this creates a problem for the user. Furthermore, having an additional window produces another problem. – It occludes other window applications, making the windows desktop more cluttered and less organized.

This problem is resolved by introducing a new concept, called Perceptual Cursor [15], which serves both the purpose of marking a position (as normal cursor) and the purpose of providing a user with the feedback on how remote user motions are perceived by a sensor. As such, Perceptual Cursor does not replace the regular cursor, but rather is used in the interface in addition to it, taking its functionality only when requested by the user.

3.3 Implementation

Definition: *Perceptual Cursor is a cursor that comprises a sensing feedback information.*

In other words, Perceptual Cursor consists in both a visual representation of the sensing video signal and a pointing object, and is used both for pointing and feedback.

The Perceptual Cursor could take on many looks, it could vary in colors, e.g. green when it is in clicking mode; red when it is no longer tracking; or different size to indicate when the user is moving towards or away from the camera; or different shapes to show other desired feedback features to the user.

When implementing Perceptual Cursor, the following conditions have to be observed.

Condition 1: Perceptual Cursor should be visible at

all times [2] and its appearance should be updated in real-time so that the information obtained from the video-processing program can be shown to a user with no delay.

Condition 2: The operation of Perceptual Cursor should not affect or intervene the operation of other window/user interface components, including the normal cursor.

Condition 3: It should not conspicuous (i.e. not taking undesired attention of the user), yet well visible when desired (i.e. activated by user).

To meet these conditions, the following procedure can be used.

Implementation Procedure:

Perceptual Cursor can be implemented as
- a Windows window of a small size - made visible at all times, i.e. constantly redrawn in Windows on top of all other windows as the highest priority window,
- with a corner (or middle, or any other highlighted part) of it used for pointing, and the rest of it showing the desired feedback information.

Operating the normal cursor hands-free using the Perceptual Cursor becomes natural with the following control procedure.

Control Procedure:
- At every time instance, the location and the appearance of the Perceptual Cursor corresponds to the remote input data registered by a sensor.
- In order to transfer control from Perceptual Cursor to the normal cursor , as set of facial motions is defined (e.g. facial events "Double-Blink", mouth opening, moving nose outside of the box, etc.).
- Of these facial motions, there are at least two that serve the purpose of a) moving the actual cursor to the current location of the Perceptual Cursor, and b) clicking the actual cursor in its current location.

The choice of different colours or shapes for the perceptual cursor appearance as a function of the current data sensing state is a matter of a design, which ideally should be intuitive and ergonomic. In Nouse PVI, Perceptual Cursor (referred to as Nouse Cursor or Nousor) is implemented as small "flying" image, one corner of which is highlighted and used to point, and the inside part of which shows a visual feedback about the current tracking/control status as shown in Figures 4 and described in next sessions.

Proposition: *The ability of the user to work with Nouse will very much depend on how the user can make use of the feedback provided by the Nousor.*

[2] More exactly, at all times when desired by user, as user may wish to switch on and off.

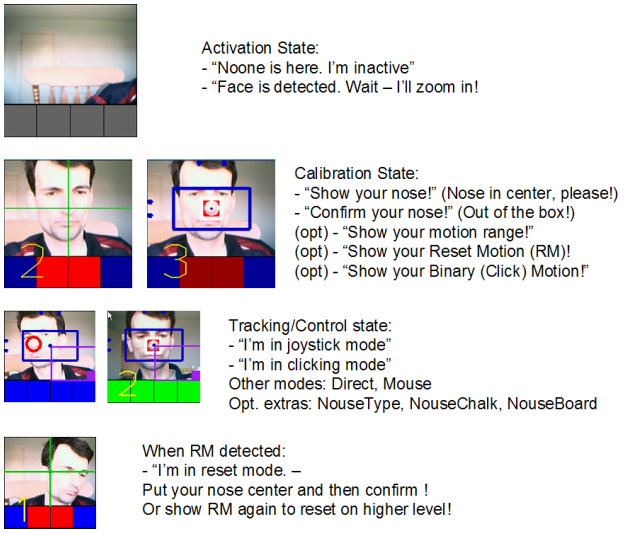

Figure 4. The appearances of Nousor (Visual feedback providing cursor) for different states.

4 Disability variability study information for designers

Within a NRC-EBRI Nouse Pilot Project Phase I, experiments have been conducted within EBRI in order to obtain the information related to the variability of facial motions in patients suffering from dextrous immobility.

Following approval of the Research Ethics Board, the occupational therapist and assistive technologist reviewed their clients from the Complex Continuing Care program to identify potentially eligible participants. In total, 15 individuals consented to be involved in the pilot study. Inclusion criteria for participation included: limited upper body movement and residency in the Complex Continuing Care Program. Participants were recruited based on medical diagnoses that could limit mobility such as Multiple Sclerosis (including Friedreich's ataxia), Cerebral Vascular Accident (stroke), Parkinson's disease, Traumatic Brain Injury and Amyotrophic Lateral Sclerosis (ALS). Participants were asked to carry out the movements necessary to activate the current version of the Nouse: eye blinks and specific neck movements: flexion, extension, and left and right rotation, all performed within the participants' functional range of motion. The students observed these movements and noted when the participants could perform them intentionally (i.e., on command). The data pertaining to the participants' perfor-

mance and environment were documented; including: approximate range of motion, quality of movements (smoothness, speed), ability to control movements, fatigue, comfort and apparent motivation, and a description of the physical environment (including room lighting, background distractions, mobility devices and time of day).

Table 2 taken from [6] summarizes the obtained findings. The importance of the data presented in this table for the developers of face-based control systems should not be undervalued, as it shows the range of conditions and constraints the systems should be able to deal with. Specifically, the designers often overestimate the range and the smoothness of head motion a handicapped user is able to perform.

When developing Nouse-PVI all these findings have been taken into account.

5 Important design concepts

Here we briefly summarize other design solutions that have been proved important for designing hands-free input devices (PVI). These solutions, called below as principles, have been found or developed through an elaborate investigation on how to minimize the occurrence of unintentionally invoked commands while at the same minimizing non-occurrence of intentionally called commands and making such perceptual devices easy to learn and to use. All of these solutions have been incorporated in Nouse-PVI.

- **Point of control Principle**: *Precise* (hands-free, in particular face-operated, is more intuitive and easier to use if it is conceived (as by the designers) and thought (in the users head) as a control by an actual point on a face, which we refer to as Point of Control or Reference Point (CP, $\vec{P} \in [0, 320]$). It may not matter to the user whether this point, measured in image pixel coordinates, is really detected or concluded from other measurable by the program data.
  – Ideally, the Reference Point is such that it provides the user with the largest range of motion and the flexibility of motion.
  – Nose tip is a perfect point to be used as Reference Point, since, due to its protruding location, it is located the furthest from all three axis of the head rotation.

- **Range of motion and relative RP:** For each user, the Range of Motion (RoM), which is defined as the area reachable by the RP during the control, must be computed. – The relative position of RP (RCP, $\vec{p} \in [-1, +1]$) measured with respect to the RoM boundary is to be used to activate PVI, as well as for many control actions such as entering Motion Codes or typing letters using Motion-based NouseBoard.

- **Rest and activation Principle:** By default, PVI is always inactive. Default position is where user rests most of the time. We call it Rest or Zero position. The activation is always initiated by user by moving outside of rest area or by moving to a predefined position (e.g. to the corners).
  – Rest (inactive) area is intuitively felt by the user (either via a Nousor or memorized by moto-reflexes)
  – Two shapes of Rest area are possible: a) box shape, b) cross-shape. We argue in the preference of cross-shape rest area, as it provides larger safety buffers, decreasing a chance of involuntary activation.

- Ideally there is buffer (or hysteresis) between the rest area and the activation areas, which are the areas where a user needs to move to in order to activate PVI.

- After no-activity period, PVI resets to inactive state.

- **Bi-modal (Head-nose) processing**:

  It is the motion of your nose tip that is used for control. Nose tip is your Point of Control (CP) Other measured data (motions) of your head are used for:

  Reset command

  Bi-modal (Head-nose) calibration

- **Reset Principle**: User should be always able to reset tracking using a predefined motion. Not to confuse this motion with moving control motion, motion video modality can be used.

- **Moving vs Binary facial motion Principle**: Different video modalities for different tasks: video modality should be for tracking, and another different video modality for binary control (reset, click).

- **Auto-zoom Principle:** PVI should be able to zoom automatically on a user face as well as to automatically provide reasonably good estimates of the position of the users reference point (such as nose tip) and range of motion. As expressed by EBRI, this is critical for handicapped users.

  .

- **"Confirm always" Principle:** No control action is done without user's intentional action. Intentional actions can be easily identified by the confirmation actions that are performed by a user in synchronization with computer requests shown to user via Nousor.

- Synchronization is achieved using the countdown shown in Nousor that shows how many seconds are left for user to perform (or to confirm) an action. If binary facial motion, which is the motion when nose remain on the same spot such as eye blinks or mouth opening, is detectable, than it can be used instead of countdown.

- **Multi-gear control Principle**: Once activated, control always starts from crude control mode and then it switches to fine control mode.
  – Nouse-PVI always starts from joystick (or direct) mode and then automatically switched to mouse (pixel-by-pixel) mode.

- **"Constant user awareness" Principle**: user must always know which control state and mode he's in, so that he can act accordingly.

- **Motion-activated clicking** is preferred to dwell clicking. This not only resolves the problem of unintentional clicking but also provides solution to invoking different types of clicks.
  – Nouse-PVI implements clicking as shown in Figure 5. - Once prompted, the user has to move outside of the White Box: going left indicates left click etc, while staying inside the box indicates no intention to click.

- **Face-motion tailored communication (NouseChalk)**: There is no need to duplicate hands-based communication for people who can't use hands. Instead, tools have to be developed for allowing users to communicate in way more that is more natural to them. E.g. writing words with a nose as with a chalk might more natural that using it to type the word's letters on on-screen keyboard. NouseChalk allows a user to do that.

- **Face-motion tailored key and command entry (NouseCodes)**: Another way of entering keys and commands that can be found more natural for hands-free operation is by typing (drawing) their codes in the air as done in NouseCodes shown in Figure 6.

- **Face-motion tailored virtual keyboard (NouseBoard)**: If on-screen keyboard has to be used for nose-operated key entry, then a) the size and layout of it should mapped to the nose motion range so as to making all parts of the board easily accessible, b) four-direction-motion-based click provides an efficient way to select a letter in a group of four. c) Board or typing should be in inactive state by default. To activate, user need to leave the rest area (or reach the activation perimeter or motion range cells).
  – This is implemented in NouseBoard shown in Figure 8 that has user's range of motion divided in blocks containing four letters each.
  – In addition, **"Lock on Area"** motion code allows Nousor to lock on any part of the screen so that Nouse can be used with any commercial onscreen keyboard.

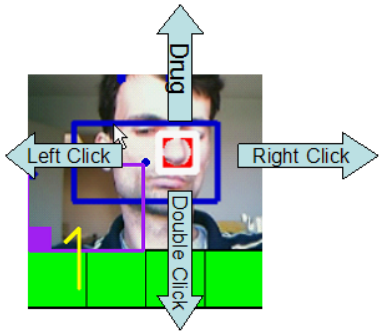

Figure 5. Performing different types of click using nose tracking

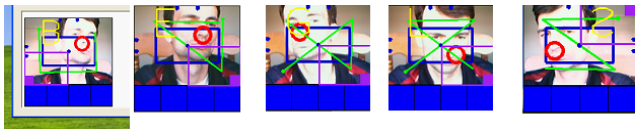

Figure 6. Some motion codes of those programmed in Nouse. More can be added using a script file, using either 4x4 or 9x9 grid.

6 Nouse-PVI software

6.1 Summary of features

Nouse-PVI is a Perceptual Vision Interface program that offers a complete solution to working with a computer in Microsoft Windows OS hands-free. Using

a camera connected to a computer, the program analyzes the motion of the user to allow him/her to use it instead of a mouse and a keyboard. The program is most suited for operating with the nose (hence the name the Nouse abbreviated from nose as mouse). It however may also be used - upon some adjustment in program settings - for hands-free computer control using any other part of the user's body or object s/he is moving. As such Nouse-PVI allows a user, to perform the basic three computer-control actions:
– i) cursor control: includes a) cursor positioning, b) cursor moving, and c) object drugging - which are normally performed using mouse motion
– ii) clicking: includes a) right-button click, b) left-button click, c) double-click, and d) holding the button down - which are normally performed using the mouse buttons
– iii) key/letter entry: includes a) typing of English letters, b) switching from capital to small letters, and to functional keys, c) entering basic MS Windows functional keys as well as Nouse functional keys - which would normally be performed using a keyboard.
The program is equipped with such features as:
- i) automatic detection of the user and his/her range of motion,
- ii) Nousor (Nouse Cursor), which is the video-feedback-providing cursor that is used to point while providing the feeling of "being in touch" with a computer.
iii) NouseBoard, which is a virtual on-screen keyboard that specially designed for face-motion-based typing, and that automatically adjusts to the user's facial motion range.
iv) NousePad that provides an easy way of typing and storing messages hands-free using face motion.
v) NouseTyper (NouseCode) that provides an easy way of switching between Nouse modes and states, and can also be used for typing the characters using a 4x4 or 9x9 postcode-like letter layout.

System requirements:

- Processor: 800MHz (minimum), Pentium IV 1 GHz or higher (recommended)
- OS: MS Windows 2000 or XP
- RAM: 1 Gb or 2 Gb (recommended)
- HD space: 20Mb - for ACE-Surveillance installation with remote archival; (optional) 1Gb per week for archival of data on local hard-drive;(optional) 100Mb Apache-based local ACE-Browser installation,
- Installed USB or USB2 camera(s), video digitizer(s) or frame-grabber(s)

Required 3rd party libraries (included in installer):

- Microsoft DirectX 8 or later SDK [required for camera functionality]: From www.microsoft.com/directx/
- Intel OpenCV (beta 4) Library for Windows [required for image capture and processing functionality] From http://sourceforge.net/projects/opencvlibrary

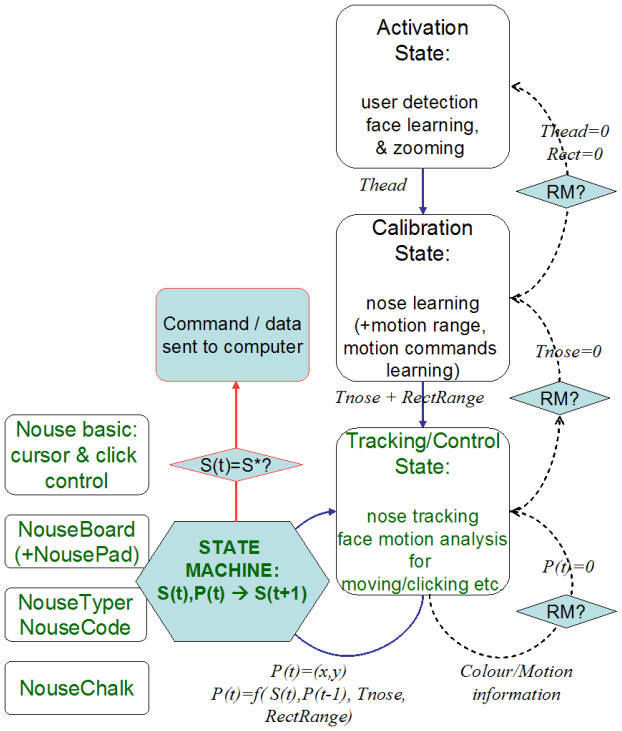

Figure 7. Nouse-PVI states and transitions

6.2 Main Nouse states and substates

Nouse needs to be launched manually by pressing Nouse-PVI icon on desktop. After that computer can be controlled entirely hand-free using the Nouse. The actual hands-free control however commences only when a user's face is detected in the camera's field of view. Until this happens, Nouse remains in inactive ("sleep"). Nouse-PVI has three main states: Activation, Calibration and Tracking/Control, with state transition diagram as shown in Figure 7. The Nousor

appearances corresponding to each state and substate (mode) along with their meanings are shown in Figure 4

In Activation state, Nouse performs head-calibration, in which face location is computed and the estimate of nose position and motion range are obtained. Activation Nousor is located in the corner of the screen, with all buttons grey.

In Calibration states, user is requested to perform nose calibration by putting his nose tip in the position estimated in Activation state (indicated by cross-hair in the Nousor). When satisfied with calibration, the user confirms it by moving the nose outside of the box, when the box appears. When Calibration state is run for the first time, the user is directed after nose calibration straight to range calibration, where the user is requested to expand a blue box by "pushing it" with a nose. When satisfied, the user quits the state by staying still for a second [3]. Calibration Nousor is blown to is maximum size and centered in the screen.

In Tracking/Control state, is where all computer control and data input tasks are executed. Specifically, based on computed in Calibration state nose attributes (T), motion range *Rect* and the current video image (I), Nouse-PVI computes a two-dimensional relative number ($\vec{p} = (x, y), x, y \in \{-1; +1\}, \vec{p} = 0$ in center(zero) point), which is passed to a state machine which decides what to do with this number. Most of the time, particularly when a user is familiar with the program and is focused on controlling a computer, this number will correspond to the position of the nose tip. When a user is not focused on computer control, is not familiar with the way the Nouse works, or simply turns away from camera, this number $\vec{p}$ may correspond to any other point in the observed image.

Two main control sub-states determine what to do with this number. These sub-state are "moving" sub-state (when a point moves) and "clicking" sub-state (when a point does not move). In mouse replacement mode (which is the basic and the main Nouse mode), these two substates move a cursor and perform click. In other modes, they are used for drawing or selecting inside the range of motion and as a confirmation/selection action, respectively.

6.3 "Clicking" motion

A user always knows when s/he is in "clicking" state, because in this state Nousor puts a WhiteBox on top of the nose, and starts countdown (Nousor buttons change their colour). By doing nothing in this state, a user ensures that nothing is executed by a computer. If a user want to execute something, s/he has move out of the box, as shown in Figure **??**. - Depending on which direction user goes out of the box, a different type of click (or confirmation) is executed.

[3] or (in some versions) by performing "click" nose motion described later

6.4 Nouse Tools

A number of data-input tools are incorporated in Nouse-PVI that make use of nose tracking number $\vec{p}$ computed by the program. These are NouseBoard, NouseTyper and NouseChalk, described above. In NouseBoard, $\vec{p}$ is converted to a cell position on virtual screen keyboard the layout of which is mapped onto the motion range. One design of which is shown in Figure 8. In NouseTyper (also called NouseCodes), $\vec{p}$ is converted to Motion Code or ASCII character, when $\vec{p}$ passes through the predefined cells of the motion range (such as corners - for entering motion commands using 2x2 rectangle layout, and corners and middle points - for typing 3x3 characters). Some Motion COdes are shown for a 2x2 code layout in Figure 6. In NouseChalk (also called called NousePaint), $\vec{p}$ is associated with a chalk that writes on blackboard that is mapped to the motion range.

In each tool, "clicking" sub-state is used to commence or terminate a command or an entry.

For more details on each tool and control procedure, refer to the Appendix.

Switching between modes and tools is accomplished using the motion codes or by pushing buttons in Nouse gui.

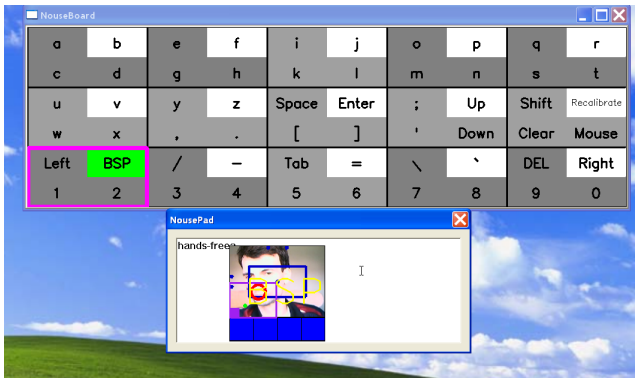

Figure 8. The appearances of NouseBoard: grouping of letters by four is made to suit four directions of "clicking" motion.

Conclusion

A number of tracking-based control principles and tools to make hands-free computer control more efficient is presented. Nouse-PVI software that is built on those principles and that includes and presented tools, is presented. Phase II of Nouse-PVI testing with EBRI is currently in place. An evaluation version of the software to be downloadable from our website is planned for release soon.

Acknowledgement

Perceptual Vision Interface Nouse is developed under the consultation with the EBRI and SCO. The testing of NousePVI software by SCO personnel is acknowledged. The feedback from SCO, in particular from Bocar N'giyae, is greatly appreciated.

® Nouse is a registered trademark of the National Research Council of Canada.

Appendix. Nouse-PVI control procedures (from user guide)

"Moving" procedure

Crude navigation. - Move the cursor to a point on the screen as with a joystick (default).

When all of the buttons are blue, this means that you are in a general mouse movement state. The nouse cursor works as a joystick. If your nose is in the center, the nouse cursor will not move. Then as you get farther away from the center in any direction, the nouse cursor will move proportionally faster is that direction. You can use the mode to place the cursor approximately near where you'll want to click. There is no need to be exact. Once you are satisfied, move your nose back to the center of the cross-hair and then wait for the buttons to change to green (or yellow if fine control is enabled).

Fine navigation(optional). - Move the cursor as by pushing a mouse.

When the buttons are all yellow this means that we are in the precise movement state. This state can be analogized to moving an actual mouse cursor when the mouse pad is very small. In order to move to the right for example, you must nudge your nose to the right and then move back to center and repeat this process. The mode is used to place the cursor exactly where you would like to click. Once you are satisfied, stay still and wait for the the cursor to switch to clicking mode.

It is important to note that when placing the cursor you should make note of which corner of the nouse cursor will be used to make the click. This is indicated by purple lines protruding from the clicking corner. If you wish to change the clicking corner, enter the correct code as described later.

"Clicking" procedure

Now that the cursor has been placed, you move into the final state; that of clicking. This is indicated by all of the buttons being coloured green. A countdown starting at 3(default) will count down and to click all you need to do is move. After you click, you will go back to crude navigation

Performing different types of Clicking

a. Choose not to click. -If you don't move, no click will be made and you will go back to the general mouse movement state.

b. Do a left click - To do a left click, move down and to the left.

c. Do a right click - To do a right click, move down and to the right.

d. Do a double click. - To do a double click, move up and to the left.

e. Drag an item. - If you select drag, the will emulate pressing down on the left mouse button and holding it down. Once you select drag, a 'D' will be drawn in the nouse cursor. When you choose you next place to click, the action will be to release the mouse no matter what direction you move.

Entering motion codes

Aside from clicking and moving the cursor, there are some actions that you may want to perform while in the nouse cursor state. The way this can be done is using the nousor motion coding scheme. Whenever you move your position to a corner of the range, that corner's corresponding number will be remembered.

- Digits indicate the corners that nose needs "enter" in order to execute the command:

0 3

1 2

- Corners size (width and height) can be increased using the sliders from (1/5 to 1/4 to 1/3) of the motion range
- Green lines are drawn on the Nousor when any valid code is being entered..
- Motion codes should be memorized as writtings in the air with the nose: e.g. "Z"-settings, —/
- The last digit always corresponds to "YES"-type head noding (vertical head motion)
- when all but last digit is entered, the Confirmation is

requested (by showing the command's

```
CODE_LENGTH = 5
CODE_BOARD  01232
CODE_CHALK  01301
CODE_SETTINGS  03123
CODE_ENTER  32032
CODE_GLUE_CURSOR 13201
CODE_LOCKONAREA 13023
```

Motion-based key entry using NouseBoard and NousePad

When you first open NouseBoard you will be in the state where you need to put your face in the center of the nouse cursor. Once you do this, the NouseBoard and NousePad will popup and you'll be ready to type.

The first state of the NouseBoard is one where the whole board in coloured white. This indicates that no key have been selected yet.

Note that for each cell of the NouseBoard, there are four characters (or commands), one at each corner. The first thing you need to do is decide which corner the character you would like to type resides in. When you move the NouseBoard selector to a corner, all of the cells will have their respective characters in that corner stay white while the rest of the NouseBoard turns gray.

Now that a corner has been selected, you can move the NouseBoard selector around to select the character you would like to type. Once you have moved the selector to the requested cell, stay still for a couple of seconds. This will cause the character or command to turn green.

Once a character has turned green you simply need to move the selector to another cell and that character will be typed in the NousePad. The NouseBoard will return to its initial all white state.

References

[1] In *CameraMouse website: http://www.cameramouse.com/ product.htm.*

[2] In *SmartEye website: http://www.smarteye.se/ demo.html.*

[3] In *MouseVision website: http://www.mousevision.com/ assistech/html/products/handieye.htm.*

[4] In *QualiEye website: http://www.qualisoftware.com/ moreinfo.php?id=4.*

[5] In *NaturalPoint website: http://www.naturalpoint.com/ trackir/products/howitworks.*

[6] J. Amyotte, G. Benoit, K. Giles, J. Turcotte, L. W. Chambers, D. O. Gorodnichy, H. Mckee, and B. Ndiaye. Interprofessional collaboration in the development of accessible technology: Nrc-ebri nouse pilot project phase i report. In *NRC-CNRC Technical Report to be submitted*, 2007.

[7] 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 Trans Neural Syst Rehabil Eng., 10(1):110*, 2002.

[8] V. Chauhan and T. Morris. Face and feature tracking for cursor control. In *12th Scandinavian Conference on Image Analysis*, 2001.

[9] T. Darrel, N. Checka, A. Oh, and L.-P. Morency. Exploring vision-based interfaces: How to use your head in dual pointing tasks. In *Technical Report AIM-2002-001, Artificial Intelligence Laboratorty@MIT*, 2002.

[10] J. W. Davis and S. Vaks. A perceptual user interface for recognizing head gesture acknowledgements. In *Workshop for Perceptive User Interface, pages 17, Orlando, FL*, 2001.

[11] L. El-Afifi, M. Karaki, and J. Korban. Hand-free interface- a fast and accurate tracking procedure for real time human computer interaction. In *FEA Student Conference, American University of Beirut*, 2004.

[12] D. Gorodnichy. On importance of nose for face tracking. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition (FG 2002)*, pages 188–196, Washington DC, May 20-21 2002.

[13] D. Gorodnichy and G. Roth. Nouse 'Use your nose as a mouse' perceptual vision technology for hands-free games and interfaces. In *Image and Video Computing, Volume 22, Issue 12 , Pages 931-942, NRC 47140*, 2004.

[14] D. O. Gorodnichy. Towards automatic retrieval of blink-based lexicon for persons suffered from brain-stem injury using video cameras. In *In CD-ROM Proc. of First IEEE CVPR Workshop on Face Processing in Video (FPIV'04), Washington DC, USA, NRC 47139*, 2004.

[15] D. O. Gorodnichy. Perceptual cursor - a solution to the broken loop problem in vision-based hands-free computer control devices. In *NRC-CNRC Technical Report. NRC/ERB-1133. February 2006. 23 pages. NRC 48472.*, 2006.

[16] J.Tu, T.Huang, and H.Tao. Face as mouse through visual face tracking. In *Second Workshop on Face Processing in Video (FPiV'05). In Proc. of CVR'05, ISBN 0-7695-2319-6*, 2005.

[17] R. Ruddarraju and et al. Perceptual user interfaces using vision-based eye tracking. In *5th international conference on Multimodal interfaces, pages 227 233, Vancouver, British Columbia, Canada*, 2003.

[18] K. Toyama. Look, ma-no hands! hands-free cursor control with real-time 3d face tracking. In *Workshop on Perceptual User Interfaces, San Fransisco*, 1998.

[19] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR 2001. Also "Robust real-time face detection", In International Journal of Computer Vision, 57 (2), 137-154, 2004.*

Observation Chart of Nouse Participants				
#	Diagnosis	Eye Blink	Movement	Eye contact with computer screen
001	ALS (fast progressing)	-Distinct at first but becomes more difficult at the end of trial	-Fluid transition -Good ROM for neck rotation on both side -Fair ROM in neck extension and flexion	-Pt. able to maintain eye contact with the computer screen
002	MS	-Difficult to recognize eye blinking because of the glasses, the eye patch and because of the poor eye contact with the screen.	-Too far sidewise -Difficulty controlling the head flexion	-Pt's has poor eye contact with the screen
003	MS Friedreich's ataxia	-Pt takes a longer period of time to blink -Difficulty performing distinct eye blink	-Demonstrates involuntary neck movement -Difficulty controlling the head flexion -L ROM > R ROM -Slow transition to center position	-Pt's has poor eye contact with the screen
004	Parkinson	-Many eye blinks -Difficult to distinguish voluntary eye blink	-Fluid transition to center position -Difficulty bringing the head extension	-Pt's has poor eye contact with the screen
005	Stroke	-Distinct	-Difficulty doing movements and maintaining head in a position for several seconds -Unstable head movements	-Pt's has poor eye contact with the screen
006	TBI	-Many eye blinks -Difficult to distinguish voluntary eye blink	-Fluid transition to center position	-Pt's has poor eye contact with the screen
007	ALS (fast progressing)	-Eyes are closed when the pt is moving -Difficulty performing distinct eye blink	-Too far sidewise -Difficult	-Pt's has poor eye contact with the screen
008	MS	-Distinct	-Fluid and smooth -Good ROM both sides -Limited ROM in neck extension/flexion	-Able to maintain eye contact with the screen
009	MS (stable)	-Distinct	-Good control of head movements -Head movement slightly limited by head rest	-Good eye contact with screen
010	TBI	-Difficulty performing distinct eye blinks -Difficult to distinguish the eye blink because of the lighting in the room	-Good control of head movements	-Pt's has poor eye contact with the screen
011	PSP – Progressive Supra-nuclear Palsy	-Slow eye blinks -Pt's tends to blink only once or to keep is eyes closed	-Slow and controlled head movements	-Pt is able to maintain eye contact with the screen and to follow the mouse
012	MS	-Distinct	-Fluid transition to center position -Good ROM	-Some difficulty maintaining eye contact with the screen
013	TBI	-Distinct	-Difficult movements -Lost of control in head flexion -L ROM > R ROM	-Pt is able to maintain eye contact with the screen
014	Parkinson	-Difficulty performing distinct eye blink	-Fluid -Good ROM	-Pt was able to maintain eye contact with the screen after further explanations
015	Spinal Cord Injury	-Distinct	-Good ROM -Controlled and fluid movements	-Pt's has poor eye contact with the screen for the first trial and is able to maintain eye contact with the screen for the 2nd trial after redirection

Table 2. Observation chart of Nouse participants: variability of head motion control for people with different disability conditions (from [6]. Medical diagnoses are Multiple Sclerosis (MS), including Friedreich's ataxia; Cerebral Vascular Accident (stroke), Parkinson's disease, Traumatic Brain Injury (TBI) and Amyotrophic Lateral Sclerosis (ALS). ROM: Range of Motion, Pt: Patient).