FaceMouse: A Human-Computer Interface for Tetraplegic People

Emanuele Perini, Simone Soria, Andrea Prati, and Rita Cucchiara

Dipartimento di Ingegneria dell'Informazione,
University of Modena and Reggio Emilia,
Italy Via Vignolese 905, Modena Italy
{perini.emanuele, soria.simone, prati.andrea, cucchiara.rita}@unimore.it

Abstract. This paper proposes a new human-machine interface particularly conceived for people with severe disabilities (specifically tetraplegic people), that allows them to interact with the computer for their everyday life by means of mouse pointer. In this system, called FaceMouse, instead of classical "pointer paradigm" that requires the user to look at the point where to move, we propose to use a paradigm called "derivative paradigm", where the user does not indicate the precise position, but the direction along which the mouse pointer must be moved. The proposed system is composed of a common, low-cost webcam, and by a set of computer vision techniques developed to identify the parts of the user's face (the only body part that a tetraplegic person can move) and exploit them for moving the pointer. Specifically, the implemented algorithm is based on template matching to track the nose of the user and on cross-correlation to calculate the best match. Finally, several real applications of the system are described and experimental results carried out by disabled people are reported.

1 Introduction

One of the human dreams is to live in an intelligent house, full of advanced devices and capable to understand the needs and satisfy them quickly. Nowadays, the technology is mature enough to realize at least part of this dream. Indeed, it is now possible to install in our house sensors of various type that carry out many functions: for example, it is possible to switch a light on when a person enters in the room or open and close windows coherently with the environmental conditions using brightness sensors.

Even though these tools seem unnecessary and expensive for most of the people, for people with motorial difficulties and handicaps they become an indispensable aid for their everyday life. Thanks to these technological aids, these people can do most of the normal things in their house, interacting with it by means of either remote controls or computers, directly from their bed or their wheel chair. For example, they can open and close doors and windows, switch the appliances on and off, write a letter, use a PC, and so on.

T.S. Huang et al. (Eds.): HCI/ECCV 2006, LNCS 3979, pp. 99 – 108, 2006.

Unfortunately, depending on the gravity of their disability, disabled people can be very limited in movements and cannot easily interact with computers or other devices. For this reason, new human-computer interfaces must be provided.

This paper presents a system called FaceMouse particularly conceived for tetraplegic people. These people, in fact, can only use the head (and with difficulty) to interact with the environment and require special adaptation. FaceMouse uses a standard webcam and computer vision techniques to track the nose of the person and use this to move the mouse pointer (in accordance with the direction of movement of the nose). The mouse pointer is used to select items on special screens, from virtual grids for interacting with the house, to virtual keyboard to allow word processing. The system has been tested on several tetraplegic people and resulted to be very effective and it is currently under commercialization.

2 Related Works

The interfaces between humans and computer proposed in scientific literature and commercial products can be grouped in three different classes: *2D synoptic interfaces*, where the user can activate remote devices selecting the relative icon, *3D virtual world* systems that simulate navigation and interaction with the real world, and, finally, the *classic graphical interface* based on windows.

The systems for disabled people want to reproduce movements of the computer mouse with different methods in order to interface with a computer. They are, typically, based on the tracking of some parts of the human body, indeed used to indicate where to point or where to move. Several approaches have been proposed for making the interface as much natural as possible: they are based on eye-tracking [1, 2], head-tracking [3, 4, 5] or gaze-control [6]. Unfortunately most of these systems are not enough reliable and robust to be usable by a seriously disabled user to pilot the mouse with precision. Moreover, in order to move, for example, the mouse pointer, these systems use a paradigm called *"pointer paradigm"* based on the idea that *"what I look is what I want"*: the user must directly indicate the point of interest on the screen [7, 8]. This task requires a precise control of the used part of the body (e.g, the head), but, unfortunately, many people with disabilities do not have this ability. For this reason, this work proposes the study of a new technique for moving the mouse pointer, exploiting on a paradigm that we called *"derivative paradigm"* based on the idea that "*where I look is where I want to go*".

Many proposals have been reported in the literature to estimate the motion of visual objects; after a preliminary phase of interesting object searching, most of these techniques employ a tracking algorithm to improve and facilitate the search at the next frame. The used tracking algorithm discriminates among these proposals: there have been proposals for probabilistic algorithms [9, 10], for algorithms based on the Kalman filter [11, 12] and on template matching [7, 8], and many others.

The proposed system, called FaceMouse, exploits a common low-cost webcam to capture images; computer vision techniques are then used to identify different parts of the user's face and to exploit them for moving the mouse pointer or generating a button click. As stated in [13], the best tracking method for user with severe disabilities is based on normalized correlation coefficient since movements in disabled people do not follow any predictable motion model. For this reason, a tracking method based on template matching, where the distance between the portion of the current image and the template is calculated using a cross-correlation function has been developed. Several improvements have been included in order to increase the stability and the robustness of the system.

3 User-Friendly HCI

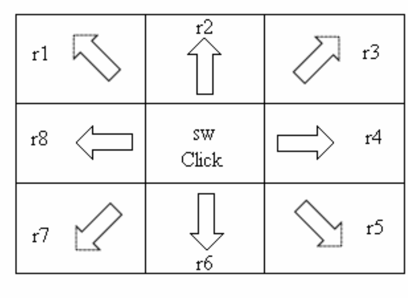
As above mentioned, for people with sever disabilities, to point at a precise position and maintain it for a while can be very hard. Moreover, the range of movements of the limbs and the head can be limited, preventing the user to point at borders of the image/target. This can be a serious problem, for instance, in Windows-like interfaces, where important elements (such as icons or status bars) are located at image borders. This is even worst for tetraplegic people where the head is used to interact with the environment.

For this reason, the most suitable approach, in these cases, is to use the derivative paradigm described before, in which the movement of the user's head does not indicate the precise position of the mouse pointer, but the direction along which the mouse must be moved. In this way, the user can interact even if he has not a precise control of his head, because he can only do small movements or he suffers from muscular spasms. To detect the chosen direction, we need to track a "good" feature on the person's head/face. Which is the best feature to follow will be discussed in the next section. This section, instead, will describe the user-friendly human-computer interface (HCI) developed for the system.

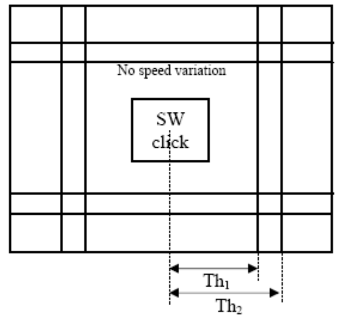
Initially, the feature is extracted by means of a semi-automatic method. The user is asked to keep the face as much still as possible and an operator (able to use standard mouse devices) selects a point on the current image centered on the chosen face feature (for example, the center of the nose). A squared template centered on this point is used as a model and saved for further matching. Automatic detection of facial features has been explored. For example, the method proposed in [14] (based on stored templates to find the nose starting from the eyes' position) seems promising and have been tried. However, automatic feature initialization comes at the cost of reduced robustness and, due to the particular final users of our system, it is not easily applicable.

The initial center of the feature is used to create a grid as basic interface. More in details, the screen is virtually divided into a grid of 3x3 windows, as reported in Fig. **1**.a. The size of each window can be adapted to the abilities of the user: smaller windows are used for users with difficulty to move the head, whereas larger windows are used for users with muscular spasms that have difficulties in keeping the head still in a given position.

After this initialization step, the face feature is tracked to detect movements and understand commands. If the center of the feature is detected inside the central region (indicated with "SW Click" in Fig. **1**, where "SW" stays for "stationary window"), it corresponds to the request of not moving the mouse pointer, and if the user maintains the feature in that area for more than a defined time T_{click}, a button click is generated.



(a)



(b)

Fig. 1. Example of the grid used as basic interface (a) and for velocity management (b)

If the user moves in a different window (r1,…,r8), the mouse pointer is moved in the corresponding direction. Three different types of mouse pointer's dynamics have been implemented:

- a dynamics with constant velocity, independently from the displacement from the center of the grid;
- a dynamics with constant acceleration, independently from the displacement from the center of the grid; a maximum allowed velocity can be set;
- a dynamics in which the velocity is function of the displacement from the grid's center.

Although the system implements all these three dynamics, the most suitable is the third one. To implement it, we further divided the image grid as shown in Fig. **1**.b. Basically, defined $D=\left(d_x,d_y\right)$ the distance of the feature's center from the center of the grid and $V=\left(v_x,v_y\right)$ the velocity of movement to be set for the mouse pointer, $V=\left(0,0\right)$ if the user points at the "SW click" zone, and $V=\left(v_x^{base},v_y^{base}\right)$ (i.e., constant velocity) if the user points inside the first area surrounding the stationary window. For the following zones, the *x* and *y* components of the velocity are updated as follows :

$$v_x=\begin{cases} f_1\left(v_x^{base},a_1,v_1^{\max}\right) & \text{if } Th_1\le d_x\le Th_2 \\ f_2\left(v_1^{\max},a_2,v_2^{\max}\right) & \text{if } Th_2\le d_x\le w \end{cases} \tag{1}$$

where w is the image width, and f_1 and f_2 are two functions consisting in constant acceleration starting from the first parameter, incrementing at each step of the acceleration given by the second parameter, and upper-bounded by the third parameter. In other words, in the outermost area, the velocity is incremented from $v_1^{\max}$ to $v_2^{\max}$ with a step of a_2. A similar approach is used for *y* component.

Summarizing, when the user wants to click on a point of the screen, he must position the mouse pointer on the desired point, return back in the stationary window SW and remain in this area for the time T_{click}. Also the user can disable the click.

Unfortunately, this approach does not work properly for users that can not correctly move the head in all the directions: for instance, people in which the head must be kept fixed by a support or a headrest, and that therefore are unable to even do small movements upwards, or users that can not move the head towards right or left without also moving it down. For this reason, an alternative solution has been implemented: during the setup phase the user (with the help of an operator) chooses a set of points that represent the positions that the user can assume in correspondence of the four directions (up, down, right, left) and of the central position. In this case the selected direction of movement is computed not looking at the position of the feature's center, but at its distance from the pre-defined positions: the positions at minimum distance identifies the selected direction.

This further possibility allows us to increase the usability of the system and the range of potential users.

4 Nose Tracking

As described in the previous section, during the initialization phase, the operator will select for the disabled user a "good" feature on his face. In principle, this feature can be whichever part of user's face (nose, eye, lips, chin, etc.), but, in order to assure a tracking more robust, it should be univocally detectable, and invariant to rotations, translations, and scale changes. It has been demonstrated [8, 15] that the features that better exhibit these characteristics are those represented by convex shapes and the only convex shape easily visible on human face is the nose tip. Assuming light conditions constant between two consecutive frames, normalized cross correlation [16, 17] can be exploited to perform at time t the template matching with respect to the template T^{t-1} at time $t-1$. The template T^0 is that saved at the initialization phase.

Several improvements have been introduced to increase the reliability and robustness of the system. First, the normalized cross-correlation is thresholded to retain only "sufficiently good" matches. The position of the best match (whose score is greater than the threshold) is used to take the next template T^t.

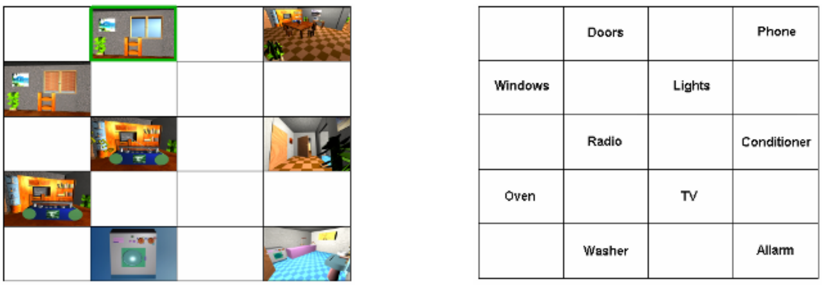
However, this tracking algorithm is very simple and, consequently, it is prone to false matches (mainly due to the face badly illuminated or to over-exposed faces). To reduce false matches, the previous template T^{t-1} is re-aligned to the initial template T^0 as soon as it becomes too different with respect to T^0 (this check is performed once every second).

5 Applications

The proposed system opens to a remarkable number of applications. In the field of aid for disabled people, the possibility to move the mouse pointer allows the following types of application:

1) the use of synoptic interfaces for controlling the house;
2) applications for the interpersonal communication and writing;
3) general Microsoft Windows applications.

For interacting with the house, the system can be interfaced either with an existing system for home automation or directly with the actuators. Fig. **2** shows two examples of the interfaces used in FaceMouse for opening/closing doors and windows, or for switching on and off the different appliances. The user moves on the window corresponding to the chosen device and stays still for a while to generate a button click and open a new interface with specific commands for that device.



(a) (b)

Fig. 2. Two examples of interface for controlling the house

For example, four automated windows can be controlled by the interface reported in Fig. **3**: they can be simply opened or closed by pushing the corresponding button, or manually opened to some extent by moving the sliding bars.

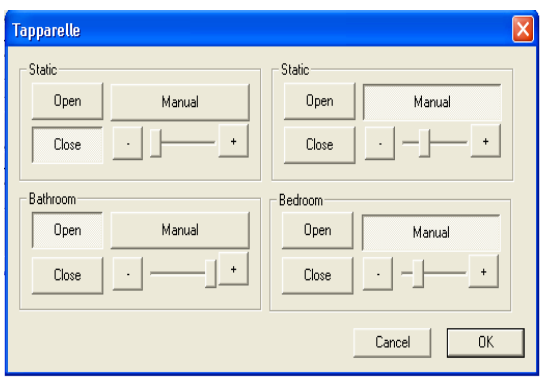


Fig. 3. Interface to control four automated windows

Regarding the interpersonal communication and writing, these are obviously very important tasks for people that sometimes do not have any other way to communicate. Compared with other systems (like supports of transparent plexiglass used as pointers to look), FaceMouse does not need an operator to help the user and it uses a more user-friendly interface, based on the virtual keyboard reported in Fig. **4**. The user can select (by moving the mouse pointer) the single characters to compose words and phrases that the system repeats by using a vocal synthesizer and transmits to the word processor.

A		B		F		G
	E		J		I	
D		C		L		H
	O	<	A	•	Y	
M		N		R		S
	K		Z		U	
Q		P	W	V		T

Fig. 4. Virtual keyboard used in FaceMouse

As above described, the selection of each character is obtained by moving on it and then returning to the stationary window to generate the button click. This procedure can result both too long lasting and not easy for some disabled users. In particular, muscular spasm prevent some categories of users to remain still in the stationary window. To solve this problem and to speed up the process of selecting the characters, several tests with tetraplegic people have been carried out. At the end, two improvements have been included in the system.

The first improvement consists in not requiring to the user to return in the stationary window, allowing to select the character by simply remaining for a while on the corresponding window. This is achieved by means of a voting procedure in which a score is assigned to every character cell and accumulated for every Δt time (fixed by the user) of permanence on the cell. When the accumulated score for a cell exceeds a given value, the corresponding character is selected and the score of all the active cells is reset. The score assigned to a cell during the permanence of the mouse pointer on it is computed by giving higher value when the pointer is close to the center of the cell. This brings to two main advantages: first, it reduces the time necessary to select a character if the user is able to point close to the center, and, second, it reduces the incorrect selections by weighting less "marginal" (and possibly erroneous) selections.

A second improvement is that of suggesting the next most probable character, given the one currently selected. Collecting statistics among different texts, we compute the probability $p\left(c\left(t+1\right)\mid c\left(t\right)\right)$ of having the character $c\left(t+1\right)$ selected given that the current character is $c\left(t\right)$. Thus, at the center of the virtual keyboard, the five most probable characters are reported (Fig. **5**) and, after every selection, the pointer is automatically moved to the central one that contains the most probable next character.

In this way, if the desired character is one of the five reported, the selection can be obtained quickly with small movements of the mouse pointer.

Finally, our system can efficiently interface with Microsoft Windows operating system allowing the user to run most of the common computer software, and, for instance, to surf on the Internet.

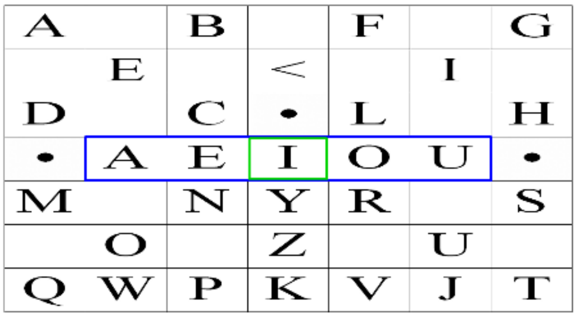


Fig. 5. The virtual keyboard of FaceMouse with, at the center, the five most probable next characters

6 Experiments

FaceMouse has been deeply tested with ten disabled users, in particular with tetraplegic users. After some hour of write training (8-10 hours distributed in some days) with our system, we have asked to the users, habituated to use a scansion system[1], to write the phrase "I am writing with my nose", of 25 characters, with their traditional system and with FaceMouse.

The tests have been carried out with two different dynamics of movement of the mouse pointer: the first one uses a constant acceleration, whereas the second varies the velocity in dependence on the position of the mouse pointer (see dynamics 2 and 3 in section 3).

Analyzing the results reported in Table **1**, several considerations can be made:

1) using the proposed system, the users can write more than two times (without prediction) or more than three times (with prediction) faster than using a traditional system;
2) the best performance can be achieved by exploiting prediction and the velocity dependent on the pointer's position; this is due to the fact that the user can accelerate and decelerate the mouse pointer; however, there can be users for which the constant acceleration is the only possibility;
3) with the prediction system the user can speed up the writing process (by 59% and 25% for dynamics 1 and 2, respectively) with respect to the case

[1] In the scansion system the computer proposes one by one all the characters (both on the screen and by means of audio) to the user. When the chosen character is proposed, the user can push a button or similar to confirm the selection.

without the prediction. Moreover, the prediction system almost nullifies the difference in performance between the two dynamics, since the required movements are greatly reduced.

Table 1. Experimental results

Type of movement	**Without prediction**		**With prediction**		
	Total time	**char/min**	**Total time**	**char/min**	
Scansion systems	6’ 06’’	4.1	**---**	**---**	**Speed up**
FaceMouse with constant acceleration	3’ 03’’	8.2	1’ 55’’	13.1	**59%**
FaceMouse with velocity function of position	2’ 19’’	10.8	1’ 51’’	13.5	**25%**

Regarding the computational requirements, FaceMouse does not demand much power and works properly also with PC with a standard 1 GHz processor, processing about 30 frames/sec using only the 50 percent of the CPU time.

7 Conclusions

In this paper we have presented a human-machine interface for helping tetraplegic people (or, more in general, disabled people) to interact with the environment and with other people. The system is based on computer vision techniques, therefore it is not necessary to apply sensors to the body of the users and the interface is user-friendly and adapted to the needs of tetraplegic people.

The tests, performed with several tetraplegic people have demonstrated that the developed system allows the users to write more than three times faster than with traditional systems.

References

1. Beach, G., Cohen, C.J., Braun, J., Moody, G. “Eye tracker system for use with head mounted displays”. In: Proceedings of IEEE Intl Conf on Systems, Man, and Cybernetics, vol. 5, (1998) 4348–4352
2. Zhu, Z., Ji, Q., Fujimura, K., Lee, K. “Combining Kalman filtering and mean shift for real time eye tracking under active IR illumination”. In: Proceedings of Intl Conf on Pattern Recognition, vol. 4, (2002) 318–321
3. Jilin Tu, Huang, T., Hai Tao. “Face as mouse through visual face tracking”. In: Proceedings of 2nd Canadian Conf on Computer and Robot Vision, (2005) 339–346
4. Qian, R.J., Sezan, M.I., Matthews, K.E. “A Robust Real-Time Face Tracking Algorithm”. In: Proceedings of IEEE Intl Conf on Image Processing, vol. 1, (1998), 131–135.
5. Toyama, K.. “Look, ma -- no hands! Hands free cursor control with real-time 3D face tracking”. In: Proceedings of Workshop on Perceptual User Interfaces, (1998) 49–54

6. Hutchinson, T.E., White, K.P., Martin, W.N., Reichert, K.C., Frey, L.A. "Human-computer interaction using eye-gaze input". IEEE Transactions on Systems, Man, and Cybernetics, 19 (1989) 1527–1534
7. Betke, M., Gips, J., Fleming, P. "The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With Severe Disabilities". IEEE Transactions on Neural Systems and Rehabilitation Engineering, 10(1) (2002) 1–10
8. Gorodnichy, D.O., Malik, S., Roth, G. "Use Your Nose as a Mouse – a New Technology for Hands-free Games and Interfaces". Computational Video Group, IIT, National Research Council, Ottawa, Canada K1A 0R6
9. Verma, R., Schmid, C., Mikolajczyk, K. "Face detection and tracking in a video by Propagatine Detection Probabilities". IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(10), (2003), 1215-1228
10. Moghaddam, B., Pentland, A. "Probabilistic Visual Learning for Object Recognition". IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7), (1997), 696-710
11. Kalman, R.E. "A new approach to linear filtring and prediction problems". Journal of Basic Engineering, Trans. of ASME, (1960)
12. Oliver, N., Pentland, A.P., Berard, F. "LAFTER: lips and face real time tracker". In: Proceedings of Computer Vision and Pattern Recognition, (1997), 123-129
13. Fagiani, C., Betke, M., Gips, J. "Evaluation of Tracking Methods for Human-Computer Interaction". In: Proceedings of Workshop on Applications of Computer Vision, (2002)
14. Campedelli, P., Casiraghi, E., Lanzerotti, R. "Detection of Facial Features". Technical Report DSI, University of Milan
15. Gorodnichy, D. "On importance of nose for face tracking". In: Proceedings of Intl Conf on Automatic Face and Gesture Recognition, (2002)
16. Horn, B.K.P. Robot Vision, Cambridge MA: MIT Press, 1986
17. Betke, M., Haritaoglu, E., Davis, L.S. "Real-time multiple vehicle detection and tracking from a moving vehicle", Machine Vision and Applications, 12(2), (2000), 69-83