TRACKING HEAD MOVEMENT FOR AUGMENTATIVE AND ALTERNATIVE COMMUNICATION

Carlos Gonçalves*, Antônio Padilha Lanari Bó*, Rogério Richa†

*Electrical Eng. Department, University of Brasilia (UnB), Brasília, Brazil

†Brazilian Nat. Inst. for Digital Convergence, University of Santa Catarina (UFSC), Florianópolis, Brazil

Emails: carlosgoncalves@lara.unb.br, antonio.plb@lara.unb.br, richa@incod.ufsc.br

Abstract— The use of computers as a communication tools is trivial nowadays, but the use of the PC by a impaired person is often a challenge. Augmentative and alternative communication (AAC) devices can empower these subjects by the use of their remaining functional movements, including head movements. Currently computer vision AAC solutions present limited performance in the presence of involuntary body movement or spasticity (stiff or rigid muscles). Our work proposes a novel human computer interface (HCI) based on the functional head movements of each user. After calibration, a Hidden Markov Model (HMM) classifier represents the desired functional movement based on the velocities components of the estimated head position. New segmented movements are then classified in valid or invalid based on the HMM. Valid segments can generate mouse "click" events that can be used with scanning virtual keyboards, enabling text editing, and within scanning based software that can control mouse functions.

Keywords— Augmentative and alternative communication, HCI, computer vision, head tracking, HMM.

Resumo— O uso de computadores como uma ferramenta de comunicação é trivial atualmente, mas o uso de um PC por uma pessoa com deficiência é na maioria das vezes um desafio. Dispositivos de comunicação alternativa e aumentativa (CAA) podem potencializar estes sujeitos pelo uso dos seus movimentos funcionais residuais, inclusive movimentos de cabeça. Atualmente, solucões de CAA baseadas em visão computacional apresentam performance limitada na presença de movimentos involuntários ou espasticidade (músculos tensos e rígidos). Nosso trabalho propõe uma nova interface homem-computador (IHC) baseada nos movimentos de cabeça funcionais de cada usuário. Após calibração, um classificador HMM representa o movimento funcional desejado baseado nos componentes de velocidades da estimativa de posição da cabeça. Novos movimentos segmentados são então classificados em válidos ou inválidos baseados na HMM. Segmentos válidos podem gerar eventos de "click"de mouse que podem ser utilizados com teclas virtuais com varredura, permitindo a edição de textos, e com softwares baseados em varredura, o controle do ponteiro do mouse.

Palavras-chave— Comunicação alternativa e aumentativa, visão computacional, rastreamento de cabeça, HMM.

1 Introduction

The use of PC is generalized in society. With them, people can communicate and work easier and faster. Unfortunately, individuals with impairments cannot use mouse or keyboard as they were designed. In order to control the computer, the user must have dexterous finger movements and large amplitude of hand and arm movements. Several medical conditions imply in the lack of control or even lost of upper body movement.

Solutions that permit the use of a computer by an impaired person can be called *Augmentative and Alternative Communication* (AAC) devices. Each individual will have its own cognitive and physical limitations that will guide the choice of one or several AAC devices (Higginbotham et al., 2007). Mechanical switches, alternative mouses and scanning virtual keyboards are the most used solutions to facilitate the use of a computer, but when the individual has only functional head or facial movements, computer vision systems are good alternatives (Tai et al., 2008).

These computer vision human computer interfaces (HCI) can be divided in marker or markerless solutions. Markerless solutions can track the eye gaze or the head movement with the use of their features. Many eye gaze tracking equipments use infrared emitters and cameras, requiring specialized hardware (*Tobii PCEye*, *Dynavox Eyemax*, *EyeTech TM4*).Their drawback is that the user must have a good sitting position and avoid gross head movements, so individuals with dystonia, spasticity, or tremors have difficulty in using these devices continuously.

Markerless systems that track the head movement use a conventional webcam. The main goal is to track the head and estimate the corresponding cursor movement (*Headmouse*, *CameraMouse*, *Enable Viacam*).

In Gorodnichy and Roth (2004), the authors used a convex-shaped nose feature to track the nose position in a frame. Nose features are also used in (Varona et al., 2008), where the eyes are tracked using the eyes histogram and a mean-shift algorithm. A head tracking system that does not use facial features is presented in (Pallejà et al., 2008), the optical flow from two consecutive frames is computed and used to identify head movement, as well as eye blink and mouth opening. In the authors latest work (Pallejà et al., 2011), the user's face is detected by the algorithm designed by Viola & Jones (Viola and Jones, 2004) implemented in the OpenCV library, and the head tracking is performed with a face template matching routine.

In (Gorodnichy and Roth, 2004) – (Pallejà et al., 2011) the fine control of the head is mandatory, prohibiting its usage by persons that suffer from involuntary movements or an impairment that interfere with the head movement.Our work presents a novel approach where a markerless head tracking system is implemented and is suitable for recognizing the functional head movement of the user. Our initial intention is to give an alternative to impaired persons to use the computer, even if they do not have a fine control of the head movement.

In view of the described limitations, we developed a method based on a histogram tracking algorithm (Feng et al., 2008), and a HMM classifier for identifying user commands. This classifier can model the functional movement of the user, taking in account the different states and state transitions that can represent the movement.

This paper is organized as follows: Section 2 presents a detailed description of the methods used to achieve our goals. In Section 3, experimental results of the implemented classifiers are discussed. Finally, Section 4 presents a conclusion of our work.

2 Methods

In this section we present the techniques used to accomplish the following tasks: face detection, face tracking and "click" movement detection.

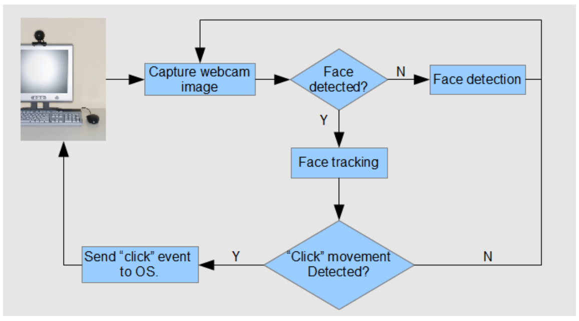

Figure 1: System's work-flow diagram.

2.1 Face detection

In order to initialize the system, a person must be positioned in front of the camera and his face must be recognized. The Viola & Jones (Viola and Jones, 2004) face detecting algorithm creates a cascade of Haar-like feature classifiers. We use the default training data from the OpenCV library to implement this functionality. As shown in figure 1, the face detection function gives the initial estimate of the head position that will be used by the face tracking method.

2.2 Face tracking

With a sample of the head image acquired by the face detection routine, the features of the target can be acquired by the system. However, to track the gross and fine head movements, our work does not rely in spatial features. Instead, we chose a histogram-based tracking method, which has proven to be an effective technique for head tracking (Comaniciu et al., 2003).

In Feng et al. (2008), the authors develop a similarity function that takes in account the histogram of both target and target searching area. Their intention is to minimize the influence of background pixels that are present in the target image that may bias the localization or even generate a matching error, as occur using the commonly used Bhattacharyya coefficient.

This new similarity function is called posterior probability measure (PPM). Considering the histograms of the target model, target candidate and search area as $\boldsymbol{q}$, $\boldsymbol{p}$ and $\boldsymbol{s}$, the PPM similarity function $\phi(\boldsymbol{p},\boldsymbol{q})$ is as follows:

$$\phi(\boldsymbol{p},\boldsymbol{q}) = \frac{1}{m}\sum_{u=1}^{m_u}\frac{p_u q_u}{s_u}, \tag{1}$$

where m_u is the number of bins in every histogram, and m is the number of pixels in the target model.

The suppression of pixels that represent the background creates a similarity function that has a more distinct sharper peak, even when the target model holds some portion of the background (Feng et al., 2008).

Another main feature of the PPM is that it can be treated in a pixel-wise manner, fastening the implementation. From Eq.(1), can be found the pixel-wise equation below:

$$\phi(\boldsymbol{p},\boldsymbol{q}) = \frac{1}{m}\sum_{j=1}^{m}\frac{q_u(j)}{s_u(j)}. \tag{2}$$

In Eq.(2), j is the pixel index of the target candidate, $\boldsymbol{q}_u(j)$ and $\boldsymbol{s}_u(j)$ are the values of the current pixel color in both histograms. The target model histogram needs to be calculated only once, and the histogram of the searching are must be calculated in every new frame.

With the PPM as similarity function, the target tracking can be done with the mean-shift algorithm (Comaniciu et al., 2003), where the new estimate of the target position is a weighted contribution of every pixel.

$$\omega_j = \frac{q_u(j)}{s_u(j)}, \tag{3}$$

$$\hat{\boldsymbol{y}}(i+1) = \frac{\sum_{j=1}^{m}\boldsymbol{x}_j(i)\omega_j}{\sum_{j=1}^{m}\omega_j}. \tag{4}$$

In Eq.(4), $\boldsymbol{x}_j$ is the position of the jth pixel in the target candidate.

2.3 "Click" movement detection

As mentioned above, our goal is to create a system that is able to identify functional commands from head movement to create "click" events. We reckon that every user may have a unique functional head movement, then our system needs to be able to learn motions models during a calibration phase, and then classify motion candidates as modeled motions or not.

2.3.1 Horizontal threshold

A simple movement classifier can be created representing the functional movement by its horizontal amplitude. A video of this implementation can be seen in *LARA Videos - Webcam Virtual Keyboard*, http://www.youtube.com/unblara.

First, the initial head position estimate is generated with (Viola and Jones, 2004). With the position of the largest face detected in the first frame, the system saves the head image to calculate its histogram. It was implemented an image histogram of the three RGB components with 16 bins for each component. In figure 2a the initial head position is the red square, and the initial search region is the blue square. For every new

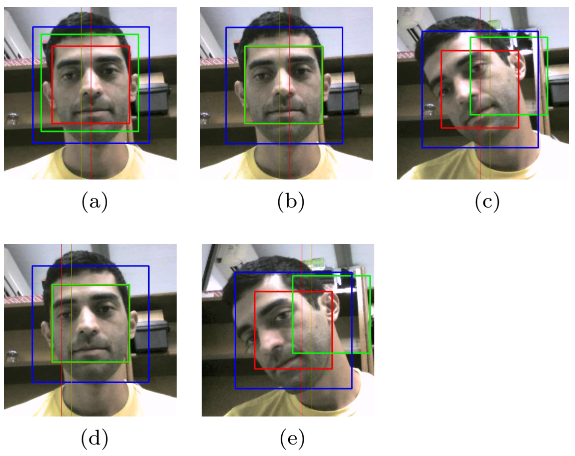

Figure 2: Screenshots of the running application. From 2a to 2e we have the following events: initialization with face detected (red square), reference update (green square), calibration movement, thresholds lines updates, "click".

frame t, the histogram of the search area at t_{-1} will be calculated and the implemented mean-shift algorithm, as in Eq.(4), will generate the new head position estimate, then the search area will be updated.

In order to calculate a horizontal movement of the estimated head independent of the initial head position, a moving reference was created. It's position is updated with the mean of a circular buffer of 30 elements. The points are added to the buffer only when the current head speed is less than 50 pixels/s.

Since the start of the tracking, the system is measuring the largest difference between the current head position and the moving reference. This value is stored and shown by the red line in figure 2b. When a desired "click" movement is performed, like in figure 2c, this line changes its position holding the horizontal amplitude of the movement, as in figure 2d. With a keyboard command this calibration is finished and the system can generate "click" events.

In order to create a threshold that could favor movements without the same maximum amplitude stored in calibration, the "click" threshold (yellow line) is 20 pixels closer to the moving reference. In figure 2e there is an example of a classified "click" movement.

2.3.2 HMM classifier

A more complex classifier can be created using a two stage classification as in (Lin and Kulic, 2011) and (Feng-Shun Lin and Kulic, 2012). First the "click" candidates are segmented using a specified pattern of a sequence of velocity peaks and zero velocity crosses (ZVC) (Fod et al., 2002). Since our classification system was designed to work with the x and y velocities (two DOF), it is difficult to create a segmentation routine based on the individual ZVCs. Our approach was to create an auxiliary variable that could combine both measurements (Fod et al., 2002). Its formula is presented below:

$$z = v_x^2 + v_y^2. \tag{5}$$

Eq.(5) has difficulties in registering zero value, then a threshold (3000) was specified based on experiments. The moments in the sequences were this threshold was crossed were labeled as ZVC. A valid segmenthad at least three ZVC and with peaks detected between the ZVCs pairs. The figures 3 and 4 show the segmentation process with more detail.

With the selected training segments, a hidden Markov model (HMM) can be set to represent the functional head movement. Initial estimates of the observation probabilities of (v_x, v_y) have to be find before HMM training. These estimates can be generated with simple cluster segmentation algorithms, and the number of cluster can be an initial guess of the number of states that represent the hidden Markov model.

After training the HMM with the Baum-Welch algorithm (Rabiner, 1989), initial states probabilities, and transition matrix are updated and represent the desired functional movement. The evaluation of a movement candidate by the HMM generates a probability of the candidate had been generated by the model, and with a trained threshold the "click" events can be generated.

3 Experiments

In this section are presented our experiments with the presented movement classification approaches. First the already implemented system with the threshold approach runs and collects the kinematic data of the target and "click" moments. This software were designed with C++ language and uses the OpenCV 2.4 library. Then the recorded data are used with MATLAB and a HMM toolbox (Murphy, n.d.). Finally the classification values of both are analyzed with two validating movement sequences.

3.1 HMM training

There were collected one training sequence of five desired "click" movements, and two validating sequences. The first one has another five different valid "click" movements, the second one has only two valid movements and other five invalid ones (two inverted "click" movements, two horizontal translations, one circular head movement). The x and y velocities data collected in the training sequence are segmented as mentioned in section 2.3.2. The ZVC points obtained with Eq.(5) and the established threshold (3000) can be seen in figures 3 and 4. With the segmented movements,

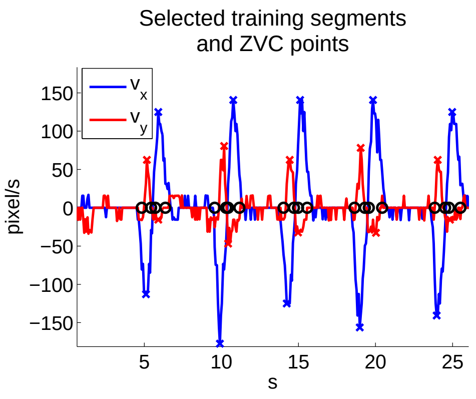

Figure 3: Training sequence segmented.

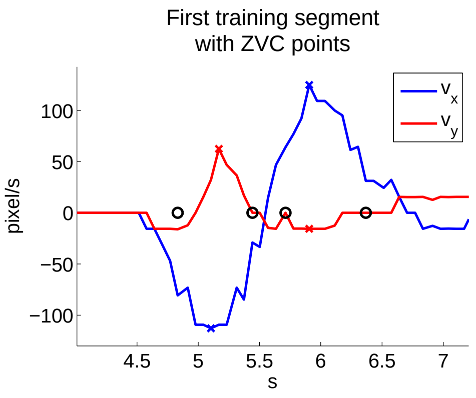

Figure 4: First segment of the training sequence.

their observation *pdfs* need to be estimated for later use in the HMM. We had found a good estimate creating clusters of (v_x, v_y) pairs with the $k-means$ algorithm. The number of possible clusters was set to four, since it gave a good probability estimation. The mean and covariance values of each cluster population were calculated and used to estimate bi-variate Gaussian *pdfs* to represent the observation probability for the desired clusters. Calculating the value for the estimated

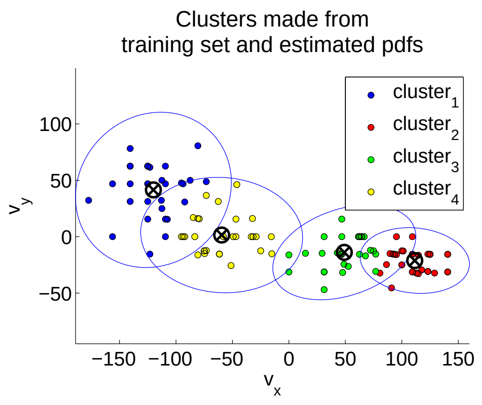

Figure 5: Created clusters for the training sequence with estimated bi-variate Gaussian 3σ regions.

pdfs during the first training segment, we generate the figure 6. From figures 5 and 6, the cre-

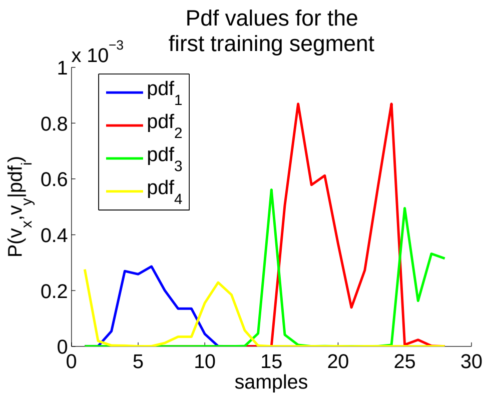

Figure 6: *Pdf* values during the first training segment.

ated clusters can be easily interpreted as different stages in the valid "click" movement, then it was decided that a four stage HMM could be trained to represent the valid desired motion. This four stage HMM had initially random values for the initial states probabilities $\boldsymbol{\pi_0}$, and transition matrix $\boldsymbol{A_0}$. The observation *pdfs* for each stage are initial bi-variate Gaussians estimated for each cluster i, $N_i(\boldsymbol{\mu_0}, \boldsymbol{\Sigma_0})$. The HMM was trained using the training sequence segments and the generated $\boldsymbol{\pi_1}$, $\boldsymbol{A_1}$, $N_i(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1})$ were obtained the with the Baum-Welch algorithm (Rabiner, 1989).

The final values of $\boldsymbol{\pi_1}$ and $\boldsymbol{A_1}$ are:

$$\boldsymbol{\pi_1} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \boldsymbol{A_1} = \begin{bmatrix} 0.85 & 0 & 0 & 0.15 \\ 0 & 0.86 & 0.14 & 0 \\ 0 & 0.22 & 0.78 & 0 \\ 0.2 & 0 & 0.2 & 0.6 \end{bmatrix}. \quad (6)$$

The results in Eq.(6) are expected from figure 6. The state 4 has initial probability of one and the possible state transitions are: $1 \to 4$, $2 \to 3$, $3 \to 2$, $4 \to 1$ and $4 \to 3$.

To complete the HMM classifier design, a log-likelihood (LL) threshold is need in order to classify segments in valid or invalid. In figure 7 the LL values are plotted from the five training segments. Based in these values, a threshold of -300 were chosen, values below the threshold are considered invalid "click" segments. With the trained HMM

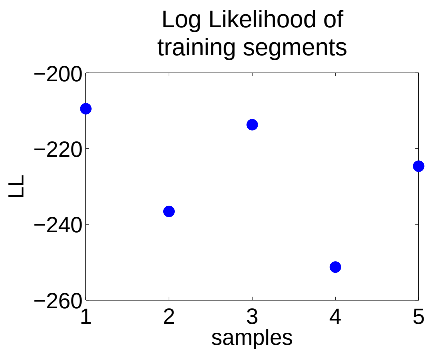

Figure 7: LL values of the training segments.

classifier, we can implement the classification of the segmented movements of the two validating sequences.

3.2 Data analysis

The first validation data was presented to the HMM classifier. It can be seen from the figure 8 that all valid "click" movements previously classified by the position threshold were also classified by the HMM classifier, the only difference is that the HMM classifier signals the "click" event at the end of the segment. This behavior may be unusual for the user initially, but can be easily overcome with training.

The figure 9 shows the LL values for the segments. The second validation data was presented to the HMM classifier and the valid movements were again correctly observed, and above all, the invalid movements that were classified as valid by the horizontal threshold classifier were considered invalid by the HMM classifier. Figures 10 and 11 show these results. In 11 the red dots correspond to LL values of invalid segments. It also can be seen that the number of segments in figure 11, six elements, is less than the performed ones, seven. The inverted "click" movements are

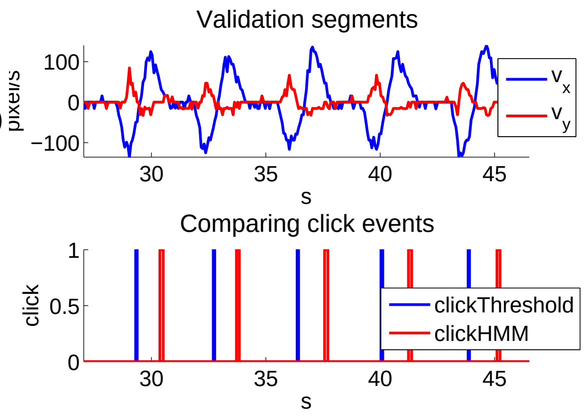

Figure 8: "Click" events generated by both classifiers.

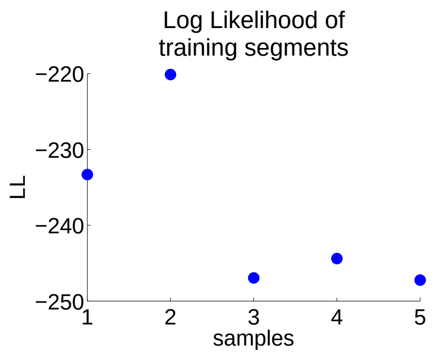

Figure 9: LL values of the validating segments.

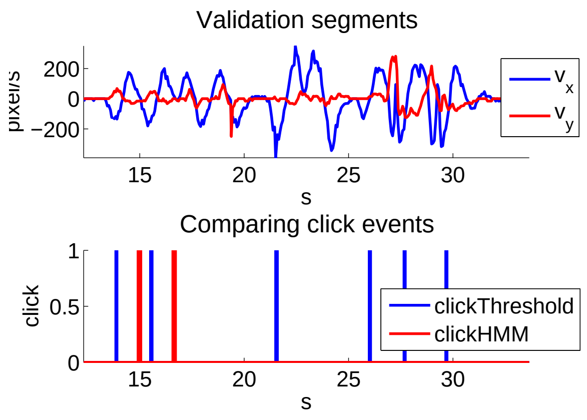

Figure 10: "Click" events generated by both classifiers.

correctly segmented and presented to the classifier, but the horizontal translations and the circular head movement are too different from the desired movement, and then they are incorrectly segmented.This example shows how the segmenting algorithm acts like an initial classifier sorting the most similar candidates to the valid movement template. With fewer candidates, the HMM classifier does not halts the system, making it feasible for a future online implementation as in (Feng-Shun Lin and Kulic, 2012).

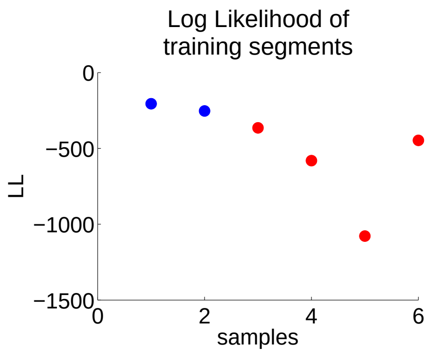

Figure 11: LL values of the validating segments.

4 Conclusion

In order to implement a AAC system that relies only on the images captured by a webcam, we have designed a method that tracks the user head movement without the use of markers. Our tracking system can cope with different amplitude and frequencies of voluntary and involuntary head movements, and also is able to model the remaining functional head movements of each user. This novel head tracking system could be well suited for patients with dystonia or spasticity that usually have to use mechanical switches mounted in their wheelchairs or beds.

Our simple classifier has a horizontal threshold approach, which can represent the amplitude of the functional movement. In order to develop a more reliable movement classifier, was designed a segmentation routine capable of trimming the received data into segmented candidates, and then using these segments into a HMM classifier, previously trained with the desired functional movement. This approach was implemented in MATLAB and had shown promising results, both classification ration (100%) and false positive ratio (0%) were satisfied.

Our future work will focus in implementing the segmentation routine and the HMM classifier online and test on real patients with different functional movements and user disabilities.

References

Comaniciu, D., Ramesh, V. and Meer, P. (2003). Kernel-based object tracking, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **25**(5): 564–577.

Feng-Shun Lin, J. and Kulic, D. (2012). Segmenting human motion for automated rehabilitation exercise analysis., *Conference proceedings: Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference* **2012**: 2881–4.

Feng, Z., Lu, N. and Jiang, P. (2008). Posterior probability measure for image matching, *Pat-tern Recogn.* **41**: 2422–2433. DOI: 10.1016/j.patcog.2007.12.013

Fod, A., Mataric, M. J. and Jenkins, O. (2002). Automated derivation of primitives for movement classification, *Autonomous robots* **12**: 39–54. DOI: 10.1023/A:1013254724861

Gorodnichy, D. O. and Roth, G. (2004). Nouse ‘use your nose as a mouse´ perceptual vision technology for hands-free games and interfaces, *Image and Vision Computing* **22**: 931–942. DOI: 10.1016/j.imavis.2004.03.021

Higginbotham, D. J., Shane, H., Russell, S. and Caves, K. (2007). Acces to aac: Present, past and future, *Augmentative and Alternative Communication* **23**: 243 − 257. DOI: 10.1080/07434610701571058

LARA Videos - Webcam Virtual Keyboard (n.d.). URL: `http://youtu.be/TyLaCza_Q5o`

Lin, J. and Kulic, D. (2011). Automatic human motion segmentation and identification using feature guided hmm for physical rehabilitation exercises, *Robotics for Neurology and Rehabilitation, . . .* pp. 1–4.

Murphy, K. P. (n.d.). Hmm toolbox. URL: `http://people.cs.ubc.ca/~murphyk/Software/HMM/hmm.html`

Pallejà, T., Guillamet, A., Tresanchez, M., Teixidó, M., Viso, a. F., Rebate, C. and Palacín, J. (2011). Implementation of a robust absolute virtual head mouse combining face detection, template matching and optical flow algorithms, *Telecommunication Systems* .

Pallejà, T., Rubión, E., Tresanchez, M. and Fernández, A. (2008). Using the optical flow to implement a relative virtual mouse controlled by head movements, *Universal Computer Science* **14**(19): 3127–3141.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition, *Proceedings of the IEEE* **77**(2): 257–286. DOI: 10.1109/5.18626

Tai, K., Blain, S. and Chau, T. (2008). A review of emerging access technologies for individuals with severe motor impairments., *Assistive Technology* **20**(4): 204 − 221. DOI: 10.1080/10400435.2008.10131947

Varona, J., Manresa-Yee, C. and Perales, F. J. (2008). Hands-free vision-based interface for computer accessibility, *J. Netw. Comput. Appl.* **31**: 357–374. DOI: 10.1016/j.jnca.2008.03.003

Viola, P. and Jones, M. J. (2004). Robust real-time face detection, *Int. J. Comput. Vision* **57**: 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb