Low-cost and Open Source Optical Capture System of Facial Performance

Gabriel P. Guarisa* Alberto T. Angonese† Sicilia F. P. P. Judice‡

FAETERJ-Petrópolis, Brazil

RESUMO

Human optical motion capture systems has been frequently used in 3D character animations and it has been proven its effectiveness due to a great realism performed, besides making the process faster. However, this technology is generally used to focus on a whole bodys motion capture, which means that it does not supports subtle motions, as facial motions. Systems and programs that make it possible are available to powerful studios or linked to other softwares of animations with restrictions. This project aims to implement a low-cost and open source optical capture system of facial performance. Therefore, the only device required is a webcam connected in a computer, and distinguishable marker on the user's face. Digital image processing techniques was also used as Camshift algorithm, which is responsible to detect the image tracking. The C++ language was chosen to implement this project with the OpenCV(Open Source Computer Vision) libraryy. The results include the capability of multiple markers selection, the data exportation into a generic output file, and the full independence of system. The low-cost and open source system proposal achieved its goal. Futhermore, the exported data are accepted by most animation programs. However, for being a low-cost design, the system has limitations of user's freedom movements. Some topics are suggested as future work, in order to improve the system proposed.

Keywords: Motion Capture, Facial Performance, Facial Animation, OpenCV.

1 INTRODUCTION

The entertainment industry always has being looking for techniques that facilitate the animation process and bring more realism to animate characters, complexity to movements and expressions [12]. A widely used method is the technique called motion capture and analysis. Research in this area is increasing since the end of last century, in view of the potential applications in various fields and advancement of computing [19]. Applications can be found in entertainment industry [5], in computer simulations [17], in sports [16] and in medicine [11]. In the last years the motion capture has being widely used in large productions of movies and games, being a form of direct animation of three-dimensional characters.

The technique of motion capture, as known by your abbreviation as mocap, is formed by the action recording process in objects or live beings [4]. In summary, the author in [14] defines the motion capture as the live movement recording process and the representation of this process in mathematical terms to be manipulated as points in space through time. When the purpose is to capture subtle movements, such as facial expressions, the more acceptable term is performance capture.

The types of motion capture are classified according to the technology used. However, the popular way of motion capture, in computer animation, uses marks in a human body, or object, and cameras to capture the movements, and this is called optical capture of motion [21]. Afterward, the generated movement is transformed into coordinates that can be applied in a computational model, to make the animation of the model.

The performance capture can be seen as motion capture complement. The performance capture by itself is a research area that is expanding with great innovations, given the importance to the characters animation, as in the case of movie and games characters animation. Nowadays, the use of motion capture is restricted to great productions or are available at a high cost, which makes the use unfeasible in academia or in small independent studios. The OpenMoCap project [2] is an exception in this scenario, being the first open source software of motion capture, however, it needs a minimal structure for it usage, which can be a barrier, given the structure costs.

In this perspective, this work presents the development of a low-cost and open source optical capture system of facial performance. The system purposed demands as physical structure only a webcam connected to a computer, and distinguishable marks in the user's face. Another research area in constant development, and directly connected to motion capture, is image processing. Because of that, the system was implemented in C++ language with the free library of computational vision OpenCV (Open Source Computer Vision), that was chosen to deal with the image processing step of the project.

The developed system has the capacity of multiple selections of marks and export of capture data in a generic format accepted by the most software animation. Another important characteristic is the independence of the system in relation to other programs, which distinguishes projects like FaceCap [22], which depends on an animation program for capture.

2 OPTICAL CAPTURE SYSTEMS

In the optical motion capture technique, one or more cameras are connected to a program that processes and extracts, frame by frame, the desired coordinates based on the movements captured. Usually, marks are used to define points of interest in a body, and they can be attached directly on the skin or in special clothes. The markers are put in joints and other points in the most significant movement of a body. In some cases markers are not used, however, scanning is also based on points of interest, based on standards such as silhouettes or striking parts of the body.

The capture technique used in this work is the optical one, since all you need is a webcam to the capture and distinguishable markers in the user face. One of the main advantages of this type of capture is the freedom given to the actors during the process, and enable the capture of more complex movements [3]. Another interesting point is the ease of changing the setting of markers in a position and quantity.

However, this kind of system has intrinsic limitations, since the markers have to be within reach of the camera, and in controlled environments with an illumination without much variation. Some optical systems facial performance capture treat the problem of markers out of reach with a helmet coupled with cameras, always getting the same surface of the user's face.

*e-mail: gabrielguarisa@gmail.com
†e-mail: angonesealberto@gmail.com
‡e-mail: sjudice@faeterj-petropolis.edu.br

2.1 General Structure for Motion Capture Flow

A structure proposed by [15] divides the capture flow into four steps, namely: initialization, tracking, pose estimation and recognition.

The initialization occurs before the system process any data and its main goal is that the system correctly interprets the scenario. In this step occurs the selection of markers or areas of interest, and any calibration should be performed as well as the definition of semantics of points, characterizing them as a structure.

The tracking is the monitoring of the coordinates of the selected points, respecting the established semantic.

The pose estimation may occur in post-processing, or as an active part of the tracking process. In both cases the objective is to group the data collected in the tracking step to create a representation of the final model. Triangulation of the cameras is made at this step, if the model is three-dimensional.

The last step is the recognition, which is the output stage, responsible for exporting of the data.

3 System

In this section are shown characteristics of the project, such as tracking, capture flow and the graphical user interface.

3.1 Tracking and Camshift

The algorithm Camshift was used to make the tracking, it uses the color as a parameter to the tracking [18].

The algorithm performs the tracking through an image with binary thresholding, where all centroids of the selected color concentrations are considered interest points, and starting with these points the tracking is performed. To do so, every frame has to be previously converted to the color system HSV (Hue, Saturation, Value). A histogram is calculated from the hue of a selected object, becoming possible the thresholding of the image.

The algorithm is implemented in the OpenCV library in the form of a function and, in the system developed, is used individually in each marker. Consequently, each marker has its own coordinate.

3.2 Motion Capture Flow

Based on the general structure of motion capture flow, presented in section 2.1, the Figure 1 shows the flowchart which presents the implemented steps in the system, adapted according to the needs of the system.

The first step, initialization, consists in selecting all the markers and defining the semantic of the model. This is an initial moment in the execution of the project, and is a process of preparation to the capture, because, if not done properly, the data will be incoherent, or even not exist.

The position of the selected markers is recalculated each frame, in the first step, keeping in mind that the user can select the markers with the video playing. This calculus occurs together with the first step of the tracking, but, no information of the position of the markers is stored while the process of the motion capture is not started.

When the capture is started, the tracking of the markers occurs and the position is calculated according to the semantic set. As explained before, the estimation of the pose can be in the post-processing, or be an active part of the tracking, as in the case of the implemented system. Afterward the final representation position is locally stored.

The recognition is the moment of the output of the stored data. The semantic of the model and all the poses through time are exported, in this case, as a file.

3.3 Graphical User Interface

All the graphic user interface present in the project was implemented using the model of Interface Control and highgui input devices, present in the OpenCV library. Two windows are constructed in the beginning of the system, one called "Original", that show the frames obtained by the camera, and another called "Trackbar", with the controls of the reach of the image in color system HSV.

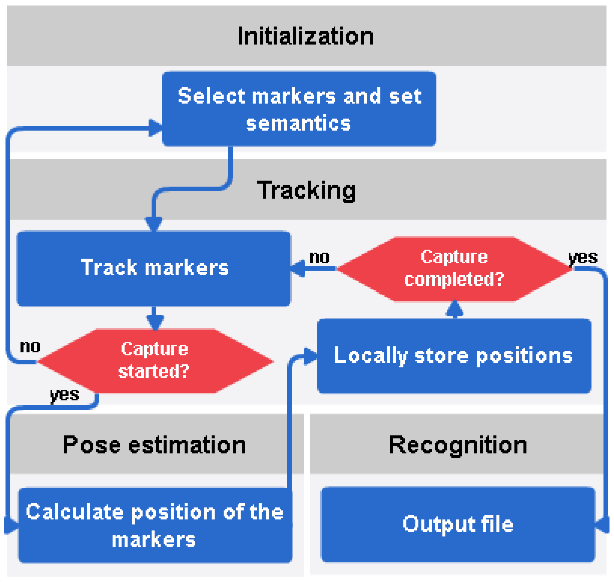


Figura 1: Motion capture flowchart implemented in the system.

The definition of the stage of capture and the control of the program are made through keyboard commands. The possible commands are listed in Table 1.

Tabela 1: Keyboard command.

Command	Description
1	Calibration.
2	Starts capture.
3	Stops capture.
Esc	Close the program.
H	Show/hide help elements.

The mouse commands only can be used in calibration. The left click is used to select marker and the right click to remove marker.

4 Implementation

The system described in this paper was implemented in the Windows 10 operational system, 64-bit version, and using the Visual Studio Community 2015 IDE. The modules of computational vision, image processing and graphical user interface, from OpenCV library, were used in the implementation of this project. The version used is 3.1, released in 2015.

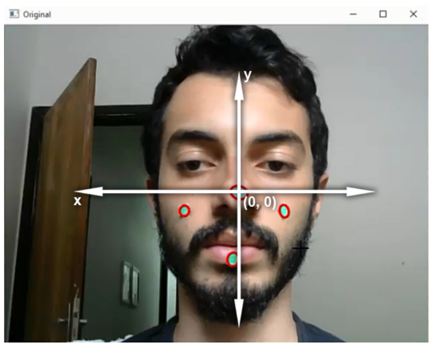
4.1 Selection and Semantic of Markers

In optical motion capture the markers have to be discrepant with the rest of the scene. In the context of the implementation of the system, it means that the marker has to be of a contrasting color, and other than black or full white.

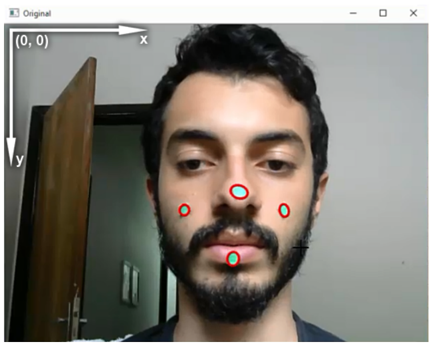
In the implemented system, the selection of the markers is based on the histogram of the selected point of the image. To select a marker the user must click in the corresponded area in the Original window.

The semantic of the markers in the system is based on the one used by Xantus Animation Studio on FaceCap [22], using a marker in the nose as a central point, being the first to be selected. The purpose of this is to serve as a reference for the others markers in the estimation of the final pose. This means that all positions of the markers are related to the position of the marker of the nose that, in the final model, will always be $(0,0)$.

4.2 Pose Estimation



(a) Position of the markers relative to the center.



(b) Position of the markers in the image.

Figura 2: Position of the markers on the screen.

It is important to notice the difference between the position of the marker in the final model, represented in Figure 2a, and the position of the marker in the image, present in Figure 2b. The position of the marker in the image is calculated through the tracking algorithm, while the position of the markers in the final model are calculated individually, and is defined in the Algorithm 1, where `marker` is a vector with all the markers.

Algorithm 1 Algorithm to determine the position of the marker in the final model.

```
1:  for $i \leftarrow 0, MAXMARKERS$ do
2:      if $marker[i].isSelected()$ then
3:          if $i == 0$ then
4:              $marker[i].x \leftarrow 0$
5:              $marker[i].y \leftarrow 0$
6:          else
7:              $marker[i].x \leftarrow marker[i].rect.x - marker[0].rect.x$
8:              $marker[i].y \leftarrow marker[0].rect.y - marker[i].rect.y$
9:          end if
10:     end if
11: end for
```

The Algorithm 1 presents the way the position of the final markers are calculated. There is a loop that will repeat until the maximum number of markers defined in the system. There are two consecutive verifications, one concerns the marker being used and, if true, a second verification is done to determine if the actual marker is from the nose, defining it as a fixed point in the final model. If the selected marker will not be the central, the final position of the marker is calculated based on the position of the marker of the nose in the image.

4.3 Output File

Whenever tracking is completed a file called `output.bvh` is created in the root directory, and the program will return at the calibration stage, for future trackings, or closing the program. The BVH (Biovision Hierarchical) format, developed by the company BioVision [1], is formed by ASCII text, which allows the creation of a common text file, with the extension `.bvh`.

The SDK available by [13] was used as a basis to implement the output file generator. It is important to highlight that the SDK code had to be adapted according to the system requirements, and corrected, as was flawed.

Files `.bvh` allow the use of the third axis, however, as the system only uses one camera, there is no depth in movement. In the developed system all values related to third axis are defined as zero.

5 RESULTS

The main results achieved in the development of the system are described below, along with the limitations encountered.

5.1 Detection of Markers

A key point in the system is the detection of markers and, as described above, is based on color. This becomes important, since the data of the capture are extracted from these markers throughout the program execution. Another key point is the selection of multiple markers, both are exposed in Figure 3.

5.2 System Execution

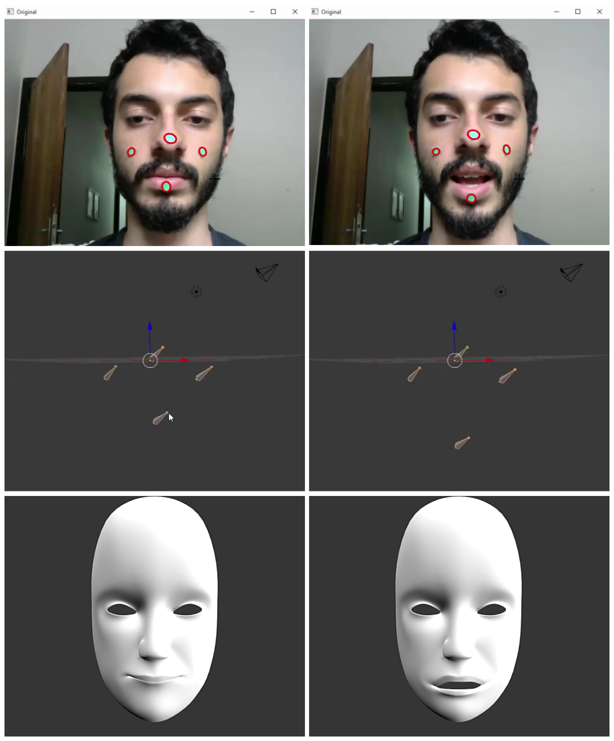


Figura 3: Frames from the motion capture and their respective animations. Free 3D model available in [20].

Examples of the use of the system are available in [6] and [7]. In the two examples cited is possible to see all the motion capture process being executed through the system developed. Besides that, the resulting output file from the motion capture is imported into Blender, where is possible to visualize the movements captured in an animation. Other motion capture process is available in [8] and the resultant animation in [9].

In Figure 3 are presented some frames from the motion capture and their respective frames from the resulted animation.

5.3 Independence

The independence of all facial motion capture process is the main characteristic achieved in the development of this system. This factor is what differs our system from other applications, as the Face-Cap [22], connected to Maya, and the motion capture module present in Blender, because it allows the execution of the facial motion capture without the need of others softwares.

5.4 Availability System

The system proposed and implemented in this paper in open source. It means that future changes in the code are expected. For this reason, the project is available in a public repository [10]. This choice is based on the possibility of versioning, and be available within easy reach, allowing future collaborations.

5.5 System Limitations

Some limitations are intrinsic, given the proposal of the project developed to be low cost. One example, already mentioned, is the depth axis, since the system uses only one camera.

Another factor that affects the end result is the resolution of the camera. In most cases, webcams embedded in notebook computers are low resolution and low cadence. The low-resolution camera prevents accurate tracking, which is an important factor in performance capture. The fact of the low cadence varies depending on the camera model.

The accuracy of the coordinates of the markers is another factor that is related to the webcam. This can occur by the low resolution of the captured image combined with the irregular incidence of light, resulting in ill-defined colors and generating oscillation in the final model's movement.

The system implemented for the tracking of the markers may be impaired or made impossible due to some factors that, beyond the camera, are the user's influence and/or the environment. These factors can involve inadequate lighting, marker out of camera range, very accelerated movements, unfeasible size marker, radial movements.

6 Conclusions and Future Works

Currently, optical motion capture has gained more and more space in the middle of big productions. However, there are few open source projects, or affordable in the area. Partly, this is a result of using multidisciplinary techniques. An area in constant development, and directly linked to motion capture, is the image processing, which has been widely used in the development of this work, especially with the help of OpenCV library.

The main objective achieved in this work was the development of an optical system for capturing facial performance, low cost and open source. The importance of this project is the spread of technology in academic and independent means. In addition, it serves as a starting point, indicating the state of the art.

The fact that the data is exported in a format acceptable by most animation programs and the program is independent, value the project. However, treatment of the user's movements were partially treated. Small movements are corrected, however, more complex movements, such as radial, are not treated. This causes the user the obligation to become immobile as possible so that the data is not compromised.

The intention is that these limitations will be treated in the future. Some ideas for possible improvements and additions arose during the development of the system, however, there was not enough time to implement them. Some of these ideas include treat more complex movements of the user, develop more efficient algorithms for tracking, implement a system of marking points which supports a third axis, soften the acquired coordinates, giving more realism to the movement.

Referências

[1] BioVision. http://www.biovision.com, 2015.

[2] D. L. Flam. Openmocap: Uma aplicação de código livre para a captura óptica de movimento. Master's thesis, Universidade Federal de Minas Gerais, 2009.

[3] M. Furniss. Motion capture: an overview. *Animation Journal*, 8(2):68–82, 2000.

[4] J. V. B. GOMIDE. Captura digital de movimento no cinema de animação. Master's thesis, Universidade Federal de Minas Gerais, 2006.

[5] A. Gray. A brief history of motion-capture in the movies. http://www.ign.com/articles/2014/07/11/a-brief-history-of-motion-capture-in-the-movies, 2014.

[6] G. P. Guarisa. Facial mocap using opencv. Video available in: https://www.youtube.com/watch?v=bm_zzgo9Ucw.

[7] G. P. Guarisa. Facial mocap using opencv v2. Video available in: https://www.youtube.com/watch?v=DjdiNKS8ZKU.

[8] G. P. Guarisa. Facial mocap using opencv v3. Video available in: https://www.youtube.com/watch?v=z3eLoCnUqFo.

[9] G. P. Guarisa. Mocap results. Video available in: https://www.youtube.com/watch?v=6rycxPtSyAU.

[10] G. P. Guarisa. Project available in repository: https://github.com/gabrielguarisa/facialMocap, 2016.

[11] J. Lasenby. Medical applications of optical motion capture. http://www-g.eng.cam.ac.uk/mmg/lifesciences/lasenby.html, 2004.

[12] F. S. LUZ. Animação digital: Reflexos dos novos médias nos conceitos tradicionais de animação. 2009.

[13] Manjunath. http://www.mindfiresolutions.com/BVH-biovision-hierarchy.htm, 2015.

[14] A. Menache. *Understanding motion capture for computer animation and video games*. Morgan Kaufmann, 2000.

[15] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer vision and image understanding*, 81(3):231–268, 2001.

[16] Nike. Nike sport research lab incubates innovation. http://news.nike.com/news/nike-sport-research-lab-incubates-innovation#/inline/21646, 2013.

[17] Organic Motion. Live for training and simulation. http://www.organicmotion.com/live/, 2013.

[18] A. Salhi and A. Y. Jammoussi. Object tracking system using camshift, meanshift and kalman filter. *World Academy of Science, Engineering and Technology*, 64:674–679, 2012.

[19] A. Sharma, M. Agarwal, A. Sharma, and PankhuriDhuria. Motion capture process, techniques and applications. 2013.

[20] Shawnahall. Free 3d model. Disponível em: http://www.turbosquid.com/3d-models/face-dae-mtl-dxf-free/487745.

[21] F. Silva. Motion capture - introdução à tecnologia. *Laboratório de Computação Gráfica, LCG. COPPE, Universidade Federal do Rio de Janeiro, UFRJ*, 1997.

[22] Xantus. Facecap. Project available in repository: https://github.com/XantusStudio/FaceCap, 2012.