https://doi.org/10.26552/com.C.2016.1.56-61

Peter Kajan - Patrik Kamencay - Jan Hlubik - Robert Hudec - Miroslav Benco - Peter Sykora *

REAL-TIME FACIAL MOTION CAPTURE USING A WEBCAM

In this paper, we create a software which would be able to combine marker and marker-less face tracking for the purpose of the creation of the accurate and real time running 3D facial animation inside the Cinema 4D. The aim is develop a powerful piece of the software that can provide facial motion capture and share the motion coordinates on the server in real time. The software solution includes tracking software which is written in C++ and provides real time face tracking. Tracking data are sent to the server in real time. Main benefit is that this solution can provide tracking data on the server for any third software. At least, the smooth run of the above mentioned real time tracking software requires simple color webcam with VGA resolution, running on 30 frames per second and a multi-core processor including at least two threads.

Keywords: *Facial Motion Capture, C++, face tracking, 3D facial animation.*

1. Introduction

The motion capture is defined as the process of recording the motion of objects, people or animals. It is used in different areas including robotics, military, sports, medical applications and entertainment. Facial expressions of the human actors are recorded and processed by means of the computer vision algorithms, which provide us with required information about the motion. This information is used to animate the digital character within 2D or 3D computer animation. Face motion capture is focused only on the facial motion of an actor. For this purpose, in order to capture facial expressions, many different approaches are used. This capture can be done in two or three dimensions, depending on the number of the cameras, which are used. To get full three dimensional information about the play of the actor's face and rotation of the head professional solutions use multi camera rigs or laser marker systems. On the other hand, two dimensional solutions use only one simple camera which cooperates with suitable software and gets two relative dimensional coordinates of the facial elements, such as lips, eyes, eyebrows [1].

The aim of this paper is the practical verification of the theoretical knowledge in the field of the computer vision and image processing. With using only one simple webcam we experimentally verified the suitable software technology for the real-time face tracking and the real-time 3D facial animation [2]. The theoretical part describes the fundamentals of the digital image processing with focus on the facial motion capture. The practical part uses this knowledge in order to create solution for facial motion capture software. This software provides the face tracking in real time and uses the tracking information in order to provide the facial animation in the environment of the Cinema4D.

The rest of the paper is organized as follows: In Section 2, state of the art is briefly reviewed. The design and creation of the face tracker is discussed in Section 3. The conclusion and comparison of our face tracker with other commercial systems is described in Section 4.

2. State of the Art

Computer vision is a part of the artificial intelligence concerned with computational understanding and processing of the visual information provided by digital video cameras. The main goal of the computer vision is to duplicate the abilities of the human vision by means of the electronically perceived and understood digital image. This enables computers and robots to see and understand in nearly the same way as human beings do. Within the frame of the computer vision, such as object tracking, we are able to solve some specific tasks with individual algorithms [3].

2.1 Object Tracking

Object tracking can be defined as a problem of how to estimate the trajectory of the specific object in the image

* **Peter Kajan, Patrik Kamencay, Jan Hlubik, Robert Hudec, Miroslav Benco, Peter Sykora**
Department of Telecommunications, Faculty of Electrical Engineering, University of Zilina, Slovakia
E-mail: patrik.kamencay@fel.uniza.sk

plane, as it moves around the scene in time. Object tracking is a complex process which can be split into several categories. The main task of the object tracker is to generate over time trajectory of the object by locating its position in every frame of the video. There are two methods how the tracker can provide the detection and establish the correspondence between the objects across the frames. First of them establishes the correspondence separately. In every frame, the possible object regions are obtained by means of the object detection algorithm and then the tracker makes correspondence between the objects across the frames. The second method performs detection and correspondence at the same time. At the same time the object region and correspondence are estimated by iteratively updated object location and region information. All information is obtained from the previous frame [4].

2.2 Object Detection

Every tracking method requires a mechanism for the object detection. The common approach for the object detection is that we try to detect a particular object separately, in every single frame. To avoid detection errors the most advanced methods of the object detection evaluate the object position in a few consecutive frames. The most popular detection methods are listed below [4]:

- Point detectors - Point detector finds interesting points for the tracking in images. Interesting points for the tracking are usually some corners and areas with a significant texture. Commonly used interest point detectors are Harris interest point detector and SIFT detector.
- Background subtraction - The background subtraction is a method based on the assumption that the tracking objects move between the frames and that the background remains static. The tracking object is detected by this background subtraction.
- Segmentation - The segmentation algorithm splits the image into perceptually similar regions. There are two very important issues the segmentation algorithm has to deal with. First, it has to set up the right criteria for a good partition, and second, it has to achieve an efficient partitioning.

3. Design and Creation of the Face Tracker

The proposed face tracker is a real-time face tracking application which has been developed for the purposes of this paper. This face tracker uses two independent threads which work simultaneously, i.e. at the same time. One of them is the main thread, which manages the whole process and provides all the necessary tasks and resources which are requested for the second thread. The second tracking thread provides the tracking for each frame and after the processing of the tracking data, these data are sent to the server.

Very important part of the proposed face tracker is the tracking core. The core is split into two parts. The first part includes detection modules and the second part contains the data processing modules. This second part is called data processing part.

3.3 Detection Modules

The main task of the detection modules is to correctly detect the absolute position of the important face elements in the current frame. The flowchart of the detection modules is illustrated in Fig. 1.

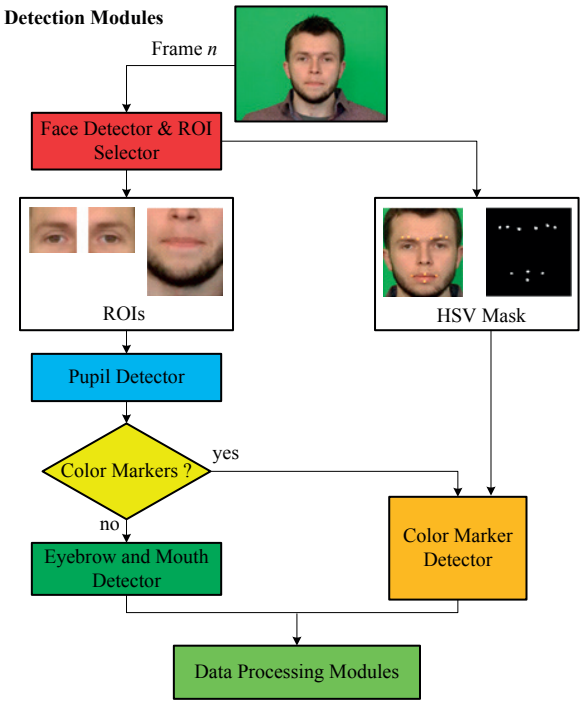

Fig. 1 Flowchart of the detection modules

There are two fundamental modes. First mode detects the face features without any color markers. This marker-less mode includes eyebrow and mouth detector. Second mode is used for more accurate tracking. This mode requires footage with color markers on the actor's face. As it is impossible to attach the markers on the eyes, it is the pupil detector which is always used for both modes.

3.3.1 Face Detector & ROI Selector

In order to detect the human face and other facial features, such as mouth or eyes, the Haar classifier cascades have to be trained. The OpenCV library contains xml files with classifiers which are trained for the human face and other facial features. In our case, we used the haarcascade_frontalface_alt.xml. This cascade is used only for the face detection with a very good accuracy – of more than 80%. The OpenCV function for the detection of the multi-scale object is used in the Face Tracker. The function returns in the vector of the rectangles. Each rectangle represents the detected face area of each face in the image. In our case, we assume only one face which is supposed to be in the image, so the function supposes to return in the vector which contains only one rectangle. In that case, more than one face will appear there, and the biggest one will be selected. The detected face is shown in Fig. 2.

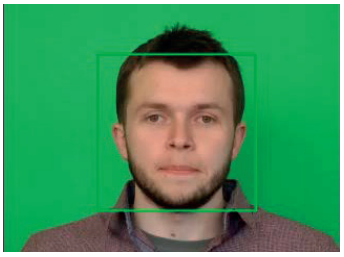

Fig. 2 Detected face ROI

Next step concerns the selection of the regions of interests (ROI) for eyes and mouth (see Fig. 3). The OpenCV Haar-Cascade also allows us to detect facial features, such as nose, eyes and mouth. Although this detection is quite accurate, a huge computational power is required. In order to keep the computational time as small as possible we have to take a different approach. According to the known proportions of an average human face, the ROIs are defined for each part.

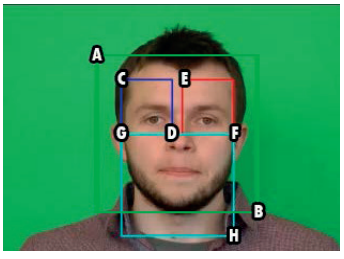

Fig. 3 ROI selection for each part of the face

Every ROI can be described by two 2D points. The width of the face rectangle is defined as follows:

$$W = Bx - Ax \tag{1}$$

The computation of other ROIs is defined below:

$$\begin{aligned} C \cdot x &= A \cdot x + (W * 0.15) \\ C \cdot y &= A \cdot y + (W * 0.15) \end{aligned} \tag{2}$$

$$\begin{aligned} F \cdot x &= A \cdot x + (W * 0.85) \\ F \cdot y &= A \cdot y + (W * 0.5) \end{aligned} \tag{3}$$

$$\begin{aligned} D \cdot x &= C \cdot x + \left(\frac{F \cdot x - C \cdot x}{2.2}\right) \\ D \cdot y &= F \cdot y \end{aligned} \tag{4}$$

$$\begin{aligned} E \cdot x &= F \cdot x + \left(\frac{F \cdot x - C \cdot x}{2.2}\right) \\ E \cdot y &= C \cdot y \end{aligned} \tag{5}$$

$$\begin{aligned} G \cdot x &= C \cdot x \\ G \cdot y &= D \cdot y \end{aligned} \tag{6}$$

$$\begin{aligned} H \cdot x &= F \cdot x \\ H \cdot y &= B \cdot y + (W * 1.15) \end{aligned} \tag{7}$$

where A, B, C, D, E, F, G, H are ROIs and W is width of face rectangle.

In order to achieve the best possible results, the constants in the above expressions are the results of many measurements and tests. Once the ROIs are defined, the facial features can be detected on the inside of those areas.

3.3.2 Pupil Detector

Correct and accurate pupil detection is crucial for the face tracking. Besides this, we can also detect the eye gaze and define more accurate ROIs for eyebrows and the relative vertical position of both pupils gives us very important information about the head rotation. There are many different approaches how to detect the eye pupil.

The pupil detector is based on the Cumulative Distribution Function (CDF) algorithm. This method assumes that the pupil is much dimmer than the cornea. The CDF algorithm is defined by:

$$CDF(r) = \sum_{w=0}^{r} p(w) \tag{8}$$

where $p(w)$ is probability to find a pixel with luminance which is equal to w.

Fig. 4 The pupil detection procedure

In the first step the ROI area is blurred and inverted. Secondly, the CDF function is applied. The next step is to restrict the ROI area only to the eye. The main reason for this restriction is that some people have very dark hair or eyebrows, which can be even darker than the pupil itself. The right down corner is blended. This could easily confuse the algorithm. After that the circle mask from the right down corner discloses the ROI area. In each step, the maximum value of the whole ROI is measured. Once the pupil is uncovered, the maximum value is reached. When the next value is smaller than the previous one the algorithm stops. This phase is illustrated in Fig. 4b. Next step is to measure the maximum value of the eye area. Then, a threshold of, for example, 95% of the whole range between min and max value is applied. The pupil candidate, which has best rate of all of those facilities, is selected as the pupil.

3.3.3 Eyebrow Detector

After detecting the pupil positions, the eyebrow ROI can be defined. The eyebrow ROI occurs inside the eye ROI with lower border above the detected eye. To detect the eyebrow shape reliably, the Sobel derivatives are used. The Sobel derivatives allow us to detect the edges in the gray scale image in horizontal or vertical direction. The eyebrow detector itself consists of two parts. The first part is the coarse detection of the eyebrow. The ROI is strongly blurred and the vertical Sobel is applied. The result is illustrated in Fig. 5b. The main reason for the strong blur filter is to eliminate the wrinkles and other features which can be considered to be the eyebrow (see Fig. 5g). After applying the adaptive threshold, the eyebrow´s thickest part is detected. Let us call this blob anchor. The anchor will be used later in the second part of the eyebrow detector. The second part provides the most accurate results in Fig. 5f. The Gaussian blur and then the Sobel filter are applied to the ROI image.

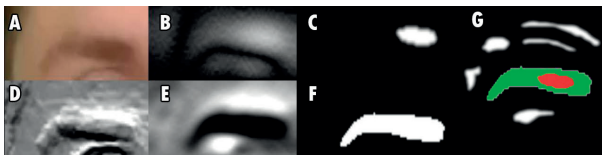

Fig. 5 Visual illustration of the eyebrow detection

To select the right candidate, the distance between the candidates and the anchor is measured. The candidate, which is positioned nearest the anchor, is the right one. Once the shape is clearly detected, the OpenCV function of find contours is applied. This function returns in the vector of points. Those contours points are illustrated in Fig. 6.

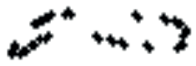

Fig. 6 The eyebrow contour points

At least a simple sorting algorithm is applied. For example, to get the right corner of the eyebrow, all points are upwardly sorted according to their horizontal position. The first three points will be selected and a 2D space centroid will be calculated. This centroid represents the right corner of the left eyebrow. As the centroid represents the average position of the three points, it will eliminate the disturbing noise.

3.3.4 Mouth Detector

The mouth detector consists of two parts. First part is color-based and provides the color transformation of the mouth ROI from the RGB color space to the gray scale I defined by transformation formula:

$$I_{(x,y)} = \frac{2G_{(x,y)} - 2R_{(x,y)} - 0.5B_{(x,y)}}{4} \quad (9)$$

where R, G, B are color components of a certain pixel in the mouth ROI and I is the illumination value of the result pixel. The transformation is based on the principle that the blue component has a reduced role within the lip and skin discrimination.

Fig. 7 Lips detection procedure

The mouth ROI after the color transformation is illustrated in Fig. 7b. The holes of the lips and nose are clearly separated from the background. When we add a binary threshold to the transformed image, we get a clean binary mask. Once we have the mask, the function of the find contours is applied. As the contours of the lips are very rugged, the approximation of an envelope is needed. The application of the convex hull function on the vector of the contours points returns to a smooth hull of the lip, Fig. 7c. This hull is also defined as the vector of points. These points are illustrated in Fig. 7d and are sorted and used to determine the corners of the lips.

3.3.5 Color Marker Detector

The above mentioned mouth and eyebrow detectors have some limitations. They are very sensitive to bad light conditions, like strong side illumination of the face. For example, if the light source has a red toned light spectrum,

this red light, which is reflected on the cheeks, can be wrongly detected as a part of the lips. The indistinctive eyebrow will not be probably detected correctly either. To avoid such mistakes and limitations, the color trackers which are glued to the actor's face are used. The trackers should have a unique color, which is clearly separated from the actor´s face and background. In the first step, the face ROI is transformed from RGB to the HSV color space. HSV (Hue, Saturation and Value) simply allows us to separate the trackers from the background. The lower and upper value of the threshold for each HSV channel is defined by the user. After that, the image is transformed into the binary mask, which is illustrated in Fig. 8.

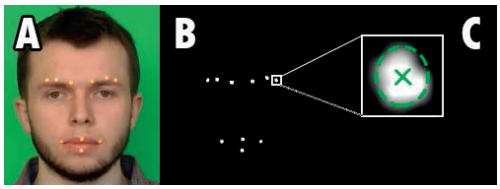

Fig. 8 The color point masking and detecting procedure

For every white blob, which represents one color tracker, the contour points are computed in the same way as in the previous cases. The contour points can be bounded by ellipse calling function fit Ellipse. The centre of the bounded ellipse is the centre of the pertaining tracker (see Fig. 8c). Detected points are sorted and put in the right order which represents the eyebrows and mouth.

3.3.6 Data Processing Modules

Processing modules transform the absolute positions of each tracking point to get some useful piece of information for the next handling. This part includes many tasks, such as alignment, relative positioning, stabilization, gaze detection, etc.

- Eye Gaze Detector - The eye gaze detector provides the eyes tracking. In our case, only the horizontal motion of the pupils is investigated.
- Trackers Smoother & Fixation Module - Besides the quantization noise, the position errors occur with the tracking process among the frames. These disturbing motions and errors should be minimized by smooth algorithm. This smooth algorithm is based on the principle of the memory buffer. In each frame, the values in the buffer shift one position down and the current value is added to the top. The last value is removed.
- Frame Drawer Module - The drawer module is required to give the user proper feedback and information about the tracking accuracy.

4. Conclusions

In this paper, the software which would be able to combine marker and marker-less face tracking for the purpose of the creation of the accurate and real time running 3D facial animation inside the Cinema 4D was created. Within the field of computer vision, the facial motion capture is a complex problematic which can afford a lot of challenges. The most difficult part is to learn the computer to be able to reliably recognize the facial features and to estimate their motions. The computer vision algorithms are very sophisticated and require a lot of computing power. In order to create a real-time face tracking software, many difficult tasks had to be solved, including multithreading, effectiveness and accuracy. However, the C++ programming language with OpenCV libraries and Qt Creator allow us to create a very stable and powerful application with minimal computing power demands. Nowadays, the movie and advertising businesses increasingly work with computer generated imagery (CGI). Furthermore, the game industry also uses the computer vision and motion capture systems in order to provide more realistic and enjoyable games.

For facial capture, a few commercial products have already been available. The proposed Face Tracker meets almost all requirements for the face tracking software. It does not need any special additional hardware and it has a very few computational requirements. The proposed system smoothly runs on double cored Intel Core i5 with 2 GB RAM.

Acknowledgements

This contribution/publication is the result of the project implementation at the Centre of excellence for systems and services of intelligent transport, ITMS 26220120028 supported by the Research & Development Operational Programme funded by the ERDF.

References

[1] GANAPATHI, V., PLAGEMANN, C., KOLLER, D., THRUN, S.: *Real Time Motion Capture using a Single Time-of-flight Camera,* IEEE Intern. Conference on Computer Vision and Pattern Recognition (CVPR), pp. 755-762, ISBN 978-1-4244-6984-0, 2010.

[2] AGARWAL, A., TRIGGS, B.: *Recovering 3D Human Pose from Monocular Images,* Image Processing, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, No. 1, pp. 44-58, ISSN 0162-8828, 2006, doi: 10.1109/TPAMI.2006.21.

[3] SPIRIK, J., ZATYIK, J.: *Image extrapolation using sparse methods*, Communications - Scientific Letters of the University of Zilina, vol. 15, No. 2A, pp.174-179, ISSN 1335-4205, 2013.

[4] MARKOVIC, I., CHAUMETTE, F., PETROVIC, I.: *Moving Object Detection, Tracking and Following using an Omnidirectional Camera on a Mobile Robot,* IEEE Intern. Conference on Robotics and Automation (ICRA), pp. 5630-5635, 2014, doi: 10.1109/ICRA.2014.6907687.