AUTOMATYKA • 2010 • Tom 14 • Zeszyt 3/1

Jaromir Przybyło*

Hand Tracking Algorithm for Augmented Reality Systems

1. Introduction

Human-Computer Interaction (HCI) is an essential field of science, aimed at providing natural ways for humans to use computers. Recently, the Augmented Reality (AR) systems have become very popular [3]. They can be applied in many areas – education, medical visualization, maintenance and repair, annotation, robot path planning, entertainment, advertisements, and military applications. According to Azuma [4], the Augmented Reality is a variation of Virtual Reality (VR). VR technologies completely immerse a user inside a synthetic environment. While immersed, the user cannot see the real world around him. In contrast, AR allows user to see the real world, with virtual objects superimposed upon or composited with the real objects. It combines computer-generated 3D imagery with real environment observed by camera, usually in real-time. Figure 1 shows an example of such system.

The essential part of AR is possibility of user interaction in real-time, with computer generated content superimposed with real image [7]. In many AR applications image of the observed scene, contains view of the user or his arms [12]. Therefore, using the hand gestures seems to be natural way of interaction.

There are two main approaches to gesture recognition. First group of methods use sensor devices for digitizing hand and finger motions [1, 6]. However, these devices are quite expensive. The second group utilizes video from camera, which enables more natural interaction between humans and computers without the use of any extra devices [9].

Many challenges in vision-based hand gesture recognition exist [8, 11]. One of them is accurate tracking of users' hand, independently of scene complexity and illumination. The second problem is initialization and calibration of algorithms parameters. In this paper, we propose a tracking algorithm for AR system. It is a part of a larger project aimed at constructing novel user interface for AR applications.

* Institute of Automatics, AGH University of Science and Technology, Krakow, Poland

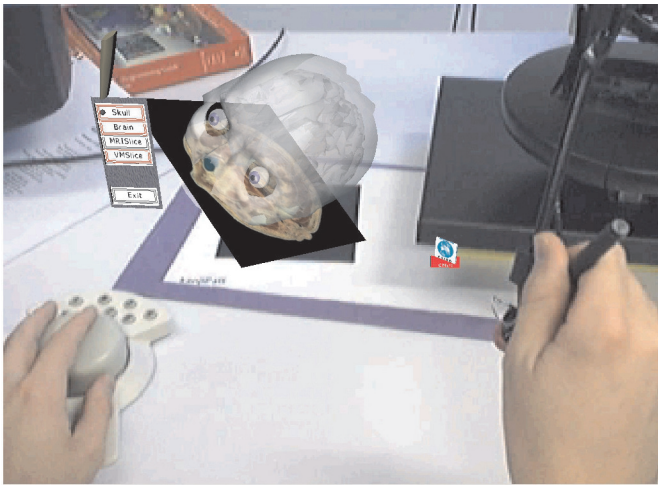

Fig. 1. Example of Augmented Reality scene, "© Copyright CSIRO Australia, (2003–2004)"

2. Tracking algorithm outline

Numerous algorithms use skin-color as one of the basic features for detecting and tracking human face or hands [2]. Widely known example is CAMSHIFT algorithm [5], used in many applications, especially for tracking human faces. It operates on color probability distributions and is able to track objects in a video scene in real time.

a) b)

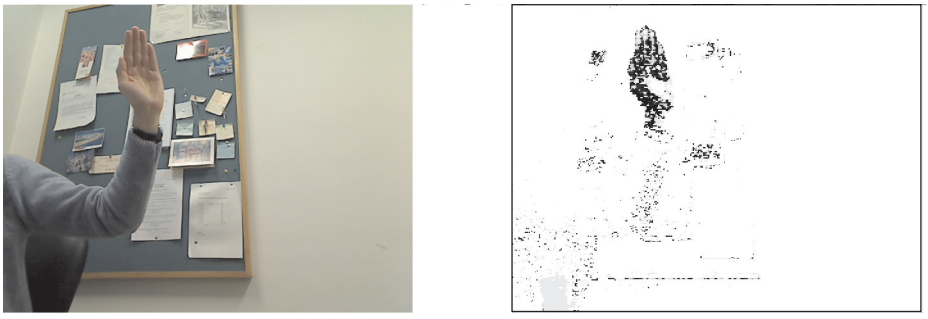

Fig. 2. Example of complex AR scene (a) and corresponding skin-color probability image (b)

The CAMSHIFT algorithm is simple, computationally efficient and works well with various scenes. However, typical AR camera view includes image of hands, face and possible other objects with skin-similar-color (Fig. 2). In such scenario, the tracked object usually is lost and tracker is attached to wrong object. Also, CAMSHIFT search window

can expand to include all skin-color parts of image. From the AR user perspective, this is inconvenient and usability of such a system decreases significantly.

Recent studies [10] suggest that the human visual system integrates color signals not only at the same location, but also along the motion trajectory. Therefore, we propose extension to the CAMSHIFT algorithm – to use both color and motion information. To our knowledge, such an approach has not been used in AR systems. The algorithm outline is given in Figure 3.

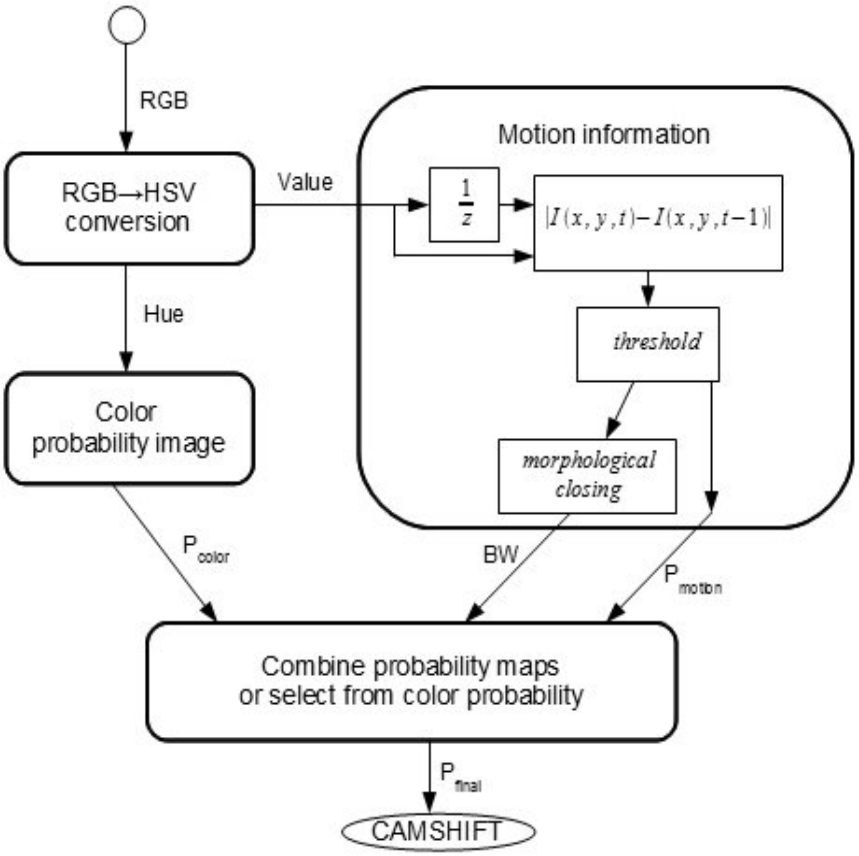

Fig. 3. Block diagram of tracking algorithm

For each frame of video signal, the RGB image is converted to a Hue-Saturation-Value (HSV) color system. HSV space separates out hue (color) from saturation (how concentrated the color is) and from luminance. Then, a color probability distribution image P_{color} (Fig. 2b) is computed from *Hue* component via a color histogram model (LUT – look-up--table). LUT contains information about color of the object being tracked (hand, skin-color) and is created during initialization phase.

a) b)

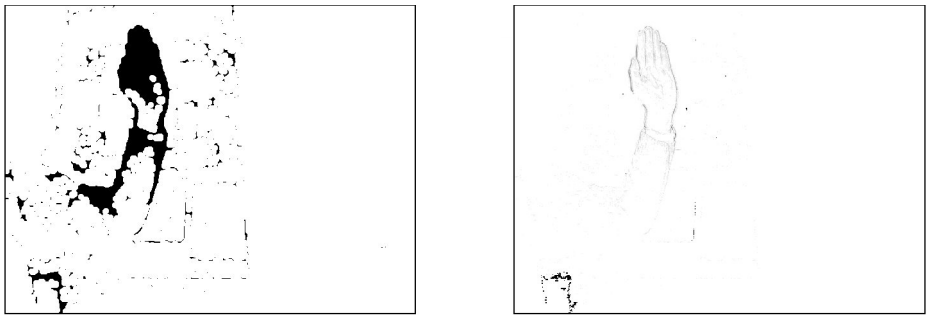

Fig. 4. Binary motion map (a) and motion probability image (b)

Motion information is extracted from luminance component by differencing previous and the current image frame. To suppress influence of image noise, the motion threshold has been introduced – equation (1). As a result binary motion map BW is created (Fig. 4a). This map is then used to select pixels from motion probability image P_{motion} (Fig. 4b) according to equation (2).

$$BW = \begin{cases} 0 & abs(D(x, y, t)) > thres \\ 1 & abs(D(x, y, t)) \le thres \end{cases} \tag{1}$$

$$P_{motion} = \begin{cases} 0 & BW = 0 \\ abs(D(x, y, t)) & BW = 1 \end{cases} \tag{2}$$

where:

$D(x, y, t)$ – image difference: current frame t and previous frame $t - 1$,
BW – binary motion map,
P_{motion} – motion probability image.

When image noise is large (especially in the case of using low-quality camera or in low-light conditions) median filtering can be optionally applied to binary motion map. Moving hand usually gives strong response only in the edges of the object, not inside it – where skin-color is more evident. Therefore, binary morphological closing can be applied to BW image before pixel selection.

To integrate color and motion information, each pixel of final probability map P_{final} is set according to the following formula (3).

$$P_{final} = \begin{cases} 0 & BW = 0 \\ \alpha \cdot P_m + (1 - \alpha) \cdot P_{color} & BW = 1 \end{cases} \tag{3}$$

where:

P_{final} – final probability image normalized to < 0–255 > range,
P_{color} – color probability image,
P_m – motion probability image,
α – blending factor,
BW – binary motion map.

Selecting pixels only from locations where motion occurs ($BW = 1$) enables integration of color along the motion trajectory. The blending factor α allows mixing of motion and color information with different proportions (Fig. 5). For example – when $\alpha = 0$, only color information is used.

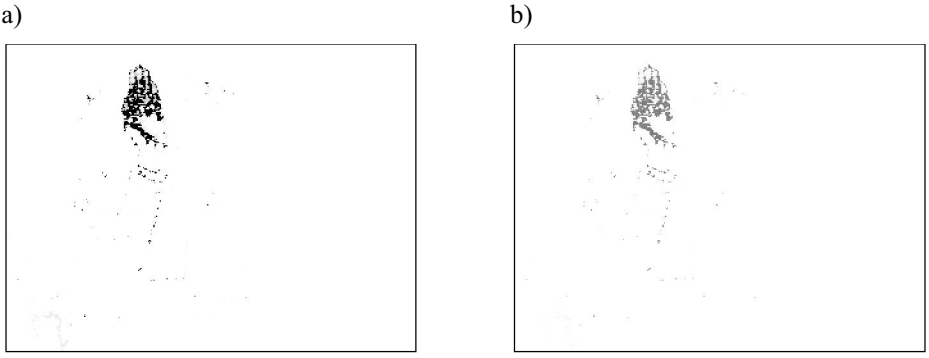

Fig. 5. Final probability map: $\alpha = 0$ (a) and $\alpha = 0.5$ (b)

To track object, CAMSHIFT algorithm with several modifications is used:

- the search window is fixed and cannot change its size,
- update of the mean position within the search window is done only, when zeroth moment is greater than threshold (this reduces position noise),
- the skin-color model and starting position are initialized interactively by moving hand to selected area of the image and making predefined gesture.

3. Experimental results

The proposed algorithm has been modeled and tested on MATLAB/Simulink platform running on Windows-based PC. MATLAB is not suitable to run complex vision algorithms in real-time (Windows operating system is not real-time system as well). Therefore, tests are performed offline on video sequences. The algorithm has been also tested on images grabbed directly from camera, however low frame rate restricts user interaction to slow hand movements.

Because tracking algorithm was designed for AR system, video sequences used in tests include several simple gestures making up example scenario of interaction. First gesture

(Fig. 6a) is user's hand movement from top to bottom of the screen. The second one (Fig. 6b) is similar, however movement is done horizontally several times. These two types of gestures can be used for example, to move virtual slider or object. Finally (Fig. 6c) there are three pointing gestures, when user moves hand with index and middle finger pointing at selected objects seen on the displayed image.

Our main requirement was that tracking algorithm should work well under various conditions. Therefore, test video sequences have been taken by different cameras in different locations (lighting conditions, background complexity, etc):

- video No. 1: good quality USB camera (logitech), complex background, uniform lighting conditions (fluorescent lamps);
- video No. 2: good quality USB camera (logitech), complex background, complex lighting conditions (overhead fluorescent lamps and rear daylight illumination);
- video No. 3: medium quality USB camera (lenovo integrated), complex background, complex lighting conditions (overhead fluorescent lamps and right-side daylight illumination).

The color model for each sequence was initialized interactively.

Fig. 6. Three types of gestures: moving hand vertically, video No. 1 (a); moving hand horizontally, video No. 2 (b); pointing gesture, video No. 3 (c)

To asses if combining color and motion information improve performance of the algorithm, tracking results were compared with manually annotated positions of user's hand. Tests were carried out using color probability image (as is original CAMSHIFT) and with combined color and motion probability image. Figure 7 shows position error (computed as euclidean distance between tracked and annotated positions) of the selected test sequence.

Tracking based only on color information works well when there is only one skin-color object on the scene. However, as it was expected, when there is more than one object – tracker fails. It is a significant limitation for AR system where the user, for example, moves his hand in front of face or the scene background is complex. This problem can be solved by implementing several independent trackers with heuristic algorithm to distinguish among objects. But, in case of noise (many small skin-color objects) this solution will not work – Figure 8. Therefore, using color and motion information is simple yet effective solution and tends tracker to be robust against noise.

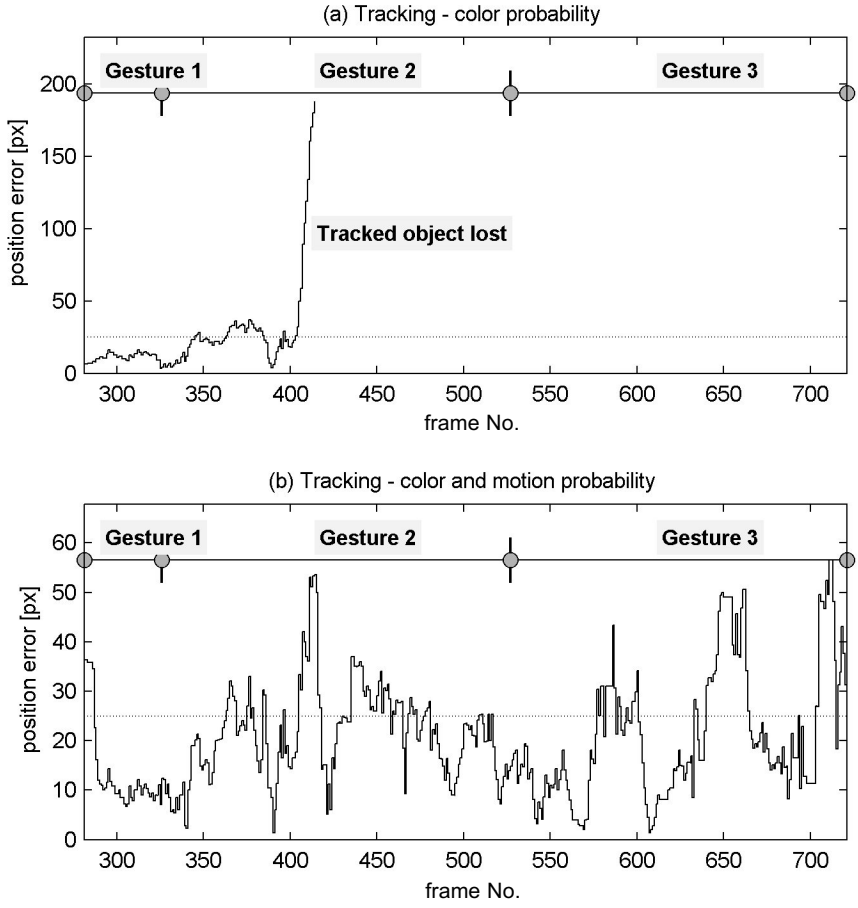

Fig. 7. Tracking position error for video sequence No. 1 – dotted line is the ROI size

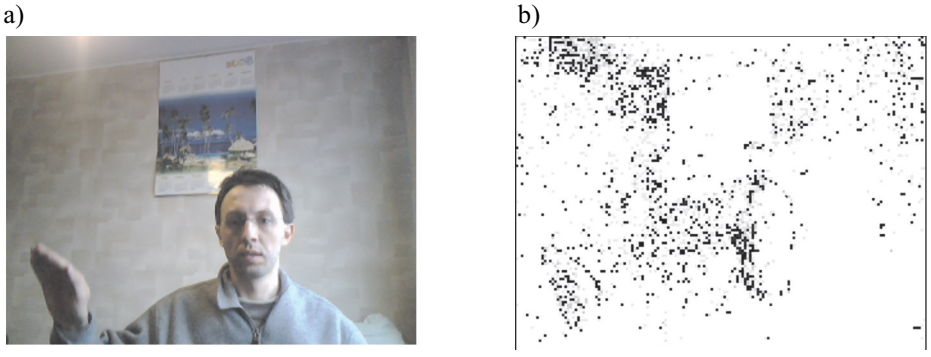

Fig. 8. Example of AR scene with large noise and many small skin-color objects

There is also disadvantage of utilizing color and motion. It results in lower accuracy of tracked object position (Tab. 1). This behavior can be explained by taking into account the following factors: small latency introduced by motion analysis, insufficient frame rate, and sensitivity of the algorithm to small finger movements. The last factor is particularly evident for pointing gesture, where user moves his fingers. It may also be useful to interaction.

Table 1
Tracking results – position error analysis

Video sequence nr	Color probability		Color and motion probability	
	Mean error [px]	Std error [px]	Mean error [px]	Std error [px]
Video No. 1	17	9	21	12
Video No. 2	8	10	24	19
Video No. 3	13	3	17	13

* for video No. 1 and color probability, tracker lost object after gesture No. 1,
for video No. 2 and color probability, tracker lost object during gesture No. 3,
for video No. 3 and color probability, tracker lost object after few frames.

4. Conclusion

Experimental results show that proposed tracking algorithm is efficient and can be used in AR system. Comparing to original CAMSHIFT algorithm, the main benefit from the use of color and motion information is robustness to noise and changes in skin-color distribution (for example – caused by changes in lighting conditions). However, the disadvantage is worse accuracy of tracked object position.

There are several issues that should be addressed in future work. Modeling algorithm in Simulink environment limits testing to offline video analysis. Therefore, integration with C++ framework using Simulink Real-Time Workshop code generation capabilities is planned. This will enable possibility of additional evaluation of algorithm performance with real user interaction scenarios.

Acknowledgement

This work is supported by AGH University Science and Technology, grant nr 10.10.120.783.

References

[1] Andrei S. *et al.*, *Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking*. 1996, 429–438.

[2] Askar S. *et al.*, *Vision-based skin-colour segmentation of moving hands for real-time applications*. 1st European Conference on Visual Media Production, March, 2004.
[3] ArToolkit http://www.hitl.washington.edu/artoolkit/.
[4] Azuma, Ronald T., *A Survey of Augmented Reality*. Presence, vol. 6, 1997, 355–385.
[5] Bradski G.R., *Computer Vision Face Tracking For Use in a Perceptual User Interface.* Intel Technology Journal, (Q2), 1998.
[6] Foxlin E., *Inertial Head-Tracker Sensor Fusion by a Complementary Separate-Bias Kalman Filter.* Proc. of IEEE Virtual Reality Annual International Symposium, 1996, 184–194.
[7] Kato H., Billinghurst M., *Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System*. Proc. of the 2nd IEEE and ACM International Workshop on Augmented Reality, Washington, DC, USA, 1999.
[8] Lars B., *et al.*, *Computer Vision Based Recognition of Hand Gestures for Human-Computer Interaction.* 2002.
[9] Marnik J.M., *Rozpoznawanie znaków Polskiego Alfabetu Palcowego z wykorzystaniem morfologii matematycznej i sieci neuronowych*. PhD thesis, AGH, 2002.
[10] Nishida S., Watanabe J., Kuriki I., Tokimoto T., *Human Visual System Integrates Color Signals along a Motion Trajectory*. Curr Biol. 2007 Feb 20;17(4):366–72. Epub 2007 Feb. 8.
[11] Pragati G., *et al.*, *Vision Based Hand Gesture Recognition*. World Academy of Science, Engineering and Technology, 49, 2009.
[12] Qiu-yu Z., Mo-yi Z., Jian-qiang H., *Hand Gesture Contour Tracking Based on Skin Color Probability and State Estimation Model*. Journal of Multimedia, vol. 4, No. 6, December 2009.