BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXVIII No. 3/2016	62 – 69	Seria Tehnică

Real Time Algorithm for Human Body Tracking with Kinect Device

Cosmina-Mihaela Roșca, Eduard Bogdan Voicilă

Petroleum – Gas University of Ploiesti, Blvd. Bucureşti 39, Ploiesti
e-mail: cosmina.rosca@upg-ploiesti.ro

Abstract

This article presents a new algorithm for real – time human body tracking systems. We will discuss how to acquire a 30 FPS video and how to post-process every frame without affecting the real time acquisition. In this paper we show the basic problems regarding the real-time image processing of 30 frames in only 1 second. This article proposes a new method to solve those issues. The algorithm keeps the CPU processing time at a normal value, does not block the graphical user interface and has a high speed for the eye tracking. The results were evaluated by using C# environment.

Key words: *image processing, input – output systems, Kinect device, real – time, body – tracking, operating systems, multi – threading.*

Introduction

In this paper we present the advantages of using Microsoft Kinect sensor with application in image gallery remote control. The algorithm developed is only a concept proof regarding the remote control application importance based on human gesture. Several studies were made in the same area, starting with Kinect technologies. Paper [1] presents Kinect device hardware structure with the following components: a depth sensor, a RGB camera, an accelerometer, a tilt motor and a multi-array microphone. The depth sensor is composed of the IR laser emitter and the IR camera. As it is presented in paper [2], the depth sensor interprets 3D scene information from a continuously - projected infrared structured light. The IR Camera operates at 30 FPS and provides images with 1200x960 pixels which are down sampled to 640x480 pixels. The RGB camera operates at 30 FPS and provides images with 640x480 pixels. It can also provide images with higher resolution of 1280x1024 pixels at 10 frames per second.

The Kinect device, as it is presented in paper [3], was initially developed as an accessory for the Xbox 360 gaming console, which allows players to control games only with body motion and gestures without the need of additional game controllers. In skeleton movements tracking, as presented in paper [4], the human body is rendered by a number of joints which represent various body parts, each joint being represented by its 3D coordinates. Microsoft Kinect is the newest motion sensor used for the Xbox gaming console. The Xbox 360 has limited computation resources, so the 3D parameters of these joints must be determined in real time, otherwise the gaming performance might be affected. The Kinect device is capable of tracking, simultaneously, up to six people in its view, including two moving players or two full skeletons.

In full skeleton mode, it is able to track twenty joints per active player, as well as ten joints with seated players in Seated mode.

Kinect system has many applications such as human face, 3D environment reconstruction, speech recognition software (used to understand basic commands) and limited gesture recognition.

According to [5], the Kinect specifications are enough to acquire the human body's motion, its rate being 6 to 10Hz (on the average).

As paper [6] presents, eye tracking is a very important field of development and has a wide range of applications in numerous areas. For instance, by using eye detection and tracking, it might facilitate the use of systems for persons with disabilities. Another use for eye detection and tracking is in the vehicle industry, with features like driver monitoring and helping him to stay focused on the road, respectively monitoring the road and helping the driver avoid possible dangers on the road [7].

According to paper [8] another important area of development for devices such as the Kinect is the field of robotics. The depth sensor combined with the RGB camera can generate a complex color 3D map of the surroundings and can even differentiate objects of the same color, therefore contributing to the robot's collision detection algorithm [9, 10].

There are many applications developed in this area. For example, in paper [11] an application was created, allowing the user to control an electro-mechanical hand using hand gestures. This was achieved by using the Kinect device to identify the center of the palm and the user fingerprints.

The Kinect sensor is cheaper than other complex systems with multiple cameras that require special equipment to generate the skeleton for the user in real time, but it is also less accurate. This is why paper [12] proposes the use of the Kinect sensor in fields that do not require such precision, like clinical rehabilitation. It shows that the Kinect can be successfully used in rehabilitation treatments, the authors of the paper having even developed an application of this matter.

Paper [13] proposes the Kinect device as a measurement instrument for people suffering from Parkinson's disease. The conventional systems for measuring movement symptoms are very expensive, demand large spaces and require substantial expertise. This paper shows that the Kinect sensor is an adequate alternative system for measuring the Parkinson's symptoms because the device needs very little space, it is cheaper and does not require qualified personnel to operate.

This paper presents a new method for tracking human gesture. As previously mentioned, the algorithm is only a concept proof using the Microsoft Kinect device by interpreting human gesture and interpreting different coordinates combinations acquired with this device. The algorithm correlate different rules with specific gestures such as crossing hands or hand movement in certain positions. In the next section is described the proposed algorithm using hands identification and their position related to head position and the block diagram of the proposed algorithm. The final section presents the application usage. Finally, concluding remarks are presented.

The Algorithm Description

Real time applications require not only the acquisition part, but also the post – processing actions to be made in parallel computing. Depending on the application type, the number of samples might be reduced, but sometimes it needs all frames, situation when the post – processing should be made in a special software thread.

The proposed technique involves a sampling algorithm with the abilities to acquire only 10 frames per second and to process in the same time key points in the capture. To reduce the computing effort, the additional information regarding the environment is neglected. For this reason, only information about hands is acquired and processed in every frame. At the beginning, the calibration step identifies hands position and follows the differences in the next frames to decide the corresponding actions depending on dedicated application. In this paper a new algorithm implemented for a photo gallery remote control is presented. The application is useful in the situation when the user cannot directly access the device. In this situation, such as driving or cooking, the user might access different images. In the same time, the user can scroll between photos. Another facility of the remote control device is the possibility to select different actions, such as closing the application and go back in the basic state.

Calibration procedure identifies the hands and the head. In this way, the algorithm rules will render the body position. Sometimes, if the body is back to the left, the distance between the head and the left hand is lower than the distance between the head and the right hand. To avoid this kind of situations, the algorithm excludes a limit in the area of head where actions are ignored. Figure 1 highlights the calibration procedure with the right area - right hand, left area - left hand and the transition area marked with black border where no action take effect.

Another important fact is represented by transitions. The basic rule is sample: if leftward movement of the right hand is done, the transition is for the next photo. If rightward movement of the left hand is done, the transition is for the previous photo. Figure 2 depicts the two situations presented, when left hand is in the area of the right hand so the transition to the previous photo will be made (fig. 2a) and right hand is in the area of left hand so the transition to the next photo will be made (fig. 2b).

To avoid multiple transitions for only one movement, a special flag is set and it is reset when the hand is back in the corresponding area.

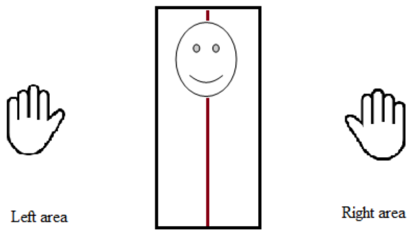


Fig. 1. Calibration procedure.

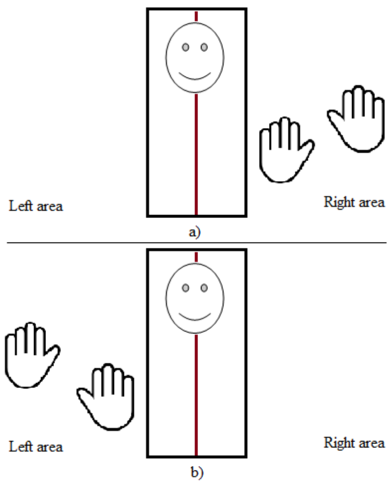


Fig. 2. Transition rules: a – Previous photo; b – Next photo.

The presented application is only a concept proof regarding the possibilities to implement in different areas such as surgery, when interactive decision should be made or scrolling a map when driving, and the user needs to navigate on map and zoom in or out without affecting concentration and hands ability. This kind of applications offer a high level of safety and security for users. The remote control facility avoid the direct involvement of the human operator, but it has the ability to interpret the human body gesture and acts in consequence.

The algorithm analyses three important aspects to provide safety and security remote control access to the Kinect device in order to process the human body gesture:

- Calibration procedure to identify head area, left hand area and right hand area;
- The transition rule for right hand in left area and left hand in right area;
- Flag variable to avoid multiple transitions.

The Proposed Algorithm Block Diagram

The algorithm defines *left_pos* and *right_pos* as the current body position using Skeleton tracking class provided by Microsoft Kinect SDK. This class is used to track in frame only human body position. The application uses HAND_RIGHT and HAND_LEFT variables to track *left_pos* and *right_pos*. Head position is recognized as a balance between hands. For this reason, head should be represented as position zero. The variables *left_flag* and *right_flag* are used to identify when the user's hands are in the opposite area and to prevent multiple photo transitions. At the beginning, these flag variables are set to false. The current photo index in directory is stored using variable *current_photo*.

While the Kinect Sensor is active, the device continuously acquires frames and identifies new positions for every hand.

The transition to a new photo is made only once, when the hand is in the opposite area. The next transition is possible when the hand goes back in the corresponding area and then returns in the opposite area. As it can be seen in figure 3, if the left hand is in the right area (*left_pos > 0*) and the hand is for the first time in this session identified in the right area (*left_flag* == false) then the algorithms sets the flag *left_flag* to true in order to prevent another transition in this session, then tests if the current photo is not the first one. If the first photo is already displayed in the directory, no other transition to the previous photo will be possible. Otherwise, the previous photo in directory will be displayed.

A session is interpreted as the time in which the hand was found in the opposite area, because the hand could be moved left and right multiple times without going back in the corresponding area. Using flag variable it will prevent multiple transitions, but also locking situations when no transition will be possible.

If the left hand returns in the corresponding area – left area (*left_pos < 0*) and the *left_flag* is set to true, then the algorithm will reset the flag variable to allow another photo transition.

Keeping the same algorithm, if the right hand is identified in the opposite area for the first time in the current session and the *right_flag* is set to *false*, then a new validation regarding the next photo availability is required. If the current photo is not the last photo in the directory, then the algorithm will display the next photo and *right_flag* will be set to *true*.

When the right hand returns to the corresponding area, the flag variable *right_flag* will be set to false for the next transitions.

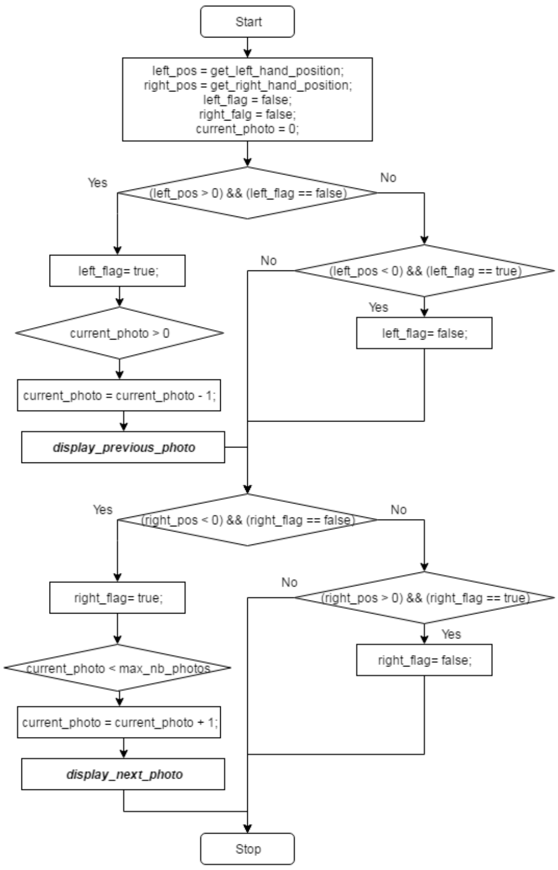


Fig. 3. Block diagram.

Application Example

Remote control applications are the best solutions for situations when the user cannot access the device, situations such as: driving, cooking, medicine, etc. Remote control applications are the best solutions for situations when the user cannot access the device, situations such as: driving, cooking, medicine, etc. ensuring the security of the user.

This application provides an efficient security system solution capable to interpret the hand movements without touching the device. The algorithm was implemented using C# based on Microsoft Kinect SDK. The application explores the selected directory and displays the corresponding picture. The user runs to the next picture even if it is located at 2 meters distance.

Application facilities:

- The user starts and stops the application using one specific gesture, by simulating pressing a button with the hand. It doesn’t matter if the user presses the button using the right or the left hand.
- Closing the application is implemented by the developed algorithm as crossed hands.

In this example, we present the application operating way. In the calibration procedure, the user must be on the same axis with the center of the Kinect device, so the head position will be identified as zero axis and every frame identifies the hands at the same distance from head. Figure 4 displays the main interface where the push button function capable of activating the photo gallery interface can be seen.

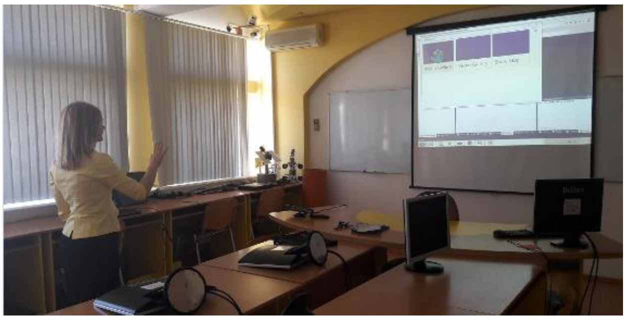
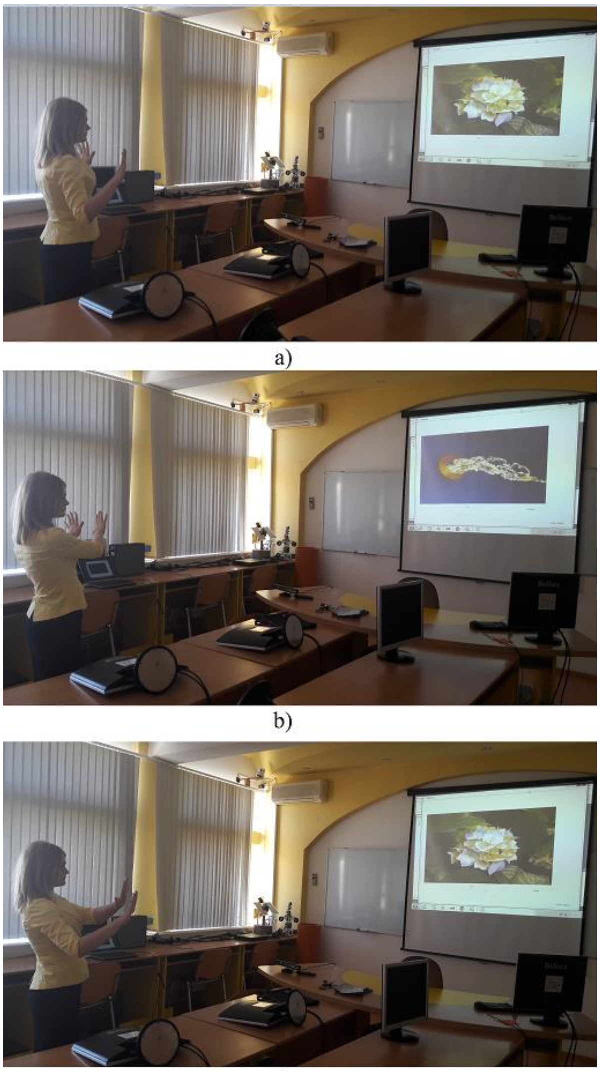


Fig. 4. Push button function using Kinect device.

Figure 5a displays the photo gallery interface and the first photo. In this figure it can be seen the hands position related to head and Kinect device (placed on desk). Figure 5b depicts the situation when the right hand is placed in the left area and the transition is made to the next photo. Figure 5c illustrates the transition to the previous photo, when the left hand is placed in the right area. In figure 5c the previous photo is the same with figure 5a.



a)

b)

c)

Fig. 5. a – The calibration procedure; b – Transition to the next photo;
c – Transition to the previous photo.

In figure 6 the crossing hands rule is presented that closes the photo gallery module and goes back to the main interface.

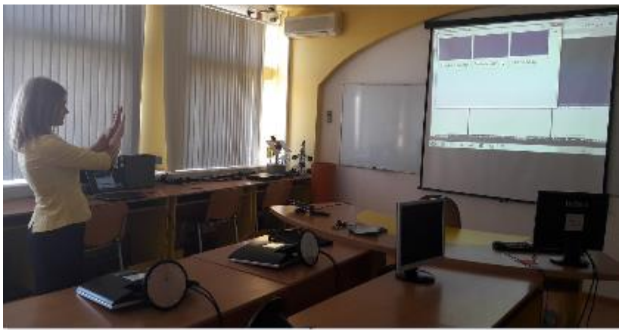


Fig. 6. Closing photo gallery module by crossing hands.

Conclusions

In this paper, a new algorithm for real time remote control application based on human gesture is presented. The algorithm is dependent on the left and right hands position related to head position. The algorithm interprets when the hand is in the opposite area using the rules associated with the coordinates acquired by the Kinect device. The presented algorithm solves the real – time acquisition and post – processing issues using sampling algorithm for signal acquisition.

References

1. Cruz, L., Lucio, D., Velho, L. – Kinect and RGBD Images: Challenges and Applications, *Proceedings of the 25th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, 2012, pp. 36–49.
2. Frati, V., Prattichizzo, D. – Using Kinect for hand tracking and rendering in wearable haptics, *IEEE World Haptics Conference (WHC)*, 2011, pp. 317–321.
3. Dikovski, B., Madjarov, G., Gjorgjevikj, D. – Evaluation of different feature sets for gait recognition using skeletal data from Kinect, *37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014, pp. 1304–1308.
4. Zhang, Z. – Microsoft Kinect Sensor and Its Effect, *IEEE MultiMedia*, Vol. 19, No. 2, Apr. 2012, pp. 4–10.
5. Chen, Y.C., Lee, H.L., Lin, K.H. – Measurement of body joint angles for physical therapy based on mean shift tracking using two low cost Kinect images, *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 703–706.
6. Al-Rahayfeh, A., Faezipour, M., Enhanced frame rate for real-time eye tracking using circular hough transform, *Systems, Applications and Technology Conference (LISAT)*, 2013 IEEE Long Island, 2013, pp. 1–6.
7. Cangea, O., Moise, A., Bucur, G., Popescu, C. – Intelligent Transportation System. Case study, *15th International Multidisciplinary Scientific Geoconference (SGEM)*, Albena, Bulgaria, June 18-24, 2015, pp.167-174.
8. Tolgyessy, M., Hubinsky, P. – *The Kinect Sensor in Robotics Education Institute of Control and Industrial Informatics*, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava.
9. Popescu, C., Paraschiv, N., Cangea, O. – Neuro-Fuzzy Controller for Mobile Robot Navigation with Avoiding Obstacles and Reaching Target Behaviors, *Annals of DAAAM for 2009 & Proceedings of the 20th International DAAAM Symposium*, Editor: Branko Katalinic, Published by DAAAM International, Vienna, Austria, 2009, pp. 298-299.
10. Moise, A., Cangea, O., Popa C. – Software Package Development for Mobile Robot

Control and Behavioral Training, *14th International Multidisciplinary Scientific Geoconference (SGEM)*, Albena, Bulgaria, June 17-26, 2014, pp. 331-338.

11. Raheja, J.L., Chaudhary, A., Singal, K. – Tracking of Fingertips and Centre of Palm using KINECT, *Proceedings of the 3rd IEEE International Conference on Computational Intelligence, Modelling and Simulation*, Malaysia, 20-22 Sept., 2011, pp. 248-252.
12. Fernandez-Baena, A., Susín, A., Lligadas, X. – Biomechanical Validation of Upper-Body and Lower-Body Joint Movements of Kinect Motion Capture Data for Rehabilitation Treatments, *4th International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, 2012, pp. 656–661.
13. Galna, B., Barry, G., Jackson, D., Mhiripiri, D., Olivier, P., Rochester, L. – Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease, *Gait & Posture*, Vol. 39, No. 4, Apr. 2014, pp. 1062–1068.

Algoritm de identificare în timp real a gesturilor umane folosind un dispozitiv Kinect

Rezumat

Acest articol prezintă un algoritm destinat identificării gesturilor umane în timp real folosind un dispozitiv Microsoft Kinect. Acest dispozitiv achiziționează 30 de frame-uri pe secundă, însă procesarea fiecărui frame necesită un minim de 100 milisecunde, ceea ce conduce la introducerea unui timp mort între achiziția frame-ului și luarea unei decizii pe baza identificării gestului uman. Din aceste considerente, prezentul articol propune un nou algoritm care va achiziționa doar coordonatele asociate mâinilor, iar în funcție de anumite reguli impuse se vor lua anumite decizii precum tranziția la poza următoare (atunci când mâna dreaptă se află în stânga capului), tranziția la poza anterioară (atunci când mâna stângă se află în dreapta capului) și închiderea aplicației atunci când cele două mâini sunt încrucișate.