

Position Tracking in 3D Space Based on a Data of a Single Camera

Iakushkin Oleg(✉), Ruslan Sevostyanov, Alexander Degtyarev, P. E. Karpiy, E. G. Kuzevanova, A. A. Kitaeva, and S. A. Sergiev

Saint-Petesburg State University, Saint Petersburg, Russia
o.yakushkin@spbu.ru, st054717@student.spbu.ru

Abstract. A high cost of equipment that solves the problem of tracking the position and direction of users in real time is one of factors that negatively affect the speed of development of the augmented reality industry. The urgency of this problem is a premise for the development of a financially available tracking system. In this research, we propose a software and hardware architecture of a system that solves three-dimensional tracking problems in a closed space and postures classification using neural network models. Distinctive feature of our system is the feasibility in borders of strictly limited computing power and the absence of any sensors placed on monitored objects. After setting the boundaries of the active area, all the necessary input data is provided by a static camera without an infrared filter. As an example of the implementation of a resource-limited solution, we present the assembly of this solution on a Raspberry Pi version 3 single board computer equipped with the Intel Neural Stick version 2 co-processor and a Raspberry version 2 NoIR camera. The first section of the article describes technical characteristics of the equipment used in the study. The second part is dedicated to the solution algorithm and its brief description. Further, in the third stage, the ways of data collection, necessary for a correct assessment of position, direction and posture are illustrated. The fourth, final section presents the results, discussion and possible directions for further work.

Keywords: Software · Optical flow · Augmented reality · Internet of things

1 Introduction

Owing to the growth of computing components capabilities of the Internet of Things, artificial intelligence allows us to solve more non-strictly formalized tasks in the field of computer vision: for example, face recognition [1] and image classification [2]. Such solutions are used in various fields: medicine [3], ecology [4], industry [5], entertainment [6].

Supported by Russian Foundation for Basic Research, grant number 17-29-04288.

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11622, pp. 772–781, 2019.
https://doi.org/10.1007/978-3-030-24305-0_58

The task of tracking is a special interest of [7]. The program proposed by the authors processes video frames and provides the position of objects regarding the surface on which the objects are located. Determining the position of an object possessing several degrees of freedom is called positional tracking. It is widely used in the field of virtual reality [8].

For example, tracking a person's position in space can be used in video games for positioning characters in augmented reality. However, most of the solutions proposed in literature are proprietary, technically complex, expensive, and use large computational powers.

The solution described in the work [9] requires the use of sensors fixed directly on each user's head and legs, which causes additional inconvenience, as well as the direct dependence of the cost of the final set of equipment on the number of players involved in the gameplay.

The solution [10], built on the basis of a convolutional network, is an assessment of posture using several cameras, which implies the use of equipment whose computing power will be sufficient for parallel processing of images from two cameras using a neural network. The essence of the method is that an RGB network processes images taken from the cameras as input data, and the output of the relative rotation and displacement as output data.

Another solution [11] is trained to assess of the current pose based on the previous postures. These scores are then appropriately combined with the output of a visual tracking system using a linear Kalman filter to ensure reliable final posture assessment. Application of such a solution for our purposes is hindered by the cost of devices capable to process images from the camera by the neural network and the subsequent application of the Kalman filter.

The high cost of equipment that solves the problem of positional tracking is one of the factors that negatively affect the speed of development of the virtual reality industry [12]. An obvious solution to this problem is to develop software that can track coordinates in real time, calculate directions, and classify users'postures on low-power systems.

The proposed task can be characterized from different sides and highlight additional aspects for further development. One of them is the problem of restoring the scene [13]. Described in the paper approach is based on the extraction of objects from the database of three-dimensional figures so that the coordinates of the object to be modeled can be determined. Owing to this approach, it is possible to replace real two-dimensional shapes with objects found in the base of three-dimensional shapes. It is necessary at the final stage when visualization occurs so that the administrator has an idea of what is happening with the player and what environment surrounds them. In the most elementary case, a set of points of three-dimensional space can be used as input data.

Also in the section of actual questions for this article, we consider object recognition [14] as the process of teaching the machine to answer the question whether the image or video contains some specific detail, a special feature. The input data for the system is the flow of images from one constant position as well as user records: their names, color of clothes, height and width of shoulders.

Tracking results are transmitted to a remote device for visualization. We have chosen the Outside-in approach, which implies the presence of a fixed observer, in this case a single camera. Similar results can be achieved by restoring original objects [15]; however, modeling using a 3D laser scanner is a very laborious process, often accompanied by an original object shape.

The characteristic feature of our solution is that it does not require large computing power which is why it can be deployed on cheap equipment.

1.1 Motivation and Value

There are many different applications of computer vision and its implementations. They, as a rule, require high performance from the equipment used, specific knowledge from the user in the field of this equipment and its applicability, and, as a result, large financial costs.

Taking it into account, it becomes clear that there is a need for development of a prototype that solves the problem of positional tracking, which would be relatively cheap, easy to install, maintain/replace components, easy to use and available for everyone to buy (Fig. 1).

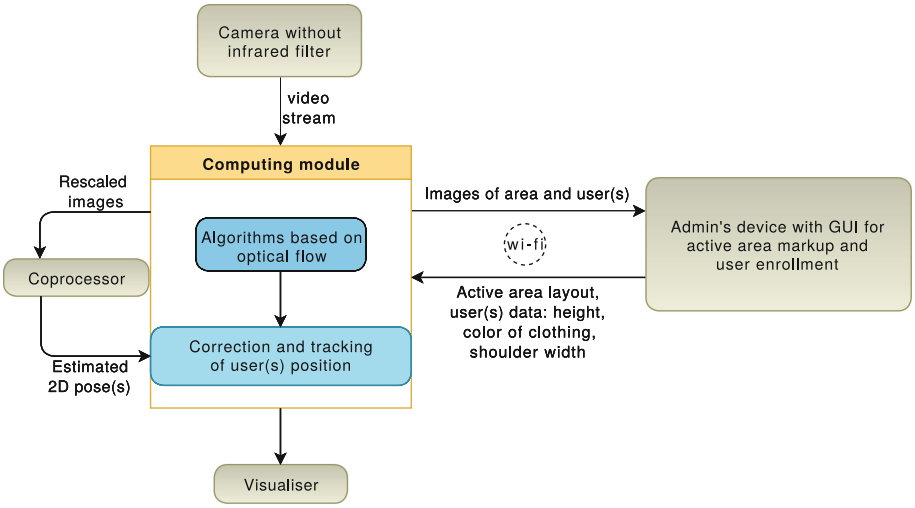

Fig. 1. Description of the interaction between the hardware

2 Existing Software and Algorithms

It was decided to use the OpenPose neural network to initially determine the player's posture from the camera's image. OpenPose is a convolutional neural network that jointly determines the position of the head, torso, arms, legs using

key points and displays it in a picture or video in real time. 18 key points are assigned to the spine and legs, 21 for each arm, 70 for the head [17]. To work correctly, OpenPose requires Python3, tensorflow 1.4.1+, protobuf, python3-tk. After the initial position of people is determined by the OpenPose network, players will be tracked using the open source OpenCV library containing computer vision processing algorithms.

The task of analyzing the position of objects and the direction of their movement in real time in one form or another is reduced to analyzing the position of these objects on separate successive frames. In the minimal scheme of our project, we plan to analyze the video signal on the Raspberry Pi 3 Model B module, which is directly connected to the camera. The data from the module is transmitted in separate frames to the administrator's device, where the user can set the camera's view area for analysis, which then returns to the Raspberry module. The video signal, along with data on the size and position of the analysis area, is transmitted to the neural network enhanced by the Intel Compute Stick, which returns the approximate coordinates of the objects. The processed data is then transmitted to the main output device, which may be the same or different from the administrative device, and become available to the user.

It is worth noting that limiting the computational capabilities of Raspberry may require subsequent modification of the scheme of our project. Moreover, the scaling of the proposed solution for a wide range of users is noticeably limited by the need to use additional hardware modules. For this reason, we will consider the possibility of transmitting the primary video signal to the user's device with the possibility of analyzing the video signal on the same device or the subsequent sending of the compressed image to the RP module. Note that the modification of the minimum scheme of the project with the exception of the RP module will separate the camera and the analyzing device, and exclude all external modules except for the motion sensor itself. Thus, the implementation of the project in the future can be reduced to a software component.

In our task, the relevance of the data to a high degree depends not only on the speed of the program analysis, but also on the quality of information transmission from the sensor to the analyzing element to the user's device. In particular, the use of the body position analysis system, which is the ultimate goal of our work, for a wide range of users inevitably affects the interaction task, which in turn imposes additional requirements on the choice of information storage and transmission format.

In the minimal scheme of the project, the main question is the choice of the format of transmitting symbolic information obtained from the analysis of the video signal by a neural network. Among the existing alternatives for the transmission and storage of structured data, we can mention such options as JSON, MessagePack, ProtocolBuffers. Nowadays the generally accepted standard for transferring information from application to application is JSON. Thus, the solution of the problem of compatibility with third-party applications of visualization developers is facilitated by the choice of this format. MessagePack, in its turn,

is fully compatible with JSON, but allows to win in data storage efficiency. By saving traffic, it is possible to transfer more data for the same period of time.

Note that the use of the ProtocolBuffers binary format can somewhat speed up the data formatting process, which is an important limitation with the continuous transfer of processed information. However, the processed data on the positions of the bodies and the directions of their movement can be compactly presented in textual form, and the expected amounts of memory required for storing and transmitting this information are small. Thus, compatibility and simplicity of debugging turn out to be more important criteria for choosing the format of data transfer. Summing up this observation with the reasoning above, we conclude that the optimal solution in our project is MessagePack. Speaking about the possibility of direct transmission of the video signal to the administrator's device with subsequent analysis that would not involve additional external systems, we need to analyze the possible options for data transmission protocols.

Currently, the most common and widely supported real-time video transmission and processing protocols are HLS and RTMP. The HLS protocol is built on splitting the video signal up to a sequence of small HTTP-based download files. In its turn, the RTMP protocol is based on the TCP and allows to maintain a continuous connection with low latency in video transmission. Recently, HLS has continuously been strengthening its position in the market and providing more than half of the views of all videos from the analysis [18] (Fig. 2), leaving a quarter of all views on RTMP. Thus, these two protocols are the most natural possible candidates for implementing our system for recognizing the position and direction of motion of objects in a video signal.

Usage of the most common HLS protocol has several undoubted advantages: higher degree of compatibility, a high degree of relevance, active support and updating of protocol versions, and so on. However, the main reason for the popularity of HLS is its compatibility with standard web services and content distribution services, which allows the developers to scale the broadcast simultaneously to multiple users. This protocol is also more convenient for working with modern mobile devices and provides the ability to adapt the signal to the state of the user's connection. However, the considered problem of analyzing the position of objects and the direction of their movement in real-time video is more sensitive to the delay in signal transmission and does not imply a continuous wide broadcast of the video signal itself to many users. Thus, the very structure of the HLS protocol, based on the transfer of small but finite files containing segments of the transmitted signal, makes it less suitable for our project.

3 Preparation

The image of the room and in particular the plane on which the users will be located is transmitted from the camera. Through RTMP, an image is transmitted to a remote device, on which the administrator will designate the boundaries of the plane by setting its four corners on the received image and entering the metric parameters of the space where users are to be searched for.

Specifying the characteristics of the space (in our case, the length and width of the plane) is necessary in order to determine the position of the camera relative to the plane. Using OpenCV library, the allocated space is projected as a quadrilateral onto the camera plane, transforming the perspective. Thus, there is a one-to-one correspondence between the points of the initial image of the space and the rectangular coordinate system specially introduced for this space. Perspective transformation occurs using a square matrix M of dimension 3.

Assuming that each participant has a T-shirt with sufficiently distinguishable colors from each other to uniquely identify each user and assign them a personal number, which is used later to distinguish the participants. The tracking process is preceded by a preparatory stage in which the shoulders of any of the participants are parallel to the camera plane. The procedure is necessary in order to assign each participant a unique ID. In order to do this, the positions of the 2 edge points on the shoulders, as well as the point between them, are estimated. Next, the average color of these points is calculated and assigned to the player that is currently being identified. After that, the administrator enters the player's shoulder width.

4 Recognition Stage

Raspberry Pi NoIR Camera transfers the image to the Raspberry Pi. Next, the Raspberry Pi through the Intel Neural Compute Stick, which is necessary for deploying a neural network on low-power equipment and speeding up calculations, returns the processed frame with defined and marked human limbs. Initially, the neural network highlights the points of the user's nose, neck, shoulders, elbows, wrists, hips, knees, ankles, feet, ears, and eyes. To solve the problem, it was necessary to choose a method for determining the pose of the object in the image. This can be done in one of several ways, depending on the choice of methodology.

The first, analytical, method requires to know the points on the object and the points on the plane as soon as a set of such points has been identified. This allows to restore the pose with the help of equations linking the known points. But all the required points for solving equations are not always known, or the systems are too cumbersome to solve.

Another method for solving our problem is to use neural networks, the weights of which can also be obtained in various ways. We settled on choosing OpenPose - a pre-trained neural network. When choosing a solution method, energy efficiency and low productivity of the proposed computing equipment were taken into account.

From the obtained marked position of a person, one can determine their position in space as follows: the position of two human feet is averaged, and using the position of the obtained point on the image and the parameters obtained earlier for the perspective transformation (data on the size of the area where the person is located and the transformation matrix M) coordinates of the point are determined, which is the average position of a person in a rectangular coordinate system associated with the area.

To determine the angle of rotation of the body, we use information about the width of the shoulders of this person, the positions of their legs and the midpoint between the shoulders. The average position of the user on the floor is the average of the coordinates of the feet. Then, using the calculated midpoints, we find the projection of the shoulder points onto the floor plane.

Calculating the distance between the points of the projections, we estimate the width of the shoulders. By comparing this result with the ground-truth width of the shoulders, it is possible to estimate the angle of the shoulders relative to the floor plane and correct the coordinates of the projections.

Further, having built a perpendicular to the line connecting the projections of the shoulder points, we obtain a line along which the intended view is directed. True direction can be obtained from information about the direction of the head relative to the camera plane, which can be determined using the OpenPose library: in the case of head pointing towards the camera, the library has the ability to determine the coordinates of the person's nose in the image, if the head is directed back to the camera there is no such possibility. There are two ways to determine a person's posture. The first approach is to estimate the angle between the thighs and the shin, indicating the smallest bend angle in the knees, at which the player is considered to be in a standing position. This method has two main problems that we encountered in this work: first, the available angle is calculated with a certain error and at positions close to the boundary ones, it can give distorted information about the player's actual pose; secondly, if the player is positioned in such a way that when they bend their knees, they move in a plane perpendicular to the plane of the camera matrix, the camera will not be able to distinguish the player's position in the sense of squat according to the specified evaluation criterion. The second approach is to estimate the distance from the center point between the shoulders to the floor, which can be calculated by knowing the position of the player (midpoint of the legs) and the distance from the camera to the player. In the second approach, a posture is considered to be in a sitting position, in which the shoulders of a person are at a smaller distance from the floor than a value equal to the product of a person's height by a factor chosen in advance.

5 Results

A successful connection of all technical devices with the highest possible performance was made. There were problems with installing software on the Raspberry Pi for the correct operation of the Intel Neural Compute Stick due to the fact that the native operating system of the board does not appear in the list of compatible operating systems.

The neural network was optimized so that it gave only the data necessary for the project, and not the full set of key points. Original ways of determining a person's pose from the available data were proposed. The frame frequency of the network, together with Optical Flow, has been improved compared with the operation of a neural network without optical flow at least 1.7 times.

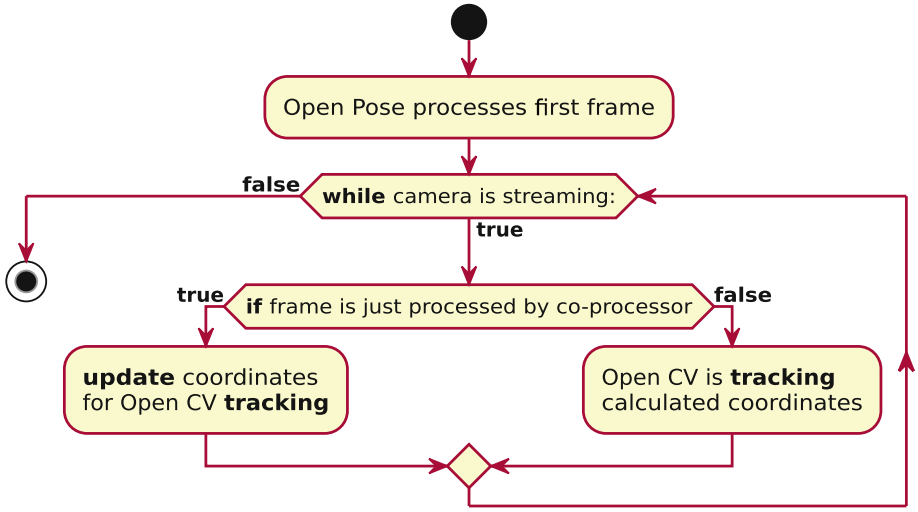

Fig. 2. Schematic description of the process of increasing the tracking frequency

6 Prospects

It is possible to increase the number of coprocessors that accelerate the network performance on the Raspberry Pi. With an increase in computing power, frames will be processed faster, which means that the information provided to the final device will have greater accuracy.

There is a need to increase the number of sensors in order to obtain more information about the object, that is, to more accurately determine the position of a person in space and the direction of the player. It makes sense to convert points from a 2D format to a 3D format [19], to help the system understand how a point appears from different points of view. Since all technical elements are easily replaceable, it is possible to try out several different combinations and determine the optimal configuration. Certainly, it is necessary to try to improve the quality of the image by technical means, as an option, using the super-resolution method [20].

In the long term there is the possibility of developing an application on the target device, more convenient for the side developer, who will be able to use the results obtained by working for their own purposes, for example, to calculate the trajectory of the character in a VR-game.

Acknowledgments. This research was partially supported by the Russian Foundation for Basic Research grants (projects no. 17-29-04288). The authors would like to acknowledge the Reviewers for the valuable recommendations that helped in the improvement of this paper.

References

1. Parkhi, O.M., Vedaldi, A., Zisserman, A.: Deep face recognition. In: British Machine Vision Conference Swansea, UK (2015)
2. Chan, T., Jia, K., Gao, S., Lu, J., Zeng, Z., Ma, Y.: A simple deep learning baseline for image classification? IEEE Trans. Image Process. **24**(12), 5017–5032 (2015)
3. Seo, J., Han, S., Lee, S., Kim, H.: Computer vision techniques for construction safety and health monitoring. Adv. Eng. Inform. **29**(2), 239–251 (2015). ISSN 1474–0346
4. Seiferling, I., Naik, N., Ratti, C., Proulx, R.: Green streets - quantifying and mapping urban trees with street-level imagery and computer vision. Landscape Urban Plann. **165**, 93–101 (2017). ISSN 0169–2046
5. Qiu, W., Yuille, A.: UnrealCV: connecting computer vision to unreal engine. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 909–916. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_75
6. Shafaei, A., Little, J.J., Schmidt, M.: Play and Learn: Using Video Games to Train Computer Vision Models. CoRR, vol. abs/1608.01745 (2016)
7. Kermadi, M., Berkouk, E.M.: Artificial intelligence-based maximum power point tracking controllers for photovoltaic systems: comparative study. Renew. Sustain. Energy Rev. **69**, 369–386 (2017). ISSN 1364–0321
8. Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., Nießner, M.: FaceVR: Real-Time Facial Reenactment and Eye Gaze Control in Virtual Reality. CoRR, vol.abs/1610.03151
9. Zank, M., Nescher, T., Kunz, A.: Tracking human locomotion by relative positional feet tracking. In: IEEE Virtual Reality, pp. 317–318, Arles (2015). https://doi.org/10.1109/VR.2015.7223423
10. Melekhov, I., Ylioinas, J., Kannala, J., Rahtu, E.: Relative camera pose estimation using convolutional neural networks. In: Blanc-Talon, J., Penne, R., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2017. LNCS, vol. 10617, pp. 675–687. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70353-4_57
11. Rambach, J.R., Tewari, A., Pagani, A., Stricker, D.: Learning to fuse: a deep learning approach to visual-inertial camera pose estimation. In: 2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 71–76 (2016)
12. Avila, L., Bailey, M.: Virtual reality for the masses. IEEE Comput. Graph. Appl. **34**(05), 103–104 (2014). https://doi.org/10.1109/MCG.2014.103
13. Li, Y., Dai, A., Guibas, L., Nießner, M.: Database-assisted object retrieval for real-time 3D reconstruction. Comput. Graph. Forum **34**(2), 435–446 (2015). https://doi.org/10.1111/cgf.12573
14. Sawyer, R., Smith, A., Rowe, J., Azevedo, R., Lester, J.: Models in game-based learning with facial expression recognition. In: Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, pp. 192–201, New York, NY, USA (2017)
15. Ruchti, P., Steder, B., Ruhnke, M, Burgard, W.: Localization on OpenStreetMap data using a 3D laser scanner. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 5260–5265, Seattle, WA (2015). https://doi.org/10.1109/ICRA.2015.7139932
16. Sajjad, M., et al.: Raspberry Pi assisted face recognition framework for enhanced law-enforcement services in smart cities. Future Gener. Comput. Syst. (2017)
17. Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., Sheikh, Y.: OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. CoRR, vol. abs/1812.08008 (2018)

18. State of streaming protocols (2018). https://blog.wmspanel.com/2018/12/state-of-streaming-protocols-2018-summary.html. Accessed 24 Dec 2018
19. Pavllo, D., Feichtenhofer, C., Grangier, D., Auli, M.: 3D human pose estimation in video with temporal convolutions and semi-supervised training. CoRR, vol. abs/1811.11742 (2018)
20. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1874–1883 (2016)