Hand Gesture Recognition with Multiple Leap Motion Devices

Vasilij Kiselev, Maxim Khlamov, Kirill Chuvilin
Moscow Institute of Physics and Technology (State University)
Moscow, Russia
kiselev.va, khlamov, kirill.chuvilin@phystech.edu

***Abstract*—Gesture recognition is often used in various fields like sign language recognition, entertainment, virtual and augmented reality applications and many others. Vision based approaches with such devices like Leap Motion controllers and Kinect are often used in such systems and allow to achieve high performance in terms of recognition accuracy. This paper shows how one can overcome the most common problem of such systems — self-occlusion. In this paper we propose the method for the hand gesture recognition which requires 3 Leap Motion controller devices. This work shows that increasing the number of cameras can leads to better recognition accuracy.. In addition we provide the methods of creating camera setup.**

I. Introduction

Human-computer interaction becomes more and more intuitive with the use of such technologies as voice, text, gesture recognition, speech generation, touch devices and so on. In this article we focus on the gesture recognition. The most common application for the gesture recognition systems are sign language recognition application and different types of entertaining systems: gaming consoles, virtual and augmented reality applications and other. As the study [1] shows there are several significant challenges related to technical and usability aspect, which prevent rapid spread of this technology into other possible areas such as health environments, assistive technologies, crisis management and human-robot interaction. High recognition accuracy, intuitiveness, usability and low response time are among these challenges. The research in the gesture recognition field includes multiple works. Two basic approaches are vision and non-vision.

Fig. 1. Leap Motion Sensor Inside

Non-vision approaches require wearable devices like bending gloves to track fingers and hand. Gloves consist of the tactical or sensory units, that are attached to the fingers or joints of the glove, with the wired interface [2]. The first systems of such type appeared in the late 1970s. The advantages include no requirement for pre-processing thus allowing to use these systems with computers with limited processing power. The disadvantages prevent this technology from spreading widely. Glove-based system are still used mostly by researchers, in virtual reality applications, realtime motion capture and so on due to it's high price. Also it's not very comfortable to wear such a device because of the wires.

Vision approaches are mostly based on the use of depth cameras. These systems have several advantages over the non-vision approach. The whole recognition process is non-intrusive and doesn't require user to wear any additional devices. The hardware used in such system is widespread and is available at low cost. As long as our work is based on the data from Leap Motion devices, we focus on vision methods.

A. Related Work

The work [3] proposes a hybrid method based on gradient and stochastic optimization methods to detect fingers and hand. The system presented by Qian et al. works in real time at 25 FPS. They made their own dataset from a 400-frame videos with the Intels Creative Interactive Gesture Camera.

Some works use Leap Motion and Kinect controllers and combination of them [4], [5], [8], [9], [10]. The Leap Motion controller device is smaller, has smaller working area, but higher resolution compared to Kinect. Such systems use Kinect as a body-tracking devices and precise it's data with the data acquired from the Leap Motion controller.

The work [4] proposes a hand gesture dataset containing 10 gestures and obtained with a single Leap Motion device. People who were involved in making the dataset were told to perform gestures within Leap Motion controller valid visual range. The authors improve the recognition accuracy with the use of 2 new features: Fingertips Tip distance and HOG feature of Leap Motion controller sensor images.

The work [5] proposes a hierarchical regression model to analyze single depth images. The method is verified on two real-world datasets: ICLV [6] and MSRA [7]. The main result of this work is good real-time performance without GPU's.

The work [8] considers the use of the Leap Motion Controller for Indian Sign Language gesture recognition. The authors propose a dataset of 45 gestures and achieve maximum recognition accuracy of 90%.

ISSN 2305-7254

The work [9] propose a hand gesture recognition system based on combination of the Leap Motion Controller and Kinect data. The hand gesture recognition scheme here is clearly focused on Leap Motion data. An ad-hoc feature set based on the positions and orientation of the fingertips is computed and fed into a multi-class SVM classifier in order to recognize the performed gestures. The authors prepared the dataset of 10 different gestures on the subset of the American Manual Alphabet. Authors achieved 91.28% accuracy by combining data from the two sensors.

The work [10] proposes the Arabic sign language recognition system based on two Leap motion controller devices. The authors used LDA classifier and achieved 97.7% accuracy. The most misclassified signs were classified to the similar ones. Their method includes data acquisition, preprocessing stage, feature extraction and classification. Unfortunately there are not many details about the cameras setup in this work. It was only mentioned that they were placed at the right angle.

The high (around 90%) accuracy achieved in the most works mentioned before can be explained by the good adjustment of machine learning features and the fact, that gestures were facing the camera when recording. One can rarely see self-occluded gestures in these datasets. We want to specifically focus on this problem in our work so we expect the recognition accuracy from a single device to be lower compared to the results of other authors.

B. Relevance

Human-computer interaction in virtual reality applications can be significantly improved with the use of the gesture recognition technology. Moreover, virtual reality devices are used in many areas lately. The impressive results of using virtual reality systems have been achieved in such fields like surgery [11], prosthetics [12], transplantology [13]. A method proposed in [14] accelerates the training rate in the management of upper limb prostheses. The paper [15] proposes a system for observing data obtained by digital holographic microscopy in virtual reality environment. The leap motion device is used to manipulate 3d models of micro-objects in this work. The works [16], [17] examine the role of the visual perception of various parameters according to the results of testing skiers on a virtual environment simulator.

The task of providing a natural and convenient interface for managing virtual objects is important for many systems from this list. In order to solve this problem the object in the virtual scene needs to be positioned precisely. This problem can be solved by tracking and recognition systems. Recently, the market for optical systems has grown significantly, and the number of implementations of related algorithms for gesture recognition and tracking the position of various items has increased. Cameras and depth sensors are of particular interest, since they are highly accurate and do not affect the degree of perception of the virtual environment due to the lack of physical contact. The development of the human-machine interface with the possibility of carrying out experimental measurements of the degree of perception of the virtual reality is crucial.

In the work [18] a classification and a brief overview of the tracking systems and the principles of their work are given. For the purposes of tracking fine motor skills, Leap Motion is one of the best choices. In addition, the Leap Motion SDK allows to receive raw data directly from the camera (raw data), which allows to develop custom methods and algorithms if necessary. We have previously worked with the Leap motion device. In the [21] we used the gesture recognition system based on a Leap Motion device to make navigation in the augmented reality application and the automatic operator actions graduation. And in [20] we used several devices to construct models in virtual reality.

C. Leap Motion device

The Leap Motion sensor includes several infrared LEDs that illuminate working area with the patternless IR light, and two cameras for image acquisition (Fig. 1 1). Leap Motion Controller device provides an API that allows to get the characteristics of the position of the fingers, palms and wrists of the user's hands and recognize some gestures. The Leap Motion can recognize 4 types of gestures by default [19]. All of them are dynamic.

- Circle. A single finger tracing a circle.
- Swipe. A long linear movement of a single finger.
- Key Tap. A tapping movement by a single finger on an imaginary horizontal surface.
- Screen Tap. A tapping movement by a single finger on an imaginary vertical surface.

D. Contribution

In this paper we want to contribute the following.

- Two hand gesture datasets: one with 5419 samples of 3 gestures and the other with 1278 samples of 7 gestures. The gestures were performed without the respect to the visual range of a single device and thus are close to the real situations.
- Several machine learning algorithms comparison and a machine learning model that will use data from multiple cameras.
- High recognition accuracy with the proposed model. The solution for the problem of self-occlusion.

E. Work structure

The paper is organized in the following way: Section 2 describes the general architecture of the recognition system. Section 3 describes the Leap Motion device and data we get from it. Section 4 describes the hand gesture dataset. Section 5 describes the setup of the Leap Motion cameras. Section 6 describes the machine learning model. Section 7 has the experimental results. Section 8 has the conclusions and future work.

II. Architecture of the Recognition System

The key element of the recognition system is the Leap motion controller device [22]. Our system consists of 3 such devices. The Fig. 1 2 shows the recognition model we designed.

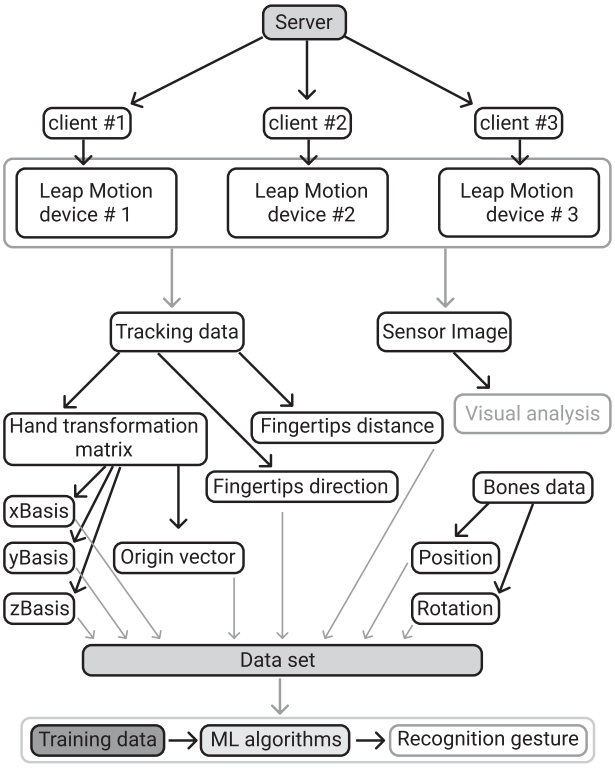

Fig. 2. System architecture

Each of the 3 devices is processed by an independent client program. These programmes are united by a local network. This program updates the data in memory (tracking data and sensor image) for each new frame arriving from Leap Motion. The server has the ability to give the command to the client to save the data. When such a command is received, the client saves the latest updated data from the memory. Each iteration of the saved data from 3 devices will be assigned a unique identifier. The dataset consists of the stored tracking data from these iterations. Leap Motion API 4.0 allows to receive the tracking data that we use. For example, the transformation matrix of the palm is represented as xBasis, yBasis, zBasis, which are the vectors specifying rotation and scale factors for the corresponding axes and origin vector specifying translation factors on all three axes. We also use the coordinates of the position and rotation of the bones. In the model provided by the leap motion controller all fingers contain 4 bones that makes up the anatomy of the finger.

The 3 cameras setup is shown in the Fig. 1 4.

III. Hand Gesture Dataset

In order to evaluate the effect of the new feature, we propose a new dataset. The dataset contains in total of 7 gestures (Fig. 1 3). We chose such gestures because it was convenient for us to work with them. Probably such gestures are used in China to show numbers. However, we perceived such gestures as random. Each gesture is repeated at least 200 times for one camera position. The cameras have 6 positions and the data set contains more than 6697 samples in total. However, the data is divided into 2 datasets. The first is used to tracking gestures for a static position of devices (1297 smaples). The second one was recorded for a static position of gestures and various positions of devices (5400 smaples). While collecting data in the static position of the leap motion devices, the distance between devices No. 2 and No. 3 was 0.19 meters, and the distance between device No. 1 and the XY plane (Fig. 1 4) was also 0.19 meters. In this case, gestures corresponding to numbers 1-4, 6, 7 from the Fig. 1 3 were used. Sensor positions for recordings with different camera positions are given in the Table I. The distance between the device 3 and the XY plane did not change and was 0.32 meters. For each iteration with different positions of devices, 3 gestures corresponding to the numbers 4, 5, 7 from the Fig. 1 3 were recorded. The values of the **Recorded gestures** column in the table indicate the total number of 3 gestures for this static iteration of the data recording process.

Below in the section *Classification of gestures*, describing the studies for the static position of the cameras, we present different tables with the results of data analysis for single (one) camera, two cameras and three cameras. We used device 1 for the setup with a single camera, devices 2 and 3 for the setup with two cameras.

TABLE I. Data entry parameters for various device positions

No.	Distance between devices No. 2 and No. 3	Recorded gestures
1	0.10 m.	845
2	0.16 m.	923
3	0.22 m.	914
4	0.28 m.	876
5	0.34 m.	911
6	0.40 m.	931

The person performing gestures was required to do it in the visual range of any of 3 Leap Motion Controller devices. The person could move and rotate the arm in any direction while collecting the data.

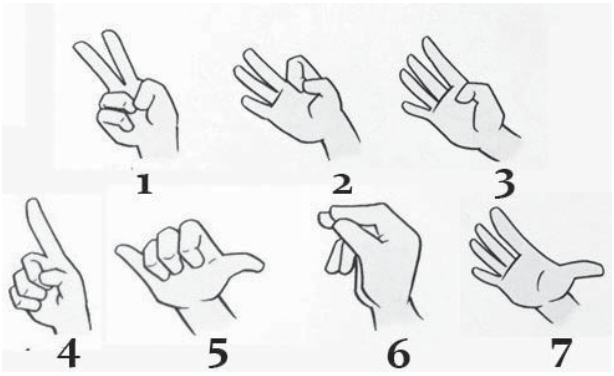

Fig. 3. Gestures in dataset

IV. Leap Motion Data

To work with the data, we used the Leap Motion 4.4.0 SDK in the Unity3D environment. The Leap Motion data model relies on a `Frame` parent object, which represents a frame that contains a set of already selected API interfaces. The objects of the `Frame` class are created this way in the `Update ()` method, data corresponding to the hands can be obtained in each frame as a list `List <Hand> hands =`

`frame.Hands`. The `Hand` class provides access to attributes describing the position, orientation, and movement of a hand detected by Leap Motion. Each `Finger` object contains an identifier that identifies which physical finger of the API leap motion associates with, and the following vectors showing the position, speed and acceleration of the fingers that entering joint (but not the metacarpophalangeal joints of the hand). The fingers are constantly connected with the hand, that means that the angular order of finger identifiers will be invariant. As the fingers move in and out, the previously obtained finger identifier could be is incorrect. Consequently, it may be necessary to take a fingers identifier exchange. All tracked properties, such as speed, will remain contiguous in the API. However, the values obtained from the API output data (for example, the history of positions) will be intermittent if they do not have an appropriate exchange of identifiers. In addition, the `Finger` objects may be invalid, which means that they do not contain reliable tracking data (they do not correspond to a physical finger). Also, invalid objects `Finger` can be the result of a query of an `Finger` object that uses an identifier from an earlier frame, if there are no `Finger` objects with this identifier in the current frame. The object `Finger` created from the *`Finger`* constructor is also invalid. The check of functioning is performed using the `Finger :: isValid()` function.

V. Cameras Setup

To accomplish the task, we collected data from 3 leap motion devices Since the recognition of gestures occurs in the near zone within a hemispherical volume from 0.2 to 1 meter, we proposed the following model for the sensors setup (Fig. 1 4).

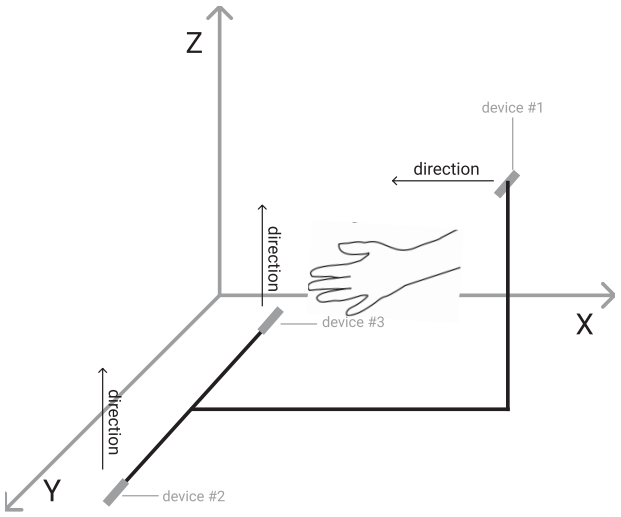

Fig. 4. Diagram of the principal location of devices (leap motion devices)

Before creating the dataset each device could be moved along the respective axis. However the devices are fixed on the corresponding axis of the structure while collecting the data. Thus we have the opportunity to adjust the position of each device before conducting an independent experiment for each position. The accuracy of each device position is 0.01 meters. The Fig. 1 5 shows the exposed sensors at the time of the experiment. The distance between the boundaries of the device No. 2 and No. 3 is 0.40 meters. The distance from the border of the device No. 1 to the XY plane (see Fig. 1 4) is 0.32 meters. This setup is quite flexible and it is very easy to change the distance between cameras. Also this setup is scalable and one can add more cameras if needed.

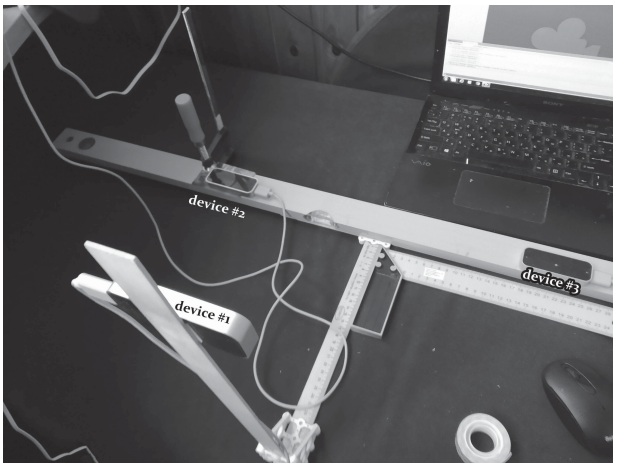

Fig. 5. The photo at the moment of experiment. The distance between devices No. 2 and No. 3 is 0.40 meters. The distance between the plane and the device No. 1 is 0.32 meters

VI. Gesture Classification

We tested logistic Regression, SVC and XGBClassifier. We used python 3 and scikit-learn machine learning library implementations of logistic Regression and SVC [23], [24]. We used python 3 and skikit-learn wrapper interface for XGBoost for XGBClassifier [25].

A. Feature extraction

Feature extraction is an important step in the gesture recognition and we use automatic methods of feature selection. We used the Recursive Feature Elimination(RFE) algorithm from scikit-learn [26]. Logistic regression model is used as an estimator in RFE. We select top 10 features for the dataset using the coefficients of the regression model by training model on the initial set of features and removing 1 feature at each iteration.

We use randomized search on hyper parameters implemented in RandomizedSearchCV [27]. It takes a fixed number of parameter setting sampled from the specified distributions. The 5-fold cross-validation is preformed for each algorithm.

For the logistic regression we search tolerance from 10^-4 to 10^-2, C from 0.7 to 1.2, maximum iterations from 80 to 200.

For the SVC algorithm we search C from 0.7 to 1.2, coef0 from -0.5 to 0.5, tolerance from 10^-4 to 10^-2, maximum iterations from 80 to 200.

For the XGBClassifier we search max depth from 2 to 5, learning rate from 0.05 to 0.2, number of estimators from 75 to 125, minimum loss reduction from 0 to 0.01, minimum child weight from 0 to 2, maximum tree weight estimation delta

step from 0 to 1, subsample ratio for the training instance, subsample ratio of columns when constructing each tree and subsample ratio of columns for each split, in each level from 0.5 to 1, L1 regularization term on weights from 0 to 1, L2 regularization term on weights from 0.5 to 1, scale positive weights from 0.5 to 1.

B. Logistic regression

Logistic regression is one of the basic classification algorithms. It's quite similar to linear regression, but uses sigmoid function (eq. 1).

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{1}$$

If t is considered a linear function, the equation for logistic regression becomes eq. 2.

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \tag{2}$$

The hyper parameter table for 1, 2 and 3 cameras for logistic regression is presented in Table II.

TABLE II. LOGISTIC REGRESSION HYPER PARAMETERS

	1 camera	2 cameras	3 cameras
Tolerance	10^-2	10^-2	10^-2
C	1.2	1.2	1.2
maximum iterations	200	200	200

C. SVC

C-Support Vector Classification algorithm is implemented in the scikit-learn Support Vector Machines module. Support Vector Machines use the concept of decision planes to separate sets of objects that belong to different classes. The libsvm implementation of the algorithm is used which can be found in [28].

The hyper parameter table for 1, 2 and 3 cameras for SVC is presented in Table III.

TABLE III. SVC HYPER PARAMETERS

	1 camera	2 cameras	3 cameras
C	0.7	1.0	1.0
coef0	-0.5	0.5	0.5
Tolerance	10^-4	10^-2	10^-2
maximum iterations	150	200	200

D. XGBClassifier

XGBClassifier is the XGBoost classification model that uses gradient boosting technique. Like other boosting methods, gradient boosting combines weak "learners" into a single strong learner in an iterative fashion. The easiest way to explain it is the least-squares regression setting, where the goal is to "teach" a model `F` to predict values of the form $\widehat{y} = F(x)$ by minimizing the mean squared error $\frac{1}{n}\sum_{i=1}^{n}(\widehat{y}_i - y_i)^2$, where `i` indexes over some training set of size `n` of actual values of the output variable `y`.

The hyper parameter table for 1, 2 and 3 cameras for XGBClassifier is presented in Table IV.

TABLE IV. XGBCLASSIFIER HYPER PARAMETERS

	1 camera	2 cameras	3 cameras
max_depth	4	4	4
learning rate	0.2	0.1	0.2
n_estimators	100	100	150
gamma	10^-2	0	0.05
min_child_weight	0	0	2
max_delta_step	1	0	1
subsample	0.5	1	0.5
colsample_bytree	1	0.5	0.5
colsample_bylevel	0.5	0.5	1
reg_alpha	0	0	1
reg_lambda	0.5	0.5	1
scale_pos_weight	1	0.5	1

VII. EXPERIMENTAL RESULTS

The recognition results for the best found classifiers are presented in tables V to VII.

TABLE V. LOGISTIC REGRESSION 3 CAMERAS REPORT

Class	Precision	Recall	F-score	Support
0	0.93	0.9	0.91	69
1	0.9	0.9	0.9	70
2	0.79	0.78	0.79	69
3	0.67	0.64	0.65	53
4	0.67	0.81	0.73	42
5	0.93	0.91	0.92	57
6	0.9	0.79	0.84	24
avg	0.83	0.83	0.83	384

TABLE VI. SVC 3 CAMERAS REPORT

Class	Precision	Recall	F-score	Support
0	0.91	0.91	0.91	69
1	0.94	0.89	0.91	70
2	0.8	0.8	0.8	69
3	0.67	0.62	0.65	53
4	0.67	0.81	0.73	42
5	0.93	0.89	0.91	57
6	0.84	0.88	0.86	24
avg	0.83	0.83	0.83	384

TABLE VII. XGBCLASSIFIER 3 CAMERAS REPORT

Class	Precision	Recall	F-score	Support
0	0.95	1.0	0.97	69
1	0.94	0.96	0.95	70
2	0.79	0.88	0.84	69
3	0.88	0.68	0.77	53
4	0.8	0.88	0.84	42
5	0.95	0.91	0.93	57
6	1.0	0.88	0.93	24
avg	0.89	0.89	0.89	384

The overall accuracy can be found in Table VIII.

TABLE VIII. OVERALL ACCURACY COMPARISON

	1 camera	2 cameras	3 cameras
Logistic regression	44.53%	61.46%	82.81%
SVC	44.53%	61.72%	83.07%
XGBClassifier	46.61%	68.23%	89.32%

The most important features are hand rotation matrix, y-component of the fingers directions and distal phalanges rotation quaternion. These features don't depend on the size of the person hand which makes the algorithm applicable for anyone.

The recognition results for one camera are quite bad (below 50%) for all algorithms. We focus on the self occlusion problem while preparing the dataset, so we expected such a result. As the table VIII shows, using more cameras does give significant growth in the recognition accuracy. The growth is about 20% when using two cameras instead of one and three cameras instead of two. We got the best result (89.32% accuracy) with the XGBClassifier for the dataset with 7 gestures.

The next step to improve the recognition setup. We do it by finding the best distances between devices No. 2 and No. 3 (Fig. 1 4). In order to that we prepared a new dataset with 3 gestures with the following distances between cameras No. 2 and No. 3: 10, 16, 22, 28, 34, 40 cm. After that we use the method described above, but only use XGBClassifier and information from 3 cameras. The results are shown on the Fig. 1 6

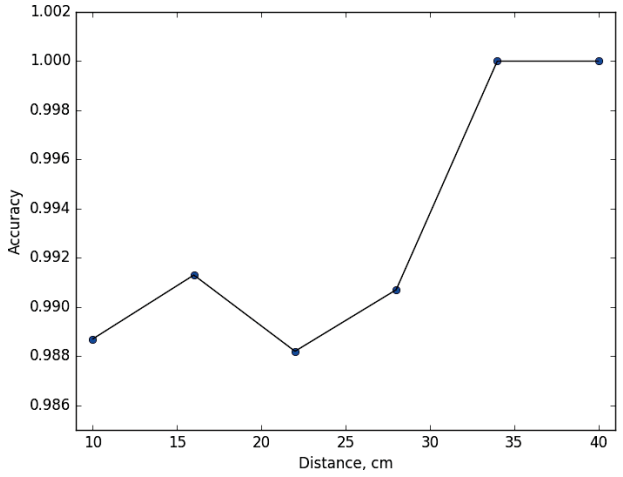

Fig. 6. Accuracy for distance between cameras No. 2 and No. 3

This experimental results indicate that the larger distance between cameras helps recognizing gestures. The optimal distance is about 35 cm, but the results for smaller distances and the best differ by less than 2%. It means that one can still achieve good recognition accuracy by making a smaller system if needed.

VIII. Conclusion

In this paper we investigated the possibility of using tracking data from multiple Leap Motion devices (up to 3 devices) in order to increase the recognition accuracy and overcome the problem of the self occlusion. We proposed our own dataset because there were no available datasets made in similar hardware setup with the 3 Leap Motion Controller devices. We focused on including gestures captured in different positions and angles in it. We didn't restrict the possible hand motion while recording the data. We provided the method of choosing the optimal distance between the cameras and choosing the best algorithm for the recognition. Three algorithms were compared: logistic regression, SVC and XGBClassifier. We have shown that using multiple cameras does increase the accuracy and can be used in systems if enough space for the setup is provided. Our setup doesn't require the user to wear any additional hardware like in non-vision based systems (e.g. glove-based control interfaces) while solving the problem of self-occlusion that often occurs when using vision based systems.

The future work includes expanding the list of possible features on the basis of the raw data from Leap Motion sensors. Also two handed gestures and dynamic gestures need to be analyzed. This will noticeably increase the intuitiveness of the system. Also we plan to use this method in augmented reality training systems to automatically determine user's gestures and actions. The capabilities of our virtual reality application for observing data obtained by the digital holographic microscopy can be increased with this method by providing more types of complex actions which can be performed with the model. Also, testing the performance of the learning and predicting is the possible future work. It is also possible to divide the space available for the Leap Motion controller devices into areas and test the accuracy of the hand gesture recognition in every of these areas. It will also give usefull information about the optimal cameras setup.

Acknowledgment

The work was published with the financial support of the Russian Foundation for Basic Research, grant 17-57-52011 MNT_a.

References

[1] J. P. Wachs, M. Kolsch, H. Stern, Y. Edan, "Vision-based hand-gesture applications. Commun ACM.", *Commun. ACM.* 54., 2011, pp. 60-71.

[2] Premaratne, Prashan, *Historical Development of Hand Gesture Recognition*. Cognitive Science and Technology, 2014.

[3] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, Jian Sun "Realtime and Robust Hand Tracking from Depth", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[4] Youchen Du, Shenglan Liu, Lin Feng, Menghui Chen, Jie Wu "Hand Gesture Recognition with Leap Motion", *CoRR*, 2017.

[5] Chengde Wan, Angela Yao, Luc Van Gool "Direction matters: hand pose estimation from local surface normals", *CoRR*, 2016.

[6] D. Tang, H.J. Chang, A. Tejani, T.K. Kim "Latent regression forest: Structured estimation of 3D articulated hand posture", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[7] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, Jian Sun "Cascaded Hand Pose Regression", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[8] Chetna Naidu, Archana Ghotkar "Hand Gesture Recognition Using Leap Motion Controller", *International Journal of Science and Research (IJSR)*, vol. 5, 2016, pp. 436-441.

[9] Giulio Marin, Fabio Dominio, Pietro Zanuttigh "Hand gesture recognition with leap motion and kinect devices", *IEEE International Conference on Image Processing, ICIP*, 2014, pp. 1565-1569.

[10] Mohamed Mohandes, Salihu Oladimeji, Mohamed Deriche "Prototype Arabic Sign language recognition using multi-sensor data fusion of two leap motion controllers", *Conference: 12th International Multi-Conference on Systems, Signals & Devices (SSD)*, 2015.

[11] Florence Am, Guillaume Lonjon, Didier Hannouche, Rmy Nizard "Effectiveness of Virtual Reality Training in Orthopaedic Surgery", *Arthroscopy, Volume 32, Issue 1*, 2015, pp. 224232.

[12] Jian Li, Hui-qun Fu , Xiu-feng Zhang, Feng-ling Ma, Teng-yu Zhang, Guo-xin Pan, Jing Tao "Research and Development for Upper Limb Amputee Training System Based on EEG and V", *Wearable Sensors and Robots, Volume 399 of the series Lecture Notes in Electrical Engineering*, pp. 263-273.

[13] Huber T., Paschold M., Lang H., Kneist W. "Influence of camera navigation training on team performance in virtual reality laparoscopy", *Journal of Surgical Simulation*, 2015, pp. 3539.

[14] Heisenberg G., Rezaei Y. A., Rothdeutsch T., Heiden W. “Arm prosthesis simulation on a virtual reality L-shaped workbench display system using a brain computer interface”, *Journal of Pain Management 9 (3)*, 2016, pp. 205-214.

[15] Kalenkov G.S. , Kalenkov S.G. , Kiselev V.A. , Klimenko S.V. , Sysoev N.A. , W. Heiden “Visualization of digital holograms of biosamples by means of virtual environment system”, *Proceedings of the International Scientific Conference “Situation Centers and IAS4i for Monitoring and Security” (SCVRT1516)*, Moscow-Protvino, 2016.

[16] Vladimir Aleshin, Valery Afanasiev, Alexander Bobkov, Stanislav Klimenko, Vitaly Kuliev, Dmitry Novgorodtsev “Visual 3D Perception of Motion Environment and Visibility Factors in Virtual Space”, *Transaction on Computer Science XVI, Lecture Notes on Computer Science 7380*, 2012.

[17] Valery Afanasiev, Vitaly Kuliev, Stanislav Klimenko, Vladimir Aleshin, Dmitry Novgorodtsev, Alexander Bobkov “Visual 3D Perception of the Ski Course and Visibility Factors at Virtual Space”, *International Conference on Cyberworlds*, Banff, Alberta Canada, 2011, pp. 222-226.

[18] Kiselev V. A., Klimenko A. S., Klimenko S.V., Mikchailyuk M. V., Pestrikov V. I., Khlamov M. A., Chuvilin K. V., Foursa M. V., Hakim N. L., Shih T. K. “3D Gesture recognition hardware tools for the Virtual Environment system”, *Proceedings of the International Scientific Conference on Physical and Technical Informatics (CPT1617)*, Moscow-Protvino, 2017.

[19] Leap Motion official documentation, Leap Motion gestures, Web: https://developer-archive.leapmotion.com/documentation/v2/csharp/devguide/Leap_Gestures.html

[20] Kiselev V. A., Pigaryov V. A., Khlamov M. A., Chuvilin K. V. “The use of multiple Leap Motion optical tracking sensors to construct models in a virtual environment”, *SCVRT2018 Proceedings of the International Scientific Conference of The Moscow Institute of Physics and Technology (State University) of The Institute of Computing for Physics and Technology*, Moscow-Protvino, 2018, pp. 132-138.

[21] Kiselev V. A., Pigaryov V. A., Khlamov M. A., Chuvilin K. V. “The use of gesture recognition in the augmented reality training application for the ACS TP NPP”, *SCVRT2018 Proceedings of the International Scientific Conference of The Moscow Institute of Physics and Technology (State University) of The Institute of Computing for Physics and Technology*, Moscow-Protvino, 2018, pp. 149-155.

[22] Leap Motion official documentation, Web: http://developer.leapmotion.com/

[23] Scikit-learn official documentation, Logistic Regression classifier, Web: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[24] Scikit-learn official documentation, C-Support Vector Classification, Web: https://scikit-learn.org/stable/modules/generated/sklearn.\newlinesvm.SVC.html

[25] Scikit-Learn Wrapper interface for XGBoost official documentation, Web: https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn

[26] Scikit-Learn Feature ranking with recursive feature elimination official documentation, Web: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html

[27] Scikit-Learn randomized search on hyper parameters official documentation, Web: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[28] Chih-Chung Chang, Chih-Jen Lin “LIBSVM: A library for support vector machines”, *ACM Transactions on Intelligent Systems and Technology. 2*, 2007.