Human Gesture Classification by Brute-Force Machine Learning for Exergaming in Physiotherapy

Francis Deboeverie, Sanne Roegiers, Gianni Allebosch, Peter Veelaert and Wilfried Philips
Department of Telecommunications and Information Processing,
Image Processing and Interpretation, UGent/iMinds,
St-Pietersnieuwstraat 41, 9000 Ghent, Belgium
Email: Francis.Deboeverie@ugent.be

***Abstract*—In this paper, a novel approach for human gesture classification on skeletal data is proposed for the application of exergaming in physiotherapy. Unlike existing methods, we propose to use a general classifier like Random Forests to recognize dynamic gestures. The temporal dimension is handled afterwards by majority voting in a sliding window over the consecutive predictions of the classifier. The gestures can have partially similar postures, such that the classifier will decide on the dissimilar postures. This brute-force classification strategy is permitted, because dynamic human gestures show sufficient dissimilar postures. Online continuous human gesture recognition can classify dynamic gestures in an early stage, which is a crucial advantage when controlling a game by automatic gesture recognition. Also, ground truth can be easily obtained, since all postures in a gesture get the same label, without any discretization into consecutive postures. This way, new gestures can be easily added, which is advantageous in adaptive game development. We evaluate our strategy by a leave-one-subject-out cross-validation on a self-captured stealth game gesture dataset and the publicly available Microsoft Research Cambridge-12 Kinect (MSRC-12) dataset. On the first dataset we achieve an excellent accuracy rate of 96.72%. Furthermore, we show that Random Forests perform better than Support Vector Machines. On the second dataset we achieve an accuracy rate of 98.37%, which is on average 3.57% better then existing methods.**

I. Introduction

Human gesture recognition [1], [2], [3], [4] is defined as automatically identifying and interpreting human body movements using a set of sensors. Human body movements may be performed with the hands, arms, body, head, etc. Human gestures may include for instance standing, lying, bending, sitting, walking, jumping, etc. Human gesture recognition has been heavily studied because it plays an important role in human computer interaction applications [5], [6], [7], [8] such as health monitoring systems, surveillance systems, motion analysis in sports, and human behavior analysis.

In this paper we perform human gesture recognition for the application of exergaming in pysiotherapy. It is not always easy for children in a rehabilitation or fitness program to sustain their efforts. Exergaming, which combines exercise and gaming, can motivate children (and adults) to keep moving. Exergaming also offers the possibility for remote monitoring and coaching in an e-environment. Coaches and therapists can select the games with the desired level of difficulty and remotely monitor the children's progress. The project wE-MOVE [1] is an innovative solution for exergaming that remotely supports the childrens rehabilitation and prompts them to move. The software consists of a gross motoric exergame and a platform allowing both the child and the coach to monitor progress. In this framework, automatic human gesture classification is needed to control the game.

Human gesture recognition is mainly performed on RGB-D (Red, Green, Blue and Depth) data [9], [10], [11], [12], [6], [13] or on skeleton data [14], [15], [16], [17], [18], [13], [19], [20], [21], where skeletal data can be extracted from RGB-D data. To recognize static gestures (i.e. postures, such as sitting, standing or lying), a general classifier [22] or a template-matcher is generally used. Dynamic gestures (i.e. consecutive postures, such as running, jumping) have a temporal dimension, which is traditionally handled by Hidden Markov Models (HMM) or motion based models. When classifying many dynamic human gestures, constructing these models is complex and time consuming. Also, the models are usually not generally applicable, so that it is difficult to extend the classifier with new gestures. Furthermore, building ground truth requires a discretization into consecutive postures of the dynamic gesture, which is again complex and time consuming.

In this work, we propose a novel approach which uses a general classifier [22], such as Random Forests (RF) [23], to recognize dynamic gestures in skeletal data. The gestures can have partially similar postures, such that the classifier will decide on the dissimilar postures. The temporal dimension is handled afterwards by majority voting in a sliding window over the consecutive predictions of the classifier. This way of online continuous human gesture recognition can recognize dynamic gestures in an early stage, since we build up reliability when sliding the window. This is a crucial advantage when controlling a game by automatic gesture recognition, since the feedback to the user should be given in real time. Also, ground truth can be easily obtained, since all postures in a dynamic gesture get the same label, without any discretization into consecutive postures. Furthermore, the classifier is general and can be easily extended with new gestures, which is advantageous in adaptive game development.

We elaborate RF combined with majority voting in a sliding

[1]More details can be found at http://www.iminds.be/en/projects/2015/03/11/we-move

window for human gesture recognition. RF are considered amongst the most robust classifiers currently available, and have been shown to perform as well as or better than Support Vector Machines (SVM), while being much less computationally expensive to train or execute. We consider normalized skeleton data provided by the Microsoft Kinect v2 [24]. The Kinect device is a motion sensing device which was originally designed for the Microsoft Xbox 360 video game console where the user is the controller. The device is composed of multiple sensors: an RGB camera to capture a colored video stream, a depth camera to compute the 3D environment and an infrared light sensor. Skeletal data is extracted from RGB-D images. Kinect v2 can detect up to six users at the same time and compute their skeletons in 3D with 25 joints representing body junctions like the feet, knees, hips, shoulders, elbows, wrists, head, etc. For each pose of a skeleton the position numbers and the angle numbers of the joints form a feature vector. These feature vectors are used to train a RF and classify gross motoric movements.

In our experiments, we evaluate the above-mentioned strategy on two datasets. The first dataset is a self-captured stealth game gesture dataset, including 5 human subjects performing 23 specific movements for a gross motor stealth game. The gestures have partial similarity, such as walking and running, or jumping low and jumping high. Our method is evaluated by leave-one-subject-out cross-validation (LOSubO CV) [25]. In the results we will show that RF and majority voting in a sliding window achieves an accuracy rate of 96.72%. Furthermore, we will show that RF perform better than SVM. The second dataset is the publicly available Microsoft Research Cambridge-12 Kinect (MSRC-12) gesture dataset [26]. The dataset includes 30 people performing 12 gestures. Among all publicly available datasets [27], [28], the MSRC-12 dataset is best suited for our application, since the ground truth annotation for each sequence marks the action point of the gesture as a single time instance at which the presence of the action is clear and that can be uniquely determined for all instances of the action. For a real-time application, such as a game, this is the point at which a recognition module is required to detect the presence of the gesture. Using LOSubO CV We achieve an accuracy rate of 98.37%, which is on average 3.57% better than existing methods [29], [30], [31].

The remainder of the paper is as follows. In Section II, we give an overview of the RF classifier. In Section III, we explain the strategy of majority voting in a sliding window. In Section IV, we evaluate and compare the proposed method on two datasets. Finally, in Section V we conlude the paper.

II. Random Forests

In this section we shortly review some basic work on RF for classification problems. A RF [23], [22] is an ensemble classifier composed of several binary decision trees, each trying to solve the same task.

Like any classifier, a decision tree takes a set of features as input, and returns a class label as its output. A decision tree consists of a set of nodes that are connected by branches. Non-leaf-nodes are called decision-nodes. In binary decision trees, each decision-node has exactly two child-nodes. Each branch that connects a decision-node with its two child-nodes, corresponds to a binary decision value. During classification, each decision-node compares a specific feature value with a threshold value, and then follows one of the two branches that corresponds to binary outcome of this test. This process is repeated until a leaf-node is reached. Leaf-nodes in a decision tree correspond to class labels, and thus represent the final decision.

Although decision tree classifiers are easy to use and can be implemented extremely efficiently, training a decision tree is a difficult problem. During tree construction, one of the several available features have to be chosen for the binary test at each decision-node. Common methods to train decision trees are the ID3 algorithm and its successor, C4.5 [32], which resort to a greedy heuristic approach to determine the splitting criteria. At each decision-node, the information gain (also know as mutual information) for each possible splitting criterion is calculated, and the criterion yielding the highest information gain is chosen.

When choosing a classifier, we consider the bias-variance trade-off. The bias of a classifier represents the number of samples that would be consistently misclassified, if the classifier would be trained on different subsets of the complete training population. The variance of a classifier measures the variability of the number of misclassifications when different subsets of the training population are used. In general, classifiers that are able to fit the data well, exhibit low bias but high variance, while classifiers that result in more general decision boundaries yield high bias but low variance. While a low-bias, low-variance classifier is desirable, lowering the variance of a classifier, often implicitly increases the bias, and vice versa.

A decision tree is a low-bias high-variance classifier. An obvious way to lower the variance is to train multiple decision trees on different subsets of the population, and then use the average decision (i.e. regression) or a voting scheme (i.e. classification). However, in practice only a limited amount of training data is available, instead of the whole population. A well know method to approximate the distribution of the complete population if only a limited number of observations are available, is to construct multiple training samples by bootstrapping the original training dataset. Bootstrapping is a resampling method that is known by statisticians as sampling with replacement. Since sampling with replacement means that samples can be selected multiple times, this means that each bootstrapped sample contains duplicated data. As a result, each classifier that is trained on a different bootstrapped sample, will have slightly different decision boundaries. By aggregating the resulting decisions of each of these possibly high-variance classifiers, by means of averaging or voting, a low-variance classifier is obtained. This concept of bootstrap aggregating is called bagging. For bagging in RF, the number of trees in the forest is an important parameter to choose. The larger the better, but also the longer it will take to compute. In addition, the results will stop getting significantly better

Figure 1. In the training phase of the classifier, all postures in a dynamic gesture (grey zone) get the same label.

beyond a critical number of trees.

RF use bagging to reduce the variance of the final ensemble classifier, compared to a single decision tree. However, bagged trees exhibit a high correlation because of the duplicates in their training data and the similarity in their training method. Highly correlated trees would therefore make the same errors in similar regions of the feature space. This means that reducing the variance by means of bagging, increases the bias of the resulting classifier. To decrease the bias of the ensemble classifier, RF ensure diversity of the tree classifiers by introducing randomness into the splitting criterion: each time the training set is split, only a randomly selected subset of all features is considered for selecting the feature for the next decision-node.

Thus, a RF, is a low-bias, low-variance ensemble classifier, trained by bootstrapping and random feature selection. RF have been shown to be almost invariant to overfitting and robust to noise. Finally, classification by means of RF can be implemented extremely efficient, since each decision tree can simply be represented by a set of conditional statements.

As feature input for the classifier, we consider normalized skeleton data provided by the Microsoft Kinect v2 [24]. The skeletons are computed in 3D with 25 joints representing body junctions, where each joint consist of a position encoded in three numbers and an angle encoded as four quaternion numbers, which is a common encoding method in robotics. For each pose of a skeleton the position numbers and the quaternion numbers of 25 joints form a 175-dimensional feature vector. These feature vectors are used to train and classify gross motoric movements.

In the training phase of the classifier, all postures in a dynamic gesture get the same label, as illustrated in Figure 1, where all different postures in the grey zone get the same label. Different gestures can have partially similar postures. In this case, the classifier will decide on the dissimilar postures. In the results we will show that the selection of human gestures in many applications shows sufficient dissimilarity in the postures. Using this brute-force strategy, ground truth can be easily obtained, because only the beginnings and the endings of the gestures have to be indicated, without any discretization into consecutive postures. Furthermore, the classifier can be easily extended with new gestures, which is advantageous in adaptive game development.

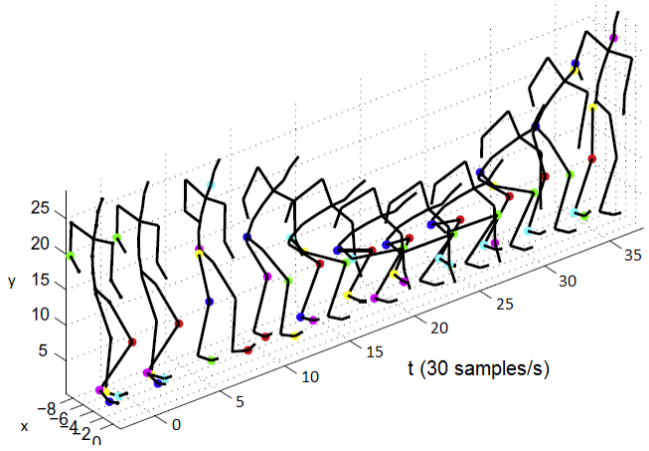

Figure 2. The temporal dimension of a human gesture in the skeletal data.

III. Majority voting in a sliding window

In an online continuous human gesture recognition mode, the temporal dimension in human gestures is handled by majority voting in a sliding window over the consecutive predictions of the classifier. An example of the temporal dimension of a human gesture, i.e. bowing, in the skeletal data is illustrated in Figure 2.

In a sliding window, we compute the observation probability of a human gesture using a number of continuing observations within the sliding window. The final gesture type is decided by a majority vote of all recognition results that are obtained between the start and end point of the window. For optimal classification, the length of the time window is dependent on the duration of the gestures, and thus the selection of gestures and the subjects performing the gestures. In this work, the size of the sliding window w_s is determined empirically; in our work we found that one second was a good value, which means a buffer of 30 classifier predictions at a skeleton rate of 30Hz.

Using a sliding window technique in human gesture recognition introduces many advantages. The first advantage is that it improves the classification performance of gesture recognition greatly, as we will show in the results. A second advantage is that it reduces the undesirable effect of an abrupt change of observations within a short interval that can be caused by erroneous and incomplete skeletons. The third advantage is that human dynamic gestures can be recognized in an early stage, since we build up reliability when sliding the window. This is a crucial advantage when controlling a game with automatic gesture recognition, since the feedback to the user should be given in real time.

IV. Results

In this section, we evaluate the proposed strategy on two datasets: a self-captured stealth game gesture dataset and the Microsoft Research Cambridge-12 Kinect (MSRC-12) dataset.

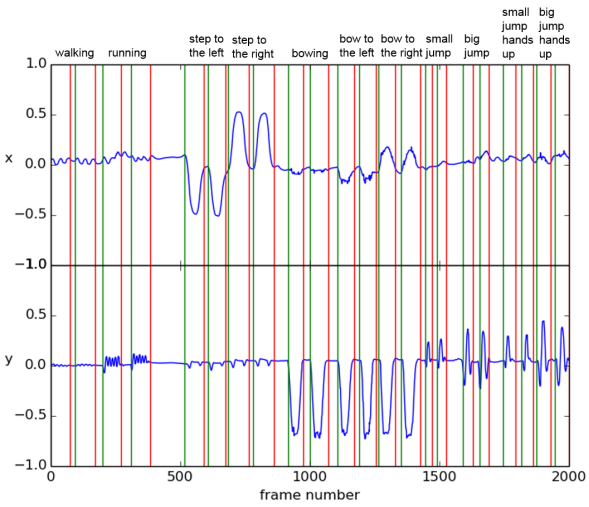

Figure 3. The normalized x and the y coordinates of the *spine_mid* joint over time for 11 gestures. The green and the red lines indicate the beginnings and the endings of the gestures (ground truth), respectively.

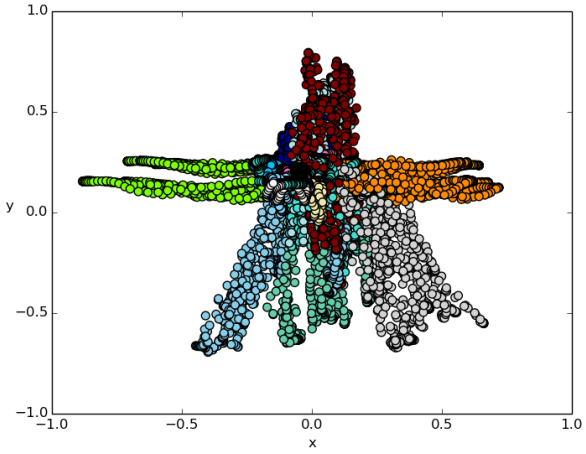

Figure 4. The y coordinates over the x coordinates of the *spine_mid* joint indicated in different colors for all gestures.

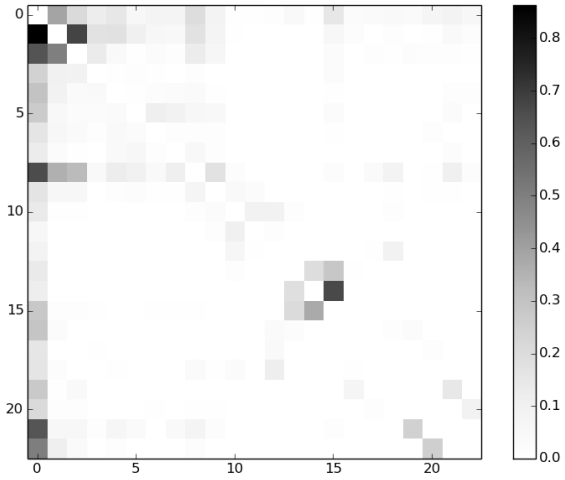

Figure 5. The percentages of overlapping postures between the different gestures.

A. Stealth game gesture dataset

In the first dataset, we recorded human subjects performing 23 specific movements for a gross motor stealth game, recorded with the Kinect v2 at the Sportlab of the department of Movement and Sport Sciences at Ghent University in Belgium. The dataset includes five subjects that repeat 23 exercises of a stealth game three times (26534 samples at 30Hz). Between every exercise the subject takes the neutral posture, which is standing up with the arms along the body.

This is the list of 23 movements with their corresponding label: 0: neutral, 1: walking, 2: running, 3: step to the left, 4: step to the right, 5: bowing, 6: bow to the left, 7: bow to the right, 8: little jump, 9: big jump, 10: little jump with the hands up, 11: big jump with the hands up, 12: climbing, 13: flying like a hummingbird, 14: flying with small arm movements, 15: flying with big arm movements, 16: punch to the left, 17: punch to the right, 18: pushing forward, 19: high kick to the left, 20: high kick to the right, 21: low kick to the left, 22: low kick to the right.

The graph in Figure 3 plots the normalized x and the y coordinates of the *spine_mid* joint over time for 11 gestures. The green and the red lines indicate the beginnings and the endings of the gestures (ground truth), respectively. In the coordinates we can clearly distinguish the different expected patterns of the gestures.

Figure 4 plots the y coordinates over the x coordinates of the *spine_mid* joint indicated in different colors for all gestures. This graph shows that the feature vectors of all postures in each dynamic gesture form a continuous cluster which is separable from the clusters of the other gestures.

Figure 5 shows the percentages of overlapping postures between the different gestures. The highest percentages of overlap are noticed between the gestures walking and running (up to 80%), and between the flying gestures (up to 60%). Also, all gestures have a high overlap with the neutral gesture. This is because the neutral gesture occurs between every two other gestures. During annotation, a few samples of the neutral gesture may be included in another gesture. In our approach, despite the overlap, the remaining non-overlapping postures are sufficient to classify gestures.

We made an annotation of the dataset by indicating the beginnings and the ending of the gestures. The number of annotations we had to made is 5 persons $\times$ 23 gestures $\times$ 3 repeats $\times$ 2 annotations $=$ 690 annotations, in stead of 26534 annotations in the case of a discretization of the dynamic gesture.

We evaluate our method by leave-one-subject-out cross-validation (LOSubO CV) [25], which is the most widely adopted evaluation protocol in action recognition algorithms towards maturity and robustness for real-world applications.

Table I
THE CLASSIFICATION ACCURACY RATES ON THE STEALTH GAME GESTURE DATASET WHEN EVALUATING AFTER 50% AND 100% FINISHING THE EXERCISE USING RF, SVM LINEAR AND SVM POLYNOMIAL COMBINED WITH MAJORITY VOTING IN A SLIDING WINDOW, RESPECTIVELY.

Method	Accuracy(%) 50% finished	Accuracy(%) 100% finished
SVM linear ($C = 4.7$) + SW	83.34 $\pm$ 7.32	94.12 $\pm$ 0.09
SVM polynomial ($D = 13$) + SW	83.71 $\pm$ 4.92	94.80 $\pm$ 0.03
RF ($N = 27$) + SW	88.26 $\pm$ 4.23	96.72 $\pm$ 0.02

LOSubO CV means that the classifier is trained with all but one subject and tested with the unseen data. This is repeated for all subjects and the average of the outcomes as the final result is reported. Thus, in our case we perform a 5-fold cross-validation.

For the RF classifier, we choose the number of decision trees N in the forest equal to 27, which we determined empirically by a parameter sweep. Beyond this number of trees, the results stop getting significantly better. The size of the sliding window (SW) w_s is equal to 30. The window is initialized with labels of class 0 (neutral). As a comparison, we also test SVM in a one-against-one approach for multi-class classification using a linear and polynomial kernel, respectively. For the SVM with a linear kernel, we choose the penalty parameter C of the error term equal to 4.7. For the SVM with a polynomial kernel, we choose the polynomial degree D equal to 13. These values have been determined empirically for an optimal accuracy rate on the dataset. Table I presents the accuracy rates when evaluating the classification after 50% and 100% finishing the exercise, respectively. The accuracy is measured as the set of labels predicted for a sample that exactly match the corresponding set of labels in the ground truth. The observational latency is an important evaluation criterion in our game application. After 50% finishing the exercise, the RF +SW already classifies 88.26% of the gestures correctly. The classification accuracy increases to 96.72% after 100% finishing the exercise. Furthermore, these numbers also shows that the RF classifier performs better and faster than the SVM classifiers. The training times of the RF classifier, the linear SVM and the polynomial SVM are 12.65 seconds, 23.59 seconds and 27.78 seconds, respectively.

The precision, recall and f1-scores per human gesture are presented in the bar chart in Figure 6. Overall, the average precion, recall and f1-score, weighted with the number of class labels, are all equal to 0.97. Our method has a little less performance in recall on jumping and walking gestures, because these gestures have many similar postures with the neutral posture. This is further illustrated in the confusion matrix in Figure 7. The confusion matrix is a matrix which shows the accuracy of a classification algorithm, where each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. We can clearly see that our method has the biggest confusion in the classification of non-neutral gestures as neutral gestures, which is due to the accuracy of the ground truth and the occurrence

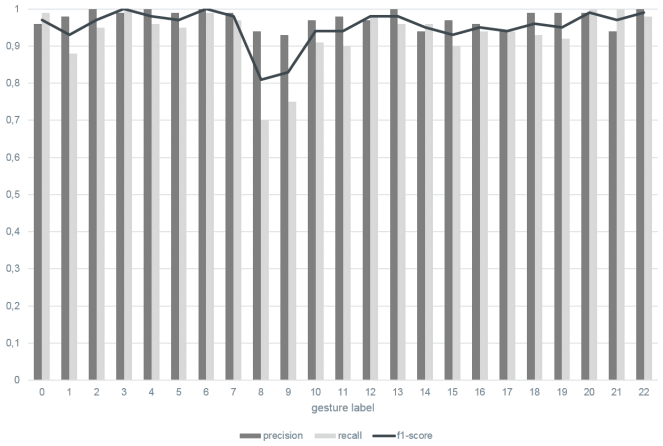

Figure 6. The precision, recall and f1-scores per human gesture.

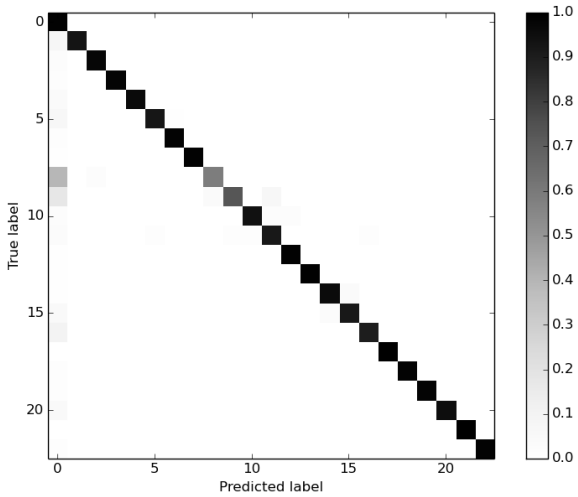

Figure 7. The confusion matrix showing the accuracy of the proposed classification algorithm.

of neutral postures in the non-neutral gestures. Regarding the accuracy of the ground truth, the beginnings and endings of the gestures can include a few overlapping neutral postures, causing the classifier to classify non-neutral gestures as neutral gesture.

Figure 8 presents a visualization of the application output. On the right hand side the skeleton of the posture of a subject performing gesture 15 (flying with big arm movements) is shown. On the left hand side we see the observational probability per gesture class in the sliding window. The probability of the class 0 (neutral) is decreasing to zero, the probability of the class 15 would increase to one in case of a perfect classification. However, in this case, the RF classifier makes a few mistakes by predicting class 14 (flying with small arm movements), due to the similar postures with class 15. These errors are handled by majority voting for the final class decision, which is printed in red color on the right hand side. Even though class 14 and 15 have similar postures, the classifier is still able to decide on the dissimilar postures. This

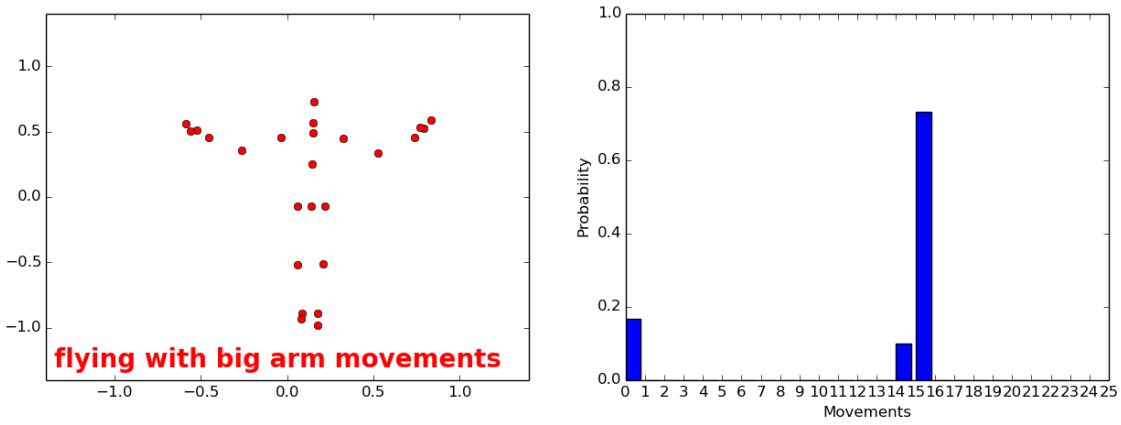

Figure 8. Visualization of the application output.

figure also demonstrates the advantage that human dynamic gestures can be recognized in an early stage, since we build up reliability in the sliding window when performing the exercise.

B. MSRC-12 dataset

The second dataset is the Microsoft Research Cambridge-12 Kinect (MSRC-12) gesture dataset [26], which consists of sequences of human movements, represented as body-part locations, and the associated gesture to be recognized by the system. The dataset includes 594 sequences and 719359 frames, approximately six hours and 40 minutes, collected from 30 people performing 12 gestures. In total, there are 6244 gesture instances. The motion files contain tracks of 20 joints estimated using the Kinect Pose Estimation pipeline. The body poses are captured at a sample rate of 30Hz with an accuracy of about two centimeters in joint positions. The list of movements with their corresponding label: 0:lift outstretched arms, 1:duck, 2:push right, 3:goggles, 4:wind it up, 5:shoot, 6:bow, 7:throw, 8:had enough, 9:change weapon, 10:beat both, 11: kick. Among all publicly available datasets [27], [28], the MSRC-12 dataset is best suited for our application, since the ground truth annotation for each sequence marks the action point of the gesture as a single time instance at which the presence of the action is clear and that can be uniquely determined for all instances of the action. For a real-time application, such as a game, this is the point at which a recognition module is required to detect the presence of the gesture.

For the RF classifier, we choose the number of decision trees N in the forest equal to 27, which gives an optimal classification rate on this dataset. The size of the sliding window is again $w_s = 30$. We compare our method to the methods in [29], [30], [31], which are the highest performing methods on this dataset as also reported in [27]. The methods implement human gesture recognition by decision forest based feature selection, a temporal hierarchy of covariance descriptors and sequence matching, respectively. Table II presents the accuracy rates when evaluating at the time instance marked in the ground truth of the MSRC-12 dataset using LOSubO CV as evaluation protocol (30-fold cross-validation).

Table II
THE ACCURACY RATES WHEN EVALUATING AT THE TIME INSTANCE MARKED IN THE GROUND TRUTH OF THE DATASET USING THE PROPOSED RF +SW METHOD AND THE METHODS IN [29], [30], [31].

Method	Accuracy(%)
RDF-selected features [29]	94.03
Cov3DJ [30]	93.60
ESM [31]	96.76
RF ($N = 23$) + SW	98.37

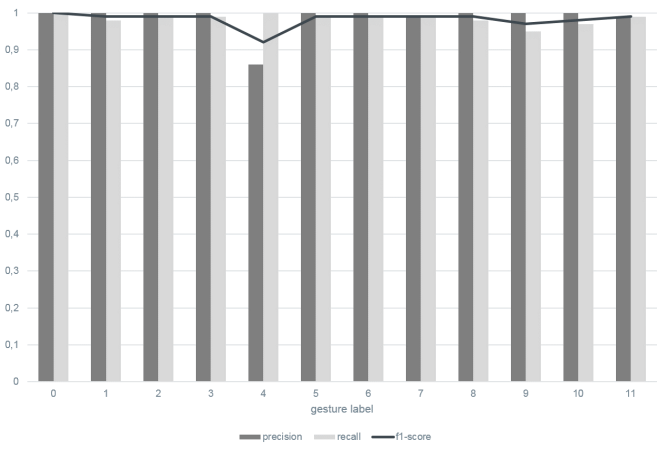

Figure 9. The precision, recall and f1-scores per human gesture on the MSRC-12 dataset.

Our method achieves an accuracy rate of 98.37%, which is on average 3.57% better then the methods to which we compare.

The precision, recall and f1-scores per human gesture are presented in the bar chart in Figure 9. Overall, the average precion, recall and f1-score, weighted with the number of class labels, are all equal to 0.98, which is among the highest in literature.

V. Conclusion

In this work, we proposed a novel approach for human gesture classification on skeletal data provided by Microsoft Kinect. We use Random Forests to recognize dynamic human gestures, where the temporal dimension is handled afterwards by majority voting in a sliding window over the consecutive predictions of the classifier. The gestures to be recognized have partially similar postures, such that the classifier decides on the dissimilar postures. We showed that this brute-force classification strategy is permitted because the selection of human gestures in many applications shows sufficient dissimilar postures. This way, ground truth can be easily obtained, because only the beginnings and the endings of the gestures have to be indicated, without any discretization into consecutive postures. Furthermore, the classifier can be easily extended with new gestures, which is advantageous in adaptive game development. Additionally, online continuous human gesture recognition can recognize dynamic gestures in an early stage, which is a crucial advantage when controlling a game by automatic gesture recognition. We evaluated our strategy by a leave-one-subject-out cross-validation on a self-captured stealth game gesture dataset and the Microsoft Research Cambridge-12 Kinect (MSRC-12) dataset. On the first dataset we achieved an accuracy rate of 96.72%. Moreover, we showed that in this application Random Forests perform better than Support Vector Machines. On the second dataset we achieved an accuracy rate of 98.37%, which is on average 3.57% better then existing methods. In this work, we proved that the proposed simple brute-force strategy of using a general classifier in combination with majority voting in a sliding window provides excellent classification results, while the annotation process went very fast.

References

[1] J. K. Aggarwal and L. Xia, "Human activity recognition from 3d data: A review," *Pattern Recognition Letters*, vol. 48, pp. 70–80, 2014.

[2] R. Lun and W. Zhao, "A survey of applications and human motion recognition with microsoft kinect," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 29, no. 05, 2015.

[3] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, 2015.

[4] L. L. Presti and M. La Cascia, "3d skeleton-based human action classification: A survey," *Pattern Recognition*, 2015.

[5] P. Paliyawan, K. Sookhanaphibarn, W. Choensawat, and R. Thawonmas, "Body motion design and analysis for fighting game interface," in *Computational Intelligence and Games (CIG), 2015 IEEE Conference on*, 2015, pp. 360–367.

[6] N. Li, Y. Dai, R. Wang, and Y. Shao, "Study on action recognition based on kinect and its application in rehabilitation training," in *Big Data and Cloud Computing (BDCloud), 2015 IEEE Fifth International Conference on*, 2015, pp. 265–269.

[7] P. K. Pisharady and M. Saerbeck, "Recent methods and databases in vision-based hand gesture recognition: A review," *Computer Vision and Image Understanding*, vol. 141, pp. 152–165, 2015.

[8] M. Devi, S. Saharia, and D. Bhattacharyya, "Dance gesture recognition: A survey," *International Journal of Computer Applications*, vol. 122, no. 5, 2015.

[9] H. Kim, S. Lee, Y. Kim, S. Lee, D. Lee, J. Ju, and H. Myung, "Weighted joint-based human behavior recognition algorithm using only depth information for low-cost intelligent video-surveillance system," *Expert Systems with Applications*, vol. 45, pp. 131–141, 2016.

[10] M. Liu and H. Liu, "Depth context: a new descriptor for human activity recognition by using sole depth sequences," *Neurocomputing*, vol. 175, pp. 747–758, 2016.

[11] A.-A. Liu, N. Xu, Y.-T. Su, H. Lin, T. Hao, and Z.-X. Yang, "Single/multi-view human action recognition via regularized multi-task learning," *Neurocomputing*, vol. 151, pp. 544–553, 2015.

[12] H. Eum, C. Yoon, H. Lee, and M. Park, "Continuous human action recognition using depth-mhi-hog and a spotter model," *Sensors*, vol. 15, no. 3, pp. 5197–5227, 2015.

[13] F. Jiang, S. Zhang, S. Wu, Y. Gao, and D. Zhao, "Multi-layered gesture recognition with kinect," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 227–254, 2015.

[14] W. Ding, K. Liu, X. Fu, and F. Cheng, "Profile hmms for skeleton-based human action recognition," *Signal Processing: Image Communication*, 2016.

[15] G. Zhu, L. Zhang, P. Shen, and J. Song, "An online continuous human action recognition algorithm based on the kinect sensor," *Sensors*, vol. 16, no. 2, p. 161, 2016.

[16] ——, "Human action recognition using multi-layer codebooks of key poses and atomic motions," *Signal Processing: Image Communication*, 2016.

[17] G. Lu, Y. Zhou, X. Li, and M. Kudo, "Efficient action recognition via local position offset of 3d skeletal body joints," *Multimedia Tools and Applications*, pp. 1–16, 2015.

[18] S. Boubou and E. Suzuki, "Classifying actions based on histogram of oriented velocity vectors," *Journal of Intelligent Information Systems*, vol. 44, no. 1, pp. 49–65, 2015.

[19] J. Ding and C.-W. Chang, "Feature design scheme for kinect-based dtw human gesture recognition," *Multimedia Tools and Applications*, pp. 1–16, 2015.

[20] ——, "An eigenspace-based method with a user adaptation scheme for human gesture recognition by using kinect 3d data," *Applied Mathematical Modelling*, vol. 39, no. 19, pp. 5769–5777, 2015.

[21] X. Chen and M. Koskela, "Skeleton-based action recognition with extreme learning machines," *Neurocomputing*, vol. 149, pp. 387–396, 2015.

[22] Y. Kodratoff, *Introduction to machine learning*. Morgan Kaufmann, 2014.

[23] L. Breiman, "Random forests," *Journal of Machine Learning*, vol. 45, no. 1, pp. 5–32, oct 2001.

[24] Z. Zhang, "Microsoft kinect sensor and its effect," *MultiMedia, IEEE*, vol. 19, no. 2, pp. 4–10, 2012.

[25] S. Arlot, A. Celisse *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[26] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *CHI*, J. A. Konstan, E. H. Chi, and K. Höök, Eds. ACM, 2012, pp. 1737–1746.

[27] J. Zhang, W. Li, P. O. Ogunbona, P. Wang, and C. Tang, "Rgb-d-based action recognition datasets: A survey," *arXiv preprint arXiv:1601.05511*, 2016.

[28] S. Ruffieux, D. Lalanne, E. Mugellini, and O. A. Khaled, "A survey of datasets for human gesture recognition," in *Human-Computer Interaction. Advanced Interaction Modalities and Techniques*, 2014, pp. 337–348.

[29] F. Negin, F. Özdemir, C. B. Akgül, K. A. Yüksel, and A. Erçil, "A decision forest based feature selection framework for action recognition from rgb-depth cameras," in *Image Analysis and Recognition*, 2013, pp. 648–657.

[30] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban, "Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations." in *IJCAI*, vol. 13, 2013, pp. 2466–2472.

[31] H.-J. Jung and K.-S. Hong, "Enhanced sequence matching for action recognition from 3d skeletal data," in *Computer Vision–ACCV 2014*, 2014, pp. 226–240.

[32] J. R. Quinlan, *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.