J Med Syst (2016) 40: 137
DOI 10.1007/s10916-016-0493-6

PATIENT FACING SYSTEMS

The adaptation of GDL motion recognition system to sport and rehabilitation techniques analysis

Tomasz Hachaj[1] · **Marek R. Ogiela**[2]

Received: 2 March 2016 / Accepted: 6 April 2016 / Published online: 22 April 2016

Abstract The main novelty of this paper is presenting the adaptation of Gesture Description Language (GDL) methodology to sport and rehabilitation data analysis and classification. In this paper we showed that Lua language can be successfully used for adaptation of the GDL classifier to those tasks. The newly applied scripting language allows easily extension and integration of classifier with other software technologies and applications. The obtained execution speed allows using the methodology in the real-time motion capture data processing where capturing frequency differs from 100 Hz to even 500 Hz depending on number of features or classes to be calculated and recognized. Due to this fact the proposed methodology can be used to the high-end motion capture system. We anticipate that using novel, efficient and effective method will highly help both sport trainers and physiotherapist in they practice. The proposed approach can be directly applied to motion capture data kinematics analysis (evaluation of motion without regard to the forces that cause that motion). The ability to apply pattern recognition methods for GDL description can be utilized in virtual reality environment and used for sport training or rehabilitation treatment.

This article is part of the Topical Collection on *Patient Facing Systems*

✉ Tomasz Hachaj
tomekhachaj@o2.pl

[1] Institute of Computer Science and Computer Methods, Pedagogical University of Krakow, 2 Podchorazych Ave, 30-084 Krakow, Poland

[2] Cryptography and Cognitive Informatics Research Group, AGH University of Science and Technology, 30 Mickiewicza Ave, 30-059 Krakow, Poland

Keywords Sport data analysis · Rehabilitation data analysis · Motion capture · Signal classification · Gesture description language

Introduction

Motion capture (MoCap) is a powerful technology with many possible applications. The dimensionality of output signal stream from MoCap system depends on number and type of sensors or tracked body joints in virtual skeleton that are used [1–4]. Mostly often each body joint has three or six degrees of freedom. Three are linear coordinates in Cartesian frame with versors (x, y, z) and other three are angles that define the orientation of the body segments. Those values are not used directly by the system but rather features values are calculated as derivatives of original data. The features selection and extraction methods are for example Gabor [5] or Haar filters [6]. Dimensionality reduction can be done with principal components analysis (PCA) [7] or other approaches [8]. The movement representation is often invariant under rigid transformation [8] and can be for example angular representation of the skeleton joints [9] where each pose is described using an angular representation of the skeleton joints.

Many methods have been yet proposed for human actions and movements evaluation and recognition. That type of analysis is important for calculation of biomechanics parameters of actions, for evaluation ones activates and lifestyle or during rehabilitation [10, 11]. Among proposed methods that can be used for signals and action recognition are approaches that are often used for signal identification and pattern recognition. The most popular are Hidden Markov Models (HMM) [10, 12, 13], support vector machines (SVM) [5, 9, 14], decision

forests [9, 15], Gaussian process dynamical models [16], K-means clustering [6], nearest neighbor classifier [17], Bayes classifier [18], dynamic Bayesian networks [19], syntactic method [6, 20, 21] and rule based methods – for example Gesture Description Language (GDL) [22–24]. GDL classifier uses a rule – based approach with memory stack. The memory stack holds the captured MoCap data frames, features and classes to which a sequence of MoCap frames are classified. GDL method uses specially designed scripts that hold the definition of features calculated from MoCap input stream. Those features are used to design rules that have if-else form and define the key frames of actions. Key frames are ordered in sequences. If the sequence of key frames appears in memory stack in a given time restriction the ongoing action is classified to a given class. This approach is somehow similar to HMM classifier.

The wide comparison of GDL methodology to other recognition system is discussed in other papers [25–27]. This comparison includes most important aspects like comparison of action description methodology, geometric interpretation of those descriptions, training algorithm and applications. The example results for various physical activates is presented in papers [22–26, 28–31] and includes common-life actions, gym exercises and Oyama and Shorin-Ryu karate techniques.

So far the GDL was used mainly for classification of MoCap data stream from multimedia devices (for example Microsoft Kinect) however we need to create a unified approach that would enable not only classification but also analysis of MoCap signals from high-end hardware. That type of devices is often used to support spot coaches in training optimization and physicians in rehabilitation process [10, 11, 32]. The requirements for this new approach is that it has to be capable to create complex features definitions (that could be for example used for kinematic analysis) and have to be fast enough for real-time data analysis (performance is very important aspect of every medical system [33]). The main novelty of this paper is presenting the adaptation of Gesture Description Language (GDL) methodology to sport and rehabilitation data analysis and classification. In the following sections we will present, evaluate and discuss proposition of such methodology.

Materials and methods

In this section we will present the novel adaptation of GDL methodology to analysis and classification of MoCap data for sport and rehabilitation applications. To do so we need to enhance the possibilities of features definition of scripting language that is inherent part of GDL classifier. We did it by replacing the old GDLs scripting language with Lua.

Lua application in GDL paradigm

Lua is a dynamically typed language that can be easily integrated with other computer languages and applications. Due its simplicity and easily extensions it is a popular scripting technology for other computer systems. Lua is used in scientific computing like linear algebra, neural networks, numeric optimization routines and many more [34–36]. Lua was also used in an aspect-oriented infrastructure to handle dynamic programming tasks [37]. Lua is also used in middleware design and development [38, 39] also in robotics and embedded systems [40–43]. This programming language is very popular tool for writing high-level scripts for other computer systems [44–46]. In paper [47] authors discuss what mechanisms Lua features to achieve its flexibility and how programmers use them for different paradigms.

The proposed Lua implementation is based on Lua 5.2 and JAVA hosting application. Lua functions are called by LuaJ library. GDL engine uses five classes and one script file Engine.lua with global variables and functions - see a class diagram presented in Fig. 1. A very basic script that can be used to detect situation when right hand is under head might looks as follow:

```
require "GDL2/Engine"
function ReturnConclusions(ActualSkeletonData, TimePeriod)
   PreReturnConclusions(ActualSkeletonData, TimePeriod)
   if Head(0).y > HandRight(0).y then
        HandUnderHead = true
   else
        HandUnderHead = false
   end
   PostReturnConclusions(st)
end
```

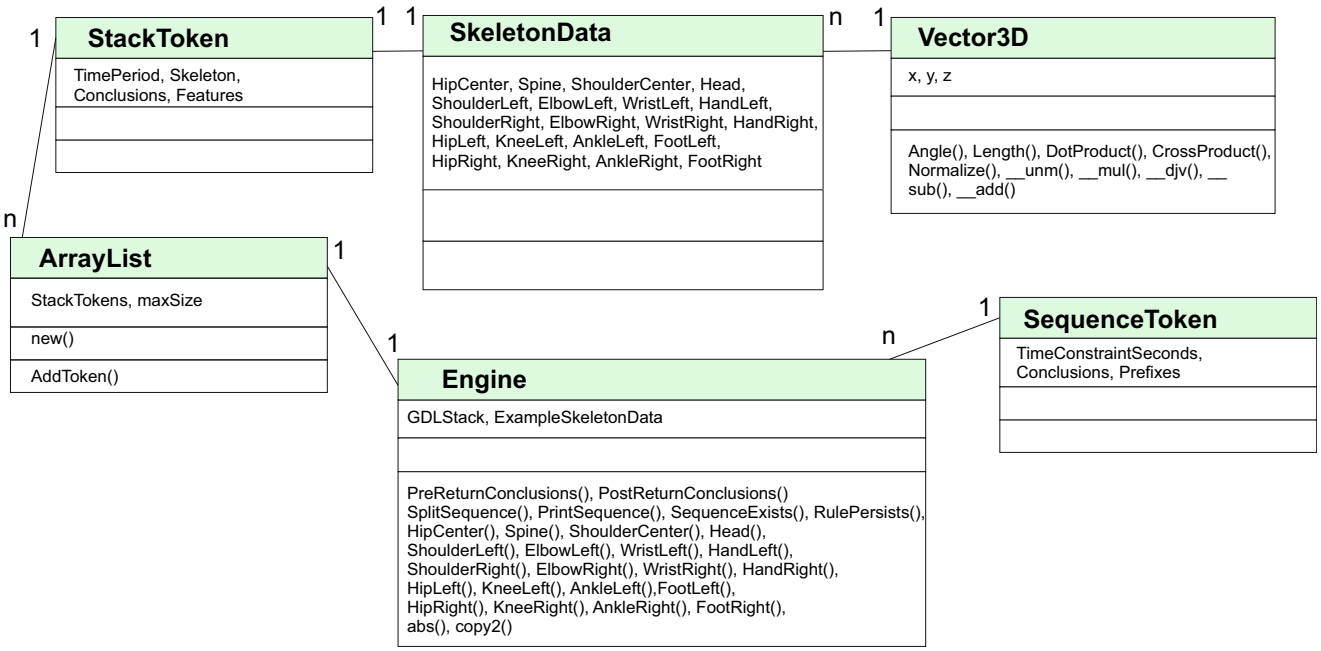

Fig. 1 This figure presents a class diagram for Lua implementation of GDL classifier

The ReturnConclusions function is called by hosting application to pass the tracking parameters. The proposed Lua – based framework can be easily adapted to anybody joints set simply by configuration of joints definition in SkeletonData class and functions in Engine.lua.

Features calculation

The adapted GDL classifier uses standard Lua syntax to define features. There are three types of features: logical, numeric and vector. The logical feature role is identical to conclusion from GDL specification and it is represented by 'boolean' data type in Lua. Only those logical values that equals 'true' are passed to hosting application. The numeric data type has a floating-point value and it is represented by 'number' data type in Lua. The vector data type has three floating-point values and is represented by user-defined class Vector3D. There are also definition of most important vectors operation like vector sum, multiplication by a number, dot product, cross product etc. that can be usable for kinematics or kinetics analysis [48] (the basic trigonometric functions like sinus are already supported by Lua).

In Fig. 2 we present vectors set that was used to generate features set that was used for recognition hiza-geri karate kick. We have defined six angle-based features:

$$\begin{cases} A_1 = \measuredangle(v_1, v_2) \\ A_2 = \measuredangle(v_3, v_2) \\ A_3 = \measuredangle(v_4, v_2) \\ A_4 = \measuredangle(v_1, v_5) \\ A_5 = \measuredangle(v_3, v_5) \\ A_6 = \measuredangle(v_4, v_5) \end{cases} \tag{1}$$

Where $A_1, \ldots, A_6$ are angles calculated between vectors $v_1, \ldots, v_5$ visualized in Fig. 2. Vectors are defined by tracked body joints.

The Lua implementation looks as follows:

```
A1 = angle(ShoulderRight(0) - ShoulderLeft(0), HipRight(0) - KneeRight(0))
A2 = angle(SpineShoulder(0) - SpineBase(0), HipRight(0) - KneeRight(0))
A3 = angle(CrossProduct(ShoulderRight(0) - ShoulderLeft(0),
       SpineShoulder(0) - SpineBase(0)), HipRight(0) - KneeRight(0))
A4 = angle(ShoulderRight(0) - ShoulderLeft(0), KneeRight(0) - AnkleRight(0))
A5 = angle(SpineShoulder(0) - SpineBase(0), KneeRight(0) - AnkleRight(0))
A6 = angle(CrossProduct(ShoulderRight(0) - ShoulderLeft(0),
       SpineShoulder(0) - SpineBase(0)), KneeRight(0) - AnkleRight(0))
```

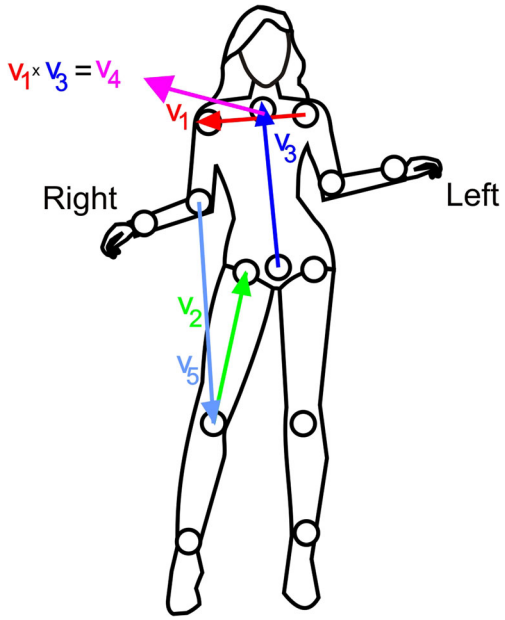

Fig. 2 This figure presents vectors set that was used to generate example features

Where angle is a function that finds angle between two vectors on the plane designated by those vectors.

In Fig. 3 we present 3D visualizations of important phases of selected karate actions we used in evaluation of our methodology.

In Fig. 4 we present plot of features values defined by (1) for a recording of single Hiza-Geri kick done with Kinect 2 depth camera. Above the plot are horizontal bars with color-coded information about key frames to which current frame was assigned by GDL classifier. Brown is the first key frame, yellow the second, cyan the third. Blue stands for lack of assignment (N – not assigned).

Pattern recognition with GDL approach

The recognition process of action pattern in adapted implementation is mainly the same as in [25]. The only difference is a reasoning module. We have found that in the previous implementations users seldom used some features of it and might even not be aware of its existence. Due to this fact hardly ever users design GDLs that reference to rule conclusion before it is defined in next rule. Also those types of constructions are not required in automatic training algorithm (R-GDL) about which we write below.

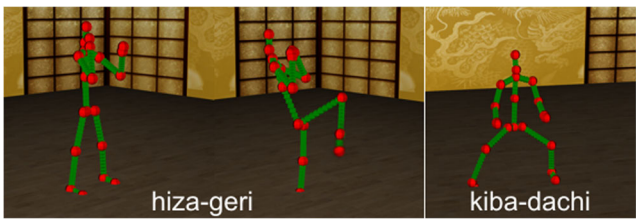

Fig. 3 This figure presents important phases of karate actions: Hiza-Geri kick and Kiba-Dachi stance. The Mo-Cap data is visualized in 3D virtual environment

The wearable body sensor enables to collect big data collections for which approaches know from other fields of big-data analysis can be applied [49, 50]. This property is also utilized in automatic training algorithm for GDL technology. While using GDL for a classification task the one of the most challenging aspect is designing of appropriate script that defines key frames of actions. To find those key frames automatically we often use Reverse-GDL (R-GDL) approach published in [25]. The R-GDL utilizes the fact that after transferring the original MoCap data to features space key frames can be detected with k-means clustering algorithm. This situation if visualized in Fig. 5 where data assigned to key frames is color coded with the same color pattern as in Fig. 4.

Results

In third section we will evaluate the average performance of our adapted methodology. The implementation can be found on official website of GDL technology [51]. We are mostly interested in how much time is required to calculate features and to classify the input dataset. We have taken into account time of transferring data from hosting application to GDL engine, Lua scripts execution time and transferring data from GDL to host application. After this whole cycle the application obtains data that can be directly used by it. We have used 20-joints data with gym exercises recordings acquired with Kinect version 1 (K1), the same as we used in [25] and 25-joints data with karate techniques recordings acquired with Kinect version 2 (K2) [26]. The Lua scripts in which actions were defined varied in number of features definition and number of GDL instructions calls. The most basic scripts were 10-features, 20-features, 30-features and 40-features sets that were defined both for K1 and K2. All features were angles defined by vectors calculated from neighboring joint. The other scripts were codes that define kiba-dachi stand (about 4 kB of code), and definition of 12 karate actions (about 33 Kb of code) [30], jumping jacks exercise (4 kB) and 9 gym exercises (40 kB) [25]. We have used actions recordings consisted of 100, 200, 500, 1000, 2000, 5000, 10,000 and 20,000 frames. Evaluation was repeated 20-times for each Lua script and recording. The proposed method was evaluated on standard PC equipped with Intel Core i7-4770 CPU 3.40 GHz, 8 GB of RAM with Windows 7 Home Premium 64 Bit. The results in Table 1 and Fig. 6 are averaged results plus – minus standard deviation.

Basing on the obtained data in Table 2 and Fig. 7 we have shown what is an average execution time of single MoCap frame calculation plus – minus standard deviation.

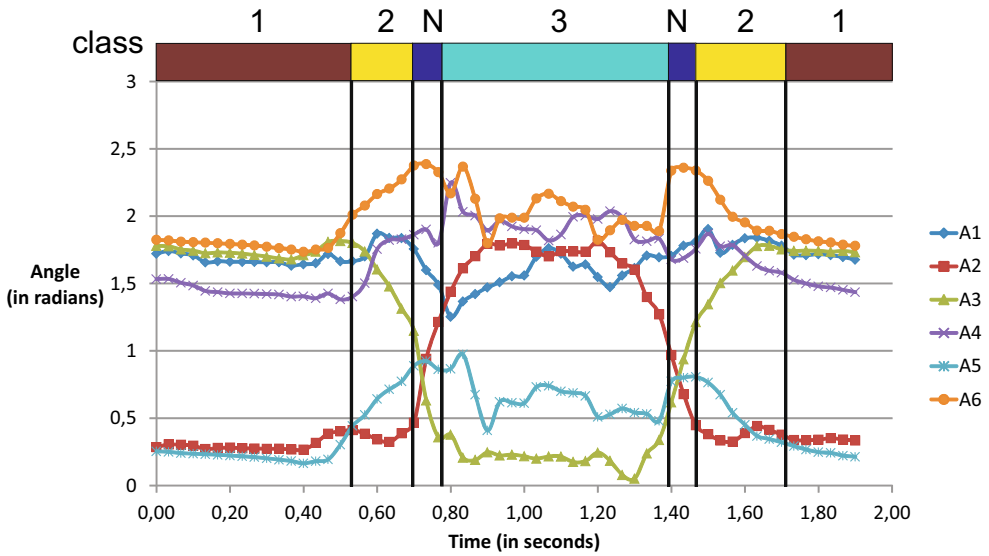

Fig. 4 This figure presents features time series generated for single Hiza-Geri kick recording. The *horizontal axis* represents time and the vertical axis the angle. Each time series stands for one of the feature from (1). On the top of the plot there are *color bars* that indicate to which GDL key frame the signal sample has been classified. Color codes are the same as in Fig. 5. Number 1, 2 and 3 are key frames numbers (there are totally three key frames in this particular Hiza-Geri definition). Symbol N represents the time sample in which signals have not been classified to any key frame

Discussion

The results presented in previous section show that processing time of Lua implementation of GDL methodology operates in fast and reliable way. As can be seen in Table 1 and Fig. 6 there is a nearly linear dependence between number of processed frames and time of processing. That proves that the method is stable and can operate without disturbance continuously. The test performed on features set for 10, 20, 30 and 40 features proves that there is no significant difference in processing time for 20 and 25 joints dataset. This is also quite natural that the larger the movement description is the more time is require to process the single MoCap frame. The execution time for 40-features K1 is 3.53 ± 0.16 milliseconds and

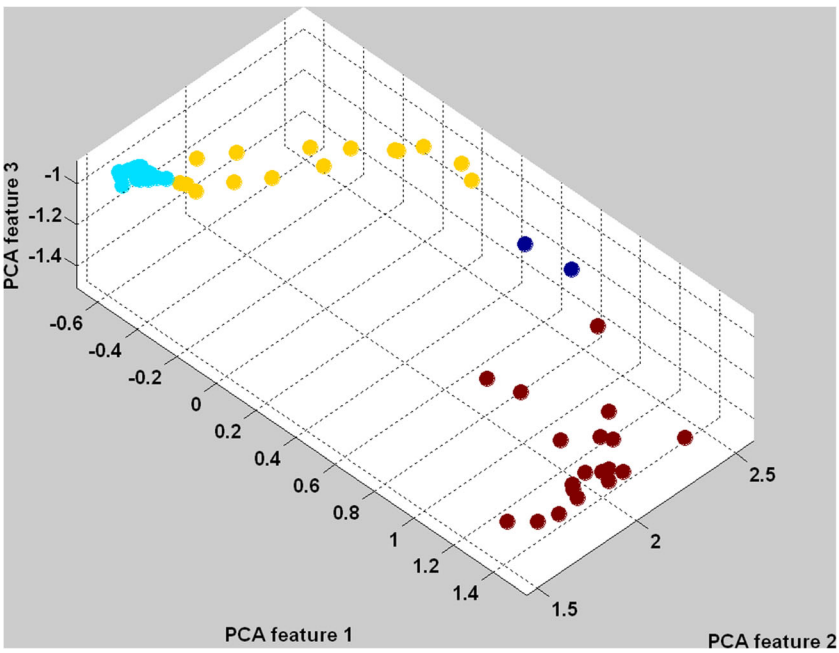

Fig. 5 This figure presents three-dimensional projection of six-dimensional feature space (1) using principal component analysis. Each point represents a single MoCap frame with color-coded GDL key frame

Table 1 This table presents averaged execution time (in milliseconds) plus-minus standard deviation of various Lua scripts that uses GDL implementation

	100	200	500	1000	2000	5000	10,000	20,000
kiba-dachi (K2)	171 ± 26	290 ± 35	754 ± 44	1683 ± 51	3062 ± 96	7824 ± 191	15,316 ± 278	31,249 ± 1218
karate (K2)	1173 ± 330	2225 ± 299	5913 ± 146	11,627 ± 358	23,114 ± 458	60,020 ± 788	116,892 ± 2052	231,344 ± 1046
10 features (K2)	125 ± 21	201 ± 22	505 ± 26	1123 ± 36	2064 ± 38	5116 ± 31	10,511 ± 377	20,566 ± 164
20 features (K2)	220 ± 21	366 ± 45	912 ± 43	2069 ± 56	3717 ± 57	9234 ± 55	18,677 ± 252	37,526 ± 915
30 features (K2)	301 ± 31	524 ± 65	1346 ± 69	2996 ± 83	5429 ± 83	13,550 ± 81	27,322 ± 374	54,419 ± 203
40 features (K2)	394 ± 35	778 ± 100	1977 ± 109	4010 ± 125	7999 ± 117	20,018 ± 140	40,615 ± 618	80,120 ± 500
jumping jacks (K1)	104 ± 16	166 ± 18	425 ± 25	1099 ± 149	1659 ± 21	4220 ± 73	8615 ± 409	16,684 ± 69
gym (K1)	722 ± 260	1251 ± 164	3710 ± 185	8806 ± 160	17,655 ± 221	43,832 ± 187	88,156 ± 646	175,715 ± 2004
10 features (K1)	121 ± 12	195 ± 22	498 ± 27	1148 ± 35	2024 ± 49	5087 ± 117	10,200 ± 288	20,211 ± 183
20 features (K1)	215 ± 25	359 ± 45	901 ± 50	2097 ± 63	3652 ± 52	9092 ± 38	18,439 ± 253	36,333 ± 38
30 features (K1)	309 ± 24	534 ± 67	1368 ± 65	3059 ± 100	5504 ± 85	13,744 ± 88	27,231 ± 500	55,012 ± 126
40 features (K1)	377 ± 31	671 ± 84	1710 ± 95	3828 ± 112	6921 ± 103	17,249 ± 115	34,723 ± 157	69,997 ± 448

Each row represents various features, action and actions groups that are evaluated for different number of motion capture frames (in columns)

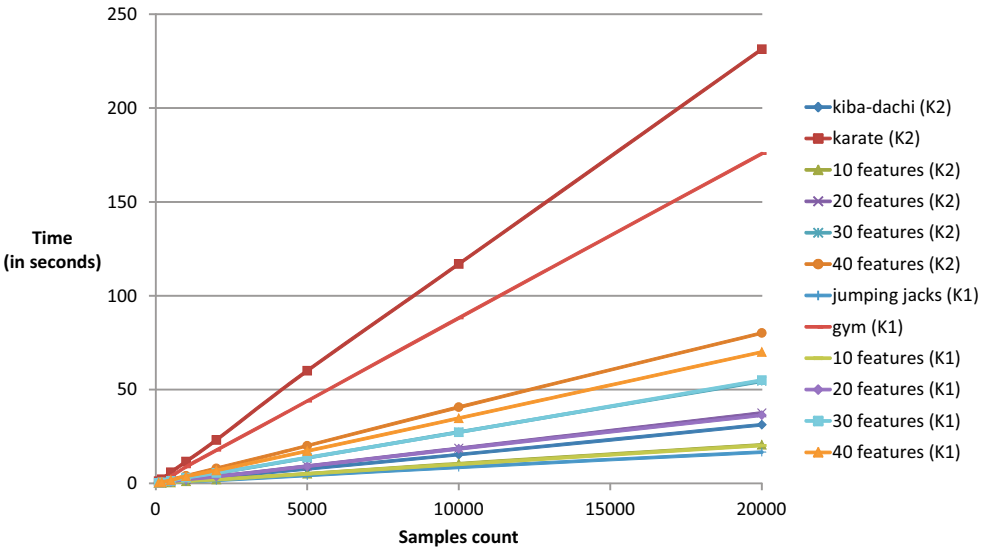

Fig. 6 This figure visualizes data from Table 1

Table 2 This table presents averaged execution time (in milliseconds) plus-minus standard deviation of various Lua scripts that uses GDL implementation for a single motion capture frame

Feature, action or action group name	Execution time (in milliseconds)
kiba-dachi (K2)	1.57 ± 0.08
karate (K2)	11.64 ± 0.24
10 features (K2)	1.06 ± 0.08
20 features (K2)	1.92 ± 0.13
30 features (K2)	2.77 ± 0.13
40 features (K2)	3.98 ± 0.05
jumping jacks (K1)	0.90 ± 0.10
gym (K1)	8.11 ± 0.94
10 features (K1)	1.05 ± 0.08
20 features (K1)	1.89 ± 0.13
30 features (K1)	2.82 ± 0.15
40 features (K1)	3.53 ± 0.16

for K2 is 3.98 ± 0.05 that means that it is possible to process MoCap dataset with frequency over 250 Hz which is sufficient for most up-to-date hardware of that type. The slowest processing was obtained for karate actions classification dataset that recognizes 12 different actions (11.64 ± 0.24 milliseconds per frame). That means that frequency of frame processing is about 85 Hz which is fast enough for signal classification task. Summing up the GDL methodology adapted to new functionalities satisfies needs of sport and rehabilitation data analysis and classification.

Conclusions

In this paper we showed that Lua language can be successfully used for adaptation of the GDL classifier to new scientific tasks.

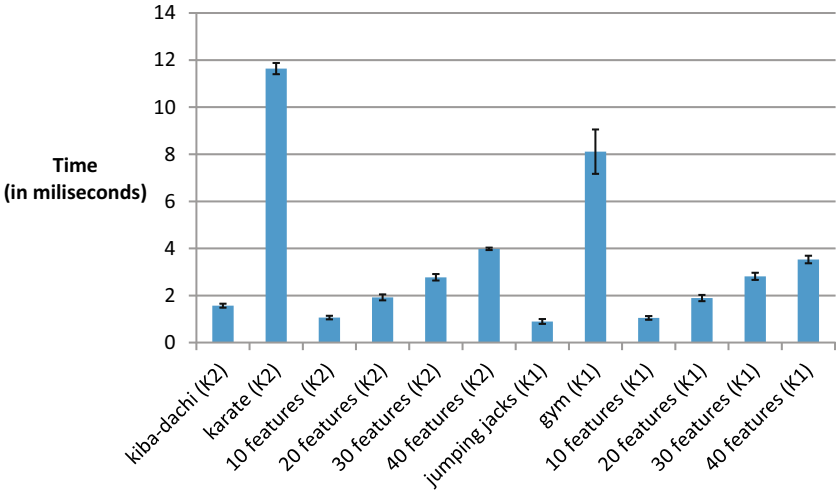

Fig. 7 This figure visualizes data from Table 2

The newly applied scripting language allows easily extension and integration of classifier with other software technologies and applications. As it was discussed in the previous section the obtained execution speed allows using the methodology in the real-time motion capture data processing where capturing frequency differs from 100 Hz to even 500 Hz depending on number of features or classes to be calculated and recognized. Due to this fact the proposed methodology can be used to the high-end motion capture system. We anticipate that using novel, efficient and effective method will highly help both sport trainers and physiotherapist in everyday tasks. For example pattern recognition and data mining methods can supply both sport and rehabilitation data evaluation. The proposed approach can be directly applied to MoCap data kinematics analysis (evaluation of motion without regard to the forces that cause that motion). However kinetics (the study of movements under the action of forces) will require additional data source beside MoCap for example ground reaction forces acquired by force plate or other force type collected with dynamometer. That additional data stream can be easily integrated with our methodology and the forces can be calculated without changing already established framework. However sole kinematics is sufficient for many applications in sport, medicine, physiotherapy and rehabilitation. Actions can be described using derivatives of displacement like velocity or acceleration. The ability to apply pattern recognition methods for GDL description can be utilized in virtual reality environment similarly to that described in [27, 52] and used for training or treatment.

Acknowledgments This work has been supported by the National Science Centre, Poland, under project number 2015/17/D/ST6/04051.

References

1. Artner, N. M., Ion, A., and Kropatsch, W. G., Multi-scale 2D tracking of articulated objects using hierarchical spring systems. Pattern Recognit.: 800–810. doi: 10.1016/j.patcog.2010.10.025, 2011.
2. Zhang, Q., Song, X., Shibasaki, R., and Zhao, H., Unsupervised skeleton extraction and motion capture from 3D deformable matching. *Neurocomputing* 100(16):170–182, 2013.
3. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A., Real-time human pose recognition in parts from single depth images, CVPR '11 Proceedings of the 2011 I.E. Conference on Computer Vision and Pattern Recognition, pp. 1297–1304, IEEE Computer Society Washington, DC, USA, 2011.
4. Schwarz, L. A., Mkhitaryan, A., Mateus, D., and Navab, N., Human skeleton tracking from depth data using geodesic distances and optical flow. *Image Vis. Comput.* 30:217–226, 2012.
5. Gupta, S., Jaafar, J., Fatimah, W., and Ahmad, W., Static hand gesture recognition using local gabor filter. *Proc. Eng.* 41:827–832, 2012.
6. Arulkarthick, V. J., and Sangeetha, D., Sign language recognition using K-means clustered haar-like, features and a stochastic context free grammar. Eur. J. Sci.
7. Taubert, N., Löffler, M., Ludolph, N., Christensen, A., Endres, D., and Giese, M. A., A virtual reality setup for controllable, stylized real-time interactions between humans and avatars with sparse gaussian process dynamical models. Proceedings of the ACM Symposium on Applied Perception, pp. 41–44, 2013.
8. Vieira, W. A., Lewiner, T., Schwartz, W. R., and Campos M. F. M., Distance matrices as invariant features for classifying MoCap data, Pattern Recognition (ICPR), 2012 21st International Conference on, pp. 2934–2937. IEEE, 2012.
9. Miranda, L., Vieira, T., Martinez, D., Lewiner, T., Vieira, A. W., and Campos, M. F. M., Online gesture recognition from pose kernel learning and decision forests. *Pattern Recogn. Lett.* 39(1):65–73, 2014.
10. Li, Z., Wei, Z., Yue, Y., Wang, H., Jia, W., Burke, L. E., Baranowski, T., and Sun, M., An adaptive hidden Markov model for activity recognition based on a wearable multi-sensor device. *J. Med. Syst.* 39(5):57, 2015. doi:10.1007/s10916-015-0239-x.
11. Rajanna, V., Vo, P., Barth, J., Mjelde, M., Grey, T., Oduola, C., and Hammond, T., KinoHaptics: An automated, wearable, Haptic assisted, physio-therapeutic system for post-surgery rehabilitation and self-care. *J. Med. Syst.* 40(3):60, 2016. doi:10.1007/s10916-015-0391-3.

12. Cholewa, M., and Głomb, P., Estimation of the number of states for gesture recognition with Hidden Markov Models based on the number of critical points in time sequence. *Pattern Recogn. Lett.* 34(5): 574–579, 2013.
13. Kang, J., Zhong, K., Qin, S., Wang, H., and Wright, D., Instant 3D design concept generation and visualization by real-time hand gesture recognition. *Comput. Ind.* 64(7):785–797, 2013.
14. López-Méndez, A., and Casas, J. R., Model-based recognition of human actions by trajectory matching in phase spaces. *Image Vis. Comput.* 30:808–816, 2012.
15. Zhu, F., Shao, L., and Lin, M., Multi-view action recognition using local similarity random forests and sensor fusion. *Pattern Recogn. Lett.* 34:20–24, 2013.
16. Gamage, N., Chow Kuang, Y., Akmeliawati, R., and Demidenko, S., Gaussian process dynamical models for hand gesture interpretation in sign language. *Pattern Recogn. Lett.* 32:2009–2014, 2011.
17. Glowacz, A., Diagnostics of synchronous motor based on analysis of acoustic signals with the use of line spectral frequencies and K-nearest neighbor classifier. *Arch. Acoust.* 39(2):189–194, 2014. doi: 10.2478/aoa-2014-0022.
18. Glowacz, A., Glowacz, A., and Glowacz, Z., Recognition of thermal images of direct current motor with application of area perimeter vector and Bayes classifier, measurement science review. 15(3): 119–126, ISSN (Online) 1335–8871. doi: 10.1515/msr-2015-0018, 2015.
19. Du, Y., Chen, F., Xu, W., and Zhang, W., Activity recognition through multi-scale motion detail analysis. *Neurocomputing* 71: 3561–3574, 2008.
20. Suma, E. A., Krum, D. M., Lange, B., Koenig, S., Rizzo, A., and Bolas, M., Adapting user interfaces for gestural interaction with the flexible action and articulated skeleton toolkit. *Comput. Graph.* 37(3):193–201, 2013.
21. Bickerstaffe, A., Lane, A., Meyer, B., and Marriott, K., Developing domain-specific gesture recognizers for smart diagram environments, graphics recognition. Recent advances and new opportunities. *Lect. Notes Comput. Sci* 5046:145–156, 2008.
22. Hachaj, T., and Ogiela, M. R., Rule-based approach to recognizing human body poses and gestures in real time. *Multimedia Systems* 20(1):81–99, 2014. doi:10.1007/s00530-013-0332-2.
23. Hachaj, T., and Ogiela, M. R., Computer karate trainer in tasks of personal and homeland security defense. In: Cuzzocrea, A., et al. (Eds.), CD-ARES 2013 Workshops, LNCS 8128, pp. 430–441, 2013.
24. Hachaj, T., Ogiela, M. R., and Piekarczyk, M., Real-time recognition of selected karate techniques using GDL approach, image processing and communications challenges 5. *Adv. Intell. Syst. Comput.* 233:99–106, 2014.
25. Hachaj, T., and Ogiela, M. R., Full body Movements recognition – unsupervised learning approach with heuristic R-GDL method. *Digital Signal Process.* 46:239–252, 2015. doi:10.1016/j.dsp.2015.07.004.
26. Hachaj, T., Ogiela, M. R., and Koptyra, K., Application of assistive computer vision methods to Oyama karate techniques recognition. *Symmetry* 7(4):1670–1698, 2015. doi:10.3390/sym7041670.
27. Hachaj, T., and Baraniewicz, D., Knowledge bricks - educational immersive reality environment. *Int. J. Inf. Manag.* 35:396–406, 2015. doi:10.1016/j.ijinfomgt.2015.01.006.
28. Hachaj, T., and Ogiela, M. R., Recognition of body movements patterns for immersive virtual reality system interface, 2014 Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 978-1-4799-4171-1/14, IEEE Computer Society Order Number E5391 ISBN-13: 978-1-4799-4171-1, pp. 290–294. doi 10.1109/3PGCIC.2014.79, 2014.
29. Hachaj, T., Ogiela, M. R., and Koptyra, K., Effectiveness comparison of Kinect and Kinect 2 for recognition of Oyama karate techniques, NBiS 2015 - The 18-th International Conference on Network-Based Information Systems (NBiS 2015), September 2–4, Taipei, Taiwan, pp. 332–337. doi 10.1109/NBiS.2015.51, ISBN: 978-1-4799-9942-2/15.
30. Hachaj, T., Ogiela, M. R., and Koptyra, K., Human actions modelling and recognition in low-dimensional feature space, BWCCA 2015, 10th International Conference on Broadband and Wireless Computing, Communication and Applications, November 4–6, 2015, Krakow, Poland, pp. 247–254. doi 10.1109/BWCCA.2015.15, 2015.
31. Hachaj, T., Ogiela, M. R., and Koptyra, K., Application of hidden Markov models and gesture description language classifiers to Oyama karate techniques recognition, Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015 9th International Conference on, 8–10 July 2015, Blumenau, pp. 160–165, ISBN 978-1-4799-8872-3. doi: 10.1109/IMIS.2015.26, 2015.
32. Palacios-Navarro, G., García-Magariño, I., and Ramos-Lorente, P., A Kinect-based system for lower limb rehabilitation in Parkinson's disease patients: a pilot study. *J. Med. Syst.* 39(9):103, 2015. doi:10.1007/s10916-015-0289-0.
33. De la Torre-Díez, I., Antón-Rodríguez, M., Díaz-Pernas, F. J., and Perozo-Rondón, F. J., Comparison of response times of a mobile-web EHRs system using PHP and JSP languages. *J. Med. Syst.* 36(6):3945–3953, 2012. doi:10.1007/s10916-012-9866-7.
34. Hauser, J. R., *Numerical methods for nonlinear engineering models*. Springer, Netherlands, 2009. doi:10.1007/978-1-4020-9920-5.
35. Torch official website (acces datee 25-02-2016) http://torch.ch/.
36. Collobert, R., Kavukcuoglu, K., and Farabet, C., Implementing neural networks efficiently, neural networks: tricks of the trade, Volume 7700 of the series Lecture Notes in Computer Science, pp. 537–557, DOI: 10.1007/978-3-642-35289-8_28.
37. Cacho, N., Batista, T., and Fernandes, F., A Lua-based AOP infrastructure. *J. Braz. Comput. Soc.* 11(3):7–20, 2005.
38. Maia, R., Cerqueira, R., Sieckenius de Souza, C., and Guisasola-Gorham, T., A qualitative human-centric evaluation of flexibility in middleware implementations. *Empir. Softw. Eng.* 17(3):166–199, 2012.
39. Soares, L. F. G., Rodrigues, R. F., Cerqueira, R., and Barbosa, S. D. J., Variable and state handling in NCL. *Multimed. Tools Appl.* 50(3): 465–489, 2010.
40. Niemüller, T., Ferrein, A., and Lakemeyer, G., A Lua-based behavior engine for controlling the humanoid robot nao, RoboCup 2009: Robot Soccer World Cup XIII, Volume 5949 of the series Lecture Notes in Computer Science, pp. 240–251. DOI: 10.1007/978-3-642-11876-0_21.
41. Codd-Downey, R., Jenkin, M., Ansell, M., Ng, H. -K., and Jasiobedzki, P., Simulating the C2SM 'Fast' robot, simulation, modeling, and programming for autonomous robots, Volume 6472 of the series Lecture Notes in Computer Science, pp. 26–37. DOI: 10.1007/978-3-642-17319-6_6.
42. Freese, M., Singh, S., Ozaki, F., and Matsuhira, N., Virtual robot experimentation platform V-REP: a versatile 3D robot simulator, simulation, modeling, and programming for autonomous robots, Volume 6472 of the series Lecture Notes in Computer Science, pp. 51–62. DOI: 10.1007/978-3-642-17319-6_8.
43. Ferrein, A., and Steinbauer, G., On the way to high-level programming for resource-limited embedded systems with Golog, Simulation, Modeling, and Programming for Autonomous Robots, Volume 6472 of the series Lecture Notes in Computer Science, pp. 229–240. doi: 10.1007/978-3-642-17319-6_23.
44. Emmerich, P., Beginning Lua with world of warcraft addons. Apress. doi: 10.1007/978-1-4302-2372-6, 2009.
45. Jordan, L., and Greyling, P., Practical android projects. Apress. doi: 10.1007/978-1-4302-3244-5, 2011.

46. Smith, W., and Wakefield, G., Computational audiovisual composition using Lua, Transdisciplinary Digital Art. Sound, Vision and the New Screen, Volume 7 of the series Communications in Computer and Information Science, pp. 213–228. doi: 10.1007/978-3-540-79486-8_19, 2008.
47. Ierusalimschy, R., Programming with multiple paradigms in lua, functional and constraint logic programming, Volume 5979 of the series Lecture Notes in Computer Science, pp. 1–12. doi: 10.1007/978-3-642-11999-6_1.
48. Karduna, A. R., Introduction to biomechanical analysis. In: Oatis, C. A., (Ed.), *Kinesiology: The Mechanics And Pathomechanics Of Human Movement*. Published by Lippincott Williams & Wilkins (2004-06-01) ISBN 10: 0781755131 / ISBN 13: 9780781755139.
49. Lin, C., Song, Z., Song, H., Zhou, Y., Wang, Y., and Wu, G., Differential privacy preserving in big data analytics for connected health. *J. Med. Syst.* 40(4):97, 2016. doi:10.1007/s10916-016-0446-0.
50. Ullah, S., Higgins, H., Braem, B., Latre, B., Blondia, C., Moerman, I., Saleem, S., Rahman, Z., and Kwak, K. S., A comprehensive survey of wireless body area networks: on PHY, MAC, and network layers solutions. *J. Med. Syst.* 36(3):1065–1094, 2012. doi:10.1007/s10916-010-9571-3.
51. Official website of GDL technology (acces datee 25-02-2016) http://gdl.org.pl/.
52. Hachaj, T., and Ogiela, M. R., Visualization of perfusion abnormalities with GPU-based volume rendering. *Comput. Graph.* 36(3):163–169, 2012. doi:10.1016/j.cag.2012.01.002.