Computer Karate Trainer in Tasks of Personal and Homeland Security Defense

Tomasz Hachaj[1] and Marek R. Ogiela[2]

[1] Pedagogical University of Krakow, 2 Podchorazych Ave, 30-084 Krakow, Poland
tomekhachaj@o2.pl
[2] AGH University of Science and Technology, 30 Mickiewicza Ave, 30-059 Krakow, Poland
mogiela@agh.edu.pl

Abstract. In this paper will be presented a new possibility of using GDL (Gesture Description Language) approach for recognition of basic combat techniques from martial arts. The GDL approach allows not only to analyze the several Shorin-Ryu Karate techniques but also to support the training and teaching activities of such arts. Moreover the GDL allow performing the human behavioral analysis, which may be important for recognition of dangerous situations while ensuring the homeland security.

Keywords: GDL, gesture description language, semantic classifier, gestures recognition, karate, self-defense**.**

1 Introduction

In case of attack of foe who has more physical strength, proper self-defense training might save the life and health of potential victim. The knowledge of defense martial arts is important aspect of personal and social security. In our opinion easy and common access to cheap self-defense courses would highly increase dexterity, confidence and composure in crisis situations of large society group. This is how popularization of personal training program can affect the homeland security. Some aspects of self-defense training require physical contact with human trainer however some exercises can be practiced alone. Those are arduous repetitions of movement sequences that have to be "remembered by body" to be quickly and subconsciously performed if there is only time for rapid reaction. We believe that those exercises can be overseen by computer application which will give advices and motivation to the adept. What is more that kind of application can be run whenever user wants to train. The overseeing computer programs of that type have to recognize specialized movement sequences – gestures of the user. The problem of computer gestures recognition has long history and there are many approaches how to deal with this task. The most popular are:

- Statistical methods [1],
- Neural networks and fuzzy sets [2],
- Optimal path finding [3],
- Semantic methods and finite state machines [4], [5].

A. Cuzzocrea et al. (Eds.): CD-ARES 2013 Workshops, LNCS 8128, pp. 430–441, 2013.

Lately appearance of new relatively cheap multimedia hardware enabled to introduce full body tracking and gestures recognition technology to personal computers and gaming consoles. For example a Kinect controller captures depth data stream which is then processed [6] in order to detect three-dimensional coordinates of 20 body joints (so called skeleton data). This approach has many important advantages:

- It does not require special markers to be placed over tracked body,
- It track skeleton relatively fast (with approximate frequency of 30 Hz),
- It has been proven that it can provide input data applicable for vast range of pattern recognition methods [7], [8].
- The multimedia controller is much cheaper than motion-capture hardware.

It was only matter of time while this technology has been applied to more sophisticated gestures that those that appears in computer games. Work [8] introduces a method for real-time gesture recognition from a noisy skeleton stream, such as the ones extracted from Kinect depth sensors. Each pose is described using a tailored angular representation of the skeleton joints. Those descriptors serve to identify key poses through a multi-class classifier derived from Support Vector learning machines. The gesture is labeled on-the-fly from the key pose sequence through a decision forest that naturally performs the gesture time warping and avoids the requirement for an initial or neutral pose.

The computer – aided training and rehabilitation was a concept of many previously published papers and implemented systems. For many years the application of gestures recognition methods were limited because of specialized motion – capture hardware requirements [9]. Lately the Kinect technology was used in number of sports video game like “My Self Defence Coach”, “UFC Personal Trainer” or “Nike+ Kinect Training” [10] however those applications are mainly done for entertainment and should not be treated as reliable computer coaching programs. One of the first well described approach of applying Kinect in personal training systems in presented in [7]. This work aims at automatically recognizing sequences of complex Karate movements and giving a measure of the quality of the movements performed. The proposed system is constituted by four different modules: skeleton representation, pose classification, temporal alignment, and scoring. The proposed system is tested on a set of different punch, kick and defense karate moves executed starting from the simplest case, i.e. fixed static stances up to sequences in which the starting stances is different from the ending one. The skeleton is represented by chosen 14 angles designated by vectors defined by selected skeleton joints. Each gesture is split into key-poses. The classification of key poses is done with a multi-class Support Vector Machine (SVM), which recognizes key poses with a one-versus-all approach. The temporal alignment of sequences is done with classic Dynamic Time Warping approach. The system proposed in that paper has however two very important limitations caused by hardware and applied methodology. It has to be remembered that depth data captured by the Kinect sensor can be visualized as relief – like plate. If particular part of the body is covered by one another it exact position cannot be accurately measured. If this situation happens the tracking software computes the position of “invisible joint” estimating its position basing on known position of neighboring joints. This approach

however might be highly insufficient in some real-life situations. The second limitation is pattern recognition method itself. Training of SVM (and other popular classifiers) requires huge data set to obtain high effectiveness. This might be a serious obstacle which limits this methodology only for research centers that manage to gather sufficiently big training dataset. However, it should be remembered that each self-defense trainer exactly knows how the particular movements should be performed and what are the key-poses that have to be present in gestures sequence. In this paper we presents our novel method GDL (Gesture Description Language) for overcoming those hardware and methodology limitations and results of the initial tests of our approach. The GDL allows not only to analyze the several Shorin-Ryu Karate techniques but also to support the training and teaching activities of such arts. Moreover the GDL allow performing the human behavioral analysis, which may be important for recognition of dangerous treats while ensuring the homeland security.

2 Methodology

In this paragraph we will discuss our approach for calibration of multi - Kinect environment that can use standard tracking libraries for skeleton segmentation. In the second part of this paragraph the karate techniques recognition approach will be presented.

2.1 Overcoming Hardware Limitation

In order to acquire real three dimensional skeleton of observed user (not only as relief – like plate projection) more than one capture device has to be used. The most intuitive hardware configuration is presented in Figure 1 – left. Three devices are positioned around object of observation on the vertices of equilateral triangle. This configuration however requires new depth stream processing algorithm. Because of that we propose another hardware configuration (Figure 1 – right). It also uses three sensors, the center one and two additional rotated around y (vertical) axis at the angle of $\frac{\pi}{4}$ and $-\frac{\pi}{4}$ to center one relatively. With this Kinects positioning it is still possible to use standard tracking software. As we have proved in paragraph 3 of this paper filming the same scene from different angles enables to acquire more tracking information. Right parts of the body are better (more efficiently) tracked by right Kinect while left parts are more efficiently tracked by left one. That is because if for example user is performing Mae-geri kick with his right leg, in the end phase of movement the right knee is in the same vertical position as right foot and hip. It cannot be reliably tracked by center (the one that is in front of user) sensor because foot cover the position of two other body joints. In the same time those three joints are not overlapping from perspective of right sensor. The tracking software supplies us with three

dimensional coordinates of body joints with information if particular joint position is obtained from direct tracking or it was estimated because that one is not visible. Knowing if joint tracked from central Kinect has exact coordinates or not we can use the coordinates from left or right additional device to make the overall skeleton position more reliable.

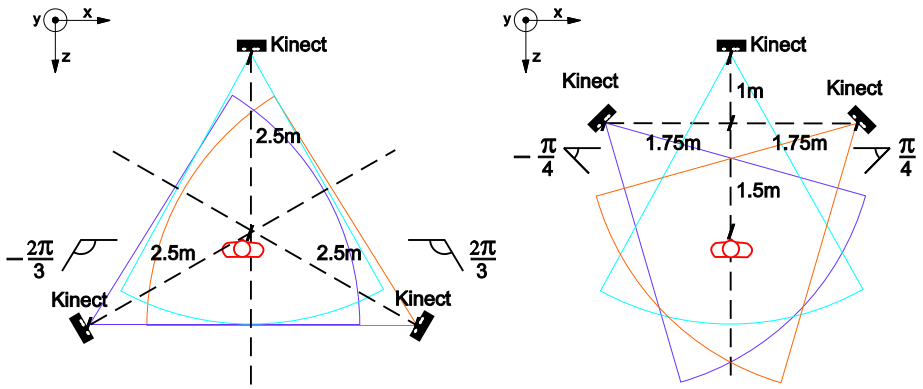

Fig. 1. Example multi Kinect setup, left – three devices are situated around the user, right – three devices are situated in front of user. Detailed description is in text.

Each Kinect measures distance to observed point in its own right-handed Cartesian frame situated relatively to sensor orientation. Because of that same point V has different coordinates $\bar{v}' = [x', y', z', 1]$ and $\bar{v} = [x, y, z, 1]$ relatively to each pair of devices. Our task now is to map all of those points to the same coordinate system. In order to do this we have to find linear transform that is represented by following matrix:

$$\bar{v}' \cdot \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix} = \bar{v} \tag{1}$$

Where a_{ij} coordinates represents the rotation and t_k translation. To compute unknown values we have to know coordinates of points in both Cartesian frames. Let us assume that a Cartesian frame that represents orientation of each Kinect was translated and rotated around y (vertical) axis relatively to each other frame. That means there are four degrees of freedom (three for translation, one for rotation). Knowing that the linear transformation that maps coordinates of a point represented by vector $\bar{v}'$ in one coordinate system to coordinates $\bar{v}$ in another one has form of following matrix:

$$\overline{v'} \cdot \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\beta) & 0 & \cos(\beta) & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix} = \overline{v} \tag{2}$$

In order to find unknown matrix coefficients following linear system has to be solved:

$$\begin{bmatrix} x_1' & z_1' & 1 & 0 \\ z_1' & -x_1' & 0 & 1 \\ x_2' & z_2' & 1 & 0 \\ z_2' & -x_2' & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) \\ \sin(\beta) \\ t_x \\ t_z \end{bmatrix} = \begin{bmatrix} x_1 \\ z_1 \\ x_2 \\ z_2 \end{bmatrix} \tag{3}$$

$$y' + t_y = y_1$$

Where $\overline{v_1} = [x_1, y_1, z_1, 1]$, $\overline{v_2} = [x_2, y_2, z_2, 1]$ are points which coordinates are known in both frames. That matrix has to be solved both for center-left Kinects set and center-right Kinects set.

2.2 GDL Classifier in Task of Shorin-Ryu Karate Recognition

GDL is a semantic classifier that uses syntactic description of gestures to be detected. The preliminary description of GDL architecture has been presented in [11]. In those papers we discussed application of this methodology to recognition basic common life gestures (like waving or clapping). However it is also possible to apply GDL to sophisticated and complex movement sequences like those from Shorin-Ryu Karate. The schema of gestures recognition pipeline with GDL classifier is presented in Figure 2. Movement is separated into key frames. Each key frame is represented by a rule in GDL script that has a conclusion. If a rule is satisfied for actual set of body joints positions (GDL uses forward chaining reasoning schema) its conclusion is memorized. It is possible to check with GDL script if some conclusion was satisfied in given time period. With this mechanism it is possible to generate chains of key frames, which create gestures. Because of space limitation we cannot present detailed description of all possible GDL script productions. Instead of that in Appendix we will show GDL script listings of all tested Karate techniques. The GDL syntax is very intuitive so we believe that short comments in source code will be sufficient for understanding.

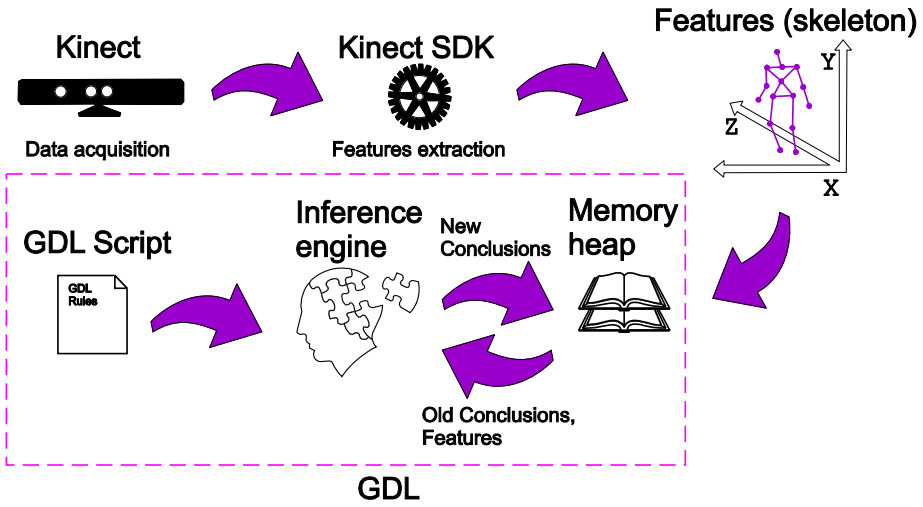

Fig. 2. The schema of gestures recognition pipeline with GDL classifier. The input data is recorded with depth sensor (for example Kinect). Then data is processed with appropriate libraries to extract features - skeleton. The skeleton is an input to GDL classifier. In GDL movement is described by set of rules consisted in GDL script file (GDL script is context – free grammar). Inference engine performs forward chaining reasoning on those rules. The conclusions of satisfied rules are stored in memory heap together with actual and previously captured skeletons and previously satisfied conclusions.

3 Results

In order to check possibilities of using GDL approach for recognition of basic combat techniques from martial arts, together with black belt instructor (3 dan) of Shorin-Ryu Karate we have created GDL script descriptions of one stationary position (Moto-dachi), one block (Age-uke) and one kick (Mae-geri). Than we recorded the instructor performing those techniques with system presented in Figure 2. Tables 1-2 summarize the classification results of our experiment. The description in first column is the actual technique (or group of techniques) that is present in particular recording. Each technique (or group of techniques) was repeated 50 times. Symbol + means that particular recording consisted of more than one technique. Description in first row is classification results. Last row sums up percentage of correct classifications of particular technique. Summing up, we had 150 recordings of karate techniques. In this experiment we check if it is possible to recognize the considered techniques and if multi sensor system setup increases overall recognition rate.

Table 1. The classification results of our experiment. Data was captured with single Kinect device (central one). Number in each cell is how many recordings belong to the certain class.

	Moto-dachi	Age-uke	Mae-geri	Not classified
Moto-dachi	50	0	0	0
Age-uke + Moto-dachi	49	21	0	1+29=30
Mae-geri	0	0	1	49
%	99.0%	42.0%	2.0%	39.5%

Table 2. The classification results of our experiment. Data was captured with three Kinect devices situated as shown in Figure 1 on the right. Number in each cell is how many recordings belong to the certain class.

	Moto-dachi	Age-uke	Mae-geri	Not classified
Moto-dachi	50	0	0	0
Age-uke + Moto-dachi	49	42	0	1+8=9
Mae-geri	0	0	32	18
%	99.0%	84.0%	64.0%	13.5%

Figure 3 graphically compares results presented in Table 1 and Table 2.

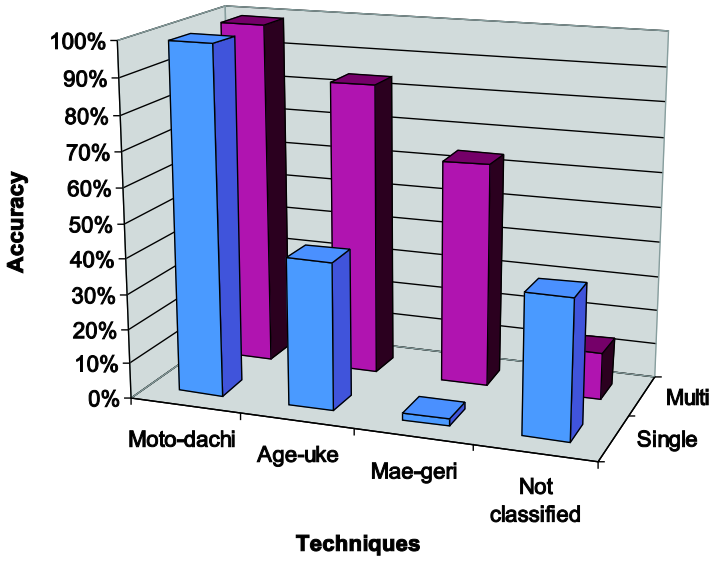

Fig. 3. Plot that graphically compares results presented in Table 1 and Table 2

4 Discussion and Conclusion

As has been shown in our experiment it is possible to overcome limitation of single capturing device by adding more devices of the same type that gather user data at different angles. What is more our approach can be used with standard tracking libraries. Our experiment has also shown that integration of tracking data acquired by several Kinect devices with standard software increases the effectiveness of GDL classifier. This is due the fact that additional sensors that are situated at different angles than central one are capable of tracking body joints that in some situations might be covered by different body parts. This condition is especially visible in case of non-static Karate techniques: Age-uke and Mae-geri. After integrating skeleton data from three sensors as it was presented in paragraph 2 the recognition rate of Age-uke was increased by 42% and Mae-geri by 62%. All recognition errors were caused by inaccurate tracking of users' body joints. We anticipate that we can minimize "not classified" error rate even more by applying device setup that was presented in Figure 1 – left. This sensor positioning will supply the system with "real" three dimensional measurements however the new tacking algorithm has to be generated. The creation of this new algorithm will be our goal for the future research.

Fig. 4. The set of photographs taken during test session of our training system. In top left and top right photo we see hardware set up. It is consisted of three PC computers connected together in local network. Each of them runs single instance of application that implements gesture recognition pipeline from Figure 2. The recordings are synchronized by synchronization server that sends timestamps to each instance by UDP protocol. In bottom left and right photo we can see system performing real-time recognition of defined Karate techniques.

We have created the prototype of our training system which was used to gather data presented in result section and to our novel methodology validation. The photographs taken during this test session are presented in Figure 4. The implementation of

system is capable of real time recognition of defined rules. The GDL approach allows not only to analyze the several Shorin-Ryu Karate techniques but also to support the training and teaching activities of such arts. In present form our approach enables karate techniques recognition that indicate to user if he or she made the gesture "enough similar to the pattern" to be classified. The proposed approach has to be expanded by additional module that would give the user the supporting information how well (how similarly to pattern) he made his gesture. In paper [7] authors proposed the scoring methodology to give as output a score representative of the effectiveness and quality of the move performed independently of the move length. The final score is obtained by regression among human judgments and the normalized dynamic time warping distances obtained by each technique. The model fitted is a 5-parameters logistic. This approach can easily be adapted to our solution. In the future we will generate another scoring methodology to make scoring dependent only to similarity of performed movement to GDL script description.
Our future goal will be development of GDL script for recognition of complete set of most popular karate techniques. The completed classifier will be than utilized in self-training multimedia application. We also plan to expand the GDL script syntactic [12] to enable creation rules that describes behavior of more than one tracked user at the same time. With this simple extension GDL will allow to perform the human behavioral and human interaction analysis, which may be important for recognition of dangerous situations while ensuring the homeland security.

Acknowledgments. We kindly acknowledge the support of this study by a Pedagogical University of Krakow Statutory Research Grant.

References

1. Vinayak, S., Murugappan, H.R., Liu, K., Vinayak, Murugappan, S., Liu, H.R., Ramani, K.: Shape-It-Up: Hand gesture based creative expression of 3D shapes using intelligent generalized cylinders. Computer-Aided Design 45, 277–287 (2013)
2. Elakkiya, R., Selvamai, K., Velumadhava Rao, R., Kannan, A.: Fuzzy Hand Gesture Recognition Based Human Computer Interface Intelligent System. UACEE International Journal of Advances in Computer Networks and its Security 2(1), 29–33
3. Augsburg University (2011), Full Body Interaction Framework: `http://hcm-lab.de/fubi.html`
4. Arulkarthick, V.J., Sangeetha, D., Umamaheswari, S.: Sign Language Recognition using K-Means Clustered Haar-Like Features and a Stochastic Context Free Grammar. European Journal of Scientific Research 78(1), 74–84 (2012) ISSN 1450-216X
5. Yeasin, M., Chaudhuri, S.: Visual understanding of dynamic hand gestures. Pattern Recognition 33, 1805–1817 (2000)
6. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR, vol. 3 (2011)
7. Bianco, S., Tisato, F.: Karate moves recognition from skeletal motion. In: Proc. SPIE 8650, Three-Dimensional Image Processing (3DIP) and Applications 2013, 86500K (March 12, 2013), doi:10.1117/12.2006229

8. Miranda, L., Vieira, T., Martinez, D., Lewiner, T., Vieira, A.W., Campos, M.F.M.: Real-time gesture recognition from depth data through key poses learning and decision forests. In: Proceedings of XXV SIBGRAPI Conference on Graphics, Patterns and Images (2012)
9. Mirabella, O., Raucea, A., Fisichella, F., Gentile, L.: A motion capture system for sport training and rehabilitation. In: 2011 4th International Conference on Human System Interactions (HSI), pp. 52–59 (2011), doi:10.1109/HSI.2011.5937342
10. List of games that utilize Kinect sensor, `http://www.xbox.com/pl-PL/Kinect/Games`
11. Hachaj, T., Ogiela, M.R.: Semantic Description and Recognition of Human Body Poses and Movement Sequences with Gesture Description Language. In: Kim, T.-h., Kang, J.-J., Grosky, W.I., Arslan, T., Pissinou, N. (eds.) MulGraB, BSBT and IUrC 2012. CCIS, vol. 353, pp. 1–8. Springer, Heidelberg (2012)
12. Ogiela, M.R., Piekarczyk, M.: Random graph languages for distorted and ambiguous patterns: single layer model. In: Proceedings of the Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS 2012), Palermo, Italy, July 4-6, pp. 108–113 (2012)

5 Appendix – GDL Script Bases and Examples

Skeleton is consisted of body joints. There are following possible names of body joints:

```
HipCenter, Spine, ShoulderCenter, Head, ShoulderLeft, El-
bowLeft, WristLeft, HandLeft, ShoulderRight, ElbowRight,
WristRight, HandRight, HipLeft, KneeLeft, AnkleLeft,
FootLeft,  HipRight, KneeRight, AnkleRight, FootRight.
```

Each rule starts with RULE keyword, and ends with conclusion name which is preceded by THEN keyword.

Syntax:

```
HipLeft.xyz[0]
```

means that you take three dimensional coordinate of joint HipLeft from the top of memory heap (actual joint position).

Synatx:

```
HipLeft.y[0]
```

means that you take only y-coordinate (vertical). The coordinate frame is right – handed as shown in Figure 1.

```
KneeRight.a[0]
```

is an angle between vectors defined by joint KneeRight and two neighboring joints (HipRight and AnkleRight), the vertex of the angle is in joint KneeRight.

```
angle(HipLeft.xyz[0] - KneeLeft.xyz[0],  HipRight.xyz[0]
- KneeRight.xyz[0])
```

is angle between two vectors defined by joints coordinates.
ABS is absolute value of real number.
DISTANCE is an Euclidean distance between vectors.
| is logical "or" operator, & is logical "and" operator.

```
sequenceexists("[MaeMiddleRight,1][MaeStart,1]")
```

is true if conclusion MaeMiddleRight has appeared in memory heap not longer than 1 second ago (this is "1" digit after comma) and MaeStart has appeared in memory heap not longer 1 second ago while MaeMiddleRight has appeared in memory heap.

The following listing is GDL script we defined together with black belt instructor (3 dan) of Shorin-Ryu Karate. Because of space limitation and intuitiveness of description we leave the detailed analysis to the reader.

```
//////////////////
//Moto-dachi
//////////////////
RULE angle(HipLeft.xyz[0] - KneeLeft.xyz[0],  Hi-
pRight.xyz[0] - KneeRight.xyz[0]) > 5
& angle(HipLeft.xyz[0] - KneeLeft.xyz[0],  Hi-
pRight.xyz[0] - KneeRight.xyz[0]) < 45
THEN MotoLegsA
RULE HipRight.z[0] < HipLeft.z[0] & KneeRight.z[0] <
KneeLeft.z[0]
& AnkleRight.z[0] < AnkleLeft.z[0]
THEN MotoLegsZRight //right leg is in front of body
RULE HipRight.z[0] > HipLeft.z[0] & KneeRight.z[0] >
KneeLeft.z[0]
& AnkleRight.z[0] > AnkleLeft.z[0]
THEN MotoLegsZLeft //left leg is in fornt of body
RULE ABS(AnkleRight.z[0] - AnkleLeft.z[0]) >
ABS(DISTANCE(HipRight.xyz[0], HipLeft.xyz[0]))
THEN MotoStepFront
RULE KneeRight.a[0] > 150 & KneeLeft.a[0] > 150
THEN MotoKnee
RULE  (MotoStepFront & MotoLegsA & MotoKnee)
& (MotoLegsZRight | MotoLegsZLeft) & StandStill
THEN Moto-dachi

//////////////////
//Age-uke
//////////////////
RULE WristRight.y[0] < WristLeft.y[0] THEN AgeUkeRStart
```

```
RULE WristRight.y[0] > WristLeft.y[0] THEN AgeUkeLStart
RULE ElbowRight.a[0] > 80 & Distance(WristRight.xyz[0],
HipRight.xyz[0]) < 200 THEN AgeUkeRightHand
RULE ElbowLeft.a[0] > 80 & Distance(WristLeft.xyz[0],
HipLeft.xyz[0]) < 200 THEN AgeUkeLeftHand
RULE ABS(WristLeft.y[0] - Head.y[0]) < 100 &
ABS(WristLeft.x[0] - Head.x[0]) < 100 & ElbowLeft.a[0] >
90 & ElbowLeft.a[0] < 150
THEN AgeUkeRightHandLeftHandBlock
RULE ABS(WristRight.y[0] - Head.y[0]) < 100 &
ABS(WristRight.x[0] - Head.x[0]) < 100 & ElbowRight.a[0]
> 90 & ElbowRight.a[0] < 150
THEN AgeUkeLeftHandRightHandBlock
RULE (AgeUkeLeftHand & AgeUkeLeftHandRightHandBlock)
THEN AgeUkeRStop
RULE (AgeUkeRightHand & AgeUkeRightHandLeftHandBlock)
THEN AgeUkeLStop
Rule (AgeUkeRStop & sequenceexists("[AgeUkeRStart,1]"))
| (AgeUkeLStop & sequenceexists("[AgeUkeLStart,1]")) &
StandStill
THEN Age-uke

//////////////////
//Mae-geri
//////////////////
RULE ABS(AnkleRight.y[0] - AnkleLeft.y[0]) < 50
THEN MaeStart
RULE (HipRight.y[0] - KneeRight.y[0]) < 100 &
ABS(KneeRight.a[0] - 90) < 30
THEN MaeMiddleRight
RULE (HipRight.y[0] - KneeRight.y[0]) < 200 & Knee-
Right.a[0] > 150
THEN MaeEndRight
RULE (HipLeft.y[0] - KneeLeft.y[0]) < 100 &
ABS(KneeLeft.a[0] - 90) < 30
THEN MaeMiddleLeft
RULE (HipLeft.y[0] - KneeLeft.y[0]) < 200 & KneeLeft.a[0]
> 150
THEN MaeEndLeft
RULE (sequenceexists("[MaeMiddleRight,1][MaeStart,1]") &
MaeEndRight)
| (sequenceexists("[MaeMiddleLeft,1][MaeStart,1]") &
MaeEndLeft)
THEN Mae-geri
```