Towards the Evolution of Indirect Communication for Social Robots

Boris Mocialov
Robotics Lab at
School of Mathematical and Computer Sciences and
School of Engineering & Physical Sciences at
Heriot-Watt University and
Edinburgh Centre for Robotics
Edinburgh, UK
Email: bm4@hw.ac.uk

Patricia A. Vargas
Robotics Lab at
School of Mathematical and Computer Sciences at
Heriot-Watt University and
Edinburgh Centre for Robotics
Edinburgh, UK
Email: P.A.Vargas@hw.ac.uk

Micael S. Couceiro
Ingeniarius Ltd.
and University of Coimbra
Mealhada, Portugal
Email: micael@ingeniarius.pt

***Abstract*—This paper presents preliminary investigations on the evolution of indirect communication between two agents. In the future, behaviours of robots in the RoboCup[1] competition should resemble the behaviours of the human players. One common trait of this behaviour is the indirect communication. Within the human–robot–interaction, indirect communication can either be the principal or supporting method for information exchange. This paper summarises previous work on the topic and presents the design of a self–organised system for gesture recognition. Although, preliminary results show that the proposed system requires further feature extraction improvements and evaluations on various public datasets, the system is capable of performing classification of gestures. Further research is required to fully investigate potential extensions to the system that would be able to support real indirect communication in human-robot interaction scenarios.**

I. Introduction

Tasks that include human subjects, such as surveillance, medical diagnosis, human–machine interaction, and sport analysis, require action and behaviour awareness [1]. For example, in sports, collective behaviour is emerged from individual behaviours within the team. These behaviours include which part of the team is attacking and who is defending [2]. There are only few attempts to modelling perception of the game [3], [4], whereas majority of the research apply pattern recognition to understand human movement in sports [5]. Due to the complexity of rules and concepts in the sports context, most of the robotic football policies in the RoboCup competition [6] are either hard–coded, present simplistic behavioural frameworks that do not represent behaviours of the real football players, or require extensive calibrations prior execution and still lack the autonomy while exhibiting very restricted human-like behaviour [4], [7], [8]. This paper presents a system that is the first step to bridging the gap between robotic football and human football. Long–term aim of this study is to pursue the goal of developing an autonomous team of robots that will defeat human players [6]. The initial step in this direction is to copy the way players understand actions and (re-)act accordingly.

Reliability of the communication via direct communication devices, present on the boards of autonomous robotic agents, can be compromised by a range of internal or external factors. While communication link failure between two agents results in temporal communication impairment that can be recovered by communication managing software, physical damage of communication devices usually leads to permanent communication loss [9].

The results of this study led to development of gesture recognition system using evolutionary approach [10], [11]. The system is intended to be uploaded to a robot and updated in real time as the robot learns new gestures from a coach or a teacher. The system had been tested on a PC using standard web camera, first with a human subject, second, with a publicly available reduced ChaLearn dataset[2], and third, with a NAO torso, performing hitting and hugging gestures. PC-based implementation had been used at this stage for convenient testing of key functionality on video data instead of intricate application directly on board of a robotic platform. However, the functionality is believed to be platform–independent. Ultimately, the system is expected to be used on a robotic platform with a standard camera, which will pose additional challenges for the system, such as change of orientation, varying illumination, motion blur, etc. This work represents the first steps towards the creation of a truly self-organised system that applies evolution to facilitate indirect communication between agents.

This paper is organised as follows. Section II reviews the related literature on gesture recognition and feature extraction, Section III describes the developed system, its layers and their functionality, Section IV describes the backbone of the system, its sub-components and their interactions, Section V presents conducted experiments' setup and preparations, while Section VI shows the results obtained from the experiments, Section VII discusses the results obtained and their implications; the project is summarised in Section VIII and the future work is proposed.

[1]RoboCup competition http://www.robocup.org/

[2]ChaLearn Gesture Dataset (CGD 2011), ChaLearn, California, 2011 http://gesture.chalearn.org/data/cgd2011

II. Related Work

In the field of human–computer interaction, two main methods are used for data collection in interaction through indirect communication. These are identified as glove–based and vision–based methods [12]. Previously, LaViola distinguished another hybrid approach that used sensor fusion of the two approaches [13]. On one hand, glove–based devices for interaction data collection generate coherent data, but make the interaction experience cumbersome for the user. On the other hand, vision–based approaches free the user, but tend to introduce additional challenges for the recognition and classification tasks. These challenges, among others, include the variation in light, camera movements, and lack of depth awareness that impacts robustness of the interaction recognition algorithms.

Any gesture recognition system should include (i) data acquisition and pre-processing, (ii) data representation and feature extraction, and (iii) classification or decision-making. These steps form a vision–based framework for the RoboCup scenario in [14]. The important distinction should be made between static and dynamic gestures in the early modelling stages as approaches for feature extraction differ as dynamic gesture recognition requires additional segmentation and tracking modules.

Most distinguished approaches to action representation include Hidden Markov Models as it is done in [15] or straightforward sequence of frames chaining [16]. These approaches consider sequences of frames as action modelling cannot be done without temporal information. Another technique, 'String of feature graphs' (SFGs) [17] represents every frame as a graph of kinematic features. This technique encodes an action by combining the sequence of graphs from every frame. As a result, a sequence of features graphs represents spatio-temporal features of an action.

Different studies on representation and recognition of gestures explored the use of different features. Used features can be classified as either global or local, where local features are more specific and detailed and global features are general and noisy. Global features can be represented as Cartesian distances between centroids of blobs that represent hands on every frame [18]. Less specific classification is done with the clouds of interest points that can represent either shape, speed, density, or all together [19]. Classification of body parts may not be necessary as it is showed by representing gestures by any arbitrary change that happens between the subsequent frames [20]. Local features, such as kinematic points, are less noisy than mentioned above global features and require less post-extraction processing and data cleaning as opposed to global features, for which the amount of noise is proportionate to the amount of data collected [21]. Local features extraction process, nevertheless, requires more precise algorithms.

III. System Design

SFGs approach is used for gesture representation [17]. Feature Graphs (FGs) capture information about kinematic features. Graph data structure allows dynamic addition of new nodes. This serves as an advantage in the gesture representation context as it is not known in advance which gesture is being represented.

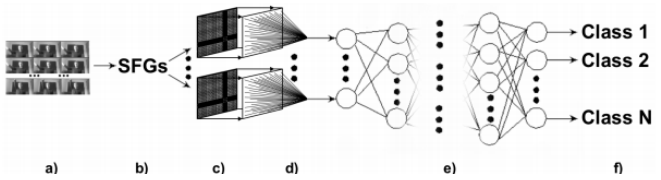

Fig. 1. Overall view of the system
a) extracted features from sequence of frames in a video stream b)) SFG for the video stream c) affinity matrix for the SFG d) feature detectors evolved neural networks e) classifier artificial neural network f) resulting classification of the video stream in a)

A. System Design: Feature Extraction

Feature extraction corresponds to layer a) from Figure 1. The implementation uses OpenCV[3] library due to its useful matrix processing functionality.

Prior to region of interest (ROI) detection, every frame is pre-processed by performing 1) Background-foreground subtraction 2) Illumination reduction and 3) Foreground edges enhancement.

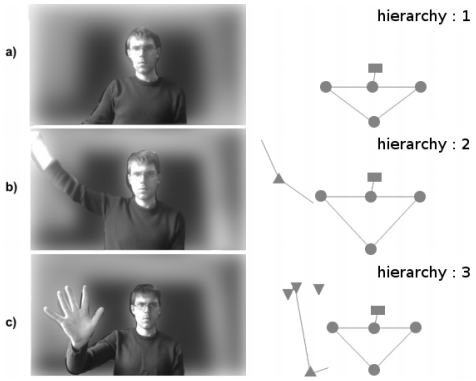

Fig. 2. ROI Detection (Segmentation)
a) Hierarchy 1 (body frame) b) Hierarchy 2 (body frame and limbs) c) Hierarchy 3 (body frame, limbs, and limb details)

1) ROI Detection: ROI consists of the following regions: face, upper body and a moving object. Face and upper body are detected using standard OpenCV Haar feature-based cascade classifier, while movement detection is the result of background subtraction and comparison of the consecutive frames' foregrounds. The movement is classified as a part of the overall body only if it originates from the upper body.

Potential limbs are analysed by looking at hull and convex defects to find break points (elbows) and smaller details (e.g. fingers). The detection is hierarchical and is performed in sequence (e.g. fingers will not be considered until this detail is needed for classification of signs in sign language and the arm has been detected).

[3]OpenCV library http://opencv.org/

2) Feature Extraction: Extracted features are joint positions in space. Following features had been chosen to represent a gesture:

- face and hand with face-hand distance
- first and second hand with hand-hand distance
- first and second shoulder with shoulder-shoulder distance
- first and second elbow with elbow-elbow distance

B. System Design: Feature Encoding

In layer b) from Figure 1, extracted features are encoded as nodes in 2D space and their relations are the Euclidean distances between the nodes encoded as edges in a undirected FG. To describe a video, all FGs are concatenated into a list to make up one SFG.

C. Affinity Matrix Calculation

$$M(a,a) = \begin{cases} \tau_1 \text{ - } d(\imath_1,\, \imath_2) & \text{if } d(\imath_1,\, \imath_2) \leq \tau_1 \\ 0 & \text{otherwise} \end{cases}$$

$$M(a,b) = \begin{cases} \tau_2 \text{ - } d(\imath_1\jmath_1,\, \imath_2\jmath_2) & \text{if } d(\imath_1\jmath_1,\, \imath_2\jmath_2) \leq \tau_2 \\ 0 & \text{otherwise} \end{cases}$$

Fig. 3. Affinity matrix definition

,where

M	: affinity matrix
a, b	: matrix indices
τ_1, τ_2	: threshold values, where τ_1 is the maximum allowed Euclidean distance between two nodes and τ_2 is the maximum allowed deviation between edges inclinations
$\imath_1, \jmath_1, \imath_2, \jmath_2$	: nodes of SFGs or edge between nodes $(\imath_1, \jmath_1)$ and $(\imath_2, \jmath_2)$
$d(\imath_1, \imath_2)$	: distance between two nodes that belong to different FGs
$d(\imath_1\jmath_1, \imath_2\jmath_2)$	: inclination between edges that belong to different FGs

In layer c) from Figure 1, SFGs are transformed into affinity matrices that hold similarity information between all frames in a single matrix. Figure 3 formally describes that the diagonal holds information about similarity between nodes, while the rest of the matrix represents similarity between edges [17].

D. System Design: Detectors

This implementation, as shown in layer d) from Figure 1, uses artificial neural networks as an alternative to spectral clustering, performed on the resulting affinity matrices as it is done in [17]. By using neural networks, the system is able to classify gestures directly from video stream without the need to compare every learned template gesture to the stream.

Kocmánek in [22] presents a method for handwritten digit recognition with HyperNEAT [23] algorithm. The algorithm evolves novel detectors that extract unique features from images. Similar approach is used in this paper with only difference in that the system is operating on the spatio-temporal data, encoded as affinity matrices.

Neural network processing (hnn[4]) package together with Python-based implementation of the HyperNEAT algorithm (peas[5]), developed in [24], are used to evolve distinct detectors for SFG gesture representations.

For all experiments 50 detectors with 100 inputs, no hidden layers, and a single output are evolved using novelty search technique. This leads to different detectors focusing on different sections of the affinity matrices.

In this implementation, the HyperNEAT algorithm is restricted to produce detectors of certain topology as described in [22]. For more complex detectors, future evolutions of detectors could be more elaborate, evolving the size and the activation functions of the detectors.

TABLE I
HYPERNEAT PARAMETERS FOR DETECTORS' EVOLUTION

Substrate	Inputs 10×10 Outputs 0×1
Generations	depending on the experiment
Population	50
Inputs per individual	100
Outputs per individual	1
Maximum depth	3
Weights range	(-3.0, 3.0)
P(new connection)	0.3
P(new node)	0.1
P(weight mutation)	0.8
P(weight reset)	0.1
P(disable connection)	0.01
P(re-enable connection)	0.01
Node types range	tanh
Evaluation function	argmax($\sum_1^k$Manhattan(k-NN($output_{detector}$)))
Minimum allowed fitness	0.05

Every detector is evolved by a separate instance of peas algorithm using parameters, given in Table I.

Substrate consists of two fully connected layers. Input layer has at most 10×10 nodes and output layer has at most 0×1 nodes. 'P' is the probability of adding new connections, adding new nodes, etc. Evaluation function objective is to maximise the Manhattan distance between all the detectors. The aim is to evolve novel detectors.

In *k*-NN, the *k*-nearest neighbour, $k = 50$ (all other detectors are considered). The problem of maximising the distance between the evolved detectors is reduced to finding the maximum Manhattan distance between a set of arrays.

A single vector is associated with every detector with as many items as there are gestures to be learned by the system. The vector is used to accumulate activations of the output neuron for every gesture. The vector describes how many times the detector detected something in affinity matrix.

E. System Design: Classifier

As can be seen in layer e) from Figure 1, the classifier has same amount of inputs as there are detectors in the system, with every detector feeding its output into the classifier's dedicated input. In this setup, the classifier has 50 inputs, 2 hidden layers with 300 neurons in each and certain amount of outputs, depending on the experiment, with every output

[4]A reasonably fast and simple neural network library https://hackage.haskell.org/package/hnn

[5]Python Evolutionary Algorithms https://github.com/noio/peas/

representing a probability of the gesture class, associated with that output.

The library[6] uses resilient backpropagation (Rprop) [25] as network training method.

IV. Sub-Systems Interaction

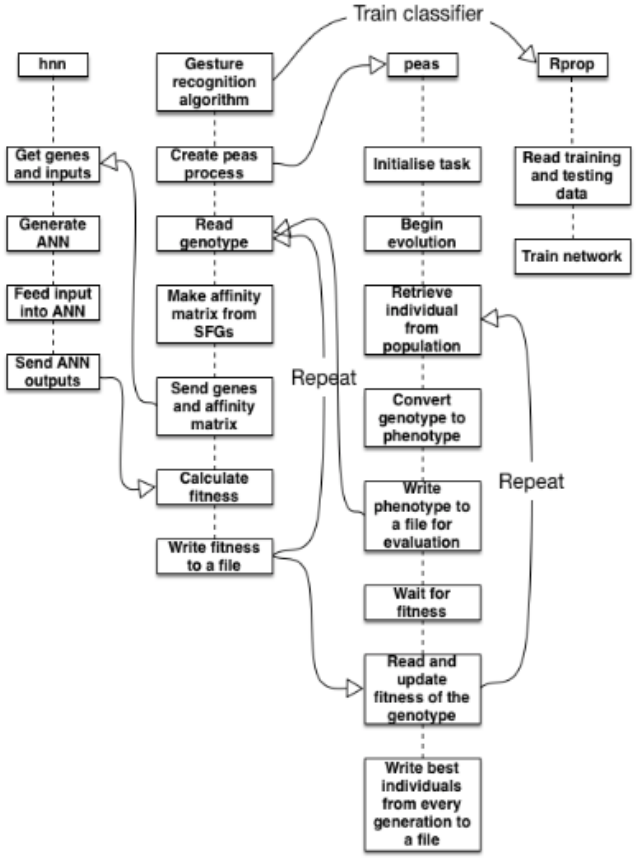

Fig. 4. Sub-Systems Execution and Communication Sequence Diagram
a) hnn (Haskell neural network library) b) C/C++ gesture recognition algorithm c) peas (Python HyperNEAT algorithm) d) Resilient Backpropagation implemented in Matlab

The system consists of 4 parts and is presented in Figure 4. Data exchange between peas and gesture recognition algorithms is done through the file system.

Gesture recognition algorithm launches the HyperNEAT [23] instances to begin the evolution of detectors. Once the instance is launched, the gesture recognition algorithm waits for generated neural networks (genotypes) from the instance. When genotype is generated, it is written to a file and the HyperNEAT is paused until the evaluation results are written to another file. Both files are used to exchange data between the two algorithms. When the genotype is received, gesture recognition algorithm evaluates it on training data, calculates the fitness, writes the fitness to the file, and pauses until the next genotype becomes available. At this time, HyperNEAT algorithm continues, reads the fitness of the genotype, and writes it for further evaluations of the population. When a genotype is available and affinity matrix is ready to be evaluated, hnn is invoked.

When the evolution of detectors has finished, the gesture recognition algorithm launches Rprop algorithm that trains classifier neural network using detectors' outputs as inputs into the classifier.

[6]Rprop training for Artificial Neural Networks
http://uk.mathworks.com/matlabcentral/fileexchange/32445-rprop

V. Experiment Setup

The system has been tested on three gesture datasets. First, single subject, self-made 4 different gestures (left hand wave, right hand wave, both hands wave simultaneously, and no hands waving). Second experiment used single subject's 10 more complicated signaling gestures from ChaLearn dataset. Third, a self-made NAO gesturing dataset was created with 2 gestures (raise one arm up as trying to hit and spread both arms apart as trying to hug).

Before the training of the model, the raw video data for all datasets were transformed into affinity matrices.

A. Experiment Preparation with Self-Made Gestures Dataset

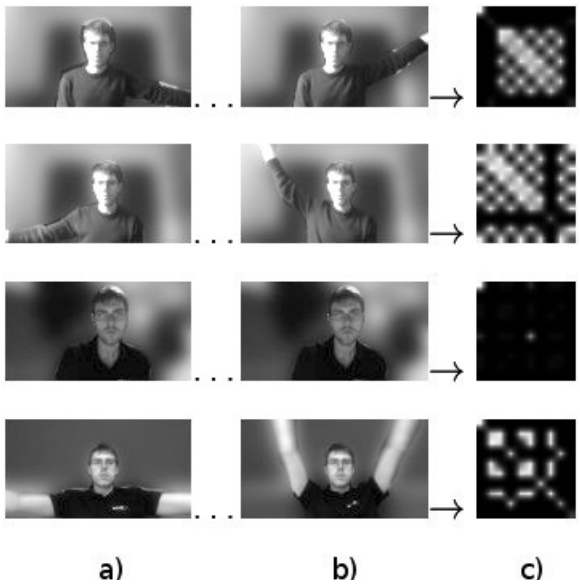

Fig. 5. Affinity matrices generation from gestures in video data from self-made gestures dataset (single gesture approx. 1-2 seconds)
a) first frame b) last frame c) generated affinity matrix

Figure 5 shows that extracted kinematic features from videos are being used to generate affinity matrices, which are different for every gesture. Showed results correspond to steps A, B, and C from Section III.

B. Experiment Preparation with ChaLearn Signaling Gestures Dataset

Figure 6 shows same kinematics features extracted from videos of another dataset without a single change to the feature extraction algorithm. Extracted features are then encoded in corresponding affinity matrices.

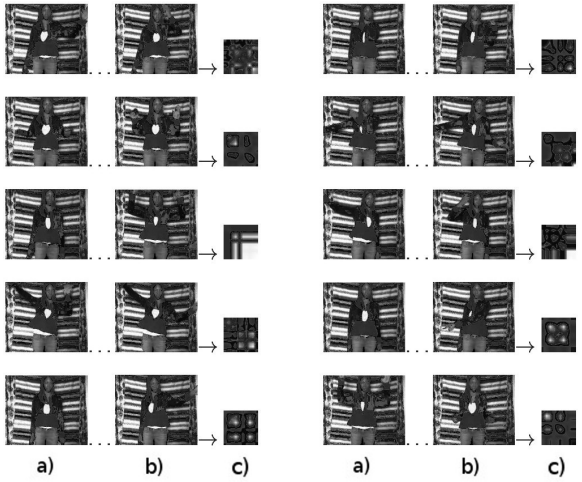

Fig. 6. Affinity matrices generation from gestures in video data from partial ChaLearn signaling gestures dataset (single gesture approx. 2-5 seconds)
a) first frame b) last frame c) generated affinity matrix

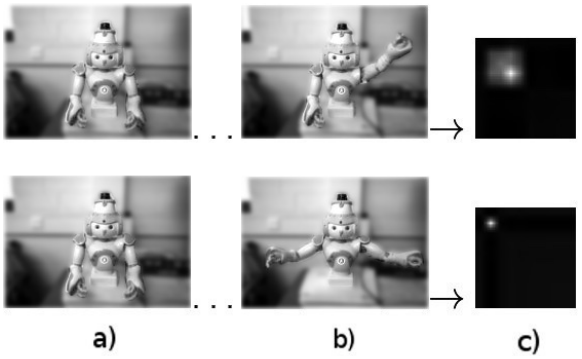

Fig. 7. Affinity matrices generation from gestures in video data from self-made NAO gestures dataset (single gesture approx. 50 seconds)
a) first frame b) last frame c) generated affinity matrix

C. Experiment Preparation with Self-Made NAO Gestures Dataset

Figure 7 shows the same feature extraction algorithm used on an artificial subject. Separate Haar feature-based cascade classifier was trained to detect face and torso the artificial subject.

VI. Experiments Results

All experiments had been conducted using leave-one-out strategy, testing on a single testing instance for every gesture class.

Neither feature extraction, nor feature encoding, nor affinity matrix algorithms have been edited between experiments, except for use of different Haar feature-based cascade classifier when capturing features of an artificial subject.

Images of the evolved detectors represent neural network weights as heatmaps. The black colour means positive weights and grey colour represents negative weights, while the white colour means the weight is zero. There is no identification or order in the presented heatmaps. Distinctiveness of the detectors for individual experiment has importance on the accuracy of the system.

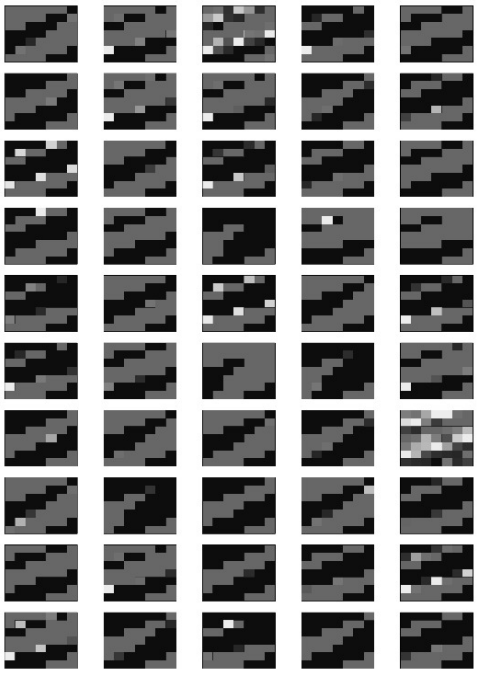

Fig. 8. Heatmaps of 50 detectors, evolved for 130 generations for one experiment, for the self-made gestures dataset

Fitness score is superficial as the evolution would never be able to reach 100% fitness. The maximum is taken as a case when all the values of affinity matrices are uniformly distributed, which is never the case with affinity matrices for gestures.

A. Experiment Preparation with Self-Made Gestures Dataset

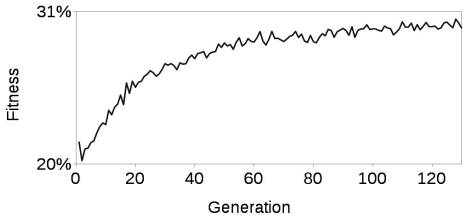

Fig. 9. Average fitness development of detectors evolved for 130 generations for self-made gestures dataset

1) Fitness Evolution: The fitness evolution appears to be steady and continues until 130_{th} generation where it becomes apparent that the fitness tends to converge at around 31% fitness. The evolution is terminated manually after 130 generations.

2) Evolved Detectors: Evolved detectors after 130 generations are presented in Figure 8. Most of the detectors on the figure are slightly different from each other. This shows that the evolutionary algorithm attempted to make detectors

distinct, but more evolutions were required to see bigger differences.

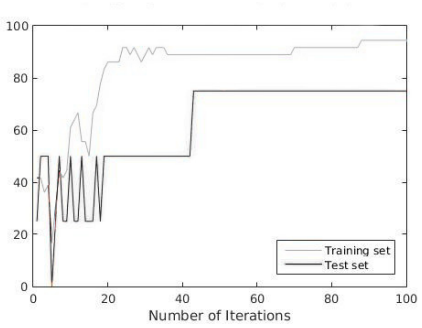

Fig. 10. Classifier training with Rprop for self-made gestures dataset

3) Classifier Training: Figure 10 shows training of the classifier for 100 generations. The training accuracy lies between 90% and 95%, while the testing accuracy achieved 75%.

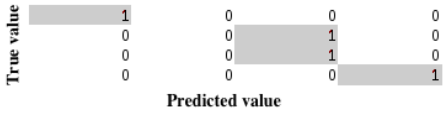

Fig. 11. Confusion matrix for self-made gestures dataset with leave-one-out strategy using one example for classifier, described in Section III-E. Average accuracy: 75%

4) Confusion Matrix: The recognition algorithm confused the raise one arm up gesture with the not raising arms up gesture. This may have happened due to the poor feature extraction on that particular case, where the arm may not have been detected by the algorithm.

B. Experiment Preparation with ChaLearn Signaling Gestures Dataset

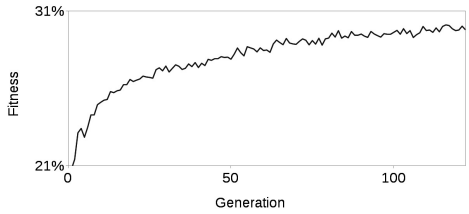

Fig. 12. Average fitness development of detectors evolved for 120 generati for ChaLearn gestures dataset

1) Fitness Evolution: The fitness development, shown Figure 12, becomes unstable after approximately 60 generations and tends to converge at around 31% fitness. The evolution is terminated after 120 generations.

2) Evolved Detectors: Figure 13 presents evolved detectors for ChaLearn dataset after 120 generations. Although the pattern is very similar for most of the detectors, definite variety can be noticed.

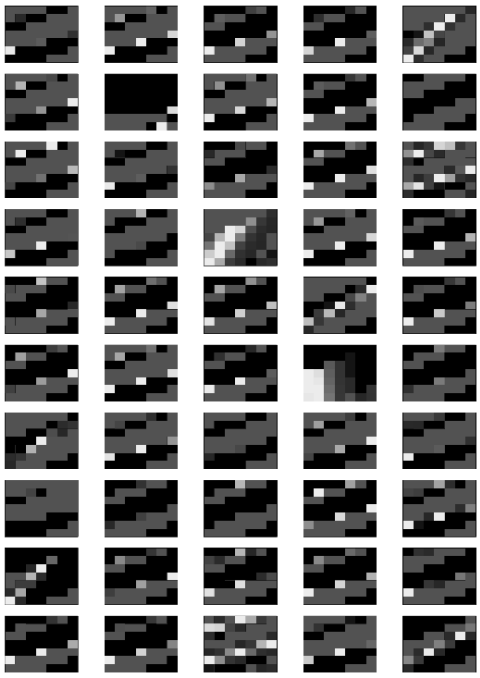

Fig. 13. Heatmaps of 50 detectors, evolved for 120 generations for one experiment, for ChaLearn gestures dataset

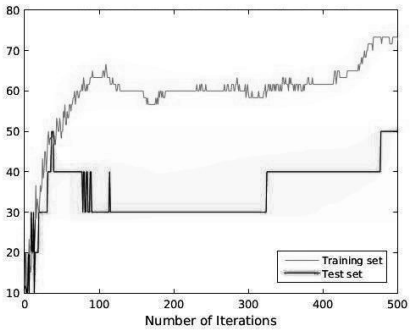

Fig. 14. Classifier training with Rprop for ChaLearn gestures datase

3) Classifier Training: Figure 14 shows training of the classifier. Accuracy of the training set lies around 70%, while the test dataset accuracy is no greater than 50%.

Fig. 15. Confusion matrix for ChaLearn gestures dataset with leave-one-out strategy using one example for classifier, described in Section III-E. Average accuracy: 50%

4) Confusion Matrix: Figure 15 shows confusion matrix for the ChaLearn dataset. It is apparent that the algorithm has many misclassifications. In particular, the classifier labels the

first gesture as the fourth one. The gestures are indeed very similar with the only difference in another hand active during the gesturing of the third gesture. This can be explained by the poor feature extraction. Same holds for gesture number 7 being mixed with gesture number 4.

C. Experiment Preparation with Self-Made NAO Gestures Dataset

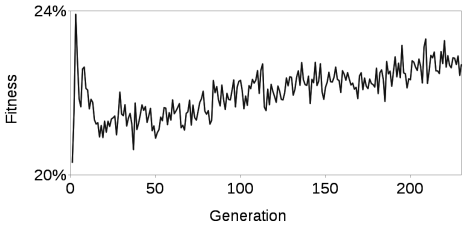

Fig. 16. Average fitness development of detectors evolved for 230 generations for self-made NAO gestures dataset

1) Fitness Evolution: Figure 16 presents fitness evolution of detectors, applied on encoded and transformed NAO gestures. The evolution is very unstable, but slowly improving. There is a spike of fitness in the first generations, for which there is no definite explanation. This may have been a feature of evolved detectors that performed very well on the training data, but that feature was lost in the next generations. The evolution is terminated after 230 generations.

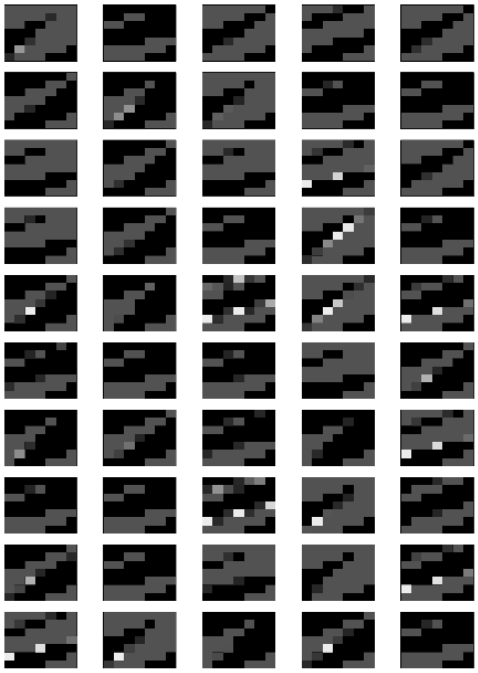

Fig. 17. Heatmaps of 50 detectors, evolved for 230 generations for one experiment, for self-made NAO gestures dataset

2) Evolved Detectors: Figure 17 shows evolved detectors after 230 generations. Some detectors are similar, but overall some variety is noticeable.

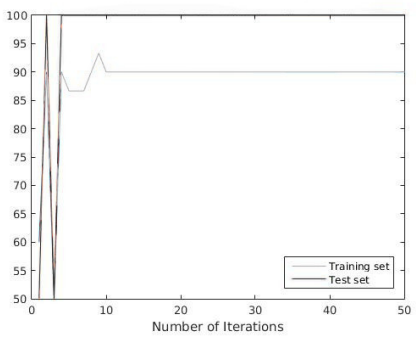

Fig. 18. Classifier training with Resilient Backpropagation for self-made NAO gestures dataset

3) Classifier Training: Figure 18 shows the training of the classifier. Although the accuracy on training dataset is around 90%, the test dataset scores 100% in just few iterations. This can be explained by the fact that only two gestures are classified.

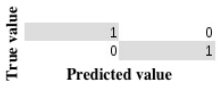

Fig. 19. Confusion matrix for self-made NAO gestures dataset with leave-one-out strategy using one example for classifier, described in Section III-E. Average accuracy: 100%

4) Confusion Matrix: Since the accuracy on the training set is 100%, confusion matrix shows that the predicted class for two gestures is always correct.

VII. Discussion

It can be seen from the affinity matrices, presented in Figures 11, 15, and 19, that the classification accuracy drops down as the number of gesture classes increases (2 gestures - 100% accuracy, 4 gestures - 75% accuracy, 10 gestures - 50% accuracy). Complexity of the gestures was not expected to be a major factor in the recognition accuracy. The major factor that affects the accuracy, on the other hand, is the feature extraction, which currently is very simple and does not account for such gesture details as seen in ChaLearn dataset. Currently, the feature extraction looks only at the face, upper body and the limbs of the subject.

During the experiments it had been noticed that the feature extraction should be tailored to every dataset due to the variations in camera positioning with respect to the subject, illumination, and others. Nevertheless, the system is robust enough, considering that nothing had been changed between the different experiments.

With different datasets, which may include more gesture details (e.g. sign languages), the system would have to be improved by extending the feature extraction.

Evolved detectors had very little variation in all conducted experiments. This may be due to few evolution generations or the incorrectness of the fitness function. Evolution of detectors

has to be studies separately to investigate how many distinct detectors can be evolved.

VIII. Conclusion

This research has planned the initial steps to research on indirect communication between two agents. As a result of the study, a real-time gesture recognition system had been produced that is partly developed with the use of the evolutionary techniques.

Preliminary results of this project show that the gesture recognition using the proposed system is possible and can be refined by improving the accuracy of the feature extraction algorithm. This work should be seen as the first step towards the creation of real self-organised systems based on evolution that can be applied to social robots and thus facilitate human-robot-interaction.

Future work lies in further testing of the system on public datasets. Further on, potential extensions to the system may include additional feature extraction to accommodate the algorithm for the sign language recognition and processing. Segmentation is another possible extension that would make the system even more robust and complete.

Acknowledgment

This work has been supported by the Heriot-Watt University School of Engineering & Physical Sciences James Watt Scholarship through Edinburgh Centre for Robotics.

References

[1] J. Aggarwal and S. Park, "Human motion: modeling and recognition of actions and interactions," in *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, Sept 2004, pp. 640–647.

[2] K. Davids, D. Araújo, and R. Shuttleworth, "Applications of dynamical systems theory to football," *Science and football V*, pp. 537–550, 2005.

[3] F. Dylla, A. Ferrein, G. Lakemeyer, J. Murray, and O. Obst, "Approaching a formal soccer theory from behaviour specifications in robotic soccer ."

[4] A. Bogdanovych, C. Stanton, X. Wang, and M.-A. Williams, *RoboCup 2011: Robot Soccer World Cup XV*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Real-Time Human-Robot Interactive Coaching System with Full-Body Control Interface, pp. 562–573. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-32060-6$_4$8

[5] Y. Kong, X. Zhang, Q. Wei, W. Hu, and Y. Jia, "Group action recognition in soccer videos," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, Dec 2008, pp. 1–4.

[6] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawai, and H. Matsubara, *RoboCup-97: Robot Soccer World Cup I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, ch. RoboCup: A challenge problem for AI and robotics, pp. 1–19. [Online]. Available: http://dx.doi.org/10.1007/3-540-64473-3$_4$6

[7] A. Bezek, M. Gams, and I. Bratko, "Multi-agent strategic modeling in a robotic soccer domain," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006, pp. 457–464.

[8] T. Nakashima, M. Takatani, M. Udo, H. Ishibuchi, and M. Nii, *RoboCup 2005: Robot Soccer World Cup IX*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, ch. Performance Evaluation of an Evolutionary Method for RoboCup Soccer Strategies, pp. 616–623. [Online]. Available: http://dx.doi.org/10.1007/11780519$_6$1

[9] R. N. Parasuraman, K. Kershaw, and M. F. Perez, "Experimental investigation of radio signal propagation in scientific facilities for telerobotic applications," *International Journal of Advanced Robotic Systems*, vol. 10, no. 364, pp. 1–11, July 2013. [Online]. Available: http://oa.upm.es/30850/

[10] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology,Intelligence,and Technology*. Cambridge, MA, USA: MIT Press, 2000.

[11] P. Vargas, E. Di Paolo, I. Harvey, and P. Husbands, *The Horizons of Evolutionary Robotics*. MIT Press, 3 2014.

[12] P. Garg, N. Aggarwal, and S. Sofat, "Vision based hand gesture recognition."

[13] J. J. LaViola, Jr., "A survey of hand posture and gesture recognition techniques and technology," Providence, RI, USA, Tech. Rep., 1999.

[14] P. Trigueiros, F. Ribeiro, and L. Reis, "Generic system for human-computer gesture interaction," in *Autonomous Robot Systems and Competitions (ICARSC), 2014 IEEE International Conference on*, May 2014, pp. 175–180.

[15] M. Malgireddy, I. Inwogu, and V. Govindaraju, "A temporal bayesian model for classifying, detecting and localizing activities in video sequences," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, June 2012, pp. 43–48.

[16] D. Faria, C. Premebida, and U. Nunes, "A probabilistic approach for human everyday activities recognition using body motion from rgb-d images," in *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on*, Aug 2014, pp. 732–737.

[17] U. Gaur, Y. Zhu, B. Song, and A. Roy-Chowdhury, "A "string of feature graphs" model for recognition of complex activities in natural videos," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, Nov 2011, pp. 2595–2602.

[18] M. Brand, N. Oliver, and A. Pentland, "Coupled hidden markov models for complex action recognition," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, Jun 1997, pp. 994–999.

[19] M. Bregonzio, S. Gong, and T. Xiang, "Recognising action as clouds of space-time interest points," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June 2009, pp. 1948–1955.

[20] G. Willems, T. Tuytelaars, and L. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *Proceedings of the 10th European Conference on Computer Vision: Part II*, ser. ECCV '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 650–663. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88688-4$_4$8

[21] D. Lisin, M. Mattar, M. Blaschko, E. Learned-Miller, and M. Benfield, "Combining local and global image features for object class recognition," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, June 2005, pp. 47–47.

[22] T. Kocmánek, "HyperNEAT and Novelty Search for Image Recognition," Master's thesis, Czech Technical University in Prague, 2015.

[23] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artif. Life*, vol. 15, no. 2, pp. 185–212, Apr. 2009. [Online]. Available: http://dx.doi.org/10.1162/artl.2009.15.2.15202

[24] T. G. van den Berg and S. Whiteson, "Critical factors in the performance of hyperneat," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13. New York, NY, USA: ACM, 2013, pp. 759–766. [Online]. Available: http://doi.acm.org/10.1145/2463372.2463460

[25] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the rprop algorithm," in *Neural Networks, 1993., IEEE International Conference on*, 1993, pp. 586–591 vol.1.