Synthesis of Spatio-Temporal Descriptors for Dynamic Hand Gesture Recognition Using Genetic Programming

Li Liu
Department of Electronic and Electrical Engineering
The University of Sheffield
S1 3JD, UK
Email: elp11ll@sheffield.ac.uk

Ling Shao
Department of Electronic and Electrical Engineering
The University of Sheffield
S1 3JD, UK
Email: ling.shao@sheffield.ac.uk

Abstract—Automatic gesture recognition has received much attention due to its potential in various applications. In this paper, we successfully apply an evolutionary method—genetic programming (GP) to synthesize machine learned spatio-temporal descriptors for automatic gesture recognition instead of using hand-crafted descriptors. In our architecture, a set of primitive low-level 3D operators are first randomly assembled as tree-based combinations, which are further evolved generation-by-generation through the GP system, and finally a well performed combination will be selected as the best descriptor for high-level gesture recognition. To the best of our knowledge, this is the first report of using GP to evolve spatio-temporal descriptors for gesture recognition.

We address this as a domain-independent optimization issue and evaluate our proposed method, respectively, on two public dynamic gesture datasets: Cambridge hand gesture dataset and Northwestern University hand gesture dataset to demonstrate its generalizability. The experimental results manifest that our GP-evolved descriptors can achieve better recognition accuracies than state-of-the-art hand-crafted techniques.

I. INTRODUCTION

Automatic gesture recognition plays a significant role in many modern applications such as human-computer interaction, interactive gaming and sign language interpretation. Research in areas including motion analysis, machine learning and pattern recognition all contributes to a reliable gesture recognition system. Many previous studies have been carried out to categorize human action and gesture classes in video sequences. A gesture recognition system usually consists of two main stages: first, low-level feature extraction and representation; second, high-level gesture classification. For feature extraction and representation, the marriage of space-time interest points such as [1] and bag-of-words [2] is a popular choice. The histogram representations are further combined with either a Support Vector Machine (SVM) or a probabilistic model [3] for classification. However, representations based on local spatio-temporal features [4], [5] are often not precise and informative due to the quantization errors during codebook construction and the loss of structural relationships among local features. On the other hand, holistic representations [6], which treat a video sequence or a cropped volume of it as a whole, can extract all the structural and motion information from both spatial and temporal dimensions. The drawback of holistic representations is that they are often sensitive to shift, scaling, occlusion and cluttering, since background subtraction and segmentation tend not to be very accurate.

In this work, we attempt to automatically design holistic spatio-temporal descriptors that are robust to shift, scaling and background changes for hand gesture recognition. Our design is based on an evolutionary method, i.e., Genetic Programming (GP) [7], which simulates the Darwinian principle of natural selection to solve optimization problems. Different from other hand-crafted techniques based on deep domain knowledge, GP is inspired by natural evolution and can be employed to automatically solve problems without prior knowledge of the solutions. Users can utilize GP to solve a wide range of practical problems, producing human-competitive results and even patentable inventions. Relying on natural and random processes, GP can escape traps by which deterministic methods may be captured. Because of this, usage of GP is not limited to any research domain and creates relatively generalized solutions for any target tasks.

Aiming for robust dynamic hand gesture recognition, we use GP to generate novel holistic descriptors for representing gesture sequences. Given a group of 3D sequence-processing operators, GP first randomly assemble them into a variety of descriptors as the initialized population. Then, the population is continually evolved and evaluated by calculating the recognition error rate to choose the hopefully better performing individuals into the next generation. In the present setting, we wish to identify the descriptor which maximizes the recognition performance in GP running. This is an NP-hard search problem which evolutionary methods may solve in a tractable amount of computer time compared to exhaustive enumerative search. After the training of GP finishes, one best-so-far individual can be automatically selected by machine as the final solution (i.e., the near-optimal descriptor).

The contributions of this paper lie in the following two
aspects:
(1) We utilize genetic programming (GP) to generate machine-learned spatio-temporal descriptors for gesture recognition.

(2) The proposed methodology and the generated descriptors can be applied to other video analysis related applications directly.

The remainder of the paper is organized as follows: In Section 2, related work is reviewed. The architecture of our methodology is detailed in Section 3. Experiments and results are described in Section 4. In Section 5, we conclude this paper.

II. RELATED WORK

A comprehensive survey of recent gesture recognition methods can be found in [8]. Since our work is on the design of descriptors, we briefly review feature representation-based methods for gesture recognition. Holte et al. [9] proposed a view-invariant gesture recognition algorithm by finding motion primitives in the 3D data using 3D optical flow and harmonic motion context representation. A probabilistic Edit Distance classifier is further applied to identify which gesture best describes a string of primitives. Cutler and Turk [10] developed a real-time, view-based gesture recognition system, in which optical flow is estimated and segmented into motion blobs. Gestures are recognized using a rule-based technique charactering the motion blobs. Campbell et al. [11] evaluated ten different features combined with HHMs for learning and recognition of gestures and the final results indicate that velocity features are superior to positional features, and partial rotational invariance is sufficient for accomplishing good performance. Bretzner et al. [12] represented hand poses using multi-scale color image features with qualitative inter-relations in terms of scale, position and orientation.

Genetic programming (GP), as a powerful machine learning method, has been gradually adopted in computer vision. Trujillo and Olague [13] used GP for the automatic generation of a 2D low-level feature extractor that can be applied to high-level computer vision tasks. Torres et al. [14] utilized GP to find a well-performed combination of the similarity functions for image retrieval. Poli [15] applied GP to automatically select an efficient optimal filter for segmenting the brain in medical images. On the same line, a GP-based detector was proposed by Howard et al. [16] to detect ships in synthetic aperture radar (SAR) images. Davis et al. [17] adopted GP to select the most discriminative features for multivariate data analysis without any prior information. In this paper, we use GP to automatically synthesize spatio-temporal descriptors from a set of 3D filters and operators for dynamic hand gesture recognition. A simplified version of our method has been applied to extract features for action recognition in [18].

III. METHODOLOGY

In this paper, we propose a domain-independent machine learning methodology to automatically generate low-level spatio-temporal descriptors for high-level gesture recognition using genetic programming (GP). A group of 3D operators are assembled to construct an effective problem-specific descriptor which is capable of selectively extracting features from hand gestures. The final evolved descriptor, combining the nice properties of those primitive 3D operators, can both extract meaningful features and form a compact gesture representation. We learn our proposed system over a training set, in which descriptors are evolved by maximizing the recognition accuracy through a fitness function, and further evaluate the GP-selected one over a testing set to demonstrate the performance of our method. The architecture of our proposed model is illustrated in Fig. 1.

Genetic programming (GP) is an evolutionary computation (EC) [19] technique that automatically solves problems without requiring the user to know or specify the form or structure of the solution in advance. The basic steps in GP is shown in Fig. 2. Generally, GP programs can be represented as a tree structure, evolved (by selection, crossover and mutation) through sexual reproduction with pairs of parents being chosen stochastically but biased in their fitness on the task at hand, and finally select the best performing individual as the terminal solution. In our method, each individual in GP represents a candidate spatio-temporal descriptor and is evolved continuously through generations. To establish the architecture of our model, three significant concepts: function set, terminal set and fitness function should be first defined.
A. Function Set and Terminal Set

A key component of GP is the function set which constitutes the internal nodes of the tree and is typically driven by the nature of the problem. To make the GP evolution process fast, more efficient operators that can extract meaningful information from gesture sequences are preferred. Our function set consists of 18 unary operators and 3 binary ones, including processing filters and basic arithmetic functions, as illustrated in Table I.

In our GP structure, we divide our function set into two tiers: filtering tier (bottom tier) and max-pooling tier (top tier). The order of these layers in our structure is always fixed. This means, in any GP-evolved program, the operators in the filtering tier must be located below the operators in the max-pooling tier. In addition, not all the operators listed in the function set have to be used in a given tree and the same operator can be used more than once. The topology of the tree is essentially unrestricted.

1) Filtering Tier: In the filtering tier, aiming to extract meaningful features from dynamic hand gestures, we adopt 3D Gaussian filters, 3D Laplacian filters, 3D wavelet filters, 3D Gabor filters and some other sequence processing operators and basic arithmetic functions.

3D Gaussian filters are adopted due to their ability for denoising and 3D Laplacian filters are used for separating signals into different spectral sub-bands. 2D Gaussian and Laplacian operators have been successfully applied to capture intensity features for scene classification in [20]. Wavelet transforms can perform multi-resolution analysis and obtain the contour information of hand gestures by using the 3D CDF ’9/7’ [21] wavelet filters. 3D Gabor filters are regarded as the most effective method to obtain the orientation information in a sequence. Following Riesenhuber and Poggio’s work [22], we simulate the biological mechanism of the visual cortex to define our Gabor filter-based operators. Firstly, we convolve an input gesture sequence with Gabor filters at six different scales \((7\times7, 9\times9\times9, 11\times11\times11, 13\times13\times13, 15\times15\times15 \text{ and } 17\times17\times17)\) under a certain orientation (i.e., 0, 45, 90, or 135 degree); We further apply the max operation to pick the maximum value across all six convolved sequences for that particular orientation. Fig. 3 illustrates the procedure of our Multi-scale-max Gabor filters for a certain orientation. The max operation among different scales is defined as follows:

\[
I_{MAX} = \max \{I_{7\times7\times7}(x, y, z, \theta_s), I_{9\times9\times9}(x, y, z, \theta_s), \ldots, I_{15\times15\times15}(x, y, z, \theta_s), I_{17\times17\times17}(x, y, z, \theta_s)\}
\]

where \(I_{MAX}\) is the output of the Multi-scale-max Gabor filter. \(I_{i\timesi\timesi}(x, y, z, \theta_s)\) denotes the convolved sequences with the scale \(i\timesi\timesi\) and the orientation \(\theta_s\).

Beyond that, several other 3D operators that are common for feature extraction are added to the function set to increase the variety of the selection for composing individuals during the GP running. Basic arithmetic functions are chosen to realize operations such as addition and subtraction of the internal nodes of the tree to make the whole evolution procedure more natural.

2) Max-pooling Tier: In the max-pooling tier, we include four functions listed in Table I, which are performed over local neighborhoods with windows varying from \(5\times5\times5\) to \(20\times20\times20\) with a shifting step of 5 pixels. This max-pooling operation (see Fig. 4) is a key mechanism for object recognition in the cortex and provides a more robust response, successfully tolerating shift and scaling, in the case of recognition in clutter or with multiple stimuli in the receptive field [22]. Given a sequence, max-pooling functions will pick out the local max values from the input and shrink it along spatial and temporal dimensions to compose a more compact representation of the input sequence. To ensure the closure property [23], we further resize outputs calculated from max-pooling functions to an identical size as inputs using linear interpolation. In this way, the sizes of inputs and outputs of our max-pooling functions are the same.

In addition, the terminal set is also a significant component of genetic programming. For gesture recognition, we consider the following aspects of our task: (1) The terminal set must capture the holistic information of each gesture sequence; (2) During the evolution process, the evaluation of the fitness function must be efficient. In our implementation, we use the raw gesture sequence as the terminal set. In each tree-based genetic structure, a gesture sequence is located as the bottom leaf of the entire tree and connects with the higher function nodes directly. A representative GP tree is illustrated in Fig. 5.

B. Fitness Function

The most important part of genetic programming is the fitness function which determines how well a program is able to solve the problem. To evaluate the candidate GP-evolved descriptors, we here adopt the classification accuracy calculated by a linear SVM classifier on the training set as the fitness function. In our GP architecture, for any of the input gesture sequences, we can obtain an output sequence with an identical size as the input due to the enclosure property.

We further divide the output by a \(10\times10\times5\) grid. To make the final representation further invariant to shift and scaling, we take the mean values of each divided sub-block and concatenate them into a 500D vector as the input to the linear-SVM as shown in Fig. 6. To obtain a more reliable fitness
<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Input</th>
<th>Function Description</th>
<th>Operator Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gau1</td>
<td>1 Sequence</td>
<td>3D Gaussian smooth filter with $\sigma = 1$</td>
<td>Filter</td>
</tr>
<tr>
<td>Gau2</td>
<td>1 Sequence</td>
<td>3D Gaussian smooth filter with $\sigma = 2$</td>
<td>Filter</td>
</tr>
<tr>
<td>LoG1</td>
<td>1 Sequence</td>
<td>3D Laplacian of Gaussian filter with $\sigma = 1$</td>
<td>Filter</td>
</tr>
<tr>
<td>LoG2</td>
<td>1 Sequence</td>
<td>3D Laplacian of Gaussian filter with $\sigma = 2$</td>
<td>Filter</td>
</tr>
<tr>
<td>GBO-0</td>
<td>1 Sequence</td>
<td>3D Multi-scale-max Gabor filter with orientation of 0 degree</td>
<td>Filter</td>
</tr>
<tr>
<td>GBO-45</td>
<td>1 Sequence</td>
<td>3D Multi-scale-max Gabor filter with orientation of 45 degrees</td>
<td>Filter</td>
</tr>
<tr>
<td>GBO-90</td>
<td>1 Sequence</td>
<td>3D Multi-scale-max Gabor filter with orientation of 90 degrees</td>
<td>Filter</td>
</tr>
<tr>
<td>GBO-135</td>
<td>1 Sequence</td>
<td>3D Multi-scale-max Gabor filter with orientation of 135 degrees</td>
<td>Filter</td>
</tr>
<tr>
<td>Wavelet</td>
<td>1 Sequence</td>
<td>3D CDF '9/7' wavelet filter</td>
<td>Filter</td>
</tr>
<tr>
<td>Aver</td>
<td>1 Sequence</td>
<td>3D Averaging filter with $5 \times 5 \times 5$ sampling window</td>
<td>Filter</td>
</tr>
<tr>
<td>Med</td>
<td>1 Sequence</td>
<td>3D Median filter with $5 \times 5 \times 5$ sampling window</td>
<td>Filter</td>
</tr>
<tr>
<td>ABS</td>
<td>1 Sequence</td>
<td>Take the absolute value pixel by pixel</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>DoF</td>
<td>1 Sequence</td>
<td>Subtract between adjacent frames of the input sequence</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>LOG2</td>
<td>1 Sequence</td>
<td>Take the logarithm of with 2 at the bottom for the input sequence</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>ADD</td>
<td>2 Sequences</td>
<td>Add two input sequences pixel by pixel</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>SUB</td>
<td>2 Sequences</td>
<td>Subtract two input sequences pixel by pixel</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>ABSSub</td>
<td>2 Sequences</td>
<td>Absolute subtract two input sequences pixel by pixel</td>
<td>Arithmetic</td>
</tr>
</tbody>
</table>

Max-pooling tier

<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Input</th>
<th>Function Description</th>
<th>Operator Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max-pooling5</td>
<td>1 Sequence</td>
<td>3D max-pooling with pooling window size $5 \times 5 \times 5$ on the input sequence</td>
<td>Filter</td>
</tr>
<tr>
<td>Max-pooling10</td>
<td>1 Sequence</td>
<td>3D max-pooling with pooling window size $10 \times 10 \times 10$ on the input sequence</td>
<td>Filter</td>
</tr>
<tr>
<td>Max-pooling15</td>
<td>1 Sequence</td>
<td>3D max-pooling with pooling window size $15 \times 15 \times 15$ on the input sequence</td>
<td>Filter</td>
</tr>
<tr>
<td>Max-pooling20</td>
<td>1 Sequence</td>
<td>3D max-pooling with pooling window size $20 \times 20 \times 20$ on the input sequence</td>
<td>Filter</td>
</tr>
</tbody>
</table>

evaluation, we adopt five-fold cross-validation for each new GP tree using SVM. We divide the GP training set randomly into five equally-sized parts and perform five repetitions of training the SVM on 4/5 of the set and testing on the remaining 1/5. The overall fitness $E_r$ is the average of the five-fold cross-validation accuracies. The corresponding fitness function is defined as follows:

$$E_r = (1 - \frac{1}{n} \sum_{i=1}^{n} (SVM[acu_i])) \times 100\%$$  \hspace{1cm} (2)
TABLE II  
PARAMETER SETTINGS FOR GENETIC PROGRAMMING RUNNING

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>300</td>
</tr>
<tr>
<td>Generation Size</td>
<td>60</td>
</tr>
<tr>
<td>Crossover Rate</td>
<td>85%</td>
</tr>
<tr>
<td>Mutation Size</td>
<td>10%</td>
</tr>
<tr>
<td>Selection for Reproduction</td>
<td>lexicotour'</td>
</tr>
<tr>
<td>Survival Method</td>
<td>'totalelitism'</td>
</tr>
<tr>
<td>Stop Condition</td>
<td>≤ 2%</td>
</tr>
</tbody>
</table>

IV. EXPERIMENTS AND RESULTS

In this section we describe the details of our GP implementation and the experimental results of our method.

A. GP Implementation

We evaluate our proposed method using 64-bit Matlab 2011a (with the genetic programming toolbox GPLAB ) on a server configured with a 6-core processor and 32G of RAM running the Linux OS. Some significant user-defined parameters for GP are listed in Table II.

For GP evolution, a lexicographic parsimony pressure has been applied as the selection method in our running. Like the original selection method, a random number of individuals are chosen from the population and further the best of them is chosen. The only difference from the original selection is that, if multiple individuals are equally fit, the shortest one (the tree with the least number of nodes) is chosen as the best. Lexicographic parsimony pressure has shown effective control for bloat in different types of problems. In addition, we have adopted the 'totalelitism' scheme as the survival module in which all the individuals from both parents and children populations are ordered by fitness alone, regardless of being parents or children. This scheme has been demonstrated to lead to promising results in many applications. We also set the GP termination of 2%, which means if the value calculated by the fitness function is equal to or lower than 2%, our GP running will be stopped and return the best-so-far individual to users.

B. Datasets

We systematically evaluated our proposed method on two dynamic hand gesture datasets, namely, the Cambridge hand gesture dataset [24] and the Northwestern University hand gesture dataset [25]. Some example frames from these two datasets are visualized in Fig. 7.

Cambridge hand gesture dataset is a commonly used benchmark gesture dataset with 900 video clips of 9 hand gesture classes defined by 3 primitive hand shapes (i.e., flat, spread, V-shape) and 3 primitive motions (i.e., Leftward, Rightward, Contract). For each class, it includes 100 sequences captured with 5 different illuminations, 10 arbitrary motions and 2 subjects. Each sequence was recorded in front of a fixed camera having coarsely isolated gestures in spatial and temporal dimensions. All video sequences are further normalized into $200 \times 200 \times 50$ in our experiments by linear interpolation. Following the experimental setting in [24], the GP training is performed on the data acquired in the single plain illumination condition, while testing is done on the data acquired in the remaining four illuminations.

Northwestern University hand gesture dataset is a more diverse dataset which contains 10 categories of dynamic hand gestures in total: move right, move left, rotate up, rotate down, move downright, move right-down, clockwise circle, counter-clockwise circle, "Z", and "cross". This dataset is performed by 15 subjects and each subject contributes 70 sequences of these ten categories with seven postures (i.e., Fist, Fingers-extended, 'Ok', Index, SideHand, SidelIndex and Thumb). Our hand gesture recognition task is similar to [25] - we just focus on recognizing the 10 dynamic gestures. Therefore the samples within the same category in different hand postures are considered as one category, and each category accordingly has 105 samples. In our experiments, we first resize all the sequences into an identical size of $240 \times 240 \times 50$ and gesture sequences from the first 8 subjects are chosen to compose the GP training set and the rest of sequences are used as the testing set. We further evaluate the performance by adopting 'leave-one-out' cross validation on the remaining 7 subjects and consider the average accuracy as the final recognition result.

C. Results

For the Cambridge hand gesture dataset, Fig. 8 shows the tree structure of the GP-evolved best-performing descriptor which finally achieves an overall accuracy of 85% on the four testing sets with different illuminations using the linear SVM classifier. For comparison, we list the results published in the original paper of the Cambridge hand gesture dataset. All these results were obtained using the same setting as ours. In addition, we also compare with prevalent hand-crafted 3D descriptors including HMHI [6], HOG/HOF [26], 3D-HOG [27] and 3D-SIFT [4]. Under the same experimental setting, we use the hierarchical motion history image (HMHI) as a holistic descriptor to extract the motion features for gesture recognition. For the other three 3D descriptors, we first divide gesture sequences with a $10 \times 10 \times 5$ grid, and describe each sub-block with one of the descriptors. Then, we concatenate the obtained features on all the sub-blocks into a vector as the final representation which is fed to the linear SVM for classification. All the relevant results are shown in Table III. It is obvious that our proposed method significantly outperforms both the state-of-the-art techniques and popular hand-crafted features. A confusion matrix for the total testing sets is plotted in Fig. 9 (a). We can see that our method can produce excellent recognition rates on gestures with flat and spread hand poses, however, the greatest confusion exists in V-shape hand pose which is hard to distinguish from the other gestures reliably.

The results on the Northwestern University hand gesture dataset are shown in Table IV with the comparison to other methods. As expected, our GP-evolved descriptor achieves a classification accuracy rate of 96.1% on the testing set.

1 http://gplab.sourceforge.net/download.html
Fig. 7. Some example frames of two datasets. Images in the left black-box are from the Cambridge hand gesture dataset and images in the right black-box are from the Northwestern University hand gesture dataset.

Fig. 8. The tree structure of the GP-generated descriptor on the Cambridge hand gesture dataset.

Fig. 9. (a) The confusion matrix of classification results on the Cambridge hand gesture dataset. (b) The confusion matrix of classification results on the Northwestern University hand gesture dataset.

Fig. 10. The LISP format of the GP-generated descriptor on the Northwestern University hand gesture dataset.

TABLE III

<table>
<thead>
<tr>
<th>Methods</th>
<th>Set1</th>
<th>Set2</th>
<th>Set3</th>
<th>Set4</th>
<th>Average accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Method</td>
<td>83</td>
<td>86</td>
<td>82</td>
<td>89</td>
<td>85</td>
</tr>
<tr>
<td>Kim et al. [24]</td>
<td>81</td>
<td>81</td>
<td>78</td>
<td>86</td>
<td>82</td>
</tr>
<tr>
<td>Niebles et al. [28]</td>
<td>70</td>
<td>57</td>
<td>68</td>
<td>71</td>
<td>67</td>
</tr>
<tr>
<td>Wong and Cipolla [29]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>HMHI</td>
<td>79</td>
<td>79</td>
<td>83</td>
<td>81</td>
<td>81</td>
</tr>
<tr>
<td>HOG/HOF</td>
<td>81</td>
<td>77</td>
<td>79</td>
<td>80</td>
<td>79</td>
</tr>
<tr>
<td>3D-HOG</td>
<td>76</td>
<td>72</td>
<td>80</td>
<td>75</td>
<td>76</td>
</tr>
<tr>
<td>3D-SIFT</td>
<td>79</td>
<td>74</td>
<td>69</td>
<td>77</td>
<td>75</td>
</tr>
</tbody>
</table>

Combining with linear SVM, since this gesture dataset is a relatively easy dataset with small intra-class variations and large inter-class variations. The result of [25] was obtained by re-implementing relevant experiments under the experimental setting as ours. Results of the popular 3D descriptors were produced in the same manner as the Cambridge dataset described above. From Table IV, we can observe that our method is comparable to Shen et al.’s and outperforms prevailing hand-
crafted descriptors. Fig. 10 shows the LISP format of the generated GP-evolved descriptor. We also illustrate the confusion matrix of the recognition results on the Northwestern University hand gesture dataset in Fig. 9 (b). We can see that our method can achieve 100% recognition rates on several gesture categories.

V. CONCLUSION

In this paper, we have proposed a domain-independent method using genetic programming (GP) to generate machine-learned descriptors for dynamic gesture recognition. Our method addresses gesture recognition as an optimization problem, and allows a computer to automatically synthesize holistic descriptors from a pool of primitive sequence processing operators without any prior knowledge. We have evaluated our method on the Cambridge hand gesture dataset and the Northwestern University hand gesture dataset and achieved accuracies of 85% and 96.1%, respectively, with the obtained best-performing descriptors evolved by GP. In both datasets, the GP-generated descriptors significantly outperform prevailing hand-crafted features.

In future work, we will investigate how to accelerate the GP evolution process, how to choose more selective 3D operators and how to evolve a general-purpose descriptor for different gesture datasets.

REFERENCES


