Genetic Programming-Evolved Spatio-Temporal Descriptor for Human-Action Recognition

Li Liu
elp11ll@sheffield.ac.uk
Ling Shao
http://lshao.staff.shef.ac.uk/
Peter Rockett
http://www.shef.ac.uk/eee/staff/p_rockett

Department of Electronic and Electrical Engineering
University of Sheffield,
Sheffield, UK

Abstract

The potential value of human action recognition has led to it becoming one of the most active research subjects in computer vision. In this paper, we propose a novel method to automatically generate low-level spatio-temporal descriptors showing good performance, for high-level human-action recognition tasks. We address this as an optimization problem using genetic programming (GP), an evolutionary method, which produces the descriptor by combining a set of primitive 3D operators. As far as we are aware, this is the first report of using GP for evolving spatio-temporal descriptors for action recognition. In our evolutionary architecture, the average cross-validation classification error calculated using the support-vector machine (SVM) classifier is used as the GP fitness function. We run GP on a mixed dataset combining the KTH and the Weizmann datasets to obtain a promising feature-descriptor solution for action recognition. To demonstrate generalizability, the best descriptor generated so far by GP has also been tested on the IXMAS dataset leading to better accuracies compared with some previous hand-crafted descriptors.

1 Introduction

Recently, human-action recognition has attracted increasing attention for a wide range of applications such as video search and retrieval, intelligent surveillance systems, and human-computer interaction.

Generally, the basic approach to action recognition contains the following main stages: (1) low-level feature extraction and representation; (2) high-level action classification. For the low-level stage, we commonly apply various techniques such as: histogram of 3D oriented gradients (3DHOG) [14], 3D scale invariant feature transforms (3DSIFT) [28], and histogram of optical flow (HOF) [20] to represent actions by extracting the most salient features (edges, corners, intensity and orientation information), the choice of which greatly influences the performance of the high-level action classification. These traditional feature representation methods, however, are usually only suitable in a given domain and often result

http://dx.doi.org/10.5244/C.26.18

in a poor performance on other applications. How to design a generalized methodology to extract spatio-temporal features for any user-defined application remains a research issue.

As an alternative to hand-crafted solutions based on deep domain knowledge, genetic programming (GP), a popular evolutionary method inspired by natural evolution, can be employed to automatically solve problems without *prior* knowledge of the solutions. In the present setting, we wish to identify the descriptor (*i.e.* the sequence of primitive operations, the composition and order of which are unknown) to maximize classification performance on the human-action recognition task. This is an NP-hard search problem which evolutionary methods may solve in a tractable amount of computer time compared to exhaustive enumerative search. GP has been used to address a wide range of practical problems producing human-competitive results and even patentable inventions. As a search framework, GP can typically escape the local minima in the optimization landscape which may trap deterministic search methods.

In this paper, we report what we believe to be the first application of genetic programming (GP) for generating a novel spatio-temporal descriptor for action recognition. Given a group of primitive 3D processing operators and a set of labelled training examples, GP evolves (hopefully) better-performing individuals in the next generation. Eventually, a best-so-far individual can be selected as the final solution. We successfully use GP to generate a new and highly-performing spatio-temporal descriptor which has been tested on a mixed KTH-Weizmann dataset, and the more challenging IXMAS dataset to demonstrate generalizability. For comparison, we also show that the proposed method is superior to some previously-published hand-crafted solutions.

The main contributions of this paper can be summarized as follows:

1. To the best of our knowledge, this is the first time that genetic programming (GP) has been successfully applied to designing descriptors in human-action recognition.

2. The proposed methodology and the generated descriptor can also be used directly in other applications.

The paper is organized as follows: In Section 2, some related work is described. The detailed architecture of our method is presented in Section 3, and relevant experiments and results are described in Section 4. In Section 5, we conclude this paper and outline possible future work.

2 Related work

Recently, evolution-based methods simulate biological evolution to automatically generate solutions for user-defined tasks. Bhanu *et al*. [1] have proposed an adaptive image segmentation system based on a genetic algorithm (GA). In their method, the GA is an effective way of searching the hyperspace of segmentation parameter combinations to determine the set which maximizes a segmentation quality criterion. Evolutionary methods have been employed to solve a range of other problems [8, 11, 12, 17].

Other evolutionary approaches have been proposed, among them genetic programming (GP) [15, 25]; GP has been widely utilized in the computer vision domain. Poli [24] has applied GP to automatically select optimal filters for segmentation of the brain in medical images. Following the same lines, Torres *et al*. [30] used GP for finding a combination of similarity functions for image retrieval. Davis *et al*. [3] have also employed GP for feature

selection in multivariate data analysis, where GP can automatically select a subset of the most discriminative features without any prior information. In addition, other researchers [6, 13, 36] have also successfully applied GP to classification tasks with improvements compared with previous methods.

Recently, GP has been exploited to assemble low-level feature detectors for high-level analysis, such as: object detection, 3D reconstruction, image tracking and matching. The first work in this area employed GP to evolve an operator for detecting interest points [4]. Trujillo and Olague [31] have also used GP to generate feature extractors for computer vision applications. In addition, a GP-based detector was proposed by Howard *et al.* [7] for detecting ship wakes in synthetic aperture radar (SAR) images. Inspired by the successful applications mentioned above, in this paper we propose evolving a spatio-temporal descriptor for human-action recognition using GP.

3 Methodology

To design a novel spatio-temporal descriptor for human action recognition using genetic programming (GP), a group of primitive 3D operators is adopted in our architecture and assembled by GP to construct a descriptor driven by maximizing its accuracy (fitness function) evaluated over a training set. The outline of our method is illustrated in Fig. 1.

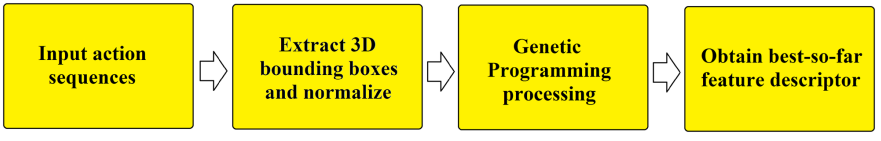


Figure 1: The main flowchart for our proposed method

3.1 Raw data pre-processing

We first coarsely extract from the original action sequences the 3D bounding boxes in which the actions are performed. The obtained bounding boxes are then all normalized to the same size of $100 \times 100 \times 70$ by using bicubic interpolation. As a result, all the action sequences are of equal sizes and all actions are approximately localized after pre-processing.

3.2 Genetic programming framework

Genetic programming (GP) [25] is one of a number of population-based evolutionary algorithms inspired by natural evolution and is widely used in machine learning. It allows a computer to automatically solve pre-defined tasks without requiring users to know or specify the form or structure of the solution in advance. In GP, we randomly generate an initial population of operation sequences which are regarded as candidate solutions. This population is then allowed to 'evolve' (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. In this way, the general fitness of the population tends to improve over time. Finally, the best performing individual obtained is taken as the final solution. It should be noted that evolutionary methods do not guarantee to find any mathematical optimum but, in practice,

usually find a 'good' solution to an NP-hard problem in an acceptable amount of computer time. The genetic programming algorithm is shown in Algorithm. 1.

Algorithm 1 Genetic Programming

Start
Initialization Randomly create an initial population of operation sequences from the available primitives (terminal set & function set).
Repeat
(1) Evaluate the fitness of each individual.
(2) Choose individuals from the population with a particular probability biased in their fitness.
(3) Create a new generation of individuals applying genetic operations.
If An acceptable solution is found or the maximum number of generations (defined by user) exceeded.
Stop
Return The best solution is selected.
End

The pre-processed action sequences form the training set and each $100 \times 100 \times 70$ block is taken, in turn, as the input to a GP individual. Each GP candidate descriptor is formulated as a tree structure, the output of which is still a $100 \times 100 \times 70$ block. A representative GP tree is illustrated in Fig. 2.

3.2.1 Function set

A key concept for GP is the function set (internal nodes of the tree) which is typically driven by the nature of the problem domain. Commonly, the choice of functions is based on the following principles:

1. The set must contain functions which can extract meaningful information.

2. To minimize the total runtime of the GP, all the operators in the function set need to be relatively simple and efficient.

3. To ensure operator *closure* [25], we have only used functions which map one or two $100 \times 100 \times 70$ 3D blocks to a single $100 \times 100 \times 70$ block. In this way, a GP tree can be an unrestricted composition of function nodes but still always produce semantically legal tree.

Note that in GP, not all functions have to be used in a given tree. Similarly, the same function can be used more than once. The topology of the tree is effectively unrestricted.

We construct our function set using 12 unary processing filters and four basic binary arithmetic functions, shown in Table 1. In our GP architecture, there are two noteworthy points:

1. For the LapPy1 and LapPy2 (Laplacian pyramid) filters, since the $100 \times 100 \times 70$ blocks from adjacent levels of the Gaussian pyramid are of different sizes, we resize them to the same dimensions before subtraction by linear interpolation.

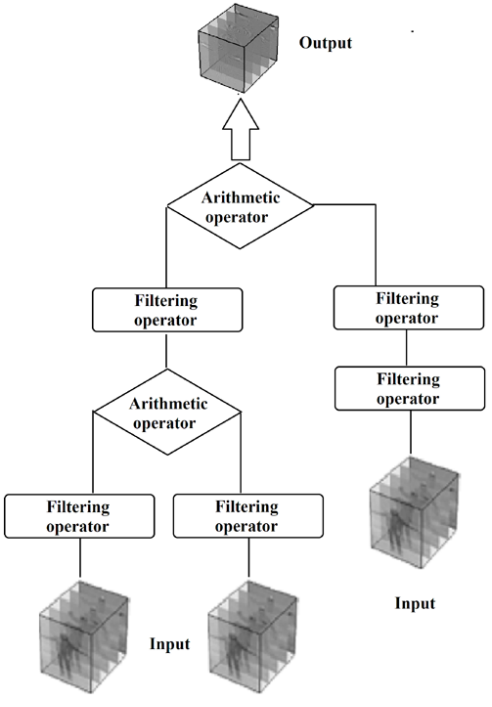


Figure 2: Illustration of a GP tree structure

2. The output of the Maxfilter operation is shrunk along the spatial and temporal dimensions relative to the input. To ensure the closure property above, we resize the outputs to a $100 \times 100 \times 70$ block.

3.2.2 Fitness function

The basis of evolutionary methods is to maximize the performance of individual solutions as gauged by some appropriate fitness function. To evaluate the candidate GP-evolved descriptors here, we estimate their classification accuracy using a linear support-vector-machine (SVM). We take the $100 \times 100 \times 70$ output of the GP tree and divide this into $10 \times 10 \times 5$ sub-blocks. The mean values of each sub-block are concatenated into a 500D vector which comprises the input of a support-vector machine (SVM), as shown in Fig. 3. To obtain a more reliable fitness evaluation, for each new GP tree we estimate the classification accuracy with the SVM using ten-fold cross-validation. We divide the GP training set randomly into ten equal parts and perform ten repetitions of training the SVM on 9/10-ths of the set and testing on the remaining tenth. The overall fitness of the candidate GP tree is taken as the average of the ten SVM test-fold accuracies.

Table 1: Function set used in genetic programming

Operator name	Inputs	Function description	Operator type
GauPy1	1	The first level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on input sequences	Filter
GauPy2	1	The second level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on sequences obtained from first level of a Gaussian pyramid	Filter
GauPy3	1	The third level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on sequences obtained from second level of a Gaussian pyramid	Filter
LapPy1	1	The first level of a Laplacian pyramid which applies a subtraction between GauPy1 and GauPy2	Filter
LapPy2	1	The second level of a Laplacian pyramid which applies a subtraction between GauPy2 and GauPy3	Filter
Wavelet1	1	The first leveHenryl of a wavelet pyramid which appliesa CDF '9/7' wavelet filter on input sequences	Filter
Wavelet2	1	The second level of a wavelet pyramid which appliesa CDF '9/7' wavelet filter once again on sequences obtained from first level of wavelet pyramid	Filter
Dof	1	Subtraction between the adjacent frames of input sequences	Filter
absDof	1	Subtraction between the adjacent frames of input sequences and then taking the absolute values of the processed sequences	Filter
Med	1	Apply median filter with filtering window size 5×5 on the input sequences	Filter
Mean	1	Apply mean filter with filtering window size 5×5 on the input sequences	Filter
Maxfilter	1	Use 3D maxpooling technique with pooling window size $4 \times 4 \times 4$ on the input sequences	Filter
Add	2	Add the input sequences	Arithmetic
Sub	2	Subtract the input sequences	Arithmetic
Mult	2	Multiply the input sequences	Arithmetic
absSub	2	Absolute subtraction of the input sequences	Arithmetic

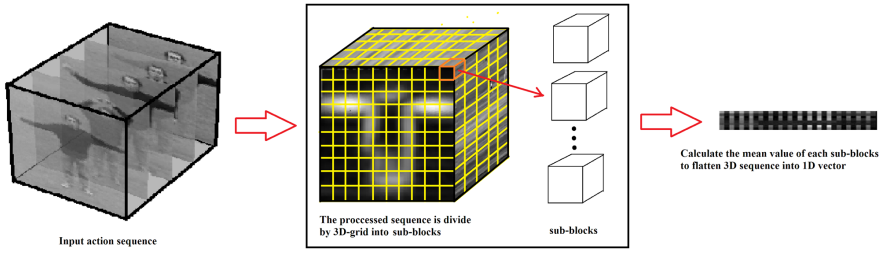


Figure 3: Spatio-temporal feature representation procedure

3.3 GP implementation

We evaluate our proposed method using Matlab 2011a (with the Genetic Programming toolbox GPLAB[1]) on a server with a six-core processor and 32GB of RAM running the Linux operating system. The total runtime was around three weeks. The user-defined GP parameters are as follows:

Training set: We combine the KTH and Weizmann datasets to create the GP training dataset which contains a total of 690 action sequences comprising 13 actions (*i.e.* boxing, handicapping, handwaving, jogging, running, walking *etc.*) performed by several people in different scenarios.

Population size: We use a population size of 100 individuals with the initial population generated with the ramped half-and-half method [25].

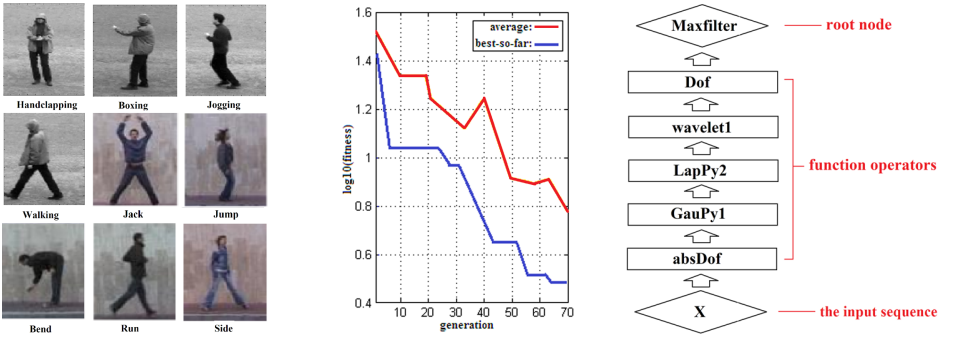
Genetic operators: We use both tree crossover and mutation [25] as our genetic operators and fix their probabilities during the GP run at 90% and 10%, respectively.

Selection for reproduction: The selection method we apply is lexicographic parsimony pressure [21] which is similar to tournament selection in choosing parents from a random sub-set of individuals in the population. However, the unique feature of lexicographic parsimony pressure is that the smallest individual (*i.e.* fewest tree nodes) will be selected if more than one individual has the same best fitness in the selection competition.

Survival method: We utilize the 'keepbest' scheme for GP. In this scheme, the best individual generated by GP is directly copied without change into the new generation. Consequently, the best-performing individual is retained from one generation to the next. This scheme has been demonstrated to lead to improved results in many different applications.

Stopping conditions: We set the GP termination criterion as the error rate falling to $\leq 2\%$ or the number of generations exceeding 70.

[1] http://gplab.sourceforge.net/download.html, A Genetic Programming Toolbox for MATLAB



(a) (b) (c)

Figure 4: (a) Some example poses from action sequences of the mixed KTH-Weizmann dataset (b) Evolved average and best-so-far values of fitness (c) Tree structure for the best-so-far descriptor.

4 Results

4.1 GP internal evaluation

To demonstrate the effectiveness and efficacy of the proposed method, we have used a mixed KTH-Weizmann dataset which combines the KTH dataset[2] [27] with the Weizmann dataset[3] [2] to evaluate the spatio-temporal descriptor generated by GP. Some sample frames from this mixed dataset are shown in Fig. 4(a). We obtain an accuracy of 96.9% with the best-so-far spatio-temporal descriptor generated by our proposed method, and shown in Fig. 4(c). This best-so-far individual is a fairly simple linear sequence of operations. Although the obtained descriptor is computer-generated, it still makes sense referring to [29] in which the authors have also used max-pooling after filtering to extract salient features. The relevant experimental results are shown in Table 2 together with comparison with other published reports. It is clear that our automatically-generated solution is comparable to Liu and Shah *et al*. [18] and superior to other methods.

4.2 Generalizability test

We further apply the best-so-far descriptor to the more challenging IXMAS dataset [34] to test the generalizability of our solution. The IXMAS dataset is a multi-view dataset composed of eleven human daily actions (*i.e.*, check watch, cross arms, scratch head, sit down, get up, turn around, talk, wave, punch, kick, pick up) performed by ten individuals and recorded from five different viewpoints—see Fig. 5(a) for some example frames.

In our experiments, we first extract bounding boxes by using the foreground masks provided with the original dataset, and normalize them to a size of $100 \times 100 \times 70$. We employ the evolved spatio-temporal descriptor to represent the corresponding IXMAS action sequences. By adopting 'leave-one-out' cross-validation, we evaluate the performance of our

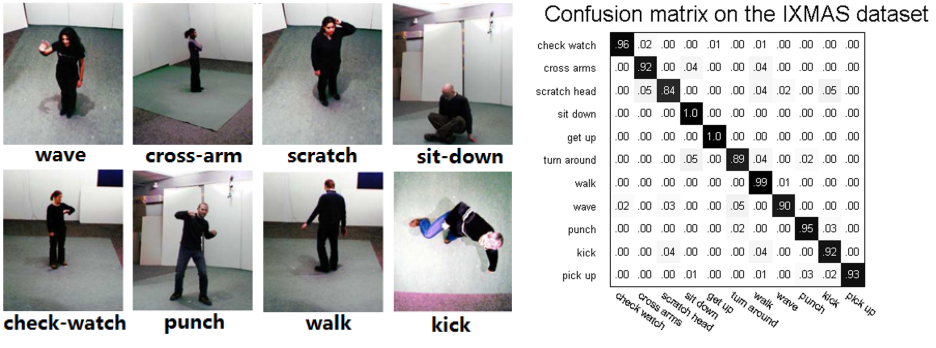
[2]The KTH dataset contains six types of human action examples (boxing, handwaving, handclapping, jogging, running and walking) performed by 25 different subjects with four scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. From http://www.nada.kth.se/cvap/actions/.

[3]The Weizmann dataset contains ten actions types (bend, jack, jump, pjump, run, side, skip, walk, wave1, wave2) performed by nine different subjects

Table 2: Action recognition accuracies (%) on the mixed KTH-Weizmann dataset, independent KTH and independent Weizmann datasets, respectively, for different methods

Methods \ Database	KTH-Weizmann	KTH	Weizmann
GP-based descriptor	96.9	93.8	100
Niebles *et al.* [23]	-	83.3	72.8
Jhuang *et al.* [9]	-	91.7	98.8
Fathi and Mori [5]	-	90.5	100
Ji *et al.* [10]	-	90.2	-
Schindler and van Gool [26]	-	92.7	-
Liu and Shah [18]	-	94.2	-

descriptor on each single-view camera as well as on the fused data from all five cameras taken together. (The fusion strategy we use is simply direct concatenation of the descriptors from all five views.) We achieve an overall accuracy of 93.6% for multi-view fusion recognition. The corresponding confusion matrix for the fused camera results is shown in Fig. 5(b). Table 3 shows a comparison between our method and some previous work from which we can observe that our method outperforms previously published results on the IXMAS dataset, in some cases by a significant margin.



(a) (b)

Figure 5: Results on the IXMAS dataset:(a) Some representative frames of action sequences (b) The confusion matrix of the multiple camera fusion result

5 Conclusions

In this paper, we have proposed a GP-based method to generate a spatio-temporal descriptor for human-action recognition. GP can automatically evolve a descriptor from a pool of primitive operators. Our proposed method does not require any prior knowledge of the structure of the descriptor. We have evaluated our method on the KTH-Weizmann dataset and achieved an accuracy of 96.9% for action recognition. In addition, the best-so-far descriptor has also been applied to the more challenging IXMAS dataset to demonstrate the

Table 3: Comparison of action recognition accuracies (%) on the IXMAS dataset for different methods

Methods \ Actions	Cam 1	Cam 2	Cam 3	Cam 4	Cam 5	Cam 1-5 fusion
GP-based descriptor	86.5	89.3	88.1	84.7	78.3	93.6
Varma and Babu [32]	76.4	74.5	73.6	71.8	60.4	81.3
Liu and Shah [18]	76.7	73.3	72.1	73.1	-	82.8
Wu *et al*. [35]	81.9	80.1	77.1	77.6	73.4	88.2
Weinland *et al*. [33]	-	-	-	-	-	93.3

generalizability of our method; we obtain a 93.6% recognition rate.

In future work, we will attempt to accelerate the GP evolution process. The reliability and flexibility of our algorithm will also be evaluated on more challenging datasets such as the HMDB51 [16], the YouTube [19] and the Hollywood [22] datasets.

References

[1] B. Bhanu, S. Lee, and J. Ming. Adaptive image segmentation using a genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics*, 25(12):1543–1567, 1995.

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *International Conference on Computer Vision*, volume 2, pages 1395–1402, Beijing, China, 2005.

[3] R.A. Davis, A.J. Charlton, S. Oehlschlager, and J.C. Wilson. Novel feature selection method for genetic programming using metabolomic HNMR data. *Chemometrics and Intelligent Laboratory Systems*, 81(1):50–59, 2006.

[4] M. Ebner and A. Zell. Evolving a task specific image operator. *Evolutionary Image Analysis, Signal Processing and Telecommunications*, 14(7):74–89, 1999.

[5] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, 2008.

[6] H. Guo, L.B. Jack, and A.K. Nandi. Feature generation using genetic programming with application to fault classification. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 35(1):89–99, 2005.

[7] D. Howard, S.C. Roberts, and R. Brankin. Target detection in SAR imagery by genetic programming. *Advances in Engineering Software*, 30(5):303–311, 1999.

[8] P. Hudak. Conception, evolution, and application of functional programming languages. *ACM Computing Surveys (CSUR)*, 21(3):359–411, 1989.

[9] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, 2007.

[10] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. In *International Conference on Machine Learning*, Haifa, Israel, 2010.

[11] N. Karaboga. A new design method based on artificial bee colony algorithm for digital iir filters. *Journal of the Franklin Institute*, 346(4):328–348, 2009.

[12] Y.J. Kim, J.H. Kim, and D.S. Kwon. Evolutionary programming-based univector field navigation method for past mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 31(3):450–458, 2001.

[13] J.K. Kishore, L.M. Patnaik, V. Mani, and V.K. Agrawal. Application of genetic programming for multicategory pattern classification. *Evolutionary Computation, IEEE Transactions on*, 4(3):242–258, 2000.

[14] A. Klaser and M. Marszalek. A spatio-temporal descriptor based on 3d-gradients. In *19th British Machine Vision Conference*, pages 995–1004, Leeds,British, 2008.

[15] J. Koza and R. Poli. Genetic programming. *Search Methodologies*, pages 127–164, 2005.

[16] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *13th IEEE International Conference on Computer Vision*, pages 2556 – 2563, Barcelona, Spain, 2011.

[17] S.M. Lee, Y. Jun, S.N. Cho, and J. Cheon. Single-crystalline star-shaped nanocrystals and their evolution: programming the geometry of nano-building blocks. *Journal of the American Chemical Society*, 124(38):11244–11245, 2002.

[18] J. Liu and M. Shah. Learning human actions via information maximization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, 2008.

[19] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos 'in the wild'. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003, Miami, FL, USA, 2009.

[20] B.D. Lukas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop*, 1981.

[21] S. Luke and L. Panait. Lexicographic parsimony pressure. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 829–836, New York, USA, 2002.

[22] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2929–2936, Miami, FL, USA, 2009.

[23] J.C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3): 299–318, 2008.

[24] R. Poli. Genetic programming for image analysis. In *Proceedings of the First Annual Conference on Genetic Programming*, pages 363–368, Stanford, California, USA, 1996.

[25] R. Poli, W.B. Langdon, and N.F. McPhee. *A field guide to genetic programming*. 2008.

[26] K. Schindler and L. Van Gool. Action snippets: How many frames does human action recognition require? In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, 2008.

[27] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In *International Conference on Pattern Recognition*, volume 3, pages 32–36, Cambridge, England, UK, 2004.

[28] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional SIFT descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, pages 357–360, Augsburg, Bavaria, Germany, 2007.

[29] D. Song and D. Tao. Biologically inspired feature manifold for scene classification. *IEEE Transactions on Image Processing*, 19(1):174–184, 2010.

[30] R.S. Torres, A.X. Falcão, M.A. Gonçalves, J.P. Papa, B. Zhang, W. Fan, and E.A. Fox. A genetic programming framework for content-based image retrieval. *Pattern Recognition*, 42(2):283–292, 2009.

[31] L. Trujillo and G. Olague. Synthesis of interest point detectors through genetic programming. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pages 887–894, seattle, Washington, 2006.

[32] M. Varma and B.R. Babu. More generality in efficient multiple kernel learning. In *the 26th Annual International Conference on Machine Learning*, pages 1065–1072, NY, USA, 2009.

[33] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006.

[34] D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3d exemplars. In *IEEE 11th International Conference on Computer Vision*, pages 1–7, Rio de Janeiro, Brazil, 2007.

[35] X. Wu, L. Xu, D.and Duan, and J. Luo. Action recognition using context and appearance distribution features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 489–496, Providence, RI, 2011.

[36] M. Zhang and W. Smart. Multiclass object classification using genetic programming. *Applications of Evolutionary Computing*, 14(3):369–378, 2004.