Implicit Context Representation Cartesian Genetic Programming for the Assessment of Visuo-spatial Ability

Stephen L. Smith and Michael A. Lones, *Member, IEEE*

***Abstract*— In this paper, a revised form of Implicit Context Representation Cartesian Genetic Programming is used in the development of a diagnostic tool for the assessment of patients with neurological dysfunction such as Alzheimer's disease. Specifically, visuo-spatial ability is assessed by analysing subjects' digitised responses to a simple figure copying task using a conventional test environment. The algorithm was trained to distinguish between classes of visuo-spatial ability based on responses to the figure copying test by 7–11 year old children in which visuo-spatial ability is at varying stages of maturity. Results from receiver operating characteristic (ROC) analysis are presented for the training and subsequent testing of the algorithm and demonstrate this technique has the potential to form the basis of an objective assessment of visuo-spatial ability.**

I. Introduction

Visuo-spatial ability can be defined as a person's manipulation of "visual representations and their spatial relationships" [5] and is used in many every day activities such as parking a car, reading a map or pouring a drink. A deficit of visuo-spatial ability is observed in many neuropsychological conditions, such as stroke, Parkinson's disease and Alzheimer's disease, and consequently is an important symptom. However, conventional measurement of visuo-spatial ability can be time consuming and is often subjective. The aim of the work described in this paper is to develop an automated, objective assessment of visuo-spatial ability that can be made easily and reliably. Section II considers the conventional evaluation of visuo-spatial ability using a traditional test environment based on figure copying. Section III gives an overview of Implicit Context Representation Cartesian Genetic Programming and a revised development process used for this research. Section IV describes how this algorithm is used to evolve classifiers which automatically classify subjects' drawings. Section V presents results. Conclusions and discussion are presented in Section VI.

II. Evaluation of visuo-spatial ability

Early diagnosis of any neurological dysfunction is highly desirable as it may permit appropriate therapies or treatment to slow the progression of the disease and minimise its symptoms. However, absolute diagnosis of many neurological conditions is only possible by examining brain tissue and is therefore impractical whilst the patient is alive. Due to this difficulty, determining the presence of a neurological disease is most often a diagnosis of exclusion, where the physician will try to find other causes of the symptoms, often by using laboratory tests and neuroimaging techniques.

An important part of the diagnosis and monitoring of the disease is to perform a neurological examination to evaluate the extent of the impairment of the patient. For example, the most common method of diagnosis based on these examinations for Alzheimer's disease (AD) is the NINCDS-ADRDA Alzheimer's Criteria [13] which examines eight cognitive domains: memory, language, perception, attention, constructive ability, orientation, problem solving and functional ability. Problems within these domains could suggest the onset of AD and the criteria leads to four possible outcomes: definite, probable, possible and unlikely AD.

Geometric shape drawing tasks are often used to evaluate visuo-spatial neglect. Several tests have been developed such as the Clock Drawing Test, the Rey-Osterrieth Complex Figure Test and cube drawing tests. Research into cube drawing ability has not only shown that it is a useful tool in the detection of AD but that it is also good at the detection of very mild AD [16]. For cube drawing assessments detailed marking criteria are used to grade the cube and hence determine the level of impairment. One example of such a criteria is presented in [1] which is used to mark the development of cube drawing ability of 7 to 10 year olds and shows many similarities with the criteria used in [16] to mark drawings of elderly and AD patients. The scoring system taken from [1] is as follows:

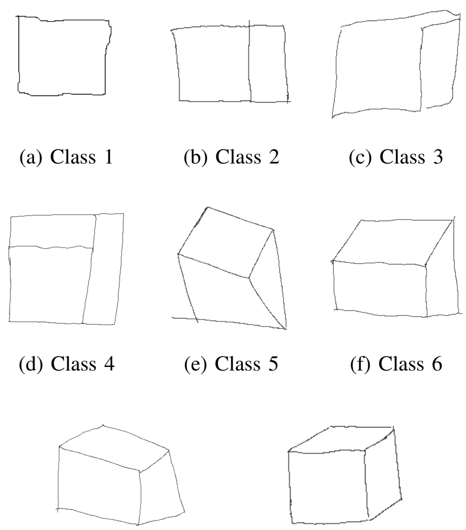
1) A single square or rectangle of any orientation.
2) A set of interconnected squares or rectangles numbering more or less than the number of visible faces in the cube (three) or single trapezoid with some appropriate use of oblique lines.
3) A set of three interconnected squares or rectangles not appropriately arranged to represent the visible arrangement of faces in the cube or a set of interconnected squared or rectangles numbering more or less than the number of visible faces in the cube including some appropriate use of oblique lines.
4) A set of three interconnected squares or rectangles appropriately arranged to represent the visible arrangement of the faces of the cube or an inappropriately arranged set of three outlines including some appropriate use of oblique lines.
5) Drawings that show only visible faces of the cube appropriately arranged (as noted previously) and that reveal crude attempts to show depth through use of oblique lines, curvature, or modification to angles.
6) Drawings that approximate to oblique projection or

Both authors are with the Intelligent Systems Research Group, Department of Electronics, University of York, Heslington, York, YO10 5DD, UK. Email: {sls5,mal503}@ohm.york.ac.uk. Web: http://www.elec.york.ac.uk/research/intSys.html.

linear perspective or drawings that approximate well to oblique projection or linear perspective but that are drawn to a horizontal ground line rather than to an oblique ground plane.

7) Drawings that are close approximations to oblique projection or linear perspective but that contain some inaccuracies in angular relations between lines.
8) Accurate portrayals of a cube in oblique projection or linear perspective.

Figure 1 shows eight example drawings which have been classified based on this system.



(a) Class 1 (b) Class 2 (c) Class 3

(d) Class 4 (e) Class 5 (f) Class 6

(g) Class 7 (h) Class 8

Fig. 1: Eight classifications of cube drawings using the marking system described by Bremner et al. [1].

Application of the assessment criteria by trained assessors can vary and, hence, is arguably unreliable, so it is desirable to produce an assessment mechanism which will be able to classify cube drawings in a completely objective way. Guest and Fairhurst [8] implement an algorithm to extract components from static hand-drawn responses for two figure copying tests and one figure completion test. First the image is 'skeletonised' then split it into its horizontal, vertical and diagonal components by using directional neighbourhood identification. The components are then assessed based on certain features, such as component omissions, length difference and spatial differences in order to examine the divergence between neglect and control responses. In [7] they extend this idea to include the analysis of dynamic performance features such as pen lifts, movement time and drawing time to improve the sensitivity of the assessment. By looking at these dynamic features they conclude that they can gain an additional understanding of the condition. However, the algorithms described in [7] and [8] use rigid sets of rules designed by the authors based on observed differences. This paper proposes a method in which this level of subjectivity is removed by using an evolutionary algorithm, Implicit Context Representation Cartesian Genetic Programming, to identify the features used for classifying subject responses.

III. Implicit Context Representation CGP

Implicit Context Representation Cartesian Genetic Programming (IRCGP) [17] is a form of Cartesian Genetic Programming (CGP) [15] which uses an implicit context representation [10, 11, 12].

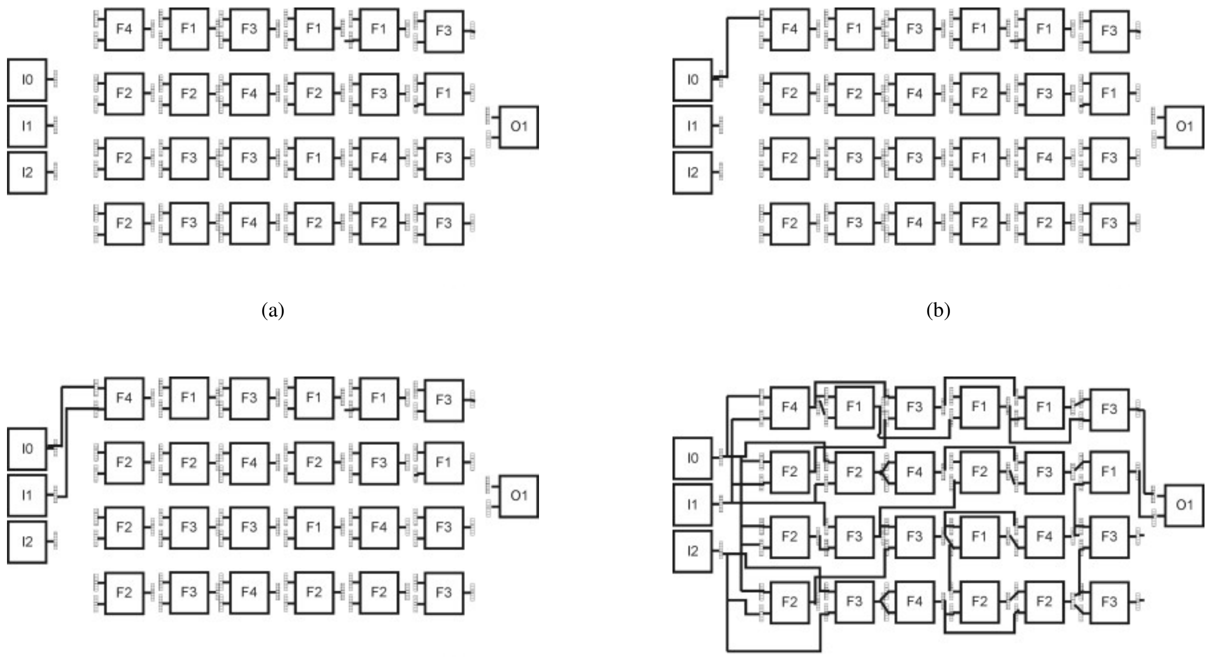
CGP is a graph-based genetic programming system which has been shown to perform well within a wide range of problem domains. A CGP solution consists of an n-dimensional grid (where n is typically 1 or 2) in which each grid location contains a function. Program inputs and outputs are delivered to and taken from specific grid cells. Interconnections between functions, inputs and outputs are expressed in terms of the grid's Cartesian co-ordinate system. Variation operators (mutation and crossover) are able to alter both the function present within a grid cell and the connections between components.

The efficacy of CGP has been attributed to both implicit reuse of sub-expressions (due to its graphical representation) and its use of functional redundancy [14]. However, CGP programs are positionally dependent, since the behavioural context of a function (i.e. where it receives its inputs from and sends its output to) is dependent upon its Cartesian co-ordinate within the program's representation. Positional dependence, in turn, causes disruptive behaviour during recombination. A consequence of this is that CGP programs, in common with standard GP programs, do not generally respond well to crossover operators [2] (except where the operator is a good match to the problem [4]).

Implicit context is a means of introducing positional independence to GP solution representations. The principle behind implicit context is that interconnections between solution components (outputs, functions and input terminals) are specified in terms of each component's *functional context within the solution* rather than its physical location within the solution. Consequently, when a component's location changes following crossover (or mutation), its expected behaviour will not change. Implicit context representation was originally developed for the Enzyme GP system [12].

In standard CGP, a function's inputs are specified by Cartesian grid references. In IRCGP, a function's inputs are specified by *functionality profiles* which are then resolved to Cartesian grid references during a simple constraint satisfaction development process. Formally, a functionality profile is a vector in an n-dimensional space where each dimension corresponds to a function or terminal. This vector describes the relative occurrence of each function and terminal, weighted by depth, within an expression. In effect, it provides a means of representing and comparing (through vector difference) the functional behaviour of an expression. Details of how functionality profiles are constructed in IRCGP can be found in Smith et al. [17].

The functionality profile(s) associated with a function component describes the sub-expression(s) from which it would prefer to receive its input(s). Prior to evaluation, an



(a) (b) (c) (d)

Fig. 2: Bottom-up development process for satisfying functionality profiles.

IRCGP solution undergoes a development process in which each function and output component attempts to find the sub-expression(s) which most closely match its functionality profile(s). In previous work, this has been achieved using the top-down development process of Enzyme GP [11]. However, the grid-structure and feed-forward nature of CGP means that this can also be achieved using a bottom-up process — which, in turn, leads to more accurate matching between functionality profiles and the sub-expressions which they connect to.

The bottom-up IRCGP development process is illustrated in Figure 2. As in standard CGP, function components (Fn) are ordered from the top left to the bottom right of the grid (in column then row order). To prevent recurrent connections, a component may only receive inputs from the outputs of components lower in the ordering. Also in common with standard CGP, connections are subject to a levels back constraint — meaning that a component may only receive inputs from one located within a specified number of preceding columns.

Starting with an initially unconnected grid (Fig. 2a), development begins at the first input of the first function component (Fig. 2b). Since there are no other function components before this component, it may only choose from the program's inputs (labelled I0–I2) — and will choose whichever one which most closely matches its functionality profile (I0 in this case). The development process will then move on to the function's other input(s) (Fig. 2c) and then up through the other function components in the network until all component inputs have been satisfied (Fig. 2d).

IV. Methodology

In previous work on automated Parkinson's diagnosis [18], it was shown that evolved IRCGP solutions are able to describe acceleration patterns which are over-represented in the movements of Parkinson's patients relative to control subjects. In this work, we extend the approach to the multi-class cube drawing classification task described in Section II. In particular, we hypothesise that the development or degradation of visuo-spatial ability is reflected in the physical movements of subjects when carrying out the cube drawing task and, furthermore, that these patterns of movement (if adequately described) can be used as a basis for automated classification.

A. Data Collection

Drawings made by children can be easily digitised by using a commercial digitising tablet. Use of an inking, wireless pen enables a traditional pen and paper environment to be preserved, reducing stress and distraction in the participants. Modern digitising tablets can sample pen movements up to 200 times per second at a spatial resolution of up to 5000 lines per inch, enabling very fine reproduction of the drawings made.

Drawings were taken from children ranging from 7 to 11 years attending a conventional state school (having obtained local ethical approval and informed consent). Each child was asked to make several attempts at drawing a copy of a cube. Once the data was collected the cubes were manually classified by two independent markers using the scheme of Bremner et al. [1] (see Section II). No drawings were identified as class 1.

In total, 120 drawings were recorded from 40 subjects (each providing 2–4 drawings). The position of the pen, both whilst drawing and whilst lifted from the tablet, was recorded using a Wacom Intuos3 pen tablet at a sampling rate of 200Hz. This position data was then converted to an acceleration sequence using discrete differentiation, truncated to one standard deviation around the mean (to remove skew) and smoothed using a moving average filter of size 2 (to reduce noise). In order to normalise with respect to drawing time, the resulting sequences were scaled (using linear interpolation if necessary) to a standard length of 4000. Acceleration values were then quantised to the integer range $[-10, 10]$. This quantisation is intended to remove minor fluctuations from the acceleration sequences, since classifiers based on minor fluctuations are likely to be less meaningful and more fragile than those based upon gross acceleration features.

Based on the manual classification of the drawings, the corresponding acceleration sequences were divided into training and test sets, maintaining a ratio of approximately $2:1$ both overall and within each class. In order to prevent possible bias, multiple drawings from an individual subject were not split between training and test sets.

B. Evaluation

An acceleration sequence is presented to an evolved IR-CGP solution as a sequence of overlapping data windows of length 30. For each of these windows, the IRCGP solution calculates an output. Any value less than zero is interpreted as a positive match. The classification of an acceleration sequence is given by the number of positive matches over all windows.

Receiver Operating Characteristic (ROC) analysis is used to measure an evolved classifier's fitness — its ability to discriminate between data classes. A ROC curve plots true positive rate (TPR) against false positive rate (FPR) across the range of possible classification thresholds, where:

$$TPR = \frac{\text{Number of positive examples correctly classified}}{\text{Number of positive examples}}$$

$$FPR = \frac{\text{Number of negative examples correctly classified}}{\text{Number of negative examples}}$$

The area under a ROC curve (known as AUC) is often used as a measure of classifier accuracy, since it is equivalent to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example [6]. AUC scores fall within the range $[0, 1]$, where 1 indicates perfect discrimination between positive and negative data sets, 0.5 indicates no ability to discriminate, and 0 indicates that negative data is always ranked higher than positive data (i.e. perfect classification can be achieved by inverting the classifier's output).

The AUC metric can be extended to multi-class classifiers by taking the mean of the AUCs of each pair of classes [9], thus measuring the overall pairwise discriminability of the classifier — in effect, how well the classifier separates the classes within its output range. Hand and Till [9] define this metric as:

$$AUC_{\text{multiclass}} = \frac{2}{|C|(|C|-1)} \sum_{\{c_i, c_j\} \in C} AUC(c_i, c_j) \quad (1)$$

where C is the set of classes and $AUC(c_i, c_j)$ is the area under the ROC curve when separating classes c_i and c_j. In this work, we do not require that the classifier ranks the classes in their original numerical order, only that it separates the classes within the output range. This is done by inversely mapping pairwise AUC scores in the range $[0, 0.5)$ to the range $(0.5, 1.0]$.

C. Parameter Settings

We carried out 5 runs of 200 generations using a population of 200 classifiers. Child solutions were generated using uniform crossover and mutation in equal proportion. The mutation rate was 6% for functions and 3% for functionality dimensions. We used a CGP grid size of 9 rows by 8 columns. These values were determined experimentally. The function set is defined in Table I.

TABLE I: Function Set

Function	Symbol	Description
Add	+	Returns the sum of its two inputs
Subtract	−	Returns the difference of its two inputs
Mean	M	Returns the mean of its two inputs
Min	$<$	Returns the lesser of its two inputs
Max	$>$	Returns the greater of its two inputs
Absolute	$\|\|$	Returns the absolute value of its input
Negate	!	Returns its input multiplied by -1

TABLE II: Pairwise AUC scores for best evolved classifier upon training and test sets. High scores ($0.2 \geqslant AUC \geqslant 0.8$) are shown in bold face. The last column indicates whether training and test scores are correlated.

Pair	Train AUC	Test AUC	Correlated?
2/3	**0.83**	**1.00**	yes
2/4	0.74	**0.80**	yes
2/5	**0.05**	**1.00**	
2/6	0.66	**0.90**	yes
2/7	**0.94**	**1.00**	yes
2/8	**0.89**	**0.90**	yes
3/4	0.69	0.60	yes
3/5	**0.00**	**1.00**	
3/6	0.39	**0.80**	
3/7	**0.87**	**1.00**	yes
3/8	**0.85**	**0.80**	yes
4/5	**0.10**	**0.80**	
4/6	0.28	0.68	
4/7	0.57	0.70	yes
4/8	0.46	0.64	
5/6	**0.86**	0.38	
5/7	**0.99**	0.21	
5/8	**0.96**	**0.13**	
6/7	**0.85**	0.44	
6/8	0.78	0.33	
7/8	0.33	0.45	yes

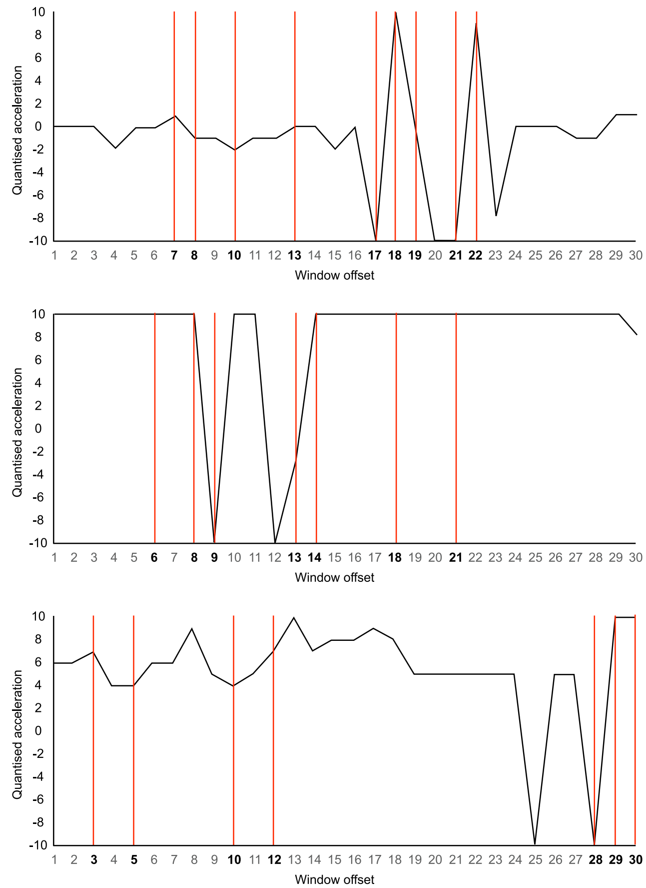


Fig. 3: Best matching acceleration sequence windows against (top) the best and two other high scoring classifiers. Vertical red lines show the window offsets used by the evolved expression.

V. Results

The best evolved classifier had a multi-class AUC score of 0.70 on both the training and test sets. Table II shows its AUC scores for each pair of classes. Of the 21 pairs of classes, 12 have high AUC scores for both the training and test sets. However, in some cases these do not correlate across the training and test sets, which may indicate over-learning. Nevertheless, the lower-numbered classes are fairly well separated from the higher-numbered classes, suggesting that the evolutionary algorithm has found a meaningful pattern.

Figure 3 shows the best matching sequence windows against three of the highest scoring classifiers. A prominent feature in these patterns is the presence of a dual deceleration peak. This can also be seen in Figure 4 which overlays all the sequence windows which were matched positively by the best of these classifiers. Whilst there is a fair amount of variance between the matches, most exhibit a dual peak and a region of relatively constant acceleration either side. This is especially the case for the stronger matches (i.e. those which receive higher classification scores), indicated by thicker lines in the diagram.

Figure 5 shows the locations of these match windows within four example drawings from different classes. It can be seen that whilst matches occur throughout the drawing in the lower-numbered classes, they tend to cluster around corners in the higher-numbered classes. This leads to the hypothesis that the evolved expression is recognising regions in which the subject hesitates or carries out jittery motion. In the case of lower-numbered classes, this may reflect the subject's general unfamiliarity with the shape they are attempting to draw; whereas in higher-numbered classes, uneven motion only occurs around changes in direction.

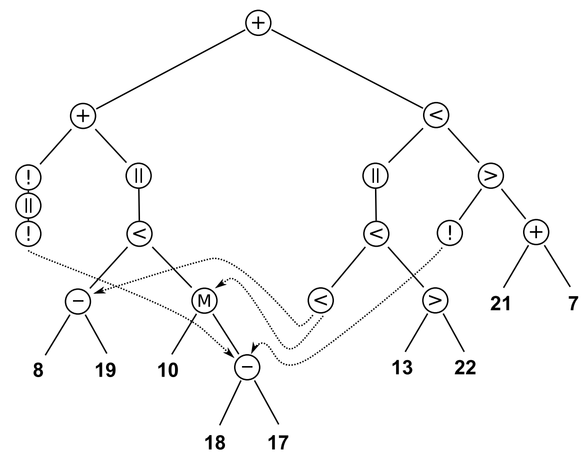


Fig. 6: Best evolved classifier. Dotted lines indicate implicit reuse of sub-expressions within the IRCGP network. Numbers indicate offsets in the matching window.

Figure 6 shows the pattern matching expression used by the best classifier. Whilst it is difficult to interpret the exact behaviour of the classifier from this expression, it can be seen that the sub-expression '$18 - 17$' is re-used four times when calculating a match. Offsets 17 and 18 correspond to the first peak in Figure 4, suggesting that the rate of change of acceleration at this point is an important feature underlying classification. It is also notable that the classifier considers multiple pairs of offsets (7/21, 8/19, 13/22, and 10 with 17/18), where one occurs in the region preceding the acceleration peaks and the other occurs within the peaks. This strengthens the idea that the classifier is looking for two different features, i.e. a region of relatively constant acceleration followed by acceleration peaks.

The different separations between the pairs of offsets (14, 11, 9 and 7-8, respectively) also suggests that the classifier is looking for the features at different time scales within the match window. The use of fixed offsets is one of the limitations of the current classifier model, since it requires relatively complex expressions in order to describe a feature occurring at different time scales. Whilst normalisation of sequence length mitigates this effect at a gross level, it is still likely that there will be timing variations between the responses of different subjects within a class. In future work, we plan to address this issue by looking at the utility of feature-based encodings, such as time domain signal coding [3], which are less sensitive to scale.

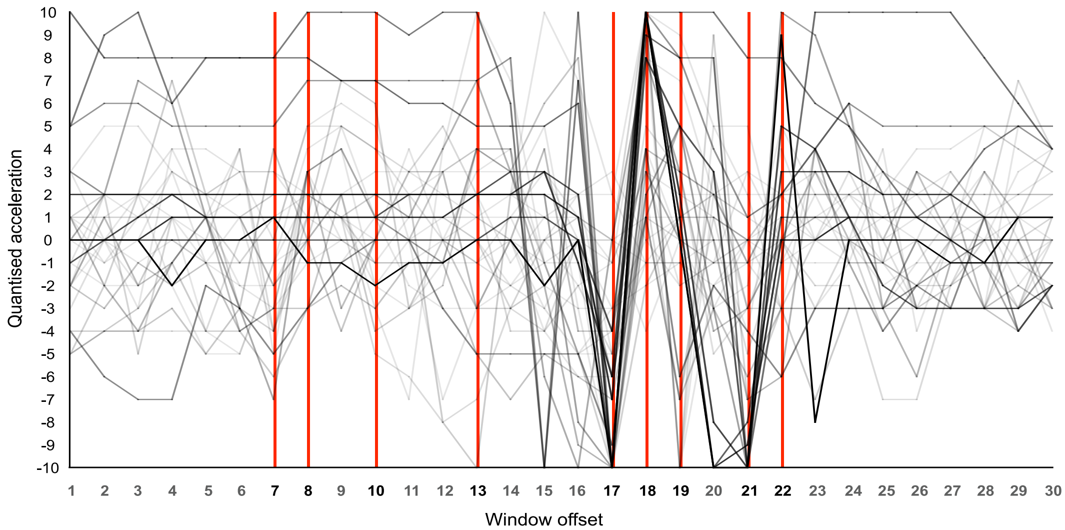


Fig. 4: Overlay of all positive matches to the best evolved classifier across all sequences. The weight of a line indicates the strength of the corresponding match. Vertical red lines show the window offsets used by the evolved expression.

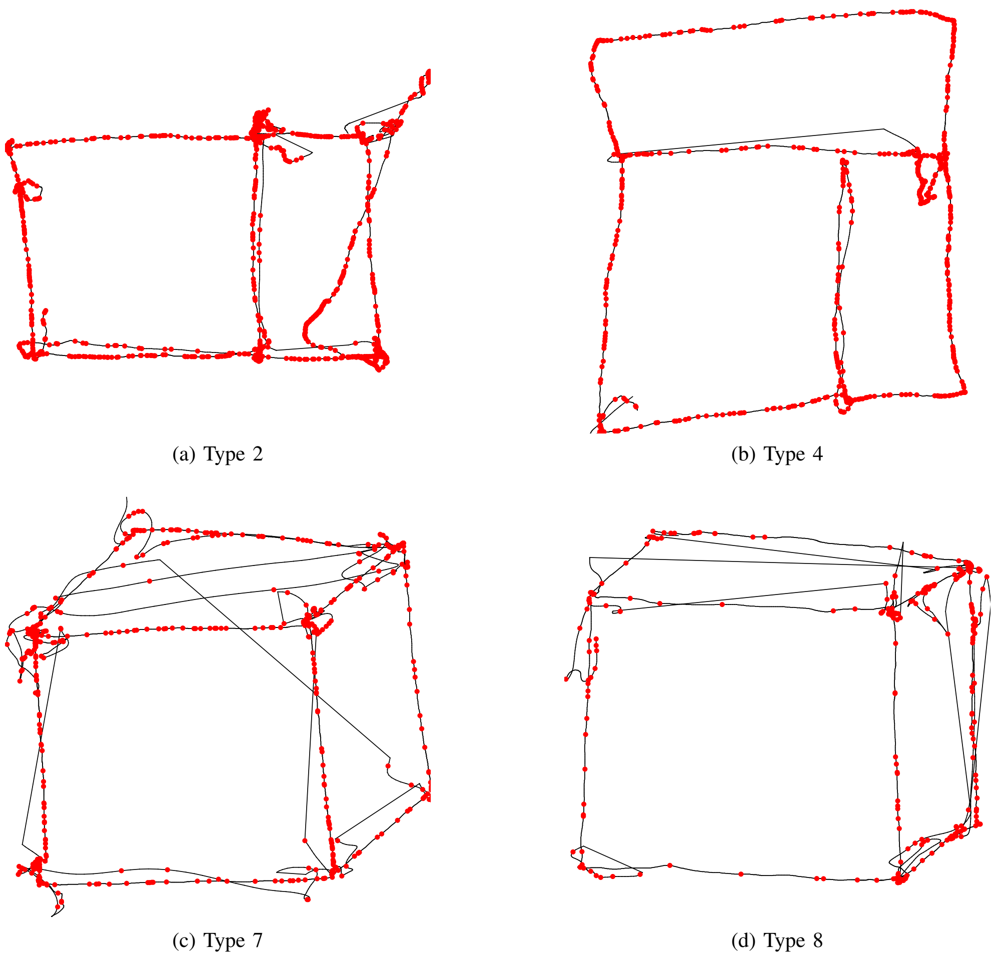


Fig. 5: Examples of drawings from four different type classes, showing match locations against the best evolved classifier. Red circles indicate the start of positively matched windows within the drawing's corresponding acceleration sequence.

VI. Conclusions

In this paper, we have demonstrated how Implicit Context Representation Cartesian Genetic Programming can be used to identify meaningful patterns of subject movement within recordings of children carrying out cube drawing tasks. Our results suggest that the resulting classifiers can be used to categorise a child's relative stage of neurological development. In future work, we hope to apply this technique to the automated diagnosis of neurological diseases such as Parkinson's and Alzheimer's.

In addition, this work has shown how Implicit Context Representation Cartesian Genetic Programming can be applied to a multi-class classification problem by using a multi-class ROC analysis based fitness function. We have also introduced a bottom-up development process to the algorithm. Our practical experience suggests that this improves the performance of the method, and in future work, we hope to expose it to a more in-depth analysis.

Acknowledgements

The authors would like to express their gratitude to the staff and pupils of Dringhouses Primary School, York, for participating in this study, and to Ricky Ledwith for administering the data acquisition process.

References

[1] J. G. Bremner, R. Morse, S. Hughes, and G. Andreasen. Relations between drawing cubes and copying line diagrams of cubes in 7- to 10-year-old children. *Child Dev*, 71(3):621–634, 2000.

[2] X. Cai, S. L. Smith, and A. M. Tyrrell. Positional independence and recombination in cartesian genetic programming. In *Proc. 2006 European Conference on Genetic Programming (EuroGP)*, pages 351–360, 2006.

[3] E. D. Chesmore. Application of time domain signal coding and artificial neural networks to passive acoustical identification of animals. *Journal of Applied Acoustics*, 62:1359–1374, 2001.

[4] J. Clegg, J. A. Walker, and J. F. Miller. A new crossover technique for cartesian genetic programming. In *Genetic and Evolutionary Computation Conference (GECCO 2007)*, 2007.

[5] W. Dorland, editor. *Dorland's Medical Dictionary for Health Consumers*. Saunders, an imprint of Elsevier, 2007.

[6] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[7] R. Guest and M. Fairhurst. A novel multi-stage approach to the detection of visuo-spatial neglect based on the analysis of figure-copying tasks. In *Proceedings of the fifth international ACM conference on Assistive technologies*, page 157 161, 2002.

[8] R. Guest, M. Fairhurst, and J. Potter. Automated extraction of image segments from clinically diagnostic hand-drawn geometric shapes. In *Proceedings of the 26th Euromicro Conference*, volume 2, page 440 446, 2000.

[9] D. J. Hand and R. J. Till. A simple generalization of the area under the ROC curve to multiple class classification problems. *Machine Learning*, 45(2):171–186, 2001.

[10] M. A. Lones. *Enzyme Genetic Programming: Modelling Biological Evolvability in Genetic Programming*. PhD thesis, Department of Electronics, University of York, 2003.

[11] M. A. Lones and A. M. Tyrrell. Biomimetic representation with enzyme genetic programming. *Genetic Programming and Evolvable Machines*, 3(2):193–217, June 2002.

[12] M. A. Lones and Andy M. Tyrrell. Modelling biological evolvability: Implicit context and variation filtering in enzyme genetic programming. *BioSystems*, 76(1–3): 229–238, August–October 2004.

[13] G. McKhann, D. Drachman, M. Folstein, R. Katzman, D. Price, and E. M. Stadlan. Clinical diagnosis of Alzheimer's disease: report of the NINCDS-ADRDA Work Group under the auspices of Department of Health and Human Services Task Force on Alzheimer's Disease. *Neurology*, 34:93944, 1984.

[14] J. F. Miller and S. L. Smith. Redundancy and computational efficiency in cartesian genetic programming. *IEEE Transactions on Evolutionary Computation*, 10 (2):167–174, 2006.

[15] J. F. Miller and P. Thomson. Cartesian genetic programming. In R. Poli, W. Banzhaf, W. B. Langdon, J. F. Miller, P. Nordin, and T. C. Fogarty, editors, *Third European Conf. Genetic Programming*, volume 1802 of *Lecture Notes in Computer Science*, 2000.

[16] Y. Shimada, K. Meguro, M. Kasai, M. Shimada, H. Ishii, S. Yamaguchi, and A. Yamadori. Necker cube copying ability in normal elderly and Alzheimer's disease. A community-based study: The Tajiri project. *Psychogeriatrics*, 6(1):4–9, 2006.

[17] S. L. Smith, S. Leggett, and A. M. Tyrrell. An implicit context representation for evolving image processing filters. In *Proceedings of the 7th Workshop on Evolutionary Computation in Image Analysis and Signal Processing*, volume 3449 of *Lecture Notes in Computer Science*, pages 407–416, 2005.

[18] S. L. Smith, P. Gaughan, D. M. Halliday, Q. Ju, N. M. Aly, and J. R. Playfer. Diagnosis of Parkinson's disease using evolutionary algorithms. *Genetic Programming and Evolvable Machines*, 8(4):433–447, 2007.