Context-Dependent Predictions and Cognitive Arm Control with XCSF

Martin V. Butz
Department of Psychology
Röntgenring 11
97070 Würzburg, Germany
butz@psychologie.uni-wuerzburg.de

Oliver Herbort
Department of Psychology
Röntgenring 11
97070 Würzburg, Germany
oliver.herbort@psychologie.uni-wuerzburg.de

ABSTRACT

While John Holland has always envisioned learning classifier systems (LCSs) as cognitive systems, most work on LCSs has focused on classification, datamining, and function approximation. In this paper, we show that the XCSF classifier system can be very suitably modified to control a robot system with redundant degrees of freedom, such as a robot arm. Inspired by recent research insights that suggest that sensorimotor codes are nearly ubiquitous in the brain and an essential ingredient for cognition in general, the XCSF system is modified to learn classifiers that encode piecewise linear sensorimotor structures, which are conditioned on prediction-relevant contextual input. In the investigated robot arm problem, we show that XCSF partitions the (contextual) posture space of the arm in such a way that accurate hand movements can be predicted given particular motor commands. Furthermore, we show that the inversion of the sensorimotor predictive structures enables accurate goal-directed closed-loop control of arm reaching movements. Besides the robot arm application, we also investigate performance of the modified XCSF system on a set of artificial functions. All results point out that XCSF is a useful tool to evolve problem space partitions that are maximally effective for the encoding of sensorimotor dependencies. A final discussion elaborates on the relation of the taken approach to actual brain structures and cognitive psychology theories of learning and behavior.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Connectionism and neural nets; I.2.8 [**Problem Solving, Control Methods, and Search**]: Plan execution, formation, and generation

General Terms

Algorithms, Design, Performance.

Keywords

Bodyspaces, Cognitive Systems, Population Codes, Sensorimotor Codes, XCSF

1. INTRODUCTION

Michigan-style LCSs [2] evolve a set of rules, the so-called population of classifiers, where a rule consists of a condition, an action, and a prediction part. While the condition structure is evolved online, the prediction part is usually developed by some form of gradient-based approximation. In the XCSF classifier system [32, 33], conditions represent hyper-rectangles in a real-valued input space and predictions are computed linearly from the condition input and an offset value. Enhancements of the base XCSF system have shown that the predictions do not necessarily need to be linear, but can also be polynomial [22] or even predictions formed by a multilayer perceptron [20]. Moreover, it was shown that conditions can be changed to, for example, hyperellipsoidal conditions [5].

Previously, XCSF research had focused on improvements of the piece-wise approximations formed by the predictions, the condition structures, or the evolutionary mechanism. Besides these modifications, though, it is also possible to form predictions from variables that are different from, but related to, the input processed by the conditions. In this case, a classifier condition can be seen as identifying the context in which a certain (linear) prediction applies. This modification is introduced and studied in the first part of this paper.

Although John Holland had introduced learning classifier systems (LCSs) as *cognitive systems* [17]—with his first working system CS1 solving a food-water maintenance task [18]—LCSs have mainly been applied to classification and datamining problems [1, 19] as well as to other prediction and control problems, such as stock-market predictions, industrial plant control, or traffic junction control [21]. Robot control applications remain sparse and focus on immediate control problems, such as light following [20]. In this paper, we show that XCSF can be used to evolve classifiers in an arm posture space for the prediction of motor activity-dependent hand position changes of a simulated robot arm. Moreover, we show that a simple inversion of the representation can be used to control the robot arm.

The resulting representation can be compared with previous simulated neural approaches for directional, cortical arm control [4]. Moreover, it is now well-known that premotor and motor cortical areas of the brain represent the body by means of *population encodings*, where each neuron covers a certain subspace of the respective body part in its *receptive field* [13, 24, 27, 28]. This paper shows that XCSF can evolve a similar representation, in which each classifier condition specifies a certain subspace and the classifier population covers the complete problem space.

Moreover, we show that XCSF evolves its population encoding (that is, the set of classifiers) in order to optimize a prediction task, which is the prediction of hand position changes dependent on contextual arm posture and predictive motor activity information sources. This is in line with recent observations that the sensory system shows patterns that can have only developed to improve motor control capabilities, and thus, that sensory representations develop for motor control purposes [12, 14, 30]. The inversion of the learned forward predictions then enable directional arm control. Thus, the resulting modified XCSF may be termed a self-developing, cognitive arm control system.

The remainder of this paper is structured as follows. First, we give a short overview over XCS and XCSF in particular. Next, we detach the computed predictions from the condition input and investigate resulting performance. In Section 4 we introduce the arm control task and evaluate XCSF's forward predictive and inverse control capabilities. After a short summary, the final discussion suggests further modifications and applications of the XCSF system for the development and study of advanced cortical representations and motor control applications.

2. XCSF: A SHORT OVERVIEW

The classifier system XCS was originally designed as an online generalizing reinforcement learning system that approximates the Q-value function of a Markov decision problem [31]. More than ten years later, though, it has become clear that XCS is not only able to approximate Q-values with a compact and highly general representation, but it is also able to approximate real-valued functions [32, 33] and has been successfully applied to datamining problems [1], among various other recent applications (cf. [3] for a collection of applications and further literature pointers). In any of these applications, XCS evolves rules to partition the experienced problem space in such a way that accurate, general classifiers emerge, which together cover the whole problem space and yield accurate predictions.

In the case of XCSF,—the real-valued XCS version with linearly computed predictions [32, 33]—the classifier structure consists of condition and prediction only[1]. The condition part of a classifier determines the function domain subspace, in which the classifier prediction applies. In XCSF, the prediction is then determined from the inner product of the problem input vector and the classifier prediction weight vector plus a (possibly scaled) offset weight. Classifier fitness is derived from the inverse of the estimated mean error of the classifier predictions relative to the errors of overlapping classifiers. Thus, XCSF may be termed an online learning, piecewise-linear function approximation system in which classifier condition structures are evolved to improve the accuracies of their corresponding local linear approximations.

[1]An action part can, however, be included easily and is mentioned as a "dummy" action in Wilson's original work.

Classifier parameters are updated in online interaction with the problem at hand, iteratively processing problem instances and corresponding actual function values. Gradient-based techniques are used to approximate prediction, error, and fitness values. Dependent on the fitness values, a niche-based steady-state genetic algorithm is applied that selects highly accurate classifiers and deletes low-fitness classifiers in overcrowded niches. Together, the resulting evolutionary pressures have been shown to inevitably evolve generalized, piecewise-linear approximations of the target function [7].

In the case of the original XCSF, overlapping hyperrectangular condition structures and linear prediction structures were evolved. We use XCSF with rotating hyperellipsoidal condition structures [8], update the predictions with the recursive least squares technique [23], and apply set-proportionate tournament selection for offspring selection [10], since these mechanisms have been shown to improve XCSF's learning speed, accuracy, and robustness. Finally, we also use the recently introduced compaction mechanism [6], which upon invocation switches to closest classifier matching and turns off mutation and crossover, to investigate the possibility of generating even more general function approximations.

In the next section, we show how prediction structures may be detached from the condition structures. Condition inputs then are used only to determine matching while additional prediction inputs are used to compute the output prediction. For further information on XCS, the interested reader is referred to the algorithmic description of the system [11] and the cited literature.

3. SEPARATED INPUTS

Previously, computed predictions processed only the input considered by the conditions. No potential other input sources have been considered. We now show that XCSF's predictions can be computed out of some problem input that may differ significantly from the input used to match classifiers. In this case, classifier conditions may be seen as context processing units, which activate the program (in this case a linear prediction) that is maximally suitable to approximate the local problem space with the available expressive capabilities. In the following experiments, XCSF conditions perceive context information and cluster that information to enable maximally accurate predictions[2].

To study the effects on classifier structures and XCSF performance, we designed the following three functions to test XCSF's learning capabilities, its robustness, as well as its structural capabilities:

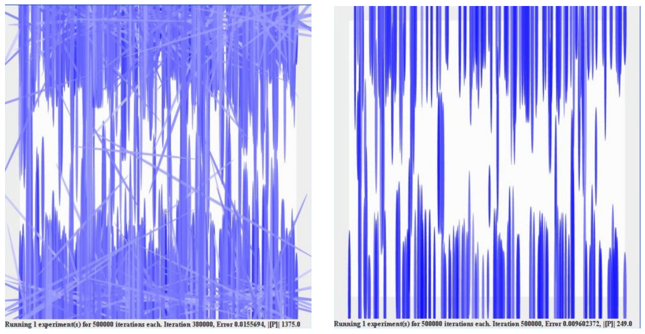
$$f_{1,m}^{n,2}(x_1,..,x_n,y_1,y_2) = a*(b*x_1+y_1+j*y_2) \quad (1)$$

$$f_{2,m}^{n,2}(x_1,..,x_n,y_1,y_2) = a*(b*\sum_i^n x_i+y_1+j*y_2) \quad (2)$$

$$f_{3,m}^{n,2}(x_1,..,x_n,y_1,y_2) = a*(\sin(b*x_j)+y_1+j*y_2) \quad (3)$$

where a and b are scalar function modifiers and n specifies the number of condition input dimensions. Each function specifies m output dimensions (with $1 \leq j \leq m$). The number of prediction input dimensions was set to 2 throughout. To distinguish condition from prediction input, we denote condition inputs by x_i and prediction inputs by y_i.

[2]If not stated differently, parameters were set to the values reported in [8]. Results are averaged over 20 runs except for the behavioral tests, for which 10 runs were averaged.



```
Running 1 experiment(s) for 500000 iterations each. Iteration 380000, Error 0.0155694, |[P]| 1375.0
Running 1 experiment(s) for 500000 iterations each. Iteration 500000, Error 0.009602372, |[P]| 249.0
```

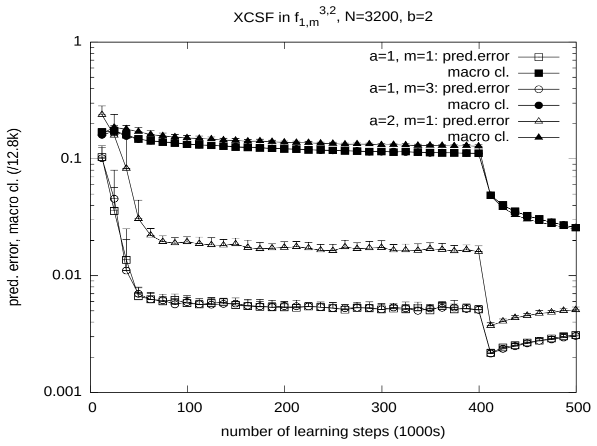
Figure 1: Classifiers only focus on partitioning the first context dimension in problem f_1 (shown with $n = 2$). Shown are the classifier conditions (20% of actual size) distributed over the condition-input space before (380k learning iterations) and after compaction (500k l.it.).

Since multiple outputs need to be predicted (that is, m outputs), each classifier learns m prediction weight vectors per classifier—one for each output dimension. The error is determined by the average absolute error over all output dimensions.

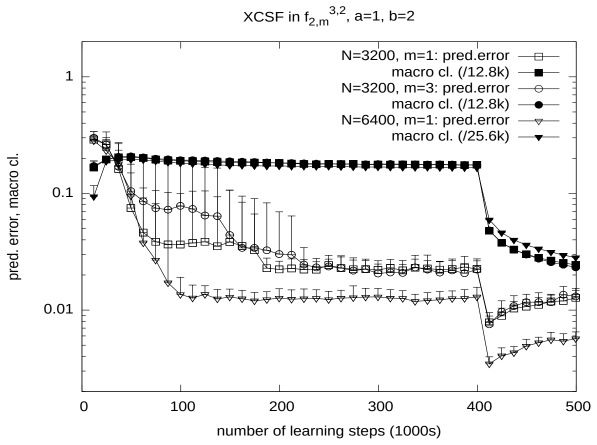
Function f_1 has a very linear structure and could be solved easily by various approximation techniques. For each separate output dimension, the function changes only linearly by a larger factor j. Other than that, inputs are only dependent on x_1, so that classifiers evolve that ignore additional input dimensions. Figure 1 shows a typical classifier distribution at the end of a learning run before and after compaction, showing the classifier conditions in 20% of their actual size. The plot confirms that classifier conditions partition the function-relevant input dimension x_1 and generalize over x_2.

Since the condition structure should be equally effective for all output dimensions in Function f_1 and the output dimensions are approximated in parallel in each classifier, the number of output dimensions should not affect learning performance. This fact can be seen in Figure 2(a), where the number of input dimensions does not affect performance, but a larger scaling factor does, because the scaling doubles the importance of the context dimension x_1. It is also interesting to see that the compaction algorithm [6, 9], which is activated after 400k learning iterations and essentially disables crossover and mutation and changes to closest classifier matching, does not only decrease population size (number of distinct classifiers) but also further decreases the error, selectively deleting inaccurate offspring and less accurate, overlapping classifiers.

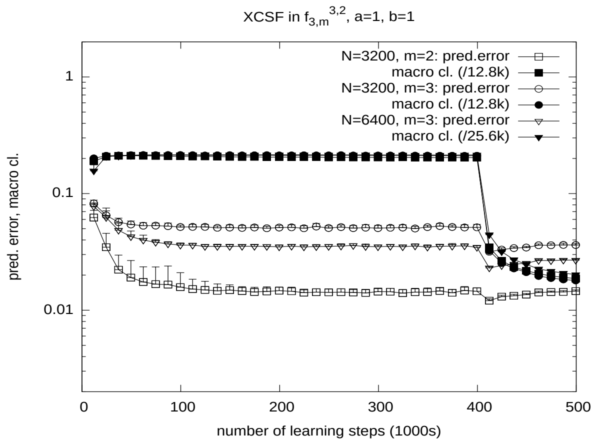
Function f_2 requires the structuring of all n context dimensions. But, as in f_1, each output dimension is equally dependent on the context input dimensions. Moreover, due to the linear combination of the context dimensions, that is, $\sum_i^n x_i$, the context space needs to be structured obliquely in the three dimensions. Figure 2(b) shows XCSF's performance in Function f_2. As in f_1, the number of output dimensions does hardly affect learning accuracy and XCSF evolves increasingly accurate space partitions. With a larger population size of $N = 6400$, an error of less than .01 is reached. Compaction further improves performance and decreases population size in all cases.



(a) XCSF performance in $f_{1,m}^{3,2}$



(b) XCSF performance in $f_{2,m}^{3,2}$



(c) XCSF performance in $f_{3,m}^{3,2}$

Figure 2: Performance of the modified XCSF in the three simple test functions shows competent, contextual processing. After $400k$ iterations, compaction was activated. In Function f_1, only one context dimension is relevant for an accurate prediction. Function f_2 requires oblique clustering of the context dimensions. Function f_3 becomes increasingly harder with $m = 3$ output dimensions.

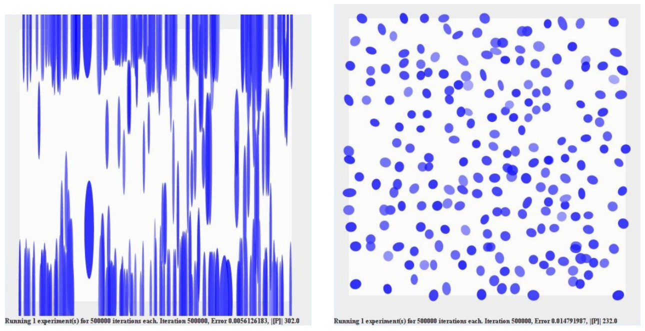


Figure 3: Dependent on the number of output dimensions in Function f_3 ($n = 2$, $m = 1$ left vs. $n = 2$, $m = 2$ right-hand side), classifier conditions partition the context input space in slices for $n = 1$ and in somewhat equally sized and distributed subspaces for $m = 2$. Classifier conditions are shown in 20% of their actual size.

The hardest challenge is posed by Function f_3 with $m > 1$, because each output dimension depends on a different contextual input dimension. Given only one output dimension ($m = 1$), f_3 is essentially identical to f_1, except for that the context is still processed by a sine function. In higher dimensions, however, each output dimension is dependent on another context dimension. Thus, since each classifier predicts all output dimensions, the problem becomes similar to a real-valued checkerboard problem [29], where each input dimension should be structured finer, the larger the absolute derivative of the sine function in this dimension. The partitioning can be observed in the evolving population of XCSF. Figure 3 shows typical final populations of XCSF runs in Function f_3 after compaction. On the left hand side, a population is shown for the setting with $n = 2$, $m = 1$, $a = 1$, and $b = 2\pi$. It can be seen that the partitioning actually reflects the sine wave with finer partitions in areas with larger derivatives. In the case of two output dimensions ($n = 2$, $m = 2$, $a = 1$, $b = 1$), classifier conditions distribute themselves somewhat equally over the condition input space, reflecting the checkerboard quality of the problem (Figure 3, right-hand side). Figure 2(c) shows XCSF's performance in the problem. Clearly, the additional output dimension results in a significantly harder problem.

In sum, the results show that XCSF can be very well extended to process contextual input, which is somewhat independent of the input used to compute the predictions. In the next section, we show one first possible application of such an XCSF system.

4. COGNITIVE ARM CONTROL

While the previous section has shown that XCSF can evolve classifiers that cluster contextual information for the generation of accurately predicting classifiers, we now move to an arm control task, which is inspired by cognitive control models.

Despite continuing progress, the understanding of human motor control is far from complete and the computational processes underlying infant motor learning are highly obscure. Various research directions suggest that infants learn internal forward models of body movements during development, which enable the inverse, goal-directed control of the body [15]. Consequently, several interactive forward-inverse models have been developed [34]. The learning process during development is obviously self-supervised and solves the challenge of building control structures for a highly dynamic and redundant plant.

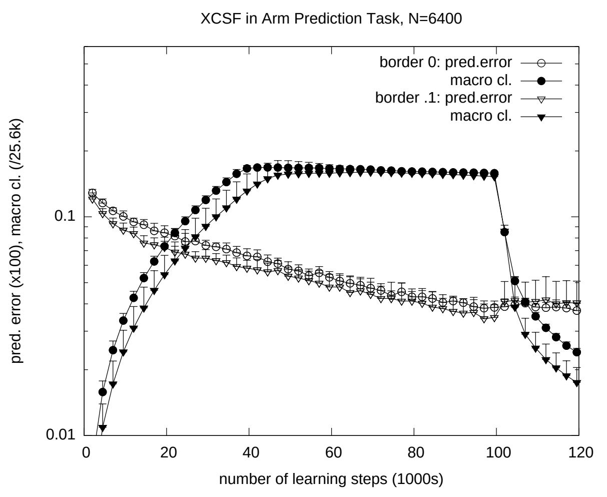
In the following, we address two aspects of this inverse model learning problem, applying it to a redundant, three-joint planar arm. First, we show that XCSF can partition the arm posture space to enable highly accurate predictions, that is, forward model representations. It learns to predict accurate hand displacements given current motor activity and hand location. Second, we show that the encoded information may be used inversely to determine the motor actions that move the hand, that is, the arm end-point towards a desired target.

Somewhat mimicking the neural network approach in [4], we train XCSF on a three-degree of freedom arm control problem with context input that encodes the current arm posture (three angles)[3]. The prediction input, on the other hand, encodes current motor activity (change in angles) but predicts the resulting change in hand location (that is, the difference in arm end-point coordinates before and after movement). We train XCSF on this prediction task but do not only evaluate the predictive capabilities but also the resulting inverse control capabilities. The question is, whether the evolving representation is able to guide the arm to given goal states by inverting the piecewise-linear mapping between motor command and hand location change.

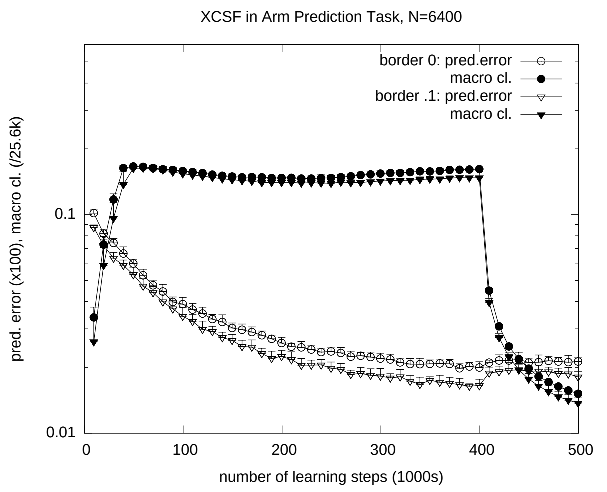
Figure 4 shows typical XCSF performance curves in the prediction task, confirming that this context-based problem is learnable by XCSF. Prediction accuracy continuously improves. The application of compaction decreases the population size by over 90% while affecting performance only marginally. This is the case after $120k$ steps (Figure 4a) and also in longer runs with $500k$ learning iterations (4b). To test if border effects might affect performance, we trained XCSF on the full problem space (border 0) and on an inner problem space with a surrounding unsampled region that covers the outer 10% of the length of each dimension. While the inner region is even easier to approximate, also the runs on the full region yield accurate predictions.

While the evaluation of prediction accuracy just confirms the prediction capabilities of the modified XCSF version, the arm control task is a new challenge for XCSF. Given a current arm posture, the current match set is determined—as in the usual prediction case— which however is then used to determine the inverse motor command as follows. For each classifier in the set, the inverse of the forward prediction is calculated. However, since there are only two linear equations, which usually are used to predict hand displacements $(\Delta x, \Delta y)$ given three angular motor commands, but now we intend to determine the three motor commands necessary to achieve a desired hand displacement, an additional constraint is necessary. A simple solution is to set one of the three motor commands to zero—effectively prohibiting the

[3]The simulated arm has limb lengths of 1.4, 1.4, .8 arm units. The joints can rotate freely between -60° and 150° (shoulder), 0° and 150° (elbow), and -10° and 150° (wrist), using the specifications from [4]. During training, random instance samples were generated that moved the arm selectively in one dimension by a uniformly chosen value between $-.18^\circ$ and $.18^\circ$. The error threshold was set to $\epsilon_0 = .0001$.



(a) Compaction after 100k learning steps



(b) Compaction after 400k learning steps

Figure 4: XCSF learns to predict motor-dependent arm displacements effectively and accurately. Compaction decreases population size by over 90% while hardly affecting prediction accuracy.

movement of one joint. However, since all three joints may need to be moved to reach a given goal state, we simply take turns and set a different motor command to zero in each movement command calculation. The overall motor command in one calculation is determined by the average movement command suggested by all (micro-)classifiers in a match set. Before executing the command, the movement vector was normalized to a total angular movement of 1.8° (smaller step sizes did not alter performance significantly).

To evaluate the inverse control performance, we tested the evolving XCSF populations on a set of 78 start-posture, goal-hand-location combinations. Particularly, we set the arm to 27 different arm postures (shoulder, elbow, and wrist angles either flexed, centered, or bent) and test it on reaching three hand locations, which are located at the position where shoulder, elbow, and wrist angles are either all flexed, centered, or bent[4]. Since extreme angular values cause non-linear border effects and consequently targeted trajectories might lead through unreachable terrain, we further restrict the maximal angles by a certain border surrounding the inner angular posture space, as specified in the graphs. To evaluate the effectiveness of the movements, we monitor not only the success rate but also the path efficiency, which we measure by dividing the Euclidean distance from hand start to hand goal location with the actual distance the hand moved from start to goal. We report the average path efficiency over all successful trials out of the 78 test cases. A goal was considered successfully reached if the hand moved into a region of .05 arm length units around the goal.

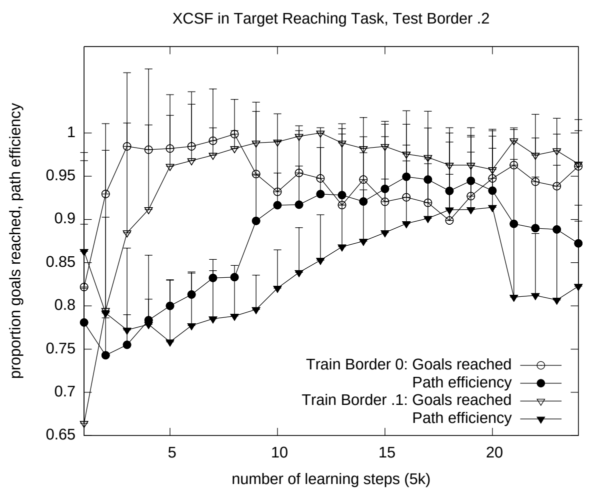
Figure 5 shows that XCSF's success rate increases during learning. Also the path efficiency improves but it plateaus after about $60k$ learning iterations. It appears that XCSF overspecializes its population for the generation of accurate forward predictions so that the representation becomes overgeneralized for the inverse computation, as can also be seen in the decreased performance after compaction was applied (after 100k iterations). Nonetheless, especially the inner area of the arm is controlled reliably and effectively by the evolved XCSF classifiers. By closer inspection of the actual movements executed, we noticed that some of the generated trajectories ended in a circling behavior around the goal location. This may be due to the simple inverse computation approach, where we iteratively set each angular movement to zero.

An alternative, non-deterministic approach can be found in the neural network literature on hierarchical vision models. Rao and Ballard [26] showed that a neural network can iteratively approximate a linear function accurately-enough to develop emergent visual cortical features, such as edge detectors and even more complex image structures. Essentially, the forward linear equation is transposed determining an approximate inverse. The resulting activity is forwarded back to the input, resulting in an error between input activity and predicted activity. The difference is used to propagate the next inverse activity by the transpose, etc. To ensure sufficient convergence of this process, we executed this forward-backward interaction for 100 iterations with an adaptation rate of .05. For further information the interested reader is referred to the original paper [26].

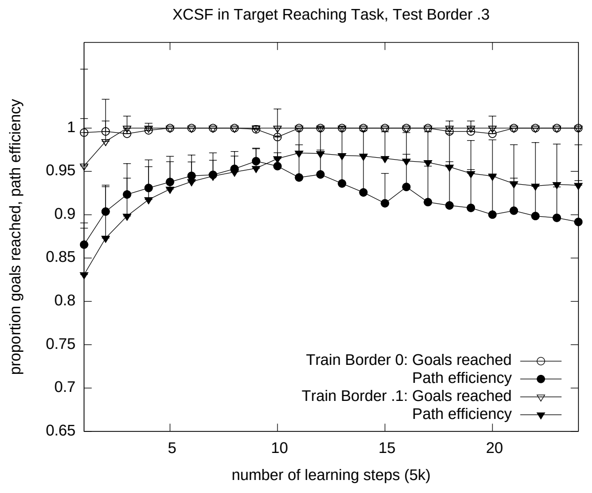
Figure 6 shows that the approximate inverse computation approach improves the success rate as well as the path efficiency of XCSF, reaching a success rate of 100% reliably before compaction. However, also in this case path efficiency stalls at an approximately 95% level and decreases after compaction is applied. As suggested before, the apparently over-fitted conditions for prediction yield a decreased inverse accuracy.

Nonetheless, a 95% path efficiency corresponds to nearly straight trajectories. A set of tested sample trajectories are shown in Figure 7. The trajectories were generated from a population of classifiers after $60k$ learning iterations, train border .1, test border .3, and the interactive, approximate inverse computation. Only the inner and outer goal location tests are shown. It can be seen that the invoked hand movements (arm end point) are rather direct and always reach

[4]Only 26 start postures were actually tested with each goal location—thus testing $26 * 3 = 78$ start-goal combinations—since there is one coinciding start posture for each of the goal locations.



(a) Wider area test



(b) Narrower area test

Figure 5: The arm learns to reach targets reliably in the inner space (b). Targets closer to extreme postures yield errors and worse trajectories due to ineffective border representations (a).

the goal location accurately.

5. SUMMARY AND CONCLUSIONS

This paper has shown that XCSF can be applied not only as a forward predictive system approximating functions but also as a context-dependent processing system that structures contextual information dependent on the predictions formed given other information. The capability of the resulting modified XCSF system was first exemplarily evaluated in three artificial function problems, in which one or more context dimensions modified the target values. Certainly various other types of contextual information, dependent on the intended application, are imaginable, such as nominal or binary represented inputs or mixed inputs. Moreover, also mixture models are imaginable in which parts of (or all) the context information may also be used for the generation of the predictions. The evaluations also showed that XCSF distributes its classifiers very effectively in such tasks.

Besides the capability of processing context input separately from predictive input, we showed that XCSF may be applied for the control of a three degree of freedom arm. In this case, XCSF learned how motor commands (changes in joint angles) affected the displacement of the hand, given the current arm posture. We showed that XCSF can be used to generate accurate forward predictions in this task. Furthermore, we also showed that these forward predictions can be inversely used to invoke directional arm control. Given a current desired direction to a goal, XCSF can be used to invoke suitable motor commands that guide the arm to the goal on an approximately optimal path. Future research needs to further evaluate these capabilities and also further investigate the slight decrease in behavioral performance after an initial significant performance increase. Alternatively, it should also be investigated if XCSF can be trained to actually optimize the inverse representation directly instead of the forward predictive representation learned in the presented work.

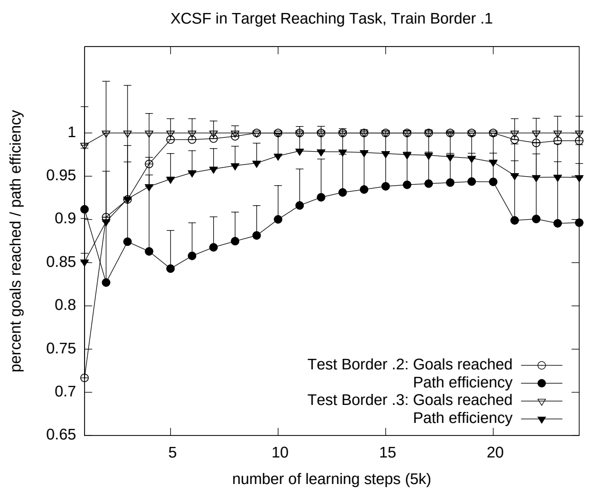


Figure 6: With the approximate inverse computation, XCSF reaches all goals also in the harder case with goal states closer to the border.

6. FINAL DISCUSSION

Current computational models of human behavioral control rely extensively on a sparse representation of perception. Such a representation requires that perceptual input is grouped into several more or less discrete units, which are of high relevance for many aspects of behavioral control. On the neural level, parietal and motor cortical areas encode sensory information in population codes, where each neuron represents a range of possible perceptions [13]. Likewise, motor-actions may also be organized in discrete units, so-called motor primitives [25]. On a higher level, also the integration of multiple skills requires a partitioning of the behavioral context in order to enable stable inverse model learning and context-dependent control [16]. The partitioning of sensory spaces, however, which underlies these models, is mostly not evolved but predefined or generated independent of the interaction between sensory input and motor activity. However, research has shown that spatial

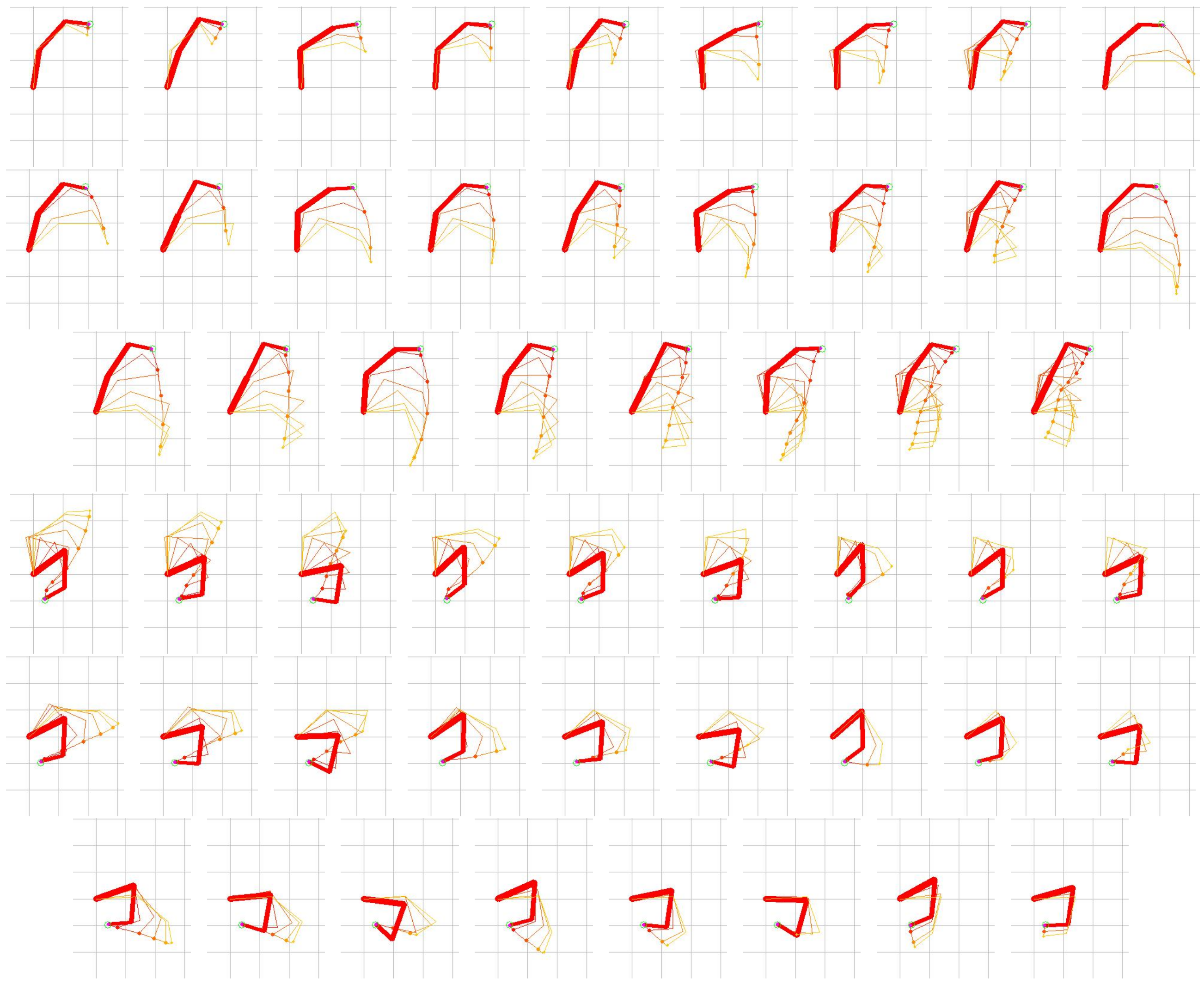


Figure 7: The resulting arm trajectories show that the hand moves approximately straight to the goal location, where the final arm posture is drawn thicker than the intermediate postures on the trajectory.

coverages are developed highly pro-actively, optimizing the spatial partitions for the individual behavioral sensorimotor demands [14].

It has been shown previously that XCSF is a system that clusters a problem space for the purpose of solving a given prediction task, that is, XCSF is a self-organizing system, which clusters for the purpose of prediction. The proposed XCSF setup evolves a bodyspace partitioning for the prediction of accurate motor activity effects on sensory input, that is, for accurate sensorimotor learning. Thus, XCSF provides a platform that can evolve population encodings for the purpose of prediction and control. In the future, XCSF inspired models may help to gain a deeper understanding of the processes that shape neural body control structures and that link different sensory modalities or goal representations with motor actions. Moreover, other context-dependent control applications may be investigated, such as the control of other body parts.

Although John Holland might have not imagined it quite this way, sensorimotor cognitive control is possible with learning classifier systems. The results have shown that XCSF can be applied as a complex cognitive control system, which develops sensorimotor structures from scratch based on initially simple environment interactions. Future research will show how more complex cognitive control tasks with, for example, more degrees of freedom in higher dimensions may be tackled by enhanced, possibly further modularized, XCSF-based systems.

Acknowledgments

The authors acknowledge funding from the Emmy Noether program of the German research foundation (grant BU1335/3-1) and like to thank their colleagues at the department of psychology and the COBOSLAB team.

7. REFERENCES

[1] E. Bernadó-Mansilla and J. M. Garrell-Guiu. Accuracy-based learning classifier systems: Models, analysis, and applications to classification tasks. *Evolutionary Computation*, 11:209–238, 2003.

[2] L. B. Booker, D. E. Goldberg, and J. H. Holland. Classifier systems and genetic algorithms. *Artificial Intelligence*, 40:235–282, 1989.

[3] L. Bull, editor. *Applications of Learning Classifier Systems*. Springer-Verlag, 2004.

[4] D. Bullock, S. Grossberg, and F. H. Guenther. A self-organizing neural model of motor equivalent reaching and tool use by a multijoint arm. *Journal of Cognitive Neuroscience*, 5:408–435, 1993.

[5] M. V. Butz. Kernel-based, ellipsoidal conditions in the real-valued XCS classifier system. *GECCO 2005: Genetic and Evolutionary Computation Conference*, pages 1835–1842, 2005.

[6] M. V. Butz. Documentation of XCSFJava 1.1 plus visualization. MEDAL Report 2007008, Missouri Estimation of Distribution Algorithms Laboratory, University of Missouri in St. Louis, MO, 2007.

[7] M. V. Butz, T. Kovacs, P. L. Lanzi, and S. W. Wilson. Toward a theory of generalization and learning in XCS. *IEEE Transactions on Evolutionary Computation*, 8:28–46, 2004.

[8] M. V. Butz, P. L. Lanzi, and S. W. Wilson. Hyper-ellipsoidal conditions in XCS: Rotation, linear approximation, and solution structure. *GECCO 2006: Genetic and Evolutionary Computation Conference*, pages 1457–1464, 2006.

[9] M. V. Butz, P. L. Lanzi, and S. W. Wilson. Function approximation with XCS: Hyperellipsoidal conditions, recursive least squares, and compaction. *IEEE Transactions on Evolutionary Computation*, in press.

[10] M. V. Butz, K. Sastry, and D. E. Goldberg. Strong, stable, and reliable fitness pressure in XCS due to tournament selection. *Genetic Programming and Evolvable Machines*, 6:53–77, 2005.

[11] M. V. Butz and S. W. Wilson. An algorithmic description of XCS. *Soft Computing*, 6:144–153, 2002.

[12] J. G. Fleischer. Neural correlates of anticipation in cerebellum, basal ganglia, and hippocampus. In M. V. Butz, O. Sigaud, G. Pezzulo, and G. Baldassarre, editors, *Anticipatory Behavior in Adaptive Learning Systems: From Brains to Individual and Social Behavior*. Springer-Verlag, 2007.

[13] A. P. Georgopoulus. Current issues in directional motor control. *Trends in Neuroscience*, 18(11):506–510, 1995.

[14] M. Graziano. The organization of behavioral repertoire in motor cortex. *Annual Review Neuroscience*, 29:105–134, 2006.

[15] A. Greenwald. Sensory feedback mechanisms in performance control: with special reference to the ideo-motor mechanism. *Psychological Review*, 77:73–99, 1970.

[16] M. Haruno, D. M. Wolpert, and M. Kawato. Mosaic model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220, 2001.

[17] J. H. Holland. A cognitive system with powers of generalization and adaptation. Unpublished manuscript, 1977.

[18] J. H. Holland and J. S. Reitman. Cognitive systems based on adaptive algorithms. In D. A. Waterman and F. Hayes-Roth, editors, *Pattern directed inference systems*, pages 313–329. Academic Press, New York, 1978.

[19] J. H. Holmes, D. R. Durbin, and F. K. Winston. A new bootstrapping method to improve classification performance in learning classifier systems. *Parallel Problem Solving from Nature PPSN*, VI:745–754, 2000.

[20] J. Hurst and L. Bull. A neural learning classifier system with self-adaptive constructivism for mobile robot learning. *Artificial Life*, 12:1–28, 2006.

[21] T. Kovacs. Bibliography of real-world classifier systems applications. In L. Bull, editor, *Applications of Learning Classifier Systems*. Springer-Verlag, Berlin Heidelberg, 2004.

[22] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg. Extending XCSF beyond linear approximation. *GECCO 2005: Genetic and Evolutionary Computation Conference*, pages 1827–1834, 2005.

[23] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg. Prediction update algorithms for XCSF: RLS, kalman filter and gain adaptation. *GECCO 2006: Genetic and Evolutionary Computation Conference*, pages 1505–1512, 2006.

[24] A. Maravita, C. Spence, and J. Driver. Multisensory integration and the body schema: Close to hand and within reach. *Current Biology*, 13:531–539, 2003.

[25] F. A. Mussa-Ivaldi and E. Bizzi. Motor learning through the combination of primitives. *Philosophical Transactions of the Royal Society: Biological Sciences*, 355:1755–1769, 2000.

[26] R. P. N. Rao and D. H. Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999.

[27] G. Rizzolatti, L. Fadiga, L. Fogassi, and V. Gallese. Enhanced: The space around us. *Science*, 277:190–191, 1997.

[28] A. B. Schwartz, D. W. Moran, and G. A. Reina. Differential representation of perception and action in the frontal cortex. *Science*, 303:380–383, 2004.

[29] C. Stone and L. Bull. An analysis of continuous-valued representations for learning classifier systems. In L. Bull and T. Kovacs, editors, *Foundations of Learning Classifier Systems*, Studies in Fuzziness and Soft Computing, pages 127–175. Springer-Verlag, Berlin Heidelberg, 2005.

[30] P. F. M. J. Verschure, T. Voegtlin, and R. J. Douglas. Environmentally mediated synergy between perception and behaviour in mobile robots. *Nature*, 425:620–624, 2003.

[31] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

[32] S. W. Wilson. Function approximation with a classifier system. *Proceedings of the Third Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 974–981, 2001.

[33] S. W. Wilson. Classifiers that approximate functions. *Natural Computing*, 1:211–234, 2002.

[34] D. M. Wolpert and M. Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11:1317–1329, 1998.