C-block: A system for learning motor plans with perceptual consequences

Martin Takac, Alistair Knott, Mark Sagar

C-block: A system for learning motor plans with perceptual consequences

Martin Takac
Soul Machines, Ltd
Comenius University in Bratislava
Slovakia
martin.takac@soulmachines.com

Alistair Knott
Soul Machines, Ltd
University of Otago
New Zealand
alistair.knott@soulmachines.com

Mark Sagar
Soul Machines, Ltd
University of Auckland
New Zealand
mark.sagar@soulmachines.com

***Abstract*—In this paper we present a connectionist model of sequence learning, chunking and planning called C-block. C-block is a fragment of a complex architecture for modeling cognitive development, embodied in a virtual baby simulator BabyX. We illustrate its operation in a drawing scenario, in which the system learns sequences (chunks) of motor gestures together with visual images they produced. The learned representation allows recognition of a plan from a partially drawn shape, individual and collaborative drawing, and drawing a picture by evoking several partial plans. C-block is based on a modified version of the self-organizing map (SOM) that allows fast learning, approximate queries and Bayesian inference.**

***Index Terms*—chunking, sequence learning, neural networks**

I. Introduction

Building a general-purpose intelligent system has been the holy grail of the AI community since its birth in 1950s. Recent advances in deep neural networks make them the state-of-the-art tool for tackling a wide range of difficult problems in image recognition, language processing and game control [1]. Despite its success, this route to intelligent behaviour is quite different from human cognition. Human cognition and language are embodied [2], grounded in sensorimotor feedback loops and situated interactions with the environment and other agents [3], [4]. More importantly, cognitive abilities develop incrementally by learning from experience [5]. In our work we adhere to these modelling principles.

We are developing BabyX [6], [7]—a hyperrealistic virtual simulation of a human baby combining models of the facial and skeletal motor system and models of basic neural systems involved in interactive behaviour and learning. The baby has a simulated body embedded in a virtual environment via real-time perception-response loops that activate her simulated emotions and internal states. The emotions are constantly expressed in her facial expressions and motor behaviours.

Human experimenters can interact with the baby in a shared virtual scene or via a camera and microphone. BabyX conceptualizes her sensorimotor experience in terms of objects and events. Her representations of these are designed to interface directly with language mechanisms [8], [9], so she is naturally configured to talk about the events she experiences in her world, in accordance with embodied models of language.

Martin Takac was partially supported by the grant VEGA 1/0796/18.

From the programming point of view, BabyX is a network of interconnected modules that run in parallel and communicate through parameter values. Multiple coupled dynamical systems (together with interaction coupling with the external world) create a potential for emergent behaviours. Individual building blocks implement partial cognitive models (attention, visual classification, working memory, etc.) in a LEGO-like fashion. In this paper we focus on one such building block—a mechanism capable of recognizing and learning sequences. We call it 'C-block' (where the 'C' stands for chunking). We describe the application of C-block for chunking of perceptuomotor gestures, one of the set topics for this workshop.

As in humans, BabyX engages with her world through sequentially structured programmes of attentional, cognitive and motor actions—*deictic routines* [10]. It is important that these sequences can be learned and represented *declaratively*—for behaviour control and planning, but also to serve as meanings of linguistic expressions. In BabyX's architecture, different C-block instances learn declarative representations of a wide variety of sequences, from low-level events involved in the production of facial expressions, through motor gestures, to high-level events associated with utterances in a dialogue. Declarative representations of sequential motor gestures are particularly important when these sequences produce recognisable perceptual stimuli—for instance, when a sequence of drawing movements produces a recognised object shape. In such cases, the baby must register that a certain object is *created*: for instance that "*Ali drew a house*". Here, a specific motor sequence is associated with a declarative object representation.

In the rest of the paper we describe the C-block for chunking of motor gestures in a drawing scenario, in which the baby and the user draw on a virtual drawing board. C-block learns sequences of individual movements as motor programs, represents them declaratively and associates them with their perceptual consequence, in this case a visual shape drawn by the sequence of movements.

II. C-block architecture

A. Motor gestures and drawing board

In the drawing scenario, the user and BabyX can both draw on a shared virtual drawing board. BabyX draws by

Fig. 1. BabyX drawing on a virtual touch screen.

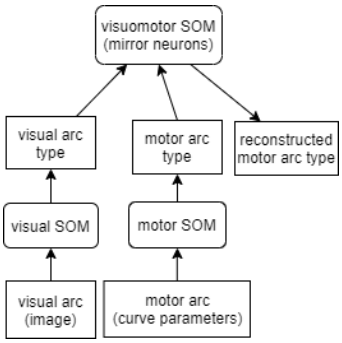

Fig. 2. The mirror system for visuo-motor transformations.

moving her arm while touching the drawing board with her finger (Fig. 1). The user draws by moving the mouse while pressing down the mouse button. We represent elementary arm movements by parabolic curves [11]. The C-block takes input in the form of parameters of each curve. In case BabyX is drawing, these are directly available from her motor system. In case the user is drawing, the curve parameters need to be inferred from the observed visual shape. For this we use a "mirror neurons system" that can be pretrained by motor babbling or directed movements to associate a variety of BabyX's own movements with their visual shapes. The system consists of two self-organizing maps (SOM) [12] for visual and motor input and another SOM trained on concatenation of the activities of the visual and motor SOMs (see Fig. 2). When the user is drawing, the trained mirror system reconstructs the activity of the motor SOM from the visual input. This is then fed in the C-block, which learns incrementally from the incoming sequence elements.

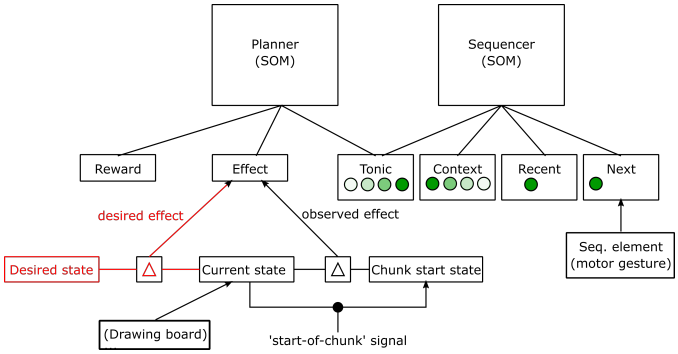

Fig. 3. The C-block architecture. C-block receives individual motor gestures via the Next field. Sequencer computes the remaining fields (Recent, Context, Tonic) and learns their association. At the end of each sequence, Planner associates the compressed representation of the whole sequence (Tonic) with Reward and the net Effect of the plan. When given a desired state (red arrows), it selects the best matching plan and uses it to drive Sequencer's prediction via the top-down-filled Tonic.

B. *Sequencer*

The C-block consists of two core components—a *Sequencer* and a *Planner* (Fig. 3). The Sequencer learns to predict the next element (Next) from the representation of the most recent element (Current), a cumulative decayed representation of previous elements (Context) and a declaratively represented hypothesis about the whole sequence (Tonic). Each time a new element arrives, the four fields feeding the Sequencer are updated and stored in an internal buffer:

1) Context is decayed and Current is added to it,
2) Next becomes the new Current,
3) Next is decayed and added to Tonic,
4) Next is replaced with the newly arrived element.

In this way both Context and Tonic contain a decaying representation of the sequence seen so far, but Tonic encodes the oldest elements most strongly, while Context is biased towards the most recent. This way of building Tonic helps to orient prediction towards those stored sequences that are consistent with the fragment seen so far.

The Sequencer is implemented as a modified version of the SOM. The SOM is trained on concatenated combinations of Tonic, Context, Current and Next. The learning is incremental and fast (1-shot). Individual parts of the input vector are compared with the memories stored in the weights of the SOM neurons using weighted Euclidean distance. This models situational queries with weighted attention towards certain parts of the input vector (e.g., 'what is the most likely Tonic, given Context, Current and Next with given attentional weights?'). As we argue in [13], this version of the SOM performs Bayesian inference: its activity represents a probability distribution over hypotheses stored in the neurons' weights and the reconstructed input corresponds to the expected value conditioned on the distribution.

The Sequencer predicts a distribution over possible Next elements at each step. However, to avoid contaminating learning with incomplete or messy sequences, the sequence is first

stored in an internal buffer and only replayed from there to the SOM (with high learning rate) once the sequence has successfully completed.

C. Sequence boundary

There are multiple criteria for detecting that a sequence has completed. One relates to **pauses:** if a long enough idle time has passed since the last element arrived, further elements will be considered a different sequence. Another relates to **surprise or uncertainty:** the uncertainty/entropy of the next element prediction is highest at chunk boundaries [14]. Alternatively, a sequence ends in case of surprise, i.e. when the actual next element deviates significantly from the prediction [15]. A final criterion relates to **reward:** if the perceptual outcome of the motor sequence is highly rewarding, it can trigger completion of the sequence.

In our drawing scenario, we combine these approaches. A positive reward arrives when the visual and auditory systems detect positive reaction from the human partner—determined by facial expression classification and vocal tone analysis of infant directed speech. This functions as a signal that a meaningful sequence has been completed, and the Sequencer is trained from the buffer. If a long-enough time has passed after the last element and no reward arrives, the sequence is forgotten: the buffer is cleared and the Sequencer is not trained.

D. Planner

We now consider the second core component of C-block—the *Planner*. Its role is to register declarative representations of successfully completed sequences and associate them with their perceptual consequences and a reward value. Thus the Planner only learns at chunk boundaries. At a chunk boundary, the Planner registers a current perceptual state (in our case, the state of the drawing board) as the initial state of a new chunk. Then, when the chunk is successfully completed, it computes the difference between the perceptual state at the chunk end and the stored initial state (the Δ circuit in Figure 3). This difference constitutes the state update—the net perceptual effect of executing the just completed motor plan. (For example, if a drawing board contained a square and now we draw a triangle on top of it, the effect of the 'triangle' program would just be the triangle shape.)

At the time of sequence completion, the Sequencer's Tonic field contains a compressed declarative representation of the whole chunk. The Planner learns to associate this representation with the chunk's perceptual effect and a reward value (if received). Because it is implemented with the same type of Bayesian SOM as the Sequencer, it supports queries from *partial input*, such as 'What is the distribution over stored plans given the match between their stored Tonic components and an observed Tonic in Sequencer given a fragment of the sequence seen so far?' Querying with partial inputs allows the C-block to perform *plan/intent recognition* from observed sequence fragments. When a new element arrives and Tonic, Context and Current are updated, the Planner is queried to reconstruct a Tonic corresponding to a *full* sequence. This reconstructed Tonic then temporarily replaces the actual Tonic for prediction of the most likely next element. The Planner also reconstructs Effect and Reward fields, so it is able to predict the expected reward value and effect from the incomplete fragment.

E. Plan execution

Besides recognition of plans from bottom-up observation, the Planner can be driven top-down, by a desired perceptual effect. In this case, it is queried with different attention weights to retrieve a distribution over stored plans best matching the desired effect (computed as a difference between a desired perceptual state and the current one, see red arrows in Fig. 3). A winner is picked from this distribution. If its reconstructed Effect sufficiently matches the desired one (i.e. the plan is appropriate), the plan is selected for execution and its reconstructed Tonic drives the Sequencer to generate elements of the plan one by one. The top-down driven execution of a plan ends if (1) the actual perceptual effect reached so far sufficiently matches the desired one, or (2) the actual next element of the plan does not match the predicted one (unexpected deviation from the plan), or (3) the stored sequence was replayed to its end.

We have also implemented an inhibition of return (IOR) mechanism operating on the activity of the Planner SOM. After a plan ends, its activity in the SOM is temporarily suppressed, so the next best plan will become a winner. This allows the system to iteratively try different plans until the goal has been reached or there are no more good plans. To illustrate, assume the C-block has already learned two motor programs, that draw a rectangle and a triangle respectively. If the desired goal is to draw a house (a triangular roof on top of a rectangle), the Planner will activate both plans, and one of them will become a winner in the SOM's activity map (e.g. the rectangle plan). If the match is sufficient, the plan will be selected for execution and the rectangle will be drawn. With the fixed desired perceptual state (the house) and the actual state (the rectangle), the desired effect now changes to their difference (the triangle). The triangle plan will now win the competition in the Planner SOM's activity map and it will be executed next.

In this way, the C-block can dynamically build a motor plan from several stored partial plans—and IOR will prevent it from getting stuck with a failing plan.

F. Collaborative drawing

Sequence observation and execution can be easily combined, leading to a scenario in which the baby collaborates with the human in the drawing task. Let us assume the C-block has already observed the human drawing and has learned several plans, among them a plan to draw a heart shape. When the user starts drawing a heart, C-block can recognize the plan and its perceptual consequences (the heart) well before the drawing is finished. Thanks to the Bayesian interpretation of the activity maps in our SOMs, and to the sparse representation

of x, y positions in Next field, we are able to measure the uncertainty of prediction as the entropy of the generated distributions. Let us now assume that the user stops before finishing the heart. If the C-block is certain about the plan and the next movement (based on low entropy), an 'impatience' value will start building up. If the user resumes the drawing, the impatience timer is turned off, but if not, the baby will soon jump in and complete the shape herself.

III. Experiments

In the presented drawing scenario, the state of the virtual drawing board (a raw bitmap) is directly sent to the C-block as the external world state for computing plan effects. The system works with bitmaps of arbitrary dimension, provided the input media and SOM dimensions are large enough or scaled accordingly. For 40x30 bitmaps that we used in our experiments, the Sequencer SOM with 2500 neurons and 20x20 encoding of each of Tonic, Context, Current and Next (i.e. 4 million weights in total) and the Planner with 900 neurons and up to 1.5 million weights give sufficient precision. Natural properties of self-organizing maps guarantee that if the SOM capacity is exceeded (i.e., the number of plans or plan elements exceeds the number of neurons), similar elements will blend and the performance will degrade gracefully. Both SOMs have a high learning rate (0.9) and a small Gaussian neighbourhood size ($\sigma = 0.6$). The decay coefficient is 0.6 for both Context and Tonic and 0.4 for plan IOR.

We pilot-tested the system with simple shapes like squares, triangles, hearts, crosses, etc. A shape was drawn by the human and followed by an affirmative communicative signal (positive reward). By inspecting internal SOM representations, we verified that C-block was able to successfully learn plans. Each plan internally consisted of approx. 2-6 sequence elements (arcs) on average.

The plans were then tested in three ways: by collaborative drawing, by setting a desired perceptual state as a goal, and in a free drawing mode. In the first case, the human user drew 1-2 arcs of one of the previously drawn shapes and paused. For all seen shapes, C-block correctly identified the right plan and completed the drawing. In the second case a predefined bitmap similar to one of the learned shapes was given as a goal (Desired state). Again, the C-block was able to correctly select and execute a plan achieving the desired goal. Free drawing simply meant that after a period of inactivity BabyX autonomously started to replay its stored plans.

We also tested if the C-block can achieve a goal by consecutive execution of several partial plans. For example, we taught the C-block to draw four sides of a square individually (pausing and giving a reward after each line). Then we gave the bitmap of a square as the desired shape. The C-block was able to select relevant plans one by one, draw a complete square and then stop.

IV. Discussion

A. Related work

Although we demonstrated the C-block's operation on low-level sequencing of motor gestures, the same machinery can be applied to event segmentation on any level of granularity. It is useful to compare our model with two existing (and related) computational models of event segmentation. The model of Reynolds and Zacks [15] learns to recognize and chunk motor sequences from 3-dimensional motion captures of individuals performing routine tasks such as chopping down a tree, of 3-4 seconds in duration. The core of their model is the simple recurrent network (SRN) [14], trained with back-propagation of error for the next element prediction. Like us, they augmented the SRN's input with a tonic signal representing the whole event, which significantly helped to stabilise the prediction of the event's elements. The main difference between the two models is that, our model, thanks to its use of modified SOMs, is able to reconstruct the declarative representation of the whole sequence before it is finished. More importantly, while a SRN trains slowly using backpropagation, the C-block's learning is incremental and fast.

The model of Gumbsch *et al.* [16] was trained to learn events in a virtual scenario with an agent manipulating objects of a different type (foe, food), colour and weight (light, heavy). During observation, the model maintained a collection of active forward models matching the observed sequence and updates it at event boundaries. The forward models were trained by means of Recursive Least Squares. The main difference between their model and ours is that, instead of maintaining the relevant prediction models explicitly, our model combines their prediction implicitly during Bayesian inference in its SOMs.

Relevant to the wider context of BabyX architecture, Dominey and Warneken [17] present a system for experiments in human-robot collaboration. Their experiment 4 (interrupting a cooperative game) is similar to ours (Sec. II-F). Both systems share the idea that the abilities to observe an action and recognize its goal are important ingredients of the human ability to cooperate with others. Due to the complexity of both systems, we will not compare them in detail in this paper.

B. Limitations and future work

In the presented experiments, we used raw bitmap images of the drawing board as the world state. The drawing board can be thought of as a verbatim external memory. Because of the Euclidean distance function inside the SOMs, the pixel-wise comparison of the the learned plan effects inside the Planner is not translation invariant, so for a plan to be selected, its stored result image must closely match the goal image in size and position. This problem could be overcome by substituting the Euclidean distance with a more sophisticated function insensitive to translation and scaling. Using a semantically more derived perceptual representation (e.g. an embedding of a CNN visual classifier) as the world state would also increase the generalization abilities of the Planner.

In general, the perceptual result of a motor sequence need not be visual; for example, the result of playing a musical instrument is a tune/melody. We plan to extend our model with perceptual consequences in other sensory modalities.

In the scenario described in this paper, the C-block learned by observing a human user, but nothing prevents it from learning from its own movements (generated either by noise-based exploration around the learned plans, or by a random motor babbling). The source of reward can be external, when the user praises the baby for a nicely drawn shape, but it can be also internal. The baby can try to classify the visual results of her own motor babbling and, if the result is recognized as a known visual shape, it will be rewarded and the sequence will be learned as a successful plan. Novelty, surprise and other types of intrinsic motivation [18] are possible too.

C. Conclusion

We presented a model of perceptuomotor chunking that learns incrementally and very fast. The learned declarative representation of motor sequences can serve as grounded meaning for sentences describing actions of creation. The model can be used for behaviour control, planning and collaborative activities. We believe that the BabyX platform provides interesting possibilities for studying development of cognition in an interactive setting.

References

[1] M. Z. Alom, T. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. Nasrin, M. Hasan, B. Essen, A. Awwal, and V. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, p. 292, 03 2019.

[2] M. L. Anderson, "Embodied cognition: A field guide," *Artificial Intelligence*, vol. 149, no. 1, pp. 91–130, sep 2003.

[3] A. Glenberg and P. Kaschak, "Grounding language in action," *Psychonomic Bulletin and Review*, vol. 9, no. 3, pp. 558–565, 2002.

[4] L. Barsalou, "Grounded cognition," *Annual Review of Psychology*, vol. 59, pp. 617–645, 2008.

[5] L. Smith and M. Gasser, "The development of embodied cognition: Six lessons from babies," *Artif. Life*, vol. 11, no. 1–2, p. 13–30, 2005.

[6] M. Sagar, D. Bullivant, O. Efimov, M. Jawed, R. Kalarot, P. Robertson, and T. Wu, "Embodying models of expressive behaviour and learning with a biomimetic virtual infant," in *4th International Conference on Development and Learning and on Epigenetic Robotics*, 2014, pp. 62–67.

[7] M. Sagar, M. Seymour, and A. Henderson, "Creating connection with autonomous facial animation," *Communications of the ACM*, vol. 59, no. 12, pp. 82–91, 2016.

[8] A. Knott, *Sensorimotor Cognition and Natural Language Syntax*. Cambridge, MA: MIT Press, 2012.

[9] M. Takac, L. Benuskova, and A. Knott, "Mapping sensorimotor sequences to word sequences: A connectionist model of language acquisition and sentence generation," *Cognition*, vol. 125, pp. 288–308, 2012.

[10] D. Ballard, M. Hayhoe, P. Pook, and R. Rao, "Deictic codes for the embodiment of cognition," *Behavioral and Brain Sciences*, vol. 20, no. 4, pp. 723–767, 1997.

[11] F. Polyakov, E. Stark, R. Drori, M. Abeles, and T. Flash, "Parabolic movement primitives and cortical states: Merging optimality with geometric invariance," *Biological cybernetics*, vol. 100, pp. 159–84, 02 2009.

[12] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.

[13] M. Takac, A. Knott, and M. Sagar, "SOM-based system for sequence chunking and planning," in *Proceedings of ICANN*, I. Farkaš, P. Masulli, and S. Wermter, Eds., 2020.

[14] J. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, pp. 179–211, 1990.

[15] J. Reynolds, J. Zacks, and T. Braver, "A computational model of event segmentation from perceptual prediction," *Cognitive Science*, vol. 31, pp. 613–643, 2007.

[16] C. Gumbsch, S. Otte, and M. V. Butz, "A computational model for the dynamical learning of event taxonomies," in *Proceedings of the 39th Annual Meeting of the Cognitive Science Society*, London, 2017.

[17] P. Dominey and F. Warneken, "The basis of shared intentions in human and robot cognition," *New Ideas in Psychology*, vol. 29, 12 2011.

[18] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in Neurorobotics*, vol. 1, p. 6, 2009.