Organizing Behavior into Temporal and Spatial Neighborhoods*

Mark Ring
IDSIA / University of Lugano / SUPSI
Galleria 1
6928 Manno-Lugano, Switzerland
Email: mark@idsia.ch

Tom Schaul
Courant Institute of Mathematical Sciences
New York University
715, Broadway, New York, NY 10003
Email: schaul@cims.nyu.edu

Abstract

The mot[1] framework (Ring, Schaul, and Schmidhuber 2011) is a system for learning behaviors while organizing them across a two-dimensional, topological map such that similar behaviors are represented in nearby regions of the map. The current paper introduces *temporal coherence* into the framework, whereby temporally extended behaviors are more likely to be represented within a small, local region of the map. In previous work, the regions of the map represented arbitrary parts of a single global policy. This paper introduces and examines several different methods for achieving temporal coherence, each applying updates to the map using both spatial and temporal neighborhoods, thus encouraging parts of the policy that commonly occur together in time to reside within a common region. These methods are analyzed experimentally in a setting modeled after a human behavior-switching game, in which players are rewarded for producing a series of short but specific behavior sequences. The new methods achieve varying degrees—in some cases high degrees—of temporal coherence. An important byproduct of these methods is the automatic decomposition of behavior sequences into cohesive groupings, each represented individually in local regions.

Background

The mot framework (Ring, Schaul, and Schmidhuber 2011) is a system for continual learning (Ring 1994) in which behaviors are organized into a two-dimensional map according to their similarity.[2] This organization was conjectured to convey many useful properties to the learning agent—properties such as robustness, non-catastrophic forgetting, and intelligent resource allocation. One method shown for achieving such a map in practice was by laying out reinforcement-learning modules—SERL modules (Ring and Schaul 2011)—in a two-dimensional grid and then updating these modules in local spatial neighborhoods, much like the nodes of self-organizing maps (SOMs) are updated in local spatial neighborhoods (Kohonen 1988). As a result, the maps show spatial *smoothness*, the property underlying most of the advantages conjectured. Our goal in the current paper is to achieve *temporal* smoothness as well, such that temporally extended, coherent behaviors tend to be represented in small, local regions of the map. The methods presented here for achieving temporal smoothness apply updates not just to local spatial neighborhoods but to local temporal neighborhoods as well.

The mot framework was inspired by recent evidence in neuroscience that the motor cortex may be laid out as a topological map organized according to behavior, where similar behaviors occur close together and very different behaviors lie far apart (Graziano and Aflalo 2007; Graziano 2009). As described by Ring et al. (2011b), this organization in and of itself conveys many surprising advantages, including: smoothness (the closer two regions are, the more likely they are to represent similar behaviors, thus providing a gradient in behavior space); robustness (should a failure occur, nearby regions can provide similar behavior); efficient localization of learning (learning can be done simultaneously across related regions); hierarchical organization (large regions tend to represent generic behaviors, smaller regions represent more specific behaviors); safe dimensionality reduction (only those sensorimotor connections needed by a region are delivered there, but all connections remain accessible somewhere in the map); intelligent use and reuse of resources (obsolete behaviors can be replaced by similar ones, and new behaviors can be learned in those regions already representing the most similar behaviors); state aggregation by policy similarity (the position in the map of the currently active behavior provides a compact representation of certain state information); continual learning (new learning is compatible with and builds on top of old learning); and graceful degradation (regions compete to best cover the useful behavior space).

The *Motmap* is the two-dimensional sheet of mots whose purpose is to achieve the above advantages for an artificial agent. Each mot has a location in the map where it receives its input and computes its output. While the mot framework is quite general and allows a large number of possible instantiations, the system underlying the methods discussed here is exactly that described in detail by Ring et al. (2011b).

*This is a revised and corrected version of a paper appearing under the same title at ICDL-EpiRob (2012).

[1] Pronounced "mōt" or "moʊt", like moat or mote, rhyming with "boat" as in "motor boat".

[2] "Continual learning" refers to the constant and incremental acquisition of new behaviors built on previously acquired behaviors, where each behavior is learned through interaction with an environment that can provide positive and negative rewards.

It is composed of a fixed number of SERL modules that learn to combine their individually limited capacities to represent a complex policy. If there are more modules than necessary, they redundantly represent large areas of behavior space (thus increasing robustness). If in the more common case there are too few modules, they spread out to cover the most important areas best.

The learning rule encourages smoothness, and the map becomes organized such that nearby mots compute similar outputs to similar inputs. Unfortunately, however, this organization does not imply that the behaviors represented in a region will be temporally cohesive, or that the input-output pairs of any frequently occurring sequential behavior will likely be represented within the same region, as seems to be evidenced by the motor cortex (Graziano and Aflalo 2007; Graziano 2009). Thus, the current paper introduces new mechanisms to encourage temporally extended behaviors to be represented in small, local regions of the map.

Formal Description

In the current system, each mot is implemented as a single SERL module, extended with a coordinate on a two-dimensional grid (Figure 1, left). Since neither SERL nor the mot system are widely known, we repeat their formal description here.

SERL is a multi-modular system for reinforcement learning (RL). In the standard RL framework (Sutton and Barto 1998), a learning agent interacts with a Markov decision process (MDP) over a series of time steps $t \in \{0, 1, 2, ...\}$. At each time step the agent takes an action $a_t \in \mathcal{A}$ from its current state $s_t \in \mathcal{S}$. As a result of the action the agent transitions to a state $s_{t+1} \in \mathcal{S}$, and receives a reward $r_t \in \mathbb{R}$. The dynamics underlying the environment are described by the state-to-state transition probabilities $\mathcal{P}^a_{ss'} = Pr\{s_{t+1}{=}s' \mid s_t{=}s, a_t{=}a\}$ and expected rewards $\mathcal{R}^a_{ss'} = \mathbb{E}\{r_{t+1} \mid s_t{=}s, a_t{=}a, s_{t+1}{=}s'\}$. The agent's decision-making process is described by a policy, $\pi(s, a) = Pr\{a_t{=}a \mid s_t{=}s\}$, which the agent refines through repeated interaction with the environment so as to maximize $Q(s, a) = \mathbb{E}\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a\}$, the total future reward (discounted by $\gamma \in [0, 1]$) that it can expect to receive by taking any action a in any state s and following policy π thereafter.

SERL is an online, incremental, modular learning method that autonomously delegates different parts of a reinforcement-learning task to different modules, requiring no intervention or prior knowledge. Each module $i \in \mathcal{M}$ receives as input an observed feature vector $\mathbf{o} \in \mathcal{O}$, which uniquely identifies the state. Every module contains two components: a controller function,

$$f^{c,i} : \mathcal{O} \rightarrow \mathbb{R}^{|\mathcal{A}|},$$

which generates a vector of action-value estimates; and an expertise estimator (also called "predictor function"),

$$f^{p,i} : \mathcal{O} \rightarrow \mathbb{R}^{|\mathcal{A}|},$$

which generates a vector of predicted action-value errors. At every time step, each module produces values based on the current observation vector, $\mathbf{o}_t$:

$$\mathbf{q}^i_t = f^{c,i}(\mathbf{o}_t)$$
$$\mathbf{p}^i_t = f^{p,i}(\mathbf{o}_t)$$

These are combined for each module to create an $|\mathcal{M}| \times |\mathcal{A}|$ matrix L_t of *lower confidence* values such that

$$L^i_t = \mathbf{q}^i_t - |\mathbf{p}^i_t|,$$

where L^i_t is the i^{th} row of L_t.

At every time step there is a winning module, w_t, which is generally one whose highest L value matches L^*_t, the highest value in L_t. But this rule is modified in an ε-greedy fashion (Sutton and Barto 1998) to allow occasional random selection of winners, based on a random value, $x_t \sim U(0, 1)$:

$$W_t = \{i \in M : \max_a L^{ia}_t = L^*_t\}$$

$$Pr\{w_t = i \mid L_t\} = \begin{cases} \frac{1}{|M|} & \text{if } x_t < \varepsilon_M \\ \frac{1}{|W_t|} & \text{if } x_t \geq \varepsilon_M \text{ and } i \in W_t \\ 0 & \text{otherwise,} \end{cases}$$

where L^{ia}_t is the value for action a in L^i_t. Once a winner is chosen, SERL calculates an ε-greedy policy based on the winner's L values: $L^{w_t}_t$, using a potentially different constant, ε_A.

Learning. The function approximators for both controllers and expertise estimators are updated with targets generated by TD-learning (Sutton 1988). Each mot is a single SERL module assigned a coordinate in an evenly spaced, two-dimensional grid. Whereas in SERL, only the winner's controller is updated, the mot system updates the controllers for a set of mots w^+_t that surround the winner in the Motmap within a given radius.

The controllers are updated using Q-learning (Watkins 1989); thus for each $i \in w^+_t$ the target for $\mathbf{q}^{ia_t}_t$ (the component of $\mathbf{q}^i_t$ corresponding to action a_t) is $r_t + \gamma L^*_{t+1}$.

Every module's expertise estimator is updated at every step; its target is the magnitude of the controller's TD error:

$$\boldsymbol{\delta}^i_t = r_t + \gamma L^*_{t+1} - \mathbf{q}^{ia_t}_t, \forall i \in \mathcal{M}.$$

Modules differentiate themselves due to random initial weights. Since no module can solve the entire task alone, each develops expertise within a niche of the task space.

Temporal Coherence

The Motmap self-organizes such that each mot's policy mapping is more similar to its neighbors' than to mots farther away. However, the policies themselves are not necessarily *temporally* coherent: in the general case, if a mot has high expertise in a given state, it may not have high expertise in any of the immediately subsequent states. Conversely, for a learned global policy in which one state frequently or always follows immediately after another, there is no increased probability that the regions of greatest expertise for the two states are nearby each other. Indeed, the individual state-action mappings of extended behavior sequences are distributed arbitrarily throughout the Motmap. Thus, it cannot be argued that extended behaviors are represented within

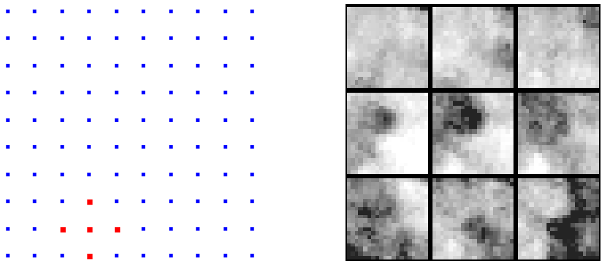

Figure 1: **Left**: A Motmap of 100 mots laid out in a two-dimensional grid. The red mots depict a learning update: the controllers and predictors of all mots within a certain radius around the winning mot are updated. For the other mots, only the predictors are updated. **Right**: Smoothly varying expertise levels (darker corresponds to greater expertise) across the motmap, on nine random observations. Both figures adapted from Ring, et al. (2011b).

local regions, as seems to be the case in the motor cortex. A temporally coherent organization, however, would be advantageous in the following ways:

Motor Vocabulary. Extended behaviors, such as picking up a glass, crawling or walking, brushing teeth, etc., are generally composed of smaller behaviors that are themselves useful, such as reaching the hand to a specific location, swinging a leg forward, grasping and holding, etc.—meaningful behaviors, that recur in a variety of situations for a variety of purposes. One of the goals of the mot framework is to learn such behaviors, to isolate them, and organize them for reuse. An important hypothesis of the system is that it is particularly beneficial to build up a vocabulary of small, useful motor behaviors that can be combined dynamically—much as words are combined to form desired meanings. Thus, it would be beneficial if each extended behavior could be located, learned, and accessed within a local region of the map.

Larger-scale smoothness. The Motmap encourages small-scale smoothness *for individual observations*: Expertise is usually concentrated somewhere in the map, gradually dropping off as distance from the center increases (see Figure 1, right). However, if two observations always occur in succession, their centers of expertise are no more likely to be near each other than if they never occur in succession. Larger-scale smoothness could be achieved if entire extended behaviors represented in one region were similar to extended behaviors represented by its neighbors.

Behavior-based hierarchy. One property expected from the Motmap (not yet demonstrated in publication) is the autonomous formation of hierarchies, in which smaller subregions represent more refined behaviors than the larger regions they make up. Without temporal coherence, these hierarchies are small scale: formed with respect to isolated observations only. Temporal coherence encourages the larger-scale correlate, where ever smaller regions represent ever more specific, more refined, extended behaviors.

Increased robustness. In the case of environmental noise, locality of behavior provides additional information for increasing robustness. If the agent preferentially chooses winners near the previous winner, it is more likely to remain within the region of state space appropriate to the current behavior.

Autonomous discovery of behavior. Perhaps the most tantalizing goal of temporal coherence is the automatic discovery of useful behaviors—the words of the motor vocabulary. We hypothesize that the continual learning of motor skills is not so much stringing together small motor skills into larger ones, but finding an ever more refined set of useful skills. That is to say, it is more about subtlety than sequencing.

But how can a good vocabulary of behaviors be discovered? How can we, as Plato (and many since) asked, "carve nature at its joints"—or in our case, carve an agent's behavior space at its joints? While we do not propose to have solved this long-standing problem, an important part of the answer may be for these motor *words* to form around the places where decisions must be made, drawing inspiration from Dawkins (Dawkins 1976).

Dawkins proposed a method for describing sequences hierarchically by focusing on decision points: the places where successors are less clearly determined by their predecessors. For example, in the sequence "AbcXyzAbcAbcXyzXyz", whenever A occurs, "bc" always follows; whenever X occurs, "yz" always follows. But there are no such deterministic successors for "c" or "z"; these are the decision points that suggest the boundaries between subsequences. The joints of behavior space are places where decisions must be made, where it is less obvious to the agent which action should occur next; i.e., where the entropy of the policy is highest. These are clues that a coherent behavior has ended and a new one can begin.

As adults we engage in an endless variety of short-term behaviors: we point, reach, touch, grasp, pick things up (each thing in its own way); we localize with our heads and eyes; we make gestures; we form a constant variety of precise tongue and mouth positions to speak, body postures and leg movements to get from place to place, and on and on. Our range of short-term behaviors is considerable. We choose each in context, combining them to achieve our desires of the moment.

But how can these meaningful components emerge through learning? Our approach is to encourage similar or temporally contiguous *responses* (state-action pairs) to be stored nearby each other.

The planning, choosing, learning loop. We hypothesize the following scenario for early learning. When it is not obvious what to do next, short behavior sequences are chosen through planning: the agent finds actions to exploit the regularities in its immediate environment (and due to these regularities, the agent frequently produces similar plans). It then acts on them and must plan again. Boundaries emerge *automatically* as a result of planning, decision making, and the statistical regularities of the environment.

Implementation and Testing

To encourage the mots to represent temporally contiguous portions of the global policy, we introduce, test, and compare six new update mechanisms. In all cases, the expertise estimators are updated as above; only the controller updates are modified here.

- Method W^{-t}: at time t all controllers in $w^+_{t-1} \cup w^+_t$ are updated using $r_{t-1} + \gamma L^*_t$ as the target. This means that if two mots tend to be temporal neighbors, they will often get trained on the same data, until one of them dominates on both.
- Method W^{-t+}: same as Method W^{-t}, but the controllers in w^+_{t+1} are also updated, making the method symmetric with respect to the past and future. (This method of course necessitates keeping the target until $t + 1$.)
- Method W^{t+}: same as Method W^{-t+}, but the controllers in w^+_{t-1} are *not* updated.

The remaining three mechanisms are more aggressive variants of the above: $W^{t+}_{2\alpha}$, $W^{-t+}_{2\alpha}$, and $W^{-t}_{2\alpha}$, in which the learning rate for the controllers in w^+_{t-1} and/or w^+_{t+1} is twice that for w^+_t. The intuition here is to force the temporally adjacent winners to assume some of the expertise from w^+_t.

Benchmark task. The video game *Dance Central* provides a useful illustration of the planning, choosing, and learning loop. In the game, the screen shows a picture representing which dance move the player should perform next. The player, whose actions are analyzed by computer as part of the Kinect$^{\text{TM}}$ gaming system, receives points for performing the sequence correctly. The game then displays a different movement sequence from a fixed library, and the process continues until the song is over. The player thus learns a broad range of different motor-control behaviors, each having sequential contiguity, punctuated by decision points in which new plans and decisions are made.

We model the game as an MDP, where each of D dance sequences is represented as a chain of N states: the starting state is at one end of the chain and the final state is at the other. In each state of the chain, the agent can choose one of A actions; a single "correct" action advances it to the next state in the chain, while all other actions take the agent back to the previous state. When the agent reaches the end state of a chain (which corresponds to successfully producing the dance sequence), it receives a reward of 1.0 and is placed at the starting state of a randomly chosen chain. (All other state transitions return a reward of zero.) The agent's observation in each state is a feature vector comprising two concatenated, binary unit subvectors: the first subvector identifies the task, while the second identifies the current step within the task. (Each subvector has a 1 in a single position, all other positions are 0.) This abstract representation captures at a high level the agent's overall goal of mapping a target dance sequence and a current progress indicator to a discrete action.

Before training begins, the actions that take the agent to the next state are assigned randomly, independently and uniformly. Therefore, the agent cannot deduce the correct action based on regularities in the inputs. For each behavior the agent must learn to produce a string of actions that are temporally coherent, but once the behavior is finished, a new one is selected at random. This scenario allows us to test the mechanisms proposed above for their ability to capture the temporal coherence of the learned behaviors and to assign them to nearby locations of the Motmap.

We examined games of different sizes, varying the number of dance sequences (D), their length (N), and the number of possible actions (A).

Measures of coherence. For the comparisons we used two measures of temporal coherence: (P) the probability of switching to a mot outside the current winner's local neighborhood, and (H) the entropy of a sequence of mot winners. For measure (P), we compared two values: (P_s) the probability of a switch during the sequence, and (P_e) the probability of a switch at the end of the sequence (when a new dance sequence is chosen); thus,

$$P_s = \frac{1}{D(N-1)} \cdot \sum_{d=1}^{D} \sum_{n=1}^{N-1} \begin{cases} 0 & \text{if adjacent}(w_{d,n}, w_{d,n+1}) \\ 1 & \text{otherwise,} \end{cases}$$

$$P_e = \frac{1}{D(D-1)} \cdot \sum_{d=1}^{D} \sum_{d' \neq d}^{D} \begin{cases} 0 & \text{if adjacent}(w_{d,N}, w_{d',1}) \\ 1 & \text{otherwise,} \end{cases}$$

where adjacent(x, y) is true if mot x is immediately above, below, left or right of mot y on the Motmap, and $w_{d,n}$ is the winning mot when the agent's input is the observation from the n^{th} step in dance sequence d.

For measure H, we define the entropy of a sequence as :

$$H(s) = -\sum_{i}^{M} p^s_i \log_M p^s_i,$$

where M is the number of mots, and p^s_i is the fraction of states in sequence s in which mot i is the winner. (If a single mot is the winner for every state in the sequence, the entropy is zero.) Then we compare H_d (the average entropy of those dance sequences to be learned) with H_r (the expected entropy of a random dance sequence):

$$H_d = \frac{1}{D} \sum_{d=1}^{D} H(s_d)$$

$$H_r = \mathbb{E}\left[H(s_r)\right],$$

where s_d is the d^{th} dance sequence to be learned, and s_r is a randomly generated sequence where for step i of the sequence, each observation is drawn uniformly from the i^{th} step of all D sequences. The expectation in H_r is approximated by averaging over 100 such sequences.

Results

Results are shown in Figure 2 for two variants of the mot system: (1) with update radius 1 (corresponding to one neighbor of the winner in each cardinal direction), denoted as the 'Motmap' scenario; and (2) with update radius 0 (no neighbors), denoted as the 'SERL' scenario. (There is no spatial organization in the latter scenario, where we study purely the dynamics of the temporal coherence.) In each case we test the six new update mechanisms from the last section on the same task. The task uses 16 dance sequences ($D = 16$), each dance sequence representing a behavior of length 4 ($N = 4$), with 10 actions possible in each state ($A = 10$). The results of all 14 variants are shown. Figure 3 then shows the results of the best method on three different task settings.

The parameter values were: $\alpha_c = 0.005$ (learning rate for controllers), $\alpha_e = 0.05$ (learning rate for expertise estimators), $\varepsilon_A = 0.02$ (exploration on actions), $\varepsilon_M = 0.5$ (exploration on mots), $\gamma = 0.95$ (reward discount), and $M = 100$ (number of mots).

Discussion

The results demonstrate temporal coherence emerging from several of the methods tested, visible both through the low entropy and the low switching probability within a sequence, in comparison to the relatively high entropy of the random sequence and the relatively high switching probability between different sequences. This shows two things. First, behavior sequences are coalescing within local regions of the map. Second, the boundaries of the behaviors are being discerned and captured as an emergent property of the system; *i.e.,*the mots are learning to carve the agent's behavior at its joints.

Interestingly, it seems that temporal coherence can be achieved without harming task performance—compare the black curves from the top row in Figure 2 to the rows below. Thus, all the benefits of temporal (and spatial) organization outlined above can be achieved nearly for free.

The shape of the two probability curves P_s and P_e depends on the update method used. Without an explicit method for encouraging temporal coherence (top row), these curves are essentially the same. Thus, the probability of switching to a mot outside the current winner's local neighborhood is not lower during the course of a dance sequence than when changing to a new sequence. In contrast, all the temporal-coherence methods successfully reduce P_s (intra-sequence switching) and H_d (intra-sequence entropy) without drastically reducing P_e (inter-sequence switching) nor H_r (random-sequence entropy). In both the Motmap and SERL case, $W^{t+}_{2\alpha}$ has the greatest impact.

In each graph a phase transition can be seen in which a sudden reduction in P_s and H_d coincide with an increase in the number of correct action choices. Before learning the sequence well, the agent generally takes many actions to reach the end; thus it repeatedly experiences the same small set of observations, and the updates are applied consistently to a small number of winning mots. However, once the agent has learned the sequence, it spends less time within sequences and relatively more time at the ends of sequences, where it is exposed to a broader range of mot winners at the following step. The agent then applies relatively more updates to the winners dedicated to one sequence with values from the following (unrelated) sequence, which may explain the gradual loss in coherence visible in some graphs after a good policy is learned. Keeping high values of ε helps to avert the loss by guaranteeing more intra-sequence updates even after the

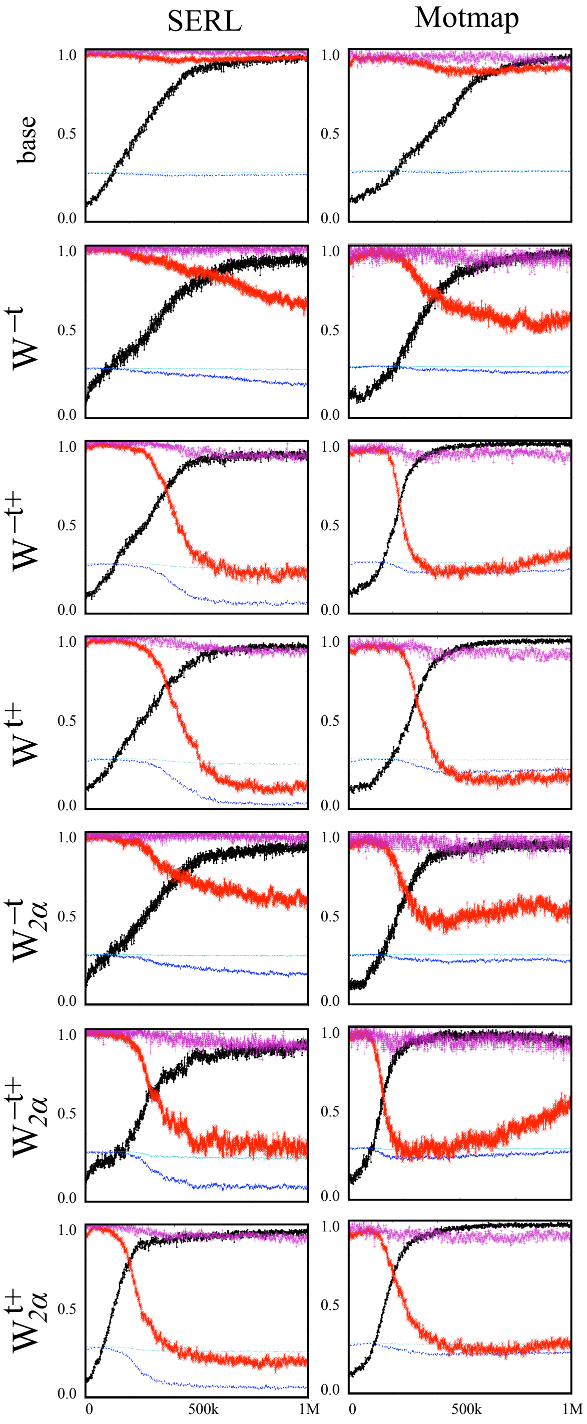

Figure 2: Average performance curves (10 trials) for one million update steps (horizontal axis), with 14 different settings. Left column is the SERL scenario, right column the motmap scenario. Baseline performance (having no temporal update mechanism) is shown in the first row, one mechanism for each of the next six rows. For each graph: vertical axis is 0 to 1; black shows the percentage of actions the agent chooses correctly; red shows P_s (see text), to compare to its reference value P_e (purple); and dark blue measures H_d (reference value is the light-blue line, H_r).

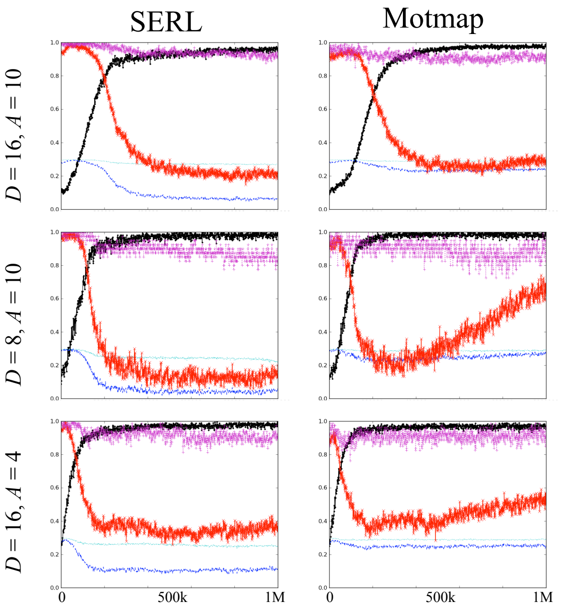

Figure 3: Results using the best mechanism found ($W_{2\alpha}^{t+}$), but now across three different tasks, where $N = 4$ in all cases and the first row coincides with the default task from before. See Figure 2 for plot details.

optimal policy has been learned (thus the high value we use for ε_M, the probability of switching to a random mot).

In addition to the six update mechanisms presented here, we tried numerous others that were less successful, including: (a) sharing eligibility traces Q(λ) among spatio-temporal neighbors; (b) giving preference to the previous winner(s), boosting their probability of being chosen again.

The goal of carving behavior at its joints is not new in AI or RL. A different method proposed for segmenting RL policies into meaningful chunks is to find bottleneck states (“doorways”) through which many possible paths can pass (Mcgovern and Barto 2001). These methods bear a certain resemblance to the current method, in that doorways are also likely to be states of higher entropy.

Conclusions

The temporal and spatial organization of behavior is of great potential benefit for continual-learning agents, promoting increased robustness, hierarchical learning, and the navigation and search of learned behaviors. Building upon our earlier work, which had introduced mechanisms for the *spatial* organization of behavior in a two-dimensional “Motmap,” the current paper introduced six related, novel update mechanisms for achieving *temporal* coherence in SERL and the Motmap. In each case, this coherence is achieved as an emergent property of the update rule.

The new mechanisms can segment behavior into cohesive chunks, representing each chunk within individual modules of a SERL system or within local neighborhoods of the two-dimensional Motmap. Both results were demonstrated using a variable set of sequential tasks patterned after the game Dance Central. The latter result, a topological map organized in both space and time, is an important step towards an agent that maintains a library of useful motor behaviors—ordered, accessible, and extensible according to their similarities. We posit that this organization will be critical for a continual-learning agent that is constantly expanding the sophistication of its behavior.

Acknowledgments

Funded in part through EU project IM-Clever (231722) and through AFR postdoc grant number 2915104, of the National Research Fund Luxembourg. We thank the anonymous reviewers for their thoughtful comments.

References

Dawkins, R. 1976. Hierarchical organisation: a candidate principle for ethology. In Bateson, P. P. G., and Hinde, R. A., eds., *Growing Points in Ethology*, 7–54. Cambridge: Cambridge University Press.

Graziano, M. S. A., and Aflalo, T. N. 2007. Rethinking cortical organization: moving away from discrete areas arranged in hierarchies. *Neuroscientist* 13(2):138–47.

Graziano, M. 2009. *The Intelligent Movement Machine: An Ethological Perspective on the Primate Motor System*. Oxford University Press.

Kohonen, T. 1988. *Self-Organization and Associative Memory*. Springer, second edition.

Mcgovern, A., and Barto, A. G. 2001. Automatic discovery of subgoals in reinforcement learning using diverse density. In *In Proceedings of the eighteenth international conference on machine learning*, 361–368. Morgan Kaufmann.

Ring, M., and Schaul, T. 2011. Q-error as a Selection Mechanism in Modular Reinforcement-Learning Systems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. To appear.

Ring, M., and Schaul, T. 2012. The organization of behavior into temporal and spatial neighborhoods. In *Development and Learning and Epigenetic Robotics (ICDL), 2012 IEEE International Conference on*, 1–6.

Ring, M. B.; Schaul, T.; and Schmidhuber, J. 2011. The Two-Dimensional Organization of Behavior. In *First Joint IEEE International Conference on Developmental Learning and Epigenetic Robotics*.

Ring, M. B. 1994. *Continual Learning in Reinforcement Environments*. Ph.D. Dissertation, University of Texas at Austin, Austin, Texas 78712.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Sutton, R. S. 1988. Learning to predict by the methods of temporal differences. *Machine Learning* 3:9–44.

Watkins, C. J. C. H. 1989. *Learning from Delayed Rewards*. Ph.D. Dissertation, King’s College.