

# Learning Fine Motion in Robotics: Experiments with the Hierarchical Extended Kohonen Map

Cristina Versino†, Luca Maria Gambardella†  
† IDSIA, Corso Elvezia 36, 6900 Lugano, Switzerland  
cristina@idsia.ch, <http://www.idsia.ch/~cristina>

**Abstract**— We present a Hierarchical Extended Kohonen Map (HEKM) and a planning system which cooperate to solve the robot path finding problem. The HEKM learns to associate appropriate actions to perceptions under the supervision of the planner. First, we argue for the utility of using the hierarchical version of the KM instead of the “flat” KM: the HEKM provides a natural and economic representation of the robot’s perceptual states. Second, we measure the benefits of cooperative learning due to the interaction of neighboring neurons in the HEKM: with cooperation, learning is slowed down in the short run, but the benefits appear later on, resulting in a more satisfactory final performance. Third, we highlight a beneficial side-effect obtained by transferring motion skill from the planner to the HEKM, namely, smoothness of motion.

## 1 Introduction

The problem of *path finding* has attracted considerable attention by robotics research. This is the problem of moving a robot from a starting position to a goal position avoiding collisions against obstacles in the workspace. Moreover, the robot path should be as *short* and *smooth* as possible. The path is *optimal* if it is the shortest from the starting position to the goal.

Path finders are methods to automatically solve the path finding problem. Traditionally, path finders are either *model-based* or *sensor-based*. While model-based systems address the path finding problem *globally* using a model of the workspace, sensor-based systems consider it *locally*, and rely on robot sensors to avoid obstacles. Both methods have limitations, which are rather complementary. Model-based systems compute optimal free-paths and recover easily from dead-ends, but require a complete description of the robot workspace and are computationally expensive. Sensor-based systems do not need a model of the workspace and require limited computation, but produce sub-optimal paths and may get trapped into dead-ends. In addition, they are difficult to program.

By integrating model-based and sensor-based methods, we can mitigate their respective drawbacks. Thus, in [10] we have described a planner working on an artificial potential field (a model-based system) and a Hierarchical Extended Kohonen Map (a sensor-based system) which cooperate to solve the path finding problem. Along related lines, several authors [7, 4, 5, 6] have proposed to automatically build the sensor-based system as the result of a learning process, where a local planner plays the role of the teacher. All the quoted works present neural networks of *distance-based* units that learn fine motion. Some of them, use self-organizing networks. For example, [4, 6] employed a Self-Organizing Map (SOM) and [5] used a dynamical variant of SOM (DSOM) based on a Growing Neural Gas network [2]. In these works, the preference for SOM-like networks seems to be justified by their *data topology-conserving* character which is supposed to favor in some way the learning of suitable  $\langle \textit{perception}, \textit{action} \rangle$  pairs. Surprisingly, none of these works provide experimental evidence for this reasonable, but not obvious, claim.

In this paper we describe a SOM-like neural network which solves path finding problems in cooperation with a planning system. The network learns to associate actions to perceptions under the supervision of the planner. By reporting this experiment we make the following contributions. *First*, we argue for the utility of using a hierarchical version of SOM instead of the basic SOM. *Second*, we measure explicitly the effect of cooperative learning due to the interaction of neighboring neurons. *Third*, we highlight a beneficial side-effect which can be obtained by transferring motion knowledge from the planner to the SOM.

## 2 A Hierarchical Extended Kohonen Map to learn fine motion

In our experiment the self-organizing network learns  $\langle \textit{perception}, \textit{action} \rangle$  pairs produced by the planner [3] while solving instances of the path finding problem. In this experiment we do not consider the problem of recovering from local minima. A *perception* is made of a vector  $o$  of readings of 24 obstacle proximity

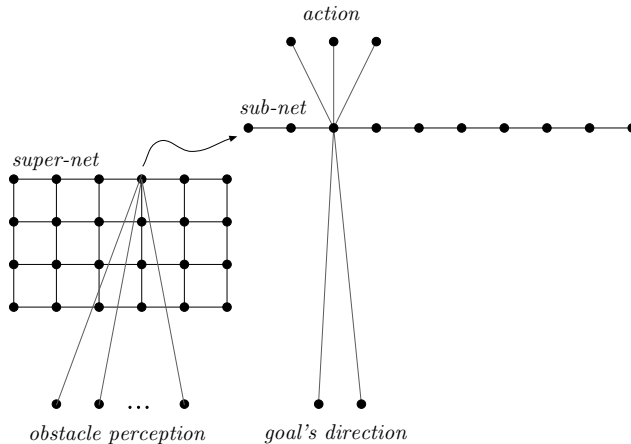


Figure 1: The HEKM network architecture.

sensors, together with the relative goal direction  $g$ , a 2 dimensional vector of unitary length. A planner *action*  $a$  is a triple representing, an  $x$ -translation, a  $y$ -translation, and a rotation with respect to the robot's current position and orientation. Both the  $xy$ -translations and the rotation take *discrete* values, and can be either positive, negative or null.

The SOM is a Hierarchical Extended Kohonen Map (HEKM) [8]. The map is “hierarchical” because it processes the input perception in sequence (this is explained in more detail later); it is “extended” because it is trained on the output action in a supervised fashion. Essentially, the network operation is divided into two steps (Figure 1). Given an input perception, the HEKM first determines which is the most similar perception out of the ones experienced sofar (matching step); and second, it triggers the action associated to that perception (triggering step). The perception matching step is carried out in two stages as well. First,  $o$  is processed by a KM *super-net*. Second,  $g$  is processed by a KM *sub-net* which is associated to the winning neuron in the super-net. Therefore, the overall network architecture is a hierarchical arrangement of subordinated sub-nets: there is a sub-net for each neuron in the super-net. The network action is retrieved as the result of this two-stage competition process.

During training, the proposed  $\langle \textit{perception}, \textit{action} \rangle$  pair is learnt by the network through the basic Kohonen's rule. A learning step involves the winner in the super-net, the winner in the corresponding sub-net, and their *neighbors* on the networks as well. To be more specific,  $o$  is learnt by the winning neuron in the super-net;  $g$  is learnt by modifying the fan-in weight vector of the winning neuron in the sub-net; finally,  $a$  is learnt by modifying the fan-out weight vector of the winning neuron in the sub-net. This learning style has been described as a *competitive-cooperative* training rule [8]. It is competitive because neurons compete to respond to input patterns. As a consequence, only that part of the network which is relevant to the current input data undergoes the learning process, and the problem of catastrophic interference is reduced [9]. The rule is also *cooperative* in that the output learnt by the winning neuron is partially associated to the weight vectors of its neighbors to enhance generalization.

For the application to path finding, we have preferred a hierarchical architecture to a “flat” one for three reasons. *First*, it avoids unnecessary repetition of  $o$  weights for different  $g$  directions, which would be costly in terms of memory requirements. *Second*, it deals naturally with the economic input representation of  $g$  as a 2 dimensional vector. A flat network would need either a more distributed codification for  $g$  (as in [7]) or a weighting of  $g$  (as in [4, 5]) so that during the matching step  $g$  does not lose importance with respect to  $o$ , whose dimensionality is rather high. *Third*, by processing the input information in two stages, we hope to simplify the adaptation process of the SOM to the perception data distribution.

A portion of the weights of the trained HEKM is depicted in Figure 2. In this experiment, the super-net is a  $4 \times 6$  grid of neurons, while each sub-net is an array of 10 neurons. The upper drawing shows the super-net weights: they represent prototypical obstacle perceptions. As an example, unit<sup>1</sup> #0 represents the perception of free-space, unit #5 represents the perception of a wall on the right-hand side, unit #7 represents the perception of a wall behind the robot's back. It is possible to observe the *data topology*-

<sup>1</sup>Units are numbered from left-to-right and top-to-bottom.

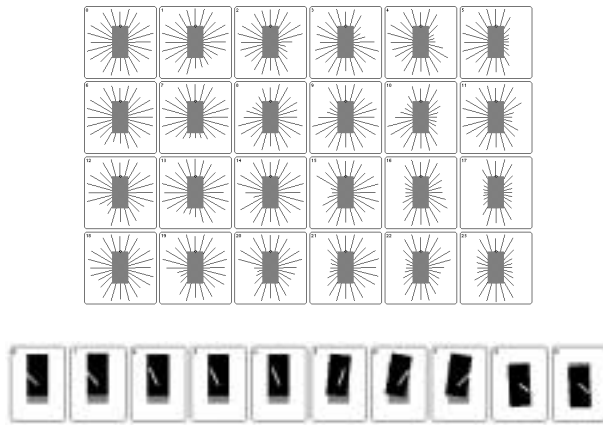


Figure 2: The obstacle perceptions learnt by the super-net (upper drawing). Goal directions and actions learnt by sub-net #17 (lower drawing).

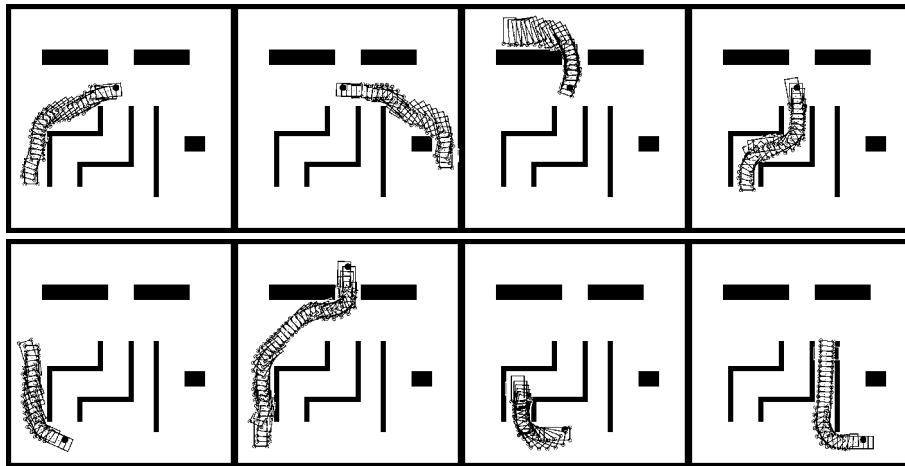


Figure 3: The robot solving the path finding problem with a fixed goal (first row) and with new goal positions (second row).

*preserving* character of the KM: perception similarity varies in a continuous way on the map. The lower drawing in Figure 2 shows the weights of sub-net #17, which is associated in the super-net to the perception of a narrow corridor. For each neuron, we represent the learnt goal direction (as a white vector) and the learnt action (the gray rectangle is the robot's initial configuration, the black rectangle is the robot's configuration after having performed the action). Again, the data topology-preserving character of the KM can be appreciated in this sub-net.

Figure 3 show some instances of path finding solved by the HEKM in cooperation with the planner. In these trajectories the planner takes control only when the action proposed by the HEKM would lead to a collision. In the first row of the Figure, the goal position (black circle) is fixed and it is the same used to generate the training examples for the HEKM. The second row depicts other trajectories with new goal positions. These runs prove that the motion skill acquired by the HEKM is independent from the chosen goal.

### 3 Why to use a SOM-like network?

We would like now to discuss the following claim: the data topology-preserving character of the HEKM could favor the learning of fine motion.

This statement can be proved experimentally by performing two separate training sessions. In the first session, the neighborhood parameters (one for the super-net, one for the sub-nets) are set to 0, while in second session they are set to values other than 0 (4 and 5, respectively). In this way, we can study the

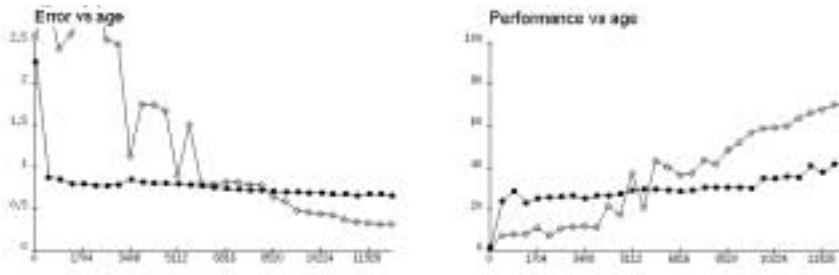


Figure 4: Error (left) and Performance (right) without cooperation (black dots) and with cooperation (white dots).



Figure 5: The planner (left) and the HEKM (right) working as stand-alone systems.

effect of cooperation during learning.

To evaluate the two methods, an error criterion and a performance criterion are used. The error measure is the mean squared error between the network output action and the target action proposed by the planner, while the performance criterion is the percentage of optimal actions learnt by the network. By definition, the optimal actions are those proposed by the planner.

Let us comment on the plots of error and performance as a function of the number of training cycles (Figure 4). As far as the error is concerned (left plot), one can see that without cooperation (curve with black dots) a certain error level is reached quite rapidly, but afterwards, no significant improvement is observed. On the contrary, with cooperation (curve with white dots) it takes more time to reach the same error level, but the final error is lower. This type of behavior seems to be typical for cooperating agents, as it reported in [1]. In our experiment, a possible explanation for this could be that, when the cooperation between the neurons is active, it takes more time to find a good “compromise” to satisfy competing learning needs. However, once the compromise is met, the final result gets improved. A corresponding behavior is observed in the performance curves (right plot). With no cooperation a certain performance level is achieved quite rapidly (42%), but after that point no further improvement occurs. With cooperation, the same performance level is obtained later, but the final result is more satisfactory (65%).

#### 4 Planner’s path versus HEKM’s path

We conclude by highlighting an interesting side-effect which can be obtained by transferring motion knowledge from the planner to the HEKM.

Our planner is a *discrete* system. By “discrete” we refer to the fact that, at each step of the robot trajectory, the planner generates a finite number of neighboring configurations, and chooses, among them, the one which approaches the goal closest while avoiding collisions. The HEKM, on the contrary, tends to produce actions which look like being *continuous*. That is because the action learnt by the network for a given perception is a kind of average action performed by the planner in similar perceptual states. To illustrate this point, we let the planner and the HEKM solve the same path finding problem as *stand-alone* systems (Figure 5). One can immediately appreciate qualitative differences in the two paths. The discrete nature of the planner is evident in the left plot: the robot motion is optimal in terms of path length, but quite abrupt. On the contrary, in the HEKM path (right plot) is smooth but not optimal. This observation can also account for the sub-optimal performance level reached by the HEKM (Figure 4) at the end of training.

## 5 Conclusions

We have presented a HEKM which learns fine motion under the control of a planner. *First*, we have discussed the utility of using a hierarchical KM instead of the usual “flat” version. The HEKM is more economic in terms of the way memory cells are used. It avoids unnecessary weight repetitions and allows for compact input representations. Clearly, one limitation of the current architecture is the fixed number of neurons and the fixed topology of the network. A growing network could be used instead [5, 2]. *Second*, we have measured the effect of cooperative learning due to the interaction between adjacent neurons. We found that *with cooperation* learning is slowed down in the short run. But the benefits appear later on, resulting in a more satisfactory final performance. Our interpretation is that, at the beginning of learning, neighboring neurons work to meet a compromise to competing needs: this effort becomes rewarding on the long run. *Third*, we have pointed out the complementary nature of the paths generated by the planner and by the HEKM as stand-alone systems. The HEKM produces sub-optimal but smooth solutions, whereas the planner seeks for optimality while sacrificing the continuity of motion. The integration of these two philosophies leads to fruitful results.

Our future work will include the implementation of these ideas on a physical robot.

## Acknowledgements

Cristina Versino is supported by the No. 2129-042413.94/1 project of the Fonds National de la Recherche Scientifique, Berne, Suisse. Thanks to Vicente Ruiz de Angulo for his comments on early drafts of this paper. Thanks to *Neuristique* (France) for providing the SN neural network simulator.

## References

- [1] Clearwater, S.H., Hogg, T., Huberman, B.A. (1992) Cooperative Problem Solving. In Huberman, B.A., Editor, *Computation: The Micro and the Macro View.*, World Scientific.
- [2] Fritzke, B. (1995) A Growing Neural Gas Network Learns Topologies. In Tesauro, G., Touretzky, D.S., Leen, T.K., Editors, *Advances in Neural Information Processing Systems 7*, MIT Press, Cambridge MA, pp. 625–632.
- [3] Gambardella, L.M., Versino, C. (1994) Learning High-Level Navigation Strategies from Sensor Information and Planner Experience. *Proc. PerAc94, From Perception to Action Conference*, Lausanne, Switzerland, September 7–9, pp. 428–431.
- [4] Heikkonen, J., Koikkalainen, P., Oja, E. (1993) Motion Behavior Learning by Self-Organization. *Proc. ICANN93, International Conference on Artificial Neural Networks*, Amsterdam, The Netherlands, September 13–16, pp.262–267.
- [5] Heikkonen, J., Millán, J. del R., Cuesta, E. (1995) Incremental Learning from Basic Reflexes in an Autonomous Mobile Robot. *Proc. EANN95, International Conference on Engineering Applications of Neural Networks*, Otaniemi, Espoo, Finland, August 21–23, pp. 119–126.
- [6] Knobbe, A.J., Kok, J.N., Overmars, M.H. (1995) Robot Motion Planning in Unknown Environments Using Neural Networks. *Proc. ICANN95, International Conference on Artificial Neural Networks*, Paris, France, October 9–13, pp. 375–380.
- [7] Millán, J. del R. (1995) Reinforcement Learning of Goal-Directed Obstacle-Avoiding Reaction Strategies in an Autonomous Mobile Robot. *Robotics and Autonomous Systems*, 15(3), pp. 275–299.
- [8] Ritter, H., Martinetz, T., Schulten, K. (1992) *Neural Computation and Self-Organizing Maps. An Introduction.* Addison-Wesley Publishing Comp.
- [9] Ruiz de Angulo, V., Torras, C. (1995) On-line Learning with Minimal Degradation in Feedforward Networks. *IEEE Transactions on Neural Networks*, Vol. 6, pp. 657–668.
- [10] Versino, C., Gambardella, L.M. (1995) Learning Fine Motion in Robotics by Using Hierarchical Neural Networks. *IDSIA-5-1995 Technical Report.*