Hebbian Learning In A Multimodal Environment

Julien Hubert, Eiko Matsuda and Takashi Ikegami

The University of Tokyo, Ikegami Laboratory, Tokyo, Japan
{jhubert,eiko,ikeg}@sacral.c.u-tokyo.ac.jp

Abstract

Hebbian learning is a classical non-supervised learning algorithm used in neural networks. Its particularity is to transcribe the correlations between couple of neurons within their connecting synapse. From this idea, we created a robotic task where 2 sensory modalities indicate the same target in order to find out if a neural network equipped with Hebbian learning could naturally exploit the relation between those modalities. Another question we explored is the difference in terms of learning between a feedforward neural network(FNN) and spiking neural network(SNN). Our results indicate that a FNN can partially exploit the relation between the modalities and the task when receiving a feedback from a teacher. We also found out that a SNN could not complete the task because of the nature of the Hebbian learning modeled.

Introduction

One important aspect of our everyday life is our capacity to acquire knowledge by experiencing our environment. Animals possessing the capacity to learn can detect and exploit the correlations present in their environment. This gives them, among other advantages, the capacity to predict their world and avoid undesirable outcomes, which is akin to Friston's free energy theory where living systems try to minimize their free-energy in order to increase their capacity of prediction of their environment(Friston (2010)).

The neurobiology of learning in the brain is driven by synaptic plasticity, also referred to as Hebbian plasticity or Hebbian learning after Donald Hebb who first proposed a theory of how learning could take place(Hebb (1949)). The general idea drawn from Hebbian learning states that when 2 neurons are connected together by a synapse, the correlated activity between the two would evoke structural modifications within the synapse such the capacity of the pre-synaptic neuron to cause potentiation in the post-synaptic one would increase(see Abbott and Nelson (2000) for a complete introduction). This phenomenon was observed at the neuronal level by Kelso et al. (1986).

The task we are interested in is a multi sensory integration task where a robot must reach a target indicated by a sound and a light source. The main question we want to address is if a neural network equipped with Hebbian learning can naturally extract and exploit the correlation between the two sources. Indeed, sound and light are two sensory modalities with different properties and different yield. It would then be interesting to know if Hebbian learning has the potential of integrating them transparently, that is without any external mechanism for this task. It is generally expected that Hebbian learning can detect correlations at the neuronal level, but our question is if it can also integrate the correlations present in the environment in order to convey additional properties or abilities to its host. For instance, in this particular task, we are interested in the robustness that Hebbian learning can provide when noise is added on the sensors. Another aspect we explore is the effects of asynchronous activation of the sensory modules on the behaviour of the robot. The particularity of our experiment is that no mechanism helping the integration of the two sensory modalities is provided, and any emergent property of the system can only be the result of the Hebbian learning.

As a side experiment, we were also interested in the comparison between classical rate based neural networks and spiking based neural networks when it comes to Hebbian learning. Spiking neural networks(SNN) are a relatively recent paradigm where information transferred between neurons is no longer a continuous value, but a temporally limited event representing a neuron discharging its membrane potential (Maass (1997)). SNNs represent a model closer to the neurobiology of the brain, and as such possess also a model of Hebbian learning directly drawn from neurophysiological recordings. The particularity of this learning is that it incorporates a window of time where a synapse can be modified, which differs from the Hebbian learning used in classical NN where the changes are instantaneous. Our question is whether this architecture would lead to different behaviors. We will see that SNNs could not reproduce the behavior obtained by the traditional neural network, and we will discuss the causes and the impact of this result.

The article is divided in 3 sections. The first presents the experimental setup and includes a description of the task and of the controllers. In the second section are presented the

results for both controllers which are discussed in the last section.

Experimental Setup

The Task

Our experiments use a robot whose task is to move toward a target area in its environment. The robot moves in an open environment where one light source and one sound source are located at the same position. Facing those sources is a grid of 7x7 cells(see figure 1). Those are used as starting positions for the robot. The task of the robot is to navigate its environment using its sensors until it reaches a point at a maximum distance of 1 grid cell from the sources. The robot has 6 minutes to complete this task. If the robot reaches the goal within that time period, the trial is considered a success; otherwise, the trial is counted as a failure.

The robot and its environment are simulated in all the experiments. The simulation of the robot is based on the e-puck robot equipped with light and sound sensors(Mondada et al. (2009)). The simulation of the environment is based on data obtained from a real robot in a real environment. A model of the light and of the sound has been built using sensor readings gathered from a robot moving in the environment. This has consequences on the simulated sensors. The simulated light sensor can not perceive the emitted light from the source outside of its field of view. The simulated sound sensor does not only perceive the sound from the source, but also the sound from the motors of the robot. The noise generated by the motors can be louder than the emitted sound, preventing the robot from perceiving it. The real source of the light is a white neon bulb while the real sound source emits white noise. Figures 2 and 3 show respectively the light and the sound perceived by the robot at the different positions in the grid, the goal being located at coordinates $(3.5, 7)$ on the graphs, i.e. at the center column of the top row. The frequency of update of the simulation depends on the neural network. In the case of the feedforward neural network, a timestep of 0.05s is used. For the spiking neural network, a timestep of 0.001s is used.

The performance of the robot is measured by its capacity to reach the goal in less than 6 minutes. For a single trial, the performance of a robot would be one if it reached the goal and zero otherwise. The duration of the trial is not considered in the performance measure.

The Controller

As mentioned in the introduction, two controllers are tested in this task. The first one is a standard feedforward neural network(FNN). The second is a spiking neural network(SNN). Both are equipped with plastic synapses following a Hebbian rule. The number of input, hidden and output neurons remains the same with both controllers. The next subsection describes the implementation using the feedforward network. The differences with the spiking neural network will be explained later on. The next section will detail the FNN.

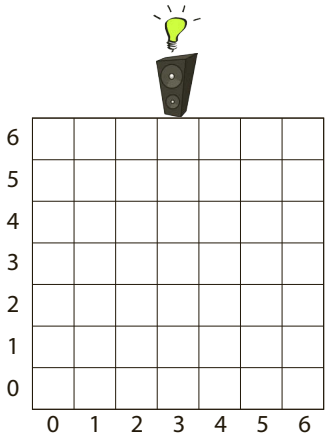

Figure 1: Experimental arena. Each cell of the grid is a possible starting position for the robot.

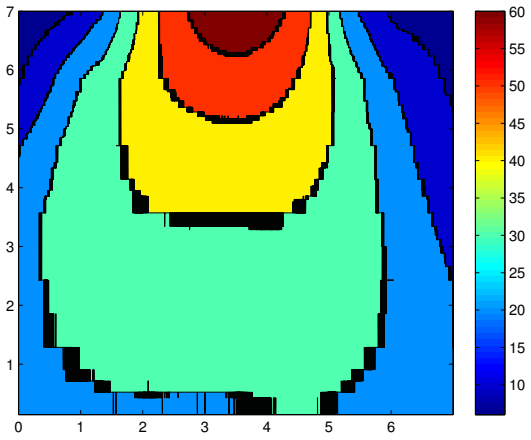

Figure 2: Robot's perception of the light.

Feedforward Neural Network The FNN possesses 7 inputs and 4 outputs. No hidden neurons are present. The input and output layers are fully interconnected. The weights are tuned using Hebbian learning(Hebb (1949)). A visualization of the network is shown in figure 4.

The inputs of the FNN are divided into two groups. The first group is composed of 4 inputs and handles the perception of the sound. The second group contains 3 inputs and receives the perception of the light. In both cases, the inputs are not the outputs of the sensors, but a pre-processed version. Each sensor is connected to a memory containing the past readings of each sensor. In the case of the sound, the size of the memory is 30 readings. For the light, it is only 10 readings. The difference in memory size can be ex-

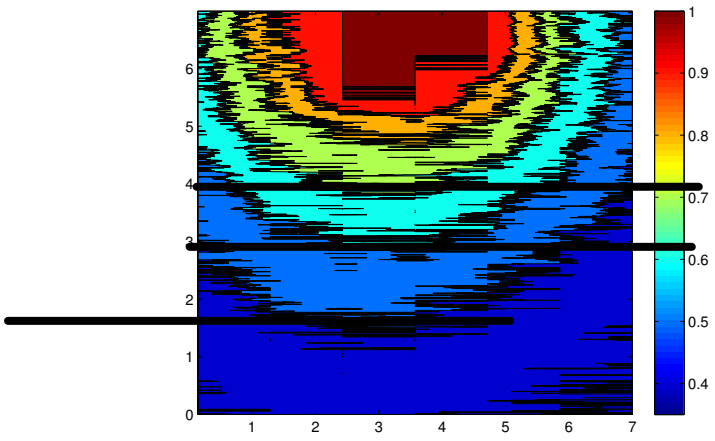

Figure 3: Robot's perception of the sound.

plained by the difference in noise between the two sources. The data from the sound sensor shows a higher level of noise while the light sensor shows a more gradual increase. This difference will become important when explaining how the inputs are computed. The light sensor also possess a single value memory updated every 120 timesteps and containing a single reading. We will refer to it as *imprint* later on. This imprint serves as an ambient light measure used by the controller to discriminate the light coming from the source.

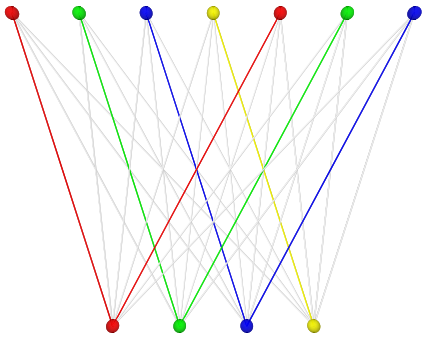

Figure 4: Network composed of 7 input neurons(top) and 4 output neurons(bottom). The lines represent the synapses. The colors used represent the ideal connections between nodes. For instance, both red inputs should be ideally connected to the red output.

The controller described below has been first designed on a real robot and then transposed to simulation. Because the noise in a real environment differs from a simulated one, this approach guarantees that the behavior in simulation remains the same as in reality.

Inputs Pre-Processing The inputs of the FNN are pre-processed in order to obtain binary inputs. The pre-processing is necessary for the Hebbian learning to be stable and is different for each type of sensor. Only one input can be activated for each sensor at every timestep. Practically this means that among the 7 inputs, 1 input from the first 4 and 1 input from the last 3 can be activated simultaneously.

The following equations describe the activation rule for the first 4 inputs relating to the sound perception:

$$I_0 = 1 \quad \text{if} \quad S(t) > S(t-30) + 0.03 \tag{1}$$
$$I_1 = 1 \quad \text{if} \quad S(t) < S(t-30) + 0.03 \tag{2}$$
$$I_2 = 1 \quad \text{if} \quad S(t) \geq S(t-30) \tag{3}$$
$$I_3 = 1 \quad \text{if} \quad S(t) \leq S(t-30) \tag{4}$$

where I are the inputs and $S(t)$ refers to the sound sensor reading at time t. Those equations are evaluated in order and the evaluation stops when one input is set to 1.

The inputs relating to the light sensor follow a set of rules also. Before deciding which input to activate, the imprint must be subtracted from every reading used. The following equations describe the activation rule the last 3 inputs relating to the light perception:

$$I_6 = 1 \quad \text{if} \quad L(t) < 0.01 \tag{5}$$
$$I_5 = 1 \quad \text{if} \quad \begin{cases} L(t) < L(t-10) - 0.01 \\ \text{the robot goes backward} \end{cases} \tag{6}$$
$$I_6 = 1 \quad \text{if} \quad \begin{cases} L(t) < L(t-10) - 0.01 \\ \text{the robot goes forward} \end{cases} \tag{7}$$
$$I_5 = 1 \quad \text{if} \quad \text{if the robot goes backward} \tag{8}$$
$$I_4 = 1 \quad \text{if} \quad \text{if the robot goes forward} \tag{9}$$

where I are the inputs and $L(t)$ refers to the light sensor reading at time t. As for the sound, those equations are evaluated in order and the evaluation stops when one input is set to 1.

The equations for the sound and the light sensors differ because of the properties of the sensors. For the sound, equations 1 and 2 target cases when the current level of sound differs from the past by more than a threshold. Equations 3 and 4 are used when the current level is below this threshold. For the light, equation 5 is applied when the light has no significant differences from the ambient light. Equations 6 and 7 deals with the case where the current level of light is lower than 10 readings ago. Finally equations 8 and 9 are used when there is no important change in the current light level. For the light, the direction of movement of the robot had to be taken into account as the sensor is directional.

The parameters used in the above equations were determined experimentally for our setup.

Outputs The outputs are squashed to a range of $[0; 1]$ using the sigmoid function and are used to activate specific behaviors. Each output is attached to one behavior through a winner-takes-all strategy, i.e. the output with the highest activation activates its corresponding behavior. The four behaviors are:

1. Output 0 maintains the current behavior.
2. Output 1 inverts the current behavior. If the robot is going forward, it will go backward at the next timestep and vice-versa.
3. Output 2 modifies the current behavior to create a left turn while maintaining the same direction.
4. Output 3 modifies the current behavior to create a right turn while inverting the current direction.

Learning As mentioned previously, the weights of the FNN are tuned through learning using Oja's Hebbian rule(Oja (1982)). The rational behind this choice is to allow the possibility for different couplings between the different sensory modalities. Despite the unsupervised nature of Hebbian learning, we cannot expect that the FNN will learn the task without supervision. For that purpose, we implemented a crude controller that will indicate which output should be activated based on the activated inputs. Once the FNN's outputs have been read and a behavior activated, those are replaced by the outputs advised by the teacher. It is possible that 2 outputs be activated at the same time as the light inputs does not necessarily indicate the same behavior as the sound inputs. In that case, two outputs are activated with a strength of 0.5. If both set of inputs concur on one behavior, its assigned output will receive an activation of 1. This will express the certainty of the decision of the teacher and influence the learning. Once the outputs have been modified, the Hebbian rule is applied and the weights are modified. The taught connections between inputs and outputs are as follows:

- Output 0 should be connected to input 0 and input 4
- Output 1 should be connected to input 1 and input 5
- Output 2 should be connected to input 2 and input 6
- Output 3 should be connected to input 3

Those ideal connections are also shown in figure 4, where inputs and outputs sharing similar colors should be ideally connected by the synapse of the same color.

Spiking Neural Network The SNN is based on Izhikevich neurons(Izhikevich (2003, 2004). Contrary to feedforward neurons who possess a membrane potential that is continuous and transferred directly to other neurons, a spiking neuron transfers information in the form of spike. A spike is an electrical impulse sent by a pre-synaptic neuron to all its post-synaptic neurons. The synaptic plasticity follows the model of STDP proposed by Song et al. (2000). This model differs from Oja's Hebbian rule used for the FNN as it incorporates a time window during which 2 spiking events will produce a synaptic modification. In other words, the firing of the pre- and post-synaptic neuron does not have to be simultaneous to produce a synaptic modification. The temporal sequence of firing is nevertheless important. If the pre-synaptic neuron fires before the post-synaptic one, the synaptic strength is reinforced. In the opposite case, the synapse is weakened.

The Izhikevich neurons are determined by 4 parameters. The values we chose to use correspond to the regular spiking model($a = 0.02$, $b = 0.2$, $c = -65mV$ and $d = 6$). The parameters for the STDP are $A_+ = 0.02$, $A_- = 0.021$, $\tau_+ = \tau_- = 0.02$. For the SNN to be mathematically stable, the time step of the simulation has been increased to 0.001s. All the timings of the simulation have been adapted to reflect this change and to provide the same setup with the SNN than with the FNN.

Results

Feed Forward Controller

Using the FFN as controller, the robot manages to reach the target from all positions within the grid. The average success rate from every starting point in the grid, over 1000 repetitions, is shown in figure 5, where the target is located on the middle top position in the arena. The 3 graphs show the success for 3 different conditions. In the top left graph, the light sensor is the only one activated which gives us an idea of its usefulness to reach the target. It is interesting to notice that our robot appears to be better at picking the light coming from its right, as the performance is higher on the left side of the arena. The top right graph shows the success when only the sound sensor is activated. We can see that the performance is not symmetrical but is much more regular over the whole arena than for the light sensor. Nevertheless, the overall performance is lower. Finally, the bottom graph shows the performance when both sensors are activated. The pattern is similar to when only the sound sensor is activated, but we can notice a faster decrease in performance when far away from the target. We can also notice a slight tendency to have a higher performance when the robot is starting from the left side of the arena, which is consistent with what we see when only the light sensors is activated. As such, it is clear that both sensors influence the performance of the robot, but it is also clear that it is not always a positive reinforcement. It seems that both sensors interfere with each other at the level of the controller.

To explore how the sensors are combined by the controller, we performed an experiment where light processing and sound processing can possess different timescales, i.e. each one can be activated independently and at different tim-

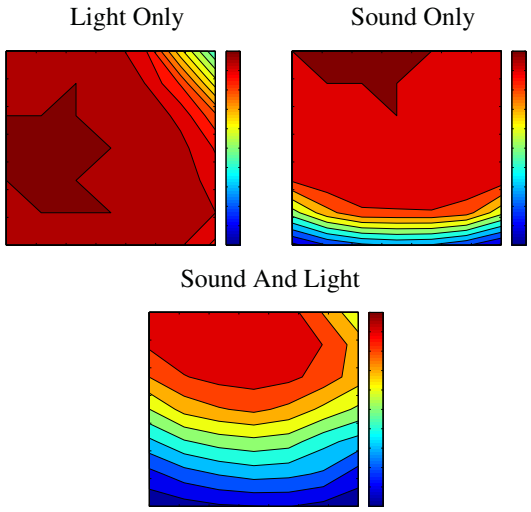

Figure 5: Rate of success for every starting cell within the arena, under 3 conditions: light sensor activated, sound sensor activated, both sensors activated. The goal is located on top central point.

ings. Also, we explored how sensory noise can influence the performance of the robot and if the plastic controller could compensate one sensor by the other. For all experiments, the level of noise describes the max value drawn from a uniform distribution that can be added to a reading of a sensor. The results are shown in figure 6. The left row shows the performance when noise is added to the sound sensor, while the right shows the same with the light sensor. Each row shows a different level of noise for their respective sensor. From top to bottom, the level of noise is increased by increment of 0.2. The X and Y axis provide the speed of update of the sensors. 1 means that it is updated every time step, while 20 means every 20 time steps. The X-axis provides the update speed of the light sensor, while the Y-axis the speed for the sound sensor. The Z-axis is the success rate.

When noise is applied, we notice that the performance is the same for any update speed, except when both modules function at the same speed. In that particular situation, the performance drops slightly, creating a ridge, which could be the consequence of an interference between the two sensory modules. When progressively adding noise, a drop in performance on one side of the ridge appears, but the other side is relatively not affected. The drop on one side only is the consequence of the difference in update speed of the 2 modules. When the noisy module is updated faster than the non-noisy one, this module drives the behavior of the robot more frequently. This is due to the fact that the commands of the noisy module will overwrite the decisions of the other one, as it is activated more frequently. As such, the robot makes more errors as it relies mainly on less reliable information. When the faster module is the non-noisy one, it can compensate for the errors of the other module, reducing the impact of the noise on the system. We can nevertheless notice that in the area surrounding the ridge, the performance is higher even if the noisy module is updated slightly faster than the other one, which means that both modules are nonetheless interacting and improving the performance of the robot under noise.

Our last test is about the importance of each sensory module in the decision process. As the leading behavior is based on a winner-takes-all strategy at the level of the outputs, it is possible to compute which module takes the decision by observing which input produced the activation in the winning output. Figure 7 shows the rate of dominance of the light module, of the sound module, and the rate of cooperation between the 2 modules for different levels of noise applied on the sound sensor. The graphs show for any cell in the grid the percentage that the light or sound module decides the behavior for that particular cell. The cooperation rate is the percentage of time where the two modules agree.

We can see that the light module is the most dominant over all cells. The sound module has a maximum rate of decision of maybe 20%, while the light module can go up to 80%. This can be expected as the light is the most reliable sensor of the two. The most interesting aspect is how the rates change with the addition of noise on the sound sensor. From 0 to 0.5, we can see that the light module is now dominant over the all arena. This means that the sound being unreliable, it is barely used to drive the behavior. There is also almost no cooperation between the two modules. The sound module becomes relevant only when the robot approaches the sound source, probably because the light sensor at this position becomes saturated and a noisy sound sensor provides more information.

Additionally, we also plotted the average success rate for each condition with a varying level of noise, which can be seen on figure 8. As can be expected, when the light module is only active, the performance remains the same as it is not applied noise to. One visible aspect is that the performance with the sound module slightly increases initially, to decrease after a pick in terms of noise. This is due to the controller having been initially implemented in a real robot where the sound sensors delivered more noise readings. Consequently, our simulated controller performs better under a weak level of noise, which allows it to escape local minima within the sound landscape. Also, we can see that the performance of the sound only condition is initially higher than for the sound & light condition. But this changes above a noise level of approximately 0.2. This implies that the controller can support higher noise levels when both modules are activated.

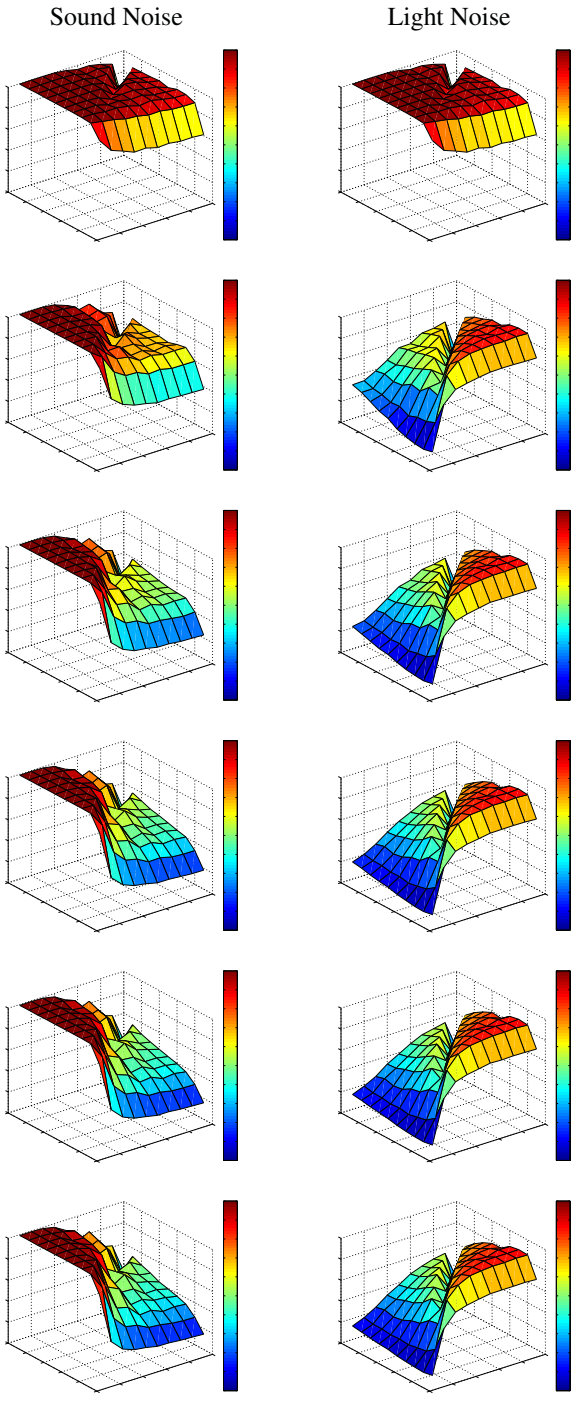

Figure 6: Variation of the performance of the robot under different timescales and different level of noise for the sensory modules. From top to bottom, the level of uniform noise is varied from 0 to 1 by increment of 0.2. The left column has noise on the sound sensor only, while the right one has noise on the light sensor only.

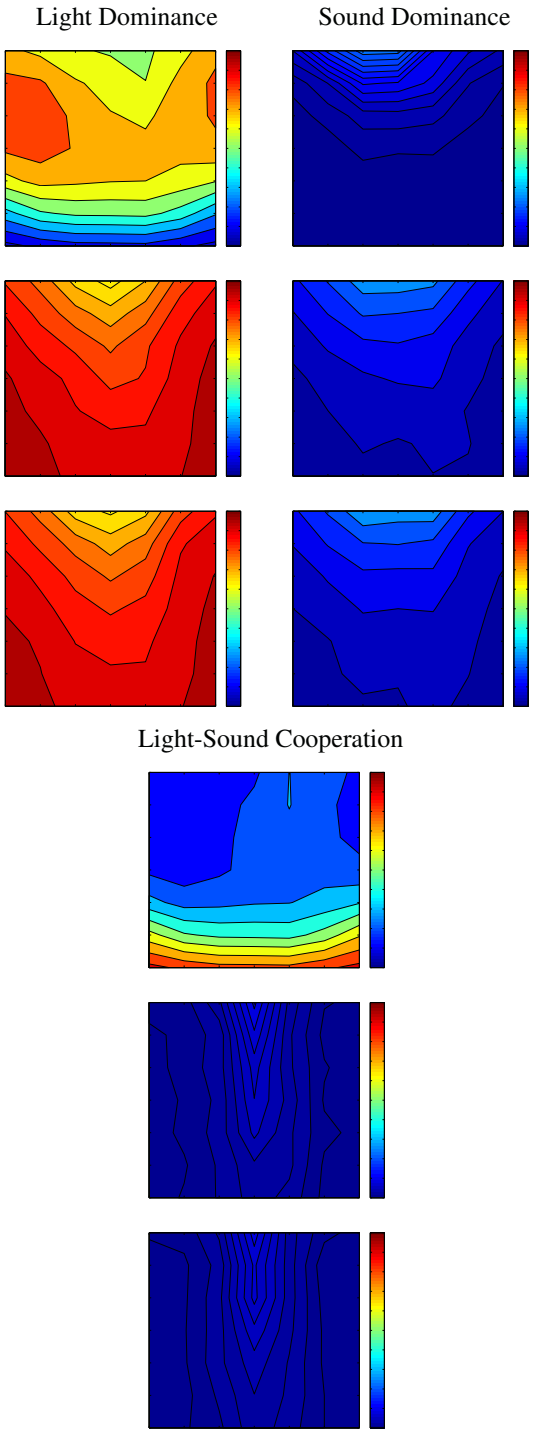

Figure 7: Rate of dominance and of cooperation between the light and sound modules for 3 levels of noise applied on the sound sensor: 0, 0.5 and 1.0.

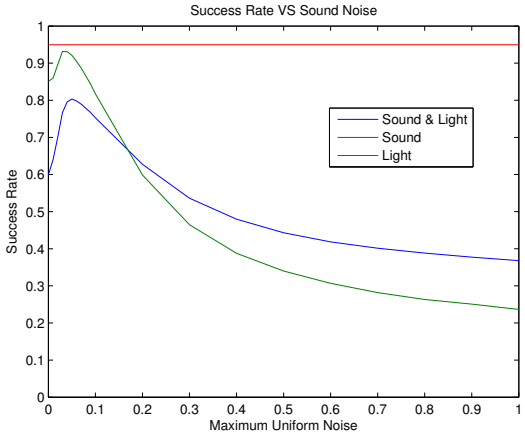

Figure 8: Comparison between the performance of each condition(light only, sound only and both) with different levels of noise on the sound module.

Spiking Controller

The results with the FFN were interesting so we expected a lot from the SNN. Unfortunately, despite many attempts, we were unable to reproduce the results obtained earlier. The SNN was not capable of extracting the right connectivity from the correlations present in the environment. Fortunately, the reason is quite clear and can shed some lights on the differences between using a FFN and a SNN, and can help orient the design of robotic experiments using a spiking controller.

The reason for the lack of replication of the task can be found in the window of time during which synapses are modified under STDP. This window is roughly 40ms where a synaptic strength can be either reduced or increased. In our case, the problem comes from alternating behaviors. Following what we described previously, input 0 would ideally be connected with output 0, and input 1 to output 1. As the inputs are never activated simultaneously, and the learning chooses a winner-take-all strategy on the outputs, the FFN obtains the right connection as the computation of the change in synaptic strength has no memory of the past activation of the neurons. In the case of STDP, such a memory exists. If two behaviors alternate frequently, and are separated by less than 40 ms, they will interfere with each other and create synaptic connections that should not have been learned. For instance, if input 0 spikes, then we would want output 0 to spike also later on. If this is the case, their connecting synapse is strengthened. If soon after, input 1 spikes, followed by output 1. Then, not only their connecting synapse will be strengthened, but also the synapse connecting input 0 and output 1. Over time, the activation of input 0 will create an activation in output 0 and output 1. This is equivalent to learning the wrong correlations from the environment.

At this point we need to mention that a SNN can of course resolve this task, but it would need a different architecture as the one we offer here. It would need one that can cope with the interferences. Our search for different parameters for the SNN, or by changing some details of the task, did not lead to a successful learning. Our next step should be to research what kind of topology can cope with the interferences. This might imply introducing some kind of modularity in the system.

Conclusion

At the beginning of the article, we wanted to explore two questions. The first was if a neural network equipped with Hebbian learning could naturally learn and exploit the correlations present in an environment where a target is indicated by two sensory modalities. The second was if there was a difference between a feed forward neural network and a spiking neural network, both equipped with their version of Hebbian learning, in terms of performance on the task. We can start answering those two questions.

Concerning the first question, it is difficult to say that Hebbian learning was sufficient to learn the correlations in the environment. Figure 5 and 8 show that the controller displays a different behavior when the light and sound modalities are provided, compared to the situation when only one of them is used. This tells us that the network is exploiting both sensory modalities, and that the combination is not a simple linear sum of both modules. Furthermore, the dominance analyses shown in figure 7 confirm that based on the amount of noise on one modality, the less noisy one becomes more important in deciding which behavior to activate. This supports the idea that Hebbian learning, combined with a teacher, could naturally exploit the most reliable information present in the environment. The teacher is important because he guarantees that the most reliable connections between inputs and outputs are more frequently reinforced compared to the more noisy ones. Following that idea, it becomes also plausible that Hebbian learning could modify the connectivity of the network to promote one sensory information if it provides the most reliable information within a specific area in the robot's environment. Without the teacher's feedback, it seems difficult for the Hebbian learning to naturally exploit this information. Indeed, deactivating it does not lead to a successful behavior. But through the teacher's interaction, the system managed to rely on the most reliable sensor to complete the task.

The answer to the second question is that there are clearly some differences between Hebbian learning in FFN and SSN. The interferences between the different behaviors prevent the network to learn the relevant correlations from the environment. This clearly demonstrates that there are important differences between the classical Hebbian learning found in FFN and the STDP found in SNN. The time window used by STDP to compute the synaptic modifications

are responsible for the failure to learn our particular task. As this mechanism has been modeled directly from real synapses going through strengthening or depression, ignoring this difference might lead to undesirable effects. For instance, the conclusions drawn from a robotic experiment using FFN and Hebbian learning might not be related to what could be happening in a real neural network because time has not been taken into consideration. As one goal of artificial life is to understand life as it is by recreating it, it might be wise to work with closer models of the brain in order to draw conclusions relevant to what we are interested in.

References

Abbott, L. F. and Nelson, S. B. (2000). Synaptic plasticity: taming the beast. *Nature Neuroscience*, 3:1178 – 1183.

Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11:127–138.

Hebb, D. O. (1949). *The Organization of Behavior*. Wiley, New York.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572.

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070.

Kelso, S. R., Ganong, A. H., and Brown, T. H. (1986). Hebbian synapses in hippocampus. *Proceedings of the National Academy of Science USA*, 83(14):5326–5330.

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, volume 1, pages 59–65.

Oja, E. (1982). Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273.

Song, S., Miller, K. D., and Abbott, L. F. (2000). Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, 3:919–926.