Istanbul Technical University, Robot Intelligence Course

Realization of Visual Representation Task on a Humanoid Robot

Emeç Erçelik

May 31, 2016

1 Introduction

It is thought that human brain uses a distributed form of abstraction to model the external environment, to extract information and to effect its state by making actions. This abstraction may be provided by the visual system that filters the information coming to the retina. This visual stimuli, which may be an object or a scene, are represented by an internal representation. The internal representation is also thought to be generated in a self-organizing way and with the help of reward [1].
In this project, the representation of an object is generated by spiking neural networks (SNN) in a self-organizing way. In addition, the generated representations are used to actuate the humanoid robot, Darwin-OP [2], to accomplish a task. So, the perception of the robot on an object is built in a biologically-plausible way with a computational model. The presented model in this report can be used to test behavioral tasks or models in real environment. For example, the representations of objects may be the input information for decision making models [3] instead of feeding the model with discrete values.

2 Task and Environment

In the considered task, the robot is expected to put a perceived object into a box which has the same color with the object. However, since Darwin-OP does not have grasping ability, the task is simplified. The robot is expected to perceive an object and move to the similar color card with the perceived object.

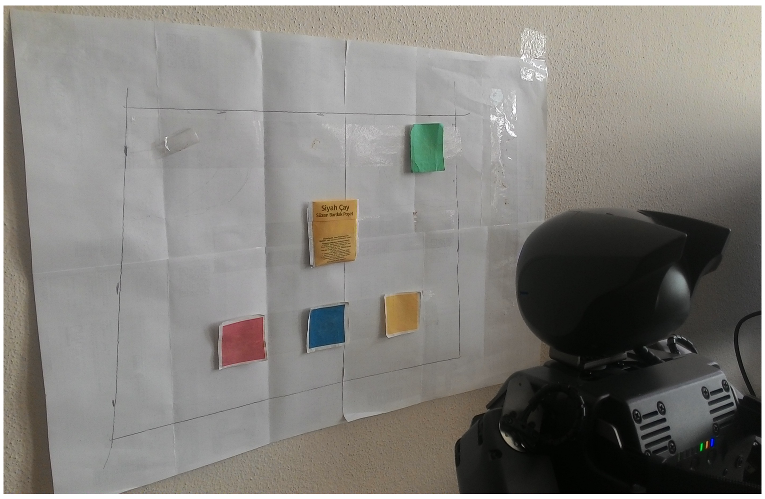

Figure 2.1: The environment that the task is realized.

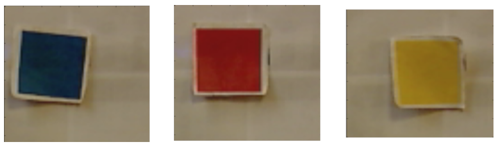

Figure 2.2: Color cards that Darwin-OP moves towards.

The task is realized in the environment that is seen in Figure 2.1. In the environment, the object is presented to the robot in the middle of the white board. The three colors seen at the bottom of the board are the color cards (Figure 2.2) that Darwin-OP moves towards according to their similarity with the object. There are three different objects to test the model as seen in Figure 2.3. Before the task is realized on Darwin-OP, the model is tested with the unnatural figures that are seen in Figure 2.4.

3 Process

To realize task two different codes run together. The code to move Darwin-OP is written in C++ and the code to simulate the computational model of representation is written in

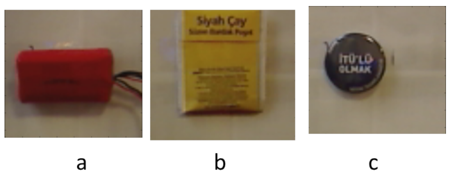

Figure 2.3: The objects that Darwin-OP perceives.

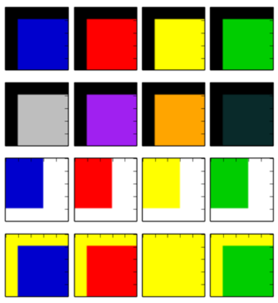

Figure 2.4: The unnatural figures that are generated to test the model before using the model on Darwin-OP.

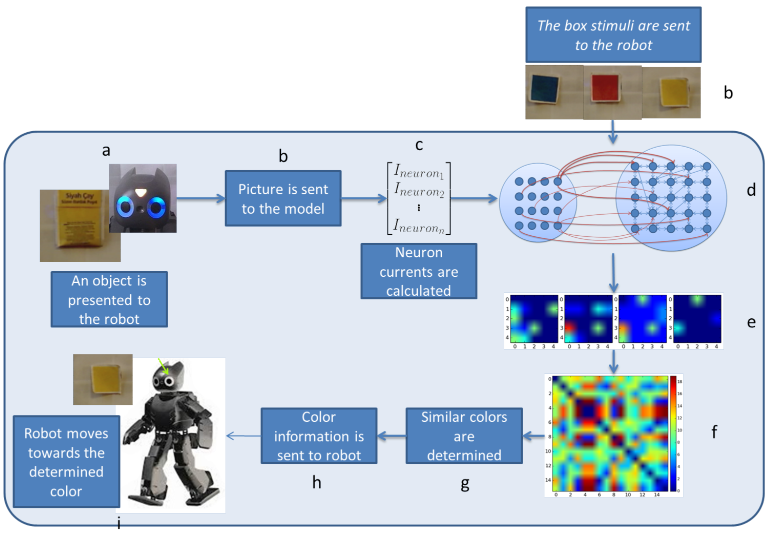
The box stimuli are sent to the robot
b
a
b
c
Picture is sent to the model
I_{neuron_1}
I_{neuron_2}
I_{neuron_n}
Neuron currents are calculated
d
An object is presented to the robot
e
f
Robot moves towards the determined color
Color information is sent to robot
Similar colors are determined
h
g
i

Figure 3.1: The process to realize the task.

Python. The NEST library ([8]) is used to simulate our computational model. The process can be seen in Figure 3.1. At the beginning the object and the color cards are presented to the robot (Figure 3.1-a). The robot takes the picture of environment and sends the picture to the Python code (Figure 3.1-b). Then the Python part splits the picture into images of object and color cards. The images are turned into the stimuli by calculating neuron currents according to their pixel values (Figure 3.1-c). Afterwards, the computational model is simulated by using the stimuli (Figure 3.1-d) and the representations of the presented images are generated according to the activity of neurons in the model (Figure 3.1-e). The representations are compared to each other to determine which of the color card is similar to the object (Figure 3.1-f,g). The information of similar color card is sent to the robot (the C++ coded part) and the robot moves towards the color (Figure 3.1-h,i).

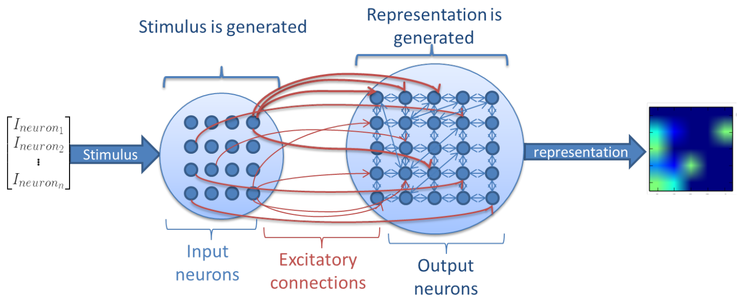

Figure 4.1: The computational model that generate representations.

4 Computational Model

The computational model consists of two different kind of spiking neuron groups as seen in Figure 4.1. The group of input neurons is at the left-hand side and the group of the output neurons is at the right-hand side. The neurons are modeled by using Izhikevich Regular Spiking neuron model [4]. Each neuron has input current to be activated and input neurons are fed by the currents that are calculated using the presented object to the robot. So, the stimulus of each neuron is one of the currents that are seen at the left-most side of the Figure 4.1. The activity of output neurons generates a representation as the one at the right-most side of the Figure 4.1.

Each of the input neurons is connected to the each of the output neurons with excitatory connections. The weights of these connections are randomly assigned at the beginning. The output neurons are connected in all-to-all manner among the group. At the beginning all connections are excitatory and the weights are selected randomly. The excitatory connections are the connections that the presynaptic neuron increases (excites) the activity of the postsynaptic neuron through this connection. Inhibitory connections are the connections that the presynaptic neuron reduces (inhibites) the activity of the postsynaptic neuron through the inhibitory connection. The connections in-between output neurons are updated by using a learning rule that is similar to Hebbian learning rule.

4.1 Learning Rule

The learning rule is inspired from the Hebbian Learning Rule ([5]) and Spike Timing Dependent Plasticity (STDP) rule ([6] & [7]). In this learning rule, the update process considers the timing of spikes. When two neurons spike in the same window (50 ms window) then the connection weight between two neurons are increased by 0.01. If there are neurons that do not have a spiking activity in the same window, then the connection weight between the spiking neurons to the passive neurons are decreased by 0.04. This decrease turns the excitatory connections into inhibitory connections. Thus, the activated neurons by a certain image inhibite the other neurons and represent the image.

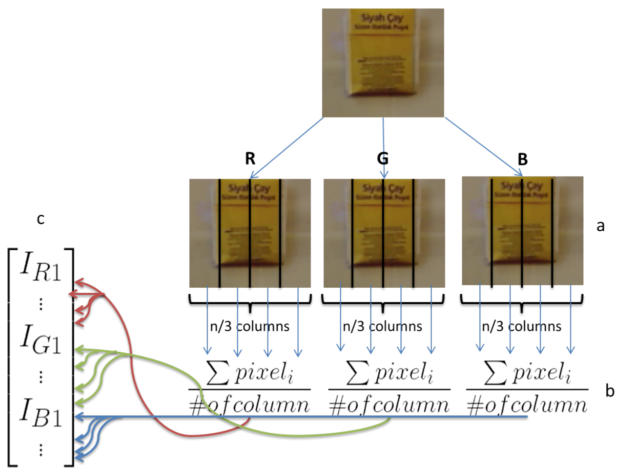
R
G
B
a
c
b
n/3 columns
n/3 columns
n/3 columns
$\frac{\sum pixel_i}{\#ofcolumn}$
$\frac{\sum pixel_i}{\#ofcolumn}$
$\frac{\sum pixel_i}{\#ofcolumn}$
I_{R1}
I_{G1}
I_{B1}

Figure 4.2: Calculation of the stimulus to stimulate the input neurons.

4.2 Method to generate stimulus

An image that will be represented by the computational model is coded by currents to feed the input neurons. To calculate the currents, the image is seperated to three images considering its RGB color values (Figure 4.2-a). If we consider n input currents for n input neurons, each of R, G, B images is seperated into $n/3$ columns and the pixel values are summed to determine the input current comes from that part of the image (Figure 4.2-b). The summed values are scaled by the number of columns and pixels. These values are ordered beginning from the R image to B image and constitute a vector of input currents as seen in Figure 4.2-c.

4.3 Simulation of Neurons

The computational model is simulated by using NEST library [8]. Each input is presented to the model for $500ms$. Neurons are stimulated by the calculated currents of images. The currents for the images are presented one by one and in order. When the input currents are presented the activity of input neurons can be seen from Figure 4.3. The images are the ones in Figure 2.4. The columns show the activity for each image. The magenta dots indicate the spike activity of the neuron. The y-axis shows the index of neuron and x-axis shows the time. For example, the neurons 12, 13, 14, 15 are highly active for the image 1.
The output neurons are activated by input neurons. The activity of output neurons are shown in Figure 4.4. The numbers at the bottom indicate the index of the presented image. This figure shows the activation of the output neurons when they are stimulated by the input neurons.
Representations are generated by using the activity of output neurons. The number of spikes in each column are summed and normalized with the maximum spike count in one column. Then the calculated value is colored to indicate the activation density. The representations of the images in Figure 2.4 can be seen in Figure 4.5.
To realize task, it is needed to determine the similarity of object to the color cards. That's

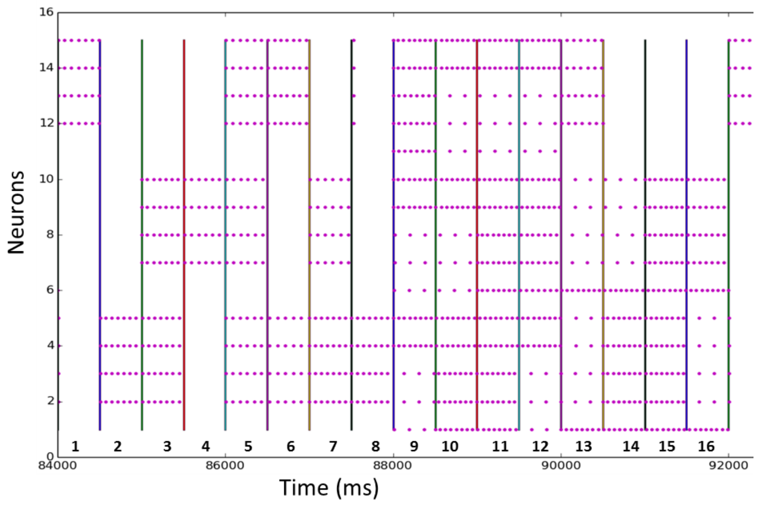

Figure 4.3: Activity of the input neurons when the images in Figure 2.4 are presented in order.

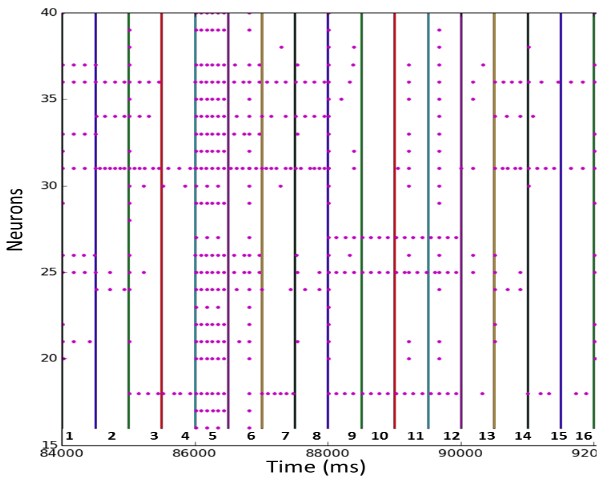

Figure 4.4: Activity of the output neurons when the images in Figure 2.4 are presented in order.

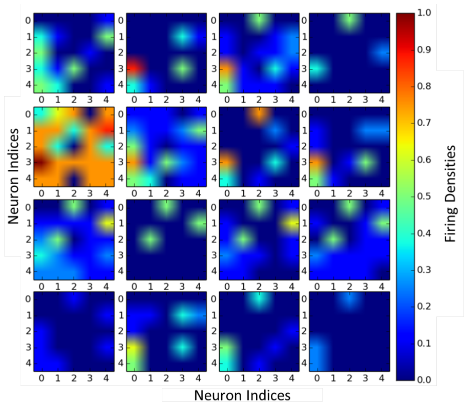

Figure 4.5: The representations of the images in Figure 2.4 (in order).

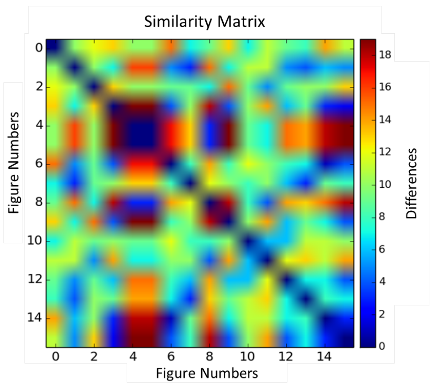

Figure 4.6: The colored similarity matrix of the representations in Figure 4.5.

why a similarity method is proposed. To determine the similarities, the activity of each output neuron for each image is calculated. The activity is normalized for of the number of spikes. Then the calculated activations are rounded to 1 or 0 according to a threshold. The calculated neuron activities of two images, for which the similarity will be measured, are subtracted. And the elements of the vector resulted from the subtraction are summed. The value indicates how similar two figures are to each other. When the value is low, this means that the two figure is similar. The similarity measurement matrix of the representations in Figure 4.5 can be seen in Figure 4.6.

5 Results

The images and the object are presented to the model and the representations are obtained (Figure 5.1). As seen in Figure 5.1, the object is a yellow tea bag (1) and the color cards (2, 3, 4). The representations of each image are shown under the images. According to these representations the object is different from the red card by 5 similarity units, is different from

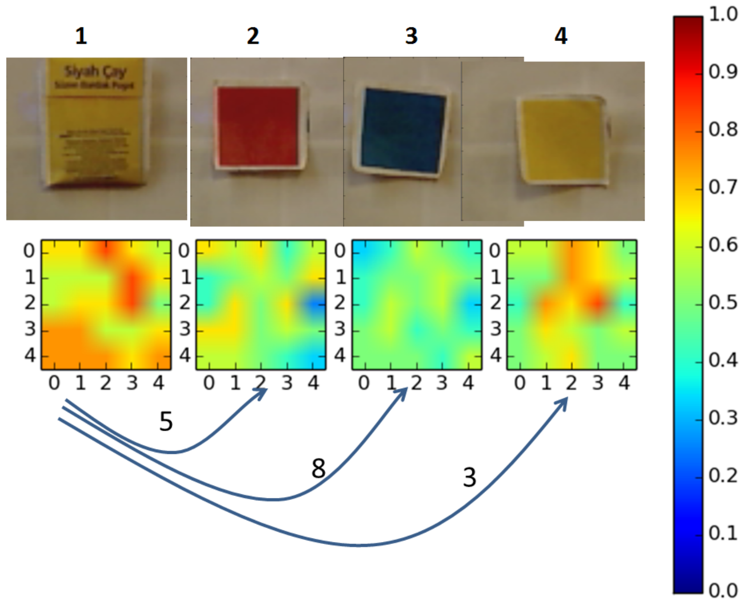

Figure 5.1: The presented images and their corresponding representations. The calculated differences of object to color cords are shown at the bottom.

the blue card by 8 units and is different from the yellow car by 3 units. So, the yellow tea bag is represented more similar to the yellow card and the robot move towards the yellow card. To reach these representations, the connection types among the output neurons are changed. At the beginning, all of the connections are in excitatory type. At the end of simulation, very few of them are excitatory and most of them are inhibitory (Figure 5.2).

6 Conclusion

In this project, a simple perception model is created for a humanoid robot, Darwin-OP, to provide its perception in real environment. This is achieved by using SNN and the activity of SNN creates the representations. This setup can be a beginning to be used in a psychological experiments to achieve behavioral results. The model can be improved by considering a reward signal during the learning process. In addition, robot has to be able to adjust itself to provide proper light. Another improvement may be to study on robot's self position adjustment ability to take better pictures of the environment. Also, a gripper can be added to the robot to realize more realistic tasks.

References

[1] Brohan, K., Gurney, K., & Dudek, P. (2010). Using reinforcement learning to guide the development of self-organised feature maps for visual orienting. In Artificial Neural

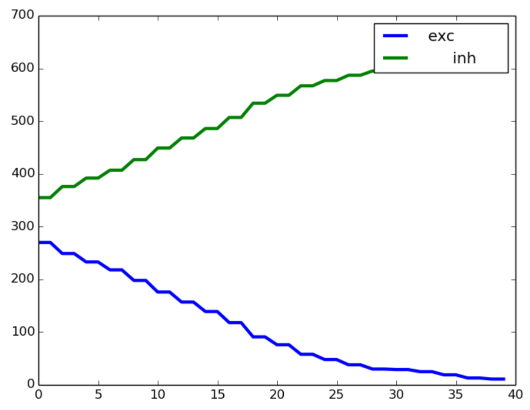

Figure 5.2: Change of connection types through iterations. Blue: Number of excitatory connections, Green: Number of inhibitory connections

Networks ICANN 2010 (pp. 180-189). Springer Berlin Heidelberg.

[2] Ha, I., Tamura, Y., Asama, H., Han, J., & Hong, D. W. (2011, September). Development of open humanoid platform DARwIn-OP. In SICE Annual Conference (SICE), 2011 Proceedings of (pp. 2178-2181). IEEE.

[3] Ercelik, E., & Sengor, N. S. (2015, July). A neurocomputational model implemented on humanoid robot for learning action selection. In Neural Networks (IJCNN), 2015 International Joint Conference on (pp. 1-6). IEEE.

[4] Izhikevich, E. M. (2003). Simple model of spiking neurons.ăIEEE Transactions on neural networks,14(6), 1569-1572.

[5] Hebb, D.O. (1949). The Organization of Behavior. New York: Wiley & Sons.

[6] Markram, H. and Sakmann, B. (1995). Action potentials propagating back into dendrites triggers changes in efficacy. Soc. Neurosci. Abs. 21

[7] Bi, G. Q. and Poo, M. M. (1998). Synaptic modifications in cultured Hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. J Neurosci, 18:10464-72

[8] Gewaltig M-O & Diesmann M (2007) NEST (Neural Simulation Tool) Scholarpedia 2(4):1430.