Task 1: clustering and classification

In this task the robot Bioloid learns to cluster and classify objects according to their shapes. Self organizing map (SOM) is used for clustering and after annotation of some of its neurons, the same map is also used for classification.
The robot is connected to Matlab running on a computer. All the computations are made in Matlab so it is easily adjustable by editing the prepared script. Furthermore there are three variables at the beginning of the script – number of training objects, number of testing objects and position variability. These variables are designed for the user to try different settings and examine relationships between variability, number of the training examples and an error rate of the classification.
The script is divided into cells*, that should be executed in sequence, otherwise it may not work properly. The following text is a description of the script and a guide to the task.

1. Connect the robot to a computer, turn the robot on and find out which COM port number is assigned to the connection. Fill the number to the variable *COM_port_number* at the beginning of the script and execute the first cell.
2. Set the number of training pictures, number of testing pictures, position variability and execute the second cell. Note that the robot can fall while it moves to its default position.
3. Execute the third cell to generate training pictures and labels.
4. Move the robot approximately 15cm in front of a screen and point camera toward to the screen. Execute the fourth cell and two figures appear. One figure is a calibrating picture that helps to move the robot to the right position and the second figure is an image from the camera. Move the robot and the calibrating picture to face each other [Fig. 1]. Press Enter in Matlab command window to update image from the camera. Once the robot sees the calibrating picture correctly [Fig. 2] write „ok“ in the Matlab command window to start the picture capturing process.
5. Wait until the step four ends and then execute the fifth cell. SOM Toolbox uses its internal function to determine the required parameters and creates and learns new SOM. Then the map is labeled – For each captured picture a winning neuron is found and the label of the picture is assigned to the neuron. After learning and labeling, a figure with U-matrix and neurons labels appears. U-matrix is a graphical visualisation of distances between neurons.
6. Some of the neurons don’t have to be labaled because there are no pictures making them a winning neurons. Execute the sixth cell to start k-means algorithm that adds labels to the unlabeled neurons. This cell is optional, neurons may stay unlabeled.
7. Execute the seventh cell to generate testing pictures and labels.
8. Execute the eighth cell to start testing the map. Pictures are presented to the robot and the robot uses the SOM to determine a shape in the picture and makes a pose according to the recognized shape.
9. To terminate the connection with the robot execute the nineth cell. Otherwise go back to cell seven or two to try different parameters or go to next task.

*cell is a block of statements between %% (double percent) and is executed by Ctrl+Enter

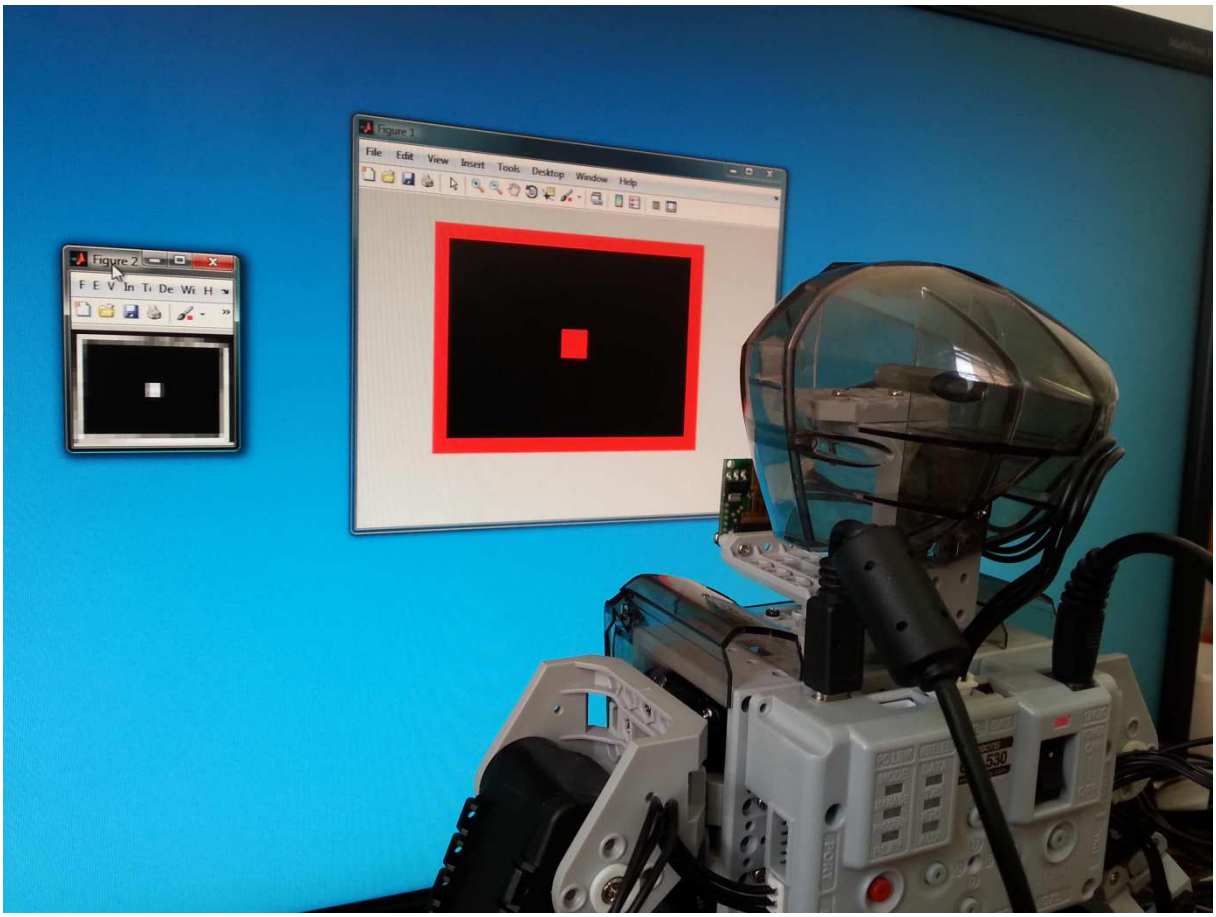

Fig. 1 – Robot looking at a calibrating figure.

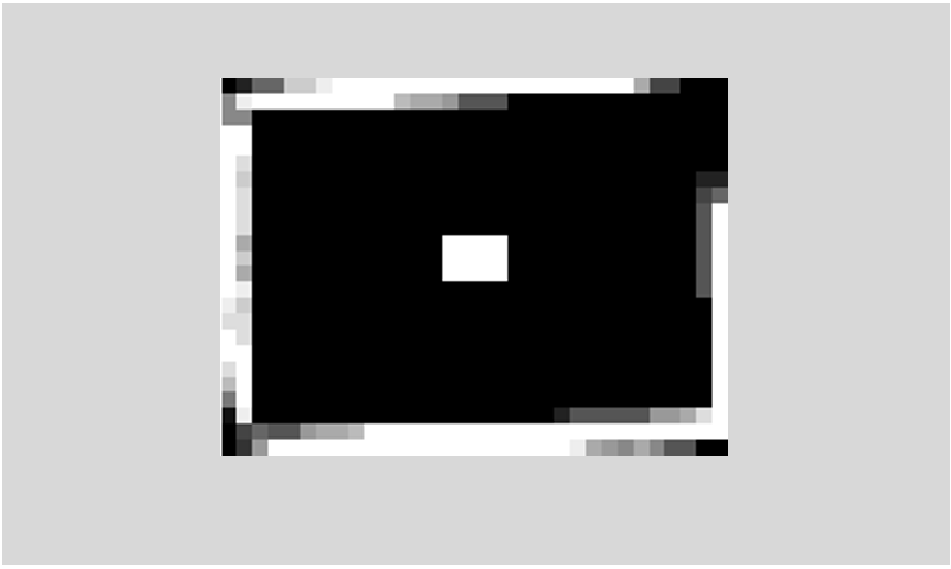

Fig. 2 – A reference image from the robot's camera. Any image similar to this one is ok.