Robotics Behaviors Derived with Genetic Algorithms

Jesus Savage, David Rosenblueth, Stalin Muñoz and Marco Negrete
Bio-Robotics Laboratory, College of Engineering,
Universidad Nacional Autónoma de México (UNAM),
robotssavage@gmail.com

25 de abril de 2025

Behaviors GUI

GUI_behaviors.py is the GUI of a system for generating robots' behaviors using genetic algorithms.

To use follow the next instructions:

1. Unpack robotics.tar.gz in the user's directory.
2. Unpack data.tar.gz in the user's directory.
3. Using Ubuntu-Linux open an Xterminal and go to the directory where the programs are:

   cd ∼/robotics

   Compile with:

   ./behaviors_make

It is possible that some warnings will appear.

Behaviors GUI

4. Go to the directory where the GUI is:

   cd ∼/robotics/gui/

Type the following command to see the system usage:

   python GUI_behaviors.py

```
biorobotica@biorobotica-ThinkPad-P50:gui->
biorobotica@biorobotica-ThinkPad-P50:gui-> python GUI_behaviors.py
********************************************************************************************
Usage: python GUI_behaviors.py num_behavior
Where num_behavior:
0 = Testing Best behaviors environment_type(0 = office, polygons)
            Examples:
                 python GUI_behaviors.py 0 0
                 python GUI_behaviors.py 9 1
1 = Stochastic FSM outputs
3 = FSM
4 = Reactive
5 = Reactive Stochastic
6 = Neural_Networks stochastic_outputs num_recurrent
               Example: python GUI_behaviors.py 6 0 0
               Example: python GUI_behaviors.py 6 1 0
               Example: python GUI_behaviors.py 6 0 4
               Example: python GUI_behaviors.py 6 1 4
7 = Potential fields num_bits_variables num_bits_fractions
               Example: python GUI_behaviors.py 7 16 8
               Example: python GUI_behaviors.py 7 8 7
8 = MDP
9 = Testing Evolved Populations num_behavior environment_type(0 = office, polygons)
            Example for testing FSM behaviors:
                              python GUI_behaviors.py 9 3 0
                              python GUI_behaviors.py 9 3 1
            Example for testing Stochastic FSM behaviors:
                              python GUI_behaviors.py 9 1 0
                              python GUI_behaviors.py 9 1 1
            Example for testing pure Reactive behaviors:
                              python GUI_behaviors.py 9 4 0
                              python GUI_behaviors.py 9 4 1
            Example for testing stochastic Reactive behaviors:
                              python GUI_behaviors.py 9 5 0
                              python GUI_behaviors.py 9 5 1
            Example for testing input/state MDP behaviors:
                              python GUI_behaviors.py 9 8 0
                              python GUI_behaviors.py 9 8 1
            Example for testing input/state Moore behaviors:
                              python GUI_behaviors.py 9 11 0
                              python GUI_behaviors.py 9 11 1
            Example for testing input/state HMM Moore behaviors:
                              python GUI_behaviors.py 9 12 0
                              python GUI_behaviors.py 9 12 1
            Example for testing NN behaviors:
                              python GUI_behaviors.py 9 6 0 0 0
                              python GUI_behaviors.py 9 6 1 0 0
            Example for testing Stochastic NN behaviors:
                              python GUI_behaviors.py 9 6 0 1 0
                              python GUI_behaviors.py 9 6 1 1 0
            Example for testing Recurrent NN behaviors:
                              python GUI_behaviors.py 9 6 0 0 4
                              python GUI_behaviors.py 9 6 1 0 4
            Example for testing Recurrent Stochastic NN behaviors:
                              python GUI_behaviors.py 9 6 0 1 4
                              python GUI_behaviors.py 9 6 1 1 4
10 = Meally FSM Input/State
11 = Moore FSM Input/State
12 = HMM Moore
********************************************************************************************
biorobotica@biorobotica-ThinkPad-P50:gui->
biorobotica@biorobotica-ThinkPad-P50:gui->
```

Behaviors GUI

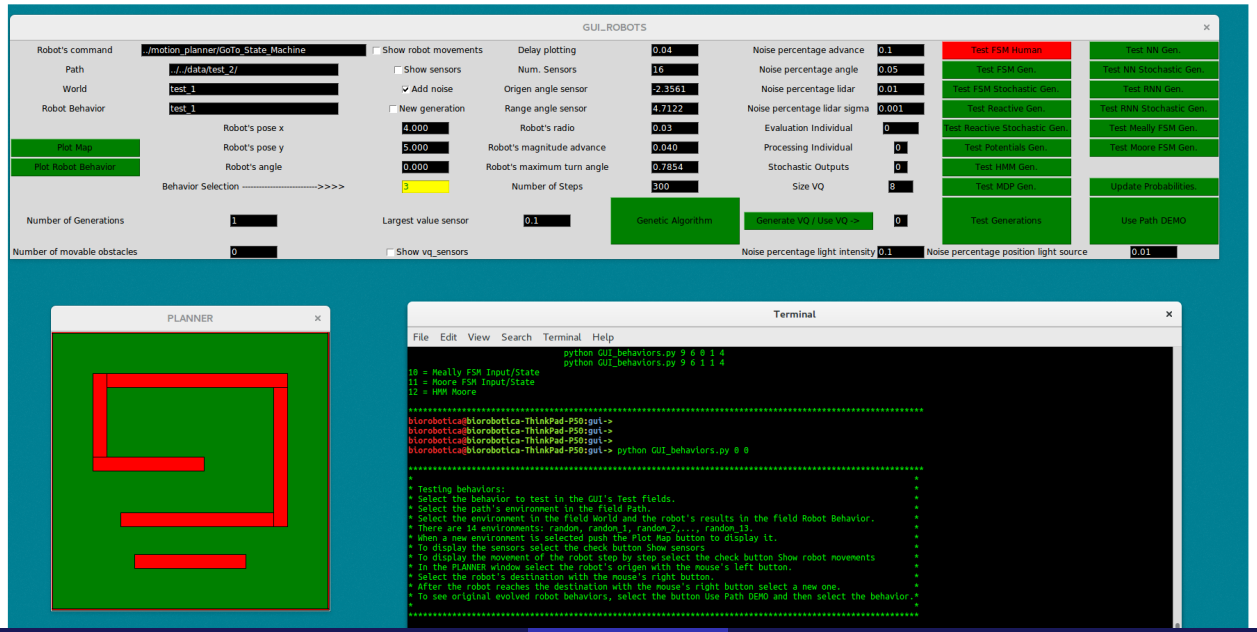
5. With option 0 the evolved behaviors that are already in /home/user/data can be tested, there are two types of environments, office and random polygons.

To test polygons obstacles type the following command:

python GUI_behaviors.py 0 1

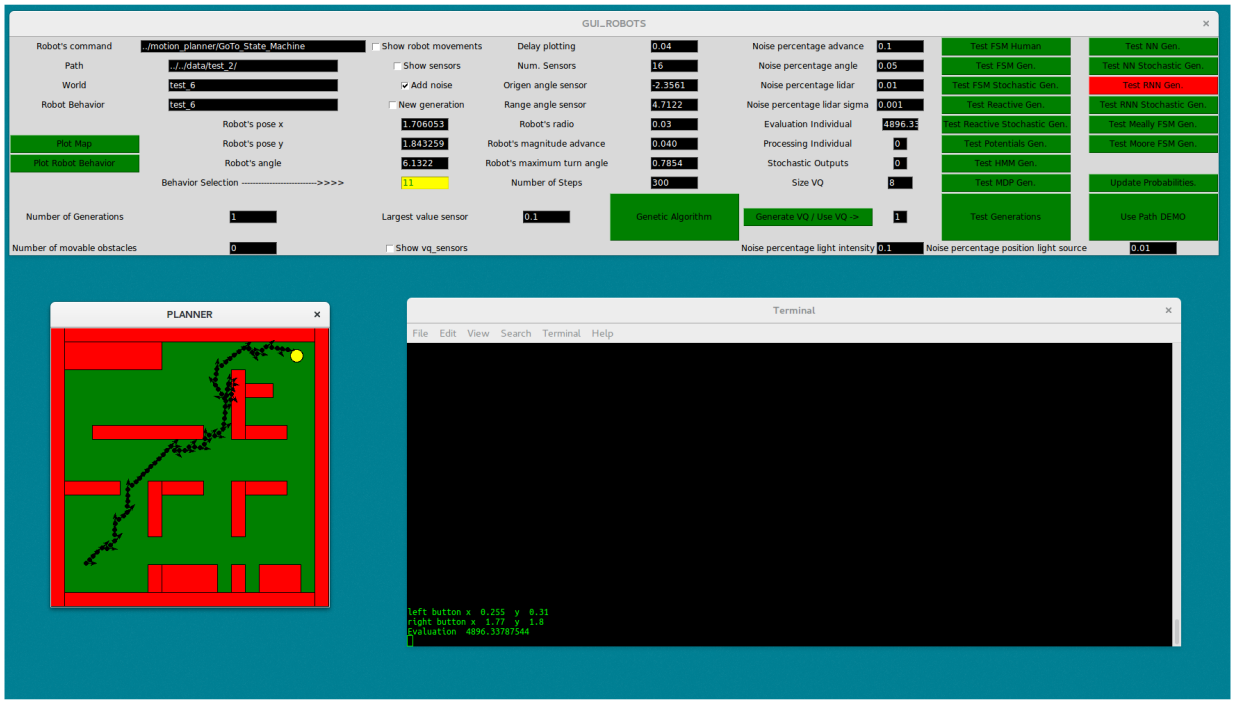
To test office environments type the following command:

python GUI_behaviors.py 0 0



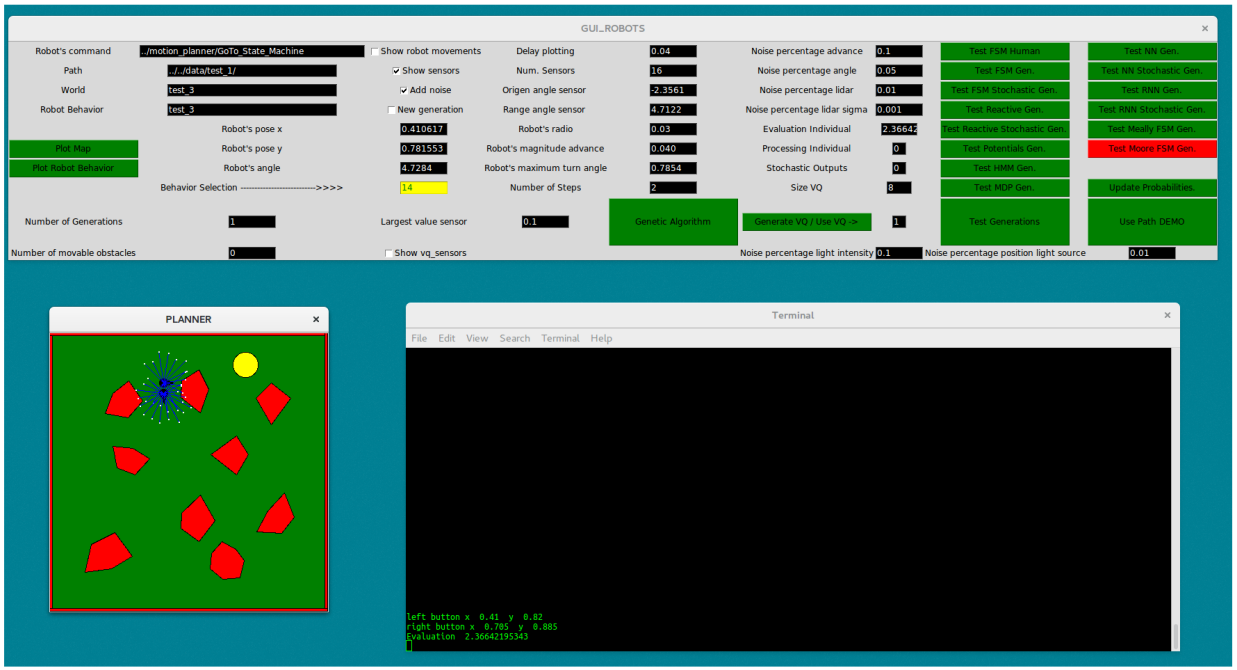
Behaviors GUI

6. Select the type of behavior in the test buttons in the right, in the PLANNER window select the robot's origin with the mouse's left button; and select the robot's destination with the mouse's right button. The environment to be tested can be changed in the World and Robot Behavior fields, in the following example a recurrent neural network behavior generated by a genetic algorithm is tested in the environment test_6.



Behaviors GUI

7. To display the robot's sensors select the check button Show sensors. To display the movement of the robot step by step select the check button Show robot movements



Behaviors GUI

7. There are 10 options for evolving behaviors:

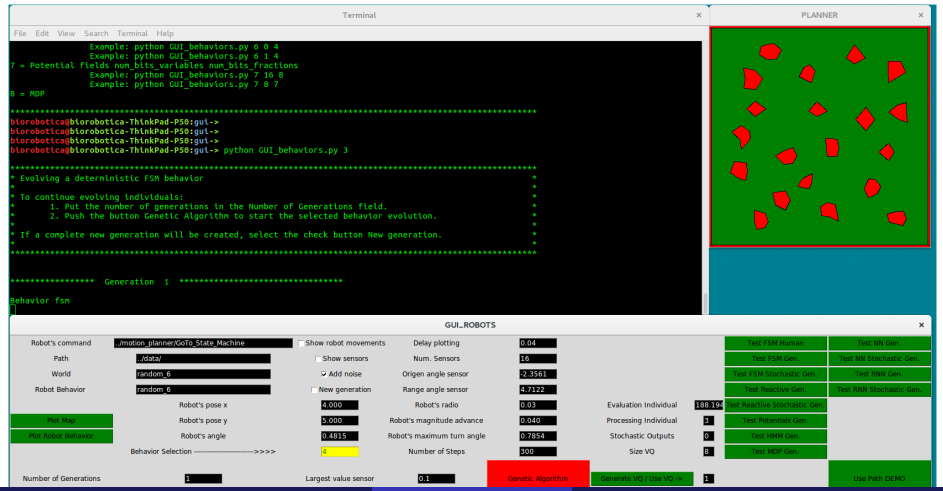
For example to evolve behaviors using Finite State Machines type the following

python GUI_behaviors.py 3

7.1. Put the number of generations in the Number of Generations field.

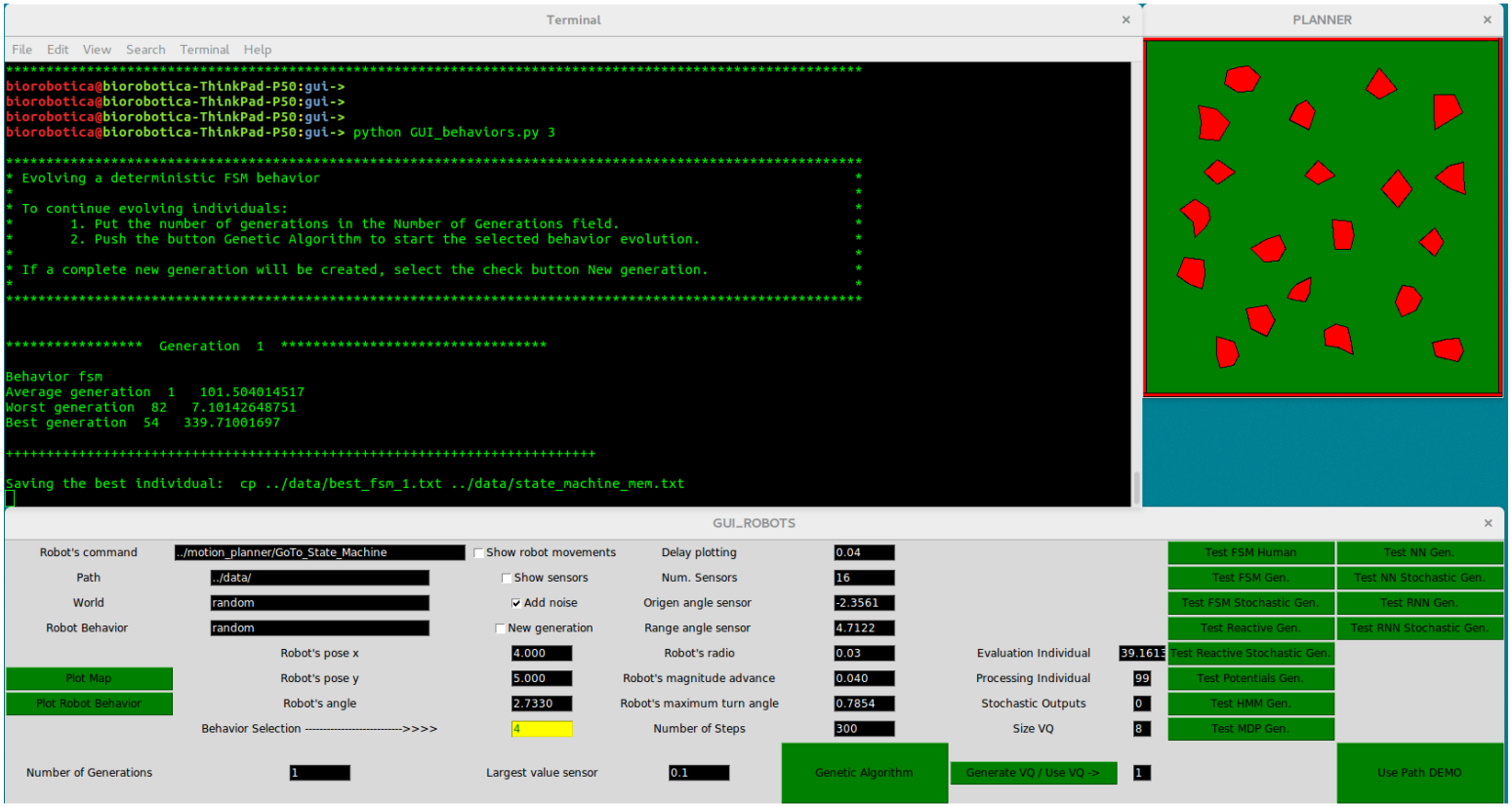
7.2. Push the button Genetic Algorithm to start the selected behavior evolution.

If a complete new generation will be created, select the check button New generation.



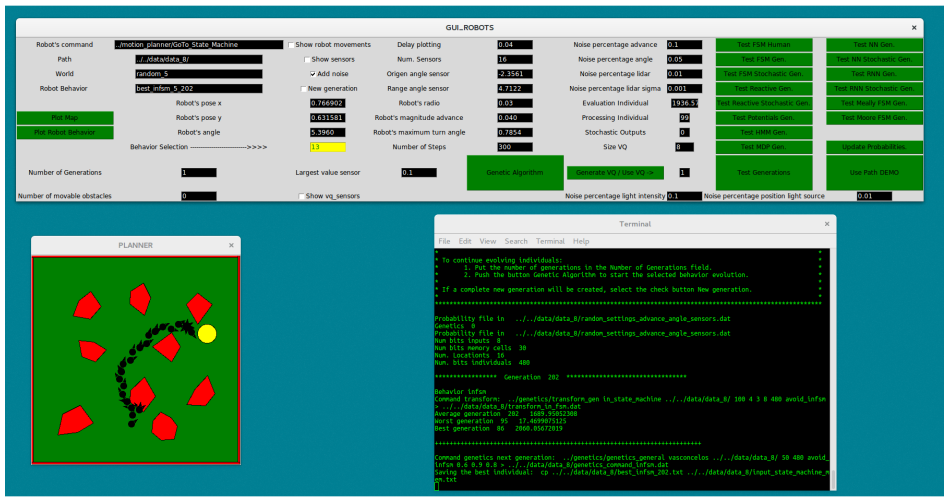
Behaviors GUI

For each evolved generation it shows the average fitness, the worst and the best generation.



Behaviors GUI

After several generations, the best individual can be seen by selecting in the Robot Behavior slot the best behavior file. For example, to select the best Mealy FSM behavior of generation 202, that was obtained on the environment random_5, put in this slot: best_infsm_5_202, and in the World slot random_5, as it is shown in the following figure. With few generations the performance of the best individual won't be satisfactory, but after several generations the performance will improve, depending also in the type of behavior evolved.

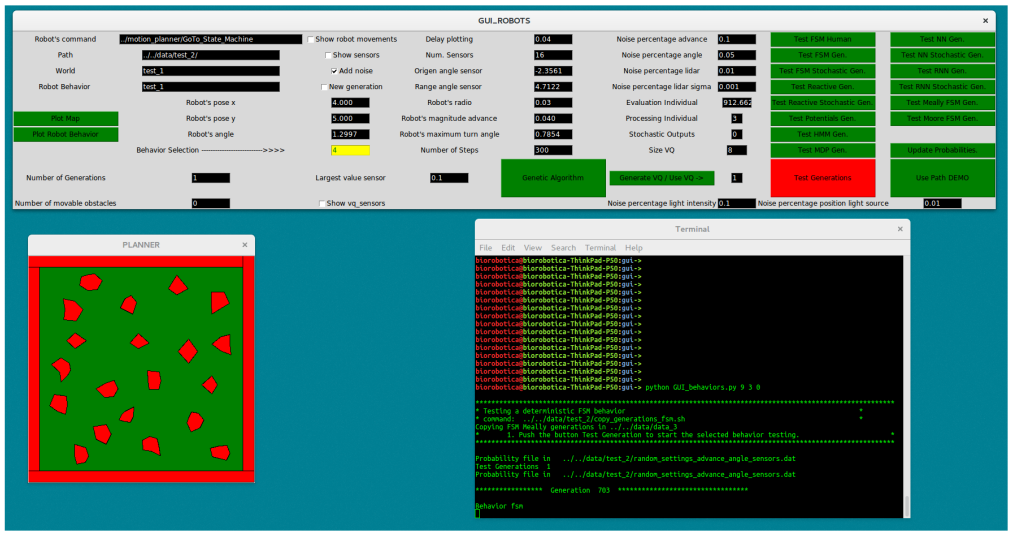


Behaviors GUI

The last generation can be tested and evaluated. For example for testing a complete generation with a behavior using Finite State Machines type the following:

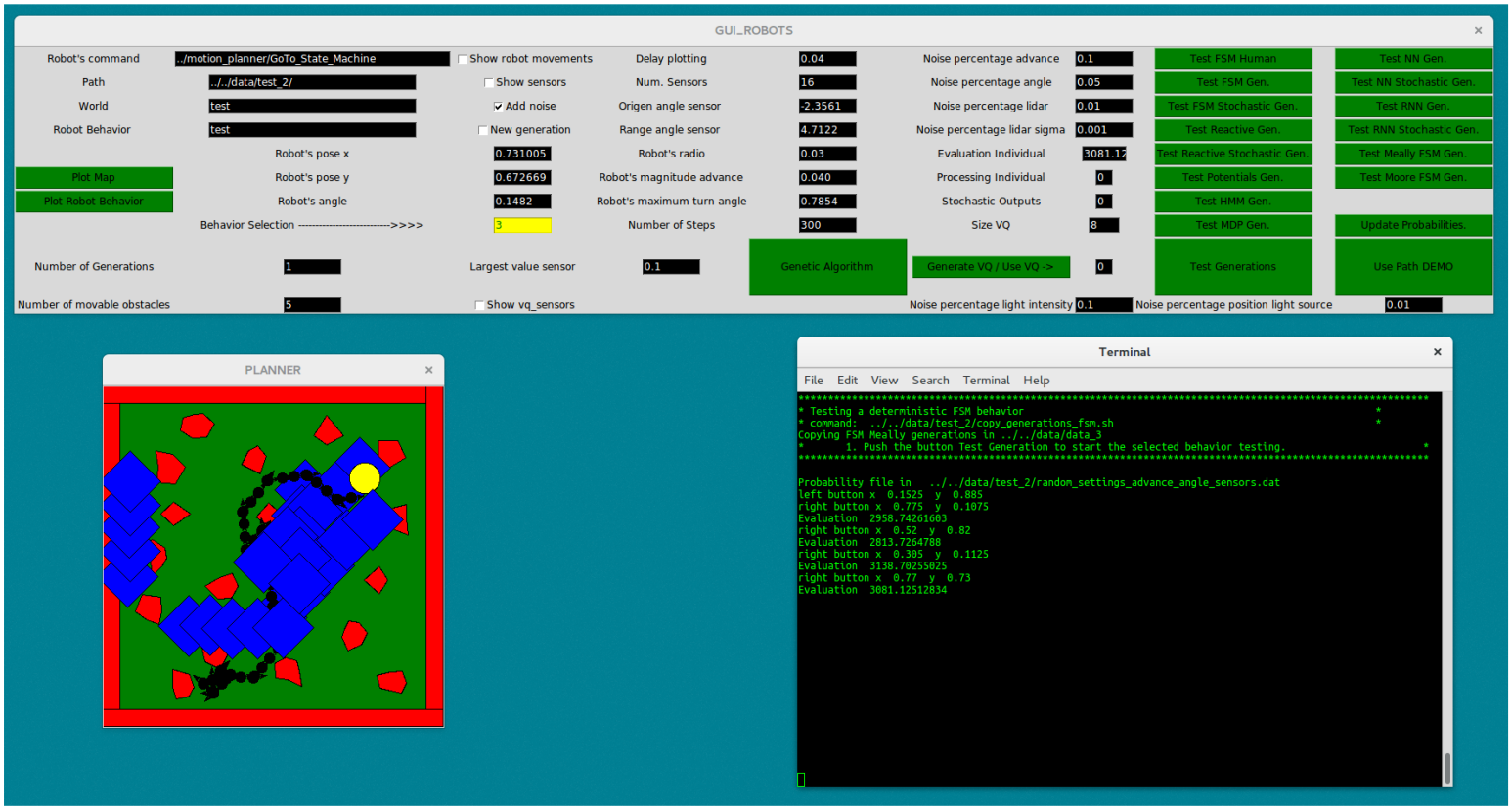
python GUI_behaviors.py 9 3 0

Then in number's of generations field put the number of generations to be tested, then select the button Test Generations. For each tested generation it shows the average fitness, the worst and the best generation.



Behaviors GUI

During the creation of the behaviors and also during testing it can be added movable obstacles by putting the number of them in the field of "Number of movable obstacles".



Behaviors GUI

During the creation of the behaviors the statistics of the evolved individuals can be visualized using MATLAB OR OCTAVE. File /home/user/robotics/genetics/plot_fitness_infsm.m plots the average, best and worst fitness functions for each of the behaviors evolved. Some of the graphics are shown in the following figure.