

Brief usage instructions for the program ER Simulator, v1.0

**Mattias Wahde
2004-09-21**

1. Introduction

The program ER Simulator, v1.0 allows the user to carry out a simple ER simulation, namely the evolution of an RNN brain for obstacle avoidance, in either a static or a dynamic arena. Note that while behavioral organization is not implemented in ER Simulator, v1.0, the program is a precursor to a general simulator implementing the UM method for behavioral organization. For more information, see

www.me.chalmers.se/~mwahde,

or send an e-mail to mattias.wahde@me.chalmers.se.

ER Simulator, v1.0 was written by Mattias Wahde (mattias.wahde@me.chalmers.se) and Jimmy Pettersson (jimmy.pettersson@me.chalmers.se).

The program was distributed as freeware to the participants of Mattias Wahde's tutorial on evolutionary robotics at IROS2004. The program is intended only as a demonstration of ER. The authors take no responsibility for the results obtained from this program – use at your own risk.

2. Setting up a simulation

Start the program by double-clicking on the program icon. The following window will appear:



Fig.1: The main window of ER simulator.

Before starting a simulation, the setup procedure must be completed. It consists of four stages, namely *experiment setup*, *arena setup*, *robot setup* and *evolutionary algorithm setup*.

Select **Setup – Experiment**. The following window appears:

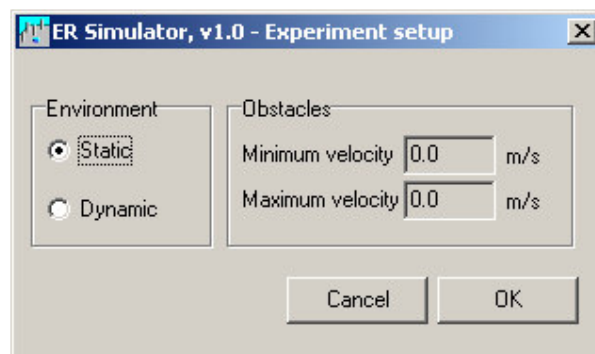


Fig.2: The setup window for the experiment.

Here, a choice can be made between static and dynamic arenas. In a static arena, all objects are stationary, whereas in a dynamic arena at least some objects are moving. The default selection is a static arena. Make a choice, and click **OK**.

Next, select **Setup – Arena**. The following window appears:



Fig. 3: The arena setup window.

Select **Browse** to search for an arena file. Four standard arena files have been included in the freeware version of ER Simulator, v1.0, namely

- staticarena.txt,
- small_staticarena.txt,
- dynamicarena.txt,
- small_dynamicarena.txt

As the names imply the static arenas are suitable for use in static simulations, and the dynamic arenas are suitable for use in dynamic simulations. However, it is possible to use any arena, regardless of the type of simulation. Note that the arena files are ordinary text files that can be edited, if the user wishes to generate a new arena. Each arena must contain an object of type TBasicFloor. It is possible to set textures on all arena objects. In order to make the program run sufficiently fast on e.g. a laptop, texture mapping has only been activated for the basic floor in the included arena files. However, the user may add any texture e.g. to the cylinders in the arena, by removing the comment symbol “#” in front of the texture definition line for a given cylinder, and setting the correct path to the texture image. For example, the following arena file would generate an arena with a floor and a single cylinder with the texture myimage.jpg: (not provided on the CD)

```
object BasicFloor: TBasicFloor
  Height = 0.1
  Position = 0.0 0.0 0.0
  Angle = 0.0
  Length = 2.0
  Width = 2.0
  TileLength = 0.25
  TileWidth = 0.25
  Texture = Textures/metalfloor.jpg
  RGBColor = 200 200 240
end

object Pillar1: TCylinderZ
  Height = 0.15
  Position = 0.29 0.00 0.0
  Angle = 0.0
  Texture = Textures/myimage.jpg
  RGBColor = 70 70 70
  Radius = 0.040
end
```

Note that all objects must have distinct names, and that, in ER Simulator 1.0, the velocities of the cylinders are set in the experiment setup (see above).

Returning to the setup, select an arena file, e.g. `staticarena.txt`, from the folder `Arenas`, and press **Open**. Next, in the arena setup window, press **OK**. The next step is the robot setup. Select **Setup – Robot**. The following window appears:

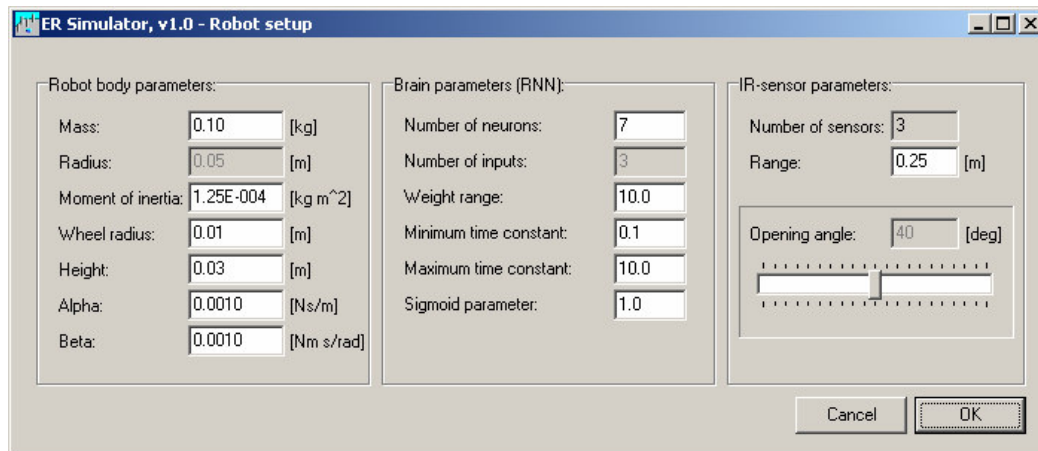


Fig. 4. The robot setup window.

Here, the parameters of the simulated robots can be selected. Note that the brain of the robot will be a recurrent neural network, and that it should contain at least 2 neurons (one for each motor). In case the small arenas are used, the sensor range should be reduced, e.g. to 0.15. After completing the selection of parameters, press **OK**.

The final step in the setup concerns the evolutionary algorithm. Select **Setup – Evolutionary algorithm**. The following window appears:

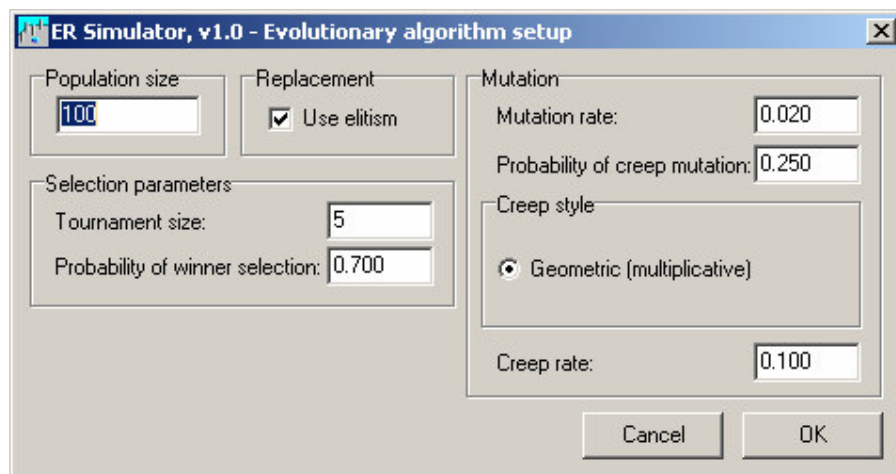


Fig. 5: The setup window for the evolutionary algorithm.

Select appropriate parameters (The default parameters are usually a good choice), and press **OK**. The setup is now complete.

3. Running a simulation

After completing the setup, select **Analysis – Run evolution**. Next, click **Initialize**. The window will have the following appearance (details may differ, depending on which arena was selected):

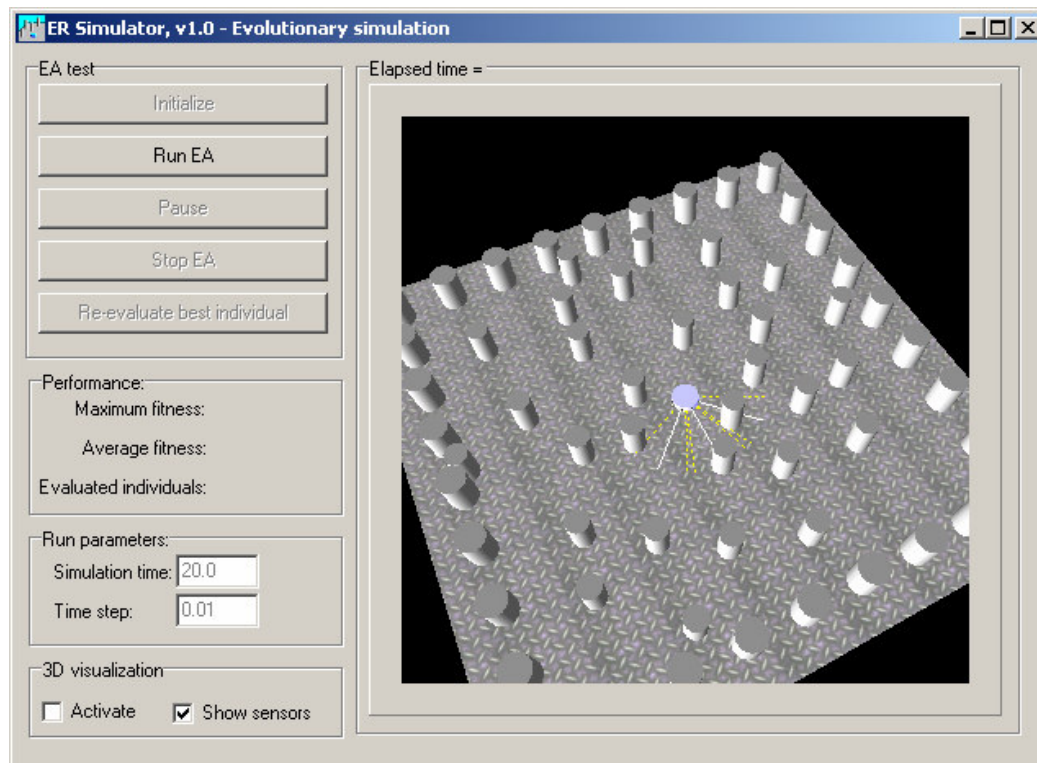


Fig. 6. The simulation window.

Next, press **Run EA**. The evolutionary algorithm starts to evolve robotic brains. The fitness measure is the total distance traveled in the forward direction. When the robot is moving backwards, it receives no fitness increase. The maximum simulation time is set under **Run parameters** in the window. The evaluation of a robot is terminated if it hits (or is hit by) an obstacle. At any point during a simulation, evolution can be paused (by clicking **Pause**). The current best individual can be re-evaluated by clicking **Re-evaluate best individual**. When the best individual has been re-evaluated, press **Resume** to allow the evolutionary algorithm to proceed. Note that it is possible to visualize every single individual, by checking the **Activate** check box (*not* recommended, as it will slow down the evolutionary algorithm considerably). In order to stop a simulation irreversibly, press **Stop EA**. Note that the EA must complete the current generation before it stops.

All windows have been adapted to a 640x480 screen size. However, the simulation window can be resized, if a larger window is desired. Note also that it is possible (using the mouse) to pan, rotate, and zoom the arena. Use the left mouse button to pan, the right mouse button to rotate, and both mouse buttons to zoom.

4. Troubleshooting

It *is* possible to crash the program, e.g. by selecting inappropriate robot parameters.

In case the program reports a “floating point error”, the problem may be caused by the regional settings used on the computer. To fix this problem, open the control panel, select “Regional and Language Options” (Win2000/XP Classic view) or “Date, Time, Language, and Regional Options” followed by “Regional and Language Options” (Win XP), and click “Customize”. Change the “Decimal symbol” to “.” (point, not comma).