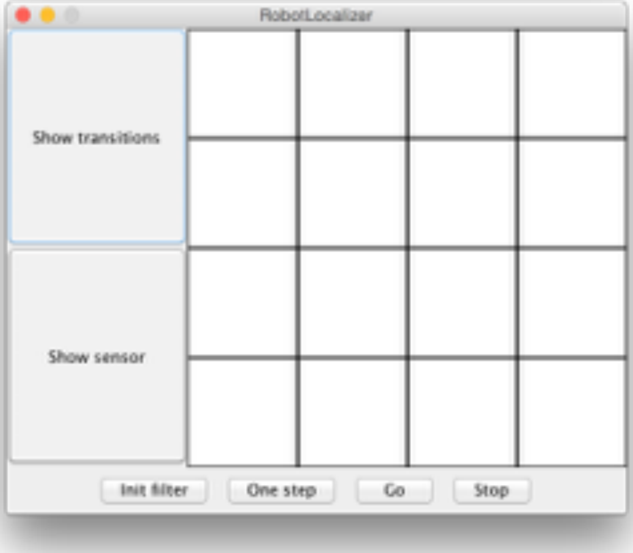
Using the RobotLocalisationViewer:

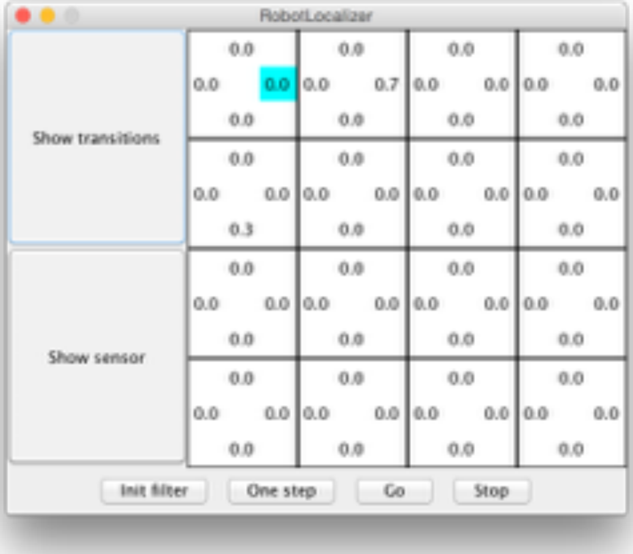
1 Starting:



The viewer starts up with an empty grid according to the dimension specifications retrievable from the EstimatorInterface methods. To get things going you can plug your own “localiser” (i.e., an instance of a class that implements EstimatorInterface) in the initialisation of the example main method (control.Main.java) - or you write your own main …

The figure shows the viewer for a 4x4-field grid

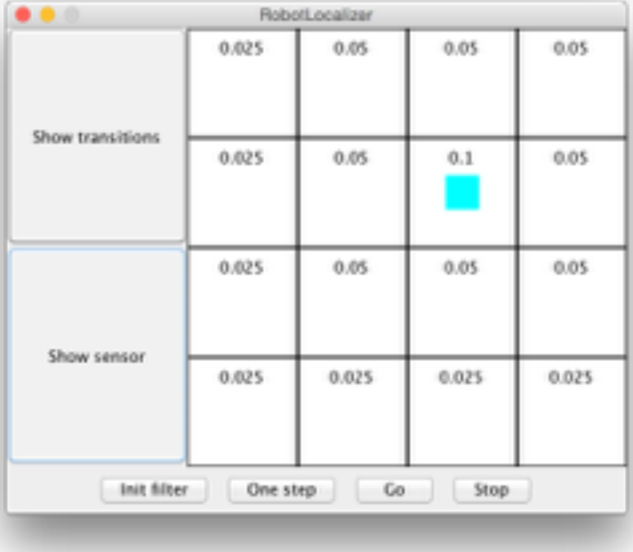
2 Checking the matrices



0.0 0.0 0.0 0.0	0.0 0.0 0.7 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.3	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0

2.1 Checking the transition matrix

Clicking on the “Show transitions”-button shows the probabilities for the different poses (x, y, heading) to be reached after having been in the given state (marked in cyan). Each further click steps through the states and wraps in the end. The figure shows the probabilities for state i = (0, 0, EAST) to end up in possible follow-up states. Only (0, 1, EAST) or (1, 0, SOUTH) are possible with p=0.7and p=0.3 respectively.

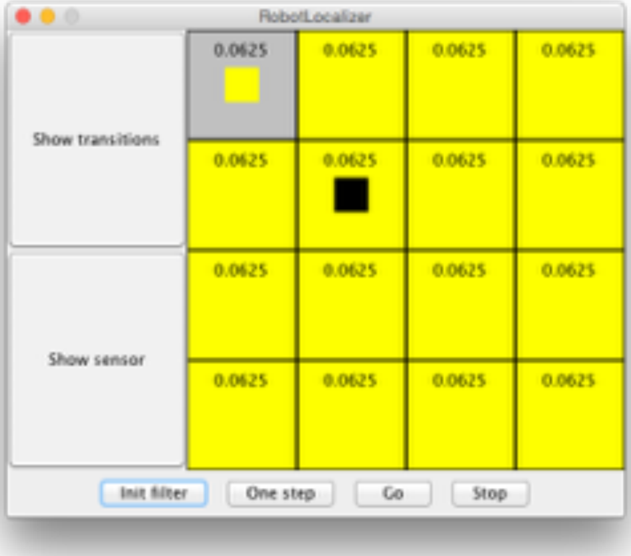


0.025	0.05	0.05	0.05
0.025	0.05	0.1	0.05
0.025	0.05	0.05	0.05
0.025	0.025	0.025	0.025

2.2 Checking the sensor matrix

Clicking on the “Show sensor”-button shows the probabilities with which the sensor reports the given position (marked cyan), when the robot is actually in the other positions. Each further click steps through the sensor readings and wraps in the end. The figure shows the probabilities for sensor reading r = (1, 2) in each position of the grid.

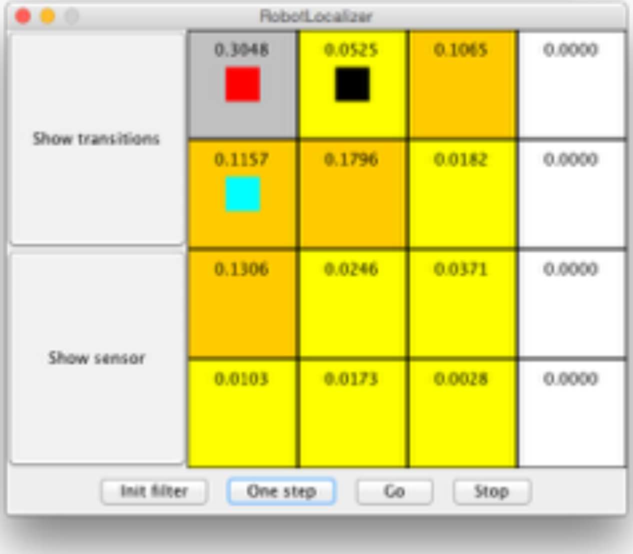
3 Visualising the filtering steps and results



0.0625	0.0625	0.0625	0.0625
0.0625	0.0625	0.0625	0.0625
0.0625	0.0625	0.0625	0.0625
0.0625	0.0625	0.0625	0.0625

3.1 Initialising

Clicking on “Init filter” initialises the viewer / localiser. This step is necessary to get further steps running properly (and it shows you the initial state of the grid). The figure shows the starting position (black) at (1, 1) and the probability distribution (normally equally distributed). A light grey “ring” marker is on the field with highest probability (i.e., the left upper corner, as everything is the same).
Please note: clicking “Init filter” again does not do anything apart from skipping the sensor reading marker in one step - the viewer is currently meant to be started for a new run from scratch.



0.3048	0.0525	0.1065	0.0000
0.1157	0.1796	0.0182	0.0000
0.1306	0.0246	0.0371	0.0000
0.0103	0.0173	0.0028	0.0000

3.2 Stepping through

With “One step” you advance one step in the filtering process (hence the update() method for one step). Black: true position, cyan: sensor reading, light grey: highest probability (only the “first” one encountered is marked). White: assumed impossible (p = 0), yellow: low probability (0.0<p<=0.1), orange: higher probabilities (0.1< p <=0.3), red: “high” (p>0.3). In the upper part of each field the actual probability is shown (truncated to four decimals).

3.3 Running continuously

With a click on “Go” you start the loop over the steps (delay according to time parameter for the driver thread).
“Stop” interrupts the loop, it is possible to go stepwise again - or loop again.

Please note: The numbers shown in the last figure are only examples of the visualisation - do not use them for validation of your results!