

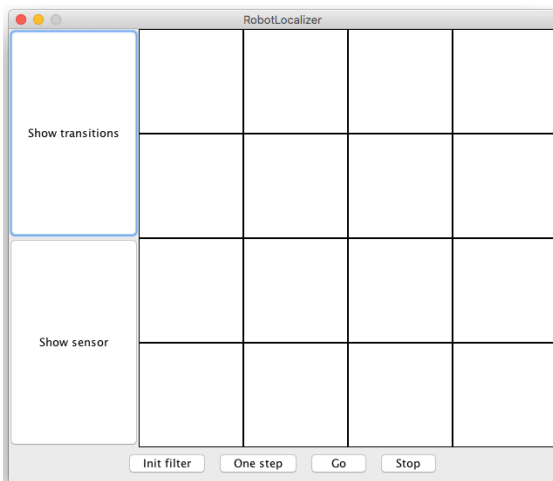
Using the RobotLocalisationViewer:

The RobotLocalisation Viewer assumes a state coding based on triplets (x,y,h), with x being the row, y the column (together the position) and h the heading of the robot in the grid world, with x=0 being the top row, y=0 the leftmost column and h running around the compass from 0 to 3 as NORTH-EAST-SOUTH-WEST. Whatever your state coding looks like internally, you will have to provide the visualisation tool with values according to this assumption.

For the sensor readings it assumes to receive n*m probabilities, one for each position (x, y), to have caused the reading r = (rX, rY) or r = (-1, -1), which means “nothing”. See the comments in the EstimatorInterface for details!

Make sure that you provide the data from the observation and transition matrices that you actually **use in your implementation** in the respective methods.

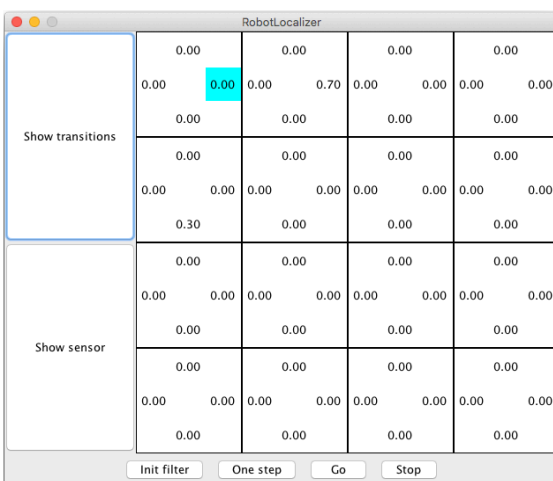
1 Starting:



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. Please observe: The dimensions shown here (4x4) are ONLY an example, easy to fit in the document. Your implementation should consider BIGGER layouts!

2 Checking the matrices



2.1 Checking the transition matrix

Clicking on the “Show transitions”-button shows the probabilities for the different poses (x, y, h) 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 (0, 0, EAST) to end up in possible follow-up states. Only (0, 1, EAST) or (1, 0, SOUTH) are possible with p=0.7 and p=0.3 respectively.

