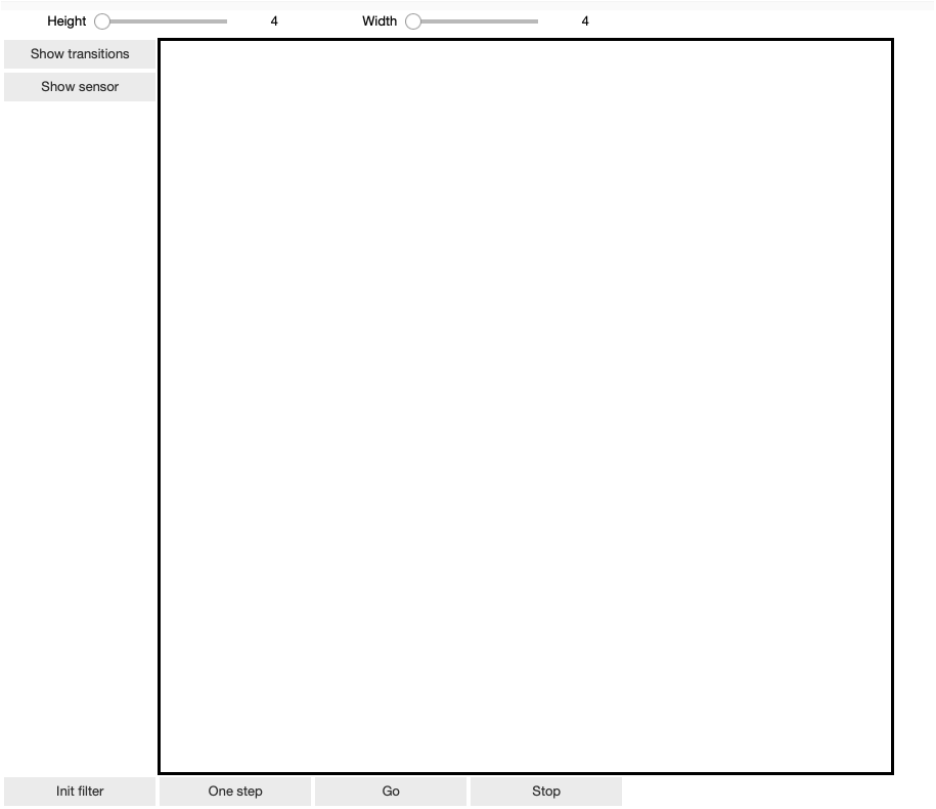
Using the RobotLocalisationViewer:

Note that this description is based on the Java-version of the handout; colours / visual experience in the Python version might differ slightly.

The RobotLocalisationViewer 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.

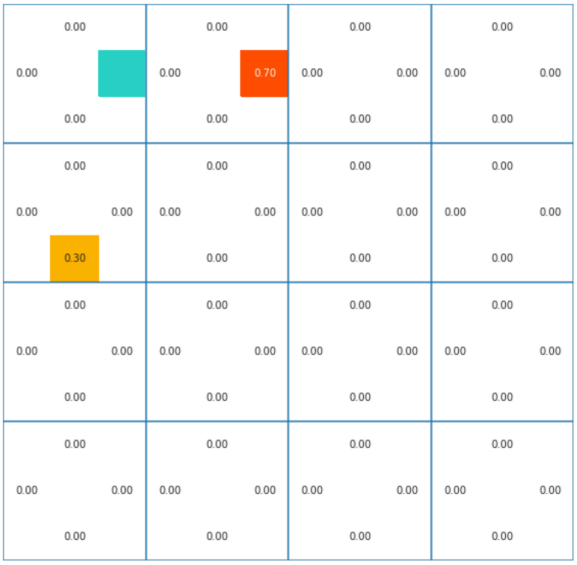
1 Starting:



The viewer starts up with an empty field, two sliders for the dimensions of the grid, and six control buttons.

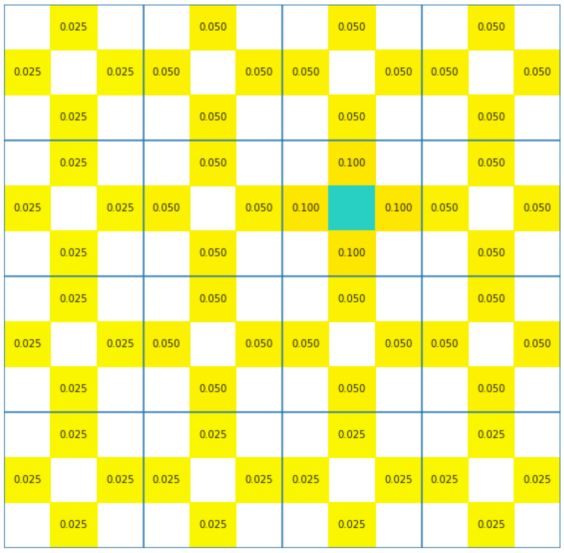
The figure shows the viewer for a 4x4-field grid. Please observe: The dimensions shown here (4x4) are ONLY an example, easy to fit in the document. Your implementation should consider BIGGER layouts!

2 Checking the models (transition and observation)



2.1 Checking the transition model

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.

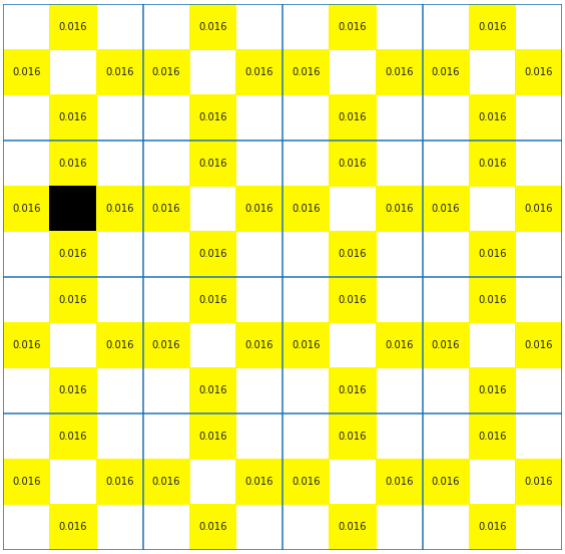


2.2 Checking the observation model

Clicking on the “Show sensor”-button shows the probabilities with which the sensor reports the given position (marked cyan) or “nothing” (all white), when the robot is actually in the other states (poses). Each further click steps through the sensor readings and wraps in the end.

The Python viewer shows the numbers as heat map, so “all white” only refers to the center cells in the visualisation.

3 Visualising the filtering steps and results



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 state (black) at (1, 0, 1) and the probability distribution you start out with.
Please note: clicking “Init filter” again resets everything except the size of the grid, that is done with the sliders.

3.2 Stepping through

With “One step” you advance one step in the filtering process (i.e. call the update() method in the Localizer once). Black: true position/state, cyan: sensor reading, red: estimated (guessed) position. The heatmap shows the probabilities for the states (poses).

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

3.3 Running continuously / stopping

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.