What are some ways a robot can navigate?
- Reactive "controls"
  - bug/wall following
  - "follow the leader"
  - etc.
- Explicitly planned paths
  - EFT
  - Wavefront

Grid maps are useful as output for path planning
- Human consumption
- Long term localization (scan matching)
- Underlying graph structure

Nice Properties:
- Easy to implement
- Straightforward to represent spectrum of traversal costs
- Easy to extend to arbitrary dimensions (though we're going to stick to 2D)
- Cost guarantees for planning w/easy to understand parameters

Sad Properties:
- Quantization error
- Fine resolutions quickly become cost prohibitive
- Issues w/dynamic environments
- Consistency/maintenance problems (localization is very important)

Tradeoffs: how do you pick a scale for your grid?
- Higher resolution: more accurate representation of environment (to some extent), but expensive
- Lower resolution: fast planning, but go too coarse and traversable routes may disappear!

Eg. Doorways

Rule of thumb: use as coarse a resolution as you can afford
Data Representation
- Typically ternary (occupied, known free space, unknown)

This allows you to easily write frontier-based exploration planners which send your robot(s) to points where reachable known free space meets unknown

Writing a gridmap
- Need a consistent reference w.r.t. world if you will be adding observations over time
- Typically, we use axis-aligned coordinates and designate some corner as our reference point in the real world. e.g.

\[
\begin{array}{c}
\begin{array}{c}
\text{grid step} = 1 \text{m} \\
(10.0, 15.0) \text{ in global coordinates}
\end{array}
\end{array}
\]

- Now you can feed point observations into your map to set grid values, e.g.

\[
\text{grid-set(grid, global-x, global-y, OCCUPIED)}
\]