Group meeting 27-05-2019

Questions

Q: Are we allowed to perform manual alterations to the supplied .json file?

A: Yes. Note: supplied .json should have Y pointing upwards.

Q: Can sleep rates be altered?

A: Yes. Perhaps r.sleep() takes input?

Q: Backup plan for localization

A: Maybe build this in.

Q: What happens when cabinet is not found?

A: Maybe build in skip cabinet if it doesn't work. Add mapping update?
Note: Cabinet corners might also be internal.

To Do

- RRT (Ruben + Bram)
  - Path planning based on .json map
  - Wall blowup improvements
  - Optimize tree (later)
    - Unknown whether short (enough) path is planned
- Dynamic obstacle avoidance (Marcel)
  - Static addition of velocities
    - Update with correct orientation
    - Update with velocity vs. distance
  - Add non-semantic avoidance
  - Convert corner (+walls/doors/etc) arrays to vectors (Ruben)
- Static obstacle avoidance (Later)
  - Add maximum effort to determine large objects to add to map and plan again
- Location fitting (Martijn + Jeroen)
  - Optimize with given start area (if necessary)
  - If possible, add dynamic localization
- JSON processing (Jeroen)
  - Check x- and y-directions for incoming data
- Create sample hospital maps (Jeroen)
- Wiki (All)
  - Update when conceptual functions are complete
- Move through planned trajectory (Bram + Ruben)
  - Cycle through points of trajectory
  - Move to new point just before reaching old one
  - Acceleration and deceleration only at start and end
  - Mind issues with odom reset and path planning

    - Prioritize rotation when moving backwards
        - Rotation check?

Test goals

Two rooms, with matching .json and sim map
Main priority: Localization
Plan to two cabinets, first far away, second back in start area, move to both
Initialization to RRT planning.
Obstacle avoidance → Test case

Agreements

- Script cleanup
    - While working on functionality, keep an eye out for outdated classes, functions, names, et cetera, and fix
- Where relevant, use methods over functions
- Use camelCase for naming variables and functions