open robotics

On multi-fleet operations with the Open Robotics Middleware Framework

19 Oct 2022, ROSCon JP 2022, Kyoto, Japan

Open Robotics Stewarded Platforms

ROS

- World's most widely-used open-source SDK for robotics
- ROS is to robotics as Android is to mobile devices

ros.org

GAZEBO

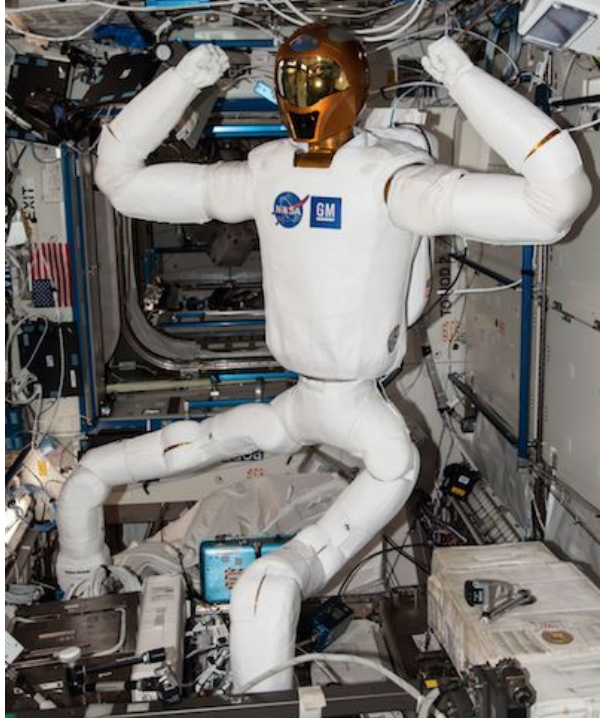
- Open-source robot simulation software
- Gazebo is to robotics what AutoCAD is to architecture

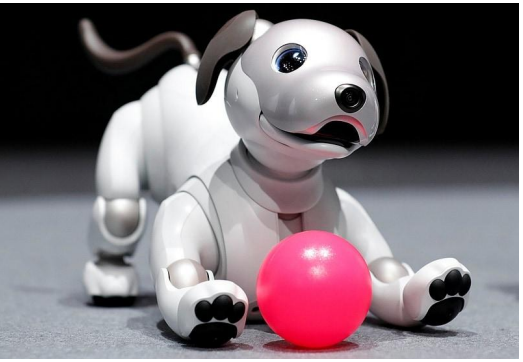
gazebosim.org

Open-RMF

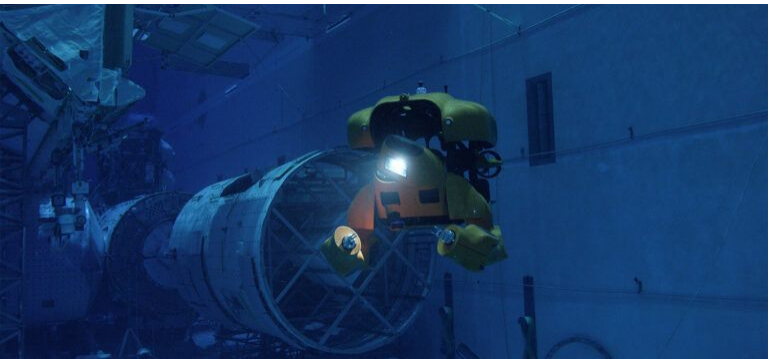
- Open-source multi-fleet management and interoperability software
- Open-RMF is to robotics as SAP is to operations

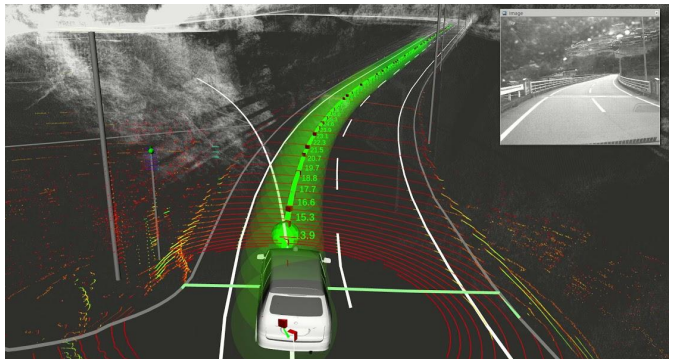
open-rmf.org



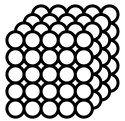


We’re asking more of robots (a lot more)





The industry is growing dramatically

~15% 

Global robot **density** CAGR

>1K 

global robot suppliers with **diverse** solutions

88% 

companies **planning robotic adoption**

53% 

companies listing **cost of adoption** as a top 5 challenge to adoption

Source IFR

Source IFR

Source Zippia

The interoperability dilemma

Many Amazing Robots

Cleaning

Security

Logistics

Delivery

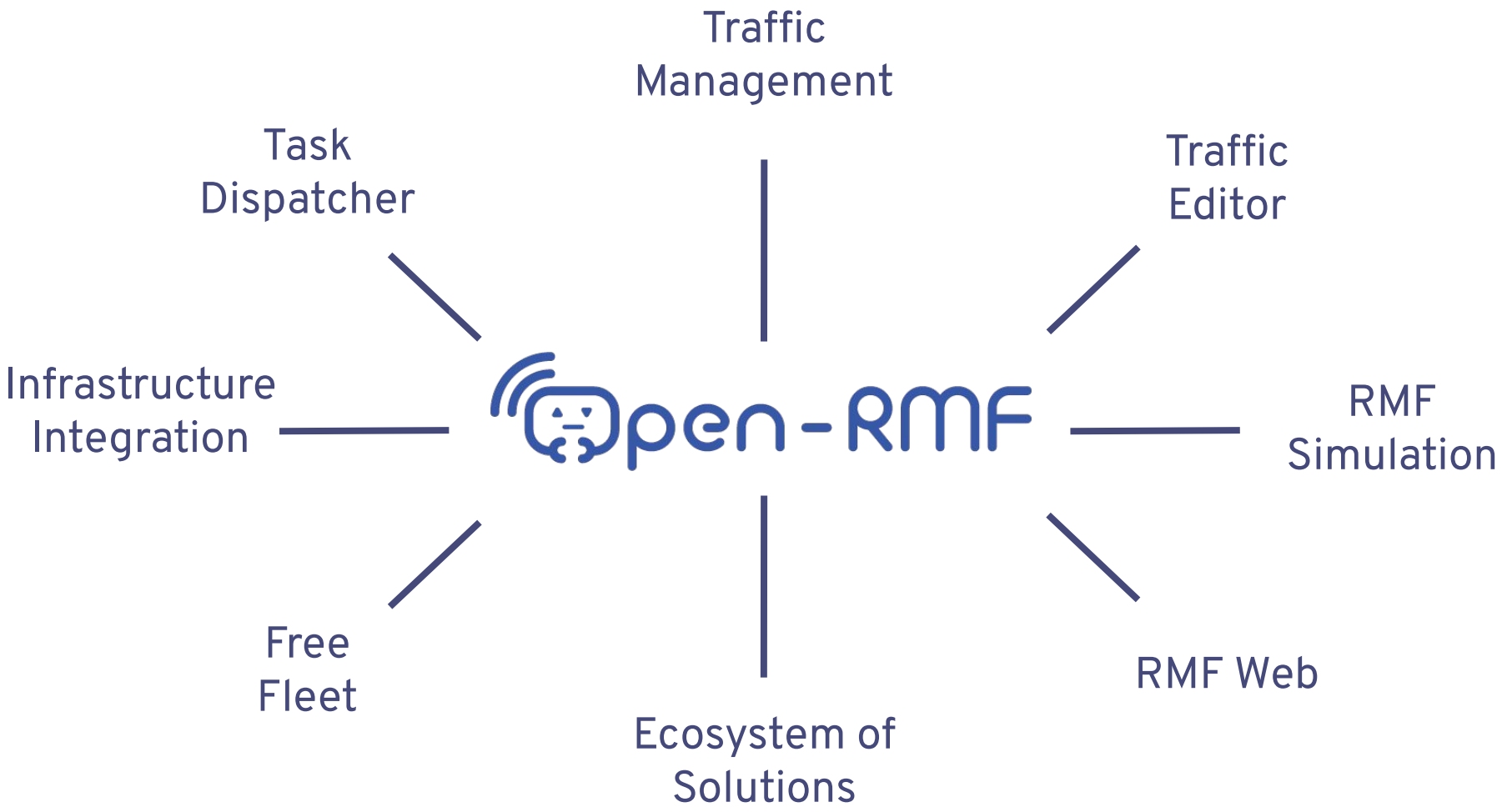
but they need to communicate with each other

Guten Tag!

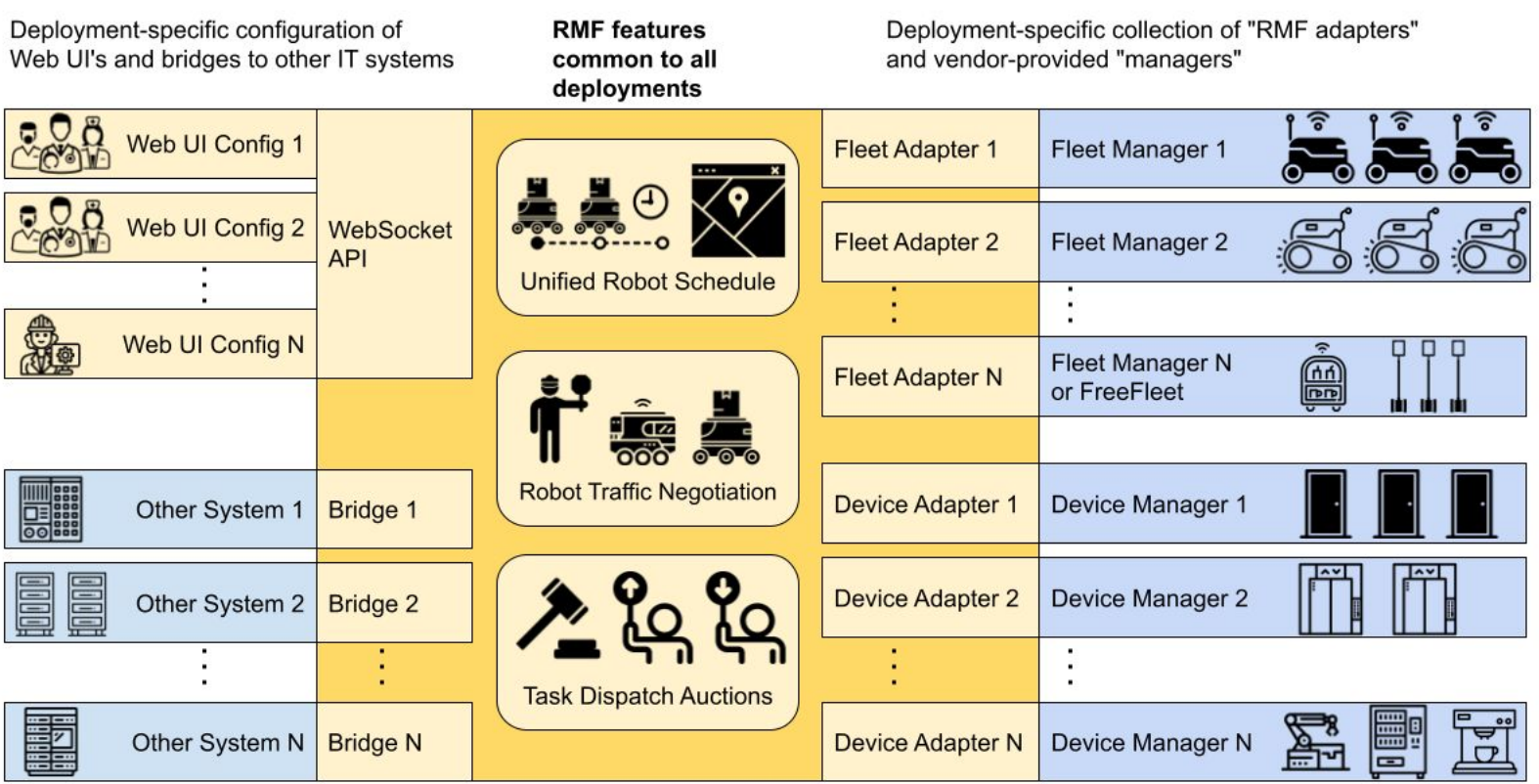
こんにちは

and resources have already been spent to develop and deploy disparate systems

We can't all get along


Traffic Management
Task Dispatcher
Traffic Editor
Infrastructure Integration
Open-RMF
RMF Simulation
Free Fleet
RMF Web
Ecosystem of Solutions

What is Open-RMF?



Open-RMF Creates Market Opportunities

Facility Owner/ Operator

Reduces TCO and frees up resources for further growth

Photo by maitree rimthong

Robot Vendor

Reduces integration costs, complexity and time

Photo by Lenin Estrada

Infra Provider

Reduces the total number of integrations

Photo by Steve Johnson

Solution Provider

Reduces integration costs, complexity and time

Photo by Pixabay

Robot Providers

AETHON

CLEARPATH ROBOTICS™

Gaussian Robotics

Hanwha

MiR A BETTER WAY

LIONSBOT

OTTO MOTORS

ncs

SESTO™

Panasonic

SENSERBOT

temi

swisslog

TurtleBot™

UBIQUITY ROBOTICS

Collaborators

Centre for Healthcare Assistive & Robotics Technology (CHART)

CHANGI airport group

GOVTECH SINGAPORE

IHiS Because IT's in Healthcare

MINISTRY OF HEALTH SINGAPORE

Robotics Empowering People NATIONAL ROBOTICS PROGRAMME

ROS industrial consortium asia pacific

Infrastructure

ADEMCO

FUJITEC

dormakaba

KONE

Xnergy Autonomous Power Technologies

SI's and Solution Providers

clobot

Formant

FREEDOM ROBOTICS

HOPE technik

HOST® A Sonepar Company

KABAM ROBOTICS

m2Labo. AGRINNOVATION

Medisys Innovation Innovation · Domain · Technology

Panasonic CONNECT

robotmanager™ by NCS

TR

Application Fields

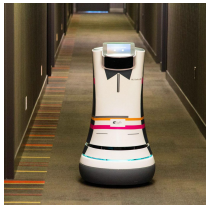
Construction

Healthcare

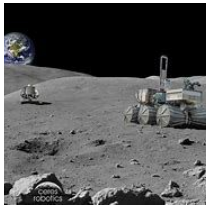
Airports

Seaports

Manufacturing & Logistics



Hospitality



Space

Agriculture

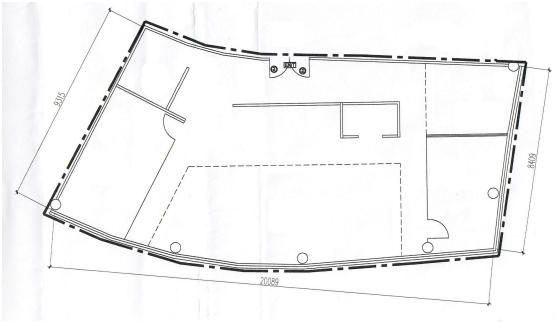
TRAFFIC EDITOR

Start using RMF in three easy steps

1. Find or create a building map
2. Create a simulation model
3. Make a robot traffic map

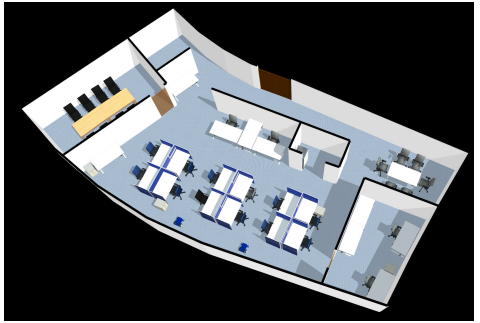
Step 1 may require human interaction.

Traffic Editor helps you with steps 2 and 3.



Building floorplan

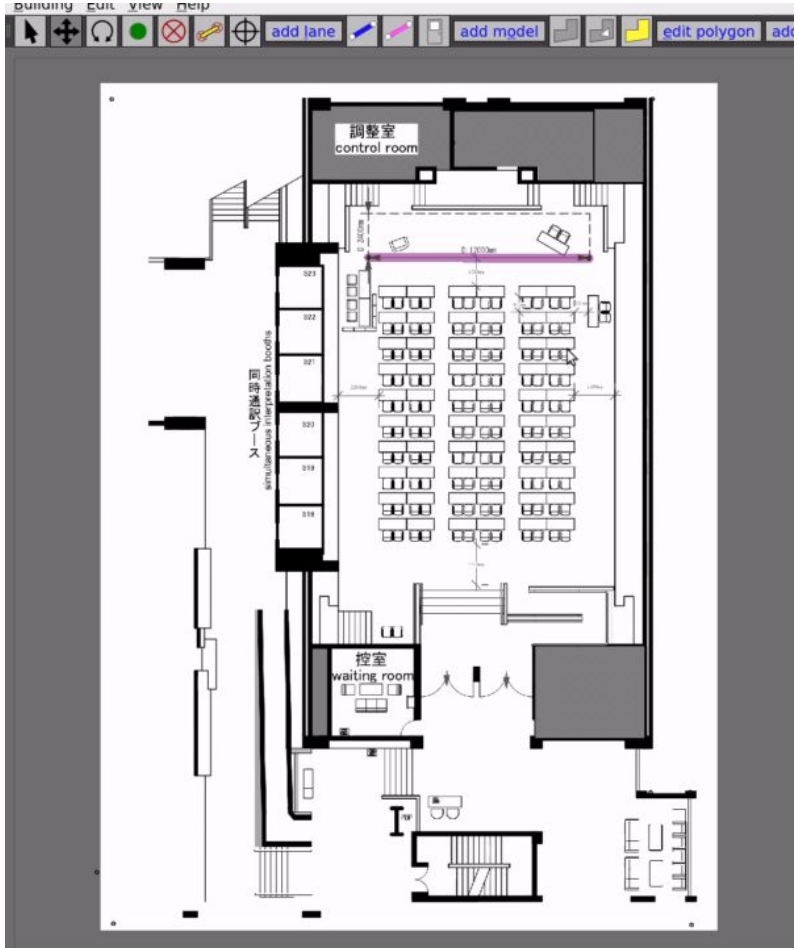
Annotated in ***traffic_editor***



Physics-based Gazebo simulation world

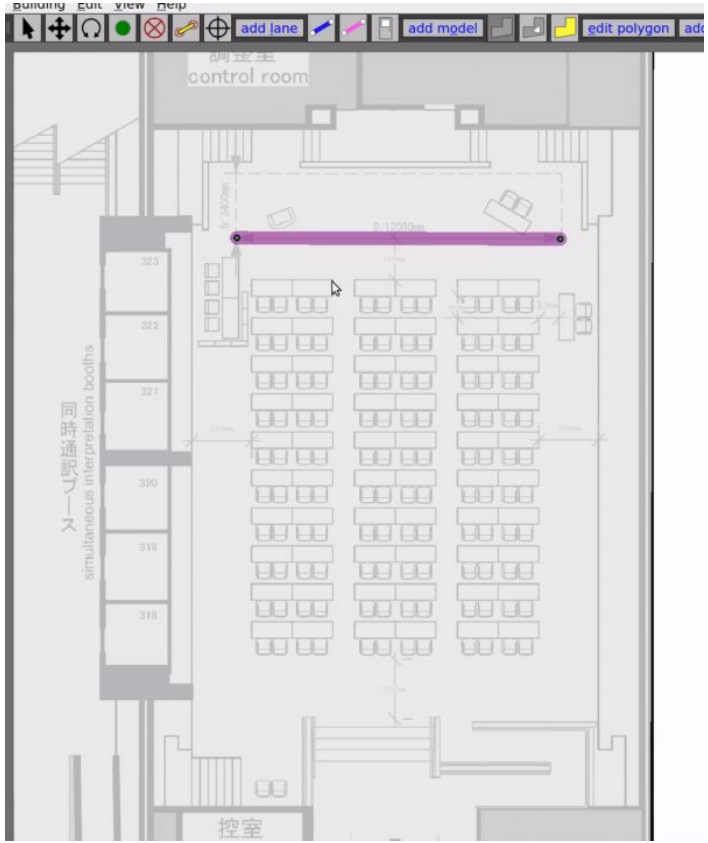
Create floor polygons

- First we need to define the floor polygon
- It can be precise to just look nice, or just a bounding-box so robots don't fall down
- Each floor polygon has the same material
- Multiple polygons can be used to model different flooring types



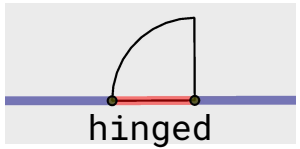
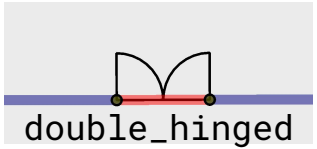
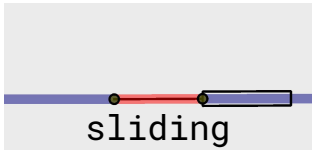
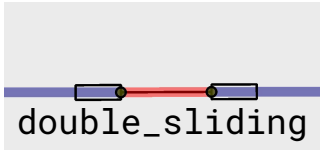
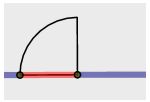
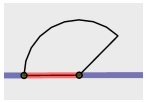
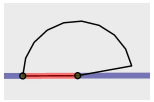
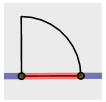
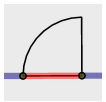
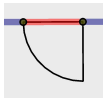
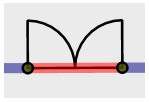
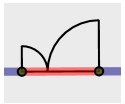
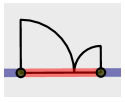
Create walls

- Walls can be traced from the floor plans
- Typically not all walls are needed or wanted
- Materials (textures) can be defined for each segment



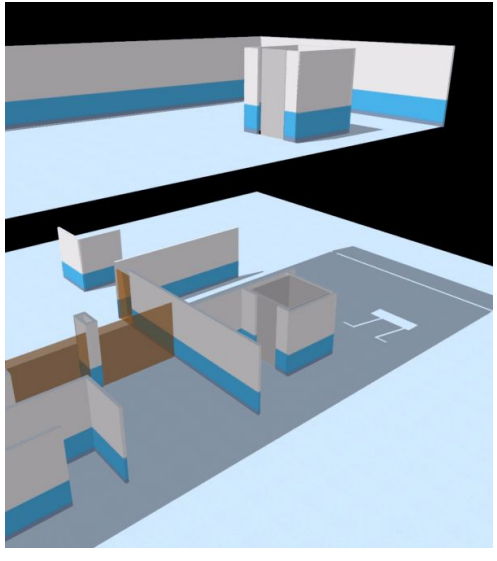
Configure door parameters

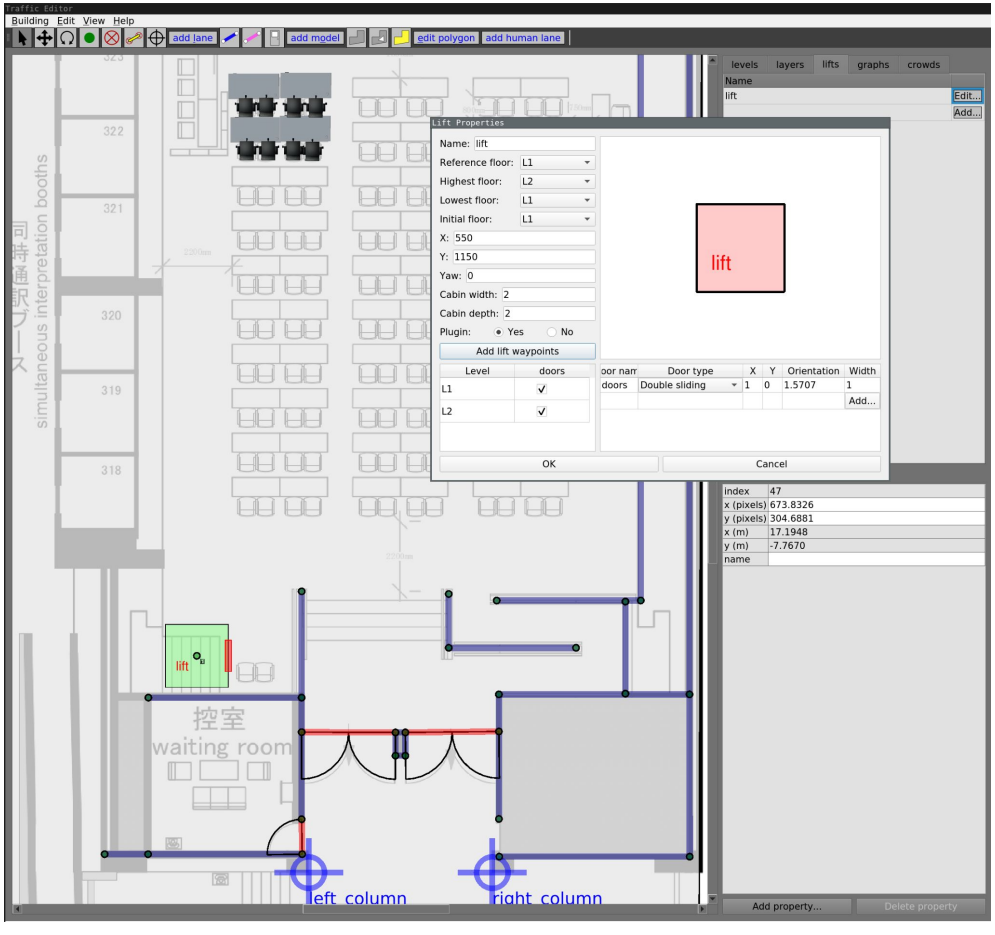
- Doors are surprisingly complicated!
- Use the property editor to configure them

type	 hinged	 double_hinged	 sliding	 double_sliding
motion_degrees	90 	135 	170 	
motion_axis	start 	end 		
motion_direction	1 	-1 		
right_left_ratio	1 	2 	0.5 	

Configure elevators

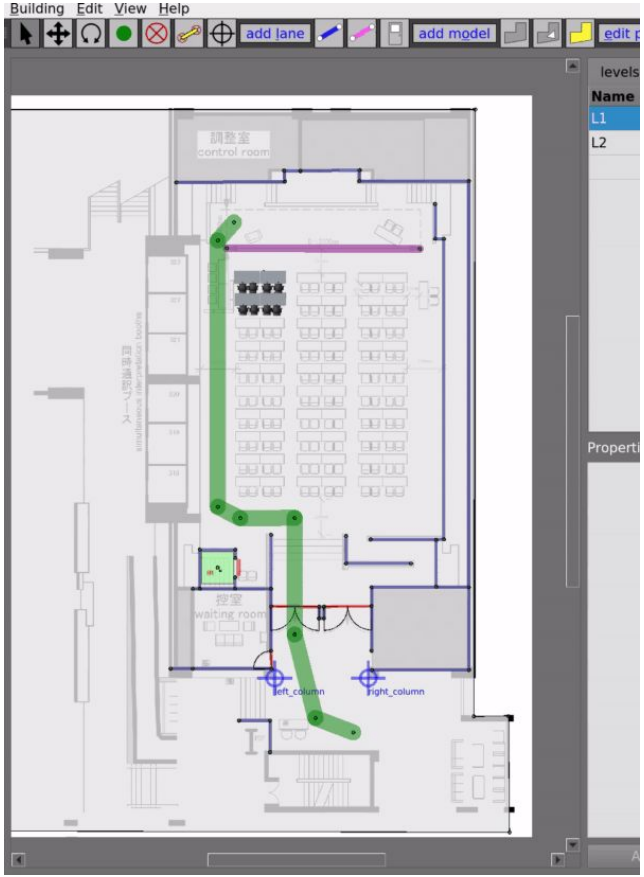
- Elevators (lifts) are a critical part of any multi-level robot site
- They are incredibly expensive (!)
- They often set the throughput limit for the entire system
- Careful modeling is important





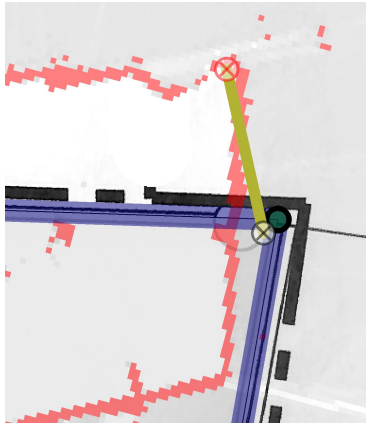
Drawing robot traffic lanes

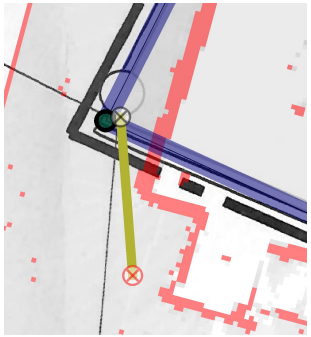
- Traffic lanes can be drawn on the map
- This helps simplify the planning problem and make the robot motion patterns easier to understand for people nearby
- Traffic lanes can be extended, branched, moved, etc., in the GUI
- Next question: how do we align this with the robot coordinate system?

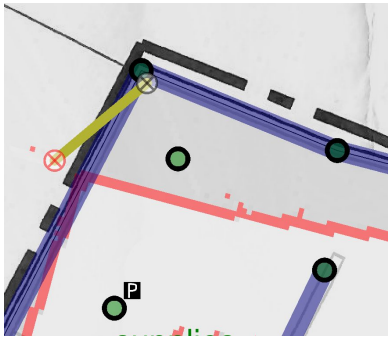


Aligning real-robot maps

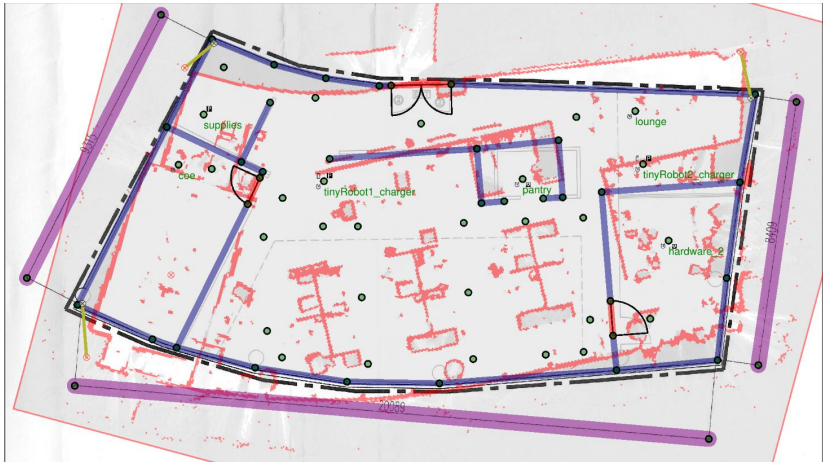
- Fully-automatic alignment is hard
  - As-built often differs from floorplan, especially wall partitions
  - Cabinets, pillars, chairs, etc., can occlude walls
- Humans can easily spot these issues and ignore them
  - This is difficult to program for computers
- "Semi-Automatic" alignment workflow:
  - Roughly align the robot map using manual transform parameters
  - Click "features" that you can see in both robot map and floorplan
    - corners of load-bearing walls
    - pillars
  - Click "constraints" between corresponding features
  - Select Edit->"Optimize Layer Transforms" (Ctrl+T)
    - Numeric solver tries to find a transformation that minimizes the constraint lengths

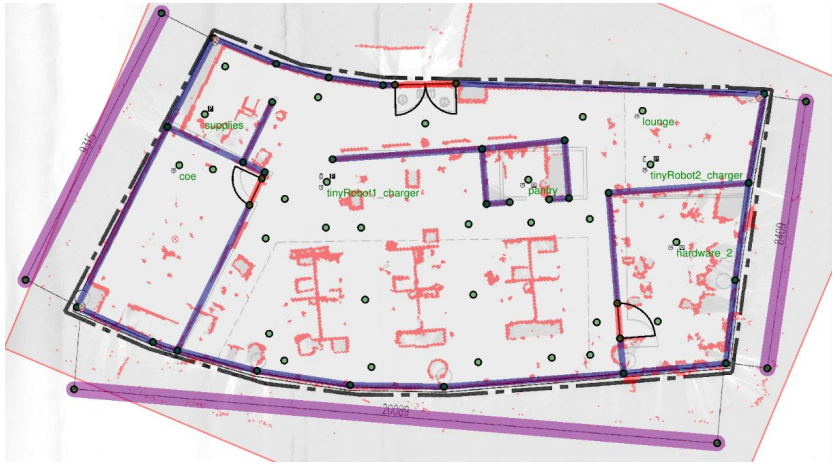






Aligning maps with numerical optimization

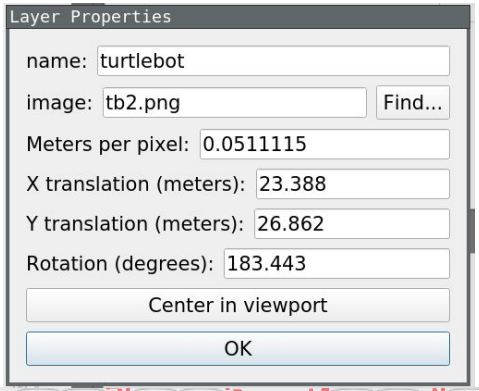


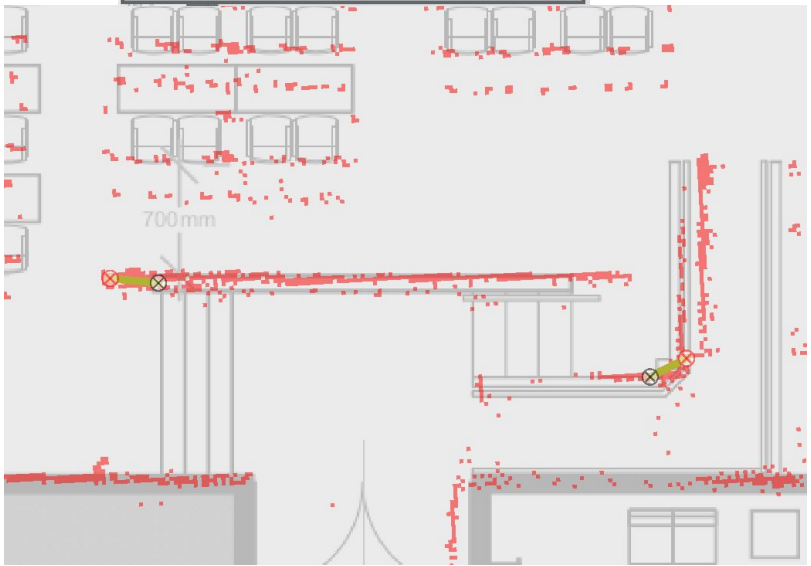


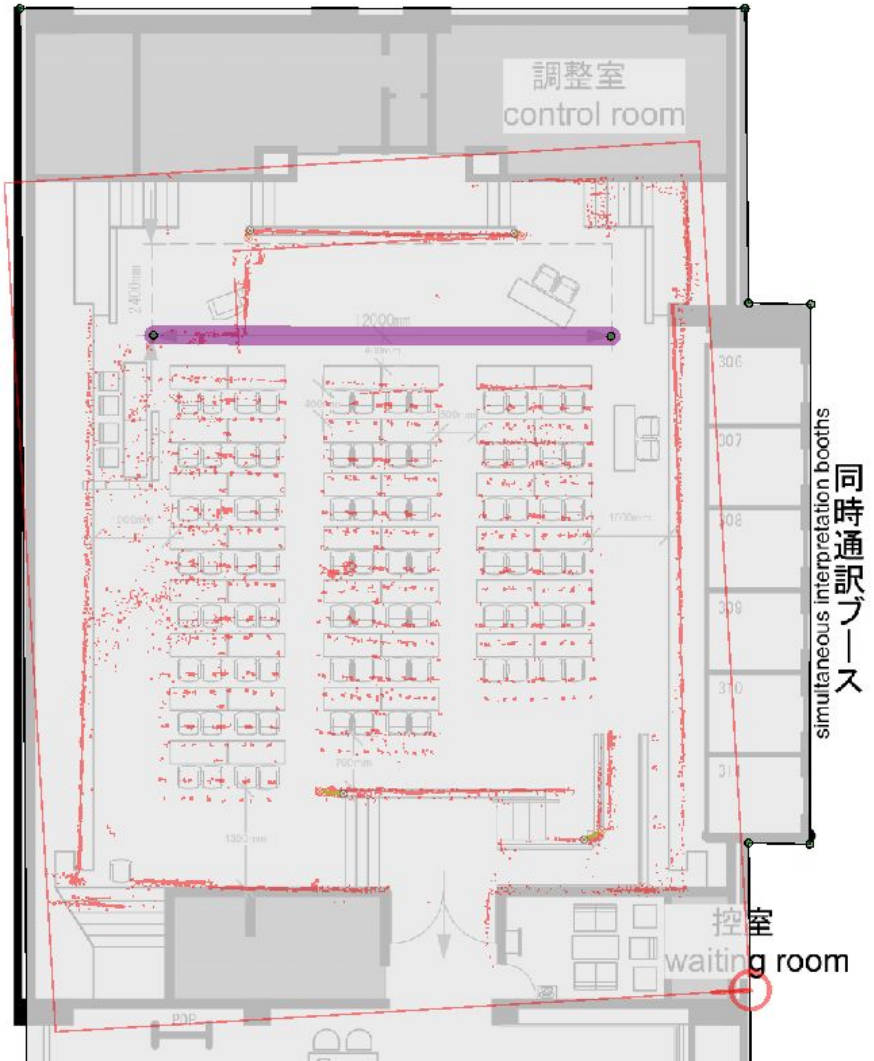
name:	turtlebot
image:	office_A4.png Find...
Meters per pixel:	0.05
X translation (meters):	5.000
Y translation (meters):	-5.000
Rotation (degrees):	-20.000
	Center in viewport
	OK

name:	turtlebot
image:	office_A4.png Find...
Meters per pixel:	0.0484015
X translation (meters):	6.904
Y translation (meters):	-5.651
Rotation (degrees):	-22.811
	Center in viewport
	OK

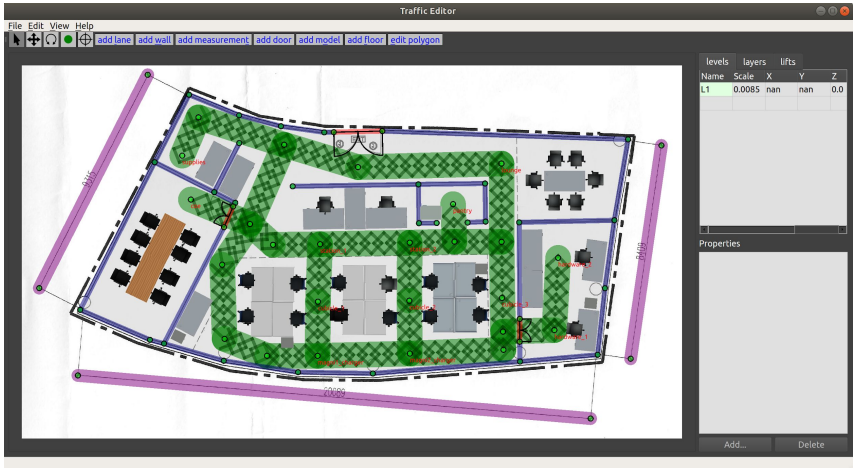
Aligning maps





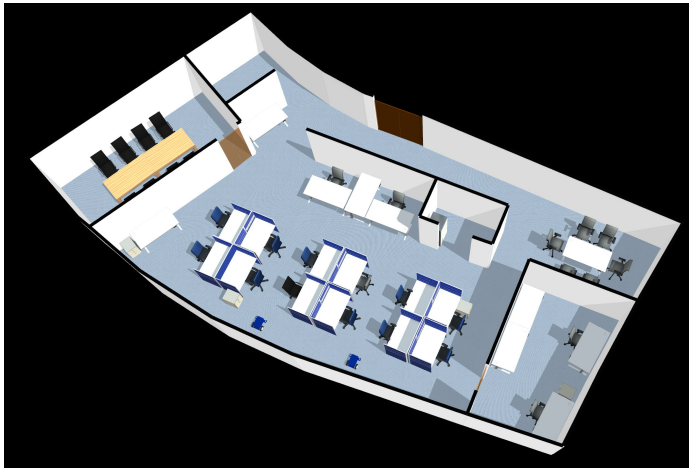


Building Map Tools



Annotated map in ***traffic_editor***

building_map_tools



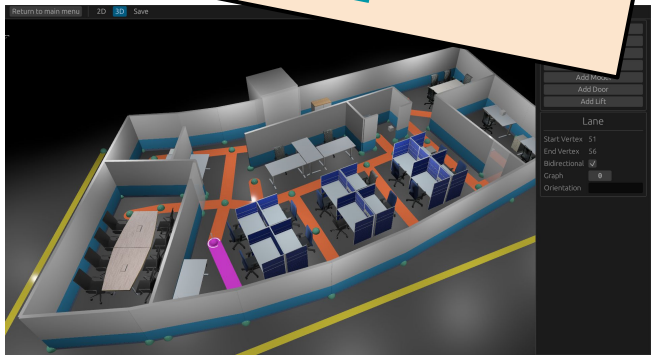
Physics-based simulation world with 3d assets, robots, doors, plugins

http://github.com/open-rmf/rmf_traffic_editor

https://github.com/open-rmf/rmf_site

The RMF Site Editor

Try the webGL version in your web browser!
https://open-rmf.github.io/rmf_site/



Next generation of RMF site editing

- Full 3D previews of the simulated facility, updated live as you edit it
- Runs natively on Linux, Windows, Mac, or web browsers
- Edit the placement of furniture, walls, doors, lifts, traffic lanes, human crowd behavior, cameras, and lights
- Easy to extend with custom plugins for application-specific features

Upcoming Features

- Preview camera feeds
- Save and link "read-only" portions, to reuse site information across different scenarios and allow manual additions to upstream data

Beyond

- A single application that can run on any computer (including Windows) to simulate a full deployment of RMF. All RMF nodes would run inside the site editor, running the same code as the real deployments. Users can add their own custom nodes as plugins.

CAPABILITIES

Overview

- Traffic Negotiation
  - Full Control
  - Traffic Light
  - Read-Only
- Shared Infrastructure
  - Doors
  - Elevators (Lifts)
  - Workcells
- Task Management
  - Customize
  - Dispatch
  - Execute

Traffic Negotiation

Full Control: Robot/Fleet API accepts arbitrary waypoint commands
Traffic Light: Robot/Fleet API accepts pause/resume commands
Read-Only: Robot/Fleet API only provides current location and destination information

	Full Control	Traffic Light	Read-Only
Can be tracked and displayed on a dashboard			
Can be avoided as an obstacle			
Can yield to higher priority agents			
Can stop at intersections to prevent simple conflicts			
Can reroute to avoid or resolve complex conflicts			
Can reroute for optimal traffic flow			

Future work
Native Integration: Negotiation takes place on board the robot and takes advantage of the robot's perception

Traffic Negotiation

The robots will negotiate whether to wait for each other or reroute depending on which is computed to require less time.

Traffic Negotiation

When rerouting is not possible, the robots will wait on each other for as long as necessary.

Head-to-head conflicts are anticipated within the long stretches, and one robot in the head-to-head conflict will stay out of the long lane while the other robot is approaching.

Shared Infrastructure

- Doors
- Elevators (Lifts)
- Workcells

The `rmf_fleet_adapter` library synchronizes robot traffic with the use of shared infrastructure

Shared Infrastructure

Robots can negotiate their traffic through a shared door.

Shared Infrastructure

Robots can also negotiate traffic around others that are using doors.

Shared Infrastructure

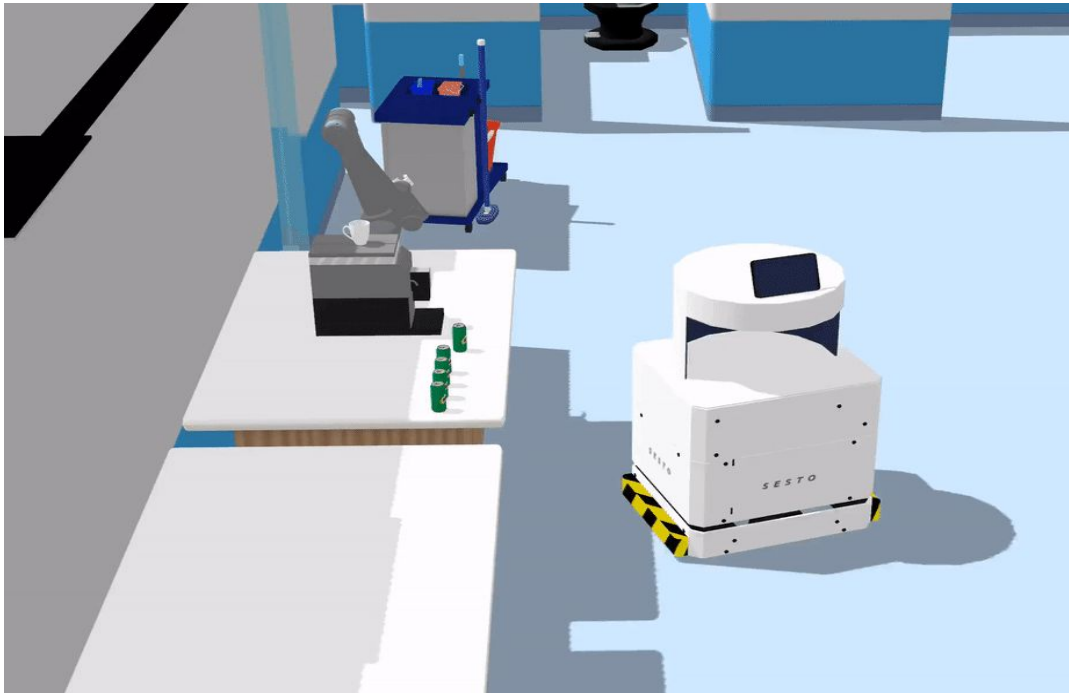
Robots can negotiate the use of elevators.

RMF integration allows any RMF-compliant robot to use any RMF-compliant elevator.

If certain elevators should be reserved for certain robots, that can be expressed in the navigation graphs that are used by the fleets.

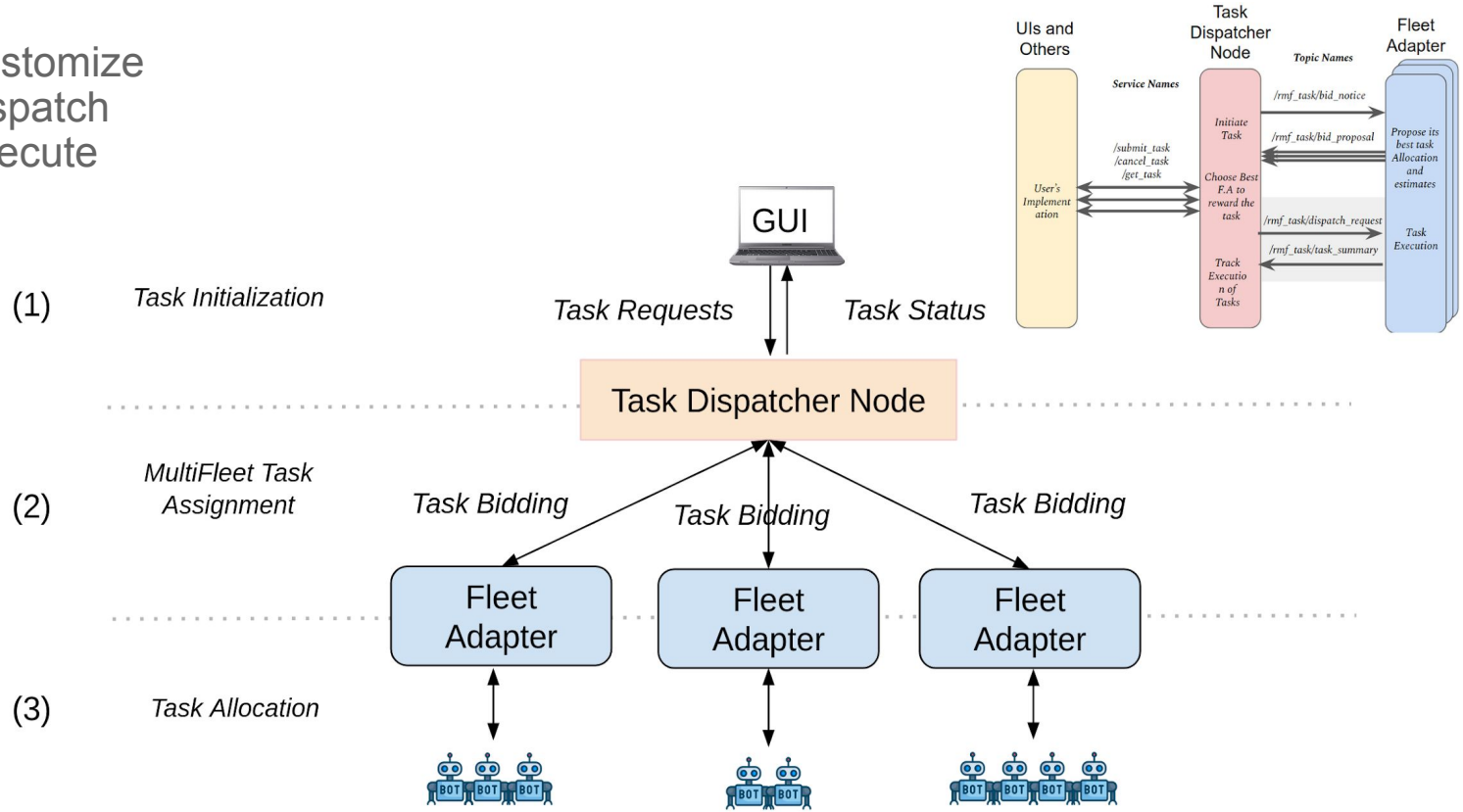
Shared Infrastructure

Workcells, such as robot arms, have standardized interfaces that allow the fleet adapter to coordinate pickups, drop-offs, docking procedures, or other interactions between robots.



Task Management

- Customize
- Dispatch
- Execute



What is a "Task"?

Task Description

serializable data structure that can be interpreted into...

Predictive Model

inputs: (initial state prediction, robot description)
output: predicted state after task completion

Provided to a multi-agent task planner to search for a "minimum-cost" assignment of tasks to robots

Runtime

generates a sequence of task "phases"

Task Phase

monitors state of robot and infrastructure to issue commands (e.g. navigate to location, open door, summon elevator) to fulfill an objective of the task

Human operators or external systems can request that a phase is skipped or repeated. This is helpful if a phase did not go as intended.

⚠️ **Current Scope** ⚠️

The current implementation assumes **each task** is assigned to **one mobile robot** and that individual tasks **do not depend** on each other.

Future versions of RMF will support **multi-agent tasks** and **constraints between tasks**.

Task Descriptions

More detailed instructions:
https://osrf.github.io/ros2multirobotbook/task_new.html

Simple, premade

```
{
    "category": "delivery",
    "description": {
        "pickup": {
            "place": "L2_pharmacy",
            "payload": [
                {"sku": "48052", "quantity": 2},
                {"sku": "37981", "quantity": 1}
            ]
        },
        "dropoff": {
            "place": "L3_ward32_bed4"
        }
    }
}
```

Common tasks can be given simple premade description schemas with a minimal set of parameters to fill in

Each **category** is associated with its own **description** schema that can be interpreted by task planners and executors.

Custom, composed

```
{
    "category": "compose",
    "description": {
        "detail": "Drop off medication and then greet the patient",
        "phases": [
            {
                "activity": {
                    "category":"pickup",
                    "description": {
                        "place":"L2 pharmacy",
                        "items": [{"sku":"48052", "quantity": 2}]
                    }
                }
            },
            {
                "activity": {
                    "category":"dropoff",
                    "description": {
                        "place":"L3 ward10 bed4",
                        "items": [{"sku":"48052", "quantity": 2}]
                    }
                },
                "on_cancel": [{
                    "category":"dropoff",
                    "description": {"place":"L2_pharmacy"}
                }]
            },
            {
                "activity": {
                    "category":"greet",
                    "description": {
                        "place":"L3 ward10 bed4",
                        "language":"Hokkien"
                    }
                }
            }
        ]
    }
}
```

Allocation of tasks

A* Solver

- Priority Assignment
  - Add a **Priority** field to task request
  - During node expansion, check if new node assignments are valid
    - Valid = high priority tasks are assigned prior to low priority ones
  - **If invalid, f(n) = g(n) + h(n) * penalty**

- Finishing Task
  - Automatically include a task that the robot has to perform at the end of its assignments
  - Park, ChargeBattery, etc

- Fleet adapters automatically replan task assignments when a task is cancelled

- Fleet adapters use rmf_battery to check they should automatically return an idle robot to its charger

Execution

Interrupt / Resume / Cancel

- Tasks can be interrupted at any time
  - Not all actions can be interrupted
  - Non-interruptible actions will finish, and then the task will switch to an interrupted state
- When a robot's task is interrupted, the robot can be given direct commands without conflicting with any commands RMF
  - Useful when a robot needs to be teleoperated
- Tasks can be canceled at any time
  - When a task is canceled, it will perform a "cancelation sequence" that depends on how far the task progressed

THANK YOU!!!

Questions?