Unlocking the Potential of Robotics Through Automated Negotiation

presented by Grey
for the ROS-Industrial Annual Summit 2023

Motivation

Even “intelligent” mobile robots struggle to behave cooperatively.

Robots **MUST** be able to share space to play a larger role in our everyday lives.

Traditional warehouse methods for controlling traffic flow are not suitable for uncontrolled and unstructured environments.

Scope

Tackle challenges preventing mobile robots from being widely deployed:

- How do they coordinate the sharing of space? e.g. crossing paths in a corridor
- How do they share infrastructure? e.g. automated doors and elevators
- How do they synchronize with other devices? e.g. a robot arm that loads a delivery

Modest scale—not too much robot density (for now):

- ✅ Hospitals (clean, telepresence, schedule and ad hoc deliveries)
- ✅ Malls, hotels, airports (clean, deliver, security, assistance)
- ✅ Libraries (clean, deliver, scan bookshelves)
- ❌ Dense warehouse

Assumptions

- Robots from multiple vendors will be needed
  No single vendor offers a robot that can do everything

- Not all mobile robots can offer the same inputs/outputs or capabilities
  Different vendors have different APIs and levels of control available
  Some platforms may have more or less “intelligence” than others, e.g. AGV vs AMR

- Dynamic environments with unpredictable elements
  The robots need to operate amidst human traffic
  Furniture or other items may be moved, creating unanticipated static obstacles

- Centralized command & control is not (always) an option
  Different platforms might have their own fleet management tools
  They might get installed and maintained by different system integrators

Design of **Open-RMF** (Robotics Middleware Framework)

- Shared traffic schedule
  All robots openly communicate their intended trajectories through space-time
- Negotiate to resolve conflicts
  A peer-to-peer negotiation process, similar to “conflict-based search” is used to fix traffic conflicts when they arise
- Auction tasks to find the “most available” agent
  If multiple different robot platforms can perform a task, they will “bid” to see who can get it done at the “lowest cost”
- Provide a stable (while evolving) SDK
  We are constantly improving the implementation and algorithms used
  We maintain stable APIs in the C++ and Python SDKs so the efforts of early adopters are not wasted
  New features/capabilities become available by expanding the APIs of the SDKs

✕ No specification or stable wire protocol (for now)
The wire protocol often needs to change as the algorithms improve
Users should integrate using the SDK, which hides the details of the wire protocol and maintains compatibility

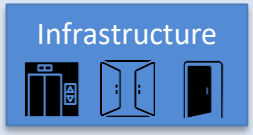
Fundamental Components

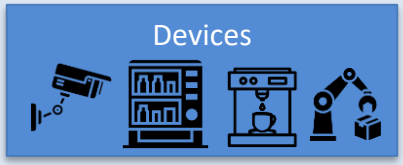
Drop-in executables that help coordinate distributed systems

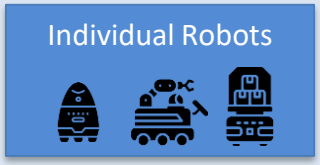
Reusable Software Libraries

Use these inside your own custom apps

Platform Adapters







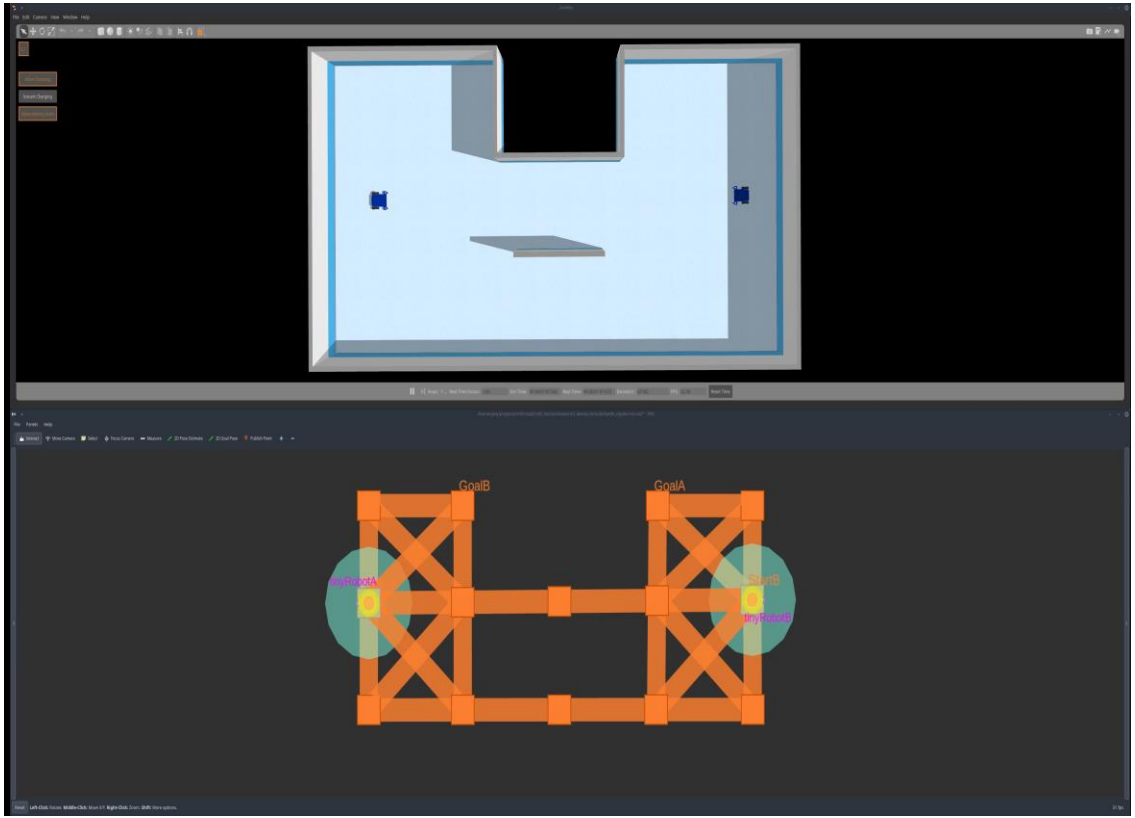
Connect hardware to your system

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			

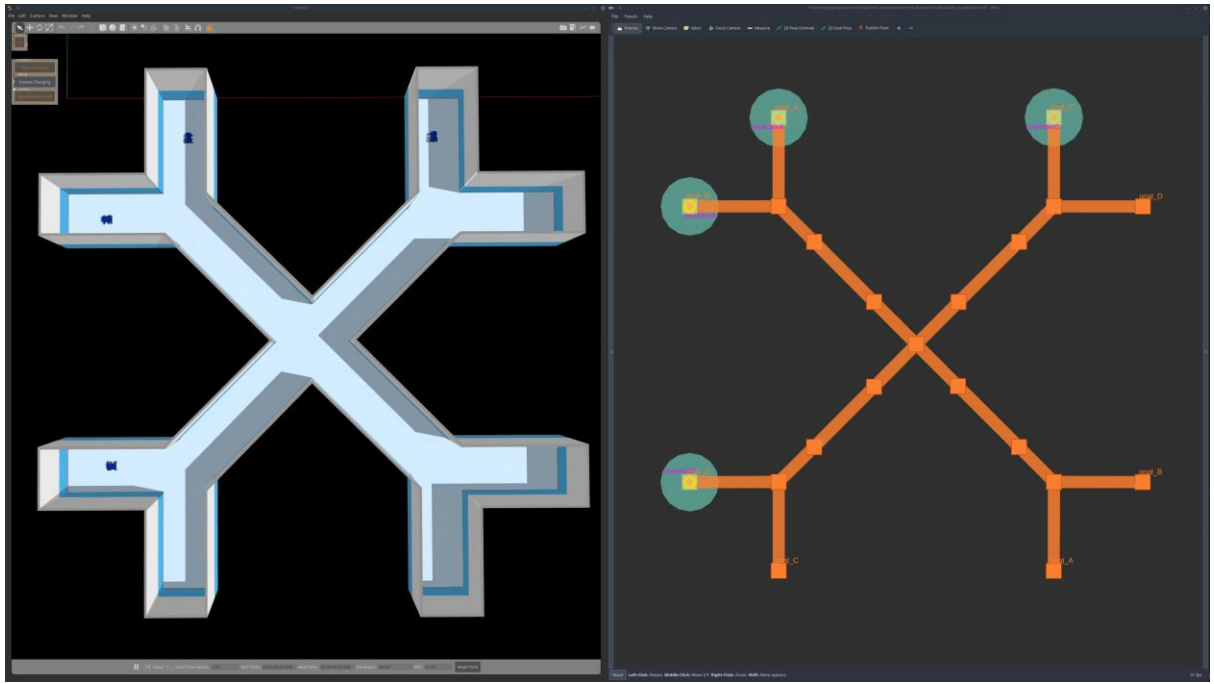
Future work
Native Integration: Negotiation takes place on board the robot and takes advantage of the robot's perception

Reroute robots in conflict



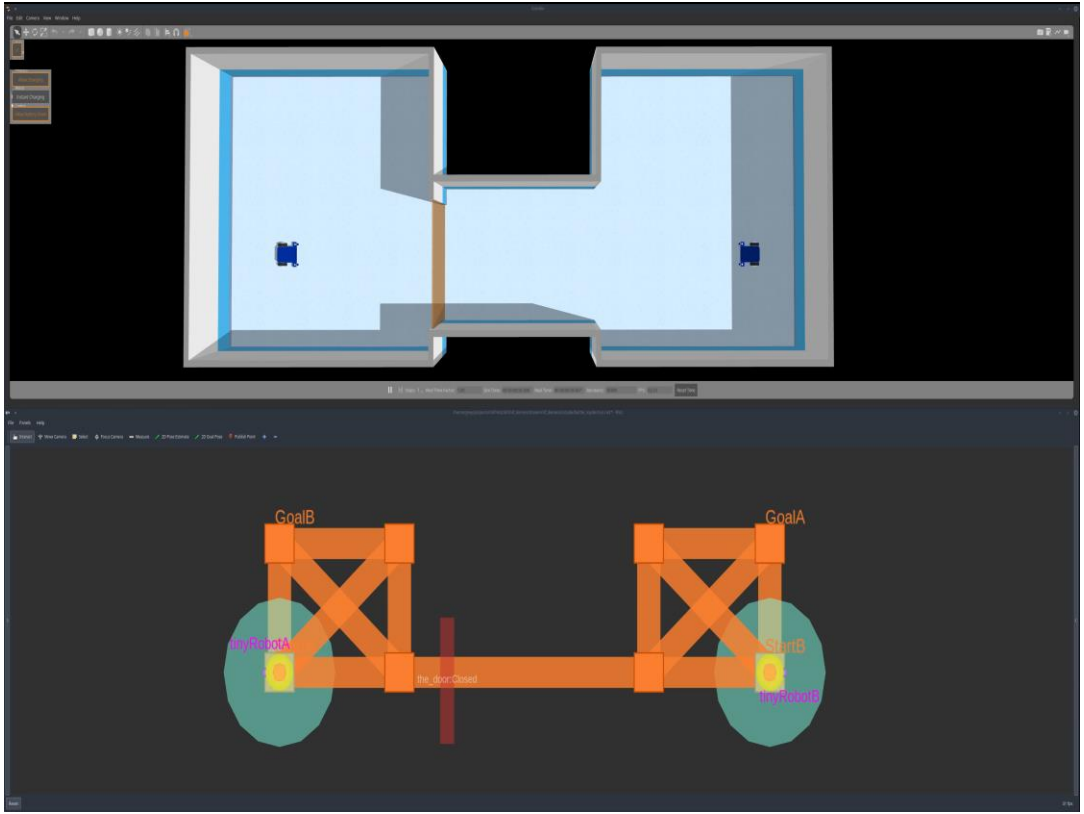
If following their shortest paths would result in a conflict, robots can be rerouted for an **overall** optimal traffic flow.

Respect traffic dependencies



When rerouting is not possible, the robots will be instructed to wait in place as long as needed to prevent a conflict from happening.

Share infrastructure



Robots will take turns using infrastructure such as doors and lifts.

Any robot integrated with Open-RMF can use any door integrated with Open-RMF.

Unified task descriptions

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"
                  }
              }
          }
      ]
  }
}
```

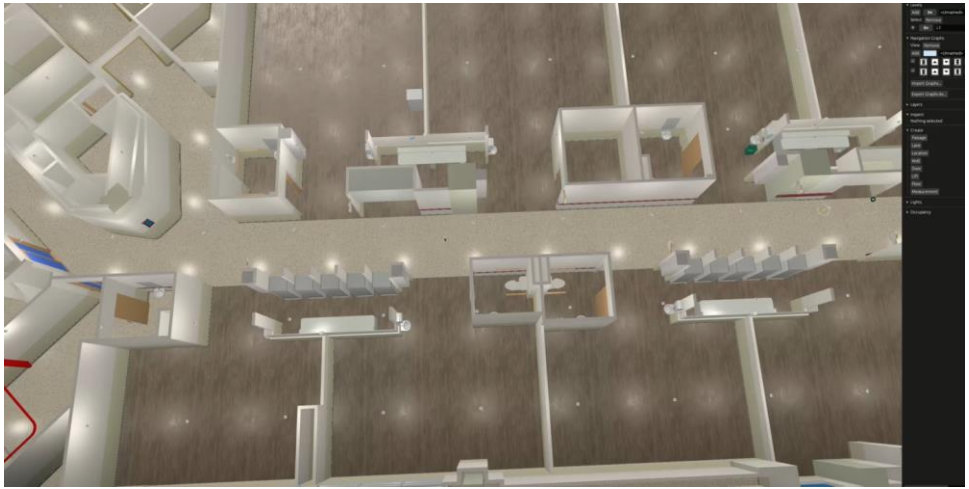
Task Auctioning

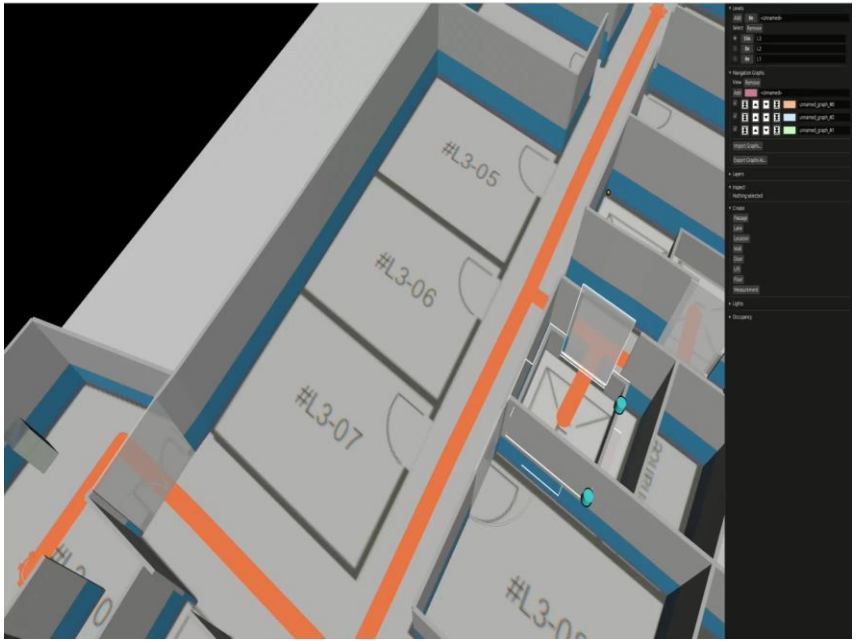
When robots from multiple fleets can perform the same task request, their fleet adapters will bid to be assigned the task.

The dispatch node will assign the task to the fleet adapter that promises the lowest overall “cost” (e.g. how long it will take to complete all the assigned tasks).

World building tools 🛠

Create “digital twins” of your site
Design the robot traffic flow
🚧 Run test scenarios to see how well the layout works





Questions?