ICRA2024
YOKOHAMA | JAPAN

APF-MRPPO: Navigating Graph-Based Maps with APF and Multi-Reward Reinforcement Learning

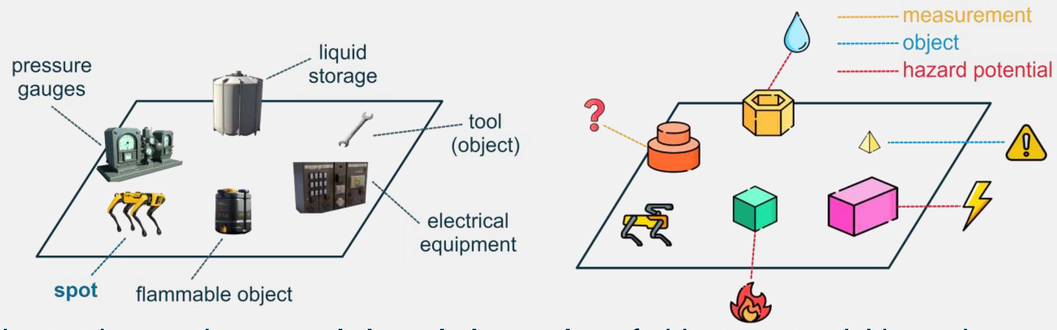
O'HARA Christopher (cohara@g.ecc.u-tokyo.ac.jp); YAIRI Takehisa (yairi@g.ecc.u-tokyo.ac.jp)
Department of Aeronautics & Astronautics, Graduate School of Engineering, University of Tokyo

Background: Mobile Robot Navigation

Robots have been increasingly deployed in **rescue scenarios** and investigating **damaged structures**, such as buildings affected by **natural disasters**. Equipped with advanced mobility and sensing capabilities, these robots can navigate challenging terrains, **reducing human exposure** to potentially **life-threatening hazards**.

We utilize intelligent robots to enhance human safety, by autonomously exploring and assessing environments too risky for human intervention.

- Boston Dynamics' Spot is a **cutting-edge** quadruped robot suitable for hazardous environments
- Effectiveness is enhanced by an **intelligent graph-based navigation** system using cameras, depth cameras, and laser scanners
- Create **detailed environmental maps**, enabling more precise navigation and **hazard avoidance**
- AI-driven robotics offers **versatile applications**, from **disaster response** to routine **industrial inspections**
  - ◇ e.g., chemical plant maintenance, **nuclear reactor decommissioning**
- Graph-based waypoints are used in robots to enhance navigation and reduce data storage complexity
  - ◇ Models balance the importance of **states and relationships**

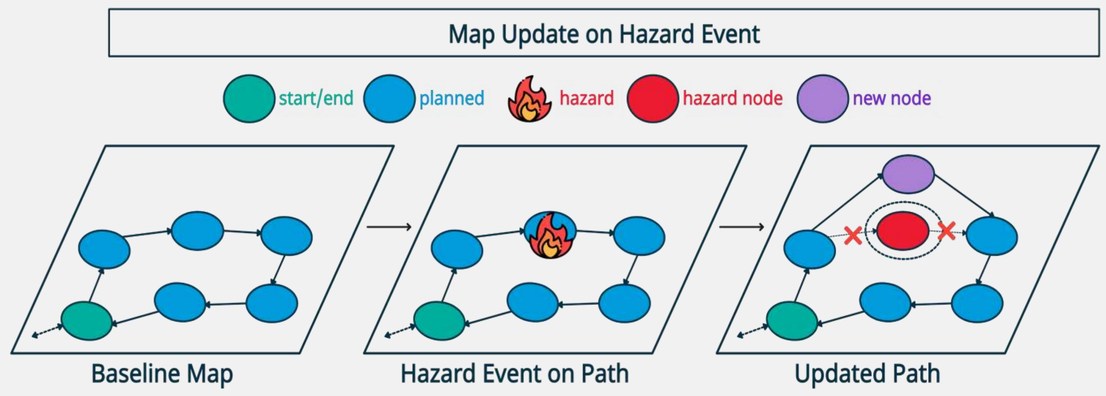


We start by creating a **graph-based abstraction** of objects, potential hazards, measurable devices, and potential obstacles.

Problem: Graph-based Hazard Avoidance

The presence of **known and unknown** obstacles in **complex environments** like nuclear reactors, chemical warehouses, or earthquake-damaged buildings calls for a solution that can **adapt in real time.** Traditional methods may encounter difficulties with unforeseen changes, trapped navigation scenarios, or burdensome **computational complexity**, particularly when the immediate reaction to newly detected hazards is **essential.**

Some challenges:
- ❑ Optimizing **hazard avoidance distance** to preserve a viable path
- ❑ Adapting the path on-the-fly without causing **instability** in navigation
- ❑ Ensuring that the new path does not **lead** to other **unforeseen hazards**
- ❑ Handling **ambiguities** in hazard recognition (e.g., shadows, reflections)
- ❑ Integration of **local** hazard avoidance with **global** path planning objectives
- ❑ Managing the **consistency** of the global map with localized hazard updates



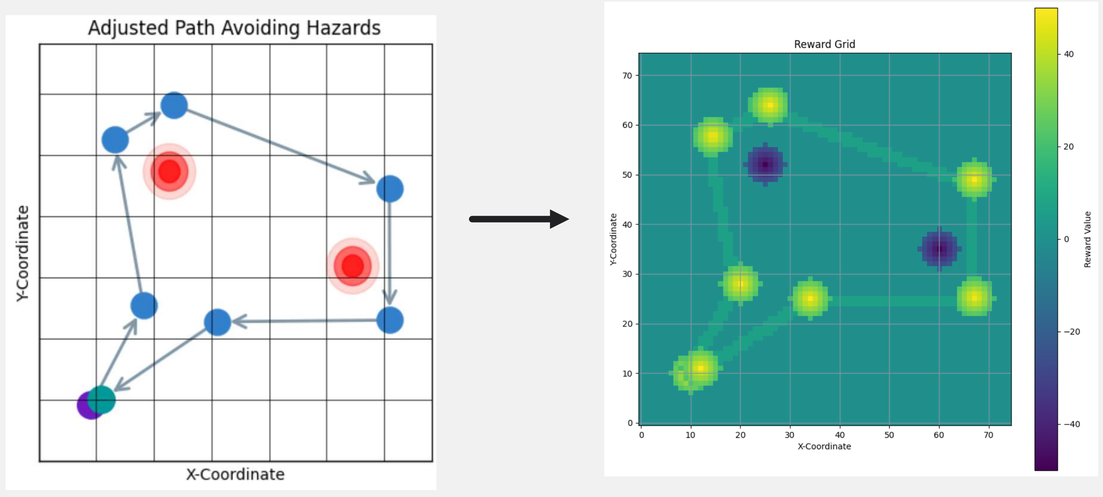
To address these challenges, we have chosen a **graph-based** approach that Boston Dynamic's Spot robot can use. Without compromising safety, the need for speed and efficiency leads to **strategies combining global path planning** with **local obstacle avoidance** within a graph-based framework. This amalgamation of planning, both **proactive and reactive**, must be skillfully balanced to ensure optimal navigation.

We utilize Unity and ROS to send commands to a virtual Spot robot. The API functions are the same as the real robot (deployable).

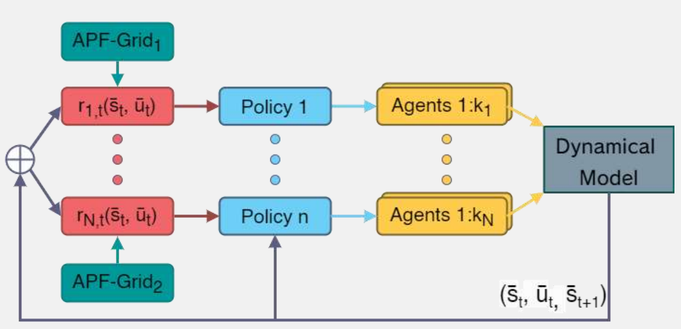
Based on: https://github.com/bdaiinstitute/spot_ros2 and https://github.com/Unity-Technologies/Unity-Robotics-Hub

Proposed Solution: RL with APF

First, we utilize geometric rules to create reference maps for obstacle avoidance. This output is used for training the Proximal Policy Optimization variant. Our APF-MRPPO algorithm converts **waypoints** into **attractive** forces and **hazards** into **repulsive** forces, while **inheriting** these characteristics into an **environmental reward map**. The **robot's** rewards are based on **success criteria** for navigating to a **new waypoint or goal.**



Dynamic environments with fixed paths require a flexible algorithm to adapt in real-time to avoid obstacles and ensure safe traversal.



Algorithm 1 APF-MRPPO for Dynamic Graph-Based Path Planning

```
robot_pos ← start_position
goal_pos ← goal_position
obstacles ← array of obstacles positions
waypoints ← array of waypoints positions
next_waypoint_index ← 0
path ← [robot_pos]
environmental_reward_map ← initialize_grid()
for cell in environmental_reward_map do
    cell.value ← calculate_initial_reward(cell.position, goal_pos, obstacles, waypoints)
end for
while not (robot_pos, goal_pos) do
    net_force ← (0, 0)
    if next_waypoint_index < len(waypoints) then
        net_force += calculate_attractive_force(waypoints[next_waypoint_index], robot_pos)
    end if
    for i, waypoint in enumerate(waypoints) do
        if i ≠ next_waypoint_index then
            net_force += calculate_repulsive_force(waypoint, robot_pos, 5)
        end if
    end for
    for obstacle in obstacles do
        net_force += calculate_repulsive_force(obstacle, robot_pos, 5)
    end for
    update_environmental_reward_map (environmental_reward_map,
        robot_pos, goal_pos, obstacles, waypoints)
    robot_pos += net_force/norm((net_force))
    path.append(robot_pos.copy())
    if norm((waypoints[next_waypoint_index] − robot_pos) < 1) then
        next_waypoint_index += 1
    end if
end while
```

Current Results & Future Work

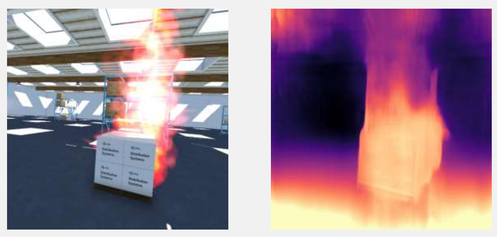
We have tested the APF-MRPPO algorithms in a **simulated** environment, demonstrating its **avoidance** capabilities. Now, we are in the process of **benchmarking** compared to other RL-based algorithms.

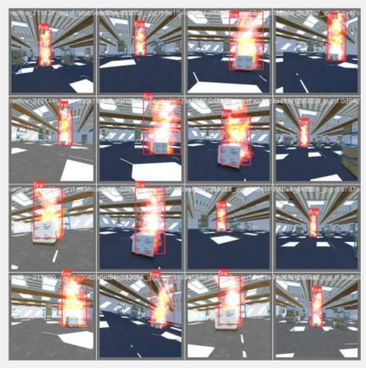
Collaborative work will extend these tests to deploy the algorithm on the Spot robot, translating the simulated successes into real-world applications

Currently:

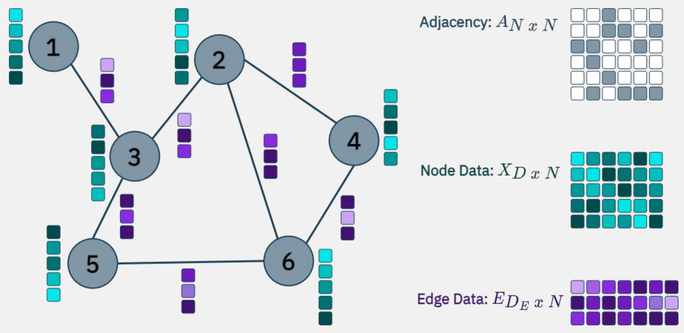
- ❑ Finds solutions 100% with obstacles near 20% of waypoints
- ❑ 85% success with **known** obstacles near 40% of waypoints
- ❑ 98% success with 10% **random** hazards, similar in size to the robot
- ❑ 83% success with hazards **3x larger** than the robot
- ❑ Tests: hazards **1 to 5 times** the robot's size, covering **~40%** of the map

Random Hazard Event Modeling
(Stereo Depth, Custom YOLO)





Future work:



Future work will include turning the **reward** functions to **improve** results. Additionally, we plan to utilize **more complex potential** fields. Finally, we will be integrating **Graph Neural Networks (GNNs)** to enable **dynamic, context-aware** path planning, improving **adaptability** and **robustness** in **complex,** ever-changing environments.

References:

1. R. Szczepanski, "Safe Artificial Potential Field - Novel Local Path Planning Algorithm Maintaining Safe Distance From Obstacles," in *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4823-4830, Aug. 2023, doi: 10.1109/LRA.2023.3290819
2. C. J. Sullivan and N. Bosanac, "Using Reinforcement Learning to Design a Low-Thrust Approach into a Periodic Orbit in a Multi-Body System," in Proc. of the AIAA SciTech 2020 Forum, Session: Artificial Intelligence in Space Flight Mechanics II, Jan. 2020, doi: 10.2514/6.2020-1914

WINGS CFS 「未来社会協創」国際卓越大学院
World-leading Innovative Graduate Study Program
Co-designing Future Society
SPRING GX