Analyzing Strategies for Safe Multi-robot Motion Planning Algorithms

Kunj Dedhia and Shaunak D. Bopardikar (Mentor)

Introduction

In order to deploy technologies in tomorrow's smart cities, systems, especially, autonomous vehicles' dependence on intra-vehicle sensors and inter-vehicle communication leaves the system vulnerable to physical and cyber-attacks to manipulate sensor data to take control of it.

- The world would spend $6 trillion dollars by 2021 on cybersecurity to protect data breaches (Forbes Technology Council)
- There are over 130 large-scale, targeted breaches in the U.S. per year, and that number is growing by 27 percent per year (Accenture, CyberTech Europe)

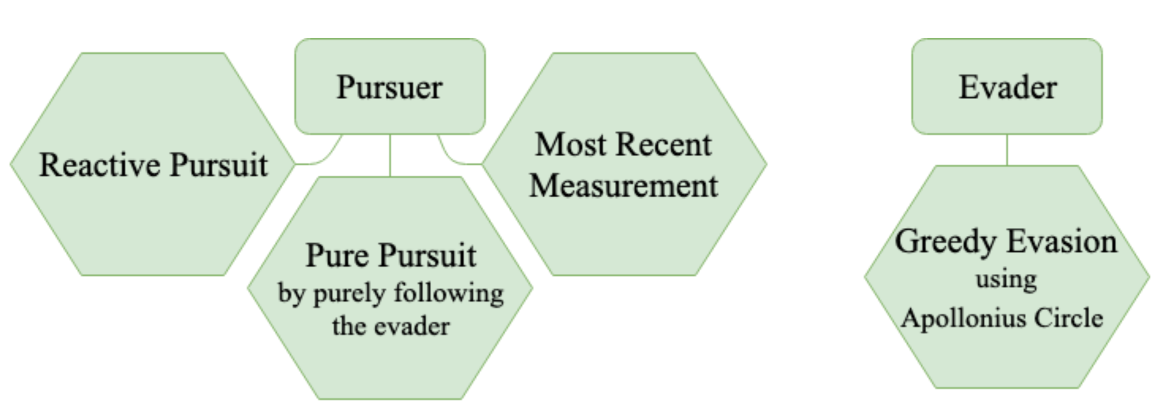
Background Research

- Contemporary research considers an attacker that aims to inject faulty data to the intra-vehicle sensors and risk to potential dangers by analyzing and estimating processes for a novel deep reinforcement learning algorithm [1]
- Another study studies GPS spoofing attacks and its vulnerability. Attacks could falsify the map, manipulate information about localization and block sensors to update information in real-time. Results confirm that the threats can be reduced to 0.01% based on the combination of multiple techniques using Bayesian Network [2]

Problem Statement

This work studies a simple pursuit problem in which a pursuer robot seeks to capture an identical evader robot in an adversarial environment that affects the pursuer's perception of the evader's position. Specifically, we assume that the pursuer can detect the evader only with a prescribed probability.

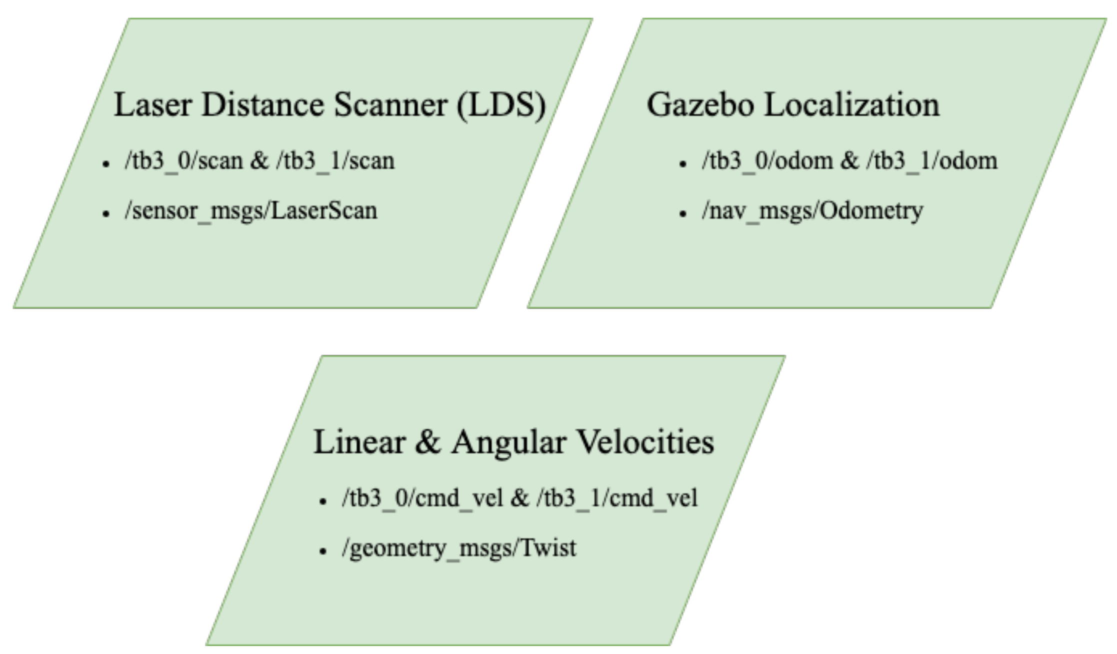
Strategies



Technologies & Frameworks

Robot Operating System (ROS)

ROS is a framework and evolving open-source software library and tools that is a robotics middleware. At the lowest level, ROS offers a message passing interface that is anonymous and asynchronous. It is often coupled with Gazebo, a multi-robot simulator with dynamic and kinematic physics and OpenCV, a computer vision library for seamless integration [3]. Below are the ROS Topics and Msgs used in the study -



TurtleBot3

TurtleBot3 is ROS-based mobile robot with its core technology revolving around SLAM (simultaneous localization and mapping), navigation and manipulation. It can run algorithms to build a map and navigate around . TurtleBots can be controlled remotely from a laptop, joypad or Android-based smart phone.

Gazebo

Gazebo is a 3D dynamic simulator with the ability to accurately and efficiently simulate populations of robots in complex indoor and outdoor environments. It offers physics simulation at a much higher degree of fidelity, a suite of sensors, information about robot localization and ROS Topics to access sensor data from the robots.

Experimental Set-Up

The setup in the study considers a Gazebo simulations of a TurtleBot3 WafflePi evader and a pursuer in bounded and unbounded regions. The robots are limited to an intermittent 2-dimensional laser distance sensor (LDS) as the only functional sources of input data streams with which the robot must compute information and make decisions. It is imperative to note all variables that may affect the result –

- Dimensions/shape of the region (if bounded)
- Linear and angular velocities of the evader and pursuer
- Initial positions of the evader and pursuer

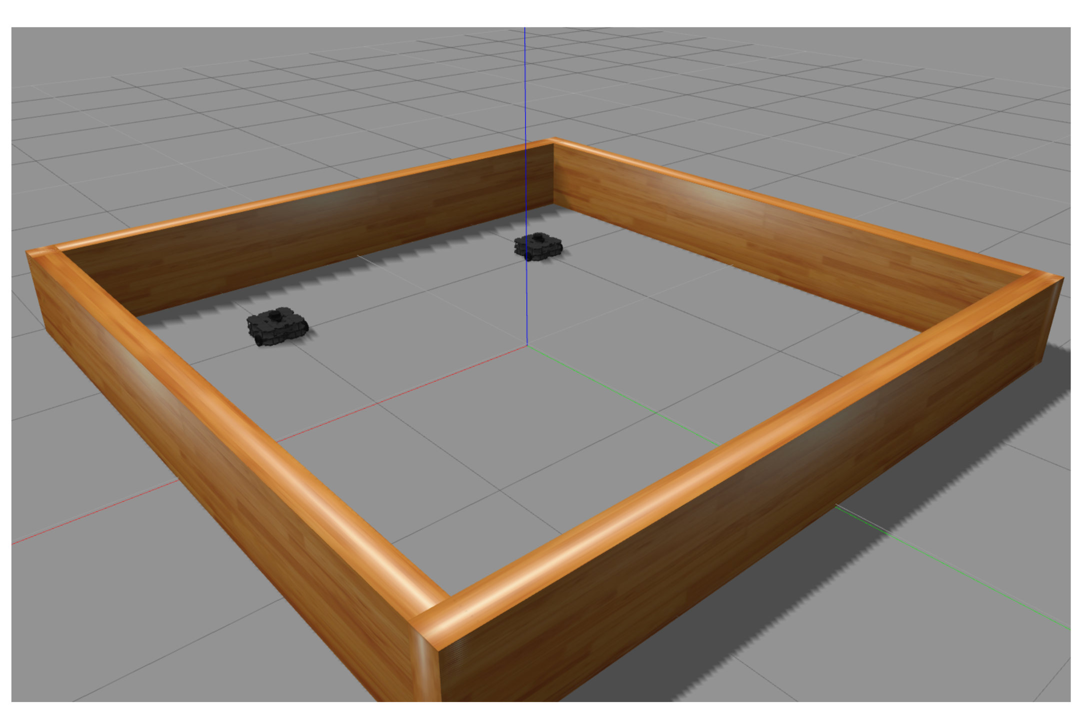


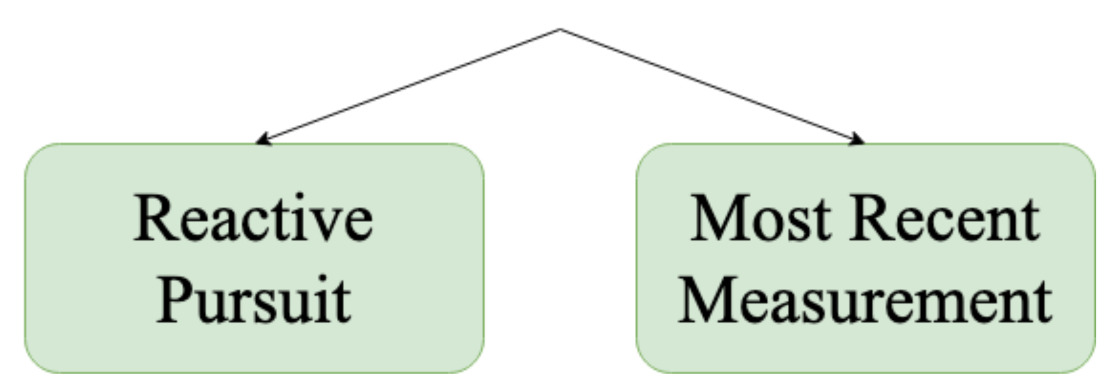
Fig. Screenshot of the experimental set-up from Gazebo for a bounded region

Pursuer Strategies

The intermittent data stream or the cyber/physical attacks on the 2D LDS are experimentally simulated in the pursuer python script using a binary probabilistic sleep function.

```
from numpy.random import choice
if (choice([0,1], 1, [p, 1-p]) == 0):
    #sleep (evader coordinates do not update)
```

The pursuer obtains access to the evader's coordinates only when choice() returns 1. Otherwise, the pursuer assumes that the evader is located at the last obtained coordinates. In such a scenario, the pursuer could use one of the two strategies described as follows -



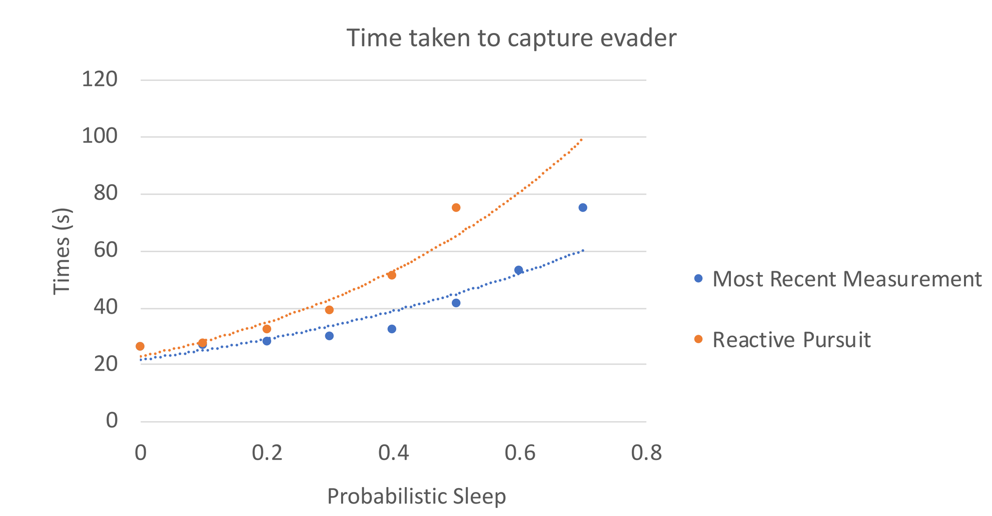
The pursuer moves one unit step only at every instance it receives a sensor-input

The pursuer continues to move using the last sensor-input, rather than remaining static

Increasing values of 'p' are tested to deduce a critical probability, at which the pursuer lacks sufficient information to capture the evader.

Results

Seven trials were conducted in a 4 units x 4 units bounded region pursuit with the pursuer and evader at equal velocities



The graphs can be described as quadratic functions. The Most Recent Measurement strategy performs better with lower time captures, even with high probabilistic sleep values and equal velocities.

Discussions

- TurtleBot3s are not a point particle. They involve a spatial extension. Given their linear and angular velocity, the path will be spiral and hence, increase time complexities
- Networking issues while replicating the simulation with TurtleBot3s

Future Scope

- Evaluate a situation with multiple pursuers against one evader such that the velocity of evader is higher than the pursuers (which can be coupled with faulty sensors)
- Approach more realistic conditions by introducing obstacles and restricting paths that are available to the TurtleBots to maneuver.
- Application of an aerial surveillance system monitoring the area and feeding real-time information to autonomous systems.
- Develop a moving average algorithm that uses information from inter-vehicle communication, but makes decision using the principles of Game Theory (Prisoner's Dilemma)

References

[1] Ferdowsi, et al. "Robust Deep Reinforcement Learning for Security and Safety in Autonomous Vehicle Systems." *ArXiv.org, 2018*

[2] Ali. "Risk Assessment of Autonomous Vehicles Using Bayesian Defence Graphs." *ArXiv.org, 2019*

[3] "ROBOTIS e." *Manual*, emanual.robotis.com/docs/en/platform/turtlebot3/overview/.