LSTM based Dynamic Obstacle Avoidance for reaching Goal

By: Srihari Subramanian, Manan Patel, Vivek Mallampati

MOTIVATION

Are there any existing approaches?

- Global Planners (ex: potential field)
- Local Planners (ex: dynamic window approach)
- Neural Network based approach

Are there any existing approaches?

- Global Planners (ex: potential field) assume complete knowledge of the environment – thus not generalizable
- Local Planners (ex: dynamic window approach) optimize trajectory over a cost function – need good heuristics and measurements
- Neural Network based approach needs an exhaustive set of learning data

Are there any existing approaches?

- Global Planners (ex: potential field) assume complete knowledge of the environment – thus not generalizable
- Local Planners (ex: dynamic window approach) optimize trajectory over a cost function – need good heuristics and measurements
- Neural Network based approach needs an exhaustive set of learning data

DEEP
REINFORCEMENT
LEARNING

Prior Work

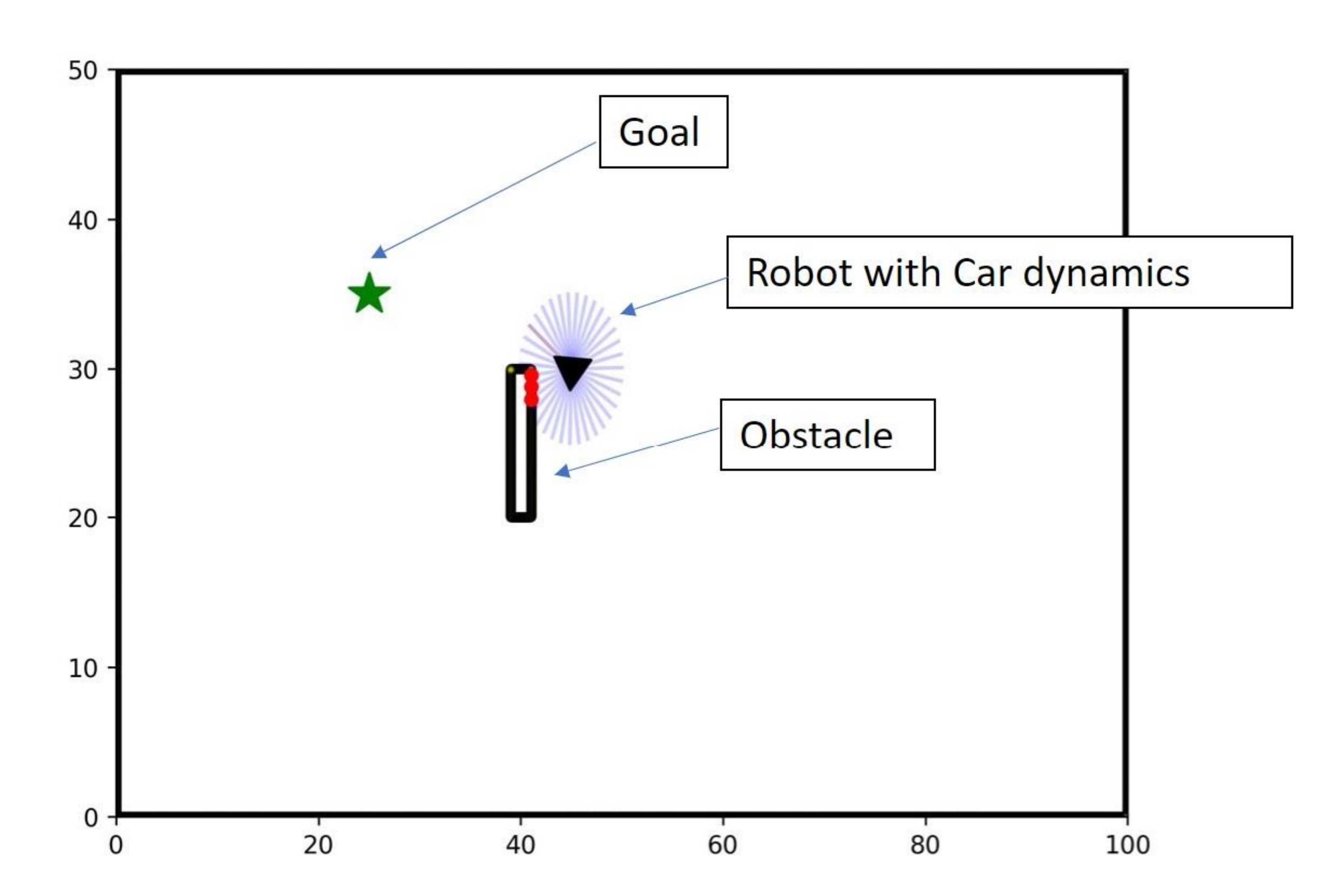
- Motion using differential drive robots
- Networks send multiple time steps of data into the network to train
- Have been implemented using sparse rewards
- Some had encoded planners

[1] Jaewan Choi. Geonhee Lee. Chibum Lee. *Reinforcement learning-based dynamic obstacle avoidance and Integration of Path Planning*.

Proposed Method

- Use **LSTM** to encode time-series information – these networks can implicitly encode critical information about the past into a hidden state and a cell state
- Use **Dense reward** functions – allows for faster training than sparse rewards as the agent is provided an incentive for all the states
- Agent motion follows and obeys the **Dynamics of a car**

Problem Setup



Architecture

- Network: LSTM with one layer
- Input State: <lidar readings, distance to goal, angle to goal, robot position>
- Output (Action): Softmax - the probability of turning left or right
- Actual action taken: Weighted steer function output

Reward Function

- Reward calculated for every (s, a, s') 3-tuple.
  - If an action is taken that will reduce the angle to the goal, the agent will be positively rewarded, if not the agent will be penalized.
  - If the summation of the difference between LIDAR readings in the current and future state is positive, the agent is rewarded
    - i.e., the agent is rewarded for moving away from an obstacle
- The effects of these rewards are tuned using a scaling factor.

Hyperparameters

- Hyperparameters were tuned specifically for each scenario (no obstacles, static obstacles, dynamic obstacles)
  - Adjusted learning rate and discount factor for neural network update
  - maximum LIDAR distance
  - rollout limit and iterations
  - maximum acceleration and angular velocity
- Scaling of reward function
  - Need to balance the obstacle distance and robot orientation factors

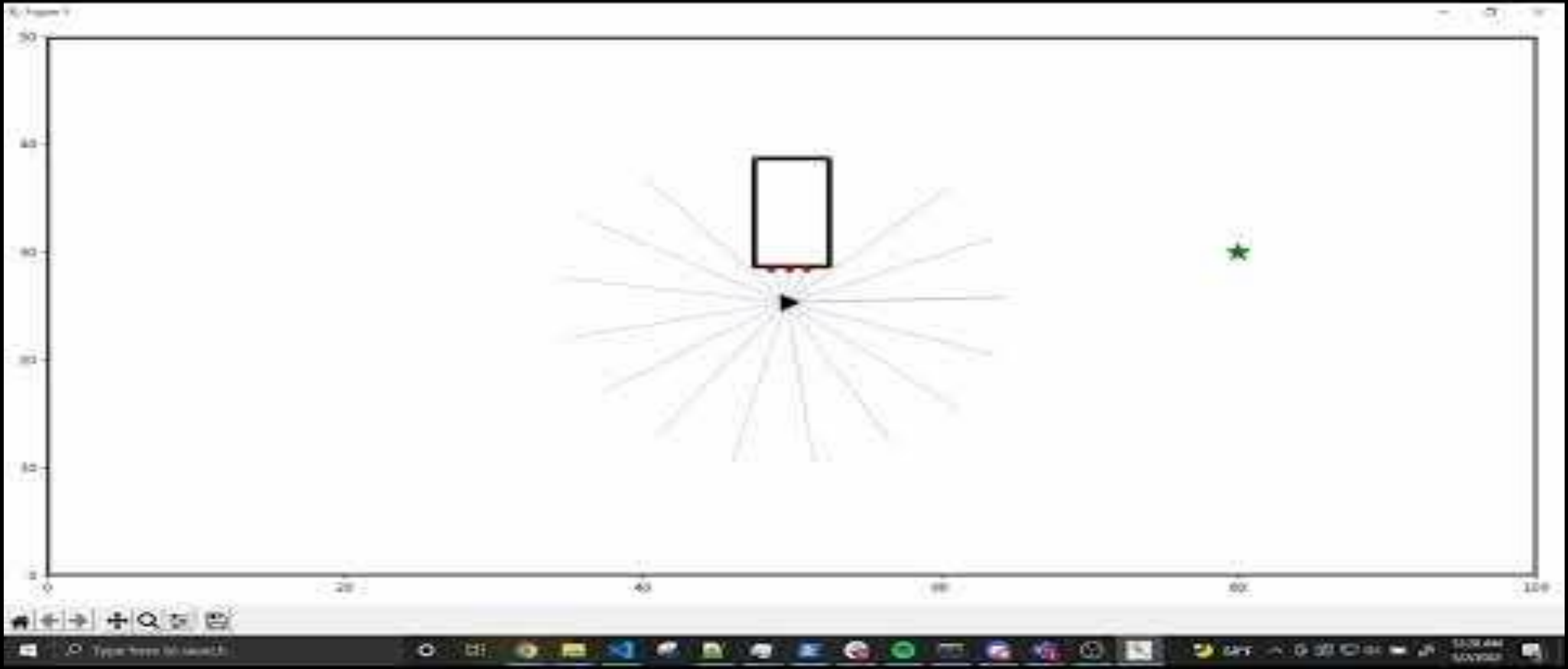
Results: No Obstacles

- Able to reach goal with static robot starting and goal positions
- Generalization for many positions, but not all
  - Lower accuracy with variable starting angles and positions
  - Attempted training on four different configurations; model began to overfit
- Very sensitive to changes in hyperparameters
  - Changes to reward function scaling also caused significant fluctuations in results

Results: Obstacles

- Able to avoid static obstacles in most cases
  - Model prioritizes avoiding obstacle to reaching goal
- Reaching the goal with moving obstacles only in very specific cases
- Model overfits to specific simulation
  - Very low generalization
  - Potentially need more iterations or training data
- Reward function may need work

Video



https://youtu.be/Tj5GInNJTI0

LSTM based Dynamic Obstacle Avoidance for reaching Goal

LSTM based Dynamic Obstacle Avoidance ~~for reaching Goal~~

Programmers: You can't just rerun your program without changing it and expect it to work

Reinforcement Learning Practitioners:

Real life
Reinforcement learning