Follow-Ahead Human-Robot Navigation via Monte Carlo Tree Search and Deep Reinforcement Learning

Archita Srivastava, Ankush Singh, Gemmin Sugiura, Jonathan Ung
Simon Fraser University

Abstract

We replicate a methodology for robotic follow-ahead applications that addresses obstacle and occlusion avoidance, originally proposed by Leisiazar et al. [1]. A high-level decision-making algorithm generates short-term navigational goals by integrating Monte Carlo Tree Search (MCTS) with Deep Reinforcement Learning (DRL). We evaluate the MCTS-DRL approach in a lightweight ROS2 simulation, validating directional prediction accuracy across multiple trajectory patterns.

Applications

Follow-ahead robots navigate in front of a target person while maintaining a safe distance and avoiding obstacles in real time. Real-world use cases include:

- Autonomous shopping carts
- Filming robots that stay ahead of people
- Security robots that should stay in front of people

Methods

Pure DRL cannot generalize to unseen obstacle configurations since it would overfit to whatever environments it was trained on, making retraining necessary for every new setting. The solution is to train DRL only on the follow-ahead behavior in an obstacle-free setting, where the Q-function learns to estimate the expected return for each action given the current relative pose of the robot and person.

$$Q^*(o,a) = \mathbb{E}[r - \gamma \max_{a'} Q^*(o',a')]$$

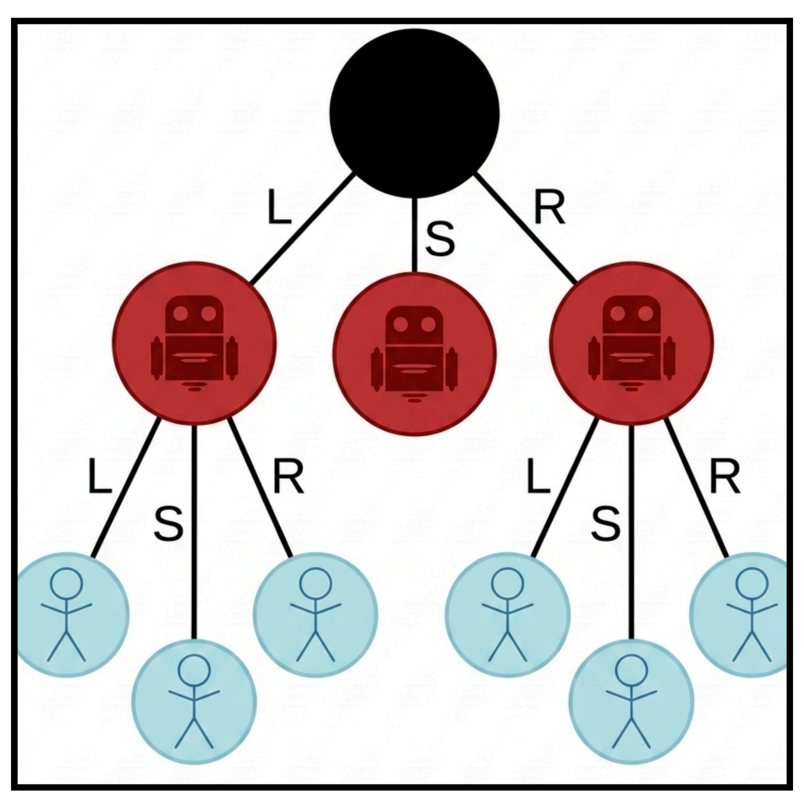
The core insight is substituting this learned Q-function directly into MCTS as the node evaluator replacing random rollouts with informed value estimates. This is where the two methods fuse: MCTS gets consistent node evaluations from DRL, while obstacle and occlusion avoidance is handled separately by pruning any candidate pose that violates the occupancy map, no retraining required.

$$\text{UCB} = \frac{w}{n^c} + c\sqrt{\frac{\log n^p}{n^c}}$$

At each time step the tree expands over a 3-second horizon, selecting nodes by UCB to balance exploration and exploitation. The leaf node with the highest score at c=0 becomes the robot's next navigational goal, passed to the ROS navigation stack for execution.

Algorithm 1: MCTS-DRL Approach

Data: robot pose $= (x_r, y_r, \theta_r)^{t_0}$
Data: $(x_h, y_h, \theta_h)^{t_0}, \ldots, (x_h, y_h, \theta_h)^{t_3}$
Data: occupancy map
Result: navigational goal point
parent node $= (x_r^{t_0}, y_r^{t_0}, \theta_r^{t_0}, x_h^{t_0}, y_h^{t_0}, \theta_h^{t_0})$;
for *num of expansion* **do**
  while *parent node is not fully expanded* **do**
    child ← simulate using an action;
    if *no collision* **then**
      if *no occlusion* **then**
        $R \leftarrow Q(o, a_i)$;
        child node state ← simulate using the action;
      else
        value ← −1;
      parent value ← parent value + value;
    else
      delete the child node;
  parent node ← leaf node with highest UCB;
goal point ← leaf node with highest UCB ($c = 0$);



Future Work

Planned deployment on Quanser QBot platform with Vicon motion capture for real-world validation. Further work includes evaluating the combined MCTS-DRL pipeline against standalone baselines and testing in obstacle-present environments.

Experimental Setup

Evaluated in a ROS2 Humble simulation using fake_vicon and fake_odom publishers with rviz2 visualization. The MCTS planner runs over a 3-second horizon with UCB-based node selection. Integration tested across five trajectory patterns: circle, stationary, square, oscillation, and zigzag.

Results

Directional prediction accuracy of the MCTS planner across straight-line and turning trajectories: Straight 96.1%, Left 75.8%, Right 77.4%, Overall 90.1%. The planner achieves high accuracy on straight trajectories. Reduced accuracy on turns suggests the 3-second planning horizon may be insufficient to capture early directional changes.

Website!

References

[1] Leisiazar et al., IROS 2023.
[2] Nikdel et al., ICRA 2021.
[3] van Hasselt et al., AAAI 2016.