Adapting to Frequent Human Direction Changes in Autonomous Frontal Following Robots

Sahar Leisiazar[1], Seyed Roozbeh Razavi Rohani[2], Edward J. Park[1], Angelica Lim[2], and Mo Chen[2]

Abstract*— This paper addresses the challenge of robot follow ahead applications where the human behavior is highly variable. We propose a novel approach that does not rely on single human trajectory prediction but instead considers multiple potential future positions of the human, along with their associated probabilities, in the robot's decision-making process. We trained an LSTM-based model to generate a probability distribution over the human's future actions. These probabilities, along with different potential actions and future positions, are integrated into the tree expansion of Monte Carlo Tree Search (MCTS). Additionally, a trained Reinforcement Learning (RL) model is used to evaluate the nodes within the tree. By incorporating the likelihood of each possible human action and using the RL model to assess the value of the different trajectories, our approach enables the robot to effectively balance between focusing on the most probable future trajectory and considering all potential trajectories. This methodology enhances the robot's ability to adapt to frequent and unpredictable changes in human direction, improving its navigation and ability to navigate in front of the person. The codes and supplementary videos of the experiments are available on the project page, which can be accessed through this *https://saharleisiazar.github.io/follow-ahead-adoption/

I. INTRODUCTION

The development of companion robots has enabled them to effectively follow and assist humans in various tasks. These robots are designed to follow a person autonomously, providing assistance in various contexts, such as carrying load and assisting with mobility. Examples like autonomous suitcases and shopping carts demonstrate the practical applications and benefits the front-following robots. These robots must navigate in complex environments, avoiding obstacles and adapting to changes while maintaining proximity to the user. For the elderly, these robots offer another application by ensuring their safety and promoting their independence, enabling them to move confidently within their homes or in public spaces while receiving assistance when needed.

Various approaches have been proposed for human-following robotic applications, aiming to detect a target person, compute their position, and plan a path for robot navigation. These applications utilize different sensors, including LiDAR [1] and cameras [2], as well as various sensor fusion techniques applied to both Unmanned Ground Vehicles (UGVs) [3], [4] and Unmanned Aerial Vehicles (UAVs) [5]. Another challenge in solving the human-following problem is computing the relative distance of the person with respect to the robot [3] and predicting the person's future trajectory. Nikdel et al. [6] proposed a model for human motion and used an Extended Kalman Filter (EKF) to estimate the velocity and direction of the person. To predict the future trajectory of the target person, Mahdavian et al. [7] and Nikdel et al. [8] proposed a non-autoregressive transformer and a Reinforcement Learning (RL) model, respectively. Additionally, various path planning approaches have also been developed to navigate the robot in complex environments. Khawaja et al. [9] and Peng et al. [10] employed a Model Predictive Control (MPC) controller to navigate the robot by utilizing predictions of the person's future trajectory.

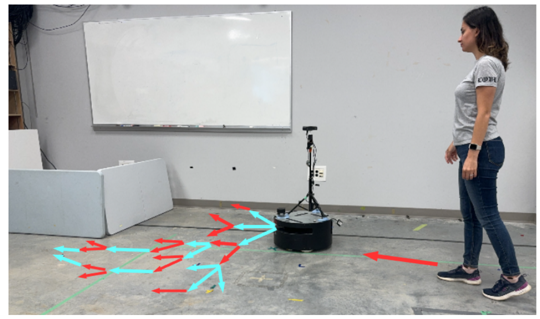


Fig. 1: Real-world experiment showcasing the robot's ability to follow a human from the front while avoiding obstacles. The figure illustrates an example of the tree expansion process, with blue and red arrows representing the potential moves for the robot and human, respectively. The robot expands the tree and selects nodes that allow it to maintain a position in front of the human, while eliminating branches that could lead to collisions with obstacles.

In our research, we have utilized Monte Carlo Tree Search (MCTS) to address the robot navigation problem in a front-following setting. MCTS has been widely applied to solve complex problems in various domains, including video games [11] and board games [12], [13]. In the field of robotics, MCTS has been utilized in various studies to address the challenge of robot path planning. For instance, Dam et al. [14] demonstrated the convergence of MCTS in path generation through analytical analysis. Furthermore, MCTS has been integrated with neural networks [15], [16] and RL [17], [18] to enhance the accuracy and performance of path planning algorithms.

Manuscript received: September 4, 2024; Revised November 24, 2024; Accepted January 19, 2025.

This paper was recommended for publication by Editor Gentiane Venture upon evaluation of the Associate Editor and Reviewers' comments.

[1]School of Mechatronics Systems Engineering, Simon Fraser University, Canada {sleisiaz,ed_park}@sfu.ca

[2]School of Computing Sciences, Simon Fraser University, Canada {srr8, angelica, mochen}@sfu.ca

Digital Object Identifier (DOI): see top of this page.

In all previous works where MCTS has been applied to various decision-making processes, either the states of a single agent are solely critical for making decisions [14] or MCTS involves two agents, with the states and actions of both considered during tree expansion. In these cases, the second agent either opposes [19] or cooperates [20] with the first agent. To the best of our knowledge, our study is the first to introduce a second agent that is neither fully cooperative nor fully adversarial towards the main agent. Here, the main agent (robot) takes into account all possible actions of the second agent (human), each assigned a specific probability.

In this paper, we build upon our previous work [18]. In that study, the robot was designed to detect a target person, predict the human's trajectory and future position, navigate, and stay in front of the person. We integrated MCTS with RL to simulate and evaluate promising moves for the robot and selecting the optimal one. However, since the human behavior is variable and may change repeatedly, we did not previously account for different future positions for the human in [18]. Instead, we now considered different potential future positions for the human, along with their probabilities, in the robot's decision-making process. We trained an LSTM-based model to generate a probability distribution over the human's future actions. The probabilities are integrated with the Upper Confidence Bound (UCB) equation in MCTS. By incorporating the likelihood of each possible human action, the robot can balance between considering a single future trajectory and considering all possible trajectories equally. Figure. 1 illustrates a real-world experiment demonstrating the proposed method from this paper. The robot observes the human's position, expands a decision tree based on potential future positions for both the human and the robot, and determines the optimal action to navigate in front of the human while avoiding obstacles.

As in the previous work, the robot avoids moves that could lead to a collision with the human and uses a map of the environment to steer clear of surrounding obstacles.

The main contributions of the paper are as follows:

- To the best of our knowledge, this is the first work in the human follow-ahead application that accounts for the human's natural tendency to change direction suddenly and frequently;
- As far as we are aware, this is the first work to implement MCTS in a scenario where two agents are neither fully adversarial nor fully cooperative, with the second agent remaining neutral to the main agent;
- Compared to our previous work, the decision-making process in this approach relies solely on the current positions of the human and robot, rather than on the human's predicted future position. This enables the robot to quickly adapt to changes in the human's direction;
- The UCB equation in MCTS has been modified to incorporate the probability of the human's future actions;
- The camera used to detect the person is mounted on the robot, providing a first-person perspective instead of a third-person view. A motor, controlled by a PID controller, rotates the camera to maintain a clear view of the person.

II. Problem Setup

In this paper, we present an algorithm designed for a mobile robot to follow a human while accounting for the dynamic nature of human movement, rather than assuming that the human has a fixed goal point. Given that humans frequently change direction, the algorithm considers a range of possible future actions for both the robot and the human. The robot is designed to evaluate the most promising and probable moves and future positions of both the human and itself a few time steps ahead, ultimately selecting the optimal action for the current time step.

The algorithm is structured into three key modules:

- Decision tree: A module that takes the poses of both the human and the robot as inputs. It expands the tree by considering various potential actions for both the robot and the human, ultimately generating the optimal action for the robot.
- State evaluation: A module that assigns a value to each node during tree expansion. Nodes with higher values indicate that the robot and human are in a more desirable relative position.
- Human future positioning probabilities: This module estimates the likelihood of different future human positions for each human node during tree expansion. This helps the algorithm to better anticipate and adapt to realistic human behavior in real-world scenarios.

A. System Modeling

Due to the nature of MCTS, which is designed for discrete settings, we considered six discrete actions for the robot: three angular velocities $\{-4, 0, 4\}$ rad/s and two linear velocities $\{0.7, 1.2\}$ m/s. For the human's action space, assuming constant linear velocity, we defined three angular velocities $\{-1.5, 0, 1.5\}$ rad/s. Given that the robot follows the human from the front, the robot's angular velocity should be greater than the human's because even small deviations in the human's direction require the robot to make larger turns to stay positioned in front of the human. Additionally, robot's linear velocity should be equal to or greater than the human's linear velocity to maintain the desired distance.

To simulate the next state of the robot and human based on their actions, we use the Dubins car dynamics:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} d\cos(\theta+\psi) \\ d\sin(\theta+\psi) \\ \psi \end{bmatrix}, \tag{1}$$

where (x, y, θ) represent the robot or human pose in the ground plane, ψ and d are the turning angle and traveled distance at each time-step, respectively.

We also utilized the receding control horizon approach in this work. This means the algorithm simulates and evaluates the future states of both the human and the robot over several time steps but only executes the optimal action for the next immediate time step. After each action, the algorithm updates with the new poses of the robot and human, then re-expands the decision tree.

The integration of the RL and LSTM models with MCTS introduces minimal computational delay, as their inference time is negligible compared to the time required for MCTS tree expansion. To ensure real-time decision-making, we implemented a stopping criterion by setting a fixed time limit of 0.15 seconds. This approach allows the robot to operate at a decision frequency of 5 Hz, corresponding to intervals of 0.2 seconds, which meets the requirements for real-time applications.

III. Methodology

In this paper, we propose a novel methodology comprising three integrated modules: RL, LSTM, and MCTS. This approach introduces a unique consideration of distinct action spaces for humans and robots, enabling the system to dynamically capture and adapt to sudden changes in human trajectories with reasonable probability. This integration builds on prior work by improving responsiveness and adaptability, addressing challenges in scenarios with frequent and unpredictable human direction changes. A detailed explanation of these modules is provided in the following sections.

A. Decision Tree

In this work, MCTS is used to determine the optimal actions for a robot following a person ahead. As a heuristic search algorithm, MCTS is effective for complex decision-making, incrementally building a search tree through selection, expansion, simulation, and backpropagation while balancing exploration and exploitation. Unlike traditional MCTS, which uses random sampling for node evaluation, this approach integrates a trained RL model for node evaluation and an LSTM-based model to predict human action probabilities. This combination enhances MCTS, enabling the robot to make more accurate and robust decisions.

The implementation of MCTS is based on the approach presented in [21] and has been modified to meet the specific requirements of our problem. The process of tree expansion utilized in this study is described by Alg. 1 and illustrated in Fig. 2. During the tree expansion, we consider two distinct layers: one for the robot and one for the human. Each pair of layers represents a single time step (0.2 seconds), during which the algorithm first evaluates the possible actions for the robot, simulates the resulting next states, and then incorporates the potential movements of the human at the same time step in the tree expansion process. In Fig. 2, robot nodes are depicted in blue, while human nodes are in red.

In the selection stage of the tree expansion, the process starts at the root node, representing the current poses of the human and the robot. From this point, the algorithm computes the Upper Confidence Bound (UCB) value for each child node using Eq. (2) and selects the child with the highest value, continuing this selection process until it reaches a leaf node. In this work, we modified the standard UCB approach [21] by incorporating the probability of selecting each node (P). The constant value (c) is set to 2 in this work. The value of each node is derived from the state evaluation module, while the probability is obtained from the human future positioning probabilities module. These components are detailed in sections III-C and III-B, respectively.

Algorithm 1: The Proposed Approach

Data: Robot current pose = $(x_r, y_r, \theta_r)^{t_0}$
Data: Human current pose = $(x_h, y_h, \theta_h)^{t_0}$
Data: RL model
Data: LSTM-fc model
Data: Occupancy map
Result: Best robot action

Root node $= (x_r, y_r, \theta_r, x_h, y_h, \theta_h)^{t_0}$;
for *0.2 second* **do**
  while *not leaf node* **do**
    for *child in children* **do**
      UCB ← V/n + c P $\sqrt{\frac{\ln n_p}{n}}$
    end
    Node ← children (max UCB)
  end
  New leaf node ← expansion(Node) ;
  if *if leaf node is safe* **then** V ← evaluation(leaf node) ;
  else Delete the node ;
  if *lead node is human node* **then** P ← Probability (leaf node) ;
  else P ← $1/6$;
  Back_propagation(V);
end
Best_action ← most_visited_child

$$\text{UCB} = P\left(\frac{V}{n} + c\sqrt{\frac{\ln n_p}{n}}\right) \tag{2}$$

During the expansion stage, all possible next states for each action are simulated. If the selected leaf node is a robot node (blue node), the algorithm uses the human's actions and, if it is a human node (red node), it uses the robot's actions. In the evaluation stage, newly added leaf nodes are first assessed for safety to ensure they do not result in collisions with obstacles or the human. If a leaf node directs the robot toward an unsafe region, the algorithm removes that node from the tree and stops further expansion from that branch. In Fig. 1, the algorithm stops expanding the tree from the right side due to existence of an obstacle.

Once safety is confirmed, the leaf nodes are evaluated using the state evaluation module. In this module, all the leaf nodes—both human and robot—are passed through the trained RL model to assign a value to each node. This value reflects how the robot's pose is close to the desired pose in relation to the human. In other words, the closer the robot is to being directly in front of the human at a certain distance, the higher the value assigned to the node. Afterward, the human-related nodes (red nodes in the Fig. 2) are processed through the human future positioning probabilities module. This module assigns a probability to each human node based on the history of the human's positions over the past 3 seconds. It's important to note that all the robot nodes (blue nodes) are assigned an equal probability of $1/6$, where 6

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3537860

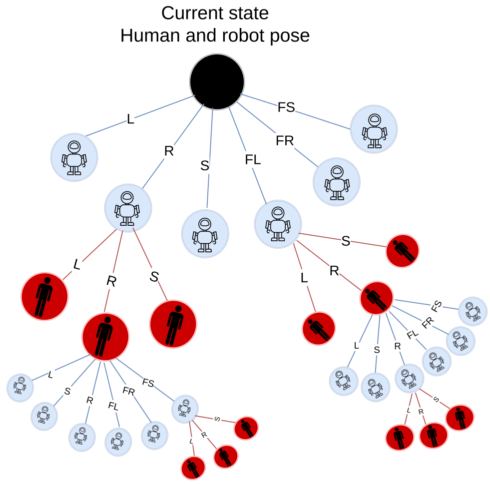


Fig. 2: Tree expansion in MCTS: Blue nodes represent possible future positions of the robot, while red nodes indicate potential future positions of the human. The letters ”L”, ”R”, ”S”, and ”F” on the edges of the tree represent the actions: Left, Right, Straight, and Fast, respectively.

represents the number of actions for the robot. Finally, In the backpropagation stage, the value obtained for each leaf node is propagated back through the tree, updating the values of all parent nodes up to the root node.

At the end of expansion, the immediate child node with the highest visit count (n) is selected as the optimal action for the next time step. The algorithm then updates with the new poses of the human and robot and re-expands the tree for the subsequent time step.

B. Human's Future Position Probability

Incorporating the probability of the human's next possible action enhances the performance of the decision-making process. We initially utilized several recent off-the-shelf methods for predicting multiple human trajectories [22], [23], [24]. However, these methods did not produce robust and accurate results for individual human predictions. The reason may be attributed to the training of these models on datasets [25], [26] include multiple individuals, where interactions influence individual behaviors.

To address this, we trained an LSTM-fc model specifically to sample a human's position over a three-second interval and generate probabilities for their next possible actions. The fully connected layer attached to the LSTM enables the model to output the likelihood of the human walking straight, turning right, or turning left. For training, we employed the Human3.6M dataset [27]. The dataset includes various human motions. For this study, we selected the “walking“ motion to align with our application focus. The dataset was downsampled from its original sampling frequency of 50 Hz to 5 Hz, resulting in a dataset of 700,000 points that represent the 2D position of a walking person. For model training, sequences of consecutive 15 points (equivalent to three seconds at 5 Hz) were extracted In each sequence, the final point was designated as the ground truth, while the preceding points served as input for the model. This approach enabled the model to predict the probability distribution for the human's future direction as left, right, or straight. The trained model achieved an evaluation accuracy of 92.47%, in predicting human directional changes.

Integrating this module with the tree expansion and utilizing the probabilities has the most impact when the human follows a consistent trajectory without changes. Without this module, the robot acts very cautiously, expecting the human to change direction at each time step. However, with this module, the robot is better able to navigate in front of the human. The module helps the tree in selecting the human leaf node with higher probabilities more frequently.

Case Study: *Consider a scenario where the human has been turning to the right for the past few time steps, and the robot is positioned on the human's left side. There is a high probability that the human will continue turning right in the future. Without this module, the robot assumes equal probabilities for turning left, right, and going straight, leading it to prefer staying on the human's left side in case the human turns left. However, with this module, the higher probability of the human turning right (compared to left or straight) causes the tree to be primarily expanded with the human leaf node indicating a right turn. Consequently, the robot is more confident in turning right to navigate in front of the human.*

As a final point, integrating this module with MCTS allows the robot to account for occasional sudden changes in the human's trajectory, while also considering consistent trajectory directions during other times.

C. Node Evaluation

Reinforcement Learning (RL) solves sequential decision-making problems using value functions to predict accumulated rewards for a policy [28]. However, they do not always prevent catastrophic failures due to the inherent nature of reward accumulation over time. Additionally, standard RL can be data-inefficient and challenging for complex tasks, as it requires training agents to make low-level decisions at every time step. To address this challenge, we trained an agent using the Asynchronous Actor-Critic (A2C) [29] method to learn a policy and its corresponding value function. We then integrated this value function with the tree expansion process to evaluate different human-robot poses and assign a value to each node in the tree. This approach helps the robot navigate in front of the human while ensuring safety.

To train the agent, we defined random trajectories for the human, conditioned only on the human's current location. This approach enhances the robustness and generalization of the learned value function across a wide range of human movements, without relying on a human motion prediction model. This design choice allows the agent to make appropriate decisions regardless of the human's actions. In the simulation, at each step, the human can randomly change direction by 20° to the right or left or continue in the current direction. The robot's observable state includes the relative distance and orientation to the human. For the action space, we use the same set of actions as described in section II-A.

To train the agent effectively, we utilized a reward function similar to that in [18], with targeted modifications informed by empirical experimentation. The key minor adjustment was to place greater emphasis on the human-robot angle, encouraging the agent to maintain an optimal position in front of the human. This adjustment improved the agent's ability to anticipate and adapt to directional changes and overall navigational accuracy. These modifications were introduced to better align the reward function with the specific requirement of this study while maintaining the core structure of the original in [18]. The modified reward function is as follows:

$$r = r_d + r_\alpha, \tag{3a}$$

$$\text{where } r_d = \begin{cases} d - 0.5 & 0.5 < d < 1 \\ 1 - |d - 1.5| & 1 < d < 2 \\ 1 - 0.25d & 2 < d < 4 \\ -1 & \text{otherwise} \end{cases} \tag{3b}$$

$$\text{and } r_\alpha = \begin{cases} (25 - |\alpha|)/25) & |\alpha| < 50 \\ -1 & \text{otherwise} \end{cases} \tag{3c}$$

The reward function consists of two components: r_α and r_d. The angle reward, r_α, directs the robot to stay positioned in front of the person, while the distance reward, r_d, helps the robot to maintain a specified distance range ahead of the person. In the equations, α denotes the angle between the person-robot vector and the person's heading direction, and d represents the relative distance between the human and the robot.

IV. EXPERIMENTS

In this paper, we propose a novel methodology comprising three integrated modules: RL, LSTM, and MCTS. This approach introduces a unique consideration of distinct action spaces for humans and robots, enabling the system to dynamically capture and adapt to sudden changes in human trajectories with reasonable probability. This integration builds on prior work by improving responsiveness and adaptability, addressing challenges in scenarios with frequent and unpredictable human direction changes. In this section, we present a series of experiments to evaluate the robot's performance and compare the results with those of previous works [18] and [8]. The experiments are conducted in both simulation and real-world settings. The algorithm requires two inputs: (i) an occupancy map of the environment, indicating the position of surrounding obstacles, and (ii) the position and orientation of both the human and the robot, which are obtained from a camera and odometry published from the robot. We used the embedded 3D object detection feature in the ZED 2 camera SDK by Stereolabs [21] to capture the real-time 3D pose of the human target relative to the camera. The camera is mounted on top of the robot and attached to a motor. Additionally, a PID controller is implemented to rotate the camera during the experiments, ensuring a clear view of the human and accurately capturing the human's pose. Experimental observations showed that allowing the motor to make small angle adjustments when the human is off-center in the frame achieved a stable, noise-free human pose for clear and reliable imaging. The ZED 2 camera captured human pose data at a frequency of 5 Hz for smooth and reliable input for the algorithm. For the real-world experiments, we used the RB1 base robot, a ZED2 camera, and a Dynamixel servo motor.

In the following sections, we compare the mean distance error and angle of the robot relative to the human with previous works. The desired distance between the human and the robot is set to 1.5 meters, and the distance error is defined as the difference between the robot's actual distance from the human and this target distance. Thus, a smaller distance error value signifies better alignment with the desired distance. Similarly, a lower angle value indicates that the robot is positioned more directly in front of the human, which is preferable. Additionally, we demonstrate how incorporating the human future position probability enhances the robot's performance. For each of the mentioned scenarios, we conducted 10 distinct simulation experiments to compare the robot's performance in comparison with previous methods. The reason for using the simulation is to ensure identical and controlled values for parameters such as the human's speed and the initial positions of both the human and the robot. The final experiment demonstrates how the robot effectively avoids collisions with surrounding obstacles, while maintaining its position in front of the human. It is worth noting that the significance of using an RL model and its role in improving the robot's performance through decision-making process has already been established in our previous work [18]. Additional supplementary materials, including videos of the experiments, can be found: *https://saharleisiazar.github.io/follow-ahead-adoption/*

A. Evaluating Robot's Performance

In this experiment, we compare the robot's performance with our previous work [18]. The experiment is conducted in both real-world and simulation environments, where different trajectories are designed for a human to walk on at a velocity of 0.7 m/s. During the experiments, we recorded the positions of both the human and the robot using a Vicon optical motion capture system (version 2.2). This data was utilized to compute the mean relative distance and angle between the human and the robot. It is important to note that the data obtained from the Vicon system was only used for evaluation and visualization purposes. The algorithm relied solely on camera input data, with the camera mounted on top of the robot and rotated by a motor to maintain a clear view of the human during the experiments.

In the subsequent experiments, two distinct types of trajectories were designed for the human subject. In one scenario; the human suddenly changed their trajectory multiple times, while in the other, the human had a smooth change of direction. To quantify and define these terms: a sudden change is characterized by the human altering their direction by more than $45°$ degrees in a single time step, whereas a smooth change involves turning approximately $10°$ degrees at each time step.

1) Sudden changes in human trajectory: The purpose of this experiment is to simulate scenarios where a human suddenly changes direction, a common occurrence in real-world situations. Figures. 3 and 4 illustrate the trajectories of the human and the robot during two different experiments conducted with the RB1 base robot. In both experiments, the human changes their trajectory twice within a time frame of about 10 to 14 minutes. The left image in both figures illustrates the human and robot trajectories using the proposed algorithm, while the right image shows the trajectories using the previous approach. The rainbow color scale indicates the time dimension, with red and purple denoting the first and last time steps, respectively. As depicted, the robot effectively adjusts to the human's new trajectories and consistently follows from the front.

In Fig. 3, the mean angle achieved using the proposed algorithm is 0.17 rad, compared to 0.72 rad with the previous work, indicating a significant improvement. In Fig. 4, the mean angle with the proposed algorithm is 0.37 rad, while the previous work yields 0.48 rad, demonstrating further enhancement in performance.

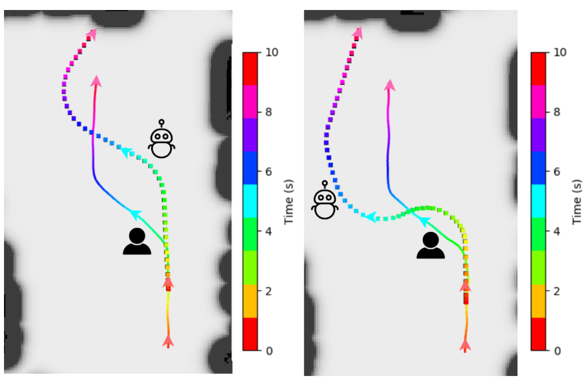


Fig. 3: Comparison of robot trajectories between the proposed algorithm and the previous method. (i) The left image shows the human and robot trajectories using the proposed algorithm, achieving 0.17 rad mean angle, (ii) the right image displays the trajectories using the previous method, achieving 0.71 rad mean angle. The rainbow color scale indicates the time dimension, with red and purple denoting the first and last time steps, respectively.

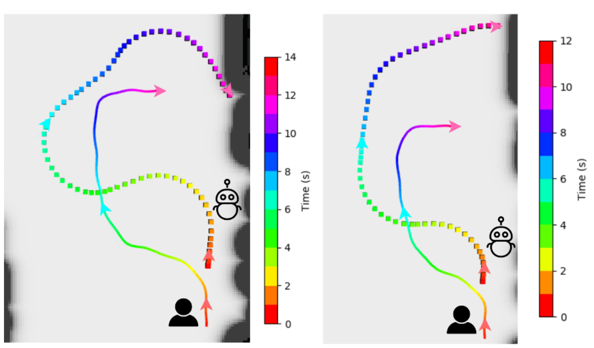


Fig. 4: Comparison of robot trajectories between the proposed algorithm and the previous method. (i) The left image shows the human and robot trajectories using the proposed algorithm, achieving 0.37 rad mean angle, (ii) the right image displays the trajectories using the previous method, achieving 0.48 rad mean angle. The rainbow color scale indicates the time dimension, with red and purple denoting the first and last time steps, respectively.

TABLE I: Follow-ahead comparative results for human trajectories with sudden changes. The closer the distance error and orientation values are to 0, the better the performance.

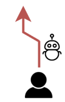
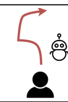
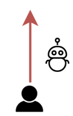
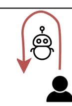
Human Trajectory	Method	Distance error (m) Mean ± std	Angle (rad) Mean ± std
	Ours	**0.05±0.09**	**0.03±0.35**
	[18]	0.12±0.06	0.15 ±0.39
	Ours	**0.07±0.09**	**0.35±0.59**
	[18]	0.08±0.09	0.28 ±0.69

TABLE II: Follow-ahead comparative results for human trajectories with smooth changes. The closer the distance error and orientation values are to 0, the better the performance.

Human Trajectory	Method	Distance error (m) Mean ± std	Angle (rad) Mean ± std
	Ours	**0.13±0.06**	**0.16±0.14**
	[18]	**0.13±0.06**	**0.16±0.14**
	[8]	0.17 ± 0.36	0.23±0.44
	Ours	**0.05±0.06**	**0.04±0.2**
	[18]	0.1±0.07	0.05 ±0.38
	[8]	0.41 ± 0.43	0.33± 0.51
	Ours	**0.22±0.23**	**0.13±0.11**
	[18]	0.24±0.15	0.20 ±0.13
	[8]	0.32 ± 0.3	0.33± 0.48

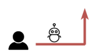
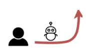
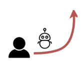
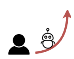
Moreover, the results of the simulation experiments comparing the two approaches are shown in Table I, presenting the mean distance error and angle achieved with each method. For each trajectory, we conducted several experiments with random initial pose of the robot with respect to the human. As a result, we conclude that with the proposed approach, the robot can adapt to the human's sudden changes more quickly and follow the human from the front more effectively.

2) Smooth changes in human trajectory: Similar to the previous experiment, additional simulation tests were conducted to compare the robot's performance when the human exhibits smooth changes in their trajectories. Table II presents the mean distance error and angle obtained using three methods: the proposed approach, the method from [18], and the method from [8]. We conclude that although the previous approaches allow the robot to navigate in front of the human, the proposed method enables the robot to follow the human more effectively and adapt to changes in the human's direction more rapidly.

B. Sharp Turn

In this experiment, four trajectories were designed for the human, with the same start and end points in all trajectories. The initial pose of the robot was also kept consistent. The

TABLE III: Comparison of the mean relative distance error and angle between the robot and human using two different methods. The data are presented for four human trajectories, each with varying turn radii, to evaluate the performance of the methods under different turning conditions.

Human Trajectory	Method	Distance error (m) Mean ± std	Angle (rad) Mean ± std
	Ours [18]	0.02±0.1 **0.0±0.1**	**0.62±0.7** 0.64 ±0.7
	Ours [18]	0.08±0.13 **0.0±0.1**	**0.61±0.5** 0.72 ±0.60
	Ours [18]	0.2±0.1 **0.13±0.2**	**0.36±0.4** 0.65 ±0.5
	Ours [18]	0.3±0.1 **0.16±0.10**	**0.10±0.10** 0.51 ±0.30

only difference was in how the human executed the turns: in the first trajectory, the human made a sharp 90° turn; while in subsequent trajectories, the radius of the turn gradually increased. Table III shows the mean relative distance error and angle across simulation experiments for both the proposed method and the method in [18]. The results indicate that the proposed method enables the robot to follow the human more effectively, maintaining a position in front of the human with a smaller angle value.

C. Ablations for Different UCBs

In this section, we perform an ablation study to determine the most effective method for integrating the probability of the human's next actions with the UCB value in MCTS. As discussed in section. III-B, the probability of the human's next action is most impactful when the human maintains a consistent trajectory without changing direction. To investigate this, we designed four trajectories for the human: walking straight, and turning to the right by $4°, 8°, 12°$ at each time step. We conducted experiments for each trajectory, varying the robot's pose relative to the human and reported the average reward, in Eq. (3), obtained from each time step.

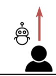
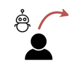
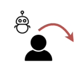
The evaluation compared different versions of the modified UCB equations (Eqs. 4 and 2) based on the mean rewards achieved, focusing on turning right due to similar results for both directions. The results, presented in Table IV, showed that Eq. 2 outperformed Eq. 4 in mean reward. Incorporating the probability of the human's next action further improved the robot's performance, particularly as the human's turning angle increased.

$$UCB_0 = \frac{V}{c} + c\sqrt{\frac{log(n_p)}{n}} \tag{4a}$$

$$UCB_1 = \frac{V}{c} + Pc\sqrt{\frac{log(n_p)}{n}} \tag{4b}$$

$$UCB_2 = P\frac{V}{c} + c\sqrt{\frac{log(n_p)}{n}} \tag{4c}$$

TABLE IV: Comparison of mean rewards achieved using different UCB equations, Eqs. (4) and Eq. (2), for various human trajectories.

Trajectory	Turn Angle	UCB_0	UCB_1	UCB_2	Eq. (2)
	0	1.64 ±0.12	1.58 ±0.16	1.58 ±0.10	**1.65** **±0.11**
	4	1.45 ±0.1	1.5 ±0.08	**1.58** **±0.11**	1.50 ±0.07
	8	0.97 ±0.23	0.9 ±0.25	1.18 ±0.26	**1.37** **±0.20**
	12	-0.19 ±0.40	-0.13 ±0.44	0.18 ±0.57	**0.30** **±0.52**

D. Obstacles Avoidance

In this section, we conducted an experiment to evaluate the robot's obstacle avoidance performance. Figure. 5 illustrates two distinct scenarios. In the left image, a box is placed in the environment, and the human walks toward it. When the robot reaches the box before the human, moving straight is no longer an option, so the robot explores turning left or right. Given that the human's direction is slightly to the right, the robot chooses to turn right. By the time the human reaches the box, the robot is already positioned on the right side of it.

In the right image, the human walks toward the obstacle and stops in front of it, while the robot turns right to avoid a collision and then turns left to reposition itself in front of the human. Similar to the previous experiments, the human's trajectory is depicted with a line, while the robot's path is shown with squares, both in rainbow-colored.

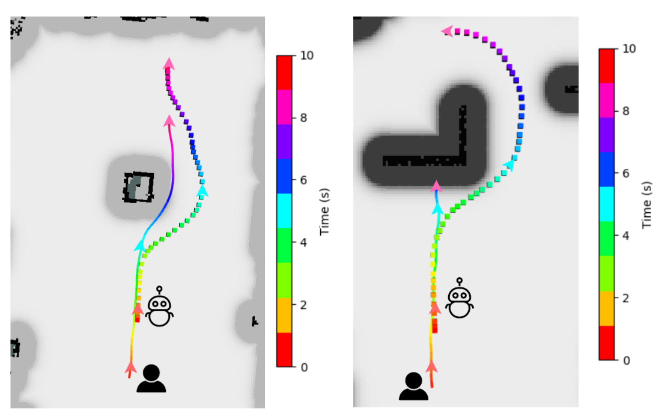


Fig. 5: Evaluation of the robot's obstacle avoidance performance. In the left image, the robot reaches the box before the human, turns right to avoid the collision. In the right image, the human walks toward the obstacle and stops in front of it, while the robot turns right to avoid the obstacle and then turns left to stay in front of the human. The human's trajectory is depicted with a rainbow-colored line, and the robot's path is shown with rainbow-colored squares.

V. CONCLUSIONS

In this paper, we introduced a novel approach for human follow-ahead robots that adapts to the unpredictable and dynamic nature of human behavior.

Unlike previous approaches that depend on predicting the human's trajectory and goal, our method considers various potential actions and corresponding future positions for the human in the decision-making process. This allows the robot to anticipate abrupt changes in the human's trajectory, enabling it to rapidly adapt to sudden shifts. Additionally, we integrated an LSTM-based model to generate a probability distribution over the human's future actions, incorporating these probabilities into the MCTS framework.

By considering different potential actions and future positions for the human during tree expansion, the robot is prepared for changes in the human's trajectory at each time step. However, integrating probability into the tree expansion ensures that the robot is not overly cautious, allowing it to follow the human more smoothly and quickly when the human walks along a trajectory without sudden direction changes.

Through extensive experiments in both simulated and real-world environments, we demonstrated the effectiveness of our approach in maintaining the robot's position in front of the human while successfully avoiding obstacles. The results demonstrate that the robot effectively navigates complex environments and swiftly adapts to changes in the human's trajectory, ensuring the safety of the user and the operational efficiency of the robot.

REFERENCES

[1] J. E. Sierra-García, V. Fernández-Rodríguez, M. Santos, and E. Quevedo, "Development and experimental validation of control algorithm for person-following autonomous robots," *Electronics*, vol. 12, no. 9, p. 2077, 2023.

[2] K. Koide, J. Miura, and E. Menegatti, "Monocular person tracking and identification with on-line deep feature selection for person following robots," *Robotics and Autonomous Systems*, vol. 124, p. 103348, 2020.

[3] N. Van Toan, S.-H. Bach, and S.-Y. Yi, "A hierarchical approach for updating targeted person states in human-following mobile robots," *Intelligent Service Robotics*, vol. 16, no. 3, pp. 287–306, 2023.

[4] H. Ye, J. Zhao, Y. Pan, W. Cherr, L. He, and H. Zhang, "Robot person following under partial occlusion," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7591–7597.

[5] J. Zhang, J. Guo, H. Chai, Q. Zhang, Y. Li, Z. Wang, and Q. Zhang, "A day/night leader-following method based on adaptive federated filter for quadruped robots," *Biomimetics*, vol. 8, no. 1, p. 20, 2023.

[6] P. Nikdel, R. Shrestha, and R. Vaughan, "The hands-free push-cart: Autonomous following in front by predicting user trajectory around obstacles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4548–4554.

[7] M. Mahdavian, P. Nikdel, M. TaherAhmadi, and M. Chen, "Stpotr: Simultaneous human trajectory and pose prediction using a non-autoregressive transformer for robot follow-ahead," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9959–9965.

[8] P. Nikdel, R. Vaughan, and M. Chen, "Lbgp: Learning based goal planning for autonomous following in front," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3140–3146.

[9] F. I. Khawaja, A. Kanazawa, J. Kinugawa, and K. Kosuge, "A human-following motion planning and control scheme for collaborative robots based on human motion prediction," *Sensors*, vol. 21, no. 24, p. 8229, 2021.

[10] J. Peng, Z. Liao, H. Yao, Z. Su, Y. Zeng, and H. Dai, "Mpc-based human-accompanying control strategy for improving the motion coordination between the target person and the robot," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 7969–7975.

[11] M. Świechowski, D. Lewiński, and R. Tyl, "Combining utility ai and mcts towards creating intelligent agents in video games, with the use case of tactical troops: Anthracite shift," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–8.

[12] B. Woo, P. Sweetser, and M. Aitchison, "Hivemind: Learning to play the cooperative chess variant bughouse with dnns and mcts," in *2023 IEEE Conference on Games (CoG)*. IEEE, 2023, pp. 1–8.

[13] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[14] T. Dam, G. Chalvatzaki, J. Peters, and J. Pajarinen, "Monte-carlo robot path planning," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 213–11 220, 2022.

[15] W. Li, Y. Liu, Y. Ma, K. Xu, J. Qiu, and Z. Gan, "A self-learning monte carlo tree search algorithm for robot path planning," *Frontiers in Neurorobotics*, vol. 17, 2023.

[16] Y. Wang, Y. Wei, X. Huang, S. Gao, and H. Zou, "Robot navigation with predictive capabilities using graph learning and monte carlo tree search," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 237, no. 5, pp. 805–814, 2023.

[17] X. Zheng, X. Zhang, and D. Xu, "Speeding up path planning via reinforcement learning in mcts for automated parking," *arXiv preprint arXiv:2403.17234*, 2024.

[18] S. Leisiazar, E. J. Park, A. Lim, and M. Chen, "An mcts-drl based obstacle and occlusion avoidance methodology in robotic follow-ahead applications," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 221–228.

[19] H. Baier and P. I. Cowling, "Evolutionary mcts for multi-action adversarial games," in *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2018, pp. 1–8.

[20] M. Daneshvaramoli, M. S. Kiarostami, S. K. Monfared, H. Karisani, K. Dehghannayeri, D. Rahmati, and S. Gorgin, "Decentralized communication-less multi-agent task assignment with cooperative monte-carlo tree search," in *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2020, pp. 612–616.

[21] T. Miller, "Monte-carlo tree search," https://gibberblot.github.io/rl-notes/single-agent/mcts.html.

[22] P. Xu, J.-B. Hayet, and I. Karamouzas, "Socialvae: Human trajectory prediction using timewise latents," in *European Conference on Computer Vision*. Springer, 2022, pp. 511–528.

[23] L. Shi, L. Wang, S. Zhou, and G. Hua, "Trajectory unified transformer for pedestrian trajectory prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9675–9684.

[24] T. Maeda and N. Ukita, "Fast inference and update of probabilistic density estimation on trajectory prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9795–9805.

[25] S. Pellegrini, A. Ess, and L. Van Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part I 11*. Springer, 2010, pp. 452–465.

[26] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer graphics forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 655–664.

[27] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 7, pp. 1325–1339, 2013.

[28] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[29] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.