A Novel Model-Free Learning Approach for Dynamic Target Tracking Problem and its Validation using Virtual Robot Experimentation Platform

Amr Elhussein and Md Suruz Miah
Electrical & Computer Eng.
Bradley University, Peoria, Illinois, USA
smiah@bradley.edu and aelhussein@mail.bradley.edu

***Abstract*—This paper advances the previous work done by authors by validating a model-free actor-critic reinforcement learning approach to solve dynamic target tracking problem for a car-like mobile robot. The learning approach generates the linear velocity and steering angle of the robot and it does not require any prior knowledge of the dynamic model of the moving target. Policy iteration approach is employed through Bellman's principle of optimality to assess the cost of the control actions derived by the proposed learning method. The algorithm is tested by coducting a set of computer expeeriments for complex secnarios using virtual robot expirementation platform widely known as CoppeliaSim.**

Need to be rewritten

***Index Terms*—Leader-follower formation, mobile robots, reinforcement learning, policy iteration, trajectory tracking**

I. Introduction

Need to discuss in details the limitations of previous work, we can also add RL applications to robotics navigation

Tracking a random moving target using a mobile robot, for instance, is a challenging task. This is mainly due to their inherent complex nonlinear dynamics dynamics. Most of the target tracking algorithms proposed in the literature either rely on A) complex mathematical models, B) simplified mathematical models, and C) nonlinear control techniques or driven by large amount of mostly offline data that leads to an overwhelming degree of computational complexity.

Over the past years mobile robots has been used in several applications in commercial and military sectors such as surveillance, search and rescue missions,coverage optimization, cooperative localization and dynamic target tracking tasks. In all of these mentioned applications a fleet of robot i.e agents interact with each other to achieve a certain goal. Leader-Follower tracking problem has recieved an extensive amount of study and research in the world of cooperative control theory. In a typical leader follower formation a number of robots referred to as followers apply local control actions to follow leadr's robots in a specific predefined path such as cyclic, circular motion and time varying communication topologies. limitations of other methods many control methods were proposed however they have had the following limitations:
(1) rely on the mathemetical model such as...
(2) rely on the dynamic model even with following a computational intillegence approach such as ..
The main contribution of this work is the development of a model-free learning approach to control the follower robot by which it overcome many of the limitations mentiond above as it does not rely on any prior information of the mathmetical model or the dynamic model. The steering angle and the linear speed of the follower robot are determined by collecting the position and orientation of both the leader and the follower. This set of information is gathered online over a finite period of time. The optimal control actions are then generated by utilizing Bellman's principal of optimality which acts as model free-reinfocment learning approach that allows the follower robot to follow the path of the leader while ovoiding collision by maintaining a safe distance. In this paper the proposed algorithm is further validated using a commercially available robot simulator, CoppeliaSim. This paper acts as first milestone in generalizing the algorithm to solve more sophisticated problem such as coverage and mapping. cite area coverage papers
The rest of the paper is orgnized as follows. Section II lays down the problem setting of the leader follower problem and mathemetical models of the robots and the state error. The model free actor-critic reinforcment learning and its key steps are described in section III.Section IV illustrates computer simulations for different secnarios that reflects the effectivness of the proposed method followed by conlusion and future work presented in section V.

Take some references from the following paragraphs

The task of localization and mapping in the context of leader-follower problems is very challenging. Such a task is becoming a ubiquitous facet of modern life due to its promising application in addressing not only the leader-follower problem but also applicable in industry industry with heavy equipment trucks, in agriculture with autonomous crop maintenance, in home life with the autonomous vacuuming robot, along with research and design as seen in [**?**], [**?**], [**?**], [**?**]. In most leader-follower problem addressed in the literature to date, it is assumed that leader/target's position/state is known *a priori*, which may not be the case in many applications such as tracking a mobile point source [**?**]. Furthermore, the

localization and mapping task is of a paramount importance in addressing the problem of mobile target tracking, which has a number of promising applications, such as robotic navigation, search and rescue mission, wildlife monitoring, autonomous surveillance, to name a few. A large body of research has been conducted in the literature to address the localization and mapping problem. Relatively few papers have addressed the localization and mapping problem in the context of leader-follower tasks, where either (i) a static leader/target is used [**?**], (ii) the leader/target's location is predetermined, or (iii) expensive hardware platforms are used to implement the mobile target tracking strategy (see the work presented in [**?**], for example).

The work done in this paper addresses some of the aforementioned issues by carrying out a cost-effective and easy-to-implement localization and mapping algorithm in the context of a leader–follower problem, a wheeled mobile robot is employed as a follower and a target moving on a two-dimensional (2D) plane is used as a leader. We emphasize that the position of the leader is unknown to the follower robot *a priori*. Therefore, the position and orientation of the follower robot and the position of the leader are to be simultaneously estimated while building the map of the environment of the robot using a set of networked wireless radio sensors placed on the ground. It is assumed that a radio sensor is mounted on the leader. The follower robot receives range measurements [herein the received signal strength indicator (RSSI) from radio sensors] from all radio sensors in its operating range. These measurements are then used to estimate the position and orientation of the follower robot, the 2D position of the leader, and the 2D positions of radio sensors placed in the robot's environment. Once the follower robot estimates its states (position and orientation) and the position of the leader, it moves towards the leader using the motion controller running onboard the follower robot. The design and implementation of the current work are based on the preliminary work of the robot navigation and mapping strategy conducted by the authors in [**?**], [**?**], [**?**]. Therefore, the main objective is to advance the work in [**?**], [**?**], [**?**] for a more general case, where a mobile robot is to follow a leader whose position is unknown *a priori*. n

II. Dynamic target tracking and problem setting

Needs to re-written

Suppose that a wheeled mobile robot (follower) with coordinate (x, y) and orientation $\theta \in [-\pi, \pi)$ rad with respect to the global X-Y coordinate has a linear speed ν and steering angle γ which are considered as the two control actions. Let the posistion and orientation (pose) of the target i.e leader be $\mathbf{q}_k \equiv [x_k, y_k, \theta_k]$. The robot and the target are deployed in a 2D configuration in which the Follower is to follow the target trajectory defined by $(\mathbf{p}_k^{[\ell]})^T = [x_k^{[\ell]}, y_k^{[\ell]}(t)]$ at time $t \geq 0$ with $t = k\,T_s$, $k \in \mathbb{N}_0$, and $T_s > 0$ being the sampling time. Note

redraw figure and add coppelia sim figures

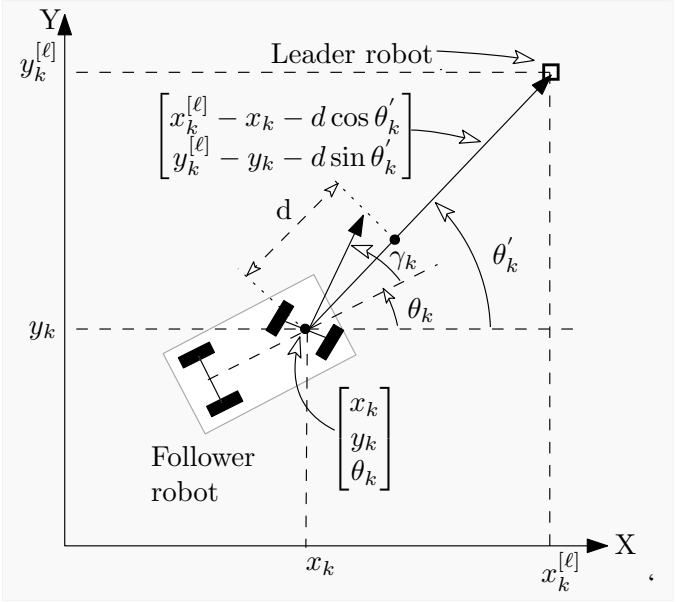

Fig. 1. Mobile robot and its leader-follower problem setup.

that this trajectory is completley random. The target motion is modeled by the following discrete-time system:

$$\mathbf{p}_{k+1}^{[\ell]} = \mathbf{p}_k^{[\ell]} + T_s\,\mathbf{u}_k^{[\ell]}, \tag{1}$$

with $\mathbf{u}_k^{[\ell]} \in \mathbb{R}^2$ being the control inputs of the target movment. The follower robot is to maintain a constant safe-distance $d > 0$ from target. The dynamic car-like model of the follower robot is defined as follows:

$$x_{k+1} = x_k + T_s\nu_k \cos(\theta_k + \gamma_k) + \zeta_1, \tag{2a}$$
$$y_{k+1} = y_k + T_s\nu_k \sin(\theta_k + \gamma_k) + \zeta_2, \tag{2b}$$
$$\theta_{k+1} = \theta_k + T_s\nu_k \frac{\sin(\gamma_k)}{l} + \zeta_3, \tag{2c}$$

Note that the follower dynamic model obeys Ackermann steering principle where $\gamma_k \in (-\frac{\pi}{2}, \frac{\pi}{2})$ is the front wheel steering angle with respect to the robot's orientation $\theta_k \in [-\pi, \pi)$, ν_k is the linear speed, l is the distance between the drive wheels of the robot, and $\zeta_1, \zeta_2, \zeta_3 \in \mathbb{R}$ are the model uncertainties. We then define the error vector as:

$$\mathbf{e}_k^T = [x_e, y_e, \theta_e]^T = \left[x_k^{[\ell]} - x_k - d\cos\theta_k', y_k^{[\ell]} - y_k - d\sin\theta_k', \theta_k' - \theta_k\right], \tag{3}$$

where $\theta_k' = \text{atan2}\left(y_k^{[\ell]} - y_k, x_k^{[\ell]} - x_k\right)$.

The control problem can then be formally stated as follows: Find ν_k and γ_k such that $\mathbf{e}_k \to \mathbf{0}$ as $k \to \infty$ subject to (**??**) and (**??**).

III. Proposed actor-critic RL Approach

Needs to be rewritten

The solution of the leader-follower formation problem is realized using a reinforcement learning approach. It employs model-free strategies for solving a temporal difference equation developed herein. This solution is equivalent to solving the underlying Bellman optimality equation for the dynamical error model (**??**). The relative importance of the states in the error vector $\mathbf{e}_k$ and the control decisions (linear velocity and steering angle) of the follower-robot are evaluated using the performance (cost) index:

$$J = \sum_{k=0}^{\infty} \frac{1}{2} \left[\mathbf{e}_k^T \, \mathbf{Q} \, \mathbf{e}_k + \mathbf{u}_k^T \, \mathbf{R} \, \mathbf{u_k} \right], \tag{4}$$

where $\mathbf{Q} \in \mathbb{R}^{3\times 3}$ and $\mathbf{R} \in \mathbb{R}^{2\times 2}$ are symmetric positive definite weighting matrices. The objective of the optimization problem, following [**?**], is to find an optimal sequence of control polices $\{\mathbf{u}_k^*\}_{k=0}^{\infty}$ that minimizes the cost index J along the state-trajectories (**??**) and (**??**). Motivated by the structure of the convex quadratic cost functional (**??**), let the solution of the tracking control problem employ the value function $V(\mathbf{e}_k, \mathbf{u}_k)$ defined by

$$V(\mathbf{e}_k, \mathbf{u}_k) = \sum_{\kappa=k}^{\infty} \frac{1}{2} \left(\mathbf{e}_\kappa^T \, \mathbf{Q} \, \mathbf{e}_\kappa + \mathbf{u}_\kappa^T \, \mathbf{R} \, \mathbf{u}_\kappa \right).$$

This structure yields a temporal difference form (i.e., Bellman equation) as follows

$$V(\mathbf{e}_k, \mathbf{u}_k) = \frac{1}{2} \left[\mathbf{e}_k^T \, \mathbf{Q} \, \mathbf{e}_k + \mathbf{u}_k^T \, \mathbf{R} \, \mathbf{u_k} \right] + V(\mathbf{e}_{k+1}, \mathbf{u}_{k+1}).$$

Applying Bellman's optimality principle yields the optimal control policies $\mathbf{u}_k^*,\ k \geq 0$, such that [**?**]

$$\mathbf{u}_k^* = \underset{\mathbf{u}_k}{\operatorname{argmin}} \left[\frac{1}{2} \left[\mathbf{e}_k^T \, \mathbf{Q} \, \mathbf{e}_k + \mathbf{u}_k^T \, \mathbf{R} \, \mathbf{u}_k \right] + V(\mathbf{e}_{k+1}, \mathbf{u}_{k+1}) \right].$$

Alternatively, this optimal policy form is equivalent to $\mathbf{u}_k^* = \operatorname{argmin}_{\mathbf{u}_k} \left[V(\mathbf{e}_k, \mathbf{u}_k) \right]$. Therefore, the underlying Bellman optimality equation follows

$$V^*(\mathbf{e}_k, \mathbf{u}_k^*) = \frac{1}{2} \left[\mathbf{e}_k^T \, \mathbf{Q} \, \mathbf{e}_k + \mathbf{u}_k^{*T} \, \mathbf{R} \, \mathbf{u}_k^* \right] + V^*(\mathbf{e}_{k+1}, \mathbf{u}_{k+1}^*),$$

where $V^*(\cdot, \cdot)$ is the optimal solution for Bellman optimality equation. This temporal difference equation is utilized by reinforcement learning process which solves the following temporal difference approximation form

$$\hat{V}(\mathbf{z}_k) = \frac{1}{2} \mathbf{z}_k^T \, \bar{\mathbf{P}} \, \mathbf{z}_k + \hat{V}(\mathbf{z}_{k+1}), \tag{5}$$

where $\mathbf{z}_k = [\mathbf{e}_k, \mathbf{u}_k]^T \in \mathbb{R}^5$, $V(\mathbf{e}_k, \mathbf{u}_k) \approx \hat{V}(\mathbf{z}_k)$, and $\bar{\mathbf{P}}$ is a symmetric block-diagonal matrix formed using $(\mathbf{Q}, \mathbf{R})$, *i.e.,* $\bar{\mathbf{P}} = \operatorname{blockdiag}(\mathbf{Q}, \mathbf{R})$. The approximation of the solving value function $\hat{V}(\mathbf{z}_k)$ employs a quadratic form so that $\hat{V}(\mathbf{z}_k) = \frac{1}{2} \mathbf{z}_k^T \, \mathbf{P} \, \mathbf{z}_k$, where $\mathbf{P} \in \mathbb{R}^{5\times 5}$ is a positive definite matrix. Hence, the optimal control strategy $\mathbf{u}_k^*$ can be expressed as follows

$$\mathbf{u}_k^* = \underset{\mathbf{u}_k}{\operatorname{argmin}} \left[\hat{V}(\mathbf{z}_k) \right] = -\, \mathbf{P}_{uu}^{-1} \, \mathbf{P}_{ue} \, \mathbf{e}_k, \tag{6}$$

where $\mathbf{P}_{uu}$ and $\mathbf{P}_{ue}$ are sub-blocks of symmetric matrix $\mathbf{P}$.

Second, the policy evaluation step of this process updates the critic weights $\boldsymbol{\omega}$ in real-time without acquiring any formation about the dynamics of the leader or follower dynamical systems (the calculation mechanism of the critic weighs $\boldsymbol{\omega}$ is explained later on). This is done to search for a strictly better policy.

$$\mathbf{z}_k^T \, \mathbf{P} \, \mathbf{z}_k - \mathbf{z}_{k+1}^T \, \mathbf{P} \, \mathbf{z}_{k+1} = \mathbf{z}_k^T \, \bar{\mathbf{P}} \, \mathbf{z}_k. \tag{7}$$

This equation is utilized repeatedly in order to evaluate a certain policy during at least $\eta \geq \bar{n}, \bar{n} = (3+2)(3+2+1)/2$ evaluation steps (i.e., the lowest evaluation interval spans k to $k + \bar{n}$ calculation samples) in order to update the critic weights vector $\boldsymbol{\omega} = \operatorname{vec}(\mathbf{P})$, which consists of connection weights between the neurons of the hidden layer and the output layer of the critic neural network shown in Fig. (**??**). The operator $\operatorname{vec}(\mathbf{P})$ forms the columns of the $\mathbf{P}$ matrix into a column vector ω of dimension $\bar{n} = 15$ since the matrix $\mathbf{P}$ is a symmetric matrix. The left hand side of (**??**) is expressed using the following critic approximation form

$$\hat{V}(\mathbf{z}_k) - \hat{V}(\mathbf{z}_{k+1}) = \boldsymbol{\omega}^T \tilde{\boldsymbol{\rho}}(\mathbf{z}_{k,k+1}),$$

where $\tilde{\boldsymbol{\rho}}(\mathbf{z}_{k,k+1}) = \boldsymbol{\rho}(\mathbf{z}_k) - \boldsymbol{\rho}(\mathbf{z}_{k+1}) \in \mathbb{R}^{15\times 1}$, $\boldsymbol{\rho}(\mathbf{z}_k) = \left(\mathbf{z}_k^q \otimes \mathbf{z}_k^h\right)$ $(q = 1, \ldots, 5,\ h = q, \ldots, 5)$, and $\boldsymbol{\omega}^T = [0.5\, P^{11}, P^{12}, P^{13}, P^{14}, P^{15}, 0.5\, P^{22}, P^{23}, P^{24}, P^{25}, 0.5\, P^{33}, P^{34}, P^{35}, 0.5\, P^{44}, P^{45}, 0.5\, P^{55}]^T \in \mathbb{R}^{1\times 15}$ (P^{ij} is the ij^{th} entry of matrix $\mathbf{P}$). The critic weights $\boldsymbol{\omega}$ are updated using a gradient descent approach, where the tuning error ε_k at each computational instance k follows $\varepsilon_k = \boldsymbol{\omega}^T \tilde{\boldsymbol{\rho}}(\mathbf{z}_{k,k+1}) - v_k$, where $v_\kappa = \frac{1}{2} \mathbf{z}_k^T \, \bar{\mathbf{P}} \, \mathbf{z}_k$. As detailed earlier, it is required to perform at least $\eta \geq \bar{n}$ evaluation steps before updating the critic weights $\boldsymbol{\omega}$ (i.e., finding the new improved policy). Hence, it is required to minimize the sum of square errors such that

$$\begin{aligned} \delta_c = \sum_{\kappa=0}^{\eta-1} \frac{1}{2} (\boldsymbol{\omega}^T \tilde{\boldsymbol{\rho}}(\mathbf{z}_{k+\kappa,k+\kappa+1}) - v_{k+\kappa})^2 &= \frac{1}{2} \|\mathbf{v} - \boldsymbol{\Lambda}\boldsymbol{\omega}\|^2 \\ &= \frac{1}{2} \left(\mathbf{v} - \boldsymbol{\Lambda}\boldsymbol{\omega}\right)^T \left(\mathbf{v} - \boldsymbol{\Lambda}\boldsymbol{\omega}\right), \end{aligned}$$

where $\boldsymbol{\Lambda} = [\mathbf{o}_0, \mathbf{o}_1, \ldots, \mathbf{o}_{\eta-1}]^T \in \mathbb{R}^{\eta\times 15}$ with $\mathbf{o}_\kappa = \tilde{\boldsymbol{\rho}}^T(\mathbf{z}_{k+\kappa,k+\kappa+1}) \in \mathbb{R}^{1\times 15}$ and $\mathbf{v} = [v_0, v_1, \ldots, v_{\eta-1}]^T \in \mathbb{R}^{\eta}$ with $v_\kappa = \frac{1}{2} \mathbf{z}_{k+\kappa}^T \, \bar{\mathbf{P}} \, \mathbf{z}_{k+\kappa}$ for $\kappa = 0, 1, \ldots, \eta - 1$. Therefore, the update law of the critic weights using the gradient decent approach for at least $\bar{n}$ samples is given by

$$\begin{aligned} \boldsymbol{\omega}^{[r+1]} = \boldsymbol{\omega}^{[r]} - \ell_c \frac{\partial \delta_c}{\partial \boldsymbol{\omega}} &= \boldsymbol{\omega}^{[r]} - \ell_c \left(-\boldsymbol{\Lambda}^T \mathbf{v} + \boldsymbol{\Lambda}^T \boldsymbol{\Lambda} \boldsymbol{\omega}^{[r]} \right) \\ &= \boldsymbol{\omega}^{[r]} - \ell_c \boldsymbol{\Lambda}^T \left(\boldsymbol{\Lambda} \boldsymbol{\omega}^{[r]} - \mathbf{v} \right), \end{aligned} \tag{8}$$

where $0 < \ell_c < 1$ is a critic learning rate and r is the update index of the critic weights.

$$\mathbf{P} = \begin{bmatrix} 2\,\omega^1 & \omega^2 & \omega^3 & \omega^4 & \omega^5 \\ \omega^2 & 2\,\omega^6 & \omega^7 & \omega^8 & \omega^9 \\ \omega^3 & \omega^7 & 2\,\omega^{10} & \omega^{11} & \omega^{12} \\ \omega^4 & \omega^8 & \omega^{11} & 2\,\omega^{13} & \omega^{14} \\ \omega^5 & \omega^9 & \omega^{12} & \omega^{14} & 2\,\omega^{15} \end{bmatrix} \in \mathbb{R}^{5\times 5},$$

where ω^i is the i^{th} entry of the weight vector $\boldsymbol{\omega}$. The complete policy iteration solution process for the leader-follower problem is detailed out in Algorithm **??**.

Algorithm 1: Model-free reinforcement learning using the policy iteration solution.

Input: Sampling-time T_s, $\mathbf{Q}$, and $\mathbf{R}$
Output: Error trajectory $\mathbf{e}_k$, for $k = 0, 1, \ldots$
begin
  $k = 0, r = 0$ /* Discrete time and policy indices */
  $\eta = (n+m)(n+m+1)/2$
  Initialize $\mathbf{P}^{[0]}$ /* Positive definite */
  Set offset distance d
  Given approximate initial poses of leader and follower, compute $\mathbf{e}_0$ using error model (**??**)
  Compute follower's input $\mathbf{u}_0^{[0]}$ using policy (**??**)
  repeat /* Main timing loop */
    Find $\mathbf{e}_{k+1}$ using (**??**)
    Compute policy $\mathbf{u}_{k+1}^{[r]}$ using (**??**)
    if $[(k+1) \text{ modulo } \eta] == 0$ **then**
      $r \leftarrow r + 1$ /* Evaluate policy */
      Solve for the critic-weights $\boldsymbol{\omega}$ using (**??**)
      Construct matrix $\mathbf{P}^{[r]}$ using vector $\boldsymbol{\omega}$
    $k \leftarrow k + 1$
  until *Tracking errors are zero*

IV. Computer Experiments and Results

In the sequel, computer experiments are conducted to validate the performance of the proposed model-free adaptive learning algorithm in real-time. The results of computer experiments highlight the dynamics of the tracking errors and the convergence characteristics of the proposed algorithm (i.e., updating the critic weights). This will judge the ability of the follower robot to track independent motion trajectory of the leader robot under two different independent leader-motion trajectories. The computer experiments are realized using MATLAB simulation environment. The weighting matrices are set to $\mathbf{Q} = \text{diag}[0.01, 0.01, 0.005]$ and $\mathbf{R} = \text{diag}[10^{-6}, 10^{-6}]$.

$$Q = 0.001 \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 5 \end{bmatrix}, R = 0.000001 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The learning rate ℓ_c is set to 0.0001. The sampling time T_s is set to $0.08\,\text{sec}$. The desired distance offset between the leader and the follower is set to $d = 0.5$ [m] for all scenarios.

Initially, the leader's position is set to be the same for a short period of time with $\nu^\ell = 0, \gamma^\ell = 0$. Then it starts to move on a horizontal line with $\nu^\ell = 0.1$ [m/s], eventually the leader starts to move on an inclined path with $\nu^\ell = 0.2$ [m/s], $\gamma^\ell = 45^\circ$. The leader is initially placed at $(x, y) = (5, 2)$ [m] with an orientation of $\theta = 0^\circ$, while the follower is initially placed at $(x, y) = (0, 0)$ [m] with an orientation of $\theta = 0^\circ$. Note that during this scenario, the critic weights converge rapidly and the tracking errors decay till the follower tracks the leader. The trajectory phase plan plot, tracking error states, control signals, and tuning of critic weights are shown in Fig. **??**.

A relatively unplanned complex trajectory-tracking scenario is considered. The leader moves according to a sinusoidal trajectory so that

The leader robot was initially placed at $(x, y) = (0, 0)$ [m] with an orientation of $\theta = 0^\circ$ while the follower robot was initially placed at a random position around $(x, y) = (0, 0)$ [m] with a random orientation. The simulation results are summarized in Figure **??**. It is observed that the critic weights take more time to converge compared to the earlier scenario, which is reluctant to the complexity of the independent trajectory of the leader. This result emphasizes the adaptability of the proposed adaptive learning mechanism to different scenarios. Further, the follower starts to move away from the leader at the beginning of the simulation before it finally converges to the leader where the tracking errors are bounded by a safe distance d.

In the third case study we rely on completley random path for the leader robot to evaluate the effectivenss of the proposed algorithm as a solution for the leader follower tracking problem. The leader is to follow a random path defined by the model described in (2), with γ being a random value, The follwer is to maitain an offset distance of $d = 0.5$ [m]. Note that the path of the leader is not pre defined and the algorithm being completlety model free. The results are summerized in Fig. **??**.

V. Conclusion

an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.