2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Simultaneous Localization of Biobotic Insects using Inertial Data and Encounter Information

Jeremy Cole[1], Alper Bozkurt[1] and Edgar Lobaton[1]

***Abstract*— Several recent research efforts have shown that the bioelectrical stimulation of their neuro-mechanical system can control the locomotion of Madagascar hissing cockroaches (*Gromphadorhina portentosa*). This has opened the possibility of using these insects to explore centimeter-scale environments, such as rubble piles in urban disaster areas. We present an inertial navigation system based on machine learning modules that is capable of localizing groups of *G. portentosa* carrying thorax-mounted inertial measurement units. The proposed navigation system uses the agents' encounters with one another as signals of opportunity to increase tracking accuracy. Results are shown for five agents that are operating on a planar (2D) surface in controlled laboratory conditions. Trajectory reconstruction accuracy is improved by 16% when we use encounter information for the agents, and up to 27% when we add a heuristic that corrects speed estimates via a search for an optimal speed-scaling factor.**

I. Introduction

Earlier studies in the literature have shown that it is possible to remote control the locomotion of a Madagascar hissing cockroach (*Gromphadorhina portentosa*) via the bioelectrical stimulation of its neuro-mechanical system [1]. It has also been shown that this particular species of cockroach exhibits extreme maneuverability, including wall climbing capability [2] and the ability of self-righting themselves. These roaches grow to be approximately 60mm long and 30mm wide, with a payload capacity of approximately 15g [1]. The roaches use a combination of pretarsal claws and adhesive pads to cling to and move on a wide variety of surfaces [2], with top speeds of several cm/sec. Their exoskeleton is a compliant structure, allowing them to fall from heights and squeeze under obstacles without issue [3]. It has been proposed that cockroaches outfitted with various electronic sensor payloads can function as a biological robot (or biobot) for disaster response, in particular for search, reconnaissance, and mapping tasks in urban ruins (e.g., for search and rescue operations) necessitating extreme mobility [4].

Our earlier work [5] has shown that it is possible to localize biobotic agents using low-cost inertial measurement units (IMUs) using an inertial navigation system (INS). In this work, we extend the prior navigation system by adding the ability to jointly track and refine the poses of multiple biobots by leveraging proximity information that registers when the biobots encounter one another—this information could be obtained from ranging sensors that are attached to the biobots themselves. Our main contribution is the development of the aforementioned multi-agent navigation system, which allows the tracking of biobots with increased accuracy. Fig. 1 illustrates a biobot following a specific trajectory through an unobserved region as would be the case for the use case under consideration.

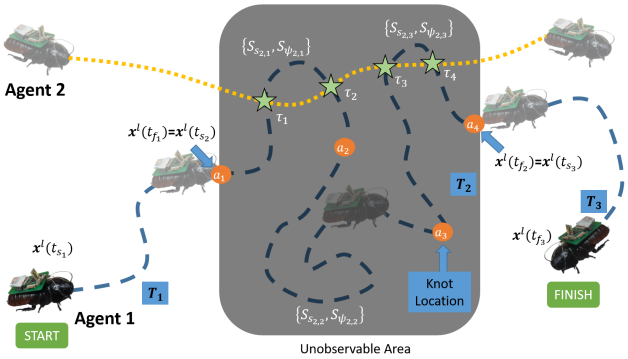

Fig. 1. Illustration of biobot trajectories: Agent 1's trajectory consists of three trajectory segments, denoted in the figure as $\mathbf{T}_1$, $\mathbf{T}_2$, and $\mathbf{T}_3$. Each trajectory segment contains a portion of the biobot's state trajectory, $\hat{\mathbf{x}}_i^l(t)$, and is defined over a time interval, $[t_{s_i}, t_{f_i}]$, where i denotes the i$^{\text{th}}$ trajectory segment. $\hat{\mathbf{x}}_i^l(t)$ is computed by perturbing the trajectory segment's speeds and headings via speed and heading perturbation splines, denoted as $S_{s_{i,j}}$ and $S_{\psi_{i,j}}$—respectively—where the subscripts denote the j$^{\text{th}}$ spline piece of the i$^{\text{th}}$ trajectory segment. The figure shows the perturbation splines associated with the second trajectory segment, where $\{a_1, a_2, a_3, a_4\}$ denote the times associated with the splines' knots. Agents 1 and 2 have four encounters, denoted by the green stars, and written as $\mathcal{E}_{12} = \{\tau_1, \tau_2, \tau_3, \tau_4\}$, where τ_k denotes the time of the k$^{\text{th}}$ encounter.

The remainder of the paper is organized as follows: Section II provides a brief summary of work pertaining to inertial navigation systems; Section III details the problem statement; Section IV describes the implementation of our multi-agent navigation system; Section V explains our experimental setup and Section VI analyzes the performance; and Section VII concludes the paper and discusses future work.

II. Related Work

Traditionally, position is estimated in an INS by extracting the linear acceleration of the IMU from its specific force readings, and performing a double integration on those linear accelerations. This causes any errors in the specific force (e.g. sensor noise) to grow quadratically with time, which makes it almost impossible to track position for more than a short period of time for all but the most precise and expensive IMUs. One way to get around this is to supplement IMUs with additional sensing modalities such

This work was funded by the National Science Foundation under award CNS 1239243 and ECCS 1554367.

[1]J. Cole, A. Bozkurt and E. Lobaton are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695, USA. E-mails: jacole@ncsu.edu, aybozkur@ncsu.edu and edgar.lobaton@ncsu.edu

as the global positioning system (GPS). Unfortunately, GPS signals are inaccessible in many enclosed environments, such as the rubble piles and under the collapsed buildings in our target application of urban search and rescue. As such, other techniques must be used to perform pose estimation using IMUs. One such technique that has been growing in popularity is to apply machine learning to IMUs to mitigate the effect of sensor noise. There are two main paradigms for incorporating machine learning: end-to-end frameworks and pseudo-measurements. The goal of an end-to-end framework (e.g. [6], [7]) is to use a learned model to directly estimate the pose of an agent using inertial data as the input. The pseudo-measurement approach (e.g. [8], [9]) uses the inertial data to train a model that is subsequently used as a measurement (e.g. a velocity estimator) in an existing navigation system such as a Kalman Filter. Some authors opt to use the learned model as a fail-safe in situations where the primary navigation system fails (e.g. [10], [11]). An extensive literature review of inertial navigation with machine learning can be found in [5].

In the absence of GPS, signals of opportunity can be used to increase localization accuracy (e.g. [12], [13]). Signals of opportunity are relative measurements to a signal source whose location can be estimated reliably. In multi-agent systems, the relative measurements between the agents can be used to achieve a desired action (e.g. formation-control) [14], [15], [16]. These relative measurements can also be used for joint localization of the agents in the system (e.g. [17]). Joint localization of agents is also explored extensively in the area of wireless sensor networks (e.g. [18], [19]), where the goal is to use relative measurements between sensors to localize the sensors and/or targets of interest.

We view the biobots as a multi-agent system, where the goal is to reduce biobot localization errors via ranging information that detects when the biobots encounter each other. The biobot encounters can be viewed as signals of opportunity that enforce consistency in the estimates of biobots' position.

III. Problem Statement

Similar to [5] and as illustrated in Fig. 1, we consider a scenario in which we need to search an area that is not easily accessible via conventional tools. A group of biobotic agents will explore the environment while collecting relevant sensor data. Once a biobot enters the target area, its pose is no longer observable; however, each biobot will eventually leave the target area, whereby its pose will once again be observable. Additionally, each biobot is equipped with a ranging sensor that can be used as a proximity sensor to detect nearby biobots. The goal is to reconstruct each biobot's trajectory using inertial data so that signals of interest can be localized.

This problem is split into two stages (see Fig. 2): Stage I aims to estimate each biobot's trajectory independently, and Stage II aims to refine the individual trajectories by leveraging ranging information from their encounters with each other. The first task was described in detail in [5], and we summarize that paper's problem setup in the proceeding

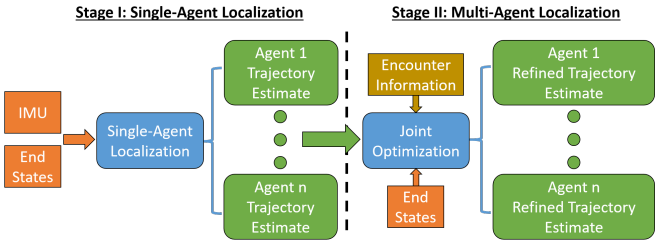

Fig. 2. Multi-agent navigation system pipeline: **Left**: In Stage I, a single-agent inertial navigation system is used for obtaining trajectory estimates for each of the biobots—details can be found in [5]. **Right**: In Stage II, the individual biobot trajectories are jointly optimized (using biobot encounter information) to obtain refined trajectory estimates that have lower error.

paragraph. Here, we assume that the agents are restricted to planar (2D) environments (see [5] for more details).

We define a biobot's state, $\mathbf{x}^l(t)$, to be its position $\mathbf{r}^l_{lb} = (x^l_{lb}, y^l_{lb})$, speed $s^l_{lb} = \sqrt{(\dot{x}^l_{lb})^2 + (\dot{y}^l_{lb})^2}$ and heading $\psi_{lb}(t)$. The notation $\mathbf{r}^l_{lb}$ means the position of the body frame b with respect to the local frame l, resolved in frame l. Frame l is a local tangent frame that uses Cartesian coordinates and used as the reference and resolving frames. Frame b is the body frame and centered on the IMU that is mounted to the body of the biobotic agent itself; the origin of frame b is $\mathbf{r}^l_{lb}$. We assume that the biobotic agent always moves in the direction that it is facing. Under this assumption, the biobot's velocity is given by $\mathbf{v}^l_{lb} = \dot{\mathbf{r}}^l_{lb} = (\dot{x}^l_{lb}, \dot{y}^l_{lb}) = (s^l_{lb} \cdot \cos(\psi_{lb}), s^l_{lb} \cdot \sin(\psi_{lb}))$. The biobot's true trajectory is a smooth mapping in $\mathbb{R}^4$, $\mathbf{x}^l(t) : [0, t_f] \rightarrow \mathbb{R}^4$, where t_f denotes the biobot's exit time. Our goal is to find a reconstruction of $\mathbf{x}^l$, $\hat{\mathbf{x}}^l(t;\theta) : [0, t_f] \rightarrow \mathbb{R}^4$, where θ denotes the set of parameters that govern $\hat{\mathbf{x}}^l$. The optimal parameters are obtained by minimizing the following cost function for each biobot:

$$\begin{aligned} J_i(\theta_i) &= \int_{t_{s_i}}^{t_{f_i}} ||\mathbf{x}^l_i(t) - \hat{\mathbf{x}}^l_i(t)||^2 \, \mathrm{dt} \\ s.t.\ \hat{\mathbf{x}}^l_i(t_{s_i}) &= \mathbf{x}^l_i(t_{s_i}) \, , \, \hat{\mathbf{x}}^l_i(t_{f_i}) = \mathbf{x}^l_i(t_{f_i}) \end{aligned} \tag{1}$$

subject to the boundary conditions, where i refers to the i^{th} biobot, and t_s and t_f denote the entry and exit times, respectively. The multi-agent extension of the navigation system can be achieved by adding the ranging information as constraints:

$$\begin{aligned} J_M(\Theta) &= \sum_i J_i(\theta_i) \\ s.t.\ \hat{\mathbf{x}}^l_i(t_{s_i}) &= \mathbf{x}^l_i(t_{s_i}) \, , \, \hat{\mathbf{x}}^l_i(t_{f_i}) = \mathbf{x}^l_i(t_{f_i}) \qquad \forall i \\ \mathcal{E}_{ij} &= \left\{ t \mid d(\hat{\mathbf{r}}^l_i(t), \hat{\mathbf{r}}^l_j(t)) \leq \delta \right\} \qquad \forall i, j \end{aligned} \tag{2}$$

subject to the boundary conditions, for each of the agents, where Θ is the set of all optimizable parameters for all of the biobots. $d(\cdot)$ is a function that returns the Euclidean distance between its two inputs—that is, biobots i and j—and δ refers to the sensing range of the proximity sensor. Finally, $\mathcal{E}_{ij} \coloneqq \left\{ t \mid d(\mathbf{r}^l_i(t), \mathbf{r}^l_j(t)) \leq \delta \right\}$ denotes the set of encounters between agents i and j and is defined to occur at the times

when the agents are, at most, δ-distance away from each other. In this paper, we assume that the proximity sensor is an ideal sensor that has perfect detection rate and zero false-positive rate. This idealized sensor is implemented using the ground truth position data for the biobots.

IV. Methodology

In Section IV-A, we summarize the navigation system (also described in [5]) since it is used for individually estimating each agent's trajectory. Afterwards, in Section IV-B, we describe the extension that is needed to jointly optimize the agent trajectories.

A. Single-Agent Localization

Each biobot has a thorax-mounted IMU that is capable of generating specific force and angular rate readings. These IMU signals are then windowed using a sliding-window approach and features are extracted from each window of IMU data. The features are then used to estimate the speed and heading of the biobot and these estimates are linearly interpolated to create continuous versions of themselves, denoted as $\hat{s}^l_{lb}(t)$ and $\hat{\psi}_{lb}(t)$—respectively.

The trajectories can include multiple areas where the agent's pose is unobservable. This is handled with the introduction of trajectory segments $\{\mathbf{T}_k\}$, where the k$^{\text{th}}$ trajectory segment is the portion of the agent's trajectory that occurs between the k$^{\text{th}}$ pair of entry/exit points. The agent state is computed by perturbing the interpolated speed/heading curves as follows:

$$\begin{aligned} \hat{s}^{l*}_{lb_k}(t) &= \hat{s}^l_{lb_k}(t) + S_{s_k}(t) \\ \hat{\psi}^*_{lb_k}(t) &= \hat{\psi}_{lb_k}(t) + S_{\psi_k}(t) \end{aligned} \tag{3}$$

where k denotes the trajectory segment, $\hat{s}^{l*}_{lb_k}$ is the perturbed speed, $\hat{\psi}^*_{lb_k}$ is the perturbed heading, and $S_{s_k}(t)$ and $S_{\psi_k}(t)$ denote the speed and heading perturbation splines, respectively. Each segment has its own perturbation splines, which are clamped piecewise-cubic splines. The perturbed speed and heading trajectories are then used to compute:

$$\begin{aligned} \hat{\mathbf{r}}^l_{lb_k}(t) &= \mathbf{r}^l_{lb_k}(t_{s_k}) + \int_{t_{s_i}}^{t} \hat{\mathbf{v}}^l_{lb_k}(t)\,\mathrm{dt} \\ \hat{\mathbf{v}}^l_{lb_k}(t) &= \left[\hat{s}^{l*}_{lb_k}(t)\cdot\cos\left(\hat{\psi}^*_{lb_k}\right),\ \hat{s}^{l*}_{lb_k}(t)\cdot\sin\left(\hat{\psi}^*_{lb_k}\right)\right]^\top \end{aligned} \tag{4}$$

The optimizable parameters of $\mathbf{T}_k$ are the coefficients of S_{s_k} and S_{ψ_k}. These coefficients can be determined by optimizing Equation 1 and adding its constraints as penalty terms—note that the constraint on the initial position, $\hat{\mathbf{x}}(t_s) = \mathbf{x}(t_s)$, is automatically satisfied because we assume that the agent's state is known at the entry point. The resulting cost function, $\tilde{J}$, is shown below for a single trajectory segment:

$$\begin{aligned} \tilde{J}(\theta) = \int_{t_s}^{t_f} &\left(W_s\cdot S_s(t;\theta_s)^2 + W_\psi\cdot S_\psi(t;\theta_\psi)^2\right)\mathrm{dt} \\ &+ W_r\cdot\left|\left|\mathbf{r}^l_{lb}(t_f) - \hat{\mathbf{r}}^l_{lb}(t_f;\theta)\right|\right|^2 \end{aligned} \tag{5}$$

where θ_s and θ_ψ denote the optimizable parameters for the speed and heading perturbation splines. The integrand is computed numerically using a sampling rate of 30 Hz, and W_s, W_ψ, and W_r are weights that are used as hyperparameters. We use trajectory segments of two-minute length and use the hyperparameter values detailed in [5]—namely, two-second knot spacing for the perturbation splines, $W_s = W_\psi = 1$, and $W_r = 120$.

B. Multi-Agent Extension

The multi-agent extension of Equation 5 that is used for jointly optimizing the agent trajectories can be achieved by reformulating the boundary conditions of Equation 2 that involve ranging measurements as a penalty terms. Each individual ranging measurement can be realized as a rectified linear unit (ReLU) [20], where $\mathrm{ReLU}(\cdot) = \max(0,\cdot)$. The ReLUs are used for defining a proximity cost that grows linearly whenever there are two agents that violate the distance requirement mandated by a particular encounter. This yields the surrogate cost function over a trajectory segment $\mathbf{T}_k$:

$$\begin{aligned} \tilde{J}_M(\Theta) &= \sum_i \tilde{J}_i(\theta_i) + W_p\cdot\sum_{i,j}\sum_{t\in\mathcal{E}_{ij}}\mathrm{ReLU}\left(\gamma_{ij}(t)\right)^2 \\ \gamma_{ij}(t) &= ||\hat{\mathbf{r}}^l_i(t) - \hat{\mathbf{r}}^l_j(t)|| - \delta \end{aligned} \tag{6}$$

where Θ denotes the optimization parameters for all of the agents that encounter one another in the k$^{\text{th}}$ trajectory segment; t is restricted to the k$^{\text{th}}$ trajectory segment, $t \in \mathbf{T}_k$; i and j denote the i$^{\text{th}}$ and j$^{\text{th}}$ agents, respectively; $\hat{\mathbf{r}}_i$ and $\hat{\mathbf{r}}_j$ denote the estimated positions of agents i and j, respectively; finally, W_p is the hyperparameter for the proximity cost and controls the extent to which the ranging measurements will influence the optimization process. We found that $W_p = 120$ yielded satisfactory results for our dataset, and we optimized Equation 6 using the *fminunc* function in MATLAB (The MathWorks Inc, Natick, MA, USA).

V. Experimental Setup

We used the same dataset that was recorded in [5]. Data was collected from an adult female Madagascar hissing cockroach (*Gromphadorhina portentosa*) using a MetaMotion C sensor board (Mbientlab Inc., San Francisco, CA, USA) that was attached to its thorax. This sensor board contained an IMU with a three-axial accelerometer and a three-axial gyroscope. The biobotic agent was placed in a circular arena with a diameter of 115 cm (see Fig. 3) and recorded using a webcam. Inertial and video data was collected for nine separate trials, each of which lasted for approximately 30 minutes. The recorded data was used for creating the biobot's ground truth trajectory via the video tracking algorithm discussed in [5].

We simulated the effect of having five agents by taking the last five trials and temporally aligning their IMU and video data so that they started at the same time. We generated the encounter information of these five agents by tracking their time-aligned ground truth positions and flagging the instances when there were two, or more, agents that came within a δ-distance of each other. This setup creates an idealized proximity sensor that has a sampling rate equal

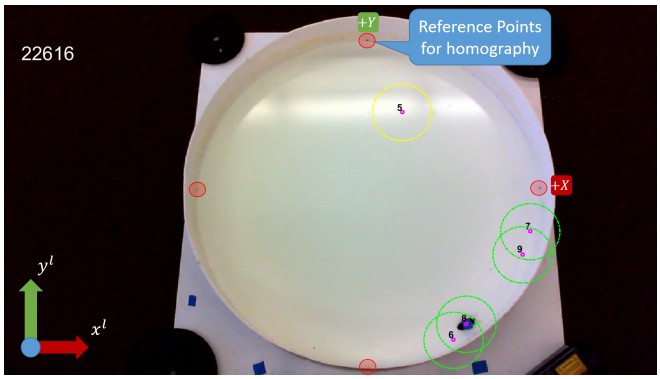

Fig. 3. Experimental setup: Planar (2D) circular arena with a 115 cm diameter. Local tangent reference frame, l, is aligned with the homography points (highlighted with **red circles**) that are used to convert points from image space (pixels) to physical space (cm). The time-aligned agents (5-9) are marked with **magenta squares** and overlaid on top of the video for the biobotic agent. Each agent's sensing range is shown with **yellow circles**, which turn into **green circles** when another agent comes within sensing range (δ= 10cm). The origin of frame l is located at the center of the circular arena. Figure adapted from [5].

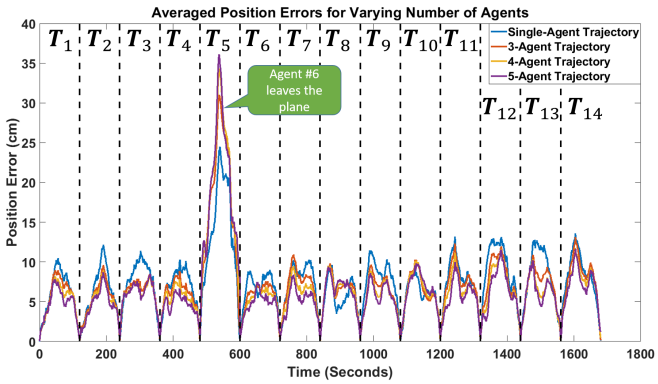

Fig. 4. Agent trajectory errors (varying number of agents): Position errors (cm) averaged across all agent subsets and all agent IDs (5-9) for varying number of agents. $\mathbf{T}_i$ denotes the i^{th} trajectory segment and the curves are defined as follows: **Blue**: averaged individual trajectories (using no speed scaling) are shown as a baseline; **Orange**: averaged curves for the 3-agent subsets; **Yellow**: averaged curves for the 4-agent subsets; **Purple**: averaged curves for the 5-agent subset.

to the video frame rate (30 Hz). The process for obtaining encounter information is shown in Fig. 3.

VI. Results and Discussion

We used the first four trials for training, and the last five trials for testing. Our analyses are based on a sensing range of $\delta = 10$cm. A video illustrating the trajectories and the results of our approach can be found online[1].

A. Agent Interaction Analysis

Table I shows the average position errors of the jointly-optimized agent trajectories in centimeters. For each specific number of agents, as specified in columns 2 through 4, there is a choice in which subset of agents to use (e.g. if three agents are to be used, then agent #5's trajectory could be optimized using agents $\{5,6,8\}$, $\{5,7,9\}$, etc.). We account for these choices by averaging the trajectory errors over all trajectory segments for all agent subsets. The "Single Agent" column refers to the average position error when the agents are individually optimized. Lastly, the final rows of the preceding tables list the column-wise averages, and the improvement $\left(1 - \frac{Error\_Multi\_Agent}{Error\_Single\_Agent}\right) \cdot 100\%$ of those averages over the single-agent case.

Table I shows that the multi-agent cost function described in Equation 6 can provide a noticeable reduction in the average position error of each agent, regardless of whether 3, 4, or 5 agents are used for the joint optimization —the "+ Scaling" column is discussed in Section VI-B. Our algorithm assumes that we have equal confidence in the accuracy of each agent's trajectory. A downside to this can be seen in the entries for agent #8 in Table I. In all cases, the joint optimization makes the average position error worse than its single-agent counterpart. This can be explained by the fact that agent #8 has the lowest trajectory segment errors of any of the five agents. As such, when other agents are used for "correcting" agent #8's trajectory, this actually worsens it. Further examples of this issue are shown in trajectory segment $\mathbf{T}_5$ (see Fig. 4), where agent #6 leaves the plane, which violates the 2D assumption made in Section III. As a result of this, agent #6 has an erroneous trajectory that worsens all agents that are jointly optimized with it—this causes all five agents to have erroneous trajectories, rather than just agent #6, which explains why the jointly optimized trajectories are noticeably worse than the averaged individual trajectories in this segment. A potential solution to this problem is to add a heuristic that can determine the confidence that should be placed in an agent's trajectory; however, this is the beyond the scope of presented work.

The runtimes for our multi-agent system were 2.8 hrs, 5.2 hrs and 8.4 hrs for the 3-agent, 4-agent, and 5-agent cases; respectively. All results were obtained using MATLAB 2018b on a desktop computer with the following specifications: Intel Core i7-7700k CPU, Nvidia GeForce GTX Titan X GPU, 64GB RAM, and 64-bit Ubuntu 16.04.6 LTS OS. For comparison, each single-agent optimized trajectory took approximately 26 min to generate.

TABLE I
Position Error (cm)

Agent ID	3-Agent	4-Agent	5-Agent	+ Scaling	Single Agent
5	7.58	7.13	6.82	4.97	9.23
6	7.84	7.24	6.64	6.00	9.62
7	5.55	5.35	5.15	4.99	6.10
8	6.51	6.60	6.57	6.06	6.07
9	6.66	6.59	6.42	5.59	6.73
Average	6.82	6.58	6.32	5.52	7.55
Improv.	9.62%	12.86%	16.29%	26.86%	N/A

[1] https://youtu.be/SCMKm8tUfXw

B. Discussion

As previously pointed out, there is flexibility in the choice of sensing range because the number of encounters has a small impact on the overall runtime of the navigation system. We found $\delta = 10$ cm to be an adequate value for this experiment. Performance could be further improved if a heuristic is added to determine the confidence that should be placed in an agent's trajectory. Such a heuristic could be used to mitigate the effects of erroneous agent trajectories (e.g. trajectory segment #5 in Fig. 4). This could even be used to improve those erroneous trajectories by providing a mechanism to increase the effect that other agents have on the erroneous agent(s). Performance could be further improved by using the biobot encounters to scale the estimated speeds of the biobot trajectories. An example of this is shown in the "+ Scaling" column of Table I where the speed scaling strategy (applied to the 5-agent scenario) consisted of three steps: First, the agent trajectories were evaluated at seven different speed scalings (80%-110%) using Equation 5 and the permutation of agent speed scalings (e.g. agent 1: 90%, agent 2: 105%, etc.), which minimized the proximity cost component of Equation 6, was determined—this type of speed scaling is identical to a constant scaling factor that is applied to the estimated speed of an entire trajectory segment. Second, the aforementioned method identified that agent 5 has abnormally low speed scaling. Third, a joint optimization was performed where agent 5's speed were scaled based on the permutation method's speed scalings, but no other agents had their speeds scaled. This simple strategy nearly doubled the performance of the 5-agent scenario from 16% to 27%. However, a full evaluation of this strategy would require a larger datasets with more agents for evaluation.

As a last point, we want to emphasize that our dataset was taken from a planar (2D) circular arena, and that further testing is needed to see how our multi-agent navigation system performs in more complex platforms similar to disaster environments.

VII. Conclusion and Future Work

In this paper, we have presented a multi-agent navigation system that jointly optimizes agent trajectories using proximity information and detects when agents encounter each other. Our multi-agent navigation system improves trajectory accuracy by refining trajectories that have been independently optimized. Future work could forgo the single-agent optimization as a preliminary step; instead, the multi-agent cost function could be directly applied to the non-optimized agent trajectories. Alternatively, future work could focus on the development of a heuristic for assessing the validity of an agent's trajectory estimate, or the incorporation of a speed scaling function. Finally, future work could also validate the proposed navigation system in more complex environments such as more realistic 3D environments with obstacles.

References

[1] T. Latif, "Tissue-electrode interface characterization for optimization of biobotic control of roach-bots." Ph.D. dissertation, North Carolina State University, 2016.

[2] A. van Casteren and J. R. Codd, "Foot morphology and substrate adhesion in the madagascan hissing cockroach, gromphadorhina portentosa," *Journal of insect science*, vol. 10, no. 1, p. 40, 2010.

[3] K. Jayaram and R. J. Full, "Cockroaches traverse crevices, crawl rapidly in confined spaces, and inspire a soft, legged robot," *Proceedings of the National Academy of Sciences*, vol. 113, no. 8, pp. E950–E957, 2016.

[4] T. Latif and A. Bozkurt, "Roach biobots: Toward reliability and optimization of control," *IEEE pulse*, vol. 8, no. 5, pp. 27–30, 2017.

[5] J. Cole, A. Bozkurt, and E. Lobaton, "Localization of biobotic insects using low-cost inertial measurement units," *Sensors*, vol. 20, no. 16, p. 4486, 2020.

[6] M. A. Esfahani, H. Wang, K. Wu, and S. Yuan, "Aboldeepio: A novel deep inertial odometry network for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[7] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[8] M. Brossard, A. Barrau, and S. Bonnabel, "Rins-w: Robust inertial navigation system on wheels," *arXiv preprint arXiv:1903.02210*, 2019.

[9] W. Liu, D. Caruso, E. Ilg, J. Dong, A. Mourikis, K. Daniilidis, V. Kumar, J. Engel, A. Valada, and T. Asfour, "Tlio: Tight learned inertial odometry," *IEEE Robotics and Automation Letters*, 2020.

[10] S. Adusumilli, D. Bhatt, H. Wang, V. Devabhaktuni, and P. Bhattacharya, "A novel hybrid approach utilizing principal component regression and random forest regression to bridge the period of gps outages," *Neurocomputing*, vol. 166, pp. 185–192, 2015.

[11] J. Li, N. Song, G. Yang, M. Li, and Q. Cai, "Improving positioning accuracy of vehicular navigation system during gps outages utilizing ensemble learning algorithm," *Information Fusion*, vol. 35, pp. 1–10, 2017.

[12] Z. Z. M. Kassas, M. Maaref, J. J. Morales, J. J. Khalife, and K. Shamei, "Robust vehicular localization and map matching in urban environments through imu, gnss, and cellular signals," *IEEE Intelligent Transportation Systems Magazine*, vol. 12, no. 3, pp. 36–52, 2020.

[13] C. Yang and A. Soloviev, "Mobile positioning with signals of opportunity in urban and urban canyon environments," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE, 2020, pp. 1043–1059.

[14] X. Ge, Q.-L. Han, D. Ding, X.-M. Zhang, and B. Ning, "A survey on recent advances in distributed sampled-data cooperative control of multi-agent systems," *Neurocomputing*, vol. 275, pp. 1684–1701, 2018.

[15] S. Knorn, Z. Chen, and R. H. Middleton, "Overview: Collective control of multiagent systems," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 4, pp. 334–347, 2015.

[16] L. Ding, Q.-L. Han, X. Ge, and X.-M. Zhang, "An overview of recent advances in event-triggered consensus of multiagent systems," *IEEE transactions on cybernetics*, vol. 48, no. 4, pp. 1110–1123, 2017.

[17] P. Zhu and W. Ren, "Multi-robot joint localization and target tracking with local sensing and communication," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 3261–3266.

[18] D. P. Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Information Fusion*, vol. 49, pp. 1–25, 2019.

[19] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications surveys & tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[20] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017.