GRAL: Localization of Floating Wireless Sensors in Pipe Networks

Martin Haug
Technische Universität Berlin
Berlin, Germany
m.haug@tu-berlin.de

Felix Lorenz*
Technische Universität Berlin
Berlin, Germany
felix.lorenz@tu-berlin.de

Lauritz Thamsen
Technische Universität Berlin
Berlin, Germany
lauritz.thamsen@tu-berlin.de

***Abstract*—Mobile wireless sensors are increasingly recognized as a valuable tool for monitoring critical infrastructures. An important use case is the discovery of leaks and inflows in pipe networks using a swarm of floating sensor nodes. While passively drifting along, the devices must track their individual positions so critical points can later be located. Since pipelines are often situated in inaccessible places, large portions of the network can be shielded from radio and satellite signals, rendering conventional positioning systems ineffective.**

In this paper, we propose a novel algorithm for assigning location estimates to recorded measurements once the sensor node leaves the inaccessible area and transmits them via a gateway. The solution is range-free and makes use of a priori information about the target pipeline network. We further describe two extended variants of our algorithm which use data of encounters with other sensor nodes to improve accuracy. Finally, we evaluate all variants with respect to various network topologies and different numbers of mobile nodes in a simulation. The results show that our algorithm localizes measurements with an average accuracy between 4.81% and 7.58%, depending on the variability of flow speed and the sparsity of reference points.

***Index Terms*—localization, mobile wireless sensors, pipeline monitoring, critical infrastructures**

I. Introduction

In order to operate and maintain a pipe network, utility operators must be able to investigate its current state and detect specific undesirable conditions such as leakages or inflows/infiltrations. However, with pipelines buried deep underground, direct access and large-scale deployment of stationary sensors for in vivo monitoring is often infeasible. There have been recent efforts to resolve this using autonomous mobile robots [1, 2] and Mobile Wireless Sensor Network (MWSN) [3–5]. A Mobile Wireless Sensor Network (MWSN) is a network of nodes that can move, detect changes in their environment and communicate wirelessly. In the special case of pipe monitoring, nodes can do entirely without active locomotion and instead float with the current of the contained liquid [6]. Thus, they can be produced at a fraction of the cost of actively moving robots and are at the same time more robust. Sensor nodes can be inserted into the system at likely points of interest. They may be recovered or disposed of at a wastewater treatment plant. For a comprehensive overview of the literature on MWSN for pipe inspection, see [7].

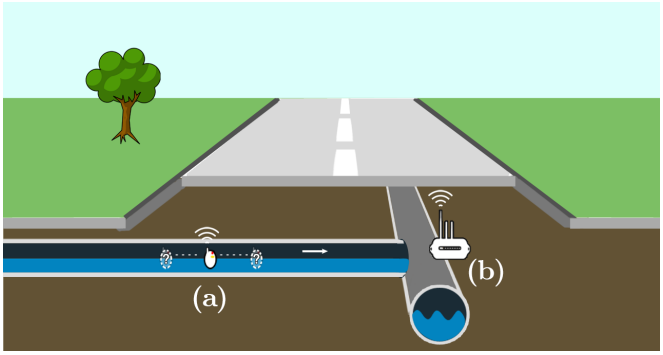


Fig. 1: The measurements taken by a sensor node (a) are submitted to the GRAL backend through a gateway (b) which resides at a junction in the pipe network.

The use of a big fleet of sensors to monitor a pipe network requires solutions for data collection and scalable analytics [8, 9]. However, to make use of the recorded data in the first place, the position in which the measurement was taken must be known. Satellite-based localization systems like GPS are a natural choice for localization but are often not applicable in MWSN due to power constraints or insufficient signal coverage. In such cases, one can instead make use of information on the proximity to other nodes and the properties of the environment to estimate the position of each reading.

In this paper, we address the localization problem of MWSNs deployed in wastewater networks. However, the approach can also be used in other pipeline-like environments like water networks or oil pipelines. We formalize the localization problem in Section II.We review existing localization algorithms for both Wireless Sensor Networks (WSNs) in general and MWSNs with floating nodes in Section III. For our work, we assume that the environment is tree-shaped, where nodes move along the edges from the leaves to the root. Gateways are installed at specific points

*Work done while at Technische Universität Berlin, now at ecospace

throughout the pipe network. We introduce the Graph-based Localization (GRAL) algorithm as a method for sensor localization in such situations (Section IV). The idea is to use MWSN gateways as anchors and interpolate the path between them, where nodes traverse areas without coverage. Reconstruction of measurement locations is done at a central backend, to which a node sends its recordings upon entering the radio range of a gateway. We analyze the performance of GRAL with several pipe network topologies using a simulation (Section V).

II. Problem Setup: Localization in Pipe Networks

Applications face the challenge of tracking the location of a sensor node while it is moving through a pipe networks. For our solution to this, we consider networks that satisfy the following requirements:

- The topology of the network can be described as
  1) *Junctions* that may feature stationary gateways with uplink connections
  2) *Links* of a known length connecting the junctions
- This topology is fixed and can be modeled as a tree-shaped graph in which the vertices correspond to junctions and the edges represent links
- The stationary gateways' approximate wireless radius and positions are known a priori
- Sensor units record relevant physical quantities such as conductivity or temperature while floating through the pipes
- Every time a sensor unit passes a gateway, it will send a batch of measurements to the backend
- The liquid flow direction in the network is never reversed. The sensor nodes thus always travel from the leaves to the root

This leads us to the *measurement localization problem* that is addressed in this paper:

Given a series of measurements from a mobile wireless sensor node floating through a tree-shaped pipe network, determine the location in that network at which each reading was taken.

III. Related Work

Classical approaches to WSN localization in arbitrary environments can be divided into two classes: *Range-based* localization schemes use observations of *anchors* with known positions to estimate distances whereas *range-free* algorithms which only rely on the binary state of connectivity [10]. *Lateration* is a classic example for a range-based localization method. It relies on the radio signal strength indicator (RSSI), together with a model for radio wave propagation [11]. Another range-based method relies on the *time difference of arrival* between a radio pulse and an ultrasonic pulse to estimate the distance to the anchors [12]. Most range-based approaches are not easily applicable for usage in MWSN due to the high number of anchors required, poor calibration of cheap radio devices, and the presence of obstacles [13].

In networks with dense anchor placement, reasonable results can already be achieved by simply using the centroid of all visible reference nodes [14], a simple range-free approach. A graph-based solution is described in [15], where a fold-free graph embedding is found, upon which mass-spring based optimization is applied to localize a node. Again, the problem with respect to the applicability of classical range-free methods in pipe networks is that a high gateway density can rarely be guaranteed in practice.

Both kinds of approaches have been used in MWSN monitoring with floating sensors: Range-free localization can be accomplished by applying RFID beacons to the outside of pipes [16]. The SewerSnort system uses a custom model of radio propagation in pipes to provide ranged localization with fewer, if more expensive, anchors [17]. Pipe networks allow for less conventional ranged approaches using the propagation characteristics of ultrasonic sound [18] or electromagnetic waves [19], which, however, require dedicated hardware.

Another idea is to use visual simultaneous localization and mapping (VSLAM), i.e., rely on camera images of the pipes' interior to track the device's location [20]. Here, the main drawback is that the nodes either have to transmit large amounts of data or do expensive computations involved in VSLAM.

For a more comprehensive overview, consult [21], a dedicated survey of localization solutions for pipe monitoring. Another approach to localize leaks and inflows into water grids may be to forgo WSN entirely and to use methods like distributed temperature sensing or gas injection instead [22].

IV. The GRAL Algorithm

The graph-based localization algorithm solves the Measurement Localization Problem from Section II by assigning positions to the individual measurement packages. For this class of applications, the position of the nodes along the length of the pipes are of interest, therefore a position can be expressed as a four-tuple. This tuple π contains two junctions in the pipe network and the offset on the shortest path between them which qualifies a node's position.

Our algorithm is based on the assumption that we know the structure of the pipe network the nodes are deployed in and, at various moments, their positions. The most obvious example, for the scenario laid out above, is that when the sensors receive the signal of a gateway with maximum strength, they are right below it. GRAL exploits this series of moments with known positions: It localizes a time-series of node measurements (called packages) by partitioning them into "epochs", each of which have a known final position. The Graph-based localization algorithm combines this information with prior knowledge of the topology of the deployment environment, encoded in a weighted, tree-shaped graph *(environment graph)*.

The fact that the signal of the gateway may be received only within a certain radius provides another positional anchor: We can tell that a node is at the boundaries of the circle this radius spans around the gateway the moment it stops or starts receiving the signal. Note that this could yield more than one possible position.

Based on the information of previous packages, we can see at which positions within the graph a node has been before and select one position for the package. Suppose that a node starts receiving a given gateway's signal, this gateway g_2 is located at a junction in the network where three pipes meet. The node is then localized to be in the pipe that is on the shortest path between the last gateway g_1 it has been at and g_2, exactly g_2's radius removed from the junction.

Below, we have to formalize the various types of epochs that are created in the application above to see when existing epochs learn about their final positions, can be localized and when new epochs need to be created. We need to introduce some notation for this: J is the maximum index of the packages for an epoch, s_i^j is the jth-strongest signal strength of a gateway in package i, and g_i^j is the gateway with the jth-strongest signal in package i. G^1, finally, is the set of gateways with the strongest signals for all packages of an epoch.

GRAL has three epoch types:

- ν epochs where no gateway's signal is received
- α epochs in which the strength of the strongest received gateway signal increases with every package
- ω epochs in which the strength of the strongest received gateway signal decreases with every package

At the most basic, a new package requires a new epoch if it does not fit the type condition for the latest existing epoch.

$$\tau(P) \coloneqq \begin{cases} \nu, & \text{if } \forall\ j = 1..J \quad G_j = \emptyset \\ \alpha, & \text{if } \forall\ j = 1..J \quad s_j^1 > s_{j-1}^1 \wedge g_j^1 = g_{j-1}^1 \\ \omega, & \text{if } \forall\ j = 1..J \quad s_j^1 \leq s_{j-1}^1 \wedge g_j^1 = g_{j-1}^1 \end{cases} \quad (1)$$

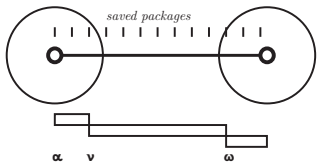


Fig. 2: Epochs created for a single node moving from a gateway to another

Because GRAL interpolates the packages' positions within an epoch between a known starting and final position along the shortest route between those in the environment graph, both of these boundary positions have to be known so that the packages can be localized, i.e. the epoch is *complete*.

There is a set of requirements of which an epoch must fulfill one to have a known final position π_f:

Data: A node's package p and a set of epochs E for that node
Result: The updated set of epochs E annotated with the correct types τ

$P_{\text{last}} \leftarrow P(\epsilon_{|E|})$; `// packages in last epoch`
`/* determine package type as Eq.(1) */`
if $\tau(P_{last} \cup p) \in \{\alpha, \omega, \nu\}$ **then**
  $P_{\text{last}} \leftarrow P_{\text{last}} \cup p$;
else
  `/* coalesce epochs if applicable */`
  if $\exists i \in \mathbb{N}. \forall j \in \mathbb{N}.\ j > i \wedge \tau(P_i) \neq \nu \wedge \tau(P_j) = \nu$ **and** $g^1(P_1(E_i)) = g^1(p)$ **then**
    coalesce $\{p\}$ and $\{P(e) | e \in \epsilon_i \dots \epsilon_{|E|}\}$;
  else
    `/* otherwise, create new epoch */`
    $E \leftarrow E \cup (\tau(\{p\}), —, p)$;
  end
end
return E;

Algorithm 1: Integrating a new package with the epoch set, $P(i)$ returns the packages in the ith epoch

- It is of the type α and it is not the last epoch (π_f is the position of g_i^1)
- It is of the type α and the RSSI of the gateway is maximal in its last package (π_f is the position of g_i^1)
- There is a subsequent epoch that is not of the type ν (π_f is the appropriate position at the border of g_{i+1}^1)
- The final position π_f is pre-set

In order to be *complete*, an epoch additionally needs a starting position which can either be pre-set or be derived from the final position of the preceding epoch (which must therefore also fulfill at least one of the above conditions).

For resiliency against positional fluctuations, all epochs at a specific gateway g after the first α epoch are merged as long as no other gateway is seen. GRAL is also fault-tolerant: If a gateway is not operable, the algorithm will just interpolate package positions with less epochs using the surrounding gateways.

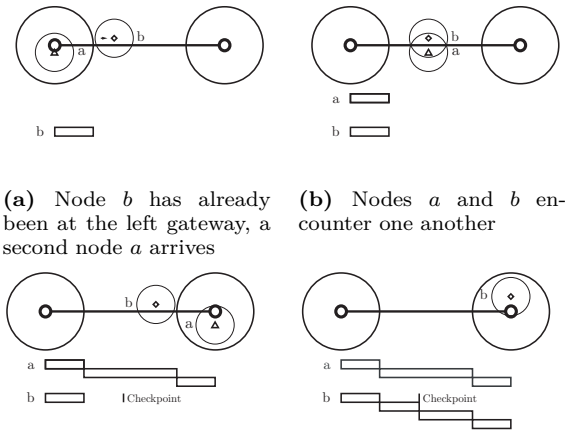
We consider GRAL to be range-free since it does not use the RSSI to measure a distance; instead, it uses preloaded data about the environment to derive distances. The RSSI, which the implementor can substitute for any other proximity indicator, is only used to determine the qualitative difference of increasing or decreasing proximity.

A. Algorithm Extensions

In networks with sparse gateway deployment, epochs of the ν type containing many packages will be created. GRAL is especially designed for use in non-pressurized wastewater systems with varying flow speeds. Because interpolation error due to these speed differences accumulates over time, localization accuracy can be low in such environments. To alleviate that issue, we propose

two algorithmic extensions, which use information about nearby other sensor nodes to refine epochs improve position estimation.

Checkpointing If there is more than one mobile node in the system, the nodes may meet. Data about these encounters can be used to make sure that both nodes get assigned a similar position for the moments where they have met. Checkpoints are a way to accomplish this. When processing an epoch's packages, a checkpoint containing a position, a timestamp and the identifier of the issuing node is added any encountered nodes. A node can only add a single checkpoint to each other node per epoch – the algorithm selects the wireless contact with the highest RSSI. When the encountered node reaches a gateway and starts to transmit its packages, GRAL creates a new epoch for the packages with a timestamp later than that of the checkpoint. The checkpoint's location is set as the position that was calculated for the node first to arrive at a gateway. Fig. 3 illustrates the process.



(a) Node b has already been at the left gateway, a second node a arrives

(b) Nodes a and b encounter one another

(c) Node a reaches the gateway, a checkpoint for b is created

(d) Node b reaches the gateway, its ν epoch is split at the checkpoint

Fig. 3: Checkpoint creation on a single link

Path rectification An encounter between two nodes can also sometimes place a lower bound on their positional estimates: Imagine two nodes approaching a junction with three pipes, like in Fig. 4c. The nodes start on the left and had contact to respective left gateways before. It is clear that, when the two nodes meet, they have to be situated behind the junction where they entered a shared path, because an encounter has not been possible on the disjunct paths they have been on before.

$\text{path}_v(v, w)$ returns the vertices on the path from v (inclusive) to w (exclusive) in the environment graph.

$$\begin{aligned}\exists v_c \in V. \neg\exists v_n \in V.\, & v_c \neq v_n \wedge v_c \in \text{path}_v(v_a, v_f) \wedge \\ & v_c \in \text{path}_v(v_b, v_f) \wedge v_n \in \text{path}_v(v_a, v_f) \wedge \\ & v_n \in \text{path}_v(v_b, v_f) \wedge \\ & |\text{path}_v(v_c, v_f)| < |\text{path}_v(v_n, v_f)| \end{aligned} \quad (2)$$

If packages for a node a coming from the position corresponding to vertex v_a in the environment graph indicate that it has seen another node b (coming from the corresponding position of vertex v_b), therefore, their *confluence vertex* is calculated.

The confluence vertex v_c is as defined above as the earliest vertex that is contained both in the paths from v_a and v_b to a shared destination v_f. Once the relevant epochs get completed, localization is first performed normally. If the results of this process for a estimate packages recording contact to b to be located before the confluence vertex, the relevant epochs are split at the earliest package recording this contact with the location of the confluence vertex as the final position π_f of the first of these two epochs, the packages are then re-localized.

V. Evaluation

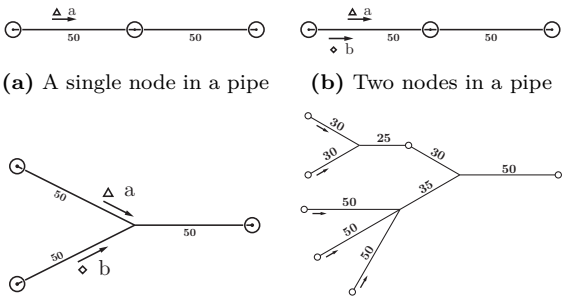
To evaluate the performance of GRAL under varying conditions, we generate several pipe network graphs through which simulated sensor nodes move, sense, and communicate. For this, we created a custom simulation environment.

A. Simulation

At each step of the simulation, all mobile node positions are moved a fixed amount towards the root junction, representing a base current within the pipes. A noise term sampled from a boolean distribution with $P(1) = \frac{2}{3}$ is added to this movement. Random fluctuations in flow speed are to be expected in real situations, due to obstructions and a non-uniform current along the cross-section of the pipe [23]. The simulation uses a fixed radio propagation model to calculate RSSI-like strength indicators for the connections. Just like a physical system, it occasionally emits batches of measurement packages to be localized by GRAL. We compare our solution with and without the described extensions against a naive approach (baseline) which consists of simple linear interpolation between gateways based on the first- and last-contact timestamps. A reference implementation of GRAL is available at `https://github.com/reknih/GRAL`.

B. Experiments

In total, our experiments cover four scenarios (overview in Fig. 4) for which 200 randomized runs (*instances*) are performed. The test scenarios were constructed to be representative of possible topologies that sensor nodes might be deployed in in real wastewater systems. Scenarios 4c and 4d include merging branches, which allow evaluation of the path rectification feature. Deviations of the calculated locations from the ground truth are given as Root Mean Square Error for both each instance *(iRMSE)* and the entire respective scenario *(dRMSE)*. We additionally use the Mean Absolute Error (MAE) to compute average error ranges in meters for real deployments.



(a) A single node in a pipe **(b)** Two nodes in a pipe

(c) Two nodes in an environment with a confluence **(d)** Complex environment

Fig. 4: Overview of the test scenarios; to scale circles representing gateway ranges

TABLE I: Evaluation results. Values correspond to RMSE over 200 instances. CP = Checkpointing, PR = Path rectification

	Baseline	GRAL			
		Vanilla	CP	PR	CP & PR
Scenario 1	9.54	**6.41**	-	-	-
Scenario 2	9.13	**6.5**	6.83	-	-
Scenario 3	11.64	10.03	9.9	10.02	**9.89**
Scenario 4	11.55	10.5	11.72	**10.48**	11.71
Total	10.1	**8.36**	-	-	-

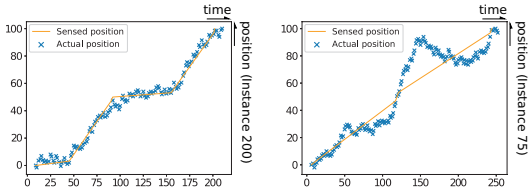
C. Results

The results of our experimental evaluation are summarized in Table I. GRAL consistently scores a lower RMSE than the baseline. In Scenario 3, the best accuracy is achieved by enabling both extensions, thus demonstrating the benefit of checkpoints in a relatively well-covered environment. Throughout the following, we give a detailed discussion of the success and failure cases and project the results onto deployments in real pipe networks.

Scenario 1 In the first scenario, the environment consists of three gateways, connected by two pipe segments and is populated with a single node (Fig. 4a). GRAL without extra features achieves a total error of dRMSE = 6.41, compared to the baseline error of $\text{dRMSE}_B = 9.54$. In this scenario, the MAE is 4.81 over 100 distance units. This means that one can expect GRAL to have an average error of 4.81% of the total route length for a localized point in this scenario, given that the ratio of pipe in range of a gateway equals $\frac{\sqrt{10}}{25}$.

Fig. 5 allows a closer look at instances with low and high errors in the first scenario, respectively. In 5a, we see that the node briefly stays in the vicinity of the first and second gateway respectively, creating prolonged and flat ω epochs. In contrast, Instance 5b shows the node to be drifting back and forth in the segment between two gateways, causing a large divergence from the interpolated values for that epoch.

Scenario 2 The next scenario as seen in Fig. 4b has the same layout but with two nodes deployed in close succession. Thus, the two are frequently within range of each other. Fig. 6 shows the distribution of iRMSE for localization with and without checkpoints respectively. The total error for the former is $\text{dRMSE}_C = 6.83$, for the latter it is dRMSE = 6.5. With the same assumptions as above, this would mean an average localization error of 5.22% for checkpoints enabled and 4.98% for checkpoints disabled.



(a) iRMSE = 3.174 **(b)** iRMSE = 10.7

Fig. 5: Instances with low or high iRMSE from the first dataset

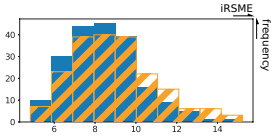
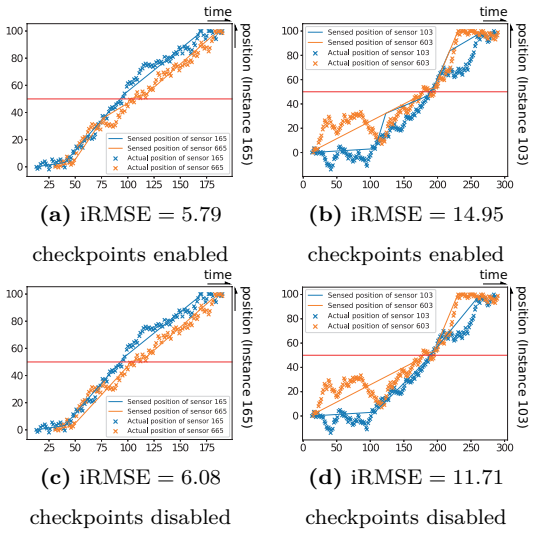


Fig. 6: Distribution of iRMSE for a dataset with two nodes and no junctions; checkpoints enabled for striped bars

The decrease in performance with checkpointing enabled is a rather surprising result. To see what has gone wrong, we again consider two exemplary instances in Fig. 7: Node 165 (blue) in 7a moves steadily, thus experiences small localization errors. The speed of node 665 (orange) fluctuates between gateways. Because node 165 is the first to arrive at a gateway, its relatively good localization can help improve the data for node 665. Conversely, in 7d we again see one node (103; blue) moving steadily and one (603; orange) experiencing turbulence. In this instance, the turbulent node 603 is the first to arrive at a gateway after it met its counterpart. Thus, it passes on its big localization error to node 103.

We see here that checkpointing, while having a sound theoretical justification, can cause the propagation of errors from one node to another in certain cases.

Scenario 3 This scenario introduces a new topology, as displayed in Fig. 4c. In this case, two initial gateways each serve as starting points for a node. They are both connected to the same junction with pipes of equal length. This center junction lacks a gateway but has another pipe leading to a last gateway. In this scenario, path rectification becomes relevant as nodes may encounter other nodes that did not share all of their path before moving on to a shared gateway. The results show that again vanilla GRAL (dRMSE = 10.03) outperforms the baseline algorithm ($\text{dRMSE}_B = 11.64$). The total error is higher than in the previous datasets because of the



(a) iRMSE = 5.79 checkpoints enabled

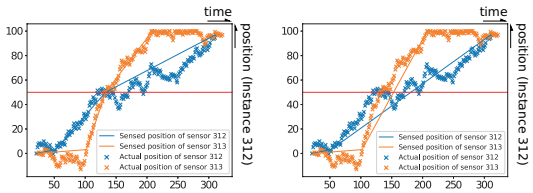
(b) iRMSE = 14.95 checkpoints enabled

(c) iRMSE = 6.08 checkpoints disabled

(d) iRMSE = 11.71 checkpoints disabled

Fig. 7: Instances with low or high iRMSE from the second dataset

uncertainty introduced by the missing center gateway. Interestingly, checkpoints seem to increase overall accuracy in this dataset ($\text{dRMSE}_C = 9.90$). The total error decreases slightly with rectification enabled ($\text{dRMSE}_P = 10.02$). Thus, simultaneously using both features, also increases localization accuracy ($\text{dRMSE}_{CP} = 9.89$). This means, that if the fraction of each path covered by a gateway is $\frac{\sqrt{10}}{50}$, the mean localization error for the worst configuration (vanilla) is 7.62%. The best configuration with checkpoints and path rectification would have an mean localization error of 7.58%.



(a) iRMSE = 9.685 rectification enabled

(b) iRMSE = 11.889 rectification disabled

Fig. 8: Comparison between enabled and disabled path rectification

Finally, we examine an instance in which path rectification is beneficial: In Fig. 8, two nodes have different speeds before and after the junction where they first meet (red line). They also, importantly, encounter each other before the final gateway. At the confluence point, new epochs are created for both nodes, permitting more accurate localization. Path rectification only moves the position up to the first vertex where the other node could have possibly been encountered; therefore, it is not possible to 'overshoot' the correct position.

Scenario 4 In this final scenario (Fig. 4d), we implemented a more realistic environment with longer paths from source to sink. There are five nodes in this scenario. The experiments show that GRAL with path rectification performs the best here ($\text{dRMSE}_P = 10.48$) while using checkpoints ($\text{dRMSE}_C = 11.72$) is outperformed by even the baseline algorithm ($\text{dRMSE}_B = 11.55$). This demonstrates that, while checkpoints may be beneficial in environments with good gateway coverage, they may severely impact the result if gateways are sparsely placed. The average localization error for the best configuration is 7.76%.

VI. Conclusion

In this paper, we introduce the GRAL algorithm for floating wireless sensor networks in pipe networks with a tree-like structure and sparse gateway coverage. Our solution does not need auxiliary positioning systems like GPS, nor does it burden the sensors with any additional computation. Instead, location estimates are computed for every measurement package once the node encounters a gateway and transmits them to a centralized backend. In contrast to other range-free approaches, GRAL can deliver location estimates for moments where a node has no contact to any other node. An important use case for our system is in vivo monitoring campaigns in pipe networks to detect leakages and infiltrations.

We evaluate GRAL and the two proposed extensions in a simulated environment with noisy node movement. As a baseline for comparison, we use simple linear interpolation between the encountered gateways. The results show that GRAL consistently outperforms the baseline and achieves a measurement localization accuracy between 4.81% and 7.76%, depending on the distance between consecutive gateways in the network and flow speed variability. Using two extensions, *checkpointing* and *path rectification* in settings with multiple nodes, a sparse deployment of gateways, and a frequently confluenting pipe network, can improve estimation accuracy (RMSE = 9.89) over vanilla GRAL (RMSE = 10.03). The advantage of *path rectification* increases for networks with more junctions and longer pipe segments.

In the future, we plan to use floating sensor node prototypes to assess how well our solution is working in the real world. It would also be promising to extend the algorithm further by taking flow characteristics of different parts of the pipe network (e.g. pipe width, materials) into account to further segment and adjust the position predictions.

Acknowledgments

This work has been supported through grants by the German Ministry for Education and Research (BMBF) as WaterGridSense 4.0 (funding mark 02WIK1475D).

References

[1] A. A. Nassiraei, Y. Kawamura, A. Ahrary, *et al.*, "Concept and design of a fully autonomous sewer pipe inspection mobile robot 'KANTARO'," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, IEEE, 2007, pp. 136–143.

[2] O. Tătar, D. Mandru, and I. Ardelean, "Development of mobile minirobots for in pipe inspection tasks.," *Mechanika*, vol. 68, no. 6, 2007.

[3] T.-t. T. Lai, Y.-h. T. Chen, P. Huang, *et al.*, "Pipeprobe: A mobile sensor droplet for mapping hidden pipeline," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, ACM, 2010, pp. 113–126.

[4] J.-H. Kim, G. Sharma, N. Boudriga, *et al.*, "Spamms: A sensor-based pipeline autonomous monitoring and maintenance system," in *2010 Second International Conference on Communication Systems and Networks (COMSNETS 2010)*, IEEE, 2010, pp. 1–10.

[5] Z. Sun, P. Wang, M. C. Vuran, *et al.*, "Mise-pipe: Magnetic induction-based wireless sensor networks for underground pipeline monitoring," *Ad Hoc Networks*, vol. 9, no. 3, pp. 218–227, 2011.

[6] S. Ishihara and D. Sato, "Active node selection in flowing wireless sensor networks," *Proc. 6th ICMU*, pp. 8–15, 2012.

[7] M. S. BenSaleh, S. M. Qasim, A. M. Obeid, *et al.*, "A review on wireless sensor network for water pipeline monitoring applications," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*, IEEE, 2013, pp. 128–131.

[8] F. Lorenz, M. Geldenhuys, H. Sommer, *et al.*, "A scalable and dependable data analytics platform for water infrastructure monitoring," in *IEEE International Conference on Big Data*, IEEE, 2020, pp. 3488–3493.

[9] M. Geldenhuys, J. Will, B. Pfister, *et al.*, "Dependable iot data stream processing for monitoring and control of urban infrastructures," in *1st International Workshop on Testing Distributed Internet of Things Systems*, IEEE, 2021.

[10] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[11] T. S. Rappaport, *Wireless communications: Principles and practice*, 2nd ed., ser. Prentice Hall communications engineering and emerging technologies series. Upper Saddle River, NJ, USA: Prentice Hall, 2002, 707 pp.

[12] B. Krishnamachari, *Networking Wireless Sensors*. New York, NY, USA: Cambridge University Press, 2005, 202 pp.

[13] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*, 4th ed. West Sussex, UK: John Wiley & Sons, 2007, 497 pp.

[14] S. P. Singh and S. C. Sharma, "Range free localization techniques in wireless sensor networks: A review," *Procedia Computer Science*, vol. 57, no. 1, pp. 7–16, 2015.

[15] N. Priyantha, H. Balakrishnan, E. Demaine, *et al.*, "Anchor-free distributed localization in sensor networks," MIT Laboratory for Computer Science, Cambridge, MA, USA, Tech Report 892, 2003-04, p. 13.

[16] A. S. Almazyad, Y. M. Seddiq, A. M. Alotaibi, *et al.*, "A proposed scalable design and simulation of wireless sensor network-based long-distance water pipeline leakage monitoring system," *Sensors*, vol. 14, no. 2, pp. 3557–3577, 2014.

[17] J. Kim, J. S. Lim, J. Friedman, *et al.*, "Sewersnort: A drifting sensor for in-situ sewer gas monitoring," in *2009 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, IEEE, 2009, pp. 1–9.

[18] Y. Bando, H. Suhara, M. Tanaka, *et al.*, "Sound-based online localization for an in-pipe snake robot," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE, 2016, pp. 207–213.

[19] T. Seco, C. Rizzo, J. Espelosín, *et al.*, "A robot localization system based on rf fadings using particle filters inside pipes," in *2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, IEEE, 2016, pp. 28–34.

[20] D. Krys and H. Najjaran, "Development of visual simultaneous localization and mapping (vslam) for a pipe inspection robot," in *2007 International Symposium on Computational Intelligence in Robotics and Automation*, IEEE, 2007, pp. 344–349.

[21] M. Z. Abbas, K. A. Baker, M. Ayaz, *et al.*, "Key factors involved in pipeline monitoring techniques using robots and wsns: Comprehensive survey," *Journal of Pipeline Systems Engineering and Practice*, vol. 9, no. 2, 2018.

[22] K. B. Adedeji, Y. Hamam, B. T. Abe, *et al.*, "Towards achieving a reliable leakage detection and localization algorithm for application in water piping networks: An overview," *IEEE Access*, vol. 5, pp. 20 272–20 285, 2017.

[23] H. Haug, *Statistische Physik*, 2nd ed., 1 vols. Berlin, Heidelberg, DE: Springer, 2006, vol. 1, 376 pp.