Pushing IoT Mobility Management to the Edge: Granting RPL Accurate Localization and Routing

Iliar Rabet*, Shunmuga Priyan Selvaraju *, Mohammad Hassan Adeli⋆, Hossein Fotouhi *
Ali Balador *, Maryam Vahabi *, Mário Alves†, Mats Björkman*
†*Politécnico do Porto, (ISEP)* Portugal
Email: *{iliar.rabet, shunmuga.selvaraju, hossein.fotouhi, ali.balador, maryam.vahabi, mats.bjorkman}@mdh.se,
⋆h.adeli@eng.uk.ac.ir, †mjf@isep.ipp.pt

***Abstract*—Accurate and timely mobility support in Internet of Things (IoT) applications is a challenging issue, considering the inherent scarce resources of IoT devices. However, the computational, memory and communication burden may be pushed into more "muscled" *Software Defined Network* (SDN) controllers. A centralised controller can exploit its global view of the network and predict the handovers and update the routing tables just in time to keep the mobile nodes connected. Although it is required to design mechanisms to enhance the mobility solution with extra link quality information to estimate the distance between mobile and static nodes and to avoid collision between the extra packets for localization and other communications in the network. In this work we present SDMob, an SDN-based mobility management architecture that lifts the burden of computation intensive filtering algorithms from resource constrained nodes and achieves accurate and fast handovers upon nodes' mobility under RPL/6LoWPAN (*Routing Protocol for Lossy Low-power Networks*, *IPv6 over Low-Power Wireless Personal Area Networks*). We show that SDMob improves the baseline RPL and the state-of-the-art mRPL in terms of packet delivery ratio leveraging more reliable routing. Applying *Particle* filter and variations of *Kalman* filter on radio signal strength data enables more accurate localization for complex real world trajectories.**

***Index Terms*—Internet of Things, Software Defined Networking (SDN), Mobility management, Localization, Kalman filter, Particle filter, Contiki, COOJA, Linux, RPL, 6LoWPAN.**

I. Introduction

There is an increasing demand for mobility support in Internet of Things (IoT) applications such as in healthcare, industrial automation and environmental monitoring. Nevertheless, the *de facto* protocol stack for low power and lossy networks (LLN) - RPL/6LoWPAN have not been designed for coping with a dynamic toplogy. In fact, they cannot provide a timely and accurate response to constant and fast topological changes in the network.

It has been shown that different mobility models affect the behavior of RPL in distinct ways [1], but it is believed that the baseline RPL protocol has proven to degrade quality-of-service upon mobility [2]. In the standard RPL, the Trickle algorithm is responsible for adapting the transmission frequency of control packets to the rate of changes in the topology. Increasing control packet's rate could result in better resiliency against mobility, but at the cost of higher communication and energy overheads. Nevertheless, since no predictive measure is taken, mobile node routes are only updated after disconnection, leading to network inaccessibility periods that will cause packet loss/delay.

A proactive approach to support seamless handoff could rely on Bayesian filters (such as a Kalman filters) or other predictive data processing techniques to forecast the future localization of mobile nodes. Here, a filter is referred to the methods that estimate the state of a temporal variable, which is usually observed under noisy measurements [3].

It is common to have *a priori* knowledge of the number of static nodes (in fixed positions) and the mobile nodes being assisted by an *Inertial Measurement Unit* (IMU). In such a scenario, it is practical to exploit statistical Bayesian prediction models (like Kalman and Particle filters) to fuse these two sources of information and, thereby, benefit from an accurate localization which leads to improvement in network responsiveness.

Kalman filter is proved to be unbiased (average error across all the recursive runs is zero), consistent (filter is neither overconfident nor under-confident) and optimal (minimum estimation error) [3]. However, in Kalman filter the posterior distribution (after the observations) can be computed in closed form only when the relation between states and observations is linear and the measurement and prediction noise follow a Gaussian distribution [4]. To tackle the nonlinear system models, *Extended Kalman Filters* (EKF) may be preferred; they use *Taylor series* to linearize the equations, trading for a negligible approximation error. On the other hand, Unscented Kalman Filter (UKF) and Particle filters have shown higher accuracy in prediction with bi-modal distribution of the error [3].

Implementing accurate predictive models may require higher computation capacity than mainstream IoT devices can afford. Resource-constrained nodes can hardly support lightweight filters (such as Kalman filters), provided that the position of the static nodes is hardcoded (in the mobile node). These limitations can be alleviated by offloading the computation burden to some external entity, such as a Software Defined Networking (SDN) controller. This can raise many challenges since the extra control packets between the SDN controller and the nodes lead to an extra traffic load. In this paper, we propose SDMob, an SDN-based mobility management architecture that relies on simple yet accurate localization mechanisms running in the SDN controller.

The main contributions of this work are listed below:

- Design of an SDN-based mobility management architecture –dubbed as SDMob– for seamless, reliable and

This work was supported by the Swedish Foundation for Strategic Research via the FiC project, and by the Swedish Research Council (Vetenskapsrådet), through the MobiFog starting grant, and by the Swedish Knowledge Foundation (KKS) through the FlexiHealth Prospekt, and the EU Celtic_Plus/Vinnova project, Health5G (Future eHealth powered by 5G).

timely mobility support.
- Implementation and fine tuning two filters (Particle filter and UKF) to determine mobile node position within a non-linear trajectory to enhance predictive routing.
- Implementation, integration and evaluation of the SDMob architecture into the RPL/6LoWPAN protocol, over a Contiki/COOJA + Linux ecosystem, comparing it against a benchmark non-SDN-based mobility solution (mRPL [5]).

This paper builds on our previous work [6], where we provided an analytical model of the proactive handoff mechanism using a Particle Filter. The model demonstrated how the expected probability of packet loss decreases with the seamless handoff managed by the controller.

The rest of the paper is organized as follows: Section II provides a brief description of the limitations of RPL upon mobility, and outlines efforts to improve this behaviour, namely through SDN-based IoT network/mobility management frameworks; it also sheds some light on the benefits of using Bayesian filters for improving location estimation and handoff decisions. Section III describes the SDMob architecture and the used filters. Section IV shows the details of SDMob implementation and test environment. Moreover, SDMob comparison with mRPL will be shown and discussed. Finally, in Section V we conclude the paper.

II. Related Work

Mobility-aware RPL routing. Mobility management can be performed at different layers of the protocol stack. There is substantial body of research exploring detecting of radio link failure and network disconnection in IPv6 Neighbor Discovery or in the MAC layer. Nevertheless, to avoid routing loops, mobility management should also be considered at the routing layer [7].

RPL is considered as the *de facto* routing protocol for IoT. While it naturally supports joining and leaving of nodes, it performs poorly upon the dynamics imposed to the network topology. RPL maintains a distributed data structure named *Destination Oriented Directed Acyclic Graph* (DODAG). The process starts with the root transmitting a DODAG *Information Object* (DIO) that embeds the needed information to construct a routing tree towards the root.

RPL allows two modes of operation –*storing* and *non-storing*– for downstream traffic. In non-storing mode, it is only the root that maintains the downward routes. This mode scales better since the memory footprint at intermediate nodes does not increase with the size of network. It should be pointed out that in RPL it is more challenging to support mobility for downstream traffic since a mobile node must notify the root (rather than only updating its parent for upstream traffic).

The authors in [7] classify RPL enhancements to support mobility into scenarios with networks including only mobile nodes (e.g. VANETs) and networks with both static and mobile nodes. For the former, the recommendations of the RPL standard to not setting the mobile nodes as routers cannot be respected. In this case, Tian et. al [8] try to adjust the Trickle timer according to mobile nodes' velocity and utilize geographical information as RPL metric. In case the mobile node is not equipped with IMU sensors, it is possible to estimate its position through the *Doppler Effect*, as explained in [9]. The most suitable model to predict the handoff depends on the available sources of data. For instance, a *Deep Learning*-based Long Short Term Memory (LSTM) model could be preferred, if a supervised data set exists for the model to learn from [10].

Location prediction models for proactive handoff. In [11], authors have proposed Kalman RPL, in which a mobile node transmits a beacon that includes its velocity information in specific intervals. After a positioning phase that estimates the current position of the mobile node using three static nodes in its vicinity, it can predict the future position of the node.

EKF-RPL [12] takes a similar approach but it employs EKF within RPL to support non-linear trajectories. There are also some efforts on adopting *on-demand* routing strategies when a node starts searching for a route for transmitting data. The *Lightweight On-Demand Adhoc Distance-vector* routing protocol - Next Generation (LOADng) [13] is one such protocol specifically designed to support any-to-any communication in LLNs, although it is not as well-studied as RPL. EKF-LOADng [14] predicts RSSI after a trilateration positioning. In the triangulation phase, a mobile node broadcasts a message asking for packets from its static neighbors. Responses from static nodes experience a random waiting time to avoid collision. Another challenge is that the mobile node has to be programmed with the position information of the anchors and perform the sophisticated filter on its own.

Particle filter leads to more accurate results and better resiliency against nonlinear moving trajectory and non-Gaussian noise compared to EKF [3]. Particle filter also known as *Sequential Monte Carlo* uses hundreds to thousands of samples to predict the future state and fuse the measurement, hence requiring a higher computation capacity than most constrained IoT nodes can provide.

Particle filter has been extensively used to support mobility in Unmanned Aerial Vehicle (UAV) assisted networks [15] or cellular communications as in [16] which also proposes a *Rao-Blackwellised* particle filter as a lightweight alternative to the baseline particle filter. There are fundamental differences between cellular networks and LLNs such as density of the network deployment, range of transmission and speed of mobile nodes.

SDN-enabled IoT network architectures. There is a growing popularity in using SDN-enabled solutions in IoT networks. It is too expensive in terms of to simply integrate the common SDN solutions and standards within constrained IoT networks without re-designing the SDN to consider IoT limitations [17]. Therefore, there is a requirement for devising solutions targeting IoT networks with reduced complexity and operational cost.

Efforts have been made (including by standardization bodies) to design solutions for managing IoT network. The *Internet Engineering Task Force* (IETF) has a recent draft for infusing data routes into the network that is called *DAO projection* [18]. It defines a framework for the root node to initialize some options in DODAG *Advertisement Objects* (DAO) through new control messages, namely *Project DAO Request* (PDR) and *PDR-Acknowledgement* (PDR-ACK). This enables the root node to install routes in either the source or intermediate nodes along the path. The mechanism is a low-overhead substitute for implementing centralized network management in IoT networks.

Coral SDN [19] is another RPL-based solutions which allows an SDN controller to manipulate RPL routing parameters such as the interval used by the Trickle timer. The interval is the duration between successive DIS messages from a leaf node, which is an important configuration to adapt the responsiveness of the network.

MobiFog [20] is our previous work on centralised mobility management, where the discovery of alternative parents is performed using the actual data packets instead of dedicated control packets (beacons).

Overall, the literature in IoT networks mostly neglect more sophisticated and computation-intensive filters for network/mobility management, as well as SDN-based architectures. We believe that SDMob paves the ground for employing more accurate filter/localization algorithms at the SDN controller, towards improved performance upon mobility in IoT networks.

III. SDMob Architecture

In this section, we present the proposed SDMob architecture in two subsections. First we describe the filter/localization process. Then, we outline the SDMob architecture with the mobility management mechanism.

A. Filter design

Filters help with the prediction of future position of Mobile Node (MN), based on radio signal strength data and velocity. A more accurate localization of the MN improves network connectivity through proactive handoff. Filters model the states (positions) and observations as a *Hidden Markov Model* like the one illustrated in Figure 1. Within the model, the *Markovian* property holds true, meaning that each state (k-th) at a given time only depends on the state before ($k-1$-th) and the states can be estimated not directly but through some noisy observations.

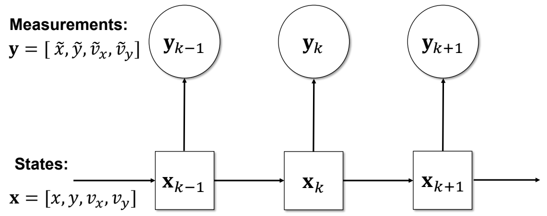


Fig. 1. Markovian dependencies for the tracking problem

The state vector and measurement vector at k-th time step are $\mathbf{x}_k = [x \quad y \quad v_x \quad v_y]^T$ and $\mathbf{y}_k = [\tilde{x} \quad \tilde{y} \quad \tilde{v}_x \quad \tilde{v}_y]^T$ respectively. x and y are positions within Cartesian coordinates. Measurements are based on the Received Signal Strength Indication (RSSI) and IMU observations. Using the path loss model in Equation 1, the controller estimates the distance between mobile nodes and three distinct static nodes and then applies triangulation. In Equation 1, P_1 denotes the received signal strength in a 1 meter distance and α is a constant describing the radio propagation in the environment [21].

$$d = 10^{(\mathrm{RSSI}-\mathrm{P}_1)/10\alpha} \tag{1}$$

The state vector can be related to the previous state (Markov Property) using the Equation 2 and relation of each state with the corresponding measurements can be described using Equation 3. The estimated probability of the current state is based on the observations up to the current state, formulated as $p(x_k|y_0, y_1, \ldots, y_k)$. Then in the prediction step the probability density $p(x_{k+1}|y_0, y_1, \ldots, y_k)$ is computed which is the probability density of the next state $k+1$ knowing the observations as of the current time step.

$$\mathbf{x}_k = F(\mathbf{x}_{k-1}) + \mathbf{n}_{k-1}, \tag{2}$$

$$\mathbf{y}_k = H(\mathbf{x}_k) + \mathbf{r}_k, \tag{3}$$

In this system model, F is the motion model function that describes the relationship between states in time, and H is the function that relates the current state to the noisy observations at the current time instance. $\mathbf{n}_k$ and $\mathbf{r}_k$ are the prediction and measurement noise respectively. These two noises are mutually independent. The aforementioned functions and noises determine applicability of the classic filters. Kalman Filter is only advantageous in linear functions and Gaussian noise. Some enhancements such as *Enhanced Kalman Filter* (EKF) focus on handling non-linear F and H functions. However, to counteract non-Gaussian noise under a non-linear trajectory in the controller, we are compelled to adopt some other techniques such as UKF or particle filter.

a) Unscented Kalman Filter: Instead of solving the intractable non-linear equations, UKF picks a number of deterministic samples that can be processed by the non-linear functions easily. These samples or *Sigma Points* and their corresponding weights satisfy some conditions defined by the Unscented Transform. Then these sigma points are mapped by a non-linear function to the new points. Finally, a good estimate of posterior mean and covariance of the transformed points is calculated using simple weighted averaging. We have implemented both filters using the python libraries introduced by [22].

b) Particle Filter: Particle filter also known as Monte Carlo Sampling maintains a set of fully random particles, though the number of samples is expected to be much higher. The filter can take advantage of any a priori knowledge of the obstacles and infeasible positions when initializing the samples. Once it receives the RSSI measurements it updates their weights and based on the IMU information moves all the particles together. On the other hand, particle filter has the feature of defining obstacles or feasible areas for the mobile node by simply removing the samples that get to be outside the legit area. In long term some of the samples can turn out to be irrelevant so particle filter can disregard those improbable particles and perform a re-sampling mechanism. When the number of effective samples (their weight crosses some threshold), the filter can take different re-sampling strategies. A systematic re-sampling in which the new samples are scattered around the state space is used in our implementation.

B. SDMob architecture

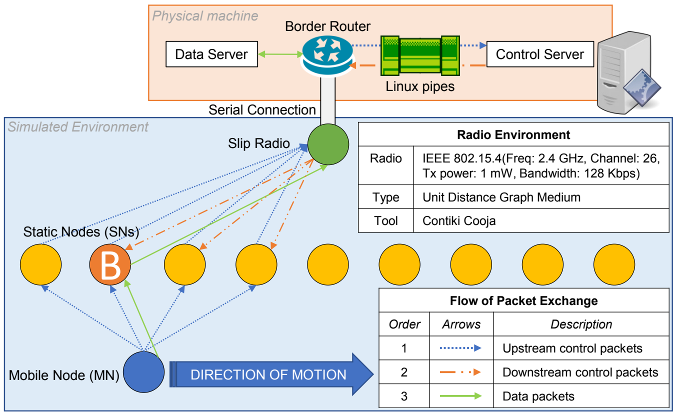
Figure 2 details the SDMob architecture, including an illustrated topology and the underlying technologies. Within WSN, network is composed of Mobile Node (MN) and Static Node (SN). The border-router utilizes the Serial Line Internet Protocol (SLIP) at border router between WSN and external SDN-Controller. SLIP converts the radio messages to a standard for serial links and vice versa, but it involves some unavoidable delay due to serialization. To interconnect the controller to the border router, we use Linux kernel pipes.

Chiefly, SNs act as repeaters or forwarders for MNs, and aid in their localization. We make two assumptions of the WSN: (i) implementation of RPL/6LowPAN protocols exists in the SNs and (ii) MNs are equipped with IMU sensors. In this design, software-defined network management supplements existing RPL/6LoWPAN with programmable network through route updating anchors which enables logically seamless connection for mobile nodes. These functionalities could be implemented either in the application layer on top of the protocol stack or integrated in the of RPL/6LoWPAN. Beacons sent from MN and forwarded by SNs towards SDN controller. Based on the observations made by beacons, controller provides route updates. The handover process is carried out in 3 steps, as illustrated in Figure 2.

- Step 1: MN broadcasts control beacon in a upstream with velocity information from IMU sensor.
- Step 2: All SNs in vicinity of MN receive and relay beacons with appended RSSI values in upstream towards the controller.
- Step 3: Controller runs the filter selects the best SN to act as new best parent, which is transmitted in a downstream packet towards SNs. Thereafter, only the best node relays the data packets generated by MN.

Seamless hand-off mechanism relies on a robust connection with the controller. Hence, low-complex design characteristics has to be involved in implementation of SDN architecture for WSN, which we detail below:

- **Avoidance of collision between control and data packets.** A centralized SDN-based controller introduces an additional control overhead. This increases the traffic through basic MAC-layer implementation of WSN, which is incapable of handling it and eventually more packet drops are experienced due to collisions. To streamline the traffic, a reserved period for control packets called *Control Window* (CW) has been implemented. CW can be adjusted based on network dynamics.
- **Configuring MN as RPL Aware Leaf (RAL).** As defined by a recent standardization effort to employ RALs in RPL [23], a leaf in RPL is a host that does not participate in further advertising the DODAG and relies on the RPL routers to forward its traffic. SDMob takes advantage of RALs to avoid excessive DIO packets as it is a major source of energy consumption when there is frequent topological changes. Another upside is reduced memory footprint in the MN. Second, this limits possibility of distinguishing MN as an intermediate node for other SNs. Last but not least, it would suffice to only notify the SNs of the current *best parent* rather than using the links to the MN that are much less reliable.
- **Sophistication of downward routing.** In standard RPL, upstream data transmission is favored as it is the most predominant traffic pattern in IoT domain. This extends to many mobility enhancements made to RPL as they also have weaker behavior in terms of downward traffic towards the mobile node. Handovers are treated locally without briefing the root node SDMob works with the non-storing mode of RPL which gathers more routing information at the root and uses source routing for downward data packets but builds upon the extra localization to also improve downward traffic towards the MN.



Radio Environment	
Radio	IEEE 802.15.4(Freq: 2.4 GHz, Channel: 26, Tx power: 1 mW, Bandwidth: 128 Kbps)
Type	Unit Distance Graph Medium
Tool	Contiki Cooja

Flow of Packet Exchange		
Order	*Arrows*	*Description*
1	┈┈▶	Upstream control packets
2	─ · ▶	Downstream control packets
3	──▶	Data packets

Fig. 2. SDMob architecture schematic with exchange flow: i) MN broadcasts beacons in upstream towards controller through SN; ii) controller broadcasts the new *best parent* in downstream to SNs; and iii) data communication. Only B (best parent) handles data packets generated by MN.

In Figure 3(a), a timeline demonstration of overall handoff process in SDMob is illustrated. In the data window, MN broadcasts data packets and selected SN or best parent forwards them to the border router. In the control window, MN only transmits the localization beacon and all static nodes receiving the control packet will append the measured RSSI to forward the packet to towards controller with CSMA-based unicast packets. The controller runs the filter and announces the new *best parent*. The old best parent stops forwarding data once it is notified of the SDN's most recent choice. After the CW, data transmission is resumed. Figure 3(b) shows a timeline of hand-off mechanism in benchmark mRPL. In mRPL, the Mobile Node (MN) operates in two phases. In the *Data Transmission phase*, the APs constantly monitor the link quality and compare the RSSI value to a threshold T_l. If the link quality degrades, the APs notify the MN with a beacon and it stops transmitting data packets and instead it will start the *Discovery Phase* by sending DIS packets to ask the APs to respond with DIO packets. Then the mobile node analyses the received DIO packets and if the RSSI values from other APs turn out to be higher than T_l it performs the hand-off and resumes data transmission, otherwise it continues sending DIS packets.

IV. Simulation Setup and Analysis

SDN controller has been implemented using Linux machine with Python-based filters which connects to C-based Contiki border router. For the IoT RPL/6LoWPAN network, we rely on the Contiki-NG/COOJA simulation environment [24]. Contiki-NG is an open-source embedded operating system which is easily portable to commodity hardware.

Simulation setup. We compare the performance of SDMob with filters (particle filter and UKF) with mRPL as well as with the default RPL. We consider one two different trajectories first a linear and second a circular trajectory. The architecture allows multiple mobile nodes given that controller can run

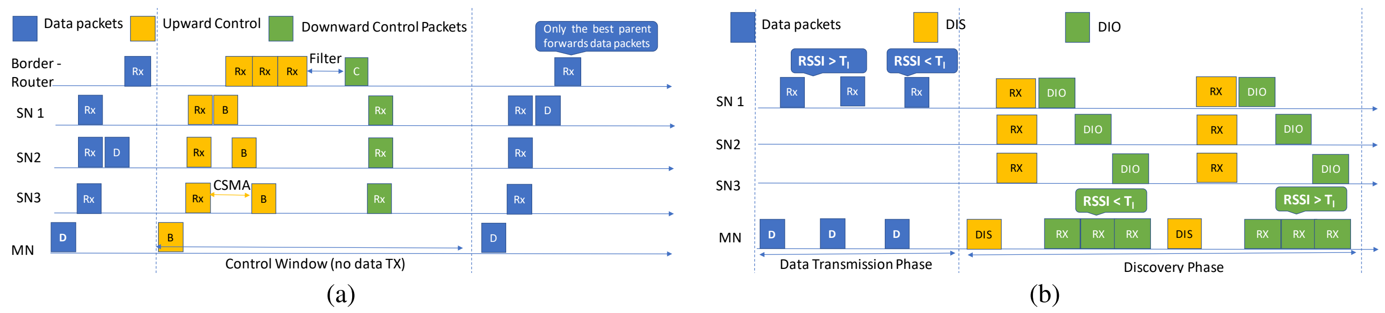


Fig. 3. A timeline diagram of handoff in (a) SDMob (b) and mRPL.

different instances of the filter and differentiate beacons as they include MN's IP address. Though due to space limitations, we share the results regarding a single mobile node with different moving tracks. The simulations consider different sampling intervals for data transmission (here the mobile node sends data packets periodically), path-loss variance and CW. We have employed mRPL [5] as the benchmark since it provides a mobility solution for IoT networks using a non-SDN-based framework. We could not find any available implementation of those related work with variations of Kalman Filter. The experimental results considers three main performance metrics, including (i) packet delivery ratio (PDR) to measure reliability, (ii) End-to-End (E2E) delay, and (iii) Root Mean Square Error (RMSE) to measure positioning accuracy . We investigate the impact of different parameters on these metrics as follows:

Impact of data transmission interval. Data transmission interval is the time interval between consecutive transmission of data packets. As shown in Figure 4(a), we analyzed the E2E delay for data packets. We observed that E2E delay is in the range of 150 ms with SDMob, while mRPL has much lower delay ($\leq$ 50 ms), slightly less that RPL's $\approx$ 100 ms. This is important to note that delay in RPL is calculated only for packets that have been successfully transmitted ($\approx$ 10%). This means that packets in RPL with mobile node is either transmitted before parent switching or getting lost due to the slow parent switching process and lack of preferred parent. Additional delay in the SDMob can be explained due to:

- CW occupies a specific time interval for control packets, which postpones the transmission of data packets.
- Serialization delay occurred by SLIP protocol to convey radio messages to the Linux-based controller. This delay includes 20ms polling intervals as well as the processing time.

Unlike mRPL which performs handoff based on the most recent RSSI averages, performance of handoff in SDMob is independent of the network traffic – see Figure 4(b). This robustness and resiliency is achieved at the cost of a constant control overhead and the incurred delay. For short data transmission intervals, mRPL provides about 80% PDR with a decreasing trend for longer intervals surpassing standard RPL's poor PDR ($\approx$ 20%). **SDMob constantly outperforms mRPL and RPL with PDR above 95% across different data rates**.

Impact of various filtering methods and trajectories. To analyze the positioning accuracy of filters, RMSE metric is mostly used in the literature (compared to the average error). The reason is that RMSE assigns a higher penalty to large errors. As it is shown in Figure 4(c), **particle filter provides a better positioning error compared to the UKF for both linear and circular trajectories with different sampling rates**. Increasing the physical speed of mobile node also deteriorated the metrics as the number of handoffs increases.

Experiments also revealed that SDMob is more resilient even under path loss variance (not illustrated in the figures) and provides better PDR. This can be explained by the fusion of measurements in the filter that takes advantage of IMU information in parallel with the non-stable RSSI observations.

Impact of CW size. Longer control window increases the stability of the network through longer network monitoring, which thereby increases the probability of a successful transmission, though it imposes a longer delay on data packets. As expected, the simulations exhibit an increasing trend both in PDR and delay while the CW length increases. The CW only needs to cover the time for a round trip time for control packets. The average delay for control packets in a two-hop network including the delays for SDMob's particle filter algorithm is about 200 ms. For CW ranging from 0 to 300 ms, we observed average delays from 175 to 220 ms, while PDR increased linearly from 89 to 100 percent.

TABLE I
COMPARING MEMORY FOOTPRINT OF SDMOB WITH CONTIKI-NG'S DEFAULT RPL IMPLEMENTATION.

Program	SD-Mob		mRPL		Contiki-NG base	
	ROM	RAM	ROM	RAM	ROM	RAM
MN	46 KB	7628	44 KB	7898	43 KB	7394
Data Server	163 KB	7400	46 KB	7916	43 KB	7348
SN	44 KB	7632	45 KB	7928	44 KB	7632

Memory footprint. Memory consumption of the nodes is a measure that can testify to offloading excessive computation to the SDN controller. Contiki-NG's implementation of RPL has optimized its code and since SDMob is based on Contiki-NG it is unfair to compare it with mRPL, which is based on Contiki 2.6. As shown in Table I, for the mobile node and the anchor nodes, SDMob's MN and SN require about 3% less RAM memory compared to mRPL and not too much overhead compared to base Contiki-NG (no mobility support). The server's memory consumption is much higher than mRPL but since SDMob's server is offloaded to the Linux machine, it will have no negative impact in the IoT network. So overall, we can argue that SDMob has outsourced the mobility related computations.

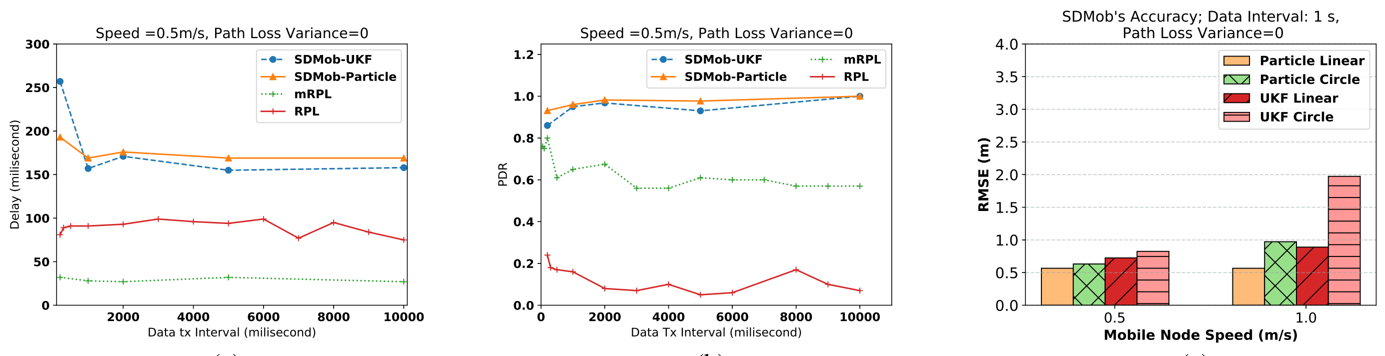


Fig. 4. Simulation results showing (a) E2E delay and (b) PDR for different sampling rates and (c) positioning accuracy for different trajectories.

V. Conclusion

In this paper, we have addressed the design and implementation of SDMob, an SDN-based mobility solution for IoT RPL/6LoWPAN networks. The proposed architecture is based on an applying filters to radio signal strength and velocity measurements captured by the anchor nodes. This mechanism enables a more accurate prediction of the mobile nodes and consequently a more precise selection of the best anchor nodes. Simulation results showed that by using a periodic beaconing mechanism, SDMob's PDR and delay are independent of the network traffic, while mRPL is tightly coupled with data transmission interval. RPL has shown to be very low responsive to dynamics in the network, leading to high packet losses. The CW mechanism and the extra control packets imposed an overhead that justifies the higher delay in SDMob. Since real-world environments exhibit more varying link behavior, future experiments will be based on a real hardware setup. Further tests on scalability of the system, impact of node density, number of mobile nodes, trajectory of movement and dynamic adaptation of CW are also envisaged.

We plan to further extend this work to support adaptive beacon rates, automatically detect radio characteristics of the environment and evaluate the scalability in terms of number of mobile nodes and density of static nodes in our future work.

References

[1] B. Safaei, A. Mohammadsalehi, K. T. Khoosani, S. Zarbaf, A. M. H. Monazzah, F. Samie, L. Bauer, J. Henkel, and A. Ejlali, "Impacts of mobility models on rpl-based mobile iot infrastructures: An evaluative comparison and survey," *IEEE Access*, vol. 8, pp. 167 779–167 829, 2020.

[2] K. C. Lee, R. Sudhaakar, J. Ning, L. Dai, S. Addepalli, J. Vasseur, and M. Gerla, "A comprehensive evaluation of RPL under mobility," *International Journal of vehicular technology*, vol. 2012, 2012.

[3] S. Särkkä, *Bayesian filtering and smoothing*. Cambridge University Press, 2013, vol. 3.

[4] O. Cappe, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[5] H. Fotouhi, D. Moreira, and M. Alves, "mRPL: Boosting mobility in the Internet of Things," *Ad Hoc Networks*, vol. 26, pp. 17–35, 2015.

[6] I. Rabet, S. P. Selvaraju, H. Fotouhi, M. Vahabi, and M. Bjorkman, "Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks," in *EWSN 2020*.

[7] P. O. Kamgueu, E. Nataf, T. D. Ndie, P. O. Kamgueu, E. Nataf, T. Djotio, and N. Survey, "Survey on RPL enhancements : a focus on topology , security and mobility To cite this version : HAL Id : hal-01713247," *Computer Communications*, 2018.

[8] B. Tian, K. M. Hou, H. Shi, X. Liu, X. Diao, J. Li, Y. Chen, and J.-P. Chanet, "Application of modified RPL under VANET-WSN communication architecture," in *2013 international conference on computational and information sciences*. IEEE, 2013, pp. 1467–1470.

[9] J. Park, K. H. Kim, and K. Kim, "An algorithm for timely transmission of solicitation messages in RPL for energy-efficient node mobility," *Sensors (Switzerland)*, vol. 17, pp. 1–21, 2017.

[10] C. Zhang, P. Patras, and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[11] M. Barcelo, A. Correa, J. L. Vicario, A. Morell, and X. Vilajosana, "Addressing Mobility in RPL with Position Assisted Metrics," *IEEE Sensors Journal*, vol. 16, no. 7, pp. 2151–2161, 2016.

[12] M. Bouaziz, A. Rachedi, and A. Belghith, "EKF-MRPL: Advanced mobility support routing protocol for internet of mobile things: Movement prediction approach," *Future Generation Computer Systems*, vol. 93, pp. 822–832, 2019. [Online]. Available: https://doi.org/10.1016/j.future.2017.12.015

[13] T. Clausen, J. Yi, and A. C. De Verdiere, "LOADng: Towards aodv version 2," in *2012 IEEE Vehicular Technology Conference (VTC Fall)*. IEEE, 2012, pp. 1–5.

[14] A. J. Gonçalves, R. A. Rabêlo, J. J. Rodrigues, and L. M. Oliveira, "A mobility solution for low power and lossy networks using the LOADng protocol," *Transactions on Emerging Telecommunications Technologies*, no. November 2019, pp. 1–24, 2020.

[15] "pafir: Particle filter routing–a predictive relaying scheme for uav-assisted iot communications in future innovated networks."

[16] L. Mihaylova, D. Angelova, S. Honary, D. R. Bull, C. N. Canagarajah, and B. Ristic, "Mobility tracking in cellular networks using particle filtering," *IEEE Transactions on Wireless Communications*, vol. 6, no. 10, pp. 3589–3599, 2007.

[17] Z. Latif, K. Sharif, F. Li, M. M. Karim, S. Biswas, and Y. Wang, "A comprehensive survey of interface protocols for software defined networks," *Journal of Network and Computer Applications*, vol. 156, p. 102563, 2020.

[18] P. Thubert, R. Jadhav, and G. M., "Root initiated routing state in RPL draft-ietf-roll-dao-projection-09," 2019. [Online]. Available: https://datatracker.ietf.org/meeting/106/agenda/roll-drafts.pdf

[19] G. Violettas, S. Petridou, and L. Mamatas, "Routing under Heterogeneity and Mobility for the Internet of Things: A Centralized Control Approach," in *2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings*, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8647237/

[20] H. Fotouhi, M. Vahabi, I. Rabet, M. Björkman, and M. Alves, "MobiFog: Mobility Management Framework for Fog-assisted IoT Networks," in *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[21] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.* IEEE, 2004, pp. 517–526.

[22] R. Labbe, "Kalman and bayesian filters in python, 2014," *URL https://github. com/rlabbe/Kalman-and-Bayesian-Filters-in-Python*, 2019.

[23] P. Thubert and M. Richardson, "Routing for RPL leaves," *Work in Progress, draft-thubert-roll-unaware-leaves-05*, 2018.

[24] F. Österlind, "A sensor network simulator for the contiki os," *SICS Research Report*, 2006.