Enhanced Reliability of Mobile Robots with Sensor Data Estimation at Edge

Victor Kathan Sarker[1], Prateeti Mukherjee[2] and Tomi Westerlund[1]

[1]Department of Future Technologies, University of Turku, Turku, Finland
[2] Department of Computer Science and Engineering, IEM, Kolkata, India
Email: [1]{vikasar, tovewe}@utu.fi, [2]mukherjeeprateeti01@gmail.com

***Abstract*—The proliferation of sensing equipment serving an expansive range of applications has led the Internet of Things (IoT) paradigm to cover technologies beyond Wireless Sensor Networks (WSN). Extensive advancement in electronics, communication methods and sensors has made it possible to leverage advanced technologies such as Machine Learning and Probabilistic Modeling in resource-constrained embedded systems. These techniques increase reliability and enhance interactions among physical elements of an IoT-based system in which data loss or corruption seems inevitable. However, traditional data estimation and reconstruction methods cannot be directly applied considering the computational limitations at the edge of the network. Therefore, mobile robots would greatly benefit from a resource efficient sensor data recovery procedure, capable of generating near-accurate estimates at the resource-constrained Edge layer. In this paper, we introduce a novel Bayesian filtering-based data reconstruction scheme, with real-time performance and precision for incoming semantic and geometric information from a varied set of sensors to increase reliability of autonomous navigation of mobile robots. Afterwards, we corrupt each stream of observations to validate model performance against a baseline. Furthermore, we also provide benchmark on execution latency, CPU usage and current draw while running the models in a practical setup.**

***Index Terms*—Reliability, Edge, Mobile Robot, Sensors, State-Space Models, Data Estimation, Bayesian Filtering.**

I. Introduction

Autonomous and semi-autonomous mobile robots have gained popularity over the past few years due to the advancement in efficient algorithms, and the development of powerful embedded hardware and sensor technology. Aiding the implementation of Industry 4.0 objectives, these robots find a wide variety of use in our daily life, with little to no human intervention [1]. The support of these units greatly increase the accuracy and pace of work, reduce manual labor and risk of injuries, and help perform tasks in a more systematic manner. In addition, these units can operate in both independent as well as coordinated manner in highly interconnected systems, with a centralized or distributed architecture through sharing of data with other system nodes. With the evolution of the Internet of Things (IoT), these systems have taken a new dimension as autonomous and semi-autonomous robots have been developed to be more interactive and environment-aware entities for integration in numerous applications. In an industrial setting, these robots play an important role, working either as individual units or alongside humans. Examples of these setups include the use of robots for menial tasks such as sorting, in-factory carriage and cleaning [2]–[4]. Successful execution of these duties, however, requires the robots to be aware of surrounding environment. To achieve this, autonomous robots make use of a variety of sensors such as proximity and distance sensors to understand their surroundings and navigate accordingly. A greater number of sensors result in more data, burdening the resource-constrained processing units onboard the robot. Furthermore, data loss is unavoidable in these systems considering unreliable wireless links and hardware failures at system nodes [5]. Malfunctioning nodes and communication breakdowns result in corrupted data transmission, a prevalent issue in both industrial automation setups as well as research. Controlled and cleverly designed systems also suffer from data loss, thereby reducing the statistical rigor in processing and leading to invalid conclusions [6].

Although robots can be powered by small fuel-powered engines, most autonomous indoor robots run on batteries. On account of the limited supply capacity of batteries, the computational unit and other electrical and electronic components within the robot must be as energy-efficient as possible. In order to ensure longer run-time of the system, it is crucial that the algorithms running onboard be of lower complexity and run efficiently on embedded controllers with inferior computational resources. Moreover, high functional accuracy of sensing equipment and low rate of faulty data contribute to a low processing cycle count since repeated measurements and analysis are not required. However, factors such as noise, interference, non-linearity due to temperature variations, mechanical errors and faults in circuitry often lead to missing observations and faulty acquisition. In such circumstances, approximating these observations and reads based on the nature of the sensor and previous observations can greatly aid proper operation of the robot units.

While traditional IoT-assisted robotic systems bear great potential, they suffer from issues such as inadequate bandwidth and high communication latency when the volume of data to be forwarded to cloud servers is large. To overcome these problems, the use of Edge and Fog gateways plays an important role for enhancing energy-efficiency and operational performance of resource-constrained indoor robots [7]. Although the hybrid Edge-Fog-Cloud architecture is an impactful approach to efficiently offload computationally expensive tasks

from nodes with limited processing capabilities, the data transmission phase is still susceptible to noise, interference, and data loss. If a major chunk of the data is lost, the offloading process will inevitably cause faulty decisions, leading to major system performance concerns. Therefore, while the benefits of offloading in IoT ecosystems are abundant, it is important to account for the hazards and ensure that algorithms are in place to mitigate data corruption and loss.

In this work, we exploit Machine Learning (ML) techniques to reconstruct missing sensor data in small autonomous mobile robots that constitute the resource-constrained Edge layer. In particular, we employ Bayesian Statistics and the concept of Hidden Markov Models (HMM) to develop two estimation models with linear time complexity in the number of time-steps. Furthermore, we simulate an experimental setup with 2D Light Detection and Ranging (LIDAR), Gyroscope, and Global Positioning System (GPS) observations for state vector representation to evaluate performance of proposed models. We then corrupt the observations and illustrate how different parameters affect the estimation outcome. The specific contributions of the paper are:

- Transform existing mathematical concepts of Recursive Bayesian State Inference to develop two domain-specific models for sensor data estimation,
- Experiment to validate estimation at the Edge with data from LIDAR, gyroscope and GPS module, and,
- Analyze results and discuss the effect of different parameters on results of estimation.

The rest of the paper is arranged as follows: Section II enumerates related works and state of the art, Section III describes the proposed method for data estimation at the Edge, Section IV demonstrates the experimental evaluation of the proposed method through simulations and presents the results. Finally, Section V concludes the paper and leaves directions for future work.

II. Related Work

Many research works exist in the field of missing data recovery and estimation, however, seldom are those advanced algorithms suitable for resource-constrained computational units. Izonin et al. [8] used stochastic integrals to increase dimensionality of their task in combination with the *AdaBoostRegressor* to search for decomposition coefficients. The authors simulated their model on real data acquired from an IoT-based air quality assessment system and compared the performance against existing schemes. While it resulted in a 6% increase in accuracy, the paper disregards the importance of other parameters such as resource utilization and energy-efficiency, if run on a resource-constrained Edge device. Ullah et al. [9] presented a method for localization based on extended Kalman Filter with Edge computing in WSN. The authors ran simulations with GPS data at different velocity settings. However, the execution impact on Edge computation units is not considered and real-life benchmarks are not performed, thus limiting the usability when multiple nodes simultaneously require processing of localization data at the Edge.

Fekade et al. [10] presented a similar approach in which Probabilistic Matrix Factorization (PMF) is performed within a group of sensors to detect similarities among the obtained data points to subsequently split these points into clusters using the *k-means* clustering scheme. The metrics for comparison of their proposed approach against the standard Support Vector Machine (SVM) and Deep Neural Network (DNN)-based algorithms include Root Mean Squared Error (RMSE) and execution time for normalized data. However, other crucial parameters are not considered and the computation-intensive process can only serve feasible results on a handful of high-end devices at the Edge.

Peng et al. [11] proposed an Incremental Space-Time-based Model (ISTM) to recover missing data points in real-time. The Incremental Multiple Linear Regression forms the basis of the ISTM scheme in which the model is continually updated in accordance with the intermediary data matrix. In addition, estimations for missing values are drawn considering nearly historical data along with the observations supplied by the neighboring sensing equipment. This approach results in increased computation time. However, given the repeated amendments to the model, the process bears potential in serving important tasks related to data imputation such as outlier detection in sensor data streams.

Lujic et al. [12] suggested an interesting semi-automatic recursive approach for missing data recovery at the Edge. In their work, the indices pertaining to gaps in data are marked prior to starting the recovery cycle which identifies the amount of missing and faulty data points. This process is repeated until all missing values have been traversed. Subsequent functional steps include data analysis to find the right setup of forecasting methodology, replacing gaps with predicted values, and checking conditions for the succeeding recovery cycle. This produces good results in terms of accuracy, however, the performance and utility of the model could be enhanced with the use of deeper statistical analytics and ML tools.

An interesting odometry estimation method in 3D LIDAR scans with Convolutional Neural Networks (CNNs) is suggested by Velas et al. [13]. Here, the sparse data points are first passed through an encoder to generate 2D matrices, which are then fed as inputs to the Neural Networks to obtain rotation probabilities. However, given the complexity of CNNs, the work lacks the computational efficacy to function in Edge devices. In a different work [14], missing points in the trajectory of a moving vehicle are recovered through microscopic traffic flow models, wherein, calibration is done by assigning weights to known data points based on their proximity to the gaps using the tri-cube weight function. Extensive comparisons are drawn with Newell's, Pipes, Intelligent-Driver Model (IDM) and Gipps' car following models, and several curves are plotted to illustrate the contrasting performance.

Vijayakumar et al. [15] suggests the use of Kalman Filters to predict missing events in sensor data streams. In their approach, the filter is applied to SQL-based event processing

systems using the standard implementation in the Bayes++ software library. While the work inspires the proffered models, the use of already existing libraries hinders domain-specific progress, and is hence avoided in our work. In addition, the presented architecture bears no correlation to practical units in an autonomous or semi-autonomous setting, thereby disregarding prevalent problems at the Edge. Furthermore, applying the classical form of the filter to modern-day problems would be an outdated approach. Transforming the mathematical concepts to suit the needs of a particular field is crucial in order to generate meaningful results and to ensure that the solutions adhere to domain-specific challenges and requirements.

III. Data Estimation Scheme

While a multitude of research exists on sensor data retrieval, handful of these works consider the hardware limitations of computational units. However, given the promising capabilities and advantages of Edge computing [16], it is crucial to explore the consolidation of statistical concepts and ML techniques to serve real-life applications where system resources are limited. To this end, the proposed algorithms, albeit based on fundamental principles of Bayesian filtering, are developed to suit the needs of the problem at hand, keeping in mind the computational limitations and power supply constraints in IoT ecosystems, to facilitate improved safety and reliability of autonomous operation.

The state vector Z_t for our problem is:

$$Z_t = \begin{bmatrix} x_t & y_t & \phi_t & v_t \end{bmatrix}^T \tag{1}$$

where, x_t and y_t represent the mapped position of a vehicle using the data transmitted by the 2-D LIDAR sensor in real-time [17], ϕ_t denotes the orientation as realized by the gyroscope, and v_t represents the velocity calculated from coordinates provided by the GPS module.

The state transition equations Z_t and Y_t are defined as:

$$Z_t = f(Z_{t-1}) + q_t \tag{2}$$

$$Y_t = HZ_t + r_t \tag{3}$$

where, $q_t \sim \mathcal{N}(q_t|0, Q)$ is the process noise, $r_t \sim \mathcal{N}(r_t|0, R)$ is the observation noise, and the non-linear state transition function f is:

$$f\left(\begin{bmatrix} x \\ y \\ \phi \\ v \end{bmatrix}\right) = \begin{bmatrix} v\Delta_t cos\phi \\ y\Delta_t sin\phi \\ \omega\Delta_t \\ a\Delta_t \end{bmatrix}$$

where, $\omega = \frac{d\phi}{dt}$ is the angular velocity and a is the acceleration. The process noise covariance Q and the observation noise covariance R are:

$$Q = \begin{bmatrix} \sigma_x^2 & 0 & \cdots & \cdots & 0 \\ 0 & \sigma_y^2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \sigma_\phi^2 & 0 \\ 0 & \cdots & \cdots & 0 & \sigma_v^2 \end{bmatrix}, \text{ and, } R = \begin{bmatrix} \sigma_{rx}^2 & 0 \\ 0 & \sigma_{ry}^2 \end{bmatrix}$$

where, the emission matrix H is described as

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Markovian representations summarize the data obtained from past and present observations for future predictions, in the best estimate of the current state vector. As stated by Kalman [18], the difference between the predicted value from this estimate at the current state and the next available observation is orthogonal to earlier observations, resulting in a simple recursive algorithm for calculation of likelihood in Gaussian processes. Furthermore, in the absence of observations, the prediction means in Kalman Filter models are mathematically equipped to serve as a plug-in replacement for the lost data points since they represent the Bayes' optimal solution [19]. Considering the nature of our problem and the computational efficiency of state space models coupled with the robustness of Kalman Filter against statistical noise and other inaccuracies, we formulate two models based on Extended Kalman Filter (EKF) and Cubature Kalman Filter (CKF) for the proposed system state.

1) EKF-based Model: The non-linearity of EKF invalidates the presumed Gaussian nature of the Process and Measurement models in regular Kalman Filters. The EKF approximates the non-linear model by a local linear model obtained from a first order Taylor expansion around the current estimate, subsequently applying the filtering process to this approximation. The matrix of partial derivatives for the proposed scheme is:

$$F_z\left(\begin{bmatrix} x \\ y \\ \phi \\ v \end{bmatrix}\right) = \begin{bmatrix} 1 & 0 & -v\Delta_t sin\phi & \Delta_t cos\phi \\ 0 & 1 & v\Delta_t cos\phi & \Delta_t sin\phi \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With initial mean (m_0), initial covariance (P_0) and observation (y_k) as input, we transform the Prediction step and introduce the Imputation step as described in Algorithm 1.

2) CKF-based Model: While EKF is a powerful mathematical tool, it might not produce satisfactory results when the system uncertainties are large. Short signal interruptions may result in a quickly diverging EKF, thereby hindering the performance of the proposed model in urban environments. Therefore, we make use of the Cubature Rule to numerically approximate the moment integrals in Bayesian filtering, while completely preserving the second-order information about the state that is contained in the sequence observations [20]. The Spherical Cubature Approximation for Gaussian Integrals [21] states that for any function h,

$$\int h(x)\mathcal{N}(x|m, P) \approx \frac{1}{2n}\sum_{n=1}^{2n} h(m + L\zeta_i)$$

where, L is the Cholesky decomposition of P_{k-1}, and,

$$\zeta_i = \begin{cases} \sqrt{n}e_i & ; i = 1...n \\ -\sqrt{n}e_{i-n} & ; i = n+1...2n \end{cases}$$

Algorithm 1 EKF-based Model

Input: $m_0, P_0, \{y_k\}_{k=1}^T$

for $k = 1, 2, 3, \ldots, T$ **do**
  Prediction Step
    Solve for m_k^- and P_k^-
    $m_k^- = f(m_{k-1})$
    $P_k^- = F_z(m_{k-1})P_{k-1}F_z^T(m_{k-1}) + Q$
  If y_k is not missing
  Update Step
    $v_k = y_k - Hm_k$
    $S_k = HP_k^-H^T + R$
    $K_k = P_k^-H^TS_k^{-1}$
    $m_k = m_k^- + K_kV_k$
    $P_k = P_k^- - K_kS_kK_k^T$
  Else
  Imputation Step
    $m_k = m_k^-$
    $P_k = P_k^-$
end for

Algorithm 2 CKF-based Model

Input: $m_0, P_0, \{y_k\}_{k=1}^T$

for $k = 1, 2, 3, \ldots, T$ **do**
  Prediction Step
    Solve for $\chi_{k-1}^{(i)}, m_k^-$ and P_k^-
    $\chi_k^{(i)} = m_{k-1} + L_{k-1}\zeta_i \quad ; i = 1...2n$
    $\hat{\chi}_{k-1}^{(i)} = f(\chi_{k-1}^{(i)})$
    $m_k^- = \frac{1}{2n}\sum_{i=1}^{2n}\hat{\chi}_{k-1}^{(i)}$
    $P_k^- = \frac{1}{2n}\sum_{i=1}^{2n}(\hat{\chi}_{k-1}^{(i)} - m_k^-)(\hat{\chi}_{k-1}^{(i)} - m_k^-)^T + Q$
  If y_k is not missing
  Update Step
    $v_k = y_k - Hm_k$
    $S_k = HP_k^-H^T + R$
    $K_k = P_k^-H^TS_k^{-1}$
    $m_k = m_k^- + K_kV_k$
    $P_k = P_k^- - K_kS_kK_k^T$
  Else
  Imputation Step
    $m_k = m_k^-$
    $P_k = P_k^-$
end for

Since the problem scenario is not altered, the state vector remains the same as described earlier in (1), as do the state transitions (2) and (3).

$$m_k^- = \int f(x_{k-1})\mathcal{N}(x_{k-1}|m_{k-1}, P_{k-1})dx_{k-1} \approx \frac{1}{2n}\sum_{n=1}^{2n} f(m_{k-1} + L_{k-1}\zeta_i) \quad (4)$$

$$P_k^- = \int (f(x_{k-1}) - m_k^-)(f(x_{k-1}) - m_k^-)^T \mathcal{N}(x_{k-1}|m_{k-1}, P_{k-1})dx_{k-1} + Q \approx \frac{1}{2n}\sum [f(m_{k-1} + L_{k-1}\zeta_i) - m_k^-] [f(m_{k-1} + L_{k-1}\zeta_i) - m_k^-]^T + Q \quad (5)$$

For the CKF-based model, the Prediction Mean m_k^- and Prediction Covariance P_k^- are described as in (4) and (5), respectively. These result in Algorithm 2 for the CKF-based model towards missing data estimation.

IV. Experiment and Results

We simulated the proposed algorithms on the aforementioned state transition model with state trajectories sampled for 1000 time steps at $10\,ms$ intervals. For the experiments in this work, the algorithms are implemented using Python programming language and executed on the Raspberry Pi 3B+, Raspberry Pi 4B, and the Aaeon Intel Up Gateway [22], [23] which serve as the Edge computation unit. In addition, we tested the performance against computers with Intel Core i5-5200U and Intel Core i7-4770 processors, both running the Microsoft© Windows 10 Pro. The comparison of their performance in terms of execution latency, CPU utilization and average current draw is listed in Table I. The execution latency denotes the total time an algorithm takes on a specific processor or platform, and CPU utilization denotes the percentage of total available processor time available.

We benchmark our EKF and CKF-based models with respect to a baseline model that uses only the unfiltered sensor data. Performance of our proposed models with data corruption of $0-80\%$ with a 10% increment per step is shown in Table II. For each model, we report the mean and standard deviation of the RMSE between the actual states and the filter means from 10 independent runs. The trends in RMSE Mean $\pm$ Standard Deviation (SD) for proposed models and the baseline is depicted in Figure 1. The observed trends are indicative of the fact that even at 80% data loss, the RMSE Mean and Standard Deviations for the proposed models remain lower than the baseline. Thus, both our models mitigate the effect of corruption significantly better than the baseline model.

We also tested the proposed models by varying the angular velocity, acceleration, and parameters of process noise covariance. We observed the effects of these variations on the Baseline and the proposed EKF and CKF-based models. In particular, we are interested in the RMSE Mean $\pm$ SD plots for each model upon changing the aforementioned parameters within the corresponding acceptable range, as illustrated by the error plots in Figure 2. For the simulations, we maintain the reference conditions listed in Table III. To study the effects of varying a single parameter, the value of every parameter excluding the one is set as stated the table.

The error plots in Figure 2 suggest that for every parameter varied within their admissible limits, the proposed models perform approximately five times better than the baseline and remain constant on expectation in most cases. Although increasing process noise parameters σ_x and σ_y cause a gradual

TABLE I
COMPARISON OF EXECUTION LATENCY, CPU UTILIZATION AND CURRENT DRAW FOR THE PROPOSED ALGORITHMS.

Platform / CPU	Operating System	Algorithm Based on	Execution Latency (ms)			CPU Utilization (%)			Avg. Current (A)	
			Min.	Max.	Avg.	Min.	Max.	Avg.	Idle	Active
Raspberry Pi 3B+ (ARMv8 Cortex-A53)	Raspbian Buster	EKF	1.159	1.194	1.167	25.00	25.80	25.36	0.49	0.71
		CKF	2.196	2.214	2.203	21.30	25.50	25.16	0.49	0.66
Raspberry Pi 4B (ARMv8 Cortex-A72)	Raspbian Buster	EKF	0.875	0.889	0.880	24.90	25.80	25.27	0.55	0.57
		CKF	1.653	1.709	1.675	24.80	25.60	25.08	0.55	0.59
Aaeon UP-GWS01 (Intel® x5-Z8350)	Ubuntu 16.04 LTS	EKF	2.545	2.604	2.572	34.60	49.10	39.20	0.78	0.85
		CKF	5.598	5.849	5.804	26.20	52.80	47.40	0.78	0.87
Core i5-5200U	Windows 10 Pro	EKF	0.312	0.390	0.350	23.60	46.70	25.64	Untested	
		CKF	0.531	0.609	0.576	23.00	26.50	24.77		
Core i7-4770	Windows 10 Pro	EKF	0.203	0.265	0.250	10.10	14.60	12.09	Untested	
		CKF	0.875	0.889	0.880	24.90	25.80	25.27		

TABLE II
RMSE OF DATA ESTIMATION MODELS DUE TO CORRUPTION. THE FIRST ROW WITH 0% ERROR INDICATES ORIGINAL, UNCORRUPTED DATA.

Corrupted Data (%)	Mean ± SD		
	Baseline Model	EKF-based Model	CKF-based Model
0	0.228 ± 0.005	0.037 ± 0.003	0.038 ± 0.003
10	0.226 ± 0.006	0.041 ± 0.004	0.041 ± 0.004
20	0.226 ± 0.005	0.047 ± 0.003	0.048 ± 0.004
30	0.226 ± 0.006	0.054 ± 0.007	0.055 ± 0.008
40	0.225 ± 0.007	0.056 ± 0.004	0.057 ± 0.006
50	0.226 ± 0.006	0.069 ± 0.008	0.071 ± 0.009
60	0.226 ± 0.004	0.079 ± 0.013	0.081 ± 0.014
70	0.228 ± 0.005	0.104 ± 0.021	0.110 ± 0.023
80	0.224 ± 0.006	0.122 ± 0.034	0.124 ± 0.034

rise in RMSE trends for the proposed models, the overall curves for EKF and CKF-based models still remain well below the baseline curve, as seen in Figure 2c and Figure 2d, therefore outperforming the baseline in each case.

The estimated trajectories for the proposed and baseline models are shown in Figure 3 for uncorrupted data (prior to forced manual corruption) and with heavily corrupted

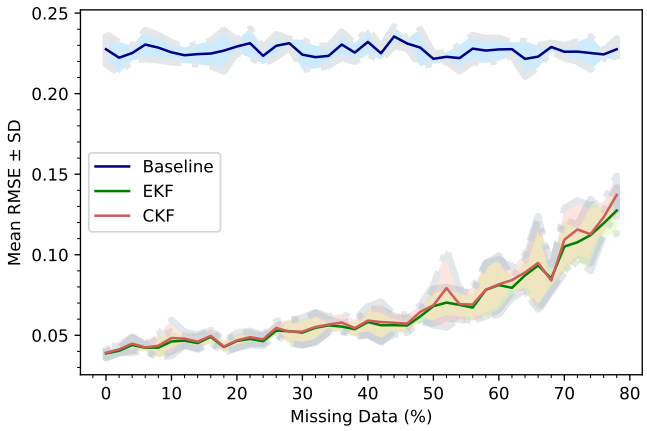


Fig. 1. Mean RMSE (bold line) ± standard deviation (shaded area) compared against baseline for increasing % of missing data.

TABLE III
REFERENCE CONDITIONS FOR SIMULATIONS IN FIGURE 2. WHEN VARYING A PARTICULAR PARAMETER, EVERY OTHER PARAMETER MAINTAINS THE VALUE LISTED IN THE TABLE

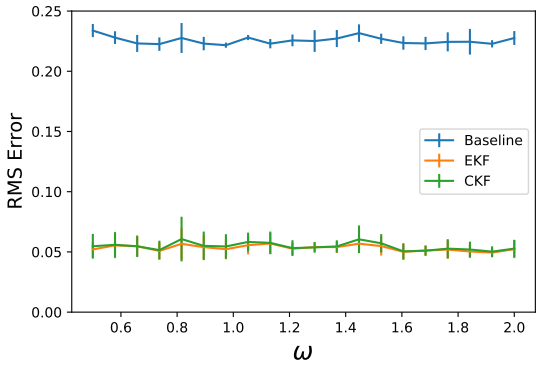
Corrupted Data (%)	a	ω	σ_x	σ_y	σ_ϕ	σ_v	σ_{rx}	σ_{ry}
30	0.2	1.3	0.06	0.1	0.2	0.1	0.45	0.50

data (forced 80% data loss). Figure 3 clearly shows that the proposed EKF and CKF-based data recovery models outperform the baseline throughout the experiment. At first the proposed models perform approximately eight times better than the baseline. Although a rising trend is observed for increasing corruption of sensor data, the proposed models still perform about 1.7 times better than the baseline at maximum corruption, when 80% of incoming data is distorted. It is important to note that even when the RMSE for the proposed data recovery models is calculated on the most corrupt data set, the RMSE for the baseline is worse despite being calculated on the original data where the observations are intact. Thus, the proposed models exhibit robust performance in the context of missing data estimation at the Edge.

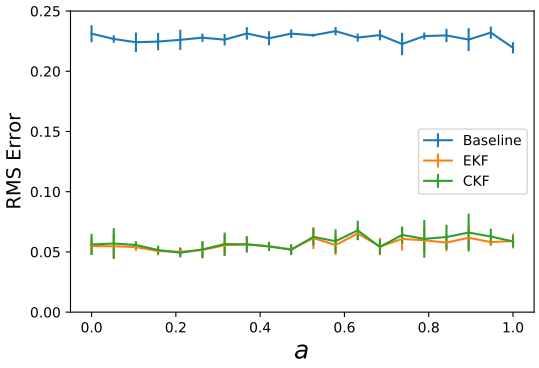
V. CONCLUSION AND FUTURE WORK

With rapid progress in electronics, sensor technology and embedded systems, automation has grown significantly owing to increased productivity, improved robustness and consistent progress in the field. Autonomous mobile units are capable of performing numerous tasks with little to no human supervision using a variety of sensors for contextual awareness, navigation and interaction. However, as many robots are resource-constrained, Edge computing can play an important role by moving the computationally heavy processing from a robot to the Edge. This results in transferring a large amount of data between the mobile robots and Edge which is susceptible to data corruption or loss during transmission due to interference, congestion and unstable network.

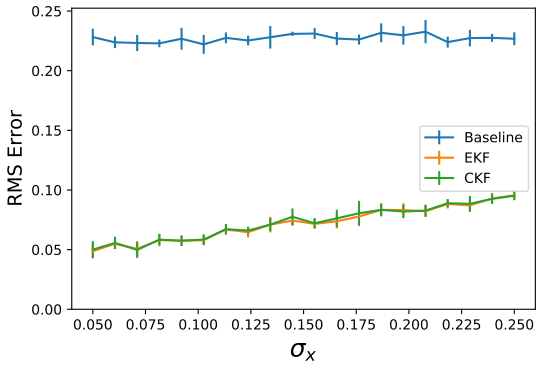
Erroneous data acquisition can lead to erroneous autonomous operation and cause personal injury or damage to property. In this paper, we proposed a novel method for estimating sensor data in real-time with reasonable operational ac-



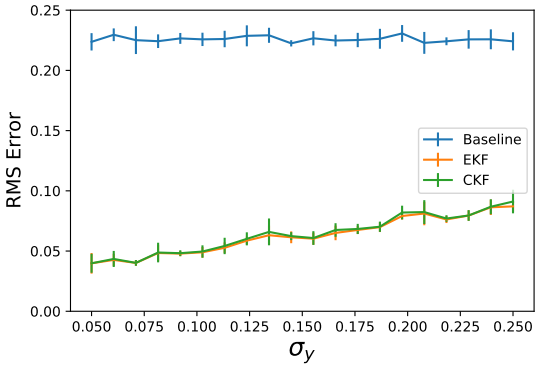
(a) Error bars when varying ω



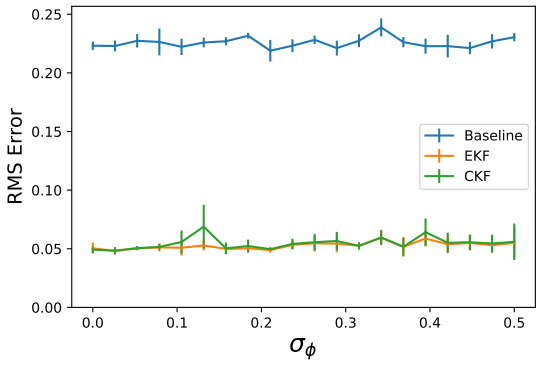
(b) Error bars when varying a



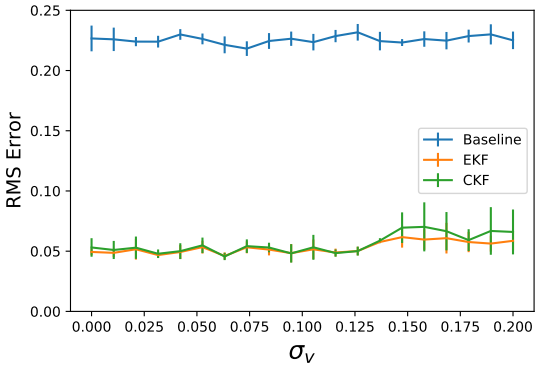
(c) Error bars when varying process noise covariance σ_x



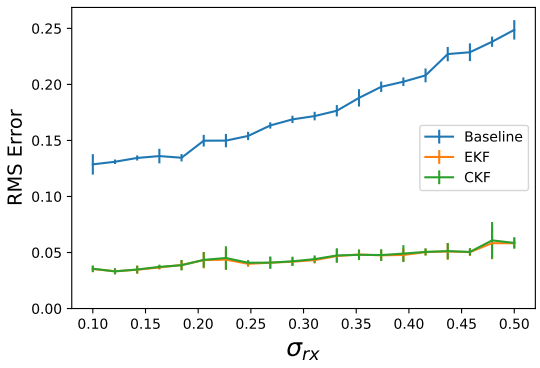
(d) Error bars when varying process noise covariance σ_y



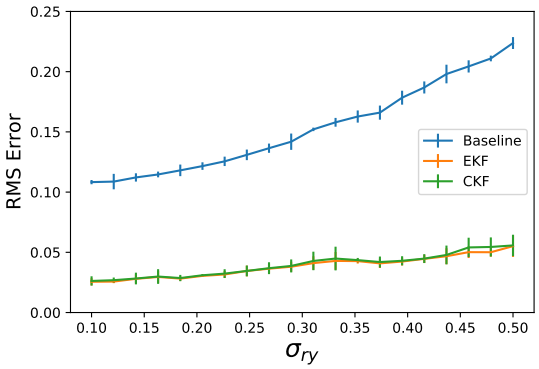
(e) Error bars when varying process noise covariance σ_ϕ



(f) Error bars when varying process noise covariance σ_v

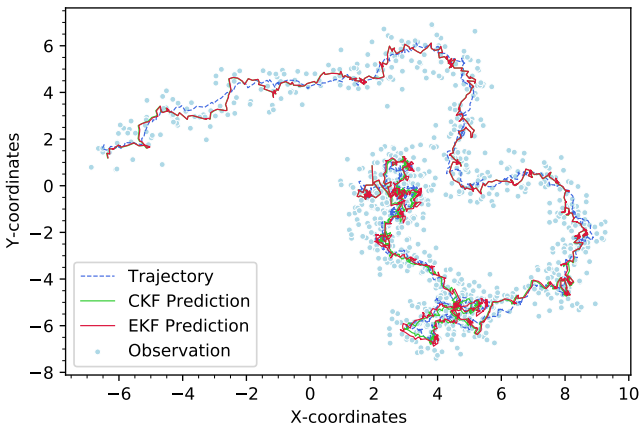


(g) Error bars when varying observation noise covariance σ_{rx}

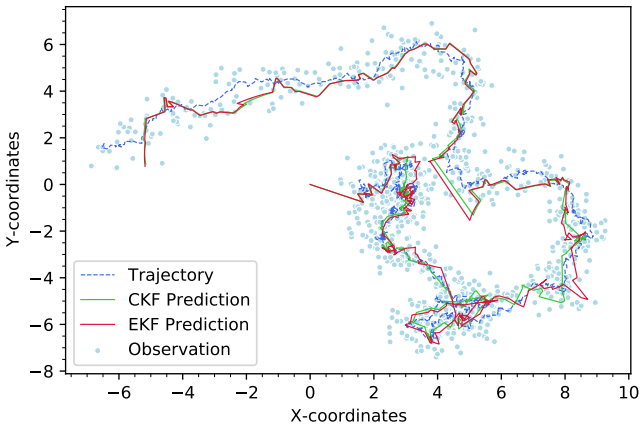


(h) Error bars when varying observation noice covariance σ_{ry}

Fig. 2. Standard Error Representations when varying individual parameters of proposed models.



(a) With no corruption.



(b) With 80% corruption.

Fig. 3. Estimated trajectories for observed data points.

curacy. We presented two different models based on Recursive Bayesian filtering. We experimented the proposed methods on sensor data streams by introducing varying amounts of noise and erasure to mimic data corruption and loss, respectively. We used localization coordinates calculated using data from a LIDAR, GPS module and orientation information provided by a gyroscope. The obtained results clearly suggest that our proposed method works well for estimating corrupted data, even when up to 80 % of incoming data is deemed unusable. Furthermore, experiments on different platforms achieved low execution latency, consumed minimal CPU resources and drew moderate current, yielding strong candidacy for sensor data estimation on computationally limited units. In future, we will take a closer look at multi-sensor systems and coordinated prediction methodologies, both on-board robots and at the Edge, to improve data prediction in more compromised circumstances. Furthermore, use of Monte Carlo methods such as particle filtering as a Bayesian statistical inference tool would be an intuitive approach for missing data imputation in resource-constrained devices.

ACKNOWLEDGMENT

This research work is supported by Academy of Finland (Grant No. 328755).

REFERENCES

[1] M. A. K. Bahrin, M. F. Othman, N. H. N. Azli, and M. F. Talib. Industry 4.0: A Review on Industrial Automation and Robotic. *Jurnal Teknologi*, 78, 2016.

[2] C. Li, Y. Ma, S. Wang, and F. Qiao. Novel industrial robot sorting technology based on machine vision. In *9th International Conference on Modelling, Identification and Control (ICMIC)*, pages 902–907, 2017.

[3] A. M. Kabir, K. N. Kaipa, J. Marvel, and S. K. Gupta. Automated Planning for Robotic Cleaning Using Multiple Setups and Oscillatory Tool Motions. *IEEE Transactions on Automation Science and Engineering*, 14(3):1364–1377, 2017.

[4] A. Vick, D. Surdilovic, A. K. Dräger, and J. Krüger. The Industrial Robot as Intelligent Tool Carrier for Human-Robot Interactive Artwork. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pages 880–885, 2014.

[5] Y. Shao and Z. Chen. Reconstruction of big sensor data. In *2nd IEEE International Conference on Computer and Communications (ICCC)*, pages 1–6, 2017.

[6] H. Kang. The Prevention and Handling of The Missing Data. *Korean J Anesthesiol*, page 402–406, 2013.

[7] V. K. Sarker, J. P. Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund. Offloading SLAM for Indoor Mobile Robots with Edge-Fog-Cloud Computing. In *1st International Conference on Advances in Science, Engineering and Robotics Technology*, pages 1–6, May 2019.

[8] I. Izonin, N. Kryvinska, R. Tkachenko, and K. Zub. An Approach towards Missing Data Recovery within IoT Smart System. *Procedia Computer Science*, 155:11–18, 2019.

[9] I. Ullah, S. Qian, Z. Deng, and J. Lee. Extended Kalman Filter-based Localization Algorithm by Edge Computing in Wireless Sensor Networks. *Digital Communications and Networks*, 2020.

[10] B. Fekade, T. Maksymyuk, M. Kyryk, and M. Jo. Probabilistic Recovery of Incomplete Sensed Data in IoT. *IEEE IoT Journal*, PP:1–1, 2017.

[11] T. Peng, S. Sellami, and O. Boucelma. IoT Data Imputation with Incremental Multiple Linear Regression. *Open Journal of Internet of Things (OJIOT)*, 2019.

[12] I. Lujic, V. De Maio, and I. Brandic. Adaptive Recovery of Incomplete Datasets for Edge Analytics. In *IEEE 2nd International Conference on Fog and Edge Computing (ICFEC)*, pages 1–10, 2018.

[13] M. Velas, M. Spanel, M. Hradis, and A. Herout. CNN for IMU Assisted Odometry Estimation Using Velodyne LiDAR. In *IEEE International Conference on Autonomous Robot Systems and Competitions*, 2018.

[14] C. Sazara, R. V. Sazara, and M. Cetin. Offline reconstruction of missing vehicle trajectory data from 3D LIDAR. *CoRR*, 2017.

[15] N. Vijayakumar and B. Plale. Prediction of Missing Events in Sensor Data Streams Using Kalman Filters. In *1st Int'l Workshop on Knowledge Discovery from Sensor Data*, pages 1–9, 2008.

[16] G. Premsankar, M. Di Francesco, and T. Taleb. Edge computing for the internet of things: A case study. *IEEE Internet of Things Journal*, 5(2):1275–1284, 2018.

[17] J. Zhang and S. Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *Robotics: Science and Systems X*, 2014.

[18] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[19] J. Durbin and S. J.Koopman. *Time Series Analysis by State Space Methods*. Oxford University Press, 2 edition, 2012.

[20] Simo Srkk. *Bayesian Filtering and Smoothing*. Cambridge University Press, USA, 2013.

[21] Haran Arasaratnam and Simon Haykin. Cubature Kalman Filters. *Automatic Control, IEEE Transactions on*, 54:1254 – 1269, 07 2009.

[22] Raspberry Pi Foundation. Products. [Online] Available: https://www.raspberrypi.org/products/. Accessed: Mar. 20, 2020.

[23] Aaeon. UP-GWS01. [Online] Available: https://www.aaeon.com/en/p/tiny-gateway-system-with-upboard. Accessed: Apr. 5, 2020.