Optimal Sensor Placement in a Smart Factory for Digital Twin Generation on Mobile Robot Movement

Ataulhaq Akbar
Computer Engineering
Rutgers University
New Brunswick, NJ, USA
aa1856@scarletmail.rutgers.edu

Abstract

In recent years, there has been a growing interest in the use of sensors for tracking and localization of mobile robots in smart factories. Optimal sensor placement is a key factor in the design and operation of these systems, as it can significantly affect the accuracy and reliability of the tracking and localization. This paper will investigate the problem of optimal sensor placement in a smart factory for mobile robot tracking using the MERLSense dataset. Optimal Sensor placement will be achieved by using PCA, K means clustering and Kalman filters to detect and remove a group of sensors that don't contribute in a significant manner.

Introduction

Optimal sensor placement refers to the process of selecting the best locations to install sensors in a system in order to maximize the performance of the system. This is often a critical issue in the design and operation of systems that rely on sensors for monitoring and control, such as industrial process control systems, structural health monitoring systems, and environmental monitoring systems.

There are several factors that must be considered when selecting the optimal locations for sensors. These may include the spatial distribution of the quantity being measured, the sensitivity and accuracy of the sensors, the cost and availability of the sensors, and the communication and power constraints of the system.

There are several approaches that can be used to solve the problem of optimal sensor placement, including analytical methods, numerical optimization techniques, and heuristics. These approaches may be used alone or in combination, depending on the specific requirements and constraints of the system.

Overall, the goal of optimal sensor placement is to select the locations for sensors that will provide the most accurate and reliable measurement of the quantity being monitored, while also taking into account the cost and other constraints of the system. This can help to improve the performance and reliability of the system, and can also reduce the cost and complexity of the sensor network.

Motivation and Background on Digital Twins

Digital twins are virtual representations of physical systems or processes that can be used to simulate and analyze the behavior of those systems or processes. The motivation for using digital twins is to improve the understanding and control of complex systems, and to enable more efficient and effective design, operation, and maintenance of those systems.

There are several key benefits of using digital twins, including:

1. Improved understanding of complex systems: Digital twins allow users to

visualize and analyze the behavior of a system in a virtual environment, which can help to improve the understanding of how the system works and how it may respond to different conditions or inputs.

2. Enhanced prediction and decision-making: Digital twins can be used to simulate the behavior of a system over time, which can help to predict its future performance and identify potential issues or problems. This can enable more effective decision-making, such as by identifying the optimal operating conditions or maintenance schedule for a system.
3. Reduced costs and risk: By using digital twins to test and optimize the design and operation of a system before it is built or deployed, it is possible to reduce the costs and risks associated with implementing the system. Digital twins can also help to reduce maintenance costs and downtime by enabling the identification and resolution of potential issues before they occur.

Overall, the motivation for using digital twins is to improve the understanding, control, and efficiency of complex systems, and to enable more effective design, operation, and maintenance of those systems.

Dataset

MERLSense is a dataset of multiview range and reflectance images of various objects and scenes. The dataset includes range and reflectance images captured using a mobile robot equipped with a multiview sensing system, as well as ground truth data and annotations for various objects and features in the images.

The multiview sensing system consists of a laser rangefinder and a camera, which are mounted on a pan-tilt unit and can be controlled to acquire images from different viewpoints. The dataset includes images captured from a variety of viewpoints and under different lighting conditions, which makes it suitable for a wide range of applications in computer vision and robotics, such as object recognition, scene reconstruction, and localization.

The MERLSense dataset was created by researchers at the Georgia Institute of Technology and the University of Michigan as part of a research project on multiview sensing and perception for mobile robots

Data Selection

For the purposes of this paper tracklets from the MERLSense dataset will be used; tracklets are sequences of range and reflectance images that are captured by the mobile robot as it moves through the environment (Figure 1). The tracklets are labeled with the ground truth pose of the robot at each time step, which includes the position, denoted by the sensorID (Figure 2), and time of activation of the sensor by the robot in the environment. In other words tracklets are spatio-temporal data.

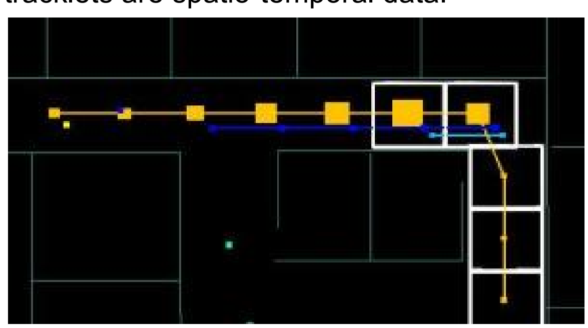

Figure 1

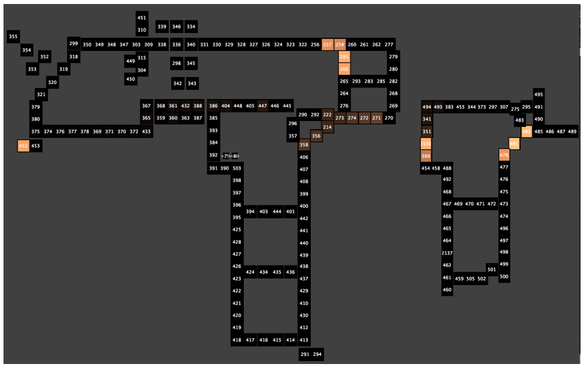

Figure 2

The tracklets are provided as a set of images and corresponding pose data, and they may contain images of various objects and scenes, including static objects, moving objects, and dynamic backgrounds. The tracklets may also contain annotations for various objects and features in the images, such as objects of interest, landmarks, and other features that may be useful for perception and localization tasks

The tracklets in the MERLSense dataset are intended to be used for evaluating and comparing algorithms for mobile robot perception and localization, and they may be used for a wide range of applications in computer vision and robotics, such as object recognition, scene reconstruction, localization and, most relevant for this paper, optimal sensor placement.

Related Work

There has been a lot of research in the field of “Optimal Sensor Placement for Mobile Robots” but many use computationally intensive frameworks such as greedy removal modeling [1], or the data they use is based on video footage [2] which can be very inaccurate in trajectory estimation.

Currently there doesn't seem to be any research that does “Optimal Sensor Placement” based on spatio-temporal data; spatio-temporal data provides far more information with respect to noise as compared to video footage and it doesn’t require very powerful computational tools to gain insights from. For this reason I picked the MERLSense dataset which contains spatio-temporal data called tracklets.

Data Preprocessing

The first thing that needs to be done is to understand how to read the MERLSense data, specifically the tracklet data. Inside the MERLSense folder there is a subfolder called “tracklets_v2” where the tracklet data is located inside of text files. Specifically for this paper “tracklets_0144.txt” will be used which is a collection of tracklets over a period of 4 months.

A jupyter notebook will be used to conduct the experiments and the tracklets text file will be converted into a dataframe for further data preprocessing.

Metrics for optimal sensor placement

There are several metrics that can be used to evaluate the effectiveness of sensor placement for detecting and tracking tracklets. Some examples of metrics that could be used include:

1. Coverage (PCA): This metric measures the percentage of the area being monitored by the sensors that is covered by the sensors. A high coverage indicates that the sensors are effectively covering a large area.
2. Sensitivity (Kalman Filters): This metric measures the ability of the sensors to detect tracklets at different distances and under different conditions/times. A high sensitivity indicates that the sensors are able to detect tracklets at a wide

range of distances and under a variety of conditions and times.

3. Detection rate (K means clustering): This metric measures the percentage of tracklets that are detected by the sensors. A high detection rate indicates that the sensors are effectively detecting a large number of tracklets.
4. Tracking accuracy: This metric measures the accuracy of the estimates of the tracklet positions and velocities produced by the sensors. A high tracking accuracy indicates that the sensors are providing reliable estimates of the tracklet positions and velocities..

By evaluating these it may be possible to determine the optimal placement of sensors for detecting and tracking mobile robots.

Feature Engineering

Feature engineering is the process of creating new features from existing data, with the goal of improving the predictive power of a machine learning model. It is a crucial step in the data preprocessing phase of any machine learning project, as the quality and quantity of features can significantly affect the performance of the model.

After loading “tracklets_0144.txt” into a dataframe feature engineering will be performed on it as per the requirements of each experiment:

1. For PCA we will subtract the “end_sensor” from the “begin_sensor” to get the distance traveled and place that value into a new column called “distance”. Likewise we will subtract “end_time” from “begin_time” to get the duration of the activation and place that value into a new column called “duration”.
2. For K means clustering we will convert “begin_time” and “end_time” into proper datetime format and extract the hour of the day that the activation occurred and put that into a new column called “hour”
3. For Kalman Filters we will also convert “begin_time” and “end_time” but instead of extracting only the hour we will also extract the month, week, minute, and second; then place each of them into a new column with their corresponding name. This is done to enable greater customization when calibrating the Kalman filter.

PCA

Principal Component Analysis (PCA) is a statistical technique that can be used to identify patterns in data and to reduce the dimensionality of the data. It is often used to identify the most important variables or features in a dataset and to reduce the number of variables that need to be considered in further analyses.

One way that PCA can be used for optimal sensor placement using tracklets is by applying PCA to the tracklet data to identify the most important features or variables that are correlated with the tracklets. For this paper the features include variables such as position (sensorID), velocity (time traveled), and relative location (distance). By identifying the most important features, it may be possible to determine which sensors are most critical for detecting and tracking the tracklets and to use this information to guide the placement of the sensors.

Implementation of PCA

To gain a clearer picture labels will be created by tagging each sensor; if the sensorID is between 256 and 290 (8th floor - East) it's label will be 1; if the sensorID is between 454 and 502 (7th floor - North) it's label will be 2; if the sensorID is between 395 and 444 (7th floor - South) the label will be 3; for all others (unknown) the label will be 0. And all sensors with the label 0 will be removed.

The number of components to result from the PCA has been programmed to be 2 so as to plot the result on the x-y axis scatter plot. And the labels will be used to color where the features came from.

Results of PCA

As can be seen in figure 3 the two largest features can be seen to come from the "8th floor - East" and the "bottom of the 8th floor", whereas the "top of the 8th floor" is barely visible. From this we can conclude that the sensors at the "top of the 8th floor" can be removed because they are not as relevant

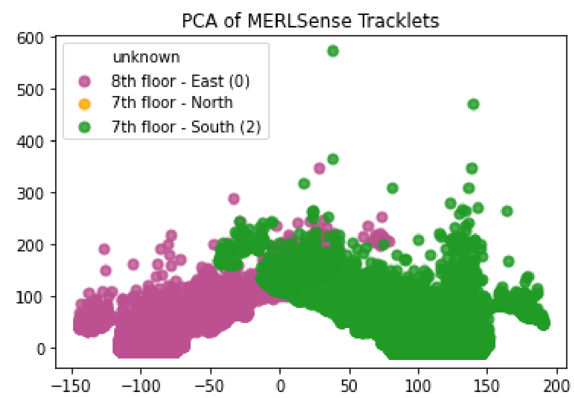

Figure 3

K means clustering

K-means clustering is a machine learning technique that is used to divide a dataset into a specified number of clusters based on the similarity of the data points within each cluster. It is often used to identify patterns or groupings within a dataset and to perform segmentation or classification tasks.

One way that K-means clustering can be used for optimal sensor placement using tracklets is by applying the K-means algorithm to the tracklet data to identify clusters of tracklets that are similar to one another in terms of certain features or variables. For this paper the features include variables such as position (sensorID), time of activation (hour).

Once the clusters of tracklets have been identified, it may be possible to use this information to guide the placement of sensors. For example, if certain clusters of tracklets are more likely to be detected by sensors placed in certain locations, sensors could be placed in these locations to optimize mobile robot detection and tracking.

Implementation of K means clustering

The number of clusters has been chosen to be 4 in order to compare if there is any correlation with the results from the PCA. For K means clustering only the first 1000 activations are fed into the algorithm in order not to overwhelm it with information. The x axis will be the hour that the sensor activation occurred and the y axis will be the sensorID of the activation.

Results of K means clustering

From figure 4 we can see that the results from K means clustering does match that with the results from PCA (figure 3). Sensors 250 to 290 form a cluster which is similar to "8th floor - East" in the PCA. Sensors 395 to 444 also form a cluster which corresponds to "7th floor - South" in the PCA. Likewise sensors 454 to 502 do not form a cluster, which is supported by the fact that their contribution in the PCA is very minimal.

Thus it can be concluded that sensors 454 to 502 “7th floor - North” can be removed because they contribute very little to the tracklet data.

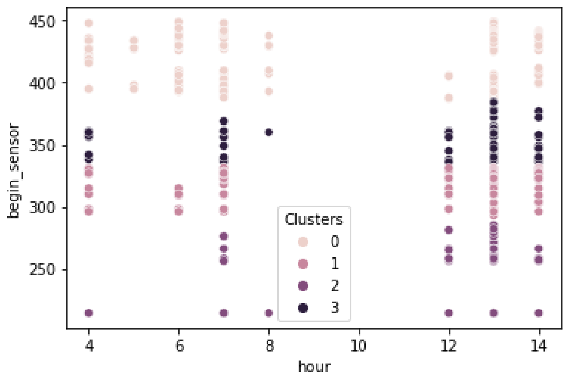

Figure 4

Kalman filters

Kalman filters are a type of mathematical tool that are used to estimate the state of a system based on a series of noisy measurements. They are often used in control systems and signal processing applications to filter out noise and to improve the accuracy of estimates of the state of the system.

One way that Kalman filters can be used for optimal sensor placement using tracklets is by using Kalman filters to process the data from the sensors that are used to detect and track the tracklets. By filtering out noise and improving the accuracy of the sensor data, Kalman filters can help to improve the accuracy of the estimates of the tracklet positions and velocities.

In terms of sensor placement, Kalman filters can be used to identify the sensors that are providing the most accurate and reliable data, which can be used to guide the placement of additional sensors. For example, if certain sensors are consistently providing more accurate estimates of the tracklet positions and velocities than others, these sensors could be prioritized for placement in locations where they are most likely to detect the tracklets.

Analysis of the time-series aspect of tracklets

Due to the time-series nature of the Kalman filter algorithm, converting the “begin_time” and “end_time” into datetime format and extracting the relevant features was necessary and useful. Despite these precautions, there have been complications due to the incredible volume of data; a huge amount of noise is present which the Kalman filter was not able to distinguish from the relevant data.

With that being said the time-series data does support our results from PCA and K-means clustering. As can be seen in figure 5 and 6 which plots “begin_sensor” on the y axis and “begin_time” on the x axis, sensors 454 to 504 have almost no activations which means we can conclude that the sensors on the “top of the 8th floor” can be removed to optimize sensor placement.

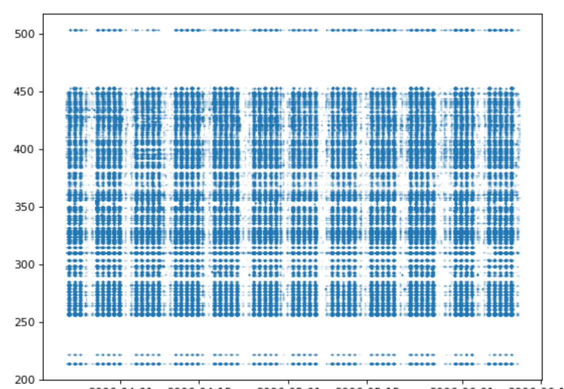

Figure 5

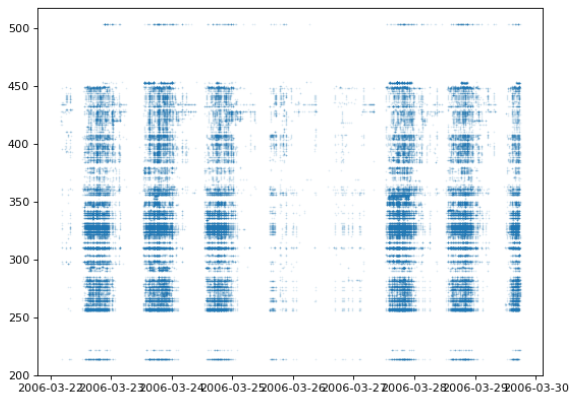

Figure 6

Challenges of implementing the Kalman filter on spatio-temporal data

While Kalman filters can be applied to a wide range of data, including spatio-temporal data, there are some challenges that need to be considered when using them for this type of data. One challenge is that spatio-temporal data often have a high degree of complexity, with multiple dependencies and correlations among different variables and across different dimensions (space and time). This can make it difficult to model the underlying dynamics of the system accurately using a Kalman filter.

Another challenge is that spatio-temporal data often have a large number of variables and dimensions, which can make the implementation of a Kalman filter computationally intensive. In addition, the performance of a Kalman filter may depend on the quality and reliability of the measurement data, which can be difficult to ensure in spatio-temporal data due to the inherent uncertainty and noise present in the data.

It is in part because of these reasons that the Kalman filter has given erratic results that do not make sense.

Conclusion

In this paper I aimed to find out where in the Mitsubishi Electric Research Lab (MERL) can the sensor placement be pruned. Using PCA and K means clustering, I have discovered that a group of sensors at the “7th floor - North” in MERL can be entirely removed. Due to technical issues I wasn’t able to conduct further experiments using Kalman filters, but through visual time-series analysis the pruning recommendation of “7th floor - North” is still supported. For future research I would recommend using distributed computing tools such as Apache Spark to speed up the graph rendering process.

Acknowledgements

Thank you to professor Ortiz for introducing me to the convoluted and interesting world of Machine Learning on IoT systems.

References

[1] D. Frisch, K. Li and U. D. Hanebeck, "Optimal Sensor Placement for Multilateration Using Alternating Greedy Removal and Placement," 2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), 2022, pp. 1-6, doi: 10.1109/MFI55806.2022.9913847.

[2] R. Stolkin and I. Florescu, "Probability of Detection and Optimal Sensor Placement for Threshold Based Detection Systems," in IEEE Sensors Journal, vol. 9, no. 1, pp. 57-60, Jan. 2009, doi: 10.1109/JSEN.2008.2008884.