Combining Different Data Sources for IIoT-based Process Monitoring

Rodrigo Gomes[1], Vasco Amaral[2][0000−0003−3791−5151], and Fernando Brito e Abreu[3][0000−0002−9086−4122]

[1] NOVA School of Science and Technology, Caparica, Portugal
ru.gomes@campus.fct.unl.pt
[2] NOVA LINCS & NOVA School of Science and Technology, Caparica, Portugal
vma@fct.unl.pt
[3] ISTAR-IUL & ISCTE-Instituto Universitário de Lisboa, Lisboa, Portugal
fba@iscte-iul.pt

Abstract. *Motivation* - Industrial internet of things (IIoT) refers to interconnected sensors, instruments, and other devices networked together with computers' industrial applications, including manufacturing and energy management. This connectivity allows for data collection, exchange, and analysis, potentially facilitating improvements in productivity and efficiency, as well as other economic benefits. IIoT provides more automation by using cloud computing to refine and optimize process controls.

Problem - Detection and classification of events inside industrial settings for process monitoring often rely on input channels of various types (e.g. energy consumption, occupation data or noise) that are typically imprecise. However, the proper identification of events is fundamental for automatic monitoring processes in the industrial setting, allowing simulation and forecast for decision support.

Methods - We have built a framework where process events are being collected in a classic cars restoration shop to detect the usage of equipment such as paint booths, sanders and polishers, using energy monitoring, temperature, humidity and vibration IoT sensors connected to a *Wifi* network. For that purpose, BLE beacons are used to locate cars being repaired within the shop floor plan. The *InfluxDB* is used for monitoring sensor data, and a server is used to perform operations on it, as well as run machine learning algorithms.

Results - By combining location data and equipment being used, we are able to infer, using ML algorithms, some steps of the restoration process each classic car is going through. This detection contributes to the ability of car owners to remotely follow the restore process, thus reducing the carbon footprint and making the whole process more transparent.

Keywords: Process activity recognition, IIoT, IoT sensors, Intrusive load monitoring, Machine learning, Indoor location, Classic cars restoration, Charter of Turin

1 Introduction

The historical importance, the aesthetics, the build quality, and the rarity are characteristic features that individually or collectively define a car as a classic. Due to the many admirers, classic cars are highly valued, sentimentally, and monetarily. Keeping the authenticity of those masterpieces, i.e., maintaining them as close as possible to when they left the factory, requires expert restoration services. Guidelines for the restoration of classic cars were proposed by FIVA [4]. They may be used for the certification of classic cars by accredited certification bodies such as the ACP[5].

Monitoring the classic car restoration process, so that pieces of evidence are recorded, is an important matter, both for managing the shop floor plan, for certification purposes, and for allowing classic car owners to follow the restoration process remotely, reducing the carbon footprint and making the whole process more transparent.

Our work aims to create and implement an IoT monitoring system to recognize the tools used and infer restoration tasks (e.g., mineral blasting, bodywork restoration, painting, painting drying, bodywork finishing stage) that a classic car is going through. We intend to use Intrusive Load Monitoring(ILM) techniques by installing energy meters in the workshop outlets and use its data in a supervised Machine Learning (ML) model for detecting the various tools used by the workers in the restoration of each classic car and combining with its location, to automatically recognize the ongoing restoration task.

This presents some challenges as classic car restoration is a complex process [2]. In the same workshop many cars may be under restoration, often each at a different stage in that process and different tools being applied. To further make detection challenging, the same tool may be shared across adjacent cars without unplugging, making power consumption-based detection imprecise.

The current work is a continuation of the one reported in [6], where a Raspberry Pi-based edge computer equipped with several sensors was attached (using magnets) to each car body on the plant shop floor, allowing to capture data on the vibrations produced by different restoring tools, as well as temperature and humidity conditions where the cars went through. Estimote BLE [6] beacons attached to the walls of the plant shop floor were also detected by the edge computer, to allow the indoor location of the car body under restoration. The raw data captured by the edge computers was then sent to the AWS [7] cloud-based platform where a ML algorithm, combining detected tools and detected position, allowed us to identify some of the tasks of the restoration process. A web application was also built to monitor the state of operation of all edge computers and beacons. Although this work presented significant progress in using IoT techniques in an industrial context (aka IIoT) for process monitoring purposes,

[4] Fédération Internationale des Véhicules Anciens (FIVA), https://fiva.org

[5] Automóvel Club de Portugal (ACP), https://www.acp.pt/classicos

[6] Bluetooth Low Energy

[7] Amazon Web Services

the developed system lacked precision in detecting the operation of some tools, as well as in indoor locating.

In Section 2, we present previously developed projects with similar objectives as ours. Then, the proposed description and architecture for our work are detailed in Section 3. In Section 4, we evaluate and discuss the obtained results. Finally, in Section 5, we present some conclusions.

2 Related Work

2.1 Intrusive Load Monitoring(ILM)

Most works on this topic, were developed in smart-home contexts. Hundred power consumption samples from three houses and the same appliances were used in [5] for feature extraction to serve as input to an Artificial Neural Network (ANN) classification algorithm. Results showed a positive overall accuracy of 95%. A second test using the ANN trained with the previous data, was executed with data from a new house, but worse results emerged, even after reducing the number of features.

An attempt to identify the different states of each appliance is described in [7]. A pre-processing with z-normalization was used for feature selection. The classification process applied a Hidden Markov Model (HMM) algorithm. Positive results were achieved with an accuracy of 94% for the test with appliances in the training set, and 74% for other appliances. An app to visualize in real-time the recognized appliance characteristics is also reported.

A prototype for collecting load data with an Arduino and an energy sensor is described in [4]. For classification purposes, different algorithms were tested, namely K-Nearest Neighbour (KNN), Random Forest (RF), Decision Tree (DT), and the neural network Long Short-Term Memory. RF presented the best results. An experimental test was also carried out to obtain the best sliding window size, i.e. the one to be used in feature extraction and classification algorithms.

An IoT architecture for ILM is presented in [1]. The features were the same as those described in [5], and three supervised learning (SL) algorithms were tried for classification. The Feed Forward Neural Network (FNNN) obtained the best accuracy results for seen data (90%).

2.2 Indoor location systems

Several techniques are used for this purpose. In some cases, the reader is linked to the object to track, and a lot of tags are dispersed through the space [8], while in others a tag is linked to the moving object, and the reader(s) is(are) fixed. In both cases, distances are calculated based on *RSSI* [8]. Trilateration can then be used for detecting the position of the moving object. For instance, *Wifi*-based indoor location can be performed through the trilateration of *RSSI*

[8] Received Signal Strength Indicator is a measurement of the power present in a received radio signal

corresponding to detected access points (APs) on a mobile phone [3]. BLE-based location technology is similar, but beacon transmitters are used instead of APs, such as in [10], where *RSSI* trilateration and fingerprinting are used. ML algorithms can be used for improved fingerprinting such as in [9], where an average estimation error of 50cm is reported.

3 Proposed System Overview

System description The detection of tools used by the workers includes the three main steps of ILM, i.e. data collection, feature extraction, and classification. Regarding data extraction, smart energy meters are installed between the tools and workshop outlets to capture the plugged tools' energy loads. Two smart meter types (Nedis Smart Plug and the Shelly Plug S.) were tested in the workshop with many available tools. They capture the electrical power in Watts(W) in real-time with a frequency of one measure per second. Both have an API to get the measured data and use *Wifi* to send this data to the internet. We chose Shelly' because its API is more straightforward, its plug is smaller (i.e. physically less intrusive), has a smaller size and its power range is enough for the tools used in the workshop (up to 2500W, compared to Nedis' 3650W).

For feature extraction, we used the technique described in [1] and [5]. A script is always running getting as input the energy sensor data and when some non-zero power value arrives, the algorithm takes the next 100 data entries (the sliding window size) and calculates all features regarding power levels and power variations. Nine features were chosen based on previous works: Maximum power value; Minimum power value; Mean power for nonzero values; Number of samples with power less than or equal to 30 W; Number of samples with power between 30 and 400 W; Number of samples with power between 400 and 1000 W; Number of samples with power greater than 1000 W; Number of power transitions between 10 and 100 W; Number of power transitions greater than 1000 W.

The group of features regarding each data window serves as input to a supervised ML model. In the ML training phase, these features are labeled with the ground-truth, i.e. the tool(target) being measured. Different algorithms are then trained with the same labelled data to find the one with the best predictions when providing unlabeled data (in the ML estimation stage). After the three ILM phases, the electrical tools used by the workers at any time of the day in the workshop are registered and available.

To complete this process of recognizing the tools, the remaining part we need to tackle is to define which car was under intervention by those same tools. As a result of a literature review about indoor location systems, some possible solutions emerged. In [6], each sensor box has a car associated, and in a real system, each vehicle would have a sensor box attached that goes with it throughout the whole workshop process. So one solution is to use a location system to track down each energy sensor, and then, as the sensor box location is available, find the closest distance between both and do the link. Another solution is to use the timestamps from the energy sensors data and match them with the detected restoration steps of the sensors boxes.

With the awareness of the tools used on each car, a combination with the information provided by the sensor boxes is made. Adding to its developed Process Identification Algorithm [6] more robustness and reliability by doing the combination of all data.

A web application is needed for system users to get feedback about the system developed and make simple changes. Some features that should be available are, for example, the list of all activated and deactivated smart plugs in the workshop, the list of all electrical tools belonging to the workshop, and the registration of more smart plugs in the system.

We decided to implement an indoor localization system for the sensor box using a ML-based BLE fingerprinting technique. The latter encompasses two phases. First is a training phase in which RSSI samples are captured throughout the entire area, and corresponding locations are used to train several ML location estimation models. Second, a validation phase where a target moves around and estimates produced by the models (a pair of x, y coordinates) are compared with ground truth measurements to assess their accuracy and choose the best one.

In our experimental setup, many BLE beacons were distributed throughout the workshop. Some measurement points were distributed across the workshop floor plan, with about 3 meters between each other. And reference points were defined, and their coordinates inside the workshop were obtained. All these points were determined through the workshop plan, as can be seen in detail in the Fig. 1 diagram.

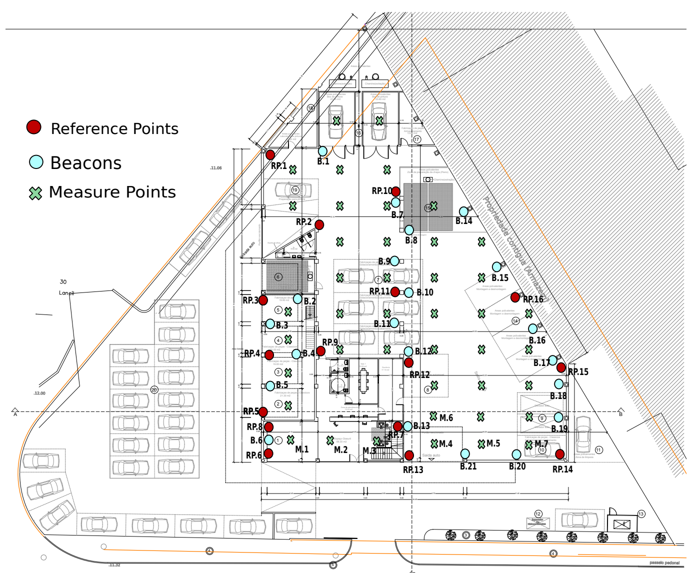

Fig. 1: Workshop floor plan with the identification of beacon's locations, reference points, and measurement points.

To obtain the coordinates of each point, a cartesian plot was placed over the floor plan of the workshop, with the axes in the same measurement scale as the available floor plan scale. Then, by going through the measurement points, using a laser distance meter, the distance to three reference points visible is pointed out, as well as the beacons *RSSI* detected values in that point and their ids. Then for each measurement point, a trilateration algorithm is used to obtain its coordinates based on the distances to the reference points and their coordinates. Having that, a ML model is trained using the coordinates of each measurement point as the target of the model and its detected beacons *RSSI* values as features. An example is shown at Table 1. During normal operation, the trained ML model, running in each sensor box/car, takes as input the detected *RSSI* values and predicts its most likely located within the workshop.

Table 1: Example of a row of the data acquired in the sensor box location method to serve as input to the ML model.

Beacon id1(*RSSI*)	Beacon id2(*RSSI*)	Beacon id3(*RSSI*)	Beacon id4(*RSSI*)	(x,y)
90.5	80.6	30.5	70.0	(9.95, 2.56)

System architecture Since we receive power consumption data from the sensors every second, we chose the open-source *InfluxDB* time series database. We installed it in a virtual machine hosted by an OpenStack platform operated by INCD.

A bucket receives electric sensors' data in the ILM part of the work, as shown in Fig. 2. The ingestion uses a *Telegraf* agent that asks the energy sensor API every second for its measurements.

A Python script performs feature extraction upon a 100 seconds sliding window of power consumption data values retrieved from the *InfluxDB* database. The results serve as input to the ML model that returns the tool predictions (i.e. which tools most likely were in use in the sliding window). Every tool prediction is then saved with its timestamp in another bucket.

To deploy the ML model after being manually trained, we saved in our server a Pickle file [9] with the trained model that is accessed every time a prediction is required.

For the new location system, the beacons data are also saved in an *InfluxDB* bucket, and the location ML model, after being trained, is also deployed in our server, using a *pickle* file, that is used in the Process Identifier Algorithm.

The Process Identifier Algorithm was developed in an *AWS Lambda* function. Still, as we want to reduce as much as possible the use of proprietary services that can later be charged, so we decided to transfer the function to a Python script running on our server, and change [6] sensor boxes data to *InfluxDB*. This way, the script queries *InfluxDB* for all the data needed to run the algorithm for identifying restoration processes, now with the help of the tools identified by the ILM module.

The Web Application front-end was implemented in [6] with *React* and communicates with the back-end via the *Amazon API Gateway*, so it is necessary

[9] Pickle is a useful Python tool that allows saving trained ML models for later use

to update and expand the front end so it can show feedback to the users about the new system features related to the predictions and smart plugs.

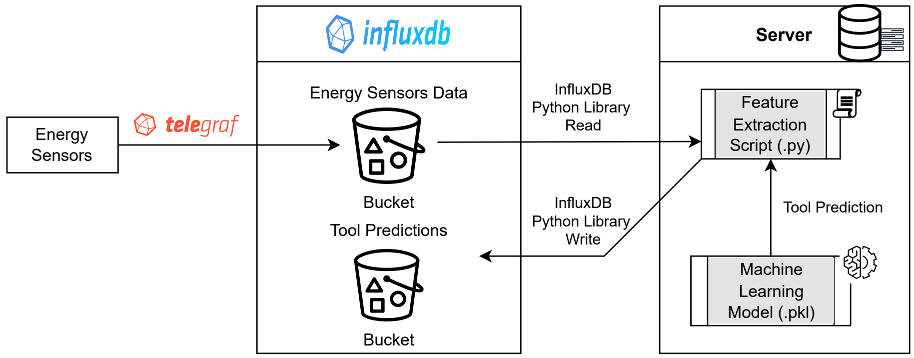

Fig. 2: Architecture of the System's Electrical Data and Feature Extraction.

This is an IoT system, so its architecture layers could be defined. We can use a five layers structure in our workshop problem and define them as follow: Physical Things - The electrical tools available in the workshop; Perception - The electrical sensors plugged in the workshop outlets; Communication Network - WiFi as this is the via that sensors use to communicate to the cloud; Middleware - All the data storage services and algorithms implemented over *InfluxDB* that interact with sensors data; Application - Web Application where the interaction with users happens.

4 Results and Discussion

To verify if predictions can be made with the electrical data and choose the best ML model, we considered the more accurate models used in the previous works detailed in Section 2. Six different ones were implemented. Random Forest(RF), K-Nearest Neighbour(KNN), Decision Tree(DT), Gaussian Naive Bayes(GNB), Gradient Boosting(GB), and the neural network algorithm Feed Forward Neural Network (FNNN). These six different supervised ML algorithms were implemented, tested, and compared.

The energy data used to test the ML models was recorded in the workshop. The electrical tools at work in the workshop, a drill, two electrical sanders, two polishers, an angle grinder, and a hot air blower, were measured for an entire afternoon with the Shelly plug.

After having the features data available, we divide it into a training set and a test set. For this, we randomly choose 70% of the entries related to each tool for training and 30% for the test set. We did it by the tool so we could have data from every available tool in the train set and the same for the test set.

In the implementation, we manually run every algorithm in a local machine and take advantage of the open-source libraries available online for ML development. For the RF, KNN, DT, GNB, and GB algorithms, we used the Sklearn library. For the FNNN, the Keras library was used. For all the algorithms, the results with or without data normalization were compared, and different parameters and hyper-parameters were used to get the best of each algorithm for a more meaningful comparison between them. However, none of the algorithms presented better results with data normalization. A feature reduction was also made and tested in the models. The most important features were the maximum power value and minimum power value. Testing with just these two features, none of the algorithms gave better results, so all the features were used to compare the algorithms.

For the FNNN model, nine input nodes were used as this is the features number, two hidden layers, and an output layer with six nodes equal to the number of different tools to predict. Then a different number of nodes in the hidden layers were tested to reach the best results.

As we can see in Table 2 the results are very positive as we were able to achieve 100% of accuracy and maximum F1-Score with two algorithms, the Random Forest and with Gradient Boosting. Also, the minimum accuracy value was 63% for the Gaussian Naive Bayes correctly predicting more than half of the tools given to the model.

Given these results, the algorithm that should be deployed to the cloud to enter the system is either Random Forest or Gradient Boosting.

Table 2: Accuracy and F1-Score of the tested ML algorithms.

Algorithm	Accuracy	F1-Score
RF	100.0%	100.0%
KNN	81.8%	78,8%
DT	81.8%	81.8%
GNB	63.6%	59.1%
GB	100.0%	100.0%
FNN	72.73%	82.0%

To test the feasibility of the sensor box location system, we took a small workshop zone to measure some data and test an ML algorithm. As described in Section 2, to get the measurement points(Ms) coordinates, we first needed to define the cartesian coordinates, relative to the workshop floor plan, of the reference points(RPs) 8,6,7. The result was the coordinates that can be seen in Fig. 3. Then all the distances from each one of the measurement point to the RPs was pointed out. The distances of M1 are shown in Fig. 3 too. Having the distances from each measuring point, a trilateration algorithm was used to obtain their coordinates. The coordinates obtained were: M1 (5.5, 2.3), M2 (9.95, 2.56), M3 (14.73, 2.53). The *RSSI* values detected in each M were pointed out.

As this location test is just the first superficial phase of the all testing procedure that must be done, just one algorithm was chosen so we could verify if predictions could be made with our acquired data. So the ML algorithm chosen to predict the sensor box spot was the K-Nearest Neighbour(KNN), as it is one of the most considered in fingerprinting-related works.

The set of *RSSI* values and the coordinates of each measurement point was used to train the ML model. Some *RSSI* values were obtained around measurement points M1, M2, and M3 and given to the model as a test set so that the spots could be predicted. Besides the small number of train and test data, the KNN achieved an accuracy of 90%.

With the results obtained, we can only realize that the tracking system could be made and expanded to the entire workshop as with just a few data received in a small zone, the ML model showed positive results. However, we must obtain more data regarding the beacons' *RSSI* values, do tests in all the spaces of the workshop, and compare different ML algorithms. Only then can we guarantee the correct functioning of the box location system.

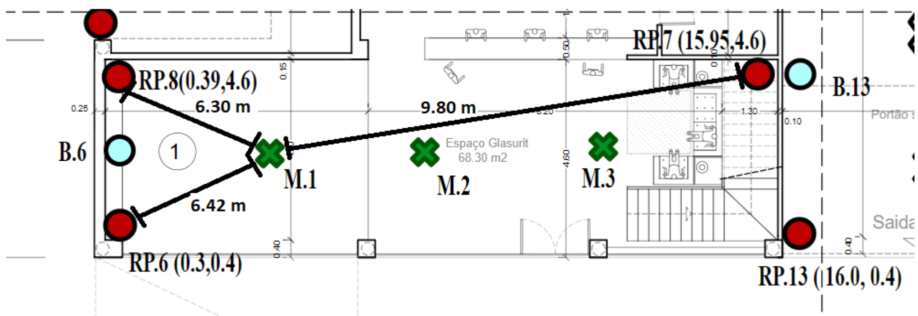

Fig. 3: Partial plan of the workshop, identifying beacons location (blue), reference points (red) and measurement points (green).

5 Conclusion and Future Work

This work presented an ILM approach for tool recognition in a workshop context and a location solution for the classic cars being restored.

With the ML algorithms tested, the results observed regarding the ILM approach demonstrate that it can clearly predict the power tools used. Missing only the identification of which car they were used. However, the data acquired to train and test the model was for the purpose of proof of concept. In future work, more data should be acquired over several days for a reliable model. The new sensor box location method should also be expanded for the whole workshop so every sensor box can be located precisely in all the floor plans.

Also, the merging with the work done in [6] should be finished by completing the web application and testing so the restoration processes can be identified and available in the application.

After completing the system, the following steps will be to create a real-time viewing of the workshop floor plan where all the detected events would be marked in the exact place. So it can be possible in an interactive way to see all the events detected by the developed IoT system.

Acknowledgement

This work was produced with the support of INCD funded by FCT and FEDER under the project 01/SAICT/2016 nº 022153, and partially supported by NOVA LINCS (FCT UIDB/04516/2020).

References

1. Franco, P., Martinez, J., Kim, Y.C., Ahmed, M.: Iot based approach for load monitoring and activity recognition in smart homes. IEEE Access **9**, 45325–45339 (2021). https://doi.org/10.1109/ACCESS.2021.3067029
2. Gibbins, K.: Charter of turin handbook. Tech. rep., Fédération Internationale des Véhicules Anciens (FIVA) (11 2018), https://fiva.org/download/turin-charter-handbook-updated-2019-english-version/
3. Khelifi, F., Bradai, A., Benslimane, A., Rawat, P., Atri, M.: A survey of localization systems in internet of things. Mobile Networks and Applications **24**(3), 761–785 (2019). https://doi.org/10.1007/s11036-018-1090-3
4. Mihailescu, R.C., Hurtig, D., Olsson, C.: End-to-end anytime solution for appliance recognition based on high-resolution current sensing with few-shot learning. Internet of Things (Netherlands) **11** (2020). https://doi.org/10.1016/j.iot.2020.100263
5. Paradiso, F., Paganelli, F., Luchetta, A., Giuli, D., Castrogiovanni, P.: ANN-based appliance recognition from low-frequency energy monitoring data. In: Proceedings of the 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2013. IEEE (2013). https://doi.org/10.1109/WoWMoM.2013.6583496
6. Pereira, D.: An automated system for monitoring and control classic cars' restorations: An IoT-based approach. Master's thesis, Monte da Caparica, Portugal (Jan 2022), http://hdl.handle.net/10362/138798
7. Ridi, A., Gisler, C., Hennebert, J.: Appliance and state recognition using Hidden Markov Models. In: Proceedings of the 2014 International Conference on Data Science and Advanced Analytics (DSAA 2014). pp. 270–276. IEEE (2014). https://doi.org/10.1109/DSAA.2014.7058084
8. Saab, S., Nakad, Z.: A Standalone RFID Indoor Positioning System Using Passive Tags. IEEE Transactions on Industrial Electronics **58**(5), 1961–1970 (2011). https://doi.org/10.1109/TIE.2010.2055774
9. Sthapit, P., Gang, H.S., Pyun, J.Y.: Bluetooth based indoor positioning using machine learning algorithms. In: 2018 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia). pp. 206–212 (2018). https://doi.org/10.1109/ICCE-ASIA.2018.8552138
10. Čabarkapa, D., Grujić, I., Pavlović, P.: Comparative analysis of the Bluetooth Low-Energy indoor positioning systems. In: 2015 12th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, TELSIKS 2015. pp. 76–79 (2015). https://doi.org/10.1109/TELSKS.2015.7357741