Motion Detection using Device-free Passive Localisation (DfPL)

Gabriel Deak*, Kevin Curran, Joan Condell and Daniel Deak†

Intelligent Systems Research Centre
University of Ulster
Magee Campus, Northland Rd.,
BT48 7JL, Derry, UK

†*S.C. Centrul de Calcul Info98 S.A.*
2 Timisoara Street,
332015, Petrosani, Romania

E-mail: *deak-g@email.ulster.ac.uk

Abstract — **The holy grail of tracking people indoors is being able to locate them when they are not carrying any wireless tracking devices. The aim is to be able to track people just through their physical body interfering with a standard wireless network that would be in most peoples home. The human body contains about 70% water which attenuates the wireless signal reacting as an absorber. The changes in the signal along with prior fingerprinting of a physical location allow identification of a person's location. This paper is focused on taking the principle of Device-free Passive Localisation (DfPL) and applying it to be able to actually distinguish if there is more than one person in the environment. In order to solve this problem, we tested classifiers such as Naive Bayes, TreeBagger in order to detect movement based on changes in the wireless signal strength.**

Keywords — **Motion Detection, Device-free Passive Localisation, Classifiers.**

I Introduction

Indoor location estimation is a crucial component in many applications. Location estimation is important for many scenarios such as asset tracking, health care, location based network access, games, manufacturing, government, logistics, industry, shopping, security, tour guides, and conference guides. Various localisation systems that can estimate the position of a person or object exist. One can select the system which offers the accuracy and precision required for a specific application.

Indoor localisation systems can be classified into active and passive systems. Location tracking techniques for active localisation require the tracked people to participate actively. The second class known as passive localisation is based on monitoring changes of characteristics dependent on peoples presence in an environment. By participating actively, we mean that a person carries an electronic device which sends information to a positioning system helping it to infer that person's position. In some cases the electronic devices can also process recorded data and send the results for further processing to an application server running the localisation algorithm. In the passive localisation case, the position is estimated based on the variance of a measured signal or video process. Thus the tracked person is not carrying any electronic devices to infer the user's position.

This work is focused on solving an extremely difficult task that is multi-occupancy detection in a passive localisation scenario. Thus the following sections will analyse one of the techniques used to deploy indoor passive localisation systems. Various DfPL systems will be presented as an introduction to indoor passive localisation. Various techniques such as Ultra-wideband (UWB), Physical Contact, Differential Air Pressure, Computer Vision, and Device-free Passive Localisation (DfPL) have been used in indoor passive localisation.

Ultra-wideband (UWB) is one of the first techniques used to deploy passive localisation systems [1]. Through-the-wall surveillance or through-wall imaging (TWI) are used to denote UWB passive systems [2, 3]. This technique has been recently used for both static and motion detection. UWB passive localisation is considered to be an extension to a technique called radio tomographic imaging due to its similarity to the medical tomographic imaging. Through-wall imaging refers to the ability of detecting and monitoring objects or people through buildings walls. This can be very useful to law enforcement agencies and can have many applications in military and civil scenarios [4]. UWB has the advantage of being able to penetrate walls. Various implementations of UWB technique have been proposed. A UWB system has the following two main components: transmitters and receivers. Short pulses are sent by a pulse generator via a horn antenna [5]. The receivers wait and monitor echoes from various objects or people.

TileTrack represents a low cost two-dimensional location estimation system based on physical contact [6]. Changes in the capacitance between transmitting and receiving electrodes (plate electrodes or wire electrodes) are monitored. The system is based on 9 floor tiles with one transmitting electrode for each tile. Each tile is 60 cm by 60 cm square-shaped made from thick chip-board with thin steel coating. The prototype used to deploy the TileTrack technique has a square tracking area with a size of 3 x 3 tiles.

AirBus estimates location based on indoors airflow disruption caused by human movement [7]. An air pressure sensor is placed within the central heating, ventilation, and air conditioning (HVAC) unit. The sensor detects pressure variations. AirBus can correctly identify an open or closed door 80% of the cases with HVAC in operation and 68% with HVAC unit switched off.

Computer vision can be considered as a DfPL system because the tracked people are not carrying any electronic devices or tags [8]. The EasyLiving project [9] is a computer vision based system which aims to transform any environment in a smart environment dependent on location information. Possible applications include switching on/off devices near to the users location, monitoring peoples behaviour and many others. The system architecture consists of three PCs (Personal Computers) and two sets of colour cameras. Each camera is connected to one PC, while the third PC is used for running the person tracker algorithms. Video processing algorithms are used to separate and track people. The system was tested with a maximum of three people in the environment. The possibility of obstructions depends on the behaviour and the number of persons.

The Device-free Passive Localisation (DfPL) [10,11] is based on monitoring the variances of the signal strength in a wireless network. The human body contains about 70% water and it is known that waters resonance frequency is 2.4 GHz. The frequency of the most common wireless networks is 2.4 GHZ, thus the human body behaves as an absorber attenuating the wireless signal [2, 4, 12–15]. This technique is the focus of our research and the remainder of the paper is based on DfPL using Wireless Sensor Networks (WSNs).

The paper is organised as follows: Section II introduces various classifiers, Section III presents the test bed and motion detection technique using the classifiers introduced in Section II. Section IV concludes the paper.

II Classifiers

The majority of applications that can be solved with classifiers and neural networks fall into the following four classes: prediction, classification, data association and data conceptualisation [16–18]. Prediction can be used to analyse stocks in the market, identify risks of various diseases in and predict location. Classification is used to identify patterns in various applications. From the localisation point of view classification can be used to identify patterns that can improve location detection or identify a location with a unique signature in a radio map fingerprint. Data association refers to data classification and detection of data containing errors. Data conceptualisation analyses inputs in order to group relationships. The most common example used nowadays in e-commerce is identifying people from a database that will most likely buy a particular product.

The focus of this paper is classification of wireless signal strength in order to detect motion in a DfPL scenario. A description of various classifiers is presented in the following subsections. The classifiers include Naive Bayes and TreeBagger. Section III presents the implementation of these classifiers with results. Future focus will implement more complex classifiers in order to solve the extremely difficult task that is multi-occupancy detection in a passive localisation scenario.

a) Naive Bayes

Naive Bayes is a simple probabilistic classifier based on Bayes Theorem which implements strong (naive) independence assumptions [19–22]. However it appears to work even when the features are not independent of one another.

The Naive Bayes classifier has two steps: training and prediction [23]. Similar to other classification methods in the training phase, Naive Bayes uses training vectors, pairs of inputs-outputs, to estimate the parameters of probability distributions. The prediction step uses unseen data and computes the posterior probabilities based on the parameters obtained in the training phase. The posterior probabilities are used to classify inputs belonging to each class.

The independence assumptions allow the classifier to compute the parameters for an accurate classification using smaller training samples compared to other classifiers. This has been shown to work even for features which are not independent of one another. The Naive Bayes classifier gives the possibility of using various distributions depending on features that needs to be identified. The following distributions are supported: normal (Gaussian), kernel, multinomial and multinomial distributions. We will not describe each distribution as this is not the focus of this paper. The input vectors we are classifying follow a normal (Gaussian) distribution thus it is not necessary to detail each distribution and we will focus on the implementation of the Naive Bayes classifier for a normal (Gaussian) distribution.

The probability model for a classifier is given by:

$$p(C|F_1, ..., F_n) \tag{1}$$

where, C represents an dependent class and $F_1, ..., F_n$ are the features variables. The Naive Bayes classifier implements Bayes theorem given by:

$$p(C|F_1, ..., F_n) = \frac{p(C)p(F_1, ..., F_n|C)}{p(F_1, ..., F_n)} \tag{2}$$

which means that the posterior probabilities are computed as follows:

$$posterior = \frac{prior \times likelihood}{evidence} \tag{3}$$

One can notice that the denominator does not depend on the class variable C, thus it is constant as the features values are known. In practice we only need to consider the numerator of Equation 3 which is equivalent to the joint probability model expressed by:

$$p(C, F_1, ..., F_n) \tag{4}$$

The joint probability model can be written:

$$\begin{aligned} p(C, F_1, ..., F_n) &\propto p(C)p(F_1|C)p(F_2|C)p(F_3|C)... \\ &\propto p(C)\prod_{i=1}^{n} p(F_i|C) \end{aligned} \tag{5}$$

Based on Equation 5 we can now rewrite conditional distribution over the class variable C which can be expressed as:

$$p(C|F_1, ..., F_n) = \frac{1}{p(F_1, ..., F_n)} p(C)\prod_{i=1}^{n} p(F_i|C) \tag{6}$$

If the data to be classified is a continuous input vector x, then then class distribution is a Gaussian distribution. The input vector is segmented into classes, and then we compute the mean and the variance of x in each class c. The mean value is usually denote by μ_c and the variance of the inputs is denoted by σ_c^2. The probabilities for the values in the input vector are computed as follows [19]:

$$p(x|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(x-\mu_c)^2}{2\sigma_c^2}} \tag{7}$$

b) TreeBagger

In statistics and machine learning ensemble methods are used to obtain better prediction performance based on various models. Fast algorithms such as decision trees, for example TreeBagger, are used with ensembles. However slower algorithms can benefit from ensembles as well.

TreeBagger can be used to solve various problems such as classifying radar returns for Ionosphere data, insurance risk rating for car imports etc [23]. It can be used in classification and regression problems. It is known that TreeBagger bags an ensemble of decision trees for either classification or regression. The method is also known as bootstrap aggregation for ensemble of decision trees [24].

In the classification case, TreeBagger uses an input vector for training. A replica of the input vector is built, known as a bootstrap replica. Every tree in the ensemble is grown on the independently drawn bootstrap replica of input data. Observations for unseen data, not included in the replica of input vector, are known as "out of bag" for this tree. An average from individual trees is computed and then used to predict the unseen data. The prediction error is determined by computing the average of the predictions over the entire ensemble and then comparing this average with the true observation values.

The bootstrap aggregation technique called bagging is a special case of a model averaging approach. Having a training set X of size n, bagging generates m new training sets X_i, each with of size $n' > n$, by sampling data from X uniformly and with replacement [21]. Sampling with replacement refers to samples where an element may appear multiple times in the one sample. In the case of $n' = n$ and if the number of values in X is large it is expected to have approximately 60% of the unique examples of X, the remainder being duplicates. A sample with these characteristics is known as a bootstrap sample. The m training sets are fitted using the m bootstrap sample obtained above and combined in the regression case by averaging the output or in the classification case by voting.

III Test Bed and Results

This section presents the experiment we conducted in order to detect motion in a DfPL scenario. First, the test bed shown in Figure 1 will be described and then Naive Bayes and TreeBagger classifiers will be used to analyse/classify motion. Finally, we compare the classification errors in Table 1.

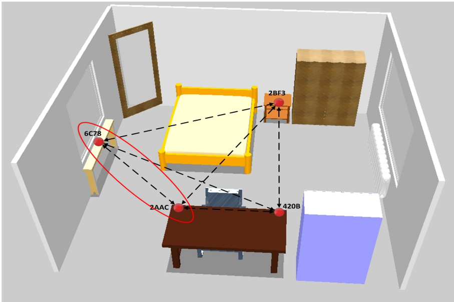

Fig. 1: The test bed with bidirectional link selected.

We have collected the data in a room of size 3.6m by 3.4m. A Wireless Sensor Network (WSN) based on four IEEE 802.14.5 Java Sun SPOT nodes and a base station was deployed in the environment. The data was recorded using a single thread collection over a period of approximately two hours. We use two data sets containing 800 values (see Figure 2). The first data set represents the training data while the second one is the test data. The nodes are broadcasting messages every 200 ms. When the messages are received, Received Signal Strength Indicator (RSSI) is added and then the messages are forwarded to the base station. However, working with a single collection thread can cause delays as the base station collects data from one node at a time.

In the case of four nodes the collection speed is good considering that we collect data from 12 links. For larger test beds multiple collection threads or more than one base station will improve the collection speed.

We have selected one bidirectional link between nodes 6C78 and 2AAC as shown in Figure 1. Both links are considered to be independent. Figure 2 shows the raw data collected from the selected links.

Both data sets are smoothed in order to filter noise. The derivative of the signal is used to normalise the data. Figure 3 shows the smoothing and derivative on one of the links. Data from the second link is processed in a similar manner. It is necessary to normalise the data in order to train and use classifiers.

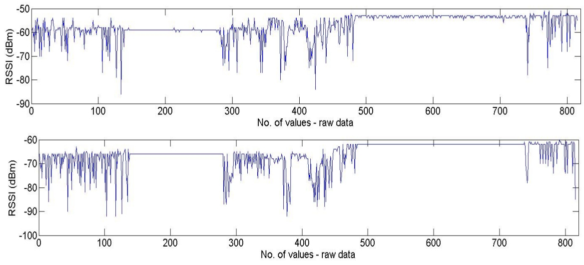

Fig. 2: Raw data from two selected links.

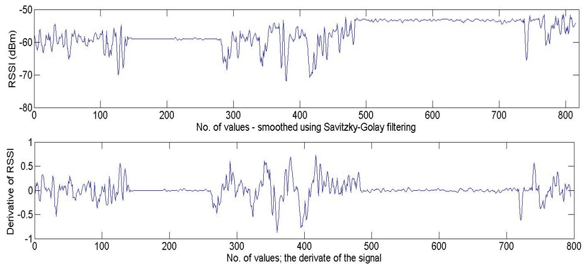

Fig. 3: Smoothing and derivative of one link.

Figure 4 shows the threshold selection considering the normalised data. The value used in this case was ± 2. Any other value above or below this threshold is considered an event which will be classified as motion. The threshold is dependent on the environment. In very noisy environments we need to modify this threshold. Thus a calibration depending on the level of noise in the environment is required.

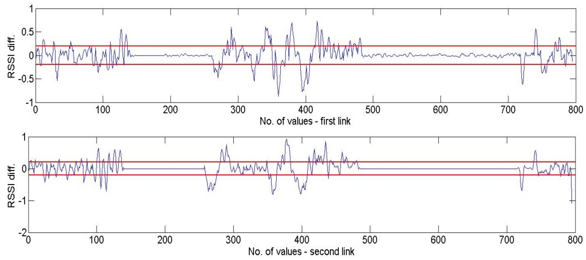

Fig. 4: Threshold selection on the derivative of the data.

We have used existent Matlab functions to analyse the two classifiers presented in Section II, Naive Bayes and TreeBagger. More details on how one can use these functions and parameters required can be found in [23].

Figure 5 shows targets vector and predicted classes using Naive Bayes with a Gaussian Distribution. One link is used to train the classifiers while the data recorded on the second link represents the test vector. The targets vector is obtained by analysing the data based on the threshold chosen above. Afterwards the test data is fed to the classifier and the output is compared with the targets vector. Due to the limited space available targets and predicted class for Naive Bayes with kernel distribution and TreeBagger will not be added. Figure 5 represents just an example on how a classifier works. As one may notice the data is classified into two classes: 'No motion' (value 1) and 'Motion' (value 2).

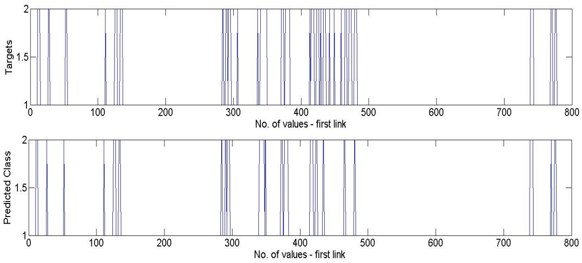

Fig. 5: Targets and predicted classes using Naive Bayes with a normal (Gaussian distribution).

Table 1 shows the errors obtained in the classification process. Considering the number of values we have used for training, we can conclude that the classifiers performed well. In the case of Naive Bayes with a Gaussian distribution the classification errors for both the train and test data ware identical which is a bit unusual. It is common to obtain a smaller error for the train data compared to the error obtained for unseen data.

Table 1: Classification errors

	Classifier	Error
Train Data	NB Gaussian	0.0352
	NB Kernel	0.0013
	TreeBagger	0.0012
Test Data	NB Gaussian	0.0352
	NB Kernel	0.0075
	TreeBagger	0.0176

The errors were computed based on confusion matrices defined as:

$$\begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix}$$

where P represents "positive", N represents "negative", T represents "true" and F represents "false".

IV Conclusion and Future Work

In this paper we presented two classifiers that enable motion detection in a DfPL scenario. Naive Bayes with Gaussian and kernel distributions, and TreeBagger classifiers were used to process wireless signal strengths in order to detect motion. The results showed the possibility of using classifiers in order to detect multi-occupancy using DfPL. We analysed a bidirectional communication between two nodes in the deployed WSN. Future work includes using timestamps in order to decide upon the number of people in the monitored environment. A person cannot affect wireless links covering different areas in the environment at the same time. Further, more complex classifiers will be analysed in order to obtain a high accuracy motion detection.

Acknowledgment

This work is supported by a Vice-Chancellor's Research Scholarship from the University of Ulster. We would like to thank Oracle for their help in supplying Java Sunspot Development Kits. It is also a pleasure to thank those who helped and supported me. I am grateful to my family for their support. I would particularly like to thank my friends Marian Baciu and Sorin Vreme for their friendship and guidance. This paper is dedicated in loving memory of my father and my friend Iulian Stoi.

References

[1] L.M. Frazier. Surveillance through walls and other opaque materials. *IEEE Aerospace and Electronic Systems Magazine*, 11(10):6 – 9, oct 1996.

[2] Lin Ma, Zhongzhao Zhang, and Xuezhi Tan. A novel through-wall imaging method using ultra wideband pulse system. In *Intelligent Information Hiding and Multimedia Signal Processing, 2006. IIH-MSP '06. International Conference on*, pages 147 – 150, dec. 2006.

[3] J. Wilson and N. Patwari. See-through walls: Motion tracking using variance-based radio tomography networks. *Mobile Computing, IEEE Transactions on*, 10(5):612 – 621, may 2011.

[4] F. Aryanfar and K. Sarabandi. Through wall imaging at microwave frequencies using space-time focusing. *Antennas and Propagation Society International Symposium, 2004. IEEE*, 3:3063 – 3066 Vol.3, june 2004.

[5] E. Gazit. Improved design of the vivaldi antenna. *Microwaves, Antennas and Propagation, IEE Proceedings H*, 135(2):89 – 92, apr 1988.

[6] M. Valtonen, J. Maentausta, and J. Vanhala. Tiletrack: Capacitive human tracking using floor tiles. In *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*, pages 1 – 10, march 2009.

[7] John Krumm. *Ubiquitous Computing Fundamentals.* CRC Press, 2010.

[8] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop on*, pages 3 – 10, 2000.

[9] Microsoft Research. Easy Living. `http://www.research.microsoft.com/`, 2011.

[10] G Deak, K Curran, and J Condell. Filters for RSSI-based measurements in a Device-free Passive Localisation Scenario. *International Journal on Image Processing & Communications*, 15(l):23 – 34, 2011.

[11] G Deak, K Curran, and J Condell. History Aware Device-free Passive (DfP) Localisation. *International Journal on Image Processing & Communications*, 16(3 - 4):21 – 30, 2012.

[12] M. Moussa and M. Youssef. Smart cevices for smart environments: Device-free passive detection in real environments. In *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on*, pages 1 – 6, march 2009.

[13] M. Youssef, M. Mah, and A. Agrawala. Challenges: device-free passive localization for wireless environments. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 222 – 229, 2007.

[14] A.E. Kosba, A. Abdelkader, and M. Youssef. Analysis of a device-free passive tracking system in typical wireless environments. In *New Technologies, Mobility and Security (NTMS), 2009 3rd International Conference on*, pages 1 – 5, dec. 2009.

[15] Lin-Ping Song, Chun Yu, and Qing Huo Liu. Through-wall imaging (twi) by radar: 2-d tomographic results and analyses. *Geoscience and Remote Sensing, IEEE Transactions on*, 43(12):2793 – 2798, dec. 2005.

[16] Harry Schwarzlander. *Probability Concepts and Theory for Engineers.* John Wiley & Sons, Ltd, Chichester, UK, January 2011.

[17] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, 4th Edition.* Elsevier, 2009.

[18] S Theodoridis, K Koutroumbas, and A Pikrakis. *Introduction to pattern recognition: a matlab approach.* Elsevier, 2010.

[19] Ke Chen. Naive Bayes Classifier Presentation, 2012.

[20] SM Ross. *Introduction to probability models.* Elsevier, 2010.

[21] Dewan Md. Farid, Mohammad Zahidur Rahman, and Chowdhury Mofizur Rahman. An Ensemble Approach to Classifier Construction based on Bootstrap Aggregation. *International Journal of Computer Applications*, 25(5):30 – 34, July 2011.

[22] WL Martinez, AR Martinez, and JL Solka. *Exploratory data analysis with MATLAB.* CRC Press, 2005.

[23] MathWorks. Matlab Statistics Toolbox User's Guide, 2011.

[24] Torsten Hothorn and Berthold Lausen. Double-bagging: combining classiffiers by bootstrap aggregation. *Pattern Recognition*, 36:1303 – 1309, 2003.