From Proximity Sensing to Spatio-Temporal Social Graphs

Claudio Martella, Matthew Dobson, Aart van Halteren, Maarten van Steen
The Network Institute, VU University Amsterdam
Email: {c.martella, m.c.dobson, a.t.van.halteren, m.r.van.steen}@vu.nl

***Abstract*—Understanding the social dynamics of a group of people can give new insights into social behavior. Physical proximity between individuals results from the interactions between them. Hence, measuring physical proximity is an important step towards a better understanding of social behavior. We discuss a novel approach to sense proximity from *within* the social dynamics. Our primary objective is to construct a spatio-temporal social graph from noisy proximity data. We address the technical and algorithmic challenges of measuring proximity reliably and accurately. Simulations and real world experiments demonstrate the feasibility and scalability of our approach. Our algorithms doubles the sensitivity of proximity detections at the cost of a slight reduction in specificity.**

I. Introduction

Widespread availability of digital traces from e-mail exchange, online social network activity, or photo sharing websites attract a growing number of scientists to the field of Computational Social Science [1]. However, these digital traces often provide a partial view of the social behavior of people in the real world. Wearable sensors further enable the emerging fields of Computational Social Science and BigData. Whether they are installed on an ad-hoc miniaturized device, or integrated into a smartphone or a smart watch, we already wear a number of sensors on a daily basis. These sensors allow scientists and practitioners to collect data about social behavior directly from the perspective of the individual, and they are capable of measuring different social signals through a variety of technologies [2], [3]. Physical proximity is one of these signals and it is the focus of this paper.

In the real world, we often get physically close to the people we interact with, for example to have a quick conversation at the bus stop, for a business meeting, or due to a queue in front of a counter. Proximity has been used as a proxy to study different types of social interactions, from the relationships in the work place [4], [5], [6] to the dynamics in a crowd [7], [8]. Depending on the specific application and goal, various aspects of physical proximity can be of interest. For instance, we can measure the frequency of interactions between members of two departments in a company or measure the precise amount of time two persons spend facing each other. Physical proximity data provides valuable insights into social behavior.

Current approaches are either based on self-reports and video cameras, or on on-body sensors. While the former present limitations of data sparsity and scalability, the latter tend to be specific to certain applications. For example, the approaches [9], [7] used to sense face-to-face interactions measure proximity only within a short distance and cover uniquely the angle of gaze. The techniques that sense colocation [10], [11] do focus on a wider area of detection, but perform the measurements with a coarse granularity over the time dimension. These approaches are suitable to study higher level social relationships, such as co-workers, friends, etc. Unfortunately, these techniques rarely deal with the specific characteristics of the device, the medium, and the environment, which cause noisy and lossy measurements. They are designed to be deployed to more controlled environments (e.g. an office, a lab, a conference hall) than those where usually crowded events take place. In this work we describe and evaluate a method that provides a reliable, fine-grained, and generally applicable proximity measurement.

A. Challenges

Designing a method to sense proximity poses a variety of challenges at different levels. At the *technological* level, the sensor must detect, in absence of physical contact, physical proximity between individuals within a specific range. Typical examples are infrared, ultrasonic, and radio-frequency sensors. This measurement should be performed consistently within a well-defined range, and as uniformly as possible over this delineated area. As the device might be worn for an extensive amount of time, it should be unobtrusive, ergonomic, and energy-efficient.

At the *processing* level, the collected detections must be filtered to overcome the inherent limitations of the technology, the medium, and the environment. These limitations provoke lossy and noisy measurements, meaning that detections might be missed or erroneously added, for example due to corruption or interference to the signal. The ability of the method to cope with these disturbances is a strong requirement. In fact, the quality of the measurement will be defined by the extent to which the method will be able to deal with these limitations.

In this paper, we propose a method to reliably measure physical proximity based on a device composed of a sensor that can detect other sensors nearby and a processing unit that can filter these detections, and a filtering algorithm. In our experiments, we used an ad-hoc device which allowed us to deal with the challenges at both levels. However, our approach is applicable to other devices as well, such as mobile phones. We dealt with the challenges that belong to the technological level in [12], [13]. In this paper we focus mostly on those that belong to the processing level.

B. Our approach

We assume that each individual wears a device with an associated unique identifier. Periodically, each sensor broadcasts its identifier, and this transmission is received by the sensors within a range d. We say that sensor u *detects* sensor

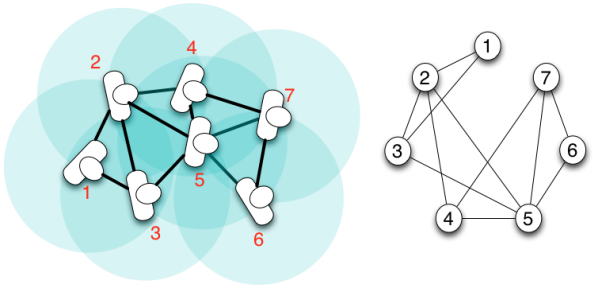


Fig. 1. A proximity graph representing a group of individuals. The blue disc represents the detection range.

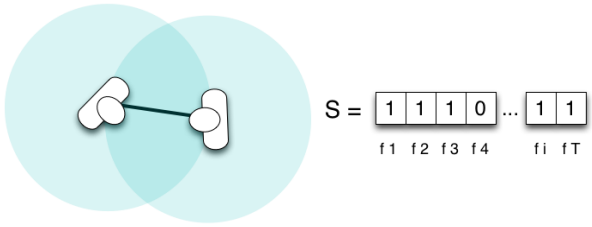


Fig. 2. A series of detections between two sensors represented through a bit string.

v at time t if it receives that particular transmission containing the identifier of v. Conceptually, each sensor collects these detections along with the time of reception t. If we model every device as a vertex, we obtain a series of what we call *proximity graphs*. Each proximity graph in this series represents a unit of time, and it contains an edge from vertex u to vertex v if sensor u detected sensor v at that specific time. In principle, the proximity graph is a directed graph, but we can drop the edge direction and consider the detection as symmetric. Figure 1 pictures an example of a proximity graph.

The proximity graph is a representation of proximity between individuals, and as such it can represent the *texture* of a crowd [14]. It can be used to study the underlying social behavior. Note that the proximity graph is a type of spatio-temporal graph, but it does not contain any location information, such as the absolute position of the individuals, or information about distance or angle of detection. An edge is "only" a boolean relationship, as it connects two individuals and it represents whether or not they were close enough to each other at a specific time. Although minimal, the evolving series of proximity graphs contains information that can be used to study systems such as epidemic models, crowd dynamics as queues and pedestrian lanes, social ties, etc. Moreover, it can be augmented with other sources of information, such as profiles or different sensor data, to correlate patterns of proximity with various aspects of social behavior.

We want to construct the series of proximity graphs over time through a number of devices. In particular, we want to perform the measurement reliably, filtering out the noisy detections and reconstructing the missing ones. To this end, we follow a *data mining* approach where we focus on higher-level analyses of the collected data instead of trying to solve the problem at a lower layer, such as by improving the sensor or the MAC protocol. This work makes the following primary research contributions. We present the design of a method based on a radio, a protocol used to create a social ad-hoc wireless network, and a processing pipeline to improve the quality of the proximity graph by reconstructing missed detections and eliminating false detections. We evaluate the reliability of the approach both in simulation and with real-world experiments.

II. Model

In this section we present the problem definition and the proposed solution, but first we proceed by introducing our model. We consider a time interval which we divide into T *frames*, each having a duration of γ time units. In this paper, γ is taken to be 250, 500, 750, or 1000 ms. Consider a series of detections $S(u,v) = \langle b_1, \ldots, b_T \rangle$ between two sensors u and v. In this case, $b_i = 1$ if and only if a detection message sent by v was successfully received by u in frame i. Hence, S can be practically represented through a bit string. Theoretically, more than one transmission can take place, but we assume that detection messages are sent regularly with a frequency of $\frac{1}{T}$. We will informally speak about a detection at time i, to mean a detection during frame b_i, and indicate frame i with bit b_i. We aggregate the detections of u about v and vice versa together in a single string, by performing a logical *OR* between the respective bit strings. This design decision was taken to overcome missing data, but a logical *AND* could also be possible, for example in case a stronger requirement of symmetry is necessary. Figure 2 shows an example of a bit string representing the proximity detections between two sensors.

A. Problem definition

Ideally, if two sensors are within a given threshold d of physical distance for an interval of time, they should be able to detect each other at every time. This would produce an ideal series S where subsequences of set bits are contiguous for the whole duration of the proximity. Extracting the edges of the proximity graphs from such a string would have a trivial solution. However, the incomplete and noisy measurement produces misdetections that cause incorrect values to be assigned to some frames. The consequence of these incorrect bit values is a number of excessive or missing edges in the proximity graphs. To overcome this, a more sophisticated solution is required. Fortunately, the nature of the measurement is *bursty*. This means that correct detections tend to appear together, and false detections tend to be isolated. This nature is inherent due to the social behavior being measured, as individuals tend to stay close for a certain interval of time, and then separate [15]. We want to identify these bursts in S. Once the bursts have been identified, we can can *set the unset bits* within the bursts, hence correcting the missed detections. Also, we can *unset the isolated bits* that are not part of any burst, hence eliminating false detections. Moreover, while minimizing the number of missed detections, we also want to minimize the number of false detections introduced by the method.

The bursty nature of the measurement implies that a detection is often surrounded by other detections within a certain window of time. This means that S is characterized by set bits surrounded by other set bits nearby. We can obtain a *smoothed* version S' of S by applying a window-based smoothing technique to S as follows. A window is defined as $window(S,i,n) = \{b_j \in S \mid j \geq i - \lfloor \frac{n}{2} \rfloor \wedge j \leq i + \lfloor \frac{n}{2} \rfloor\}$, with $n > 1$

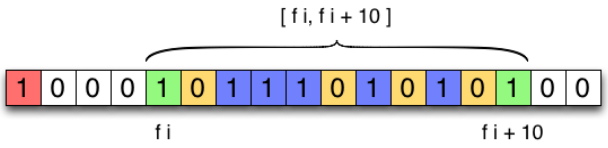


Fig. 3. A series of detections after applying density-based clustering. Blue frames are part of the core of the cluster, while green frames are density-reachable from them. The red frame represents noise and the bit in it will be unset, while the bits in the yellow frames will be set. (minPts=3, epsilon = 2)

and odd (we ignore the case where $n = 1$ as the technique has no effect). Parameter n defines the size of the window. Given a bit string $S = \langle b_1, \ldots, b_T \rangle$, we define S' as $S' = \langle b'_1, \ldots, b'_T \rangle$ where $b'_i = 1$ iif $\|\{b_j \in window(S,i,n) \mid b_j = 1\}\| > \rho$, with $\lfloor \frac{n}{2} \rfloor < i \leq T - \lfloor \frac{n}{2} \rfloor$. We define $b'_i = 0$, with $i \leq \lfloor \frac{n}{2} \rfloor \vee i > T - \lfloor \frac{n}{2} \rfloor$. Informally, we let the window *slide* over S to produce a new bit string where at each slot a bit is set if and only if there are enough set bits around the corresponding bit in the original string (including that bit). This effectively sets the unset bits in a burst and unsets the isolated ones (when $\rho > 1$). While simple and effective, the behavior of the technique depends on the choice of the two parameters n, ρ, for which no well-informed heuristics exists, hence limiting the applicability of the technique. In the next section, we propose an alternative technique that behaves similarly, but permits an estimation of its parameters. The strength of this approach lies in the consistency between the estimation technique and the semantics of the estimated parameters.

B. Density-based clustering

Given the bursty nature of S, we can use a density-based clustering algorithm and cluster dense subsequences of bits together. These clusters would correspond to the bursts of detections we are looking for, while the outliers would correspond to the noisy detections to be ignored. In particular, we propose to use DBSCAN [16]. In S, the distance between two bits b_i and b_j is defined as $dist(b_i, b_j) = |i - j|$. A bit b_i is *directly density-reachable* from another bit b_j if their distance is not more than ε (in that case it is said to be part of its ε-neighborhood) and if b_j is surrounded by a sufficient number of bits to consider them a cluster, denoted as $minPTS$. A bit b_i is called *density-reachable* from b_j if there is a sequence of bits $b_1, \ldots, b_k, \ldots, b_n$ with $b_1 = b_j$ and $b_n = b_i$ where each b_{k+1} is directly density-reachable from b_k. Two bits b_i and b_j are *density-connected* if there is a third bit b_k that is density-reachable from both. A cluster is a subset of S composed of mutually density-connected bits, plus any bit that is density-connected to them. A point that is not part of a cluster is considered noise. Figure 3 shows the result of the clustering algorithm on a series of detections between two sensors.

Analysis. One can see a correspondence between the window-based smoothing technique and the density-based clustering technique. In both cases, we identify bursts by thresholding the number of set bits within a certain "radius", and we set the unset bits in the burst. In the former, we set a bit in S' if there are at least ρ set bits within a certain distance, defined by the window size n, in the original string. In the latter, we set the unset bits between density-reachable bits, which are similarly identified based on the minimum number of set bits $minPTS$ within an ε-neighborhood. For this reason, one would expect to obtain similar results through the two techniques. Note that the application of DBSCAN to our model affords a more efficient implementation than general DBSCAN. The runtime complexity of general DBSCAN is $O(N^2)$, as each data point needs to be evaluated against all other data points to find its ε-neighbours. This runtime complexity can be reduced to $O(NlogN)$ by means of an accelerated indexing structure such as a spatial index. In our model based on bit strings, collecting the ε-neighborhood of a bit has a constant cost as it requires to evaluate the bits within the ε-distance. As ε is fixed and evaluating a bit in a string has a constant cost, the cost of DBSCAN applied to our model is linear to the number of bits in the string. This makes the runtime cost of the density-based clustering technique comparable to the cost of the window-based smoothing technique.

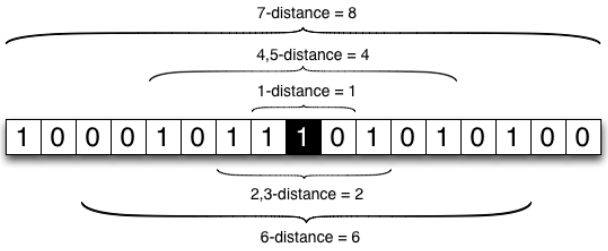


Fig. 4. The k-distance of b_i for different values of k. The black frame indicates detection b_i.

DBSCAN does not need any training phase and it does not require to define a number of clusters in advance. However, it does require that the data corresponds to a single density distribution function. In DBSCAN, the expected density distribution is defined through the parameters ε and $minPTS$, and the algorithm is very sensitive to them. As we do not know the density distribution in advance, we need to estimate these parameters. We propose such a technique in the next section.

C. K-nearest neighbors analysis

To study the density distribution of S we propose to compute the distance within which a bit will find at least k bits, by also taking into account that more bits can be at the same distance. By combining this notion of distance with the number of bits within that distance, we can estimate density. In particular, we propose to compute the distance of the *k-Nearest Neighbors* (k-NN), denoted as k-distance(b_i). Given the definition of $dist(b_i, b_j)$, the k-distance of detection b_i is the δ such that:

(i) $\|\{b^* \mid b^* = 1 \wedge dist(b_i, b^*) \leq \delta\}\| \geq k$.
(ii) $\|\{b^* \mid b^* = 1 \wedge dist(b_i, b^*) < \delta\}\| < k$.

$\|\{b_j, \ldots, b_k\}\|$ denotes the cardinality of the set and hence the number of detections. Figure 4 shows the result of the k-NN analysis run on the series used in Figure 3.

Computing the k-distance of each detection in the string permits an overview over the relationship between distance and neighborhood size. These in turn relate to the two parameters of DBSCAN ε and $minPTS$. In fact, the k-distance of b_i is the ε necessary to collect *at least* k other detections around it. As such, it can be used to select the correct ε that satisfies $minPTS$ for the detections in the bursts. Usually, $minPTS$ is known in advance and is bounded to the definition of a cluster and the

noise in the data, and it can be used as a basis for the value of k. With large enough clusters, the detections should present a common k-distance that corresponds to a representative density of the clusters, except for the noise or the members of clusters smaller than k. Selecting an ε that is close to this common k-distance guarantees that DBSCAN will cluster the detections together with the others within the k-distance.

The *k-distance plot*[1] is a means to study a k-distance distribution. This plot is constructed as follows. For a particular value of k, the k-distance for each point is computed. These values are sorted in ascending order. The sorted list of points is plotted on a Cartesian plane, by using the k-distance value as the y coordinate and the position in the sorted list as the x coordinate. We will use and show k-distance plots in the evaluation sections, to select the DBSCAN parameters. Note that in this work we selected ε manually to show the impact of the different choices on the behavior of the method. However, approaches have been proposed to select ε automatically through a numerical analysis of the k-distance plot[17].

D. Proposed solution

Our proposed solution to extract a series of proximity graphs from a series of detections consists of the following steps. The detections are collected with their timestamps from each sensor, and they are aggregated and converted to a bit string representation. The k-NN analysis is performed on all the bit strings together, by computing the k-distance of each detection in each string, and sorting all these values together into a single list. At this point, depending on the technology being used to perform the measurement, usually a range of valid values of $minPTS$ is known in advance and it is used as k to compute the k-distances. Finally, each bit string is clustered with the representative k-distance as ε and k as $minPTS$. The extracted clusters and noise are used to correct the misdetections, and finally the series of proximity graphs can be generated.

III. Evaluation in Simulation

In this section we present the evaluation in simulation of our proposed solution. The objective of this evaluation is to validate whether the method can construct correctly a proximity graph in a controlled environment, in the face of both data loss and noise.

A. Methodology

For a scenario, we selected a social environment where individuals walk around and form groups dynamically, to simulate a social location such as a museum or a conference. Here we describe the different components we used to simulate our scenario and the metrics we used to evaluate the performance of our solution.

Mobility patterns. We generated 10 mobility patterns with BonnMotion [18] using the Reference Point Group Mobility (RPGM) model [19] for a simulated crowd of 100 individuals moving for 60 minutes inside of a square arena of 25x25m. Each pattern was generated through a different random seed, but with the same set of parameters. The generated patterns lead to a list of (x,y) coordinates describing where each individual will be at any given time. The individuals move with a speed of between 0.3 and 1.2 meters per second, but they can pause for up to 60 seconds. They form groups of a minimum size of 2 individuals and a maximum of 4, and when an individual comes in proximity of a different group it will join the new group with a probability of 0.9. Each group has a semi-circular shape with an average diameter of 2 meters. These parameters were chosen to simulate realistically the size, the inter-distance and the speed of movement of groups of individuals in the desired scenario, while maximizing dynamicity.

Simulator. Each individual wears a simulated device. We simulated the devices in accordance with the devices we used for our real world evaluation [12]. These nodes have an Atmel ATXMega128 CPU with 8k of RAM, 128k of Flash memory, and a Nordic nRF24L01+ wireless radio. To simulate the wireless network we used the OMNeT++[2] simulation engine with the MiXiM[3] framework for wireless and mobile network, the de-facto simulation environment for mobile ad-hoc and sensor networks [20]. The devices form an ad-hoc wireless network through an extension to the GMAC protocol [13]. In perfect conditions, the simulated transmission range is some 3.5 meters. For our purposes, we influenced these conditions in multiple ways through the simulator and protocol parameters.

Metrics. We extracted a series of proximity graphs directly from the coordinates of each mobility pattern and we consider these as our *ground truth*. We used a distance range of 3.5 meters to construct the proximity graph, as this is the ideal transmission range of our simulated devices. To measure the performance of our proposed solution we compute the number of:

- *false positives* (FP): detections that are present in the measured proximity graph but not in the ground truth.
- *false negatives* (FN): detections that are present in the ground truth but not in the measured proximity graph.
- *true positives* (TP): detections that are present in the measured proximity graph and in the ground truth.
- *true negatives* (TN): detections that are missing in the measured proximity graph and in the ground truth.

We use these tests to compute two statistical measures of performance for binary classification tests. Sensitivity can be used to measure the ability of the test to identify positive results and is defined as $sensitivity = \frac{TP}{TP+FN}$. Specificity can be used to measure the ability of the test to identify negative results and is defined as $specificity = \frac{TN}{TN+FP}$. Intuitively, they measure the ability of the sensor to detect correctly proximity and the absence of it, respectively.

B. Setups

According to our approach, each sensor periodically broadcasts its ID through the radio, and a reception of such a transmission is considered as a detection. There are three main parameters we used to influence the behavior of the wireless network, and as such of the detections. We evaluated each of the 16 combinations of the three parameters over all the 10

[1] In the literature this is usually called *k-distance graph*, but we use the term *plot* to avoid confusion.

[2] http://www.omnetpp.org

[3] http://mixim.sourceforge.net

	Raw measurement		Density-based		Window-based	
Setup	Sensitivity	Specificity	Sensitivity	Specificity	Sensitivity	Specificity
100slots0SNR1000ms	0.972 ± 0.002	0.999838 ± 0.000008	0.973 ± 0.002	0.99970 ± 0.00003	0.982 ± 0.001	0.99874 ± 0.00004
100slots0SNR750ms	0.972 ± 0.002	0.999876 ± 0.000008	0.972 ± 0.002	0.99983 ± 0.00001	0.982 ± 0.001	0.99894 ± 0.00004
100slots0SNR500ms	0.972 ± 0.002	0.999926 ± 0.000003	0.973 ± 0.002	0.999915 ± 0.000003	0.982 ± 0.001	0.99926 ± 0.00003
100slots0SNR250ms	0.973 ± 0.002	0.999986 ± 0.000001	0.973 ± 0.002	0.999986 ± 0.000000	0.978 ± 0.001	0.99979 ± 0.00001
100slots10SNR1000ms	0.856 ± 0.008	0.999997 ± 0.000001	0.857 ± 0.008	0.999989 ± 0.000001	0.876 ± 0.007	0.999949 ± 0.000004
100slots10SNR750ms	0.856 ± 0.008	0.999998 ± 0.000001	0.857 ± 0.008	0.999997 ± 0.000001	0.872 ± 0.007	0.999985 ± 0.000002
100slots10SNR500ms	0.857 ± 0.008	0.999999 ± 0.000001	0.857 ± 0.008	0.999999 ± 0.000001	0.867 ± 0.008	0.999995 ± 0.000001
100slots10SNR250ms	0.857 ± 0.008	0.999999 ± 0.000000	0.857 ± 0.008	0.999999 ± 0.000000	0.862 ± 0.008	0.999999 ± 0.000000
4slots0SNR1000ms	0.54 ± 0.01	0.996899 ± 0.000001	0.938 ± 0.006	0.99746 ± 0.00007	0.932 ± 0.006	0.9971 ± 0.0001
4slots0SNR750ms	0.53 ± 0.01	0.999997 ± 0.000000	0.946 ± 0.003	0.99941 ± 0.00006	0.927 ± 0.006	0.9974 ± 0.0001
4slots0SNR500ms	0.53 ± 0.01	0.999998 ± 0.000000	0.95 ± 0.01	0.99975 ± 0.00003	0.921 ± 0.008	0.9979 ± 0.0001
4slots0SNR250ms	0.51 ± 0.02	0.999999 ± 0.000001	0.93 ± 0.01	0.99995 ± 0.00001	0.892 ± 0.016	0.9988 ± 0.0001
4slots10SNR1000ms	0.33 ± 0.01	0.999999 ± 0.000000	0.79 ± 0.02	0.99961 ± 0.00008	0.799 ± 0.009	0.9987 ± 0.0001
4slots10SNR750ms	0.33 ± 0.01	0.999999 ± 0.000000	0.797 ± 0.008	0.99961 ± 0.00007	0.79 ± 0.01	0.99923 ± 0.00005
4slots10SNR500ms	0.32 ± 0.01	0.999999 ± 0.000000	0.809 ± 0.008	0.99988 ± 0.00003	0.778 ± 0.009	0.99971 ± 0.00002
4slots10SNR250ms	0.31 ± 0.01	0.999999 ± 0.000001	0.79 ± 0.01	0.999987 ± 0.000009	0.74 ± 0.01	0.99997 ± 0.00001

TABLE I. COMPARISON OF THE RESULTS OF THE RECONSTRUCTION OF THE PROXIMITY GRAPH FROM RAW DATA AND WITH THE TWO TECHNIQUES. THE RESULTS PRESENT THE AVERAGE OUTCOME OVER THE 10 MOBILITY PATTERNS FOR EACH COMBINATION. FOR THE DENSITY-BASED CLUSTERING TECHNIQUE, THE CHOSEN PARAMETERS WERE $minPTS = 1$, AND $\varepsilon = 5, 8, 8, 14$ FOR EACH GROUP OF SCENARIO RESPECTIVELY. THE VALUES FOR ε WERE CHOSEN BY STUDYING THE K-DISTANCE PLOT FOR EACH GROUP. FOR THE WINDOW-BASED SMOOTHING TECHNIQUE, THE CHOSEN PARAMETERS WERE $\rho = 1$, AND $n = 3, 5, 7, 9$ FOR EACH GROUP OF SCENARIO RESPECTIVELY. THE VALUES FOR n WERE CHOSEN EXPERIMENTALLY TO OBTAIN COMPARABLE RESULTS WITH RESPECT TO BOTH SENSITIVITY AND SPECIFICITY.

mobility patterns, for a total of 160 simulation runs.

First, we can choose the number of *slots* used by the GMAC layer for the broadcasts. In GMAC, time is divided into a sequence of frames, and each frame is composed of a number of slots. The slot allocation algorithms of GMAC are based on Slotted Aloha [21], and are used to share access to the wireless medium. At a high level, the main difference between GMAC and Slotted Aloha is that GMAC exploits duty cycling. Slotted Aloha is an always-on protocol, whereas GMAC turns the radio on only for a small percentage of the total slots in order to save energy. Slots in which the radio is powered up are known as active slots, in contrast to inactive, or idle slots, where the radio is powered down. By controlling the number of active slots and their allocation strategy, we can control the number of "collisions" when multiple sensors are within the range d, and hence of loss of detections. In particular we experimented with the **4slots** setup, where four randomly allocated active slots are used (introducing a large number of collisions), and the **100slots** setup, where each node has a slot assigned statically according to its ID (ideally avoiding collisions completely).

Second, *radio irregularity* can introduce variations in packet reception and it can be caused by factors related to the device and to the medium. Examples of the first ones include the antenna type, the transmission power, antenna gains, receiver sensitivity, receiver threshold and the Signal-to-Noise Ratio (SNR). Factors related to the medium are the medium type, the background noise, and some other environmental factors, such as the temperature and obstacles within the propagation media [22], [23]. While simulating some of the medium factors is more complicated, most of these factors are considered explicitly by the simulation engine [24], including those specific to the characteristics of our devices. Here, we concentrate in particular on the SNR, relaxing the assumption of a uniform disc of transmission and reception. We experimented with the **10SNR** setup, where a SNR threshold of 10 is used, and the **0SNR** setup, where the threshold is completely ignored.

Third, the *frame length* is expressed in time units, and it defines the periodicity with which nodes broadcast their ID. It defines the rate at which detections are collected. This rate acts as a sampling rate for our sensor, and it should be chosen according to the dynamicity of the behavior to be measured. For instance, it is desirable to use a frame length a number of times shorter than the minimum amount of time two sensors can be in each other's proximity. This allows to obtain potentially multiple detections during that interval and overcome possible loss of detections. We will show that increasing the sampling rate is very important to filter out noise. On the other hand, a higher sampling rate increases energy consumption, as the radio is used more frequently. There is a trade-off between energy efficiency and the sensitivity of the instrument. We experimented with the **1000ms**, **750ms**, **500ms**, **250ms** setups, which use a frame length as their name suggests.

C. Results

We ran the 160 runs of simulation and computed the sensitivity and specificity of the proximity graphs extracted from the raw measurements, and obtained through window-based smoothing and density-based clustering. Table I shows the results. For the raw measurement, the sensitivity simply measures the number of recorded detections, as it represents the naive approach that uses only set bits for the construction of the edges. For the raw measurement, two things should be noted. First, changing the number of slots and the allocation algorithm causes nearly 50% of the detections to be missed due to collisions. Second, the change in SNR causes an additional miss of detections between 10% and 20%, depending on the number of slots used.

Looking at the results of our approach, one can notice that the two techniques do provide very similar results, as expected.

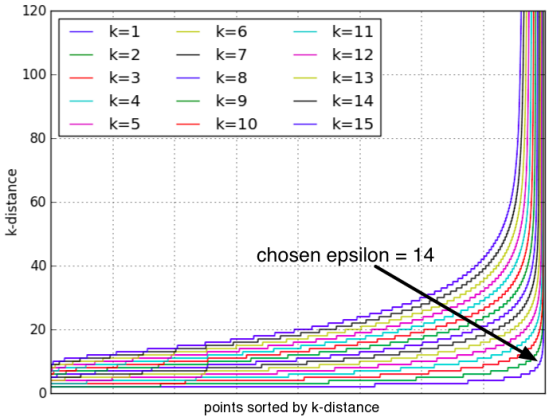


Fig. 5. k-distance plot for scenario 4slots10SNR250ms. For readability, the plot is limited to points with a k-distance smaller than 120. The k-distance is expressed in frames, hence each frame here represents 250ms.

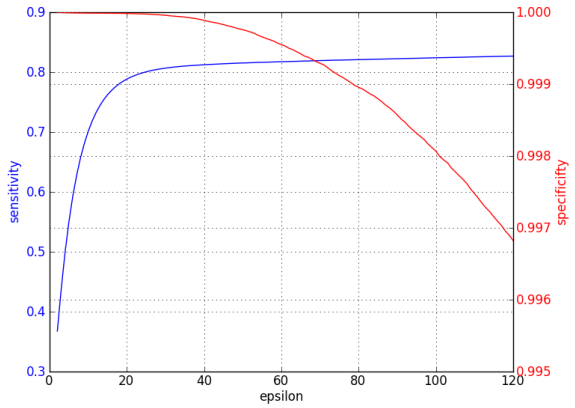


Fig. 6. Relationship between ε, sensitivity and specificity for scenario 4slots10SNR250ms and minPTS = 1.

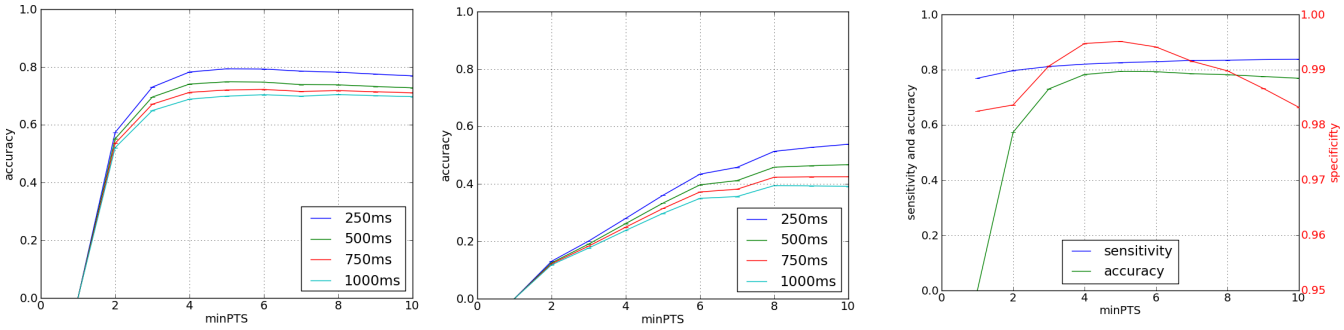
They are able to reconstruct the detections missed due to collisions consistently, i.e. a very low standard deviation across the different runs for the same setup, and effectively, i.e. more than doubling the sensitivity in all the 4slots scenarios. On the other hand, our approach cannot improve the results for the 100slots scenarios. While this is understandable for the almost perfect conditions of the 100slots0SNR setups, one would imagine that our approach would reconstruct some of the data missed due to the SNR threshold. Moreover, the improvement introduced by our approach to both the 4slots scenarios brings the sensitivity close to the results for their corresponding 100slots scenarios, suggesting that most of the reconstructed data comes from the data missed due to collisions only. The reason is to be found in the very consistent and fast drop of coverage at the borders of the detection disc due to the SNR threshold in simulation. In a real deployment, the effect of the SNR threshold is to increase the probability of a radio transmission to be missed by another radio, as the sender reaches the border of the disc of the receiver. This probability increases quickly as the radio reaches the disc border, but still some transmissions should be received. What we experienced in simulation is that the radio passed from receiving everything to nothing. This basic on/off behavior is to be expected in absence of interference [25].

In effect, the SNR threshold causes the detection range to simply shrink. This hypothesis is confirmed by the absence of improvement when the frame length is decreased in the 10SNR scenarios. In fact, with a higher detection rate and without this on/off behavior, one would expect a higher chance that at least a few detections would make it through the disc border. Fortunately, this behavior is expected only in simulation, as it has been shown that the behavior of real links in low-power wireless networks, such as ours, deviates to a large extent from the ideal binary model used in several simulation studies. In particular, there is a large transitional region in wireless link quality that is characterized by signicant levels of unreliability and asymmetry [26]. In the remainder of this paper, we stick to the density-based clustering technique only, as it allows for an estimation of its parameters through the K-nearest neighbours analysis.

Choosing epsilon. To obtain the results presented in Table I, we have carefully chosen the values for ε by using the k-distance plot as a reference. Figure 5 shows one of these plots and in particular the plot for our most realistic scenario 4slots10SNR250ms, for $k = 1, \ldots, 15$. For small values of k, the plot presents lines with a slope characterized by points with low k-distance values, and whose gradient changes once and very quickly towards the end. The points in the part of the curve with a low slope represent detections inside of clusters, and their k-distance value is the distance from their k-NNs. Note that all curves are consistently bimodal, meaning that the data presents a single density distribution, as expected and as required by DBSCAN. For readability, the plot presents only points with k-distance values under 120, but the data presents points with k-distance up to around 10000. These are the points that represent the distance between two detections that happened during two distinct and temporally distant moments. The k-distance is expressed in frames, so a k-distance of 1 represents 250ms.

To choose ε with $k = 1$, we concentrated on the elbow of the curve where the slope increases quickly, i.e. between k-distance 10 and 20 (for our experiments we chose a prudent value of 14 to preserve specificity). The points in this range represent detections that are in the sparse areas of the clusters or at their boundaries. Their k-distances are good candidates for our choice of ε. One has to be careful though, as a larger ε increases the probability to merge adjacent clusters. When this happens, the algorithm sets also the bits that correspond to moments when the sensors were not in physical proximity. The consequence is an increase in the number of false positives. Although false positives do not affect the sensitivity of the measurement, they decrease the specificity. To investigate the relationship between ε, the sensitivity and the specificity we ran DBSCAN on the data that belongs to scenario 4slots10SNR250ms with different values of ε and $minPTS = 1$. The results are presented in Figure 6.

The results show that the sensitivity of the measurement increases with larger values of ε, as false negatives are turned into true positives by the algorithm. The sensitivity slows its slope after around $\varepsilon = 20$, as the optimal solution is reached. After this point, adjacent clusters start being merged, with true negatives being turned into false positives. This is evident by the specificity curve that starts decreasing around that same value of ε. These results show that good values of ε can be chosen by a correct read of the k-distance plot. Similar results can be obtained with larger values of $minPTS$. Note that the k-



(a) Accuracy for 1% of noise (b) Accuracy for 5% of noise (c) Sensitivity and specificity for 1% of noise

Fig. 7. Recognition of noise through different values of minPTS. Epsilon was chosen for each minPTS by reading the k-distance plot.

distance plot is often the only instrument available to estimate ε, as the measurement of sensitivity and specificity requires ground truth data.

Filtering out noise. Until now, we have kept *minPTS* fixed to 1, so that also single detections could generate edges in the proximity graph. In a scenario where isolated detections can be generated by corruptions and interferences to the signal, a larger value might be necessary. This way, DBSCAN requires a larger minimum number of detections to establish a cluster, and this enables it to recognise noise. The downside is that short intervals of physical proximity risk to be filtered out as well, when the number of detections that characterise them is smaller than *minPTS*. This problem is particularly relevant for larger frame lengths, as they produce a lower detection rate and hence less detections for the same interval of time.

To investigate this hypothesis, we injected artificial noise into the data generated by the simulators. We ran three experiments where we set 1%, 5% and 10% of the bits in each bit string, respectively. For each pair of nodes, the bits were chosen uniformly at random. Each test was repeated 10 times and the results in Figure 7 show the average values of accuracy in identifying the injected noise. The plots do present error bars, but due to the very low standard deviation they are not visible.

The results show that for 1% of noise the algorithm is able to recognise 80% of the noise already with a *minPTS* of 4. For the experiment with 5%, 60% of the noise is recognised correctly with a *minPTS* of 10. We do not present results for the experiment with 10% of noise as with *minPTS* of 10 the accuracy was below 1%. These results validate our hypotheses. In fact, by specifying a larger value for *minPTS* we are able to recognise an increasing amount of noise. Moreover, by increasing the detection rate the algorithm is able to recognise more noise, i.e. a difference of 10% between the 1000ms and 250ms frame length respectively. Note that, in particular with higher rates of noise, some of the randomly selected bits might be already set or belong to already existing bursts, hence not producing actual noise. This means that a 100% accuracy is an unrealistic result to reach. Overall, the results of these experiments show that, for realistic noise rates, the algorithm is able to recognise and filter out noise without a big impact on the sensitivity and specificity of the instrument. Also, they show the function of the detection rate with respect to the ability to filter out noise.

IV. Evaluation in the Real World

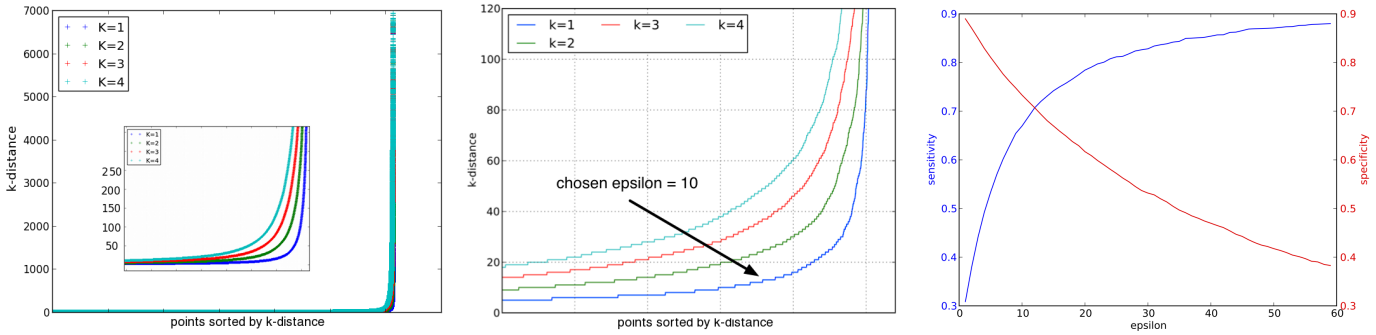
In this section we present the results obtained by applying the proposed approach to data collected during two real world social events.

A. Methodology

To explore the behavior of our approach on real data, we measured proximity information at two social events. In particular, we first conducted a controlled experiment in our laboratory, where we collected both proximity information and ground truth. Afterwards, we conducted a large-scale experiment during an IT conference attended by around 250 individuals. The devices were running the same software and protocol as the virtual devices in our simulations, and were hence using the GMAC protocol to broadcast their IDs every second to a short distance of 2-3 meters. Every device recorded the list of IDs received by the sensor at each second in the on-board storage unit, and we later collected this data for our offline analyses. The sensors were using 64 active slots and would choose randomly which slot to use to broadcast their ID. This combination represents a compromise between the 100slots and 4slots simulated setups. The devices were also broadcasting a second type of transmission. In fact, between two short-range transmissions, a long-range transmission was broadcast up to about 20 meters. This second broadcast contained the list of IDs received by each device during the previous frame, hence comprising only short-range detections. We captured these long-range transmissions through stationary devices called “sinks”. Sinks capture the radio transmissions exchanged by the devices, but do not transmit any data over the ad-hoc wireless network. We used this data to visualize at the events the evolution of the proximity graphs in real-time. Long-range transmissions were not used to detect proximity.

B. Evaluation against ground truth data

We conducted a first experiment in a controlled environment to collect ground truth data about physical proximity. For the experiment, 12 individuals participated to a cooperative social game in our laboratory. The experiment lasted about 45 minutes and was designed to replicate a conference hall scenario. The game required the individuals to form groups and interact to solve a quiz. At any time the individuals could switch groups, mimicking the way people switch conversations during a coffee break. During the experiment, the individuals were wearing our badge devices on their chest, in a similar



(a) k-distance plot (with zoomed curve) of the ICT conference dataset

(b) (zoomed) k-distance plot for the controlled experiment

(c) Sensitivity and specificity study for the controlled experiment

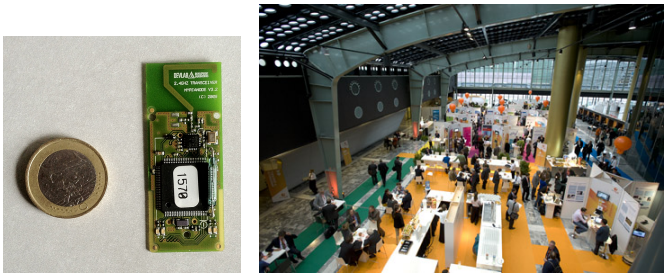
Fig. 8. Analyses of the bit strings extracted during two real-world experiments.

way as a typical conference badge is worn. In addition to our badges, we used a real-time location system (RLTS) [4] to track the position of each individual at each time. Moreover, we annotated the evolution of the formation of the groups through 5 video cameras and a human observer. From these two sources, we extracted two ground truth data sets.

Results. We performed the study of the sensitivity and the specificity of our proximity measurements against both ground truth data sets. The results were consistent with both ground truth sets but, due to space constraints, we report in Figure 8(c) only the results against the RLTS data, as it was a harder task. In fact, when our badge is worn on the chest, the body partially shields the transmissions, biasing the sensing area towards the front. On the other hand, the ground truth was extracted from the RLTS following the assumption of a uniform disc. As a result, some detections might be missed by our badges (e.g. when two individuals stand back to back), decreasing the sensitivity of the measurement. Reading the k-distance plot shown in Figure 8(b), we chose for ε a value of 10. The validity of the choice is confirmed by the study of the sensitivity and specificity for different values of ε shown in Figure 8(c). Precisely, using our approach with this parameter the sensitivity of the measurement improved from 0.309 (raw measurement) to 0.670, while the specificity decreased from 0.890 (raw measurement) to 0.733. Similar results were obtained against the annotated ground truth (sensitivity from 0.389 to 0.792 and specificity from 0.908 to 0.759). These results are strongly consistent with our simulations, as the sensitivity is more than *doubled* in the face of a decrease in specificity of only around 17%. This confirms the effectiveness of our approach also in a real-world scenario and make us confident of the quality of the measured proximity graphs.

C. Evaluation at an IT conference

We conducted a second experiment at an IT conference attended by around 250 individuals, 137 of which were wearing one of our devices. The conference presented six different tracks, and the track rooms were all scattered around the main hall of the event location. This main hall hosted the coffee breaks, the lunch banquet, and the poster session. We recorded proximity information for the whole event, but here we focus on the two hours between 12:00 and 14:00 when the lunch banquet and the poster session took place. No talks were given during this period, and the individuals were all gathered in the main hall. Figure 10 shows the device and the main hall. Due to the nature and the scale of the experiment, it was not possible to collect ground truth data.



(a) The device

(b) The experiment location

Fig. 10. Setup of the real world experiment at an IT conference.

Results. After the experiment, we collected from each device the logs containing the list of IDs each device detected at each moment, and we converted the logs for each pair of devices to our binary format. Figure 8(a) shows the k-distance plot for the two hours we focus on. Also here, the k-distance plot presents a single density distribution, with a stable slope that increases only towards the right side of the plot. This result was expected and is consistent with the data previously obtained. We used this k-distance plot to choose the parameters to extract the proximity graphs from the dataset. Figure 9 shows the effect of the different values for the parameters used to cluster the detections between two specific individuals. Each plot represents the same bit string, and each vertical black tick in it represents a set bit in the raw data. The horizontal colored lines over the ticks represent the clusters the ticks have been assigned to. Each bit under these lines will result in an edge in a proximity graph. Ticks that are not beneath a colored line have been classified as noise. The top plot shows the results of clustering the ticks with $\varepsilon = 30$ and $minPTS = 2$, the center plot shows the results of clustering the ticks with $\varepsilon = 60$ and $minPTS = 4$, and the bottom plot shows the results of clustering the ticks with $\varepsilon = 120$ and $minPTS = 2$. These plots show how, by increasing ε, clusters grow by merging with adjacent clusters or by including isolated detections nearby. Also, they show how certain isolated detections can be classified as noise as a result of larger values of $minPTS$. Looking at these plots, it is possible to conclude that the two

[4] Ubisense: http//www.ubisense.net

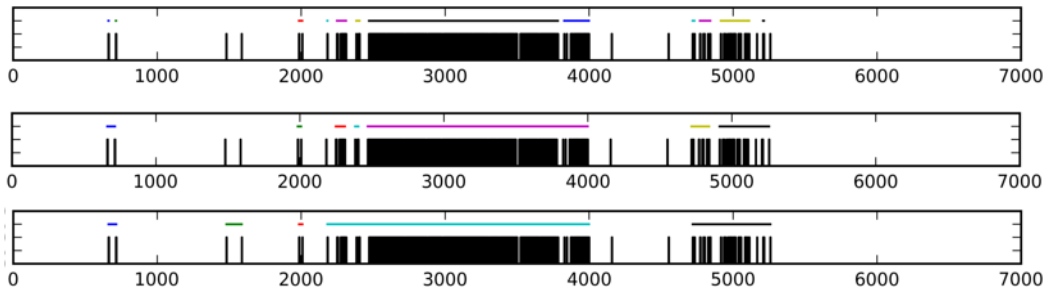


Fig. 9. The results of clustering detections using different values of epsilon and minPTS. Each plot represents the same proximity data between a pair of devices. (Top) epsilon = 30, minPTS = 2 (Center) epsilon = 60, minPTS = 4 (Bottom) epsilon = 120, minPTS = 2. Each vertical tick represents a set bit, the colored horizontal lines on top of the ticks represent the clusters those ticks have been assigned to. Ticks that are not beneath any line are classified as noise.

individuals were in physical proximity multiple times, with two larger intervals and a few shorter ones.

When designing an instrument that will be deployed to real environments, sources of misdetections should be considered with high priority. During the experiment, we also measured the radio transmissions with a special device called an RF sniffer. This device scans a tunable band of radio frequency waves and attempts to interpret the observed signals as packets at high resolution. Where a normal device will observe nothing in the event of two messages colliding, an RF sniffer will observe some combination of the two messages generated by the overlapping of the two individual radio signals. As such, this device allows us to make some observations about the number of collisions/corrupt packets that would be experienced by devices in the vicinity of this sniffer. During the experiment our RF sniffer observed 277764 packets. Of these packets, 137077 were *valid* and 140687 (or just over 50%) were *corrupt*. These corrupt transmissions could be the result of many factors (e.g., interference from nearby WiFi devices), but the most likely cause is message collisions with other devices. These collisions will result in missed detections. In addition, it is likely that some corrupt packets (incorrectly) passed the CRC check, resulting in being accepted as valid data. The effect of these messages would be false positives, as the corrupt data would cause the device to detect other devices that are not truly at distance range, for example by modifying the transmitted ID with the ID of another device due to a flipped bit. These statistics show the necessity for a method such as ours to cope with these effects.

V. Discussion and Future Work

We have shown how the application of data mining techniques to data collected through a number of wearable sensors can increase the reliability and the accuracy of physical proximity measurements. We have also shown how the setting of the parameters can influence sensitivity and specificity. In particular, trying to increase sensitivity exceedingly can cause specificity to decrease. Using an overly large value for ε can cause true negatives to be turned into false positives. Hence, there is a trade-off between the number of false negatives that can be turned into true positives and the number of true negatives that will be turned into false positives. It depends on the different applications and algorithms whether false negatives should be preferred to false positives and vice versa, putting the choice in the hands of practitioners. Regardless of this choice, one cannot assume a filtering of the data that yields to a perfect reconstruction of the measurement. In fact, algorithms and applications should be designed to cope with unreliable and inaccurate data, instead of assuming perfect measurements.

The presented approach assumes a single density distribution of detections that is valid for all pairs of devices at each time. While this assumption is reasonable for scenarios such as those described in this paper, where the density distribution of individuals in space tends to be uniform and stable, more dynamic scenarios and longer measurements require a refinement of our approach. For example, individuals located in highly dense areas will experience a higher rate of false negatives and will require a larger ε. The same individuals, at a different time, might experience a lower rate in areas characterized by lower density. For this reason, for our future work we plan to extend our technique to consider data collected only during a limited amount of time, and adapt the estimate of ε accordingly. Moreover, we plan to investigate a stream-based technique that will allow us to consider only these most recent measurements in an *online* fashion. Ultimately, this more localized approach should enable the integration of the techniques directly on the devices.

VI. Related Work

Extracting proximity information about social behavior with on-body sensors is not a new idea. The Sociometer [9] is a wearable device that can measure social signals through IR sensors, microphones, Bluetooth, accelerometers, etc. As the focus of the Sociometer is to measure face-to-face interactions, proximity is sensed within a narrow frontal cone-shaped region of up to two meters. Given that people move around when they are talking, also the Sociometer suffers from bursty measurements. For this reason, the data extracted from the four IR sensors is processed with a Hidden Markov Model that is trained to learn patterns of IR signals over an annotated dataset. The focus on a frontal measurement and the dependence on annotated data pushed us to persue a different approach. Hitachi's Microscope [5] is a device that provides similar features to those characterizing the Sociometer.

In [7] an approach based on RFIDs is presented. Similar to our approach, the device is worn on the chest of the individuals, and the devices periodically exchange their IDs through radio packets to detect physical proximity at close range. One of such detections accounts for an interval of proximity of 20 seconds. According to the authors, the devices operate with parameters that can assess physical proximity

with a probability in excess of 99% over such time interval. As the parameters and the method are not described, it is difficult to identify the reliability of this assessment. Moreover, we aim at a more sensitive measurement than the one provided over those 20 seconds of potential false positives.

A different approach is followed for the iBadge [27]. The iBadge is a wearable device that can measure location, orientation and tilt, environmental settings, and speech. Orientation and tilt are measured from acceleration and magnetic sensors. The localization process is based on the location knowledge of fixed beacons (devices with known location) attached to the ceilings and the ability of the iBadge to accurately measure the distance between itself, the fixed beacons, and other iBadges in the room. The distance is measured by recording the arrival time difference of a simultaneous transmission of an RF signal and an ultrasonic burst. Using the distance measurements from multiple beacons in the room, iBadge estimates its precise 3-D location. Although iBadges cooperate in this localization process, the measurement of physical proximity is based on beacons. We consider this approach limiting and prefer a decentralised approach that does not need or produce location information.

VII. Conclusions

In this work we have presented a method to sense physical proximity reliably. The method is based on a miniaturized device running an energy-efficient protocol for social ad-hoc wireless networks. The instrument can operate for weeks with one battery charge, making it suitable for long studies. The detections extracted from the network can be processed through a data mining technique that reconstructs missed detections and filters out noise. We evaluated both in simulation and in real-world experiments that the approach is able to increase the sensitivity of the sensor without lowering its specificity too much. The approach does not require a learning phase, but only a few parameters. We have provided a non-parametric method to choose the values for these parameters. The method provides a reliable measurement of proximity that is not bounded to the particular technology used to sense physical proximity, and can hence be used in a variety of scenarios and applications. Our approach is not specific to our devices or to a particular medium, but it is applicable to different technologies and protocols, as long as the technological layer underneath can map onto the proposed binary model.

Acknowledgements

This publication was supported by the Dutch national program COMMIT. The authors would also like to thank Marco Cattani and Dimo Stoyanov for their help with the deployment of the real-world experiments.

References

[1] D. Lazer, A. Pentland, L. Adamic, S. Aral, A. L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. Van Alstyne, "Social science: Computational social science," *Science*, 2009.

[2] A. Vinciarelli, M. Pantic, and H. Bourlard, "Social signal processing: Survey of an emerging domain," *Image Vision Comput.*, 2009.

[3] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Comm. Mag.*, 2010.

[4] A. Pentland, T. Choudhury, N. Eagle, and P. Singh, "Human dynamics: computation for organizations," *Pattern Recognition Letters*, 2005.

[5] K. Yano, K. Ara, N. Moriwaki, and H. Kuriyama, "Measurement of human behavior: Creating a society for discovering opportunities," *Hitachi Review*, 2009.

[6] T. M. T. Do and D. Gatica-Perez, "Human interaction discovery in smartphone proximity networks," *Personal and Ubiquitous Computing*, 2013.

[7] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.-F. Pinton, and W. Van den Broeck, "What's in a crowd? analysis of face-to-face behavioral networks," *Journal of theoretical biology*, 2011.

[8] M. B. Kjærgaard, M. Wirz, D. Roggen, and G. Troster, "Mobile sensing of pedestrian flocks in indoor environments using wifi signals," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, 2012.

[9] T. Choudhury and A. Pentland, "Sensing and Modeling Human Networks using the Sociometer," in *Proceedings of the International Conference on Wearable Computing*, 2003.

[10] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, "Social fmri: Investigating and shaping social mechanisms in the real world," *Pervasive Mob. Comput.*, 2011.

[11] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Personal Ubiquitous Comput.*, 2006.

[12] M. Dobson, S. Voulgaris, and M. van Steen, "Network-level synchronization in decentralized social ad-hoc networks," in *5th International Conference on Pervasive Computing and Applications (ICPCA 2010)*, 2010.

[13] ——, "Merging ultra-low duty cycle networks," in *Proceedings of the 41st International Conference on Dependable Systems & Networks (DSN 2011)*, 2011.

[14] C. Martella, A. van Halteren, M. van Steen, C. Conrado, and J. Li, "Crowd textures as proximity graphs," in *IEEE Communications Magazine*, 2014.

[15] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, 2007.

[16] M. Ester, H. P. Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996.

[17] Z. Xiong, R. Chen, Y. Zhang, and X. Zhang, "Multi-density dbscan algorithm based on density levels partitioning," *Journal of Information and Computational Science*, 2012.

[18] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "Bonnmotion: a mobility scenario generation and analysis tool," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '10, 2010.

[19] X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, "A group mobility model for ad hoc wireless networks," in *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '99, 1999.

[20] E. Weingärtner, H. Vom Lehn, and K. Wehrle, "A performance comparison of recent network simulators," in *Proceedings of the 2009 IEEE international Conference on Communications*, ser. ICC'09, 2009.

[21] N. Abramson, "The throughput of packet broadcasting channels," *IEEE TRANSACTIONS ON COMMUNICATIONS*, 1977.

[22] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, ser. MobiSys '04, 2004.

[23] N. Ababneh, "Radio irregularity problem in wireless sensor networks: new experimental results," in *Proceedings of the 32nd international conference on Sarnoff symposium*, ser. SARNOFF'09, 2009.

[24] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin, "Simulating wireless and mobile networks in omnet++ the mixim vision," in *Proceedings of the 1st international conference on Simulation tools and techniques*, ser. Simutools '08, 2008.

[25] G. Pei and T. R. Henderson, "Validation of ofdm error rate model in ns-3," *Online: http://www. nsnam. org/˜ pei/80211ofdm. pdf*, 2010.

[26] M. Z. n. Zamalloa and B. Krishnamachari, "An analysis of unreliability and asymmetry in low-power wireless links," *ACM Trans. Sen. Netw.*, 2007.

[27] A. Savvides, "Design of a wearable sensor badge for smart kindergarten," in *Proceedings of the 6th IEEE International Symposium on Wearable Computers*, ser. ISWC '02, 2002.