Sensor Data Synchronization in a IoT Environment for Infants Motricity Measurement

Simone Sguazza[1], Alessandro Puiatti[1], Sandra Bernaschina[1], Francesca Faraci[1], Gianpaolo Ramelli[3], Vincenzo D'Apuzzo[2], Emmanuelle Rossini[1], and Michela Papandrea[1(✉)]

[1] University of Applied Sciences and Arts of Southern Switzerland (SUPSI), Manno, Switzerland
{simone.sguazza,alessandro.puiatti,sandra.bernaschina,francesca.faraci, emmanuelle.rossini,michela.papandrea}@supsi.ch
[2] Centro Pediatrico del Mendrisiotto (CPM), Mendrisio, Switzerland
[3] Ente Ospedaliero Cantonale (EOC), Bellinzona, Switzerland

Abstract. Sensor data synchronization is a critical issue in the Internet of Things environments. In general, when a measurement environment includes different independent devices, it is paramount to ensure a global data consistency to a reference timestamp. Additionally, sensor nodes clocks are typically affected by environmental effects and by energy constraints which generate clock drifts. In this work, we present a specific Internet of Things architecture composed by seven Inertial Measurement Unit nodes, three Raspberry Pi 3, three video cameras and a laptop. In specific, we present an off-line data-driven synchronization solution which handles data of different nature and sampled at different frequencies. The solution solves both the data synchronization issue and the data-time alignment due to clock drift problems. The proposed methodology has been implemented and deployed within a measurement context involving infants (from 8 to 15 months old), within the scope of the AutoPlay project, whose goal is the analysis of infants ludic motricity data in order to possibly anticipate the identification of neurodevelopmental disorders.

Keywords: IoT system · Sensor data synchronization · Activity inference · NDD early detection · Healthy youth · Infants · Play

1 Introduction

The AutoPlay project [3] consists in the design and implementation of an IoT solution as an application for supporting pediatricians in the early diagnosis of neuro-developmental disorders. In particular, AutoPlay investigates the development of toys manipulation patterns in the very small children ludic behavior

N. M. Garcia et al. (Eds.): HealthyIoT 2019, LNICST 314, pp. 3–21, 2020.
https://doi.org/10.1007/978-3-030-42029-1_1

(less than 2 years old). The infants manipulation analysis results are then exploited in order to build infant ludic behavioural models, and to create a prediction model able to support pediatricians in the identification of atypical infant ludic behaviour, which could be related to social problems or neurodevelopmental disorders.

The very final objective of AutoPlay is reaching a social systematic change on how ludic behavior is observed, analysed and considered for evaluating and identifying early signs of neurodevelopmental disorders and social problems. This will allow the anticipation of the diagnosis and the necessary therapy, hence increasing the opportunity for a better quality of life.

In this work we describe the IoT solution exploited for an initial pilot study, aimed at creating a first database of infants ludic behavioural data. The data collected during this pilot consists in video recordings and inertial measurements data. In particular, in this work we present the preprocessing analysis performed over the collected data, and in particular the designed and implemented data synchronization procedure, necessary for quantifying and handling the non constant time shift between the chock of the different data sources. The main goal of the presented data synchronization procedure is to generate a reliable and usable dataset which can be subsequently applied to a machine learning analysis for manipulation activity inference.

The rest of the paper is structured as follows. Section 2 briefly summarizes the state of the art approaches for sensor data synchronization in IoT environments. Section 3 presents the materials and methodologies applied for the AutoPlay project and the related design and implementation of the IoT system architecture. Section 4 is the main part of the paper: it provides a presentation of the pre-processing task and related issues, and provides a detailed description of the implemented two/steps synchronization solution. Finally, Sect. 5 presents a final discussion on the presented work, future works and possible alternative solutions.

2 Related Works

Time synchronization is a critical issue in Wireless Sensor Networks (WSN) and Internet of Things (IoT) environments. In general, when a measurement environment includes different independent devices, it is necessary to ensure data consistency to a global reference timestamp: all data captured from different devices, measuring events at the same time, should have the same timestamp. A logical synchronization approach (i.e., the Lamport algorithm for ordering events happened in a distributed environment) does not solve our issue. What we need to identify is a global reference clock, and to synchronize all the heterogeneous sampling devices to it, with a precision in the order of ms. Additionally, we also need to compensate the clock drifts of each sensor nodes, due to environmental effects (i.e., fluctuations in temperature, pressure, humidity) and energy constraints (i.e., limited power resources).

There are different solutions developed for the WSN, where the synchronization procedure is performed by means of wireless messages exchange between

pairs of devices [4], or broadcasted to the network [2,8], where the content of the message consists in the actual global timestamp [5,7,12]. Skiadopoulosa et al. [11] presents an approach slightly different from the traditional ones for WSN which induces negligible extra overhead, in fact it does not require extra messages and re-synchronization procedures for handling the clock drift problem: in their approach, data measurements is synchronized instead of node clocks.

These approaches require the establishment of a wireless network between sensor devices as a mesh (all the sensors devices are connected through neighbor nodes) or as a star (each sensor device is connected to a central node, typically a master node).

Traditional data synchronization strategies cannot be applied in many IoT contexts, where sensor devices have intermittent or no connectivity to the network and have limited power resources. There exist different solutions which implement real-time data synchronization [10], or which requires data streaming, and solutions which simply tolerate un-synchronized data. Other synchronization approaches are only based on sensor data and are performed off-line. Luckac et al. [9] present an approach relying on regularly occurring events and a model of the event propagation to allow correction of the time information. This solution is limited in that it requires a known regularly occurring seismic event. Harashima et al. present a synchronization solution assisted by environmental signals [6]. The environmental signals being measured are used as an additive noise that synchronizes the sensors. Because it is expected that the sensors in a limited range will see the same environmental signals, these signals can be used to aid the synchronization. Bettet et al. presents a similar synchronization procedure [1], where the communication between sensors is not required, and it can handle heterogeneous devices.

The technique proposed in this work allows for time synchronization between heterogeneous sensor devices, without the necessity of communication between them. The synchronization is performed off line, allowing the devices to reserve their power resources for the sampling task. The proposed approach is similar to already existing approaches because based on the presence of a reference synchronization event measured by all the devices at the same time. However, differently from the other approaches, it implements a supervised procedure for the identification of the time shift: it maximizes the proximity between the sensors aggregated data signals and a ground truth signal, and measures the clock shifts from the result of the optimization procedure. The usage of the aggregated signal increases the accuracy of the synchronization, which is in the order of milliseconds. The proposed approach works well for environments with multiple sensor devices. If the number of devices is too large, a dynamic device sampling procedure is required for ensuring the procedure scalability.

3 Materials and Data Collection Architecture

The focus of AutoPlay is to understand the way children use toys, in order to detect infants atypical behavioural patterns. To achieve this goal we identified

a set of toys belonging to the main sensory-motor classes of play: mouthing, simple manipulation, functional, relational, and functional-relational. More in particular, with the support of a professional toy designer [1] we developed the AutoPlay toys-kit, composed by: a ball, a doll and a spoon, 3 cubes, and a car (Fig. 1).

Fig. 1. AutoPlay toys kit

Each toy embeds one 9 DoF IMU (9 Degrees of Freedom Inertial Measurement Unit) sensor node that collects:

- 3D-acceleration data,
- 3D-gyroscope data,
- 3D-magnitude data.

The employed device is the Shimmer3 IMU Unit. The car includes also two rotary encoders attached to the sensor node itself, which are used to measure the movements and directions of the car's wheels (independently of the movement of the car main body).

The IMU sampling frequency is $f_{data} = 100.21$ Hz for all the toys except the car, whose sampling frequency is $f_{data} = 504.12$ Hz. The sampling frequency of the car has been set in order to be able to identify complete wheels rotations (the used rotary encoder is a mechanical encoder with 24 pulses per revolution).

While a child is playing with the toys data are collected locally by the embedded sensors and stored within the sensor node micro-SD card. At the end of each playing session, data are downloaded on a central repository and prepared for the following off-line analysis.

The data collected by sensors are used for the identification of the toy movements. This information is directly correlated to the manipulation activities performed by the infant, thus allowing us to analyze the infant ludic behaviour.

[1] Pepe Hiller http://www.pepehiller.com/.

We collect also video recordings of each infant's play session at 25 fps (frames per seconds). These recordings are collected in order to have ground truth (GT) data for the manipulation activities performed by infants. All videos are visualized and used for the creation of corresponding video logs (Fig. 2): each activity performed by the infant during a video recording (playing session) is logged as a row in a log file, which specifies the name of the activity (*Activity*) (from a list of micro-activities initially identified with the support of a clinician), the name of the involved toy (*Toy*), the relative time of the activity within the video in frames (*Frame start* and *Frame and*), the camera from which the activity is visualized and logged, and a categorical value specifying if the activity is seemingly unintentional (*N*) or performed as a consequence of a stimulus from an adult person, the educator (*E*). The timestamp of the video recording is stored as meta-data, and used to keep track of the global timestamp of each activity. The log data is anonymised, in the sense that there is no connection between the collected data and the corresponding playing infant.

The log files generated from the collected videos are used as digital target data, for subsequently training a machine learning model for inferring *infant ludic manipulation activities*.

N o E ?	Activity	Toy	Frame start	Frame end	Cam1	Cam2	Cam3
N	Hit	EV	1012	1051		x	
N	Hit	EV	1054	1075		x	
	Take	EB	1223	1244		x	
	Hold	EB	1245	2346		x	
N	Hit	EA	1567	1575		x	
	Hit	PA	1583	1587		x	
	Roll	PA	1588	1609		x	
N	Hit	CU	1615	1627		x	
N	Hit	EA	1629	1634		x	
	Turn	EB	2153	2215		x	
	Support	EB	2347	2361		x	
N	Hit	EB	2444	2457		x	
N	Hit	EA	2460	2467		x	

Fig. 2. Ground truth log file example

3.1 Architecture

In order to collect manipulation data, we created an ad-hoc Internet of Things Measuring Environment. The networked devices involved in the measurement process are listed below.

1. Three cameras which record the infant activities from different angles. The cameras are located at a distance less than 2 m form the playing area. Their position, with respect to the center of the playing area, is equally distributed in order to be able to visualize all the movements performed by the infant. Each camera is an ad-hoc module applied to a Raspberry Pi 3 Model B.

2. A laptop which is wired connected to the cameras through a switch. The laptop is an interface for the person which manages the playing measurement sessions (the educator). It is used as Network Time Protocol (NTP) server to synchronize the clock of each raspberry device, to remotely access one of the raspberry devices performing a server role (for starting and stopping the video recording for all the cameras), and to collect and store the recorded videos.
3. An IMU sensor node per each toy within the AutoPlay toys-kit.

This ad-hoc environment has been deployed within two local kindergartens in Tessin (Switzerland). The data collection pilot lasted for 15 months and involved 24 infants in the age of 8 to 15 months. The study was conducted in accordance with the Declaration of Helsinki and written consent was provided by all participants families. The study was reviewed and approved by the competent ethics committee (Swiss Ethical Committee of Kanton Tessin, ref. PB 2016-00056) before the start of participants inclusions.

In order to protect infant privacy, the laptop and the raspberry Pi involved in the study, do not have access to the Internet. All sensitive data (video recording, infants demographics and IMU data) are not transmitted via wireless. This particular settings have been decided in accordance with the Swiss Ethical Committee. All the devices wire-connected (within the measurement LAN) are synchronized with each other. A schema of the wired measuring environment is provided in Fig. 3.

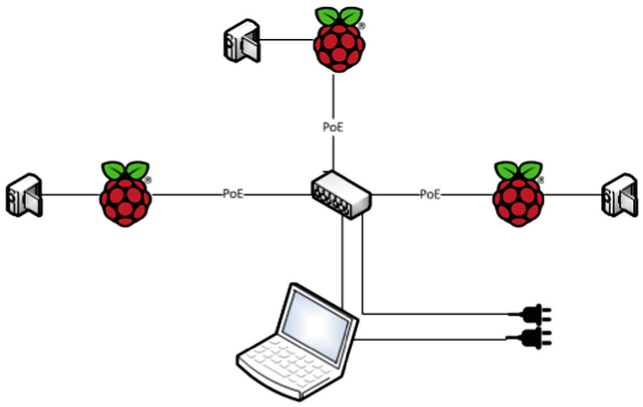


Fig. 3. Measuring environment LAN

The sensor nodes are independent of the other IoT devices. During the described pilot study we tested two possible configurations:

1. sensor nodes synchronized with the measurement LAN by means of BLE (Bluetooth Low Energy) communication with the laptop (data-time synchronization is performed before the video-data collection);

2. sensor nodes completely independent of the sensor LAN during the measurement sessions, the synchronization between the recorded videos and the data collected is performed later on, during the preprocessing analysis.

Details about the two solutions are described in Sect. 4.

4 Methodology

Within the presented architecture, the sensor nodes and the video cameras are independent from each other, generating then a synchronization issue in the collected data. In this paragraph we are going to describe how we handled the synchronization issue between the videos recorded and the sensors collected data. As described in Sect. 3, we implemented two configuration for the sensor data collection. In a first configuration we established a BLE connection between the laptop and each sensor device: the connection allowed the timestamp synchronization (the employed sensor does not have a real-time clock), the sampling frequency configuration, and the sampling session management (start-stop sampling commands). With this measurement configuration, the collected data are already synchronized with the videos recorded by the cameras.

In the second configuration, all the sensor devices are configured and synchronized with each other before the measurement session, but they are not synchronized with the rest of the IoT devices. In this case a synchronization between the video recordings and the sensor collected data is required, and performed offline, during the data preprocessing phase.

The first approach would be the preferable one, because it requires less preprocessing actions. However we opted for the second approach, because even if it requires a synchronization pre-processing procedure, it ensure the absence of data loss, which is indeed very frequent with the first approach. In fact, the BLE communication had reliability issues within our pilot measurement environment. The laptop was not always able to reach all the sensor nodes and the sampling sessions was not always starting when necessary, causing a considerable loss of data.

Within the context of our pilot, data loss is an important negative aspect compared to the increased pre-processing computational effort. The limited availability of infants and families giving their consent for the data collection, and the difficult selection procedure of the appropriate moment during the day for the infant playing sessions, gives a greater value to the actual presence of collected data, even if this data require a pre-processing procedure.

In order to implement the second approach, within the measuring environment, an adult person responsible for the playing sessions (typically a kindergarten educator) had to manually start and stop the sampling procedure on each device. The toys have been designed in such a way that the start/stop sampling button was easily reachable, without completely disassembling the toy (Fig. 4). At the same time, the toys have been designed in such a way that the sampling button was not reachable by an infant.

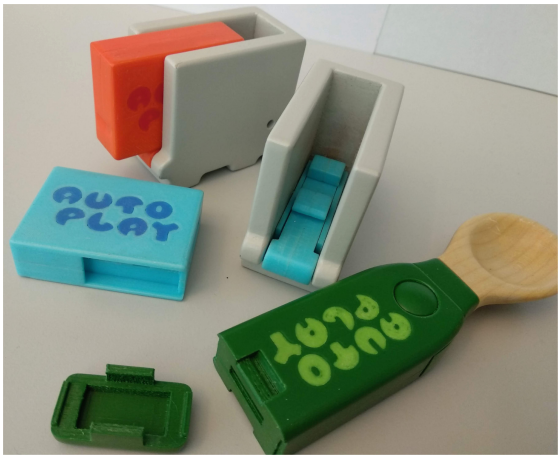


Fig. 4. AutoPlay toys plug details

Additionally, for both configurations, a second phase data-time alignment is required, in order to handle sensor related clock drift issues.

4.1 Sensor Data Synchronization

As explained above, in order to analyze the collected data, an important step is the timestamp synchronization between the recorded videos and the sensors data. The main goal of this task is to identify the *time shift* between them in order to align the sensors data with the ground truth videos.

Additionally, there exist a not negligible clock drift in the collected data, which requires a two-steps synchronization procedure:

1. identification of the overall synchronization *time shift*;
2. perform a per-measurement post-synchronization data-time alignment.

For the synchronization task, we start from the assumption that all the devices timestamp are synchronized to the actual date and time of the day: the synchronization time shift we are searching for, in the worst case, is in the order of minutes. To perform the timestamp synchronization we implement the following steps (Fig. 5).

1. We record a first video i during which we perform a *synchronization event*.
2. We select the event starting frame F_i^{event} from the video and encode it in timestamp format in milliseconds T_i^{event}.
3. We select from the sensors raw data a data time window of 4 min W_i^{sync}, where T_i^{event} refers to the middle point of the window. Within this time window we search for the synchronization event.
4. We identify the event starting time in milliseconds T_i^{sync} within the W_i^{sync}, and we calculate the synchronization time shift $S^i = |T_i^{sync} - T_i^{event}|$, which subsequently is applied to all the data collected during session i.

This procedure has a sort of analogy to the audio-video synchronization procedure in a film production. Also in that case, audio and video signals are recorder independently. In order to synchronize the two, the producer creates a characteristic event using a clapperboard which can be recognized in both audio and video signals.

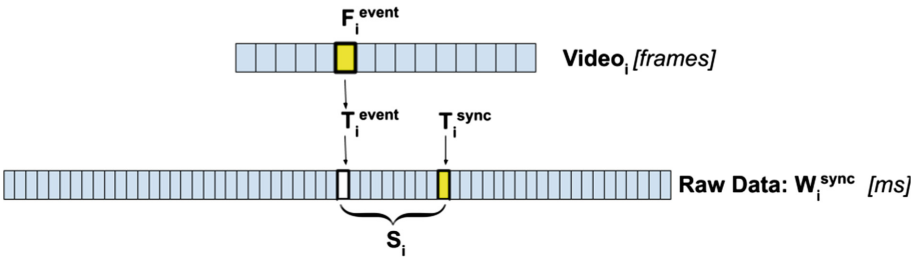


Fig. 5. First step synchronization

4.2 Identification of the Synchronization Event

In order to generate the synchronization event, we performed a characteristic movement with the toy (the reference toy for this analysis is a *cube*). In particular we overturn the toy to stand on a different face, multiple times (minimum 2 times). We exploited also different movements (i.e., trowing the toy, lifting and spinning the toy), resulting in a lower accuracy synchronization. The reference signal we exploit for synchronization is the acceleration, and in particular we implemented the T_i^{sync}-search procedure on the acceleration axis over which the toy is overturned.

Considering the data recorded during the synchronization overturning movement: the data are first filtered for removing the signal noise; subsequently the derivative of the signal is calculated and exploited for the event identification.

In order to use the acceleration data collected while performing the synchronization event to identify the T_i^{sync} we process the signal through a low pass filter for signal noise removal: we decided to use a filter which works on the spatial domain (a Gaussian filter), instead of a frequency domain filter which might add some signal artifacts. In Fig. 6 the row signal is represented in blue, the filtered signal without noise is in orange. Afterwards we calculate a discrete derivative on the filtered signal, in order to detect the time of the synchronization event (green in Fig. 6).

The derivative values have a normal distribution with zero mean. We then identify the synchronization time, the first time the derivative signal values falls outside the interval $[\mu - 1.45 * \sigma, \mu + 1.45 * \sigma]$ (where $\mu = 0$ and σ are mean and standard deviation of the derivative signal). The 1.45 multiplicative factor has been chosen empirically. The result is depicted in Fig. 6, where the red line is a step function, different from zero when the toy is performing an overturn movement, and the vertical solid yellow line corresponds to the identified T_i^{sync}.

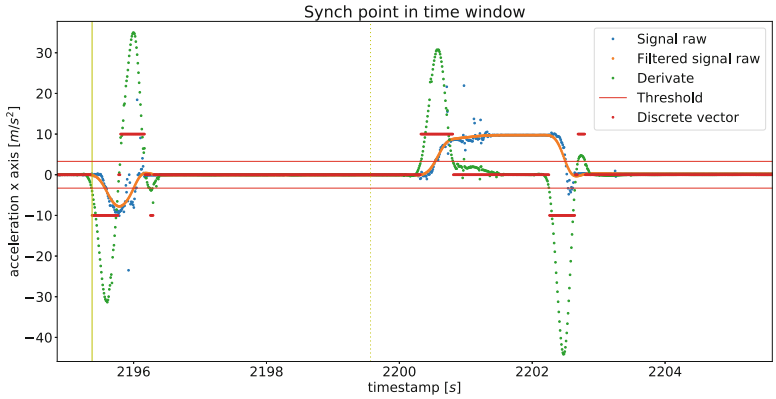
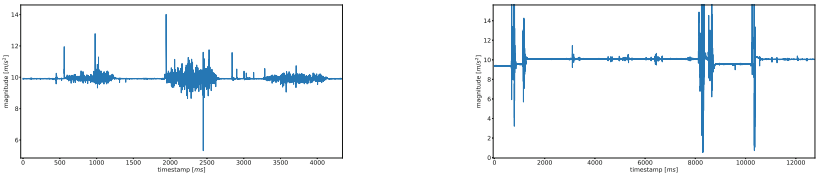


Fig. 6. Acceleration data: example of synchronization procedure with four overturn movements (Color figure online)



(a) Constant gravity (b) Non constant gravity

Fig. 7. Measured acceleration magnitude

At this point we can calculate the synchronization time shift as reported in Eq. 1.

$$S^i = \left| T_i^{sync} - T_i^{event} \right| \tag{1}$$

4.3 Gravity Acceleration Removal

As part of the data preprocessing, we had to discriminate signal noise (*non-activity*) from actual toy movement (*activity*), this allows the second step of the synchronization: post-synchronization data-time alignment.

In order to be able to differentiate between activity and non-activity we need to first remove the constant gravitational component from the acceleration data. For this task we work on the *magnitude* of the acceleration vector.

The acceleration magnitude signal is the sum of two components: a dynamic component, proportional to the sum of the forces applied to the sensor node, and a constant component, which correspond to the gravity. In our case, the collected signal, besides being very noisy, it does not always have a longitudinally constant gravity component, however it can change over time. This clearly adds complexity to the synchronization task. For the case of a sampled signal with constant gravity (see Fig. 7a), we remove the gravity component with an high

pass filter with cut off frequency $f_c = 0.001$ Hz. For the case of a sampled signal with non constant gravity (see Fig. 7b), instead, we use a *Notch filter*, with *parameter* $\lambda = 0.7$: the selected value minimizes the effects of the filter due to the applied transfer function. Focusing on the goal of this task, we need to select a filter which removes the gravitational component, producing an output signal which allows us to differentiate activity from non-activity. Applying the Notch filter with the specified parameter value, produces some changes to the original signal but has a faster response in the gravity removal task, compared to other filters and parameter values.

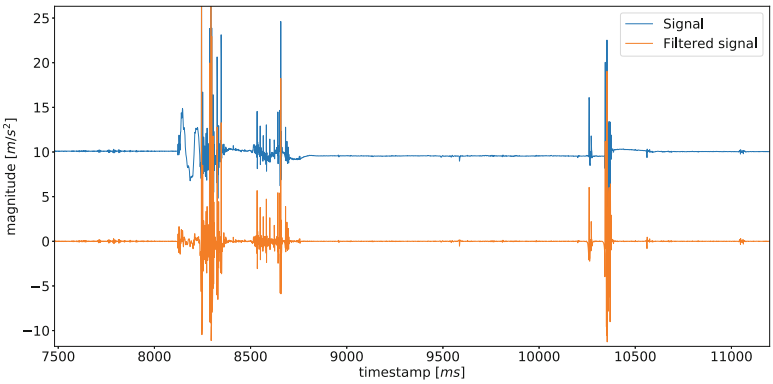


Fig. 8. Example of application of Notch filter, $\lambda = 0.7$ (Color figure online)

Figure 8 represents an example of application of the Notch filter on a non-constant gravity acceleration signal: in blue the magnitude of the acceleration raw data, and in orange the result of the applied filter.

4.4 Activity Discretization: Differentiation Between Activity/Non-activity

In order to finally differentiate between activity and non-activity we perform a statistical analysis of the acceleration signal values. From the acceleration dataset $<a_x, a_y, a_z>$ we compute the features vector $<p_a^n, m_a>$ where:

- p_a^n corresponds to the *power* of the acceleration magnitude $\bar{m}_a$, filtered for gravity removal, calculated over a window of n samples;
- m_a corresponds to the per-sample magnitude of the acceleration, calculated after the application of the gravity removal filter, independently on each axis of the acceleration.

We study the distribution of the computed feature values for two acceleration signals, both corresponding to a time window of 30 s, one referred to an *activity* session (signal generated by the concatenation of different real activity signals),

and the other related to a *non-activity* session, both sampled at 100, 21 Hz. For the power feature, we evaluate the impact of the window size (number of samples n) on the values distribution.

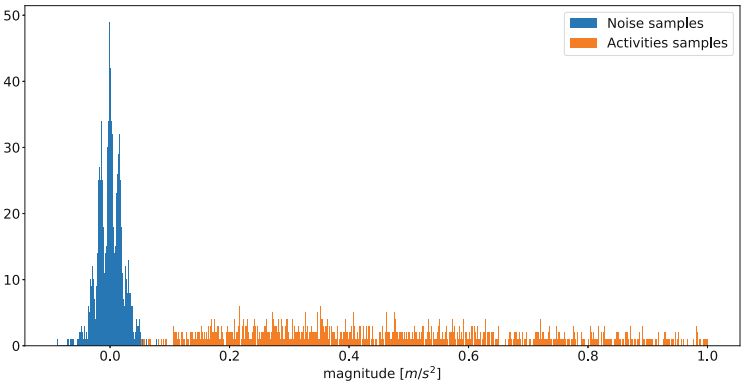


Fig. 9. Magnitude values distribution

Figure 9 represents the distribution of magnitude values m_a for both acceleration signals. While Fig. 10 represents an example of distribution of the power values p_a^8, for an averaging window of 8 samples. For both figures, the blue bars are referred to non-activity signal, while the orange bars are referred to activity.

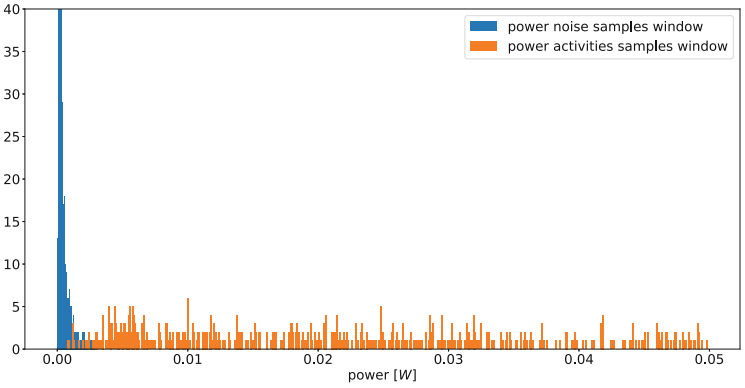


Fig. 10. Power values distribution, dimension of the averaging window $n = 8$

In both cases we can identify a *threshold value* for the data discretization, and for the power feature we can also identify a *window size* value. In order to identify these parameters, we implement an *error minimization* approach, calculating the error in time, over a signal which is associated with its ground truth (start and stop time of each activity). The error is measured in time

(milliseconds) as the absolute value of the difference between the ground truth data and the computed signal discretization (activity/non-activity), varying the discretization parameters (threshold and window size).

For the case of signal power based discretization, we applied threshold values in the range of $th_{power} = [0.001; 0.036]$ selected empirically. We applied window size values in the range $n = [2; 18]$. Window size is determined by some domain constraints: the minimum size of an intentional movement of an infant corresponds to 2 video frames (recorded at 25 fps), which means 8 samples of acceleration data (sampled at 100, 21 Hz).

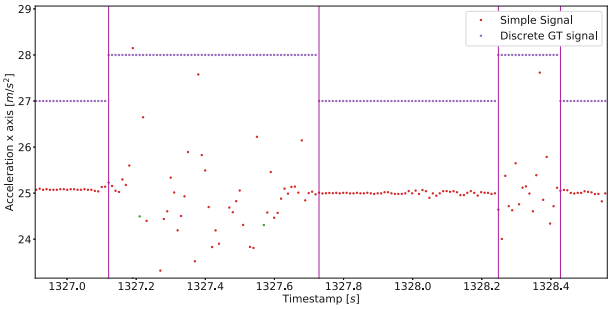


Fig. 11. Single signal with associated GT

Figure 12 shows the sum of absolute error calculated over a single signal (Fig. 11) involving four changes of state (activity/non-activity). We can see from the graph that a window of size $n = 7$ (approximately the length of two video frames) reached a minimum error with threshold values in the range $th_{power} = [0.008; 0.036]$.

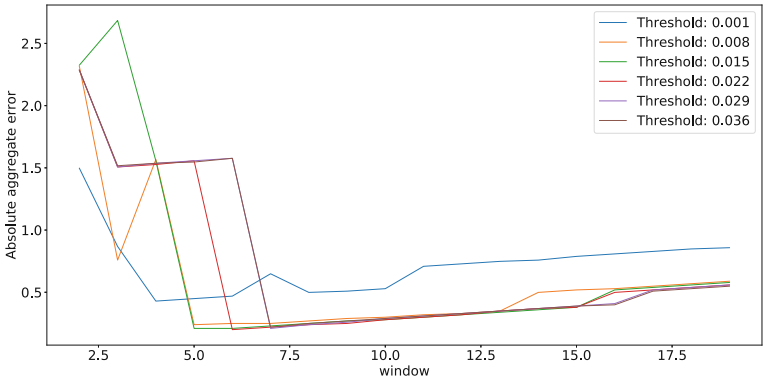


Fig. 12. Sum of absolute error calculated over a single signal, discretization on signal power

In order to implement a more accurate solution, we perform the signal discretization over a multiple sources signal. The assumption for this solution is

that all involved sources (sensor nodes) are synchronized with each other. In this case we perform an *Aggregate Error Minimization*: the error minimization task is applied to all the sensors data, computing an *aggregate sum of absolute error*.

This results in a convergence curve: the aggregated error converges to a value. The search for the convergence value stops when the procedure calculates 5 consecutive equal error values. Figure 13 represents the convergence curve of the aggregated error calculated over a real experiment. In this case, for all window sizes $n = [8; 12]$, the convergence value for an optimal aggregate error corresponds to a threshold of $th_{power} = 0.015$.

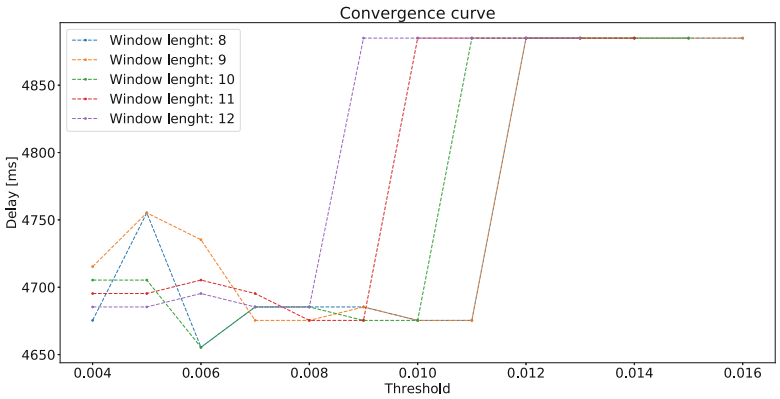


Fig. 13. Convergence curve

We performed a qualitative evaluation of the discretization procedure on the sampled data. Figure 14 represents an example of optimal delays found at different threshold values, with a fixed window size of $n = 10$. The delay represented in the figure is the time difference between the GT timestamp and the raw data timestamp, before the first step synchronization.

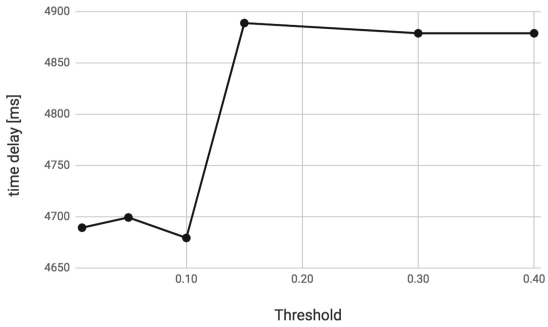


Fig. 14. Delay vs threshold on power discretization, window size $n = 10$

This discretization procedure can be seen as a binary classification task, where the positive class is the *activity* and the negative class in the *non-activity*. Lowering the threshold value we increase the True Positives (activity correctly identified as activity), however very low threshold values increases the False Positives (real non-activity, as signal noise, identified as activity). Increasing the threshold value we increase the False Negatives (real activity identified as non-activity).

Figure 15 represents an example of the discretization procedure on the sampled data. All the lines are step functions equal to 0 in correspondence of *non-activity*, and 1 for the *activity* (the signals have been translated vertically for visualization purposes). The blu plot represents the aggregated ground truth (aggregated over multiple sensors) and generated from the video recording: the signal is equal to 1 if, from the recorded video, the infant is performing a manipulation activity with one of the available toys. The orange plot represents the same signal plotted above after the application of the overall synchronization time shift. The green plot is referred to the acceleration data: it is the result of the discretization procedure based on power feature, with parameter values $th_{power} = 0.030, n = 10$. Analogue results are reached with all the rest of the data in the sample dataset.

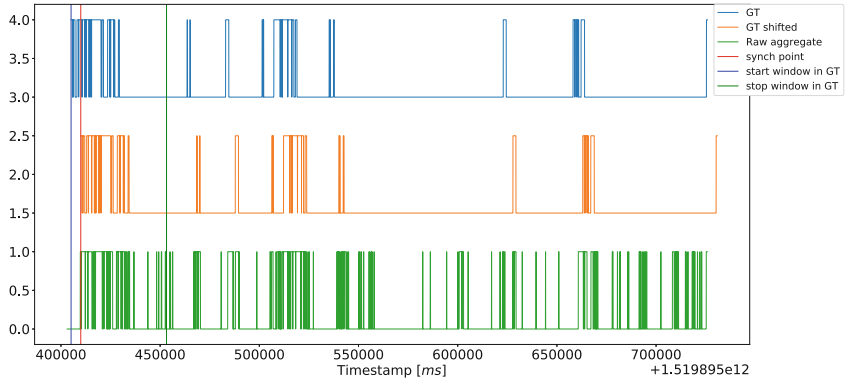


Fig. 15. Discretization example, $th = 0.030, n = 10$

We can see in Fig. 16 the application of both the discratization procedure based on power and on magnitude. The procedure has been applied to a single sensor data for visualization purposes. In the figure, the blue plot represents the sampled acceleration magnitude, while the green data corresponds to the acceleration magnitude after removing the gravitational component. All the remaining signals have been translated vertically for visualization purposes (for all of them, the real minimum value is 0). In brown the result of the discretization based on power (power in orange). In pink the result of the discretization based on magnitude (magnitude in grey). In red the infant activities identified in the ground

truth video recordings. Both methods perform well, however in the case of magnitude based approach the results present an higher value of False Positives (real non-activity identified as activity). We decide then to implement the power based method.

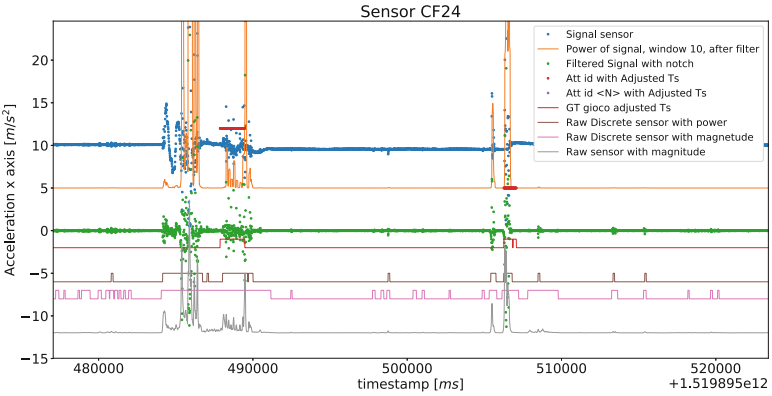


Fig. 16. Discretization procedure application on a single sensor signal raw acceleration data (Color figure online)

4.5 Post-synchronization Data Time Alignment

We described above the procedure for the identification of the overall synchronization time shift between sensors data and video recording. However the synchronization requires a second step in order to deal with the clock drift issue: each measurement session requires a specific data-time alignment in order to adjust the synchronization error introduced by the sensors clock drift. This step is also required in the BLE measurement configuration, introduced in paragraph Sect. 3.

When we perform the *post-synchronization data alignment* we have to consider two important issues. One consists in the fact that we have a video recorded ground truth sampled at a frequency of $f_{video} = 25$ Hz, while the IMU data is sampled at $f_{data} = 100.21$ Hz ($f_{data} = 504.12$ Hz for the car). When we perform the alignment of the ground truth with the raw data, we have to consider the fact that the two frequencies are not multiples of each other. A second issue is related to the information present in the GT data, and in the collected raw data: the reference GT associated to a video consists in a list of all the activities performed by the child, with related information about first and last frame of the activity; the raw data instead includes information about all the movements of the toy, including the movements not directly related to the activities of the infant (e.g., interaction with the educator, with other toys, with the surrounding environment). The usage of the aggregated raw data (data collected by more than one sensor) reduces the effects of this last issue. These issues however force

a lower bound for the data synchronization error: the acceptable error is in the range $[-40, +40]$ ms, where 40 ms corresponds to the interval of time between two video frames, and 10 ms corresponds to the interval of time between two inertial sample data.

In order to apply the post-synchronization alignment, we perform a similarity analysis between the *synchronized aggregated raw data discretization signal* and the related *aggregated GT discretization signal*. This analysis has the goal to maximize the similarity between the two signals, searching for the maximum similarity when performing a convolution of a portion of the *GT signal* (typically one half of the GT time series) over the one related to raw data. This procedure allows the identification of a *per-measurement time alignment* which is then applied to the data, as a final step of the synchronization and data pre-processing procedure.

4.6 Final Results

Summarizing the analysis described above, in order to align the raw sensor data to the GT signal we apply a time shift correction to the raw sensor data, as reported in Eq. 2: where j refers to a measurement session recording, while i refers to a prior synchronization event video. Equation 3 represents the delay component $D_{i,j}$, which corresponds to the sum of two time intervals. dt_i^{sync} is the delay measured during the first synchronization phase, while dt_j^{align} is the time alignment delay for the specific measurement j.

$$T_{i,j}^{data} + D_{i,j} = T_j^{video} \tag{2}$$

$$D_{i,j} = dt_i^{sync} + dt_j^{align} \tag{3}$$

As a result of the analysis described in this work, we have a data-set of sensor data with associated GT, ready for the application of machine learning algorithms for infant ludic behavioural analysis.

5 Conclusions and Future Works

5.1 Conclusions

We presented in this work a sensor data-driven synchronization methodology and its application within a multi-sensor environment, storing different types of data (inertial data and video recordings) at different sampling frequencies. In order to apply the presented methodology, network communication between the devices is not required. The presented methodology consists in a two steps procedure: the first one performs the synchronization between the sensor nodes data and the recorded videos; while the second step synchronization allows the data-time alignment when a sensor data clock drift is present. The methodology has been implemented within the context of the AutoPlay project, involving infants and toys for the data acquisition, in a constrained environment.

5.2 Future Works

The proposed procedure works well in the test pilot environment. However, we are currently working on future developments. In the current state of the work, we do not include the ball in the synchronization procedure, for which we are developing a different procedure for generating a reliable synchronization event. Moreover, we are implementing a machine learning algorithm for the signal discretization: building a binary classifier in order to recognize if a given signal refers to a toy manipulation activity or not.

Acknowledgment. This work was supported by Gebert Rŭf Stiftung (GRS-054/16). We thank SUPSInido and CullaBabyStar for their support and involvement during the measurement pilot study. We would also like to show our gratitude to all the families which contributed to the project, allowing their kids to participate to the pilot study. Additionally we thanks all SUPSI students and collaborators which have been involved during the GT logs generation, and during the pilot study for the sensor devices and the measuring environments management.

References

1. Bennett, T.R., Gans, N., Jafari, R.: Data-driven synchronization for internet-of-things systems. ACM Trans. Embed. Comput. Syst. **16**(3), 69:1–69:24 (2017). https://doi.org/10.1145/2983627
2. Elson, J., Girod, L., Estrin, D.: Fine-grained network time synchronization using reference broadcasts. ACM SIGOPS Oper. Syst. Rev. **36**(SI), 147–163 (2002)
3. Faraci, F.D., et al.: Autoplay: a smart toys-kit for an objective analysis of children ludic behavior and development. In: 2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA), pp. 1–6. IEEE (2018)
4. Ganeriwal, S., Kumar, R., Srivastava, M.B.: Timing-sync protocol for sensor networks. In: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, pp. 138–149. ACM (2003)
5. Guidoni, D.L., Boukerche, A., Oliveira, H.A., Mini, R.A., Loureiro, A.A.: A small world model to improve synchronization algorithms for wireless sensor networks. In: The IEEE symposium on Computers and Communications, pp. 229–234. IEEE (2010)
6. Harashima, M., Yasuda, H., Hasegawa, M.: Synchronization of wireless sensor networks using natural environmental signals based on noise-induced phase synchronization phenomenon. In: 2012 IEEE 75th Vehicular Technology Conference (VTC Spring), pp. 1–5. IEEE (2012)
7. Huang, Y.H., Wu, S.H.: Time synchronization protocol for small-scale wireless sensor networks. In: 2010 IEEE Wireless Communication and Networking Conference, pp. 1–5. IEEE (2010)
8. Jain, S., Sharma, Y.: Optimal performance reference broadcast synchronization (OPRBS) for time synchronization in wireless sensor networks. In: 2011 International Conference on Computer, Communication and Electrical Technology (ICCCET), pp. 171–175. IEEE (2011)
9. Lukac, M., Davis, P., Clayton, R., Estrin, D.: Recovering temporal integrity with data driven time synchronization. In: Proceedings of the 2009 International Conference on Information Processing in Sensor Networks, pp. 61–72. IEEE Computer Society (2009)

10. Qiu, T., Liu, X., Han, M., Li, M., Zhang, Y.: SRTS: a self-recoverable time synchronization for sensor networks of healthcare IoT. Comput. Netw. **129**, 481–492 (2017)
11. Skiadopoulos, K., et al.: Synchronization of data measurements in wireless sensor networks for IoT applications. Ad Hoc Netw. **89**, 47–57 (2019)
12. Yildirim, K.S., Kantarci, A.: Time synchronization based on slow-flooding in wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. **25**(1), 244–253 (2013)