Synchronization between Sensors and Cameras in Movement Data Labeling Frameworks

Jacob W. Kamminga
University of Twente
Enschede, The Netherlands
j.w.kamminga@utwente.nl

Michael Jones
Brigham Young University
Provo, USA
jones@cs.byu.edu

Kevin Seppi
Brigham Young University
Provo, USA
kseppi@cs.byu.edu

Nirvana Meratnia
University of Twente
Enschede, The Netherlands
n.meratnia@utwente.nl

Paul J.M. Havinga
University of Twente
Enschede, The Netherlands
p.j.m.havinga@utwente.nl

ABSTRACT

Obtaining labeled data for activity recognition tasks is a tremendously time consuming, tedious, and labor-intensive task. Often, ground-truth video of the activity is recorded along with sensor-data recorded during the activity. The data must be synchronized with the recorded video to be useful. In this paper, we present and compare two labeling frameworks that each has a different approach to synchronization. Approach A uses time-stamped visual indicators positioned on the data loggers. The approach results in accurate synchronization between video and data but adds more overhead and is not practical when using multiple sensors, subjects, and cameras simultaneously. Also, synchronization needs to be redone for each recording session. Approach B uses Real-Time Clocks (RTCs) on the devices for synchronization, which is less accurate but has several advantages: multiple subjects can be recorded on various cameras, it becomes easier to collect more data, and synchronization only needs to be done once across multiple recording sessions. Therefore, it is easier to collect more data which increases the probability of capturing an unusual activity. The best way forward is likely a combination of both approaches.

CCS CONCEPTS

• **Software and its engineering** → *Software design tradeoffs.*

KEYWORDS

Labeling, Annotating, Synchronization, Video, Camera, IMU

ACM Reference Format:
Jacob W. Kamminga, Michael Jones, Kevin Seppi, Nirvana Meratnia, and Paul J.M. Havinga. 2019. Synchronization between Sensors and Cameras in Movement Data Labeling Frameworks. In *The 2nd Workshop on Data Acquisition To Analysis (DATA'19), November 10, 2019, New York, NY, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3359427.3361920

DATA'19, November 10, 2019, New York, NY, USA

ACM ISBN 978-1-4503-6993-0/19/11…$15.00
https://doi.org/10.1145/3359427.3361920

1 INTRODUCTION

Obtaining labeled data is a tremendously time consuming, tedious, and labor-intensive task. Nevertheless, most recent published papers in the field of machine learning and Activity Recognition (AR) heavily rely on the availability of labeled datasets. Ground truth is often collected using camera(s) that record subjects while wearing one or multiple data loggers that usually contain Inertial Measurement Units (IMUs). The data is often displayed along with the recorded video so that a person can inspect segments of the data with corresponding video frames. This simultaneous display may make it easier to identify the particular ground-truth activity that belongs to that segment of movement data. For this to happen though, the data have to be synchronized with the video prior to annotation. Often the annotation is part of the methodology section in papers on AR that use labeled data [1, 2, 10, 11]. However, these papers do not discuss how the data was synchronized with the videos in detail. In this paper, we present and compare two frameworks that each has a different approach to the task of synchronizing video with sensor data and discuss their advantages and disadvantages.

2 LABELING FRAMEWORKS

2.1 Approach A: Synchronization using Visual key

To collect data using this approach, the user attaches the data logger somewhere on the subject. The user then records video of the subject in motion. Note that only one data logger and one video stream are captured. This approach has been used in data collection for Alpine skiing [4], hiking [3], and rock climbing [3].

Synchronization in this process involves capturing, on video, a red flash emitted by the data logger ten seconds after the data logger is turned on. In this approach, the user turns on the data logger, aims the video camera at the data logger, presses "record" on the camera and waits to record the red flash. The data logger logs the time of the first red flash (in milliseconds elapsed since the processor was turned on). At the beginning of labeling in Approach A, the user locates and marks the first video frame that contains the first red flash. The labeling tool can then synchronize the video and data for as long as both continue recording. If either is stopped or turned off, synchronization must be repeated.

The accuracy of this approach is limited by the video camera frame capture rate. For example, if the video is captured at 30 frames per second then synchronization of the video and data is off by at most one-thirtieth a second or 33 ms. Cameras with higher frame rates can be used to synchronize the data with greater accuracy (up to the temporal resolution of the data logger sampling rate).

2.2 Approach B: Synchronization using Real-Time Clocks

In order to label movement data of sheep [6], goats [7], and horses [8, 9], a labeling framework was developed using a Matlab GUI [12]. The code of the framework is publicly available [5]. A screen capture of the GUI is shown in Figure 1. The application displays a video along with the magnitude of the accelerometer vector (1) to visualize the sensor data. The magnitude of the accelerometer vector is defined as:

$$M(t) = \sqrt{s_x(t)^2 + s_y(t)^2 + s_z(t)^2}\,, \tag{1}$$

where, s_x, s_y, and s_z are the three respective axes of the sensor. A vertical led line in the center of the graph denotes the current time in the video. The data can be labeled by clicking at the point representing a change in behavior on the graph. The framework

Figure 1: Screenshot of the labeling application GUI

assumes that each camera and data logger contains an internal RTC that can be set prior to the data collection campaign, resulting in a coarse synchronization. The RTCs are used to timestamp each video and sensor-data file. The GUI is used subsequently to synchronize videos with sensor data more accurately. Figure 2 shows a schematic diagram of the offset $\sigma^{(l)}_{m,n}$ between a given camera m and sensor n on day l. The offset σ is determined by using a distinctive event in the video and adjusting σ until $M(t+\sigma)$ aligns with the event shown in the video. In theory, the synchronization only needs to be done once. In practice, σ will probably need minor adjustments over time due to clock drift in the RTCs of the camera and sensors. Therefore the camera-sensor offsets are stored per day and the user can update the offset when the synchronization becomes too coarse. An easy way to synchronize multiple data loggers with one camera is to shake or tap all of them simultaneously in one

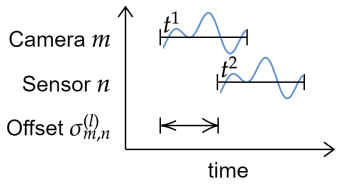

Figure 2: Synchronization diagram. The blue signals denote the movement from the same event.

video and follow the aforementioned procedure for each data logger. Because the synchronization is done visually, the accuracy of the synchronization mostly depends on the frame rate of the video plus a small error that depends on the level of difficulty in correlating the event in the video with $M(t)$. Thus, with a frame rate of 30 fps the accuracy is maximally 33 ms. However, the timestamp for the label is determined by the annotation in the graph. It is often easier to observe when the activity pattern changes by looking at the visualized movement data and this accuracy is bounded by the sampling rate of the sensor.

3 DISCUSSION

Approach A can achieve a high accuracy in the synchronization due to the red flash that occurs in the video stream. The synchronization accuracy in Approach B is lower because the annotator needs to determine how the movement of the subject correlates to the sensor data graph. A disadvantage to Approach A is that synchronization needs to be performed again each time a sensor or camera is restarted and the user must make sure to record the flash on the data logger. Approach B has a number of advantages: (1) Synchronization only needs to be done once, and slightly adjusted when collecting data over multiple days (2) It becomes easier to collect more data because multiple subjects can be recorded simultaneously (3) There is a higher chance an unusual activity is exercised by one of the subjects on camera (4) It is possible to track subjects from multiple angles and around corners (5) Data loggers are synchronized among each other after synchronizing to the same camera.

4 CONCLUSION

Timestamped visual indications on the data loggers result in accurate synchronization between the sensor data and ground truth video. However, solely using this technique for synchronization does not allow the use of multiple data loggers and cameras simultaneously. Synchronizing cameras with data loggers throughout a collection campaign using RTCs is less accurate in terms of synchronization but has multiple advantages that support the approach.

In future work, the best idea seems to be to combine both approaches and use visual keys on the data loggers to synchronize the RTC between cameras and sensors. The potential framework could be further optimized using automated pattern detection in the video instead of shaking or tapping the sensors. Each sensor can emit a unique n-bit blinking pattern. Computer vision can be used to interpret the blinking patterns in the recorded video(s) and

identify the sensors. Time information can be added to the pattern so that the offset between each camera and each sensor can be calibrated automatically. The synchronization would only need to be performed once for each camera in a dedicated recording where all the sensors and the blinking LEDs are visible to the camera. If clock drift error accumulates over time this can be corrected manually by the user (adding a small offset), by using the time information in the blinking pattern (if this is visible), or by repeating the synchronization recording. A common clock can be included in the video recordings to synchronize multiple cameras with each other.

ACKNOWLEDGMENTS

This research was supported by the Smart Parks Project, which involves the University of Twente, Wageningen University & Research, ASTRON Dwingeloo, and Leiden University. The Smart Parks Project is funded by the Netherlands Organisation for Scientific Research (NWO). This work was also supported by the United States National Science Foundation (NSF) under grant IIS-1406578.

REFERENCES

[1] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. 2013. A Public Domain Dataset for Human Activity Recognition Using Smartphones. In Proceedings of the 21th International European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. i6doc, Bruges, 437–442. http://www.i6doc.com/en/livre/?GCOI=28001100131010

[2] David Bannach, Kai Kunze, Jens Weppner, and Paul Lukowicz. 2010. Integrated tool chain for recording and handling large, multimodal context recognition data sets. In Proceedings of the 12th ACM International Conference Adjunct Papers on Ubiquitous Computing - Adjunct. ACM, Copenhagen, Denmark, 357–358. https://doi.org/10.1145/1864431.1864434

[3] Michael Jones and Zann Anderson. 2017. Accelerometer Data and Video Collected While Hiking and Climbing at UbiMount 2016. In Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers (UbiComp '17). ACM, New York, NY, USA, 1043–1046. https://doi.org/10.1145/3123024.3124445

[4] Michael Jones, Casey Walker, Zann Anderson, and Lawrence Thatcher. 2016. Automatic Detection of Alpine Ski Turns in Sensor Data. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16). ACM, New York, NY, USA, 856–860. https://doi.org/10.1145/2968219.2968535

[5] Jacob W. Kamminga. 2019. Matlab Movement Data Labeling Tool. (8 2019). https://doi.org/10.5281/zenodo.3364004

[6] Jacob W. Kamminga, Helena C. Bisby, Duc V. Le, Nirvana Meratnia, and Paul J. M. Havinga. 2017. Generic online animal activity recognition on collar tags. In Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers on - UbiComp '17. Association for Computing Machinery, Maui, HI, 597–606. https://doi.org/10.1145/3123024.3124407

[7] Jacob Wilhelm Kamminga, Duc V. Le, Jan Pieter Meijers, Helena Bisby, Nirvana Meratnia, and Paul J.M. Havinga. 2018. Robust Sensor-Orientation-Independent Feature Selection for Animal Activity Recognition on Collar Tags. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies IMWUT 2, 1 (2018), 1–27. https://doi.org/10.1145/3191747

[8] Jacob W Kamminga, Duc V. Le, Nirvana Meratnia, and Paul J.M. Havinga. 2019. Deep Unsupervised Representation Learning for Animal Activity Recognition. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. (2019), submitted.

[9] Jacob W Kamminga, Nirvana Meratnia, and Paul J.M. Havinga. 2019. Dataset: Horse Movement Data and Analysis of its Potential for Activity Recognition. In The 2nd Workshop on Data Acquisition To Analysis (DATA'19), November 10, 2019, New York, NY, USA. Association for Computing Machinery, New York, NY. https://doi.org/10.1145/3359427.3361908

[10] Cassim Ladha, Nils Hammerla, Emma Hughes, Patrick Olivier, and Thomas Ploetz. 2013. Dog's life: Wearable Activity Recognition for Dogs. In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13. ACM, Zurich, 415–418. https://doi.org/10.1145/2493432.2493519

[11] Paul Lukowicz, Gerald Pirkl, D Bannach, F Wagner, Alberto Calatroni, Kilian Förster, Thomas Holleczek, M Rossi, Daniel Roggen, Gerhard Tröster, J Doppler, C Holzmann, A Riener, Alois Ferscha, and Ricardo Chavarriaga. 2010. Recording a complex, multi modal activity data set for context recogntion. In 1st Workshop on Context-Systems Design, Evaluation and Optimisation at ARCS, 2010. Springer, Hannover, 1–6. http://www.duslab.de/cosdeo/

[12] MATLAB. 2018. version 9.5.0.944444 (R2018b). The MathWorks Inc., Natick, Massachusetts.