Time Synchronization on Android Devices for Mobile Construction Assessment

Maik Benndorf and Thomas Haenselmann
University Of Applied Sciences Mittweida
Faculty of Applied Computer Sciences & Biosciences
Germany, Mittweida 09648
Email: {benndorf,thomas.haenselmann}@hs-mittweida.de

***Abstract*—Within this work we aim to assess the structural integrity of buildings in the case of catastrophic events using several off-the shelf smart phones featuring vibration sensors. In order to compare the vibration samples obtained from different devices, precisely synchronized clocks are needed. In this article, we suggest how to align clocks based on sound beacons to mutually take clock drift and skew into account, in a precision which can be expected from traditional synchronization approaches like Network Time Protocol (NTP).**

***Keywords*–sensor networks; time synchronization; clock skew; clock drift**

I. INTRODUCTION

In cases of natural disasters like flooding, timely access to information about transport infrastructure is crucial for the first-aider. In particular, bridges could be fit for traffic, partially usable or be completely destroyed. During this research project we scrutinize, whether destruction influences the resonance frequency of a bridge. For that reason, we applied vibrometers of the type ”Beitzer System 9000” [1].

Figure 1. This figure shows the bridge for our tests.

However, in many cases such a device is not available in a timely manner in the areas struck by a disaster. Nonetheless, information about the state of transport routes is crucial for affected people as well as for aid organizations [2]. Hence, we scrutinized how the experiment mentioned above can be conducted with the help of acceleration sensors in mobile phones. During our experiments, the sensors were positioned at different points on the surface and the beams of the bridge. In order to compare vibrations of the devices every vibrometer is at least equipped with two mobile phones. Figure 1 shows the bridge of our tests. The bridge has a length of 30 meters, a width of 4 meters and weighs about 75 tons.

Figure 2 shows the recorded vibrations caused by a walking pedestrian. A large difference between both systems is the sample rate. While vibrometers measure with approximately 2,500 Hz, the sensors of mobile phones only

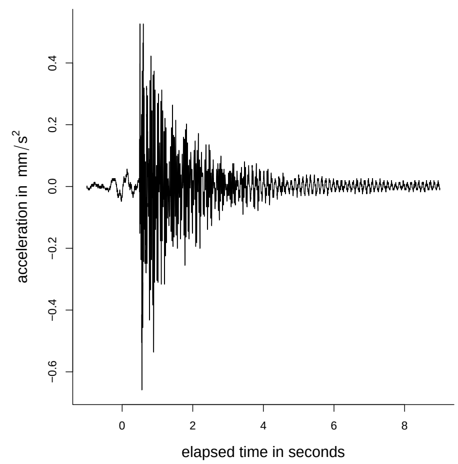


Figure 2. Vibrations of the bridge recorded by the vibrometer and triggered by a walking pedestrian.

measure with 100 Hz. In order to sample with a much higher virtual frequency, many mobile phones should be connected with each other. The start of measurements from different devices are naturally shifted against one another. In an ideal case, this leads to a higher sampling rate. For that purpose, the offset between these devices needs to be exactly determined, which requires all of them to agree on a common time base. Therefore, the time source should be the hardware clock. The literature follows different approaches to synchronize those devices. Most of these methods have limitations, e.g., a network connection for a NTP-based synchronization. In this article, we introduce an approach making use of the microphone of a mobile phone in order to synchronize the time and align the vibrations of different devices.

This paper is organized as follows: Section II is devoted to explain the related work in the field of synchronizing methods. Section III explains in detail our approach. Section IV shows how the clock-skew influences synchronization methods. A summary and an outlook for further research are shown in Section V.

II. RELATED WORK

The problem is related to the time synchronization in sensor networks. A lot of research has been done in this area. There are several methods for synchronizing physical clocks. These methods can be classified as ”internal” and ”external” synchronization. For an internal clock synchronization, all nodes accept the time of a reference node in a network [3].

 ISBN: 978-1-61208-490-9

In external clock synchronization, the value is taken from an external clock source, such as a common time service (e.g., NTP) or the Global Positioning System (GPS) [4].

A. Synchronization by NTP

NTP is a protocol for synchronizing computer clocks using a set of distributed servers around the world. This protocol is also known as Simple network time protocol. It is built on top of the User Datagram Protocol (UDP) [5]. The protocol was announced with a precision in the range of nanoseconds [6] [7].This protocol has been utilized in numerous clients for several years. Juda Levine [8] reports in 2011 about 5×10^9 requests per day. The accuracy of the Protocol and the related assets have been studied in numerous works [9]–[11]. The network latency has a major impact on the accuracy. Zhao et al. [10] evaluated the accuracy with less then 10 ms under Local Area Network (LAN) condition and less then 100 ms under Internet conditions. As a reference for their evaluations they used the time of the GPS.

B. Synchronization by GPS

The Global Positioning system was designed and is still under the control of the United States Department of Defense. Nevertheless, it is also freely accessible by anyone. The system consists currently of 32 operating satellites that are orbiting the earth at an altitude of approximately 20,000 km. Every satellite contains multiple atomic clocks that support very precise timing data [12]. For determining the position, the receiver needs signals from at least three of these satellites. The position of the receiver can be calculated by the difference between signal sent and received by the receiver. With this ability to receive very accurate data from multiple sources an accurate time can be obtained. In 2015 Mazur et al. [12] designates the accuracy of such time synchronizations within up to billionth of a second. This system is available anywhere in the world and has a very high accuracy. But it requires a direct line of sight to the satellite and an initial connection takes a long time in some cases.

C. Clock-Skew and Drift

Most computing devices are equipped with a hardware oscillator assisted computer clock. The frequency of the hardware oscillator determines the rate at which the clock runs [13]. This clock becomes inaccurate because the frequency varies. Figure 3 shows the difference between the clock drift - in this case the clock is below or ahead by a fix offset, the clock skew - here the offset is growing during the time, and the jittering - in this case the device clock is affected by internal (e.g., processor utilization) or external (e.g., temperature, humidity [14]) fluctuations. To keep these clocks in time, Zhenjiang Li et al. [15] uses the flickering lights of fluorescent lights.

D. Time Synchronization on Mobile Devices

One recent work in this area is provided by Lazik et al. [17]. They used ultrasonic beacons to synchronize the time on mobile devices. Therefore, they built up a network with one network master. The master is connected to a GPS receiver and transmits ultrasonic chirps in a frequency that is outside of the human hearing but still detectable by the microphone of smartphones. They reported that the devices could be synchronized with an average accuracy of 720 μs. At the beginning of their experiments they investigated devices with Android and iOS. They reported a high level jitter on the Android device (in the order of milliseconds and higher) and chose the iOS devices for the rest of their experiments. The high jittering is justified by Android's task scheduler. Within this work they also benchmarked the NTP timing performance (on iOS). They ran their experiments with three different communication channels, Long Term Evolution Technologie (LTE), Wireless Local Area Network (WLAN) and one idle WLAN router that is directly connected to a Stratum 1 NTP server fed by a dedicated GPS clock. Using LTE they measured an average jitter of 47 ms, the average WLAN jitter is measured with 30ms and finally the average jitter in the ideal case with the WLAN router connected to NTP Server is measured with 19.3 ms.

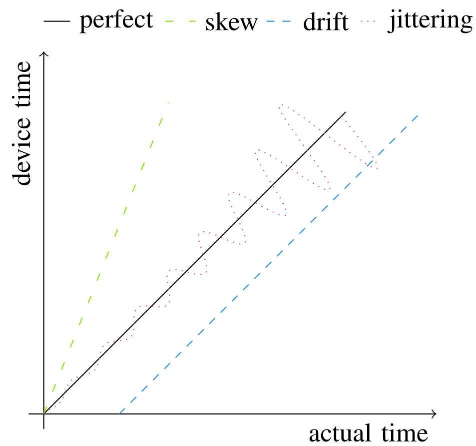


Figure 3. This figure shows the difference between a perfect clock, one with drift, one with a skew and one, as it is likely to occur in real [16].

III. SYNCHRONIZATION

In the previous section, we explained some synchronizing methods. Synchronization with a common time service will not be considered at the moment. This is because we can not make any assumptions about the availability, connection and bandwidth. A prerequisite of a synchronization among different devices is that they share the same network. In order to make the use case as simple as possible, it is not planned to create a network between those devices. A quite simple example for synchronization comes from sports. With a 100 m sprint, eight athletes stand in starting blocks. The race starts with an external signal which is same for each athlete.

This idea will be applied here. First, a sender and a receiver needs to be chosen. The sensors of modern mobile phone are suitable as receivers. A mobile phone is equipped with numerous sensors, e.g., accelerometer, gyroscope, microphone, etc. For synchronizing the time, a sensor with the highest possible sampling rate is required. The sample rate of the accelerometer is nearly 100 Hz. The gyroscope samples at a frequency up to 200 Hz. For these sensors, there is no advantage to the outlined problem. In contrast, the microphone samples up to 44,100 Hz. Therefore, we chose the microphone as receiver.

The sender (sound source) in our experiments is a choke used in athletic sports. This sound creates a distinctive peak (up to 93 dB) in the amplitude. It has almost no reverberation. Hence the time can be determined exactly. During our test the distance between sender and receiver was up to one meter almost the same for all devices. So the velocity of sound did not influence the result. The recorded peak and the corresponding timestamp can be used to determine an exact timestamp for the start. So, the data of different devices can easily be aligned.

A. *Devices used*

During the experiments, we used three smart phones with android operating system (Table I).

TABLE I. DEVICES USED

Manufacturer	Model	OS Version	Microphones
Motorola	MotoG (2nd Edition)	5.1.1	2
Sony	Xperia M2	5.1.1	1
Samsung	Galaxy A3	5.0.1	2

These devices are equipped with up to three MEMS-Microphones (micro-electro-mechanical-systems).

B. *Specifics*

Due to the operating system, there are some specifics that need to be mentioned.

- The sound is encoded by 16 bit. So values in the range from -32,768 to 32,767 with a maximum signal to noise ratio of 96.33 dB can be achieved.
- The audio samples are provided by the operating system as chunks. So only the time when a package is received can be measured. The time for the samples is calculated from the sample rate by interpolation. The size of the chunks depends on the buffersize that is proposed by android (for the three devices with 3,584 samples per chunk the same).
- There are many discussions about the usage of the function *System.nanoTime* [18]–[20]. So we choose the function *System.currentTimeMillis*. This function returns the elapsed time in milliseconds since midnight, January 1, 1970 (UTC).

To reduce external influences, we disable most of the applications on the device. In addition, the recorded data during the process is only held in memory and written after completion. While using this synchronization method, other mechanisms resulting in a synchronization, e.g., synchronization by local time with NTP Server, need to be switched off.

C. *Experiments*

We conducted about thirty experiments in six days. Within each experiment we created several peaks at distances between 5 seconds up to 1 minute. So we can compare the results from 5 seconds up to 24 hours. Figure 4 shows one raw experiment within four peaks at 0 s, 10 s, 30 s and 60 s. The data in this figure is aligned by the time on device. Please note that, there is an initial offset for each clock in the beginning. To align these samples by the amplitude, we first calculated their absolute values.

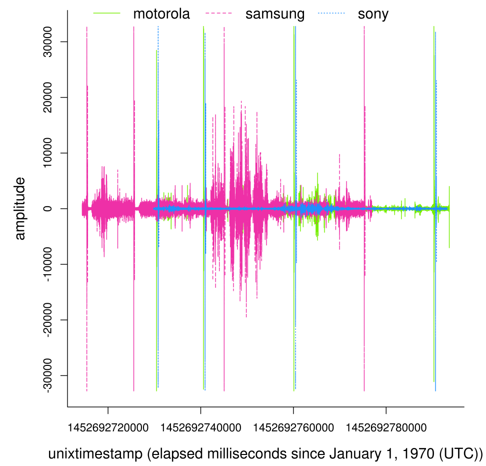


Figure 4. In this figure, the peaks are aligned by the unixtimestamp. Please note the offset between the devices, especially, the samsung is 20 seconds ahead of the other devices.

In this example, the first peak is used to align the samples of different devices. The alignment result for these three devices can be seen in Figure 5a. In Figure 5b, the data is reduced to an absolute amplitude above 32,000. In this perspective, all the data is aligned. By zooming in at 60 seconds (Figure 5c) an offset between these devices can be seen. It can be noticed that the offset between the devices becomes significantly larger during the time. This fact leads us to look deeper into the clock skew of the devices.

IV. Clock Skew

Since the figures show an obvious offset after only 60 seconds, we scrutinize the skew of the clocks further.

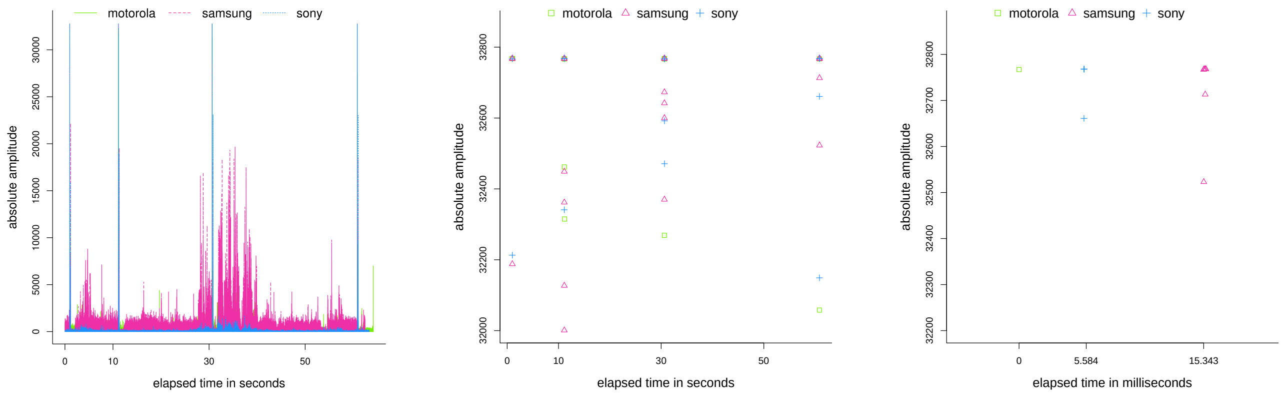
TABLE II. THE DISTANCES BETWEEN THE DEVICES CALCULATED BY THE PEAKS IN MILLISECONDS.

time	Motorola - Samsung	Motorola - Sony	Samsung - Sony
10 s	-1	5	6
20 s	-8	-26	-18
30 s	-9	-21	-12
50 s	-13	-10	3
60 s	-15	-6	9

Table II depicts the offset of the clocks from the above illustrated Figure (5c).

TABLE III. THIS TABLE SHOWS THE MEAN DISTANCES BETWEEN THE DEVICES AND THEIR STANDARD DEVIATION AFTER ELAPSED TIME IN MILLISECONDS.

	Motorola - Samsung		Motorola - Sony		Samsung - Sony		
time	mean	std	mean	std	mean	std	values
5 s	-0.57	3.45	0.14	7.10	0.71	4.38	7
10 s	-0.31	4.39	-0.22	4.63	0.0	4.42	105
20 s	0.56	5.16	0.36	5.13	-0.2	4.091	80
30 s	0.42	6.71	0.19	5.94	-0.22	3.91	127
60 s	-0.09	6.32	-0.12	5.46	-0.02	4.88	72
90 s	-0.16	5.08	-0.81	4.19	-0.65	4.68	49
2 m	-0.30	4.52	-1.27	5.25	-0.96	5.12	33
1 h	-8.71	4.68	-19.09	4.32	-10.38	4.28	21
6 h	-54.44	8.53	-99.22	7.32	-44.77	3.34	9
12 h	-158.33	4.22	-424.67	2.16	-266.34	4.92	6
24 h	-212.33	3.21	-531	6.55	-318.67	4.16	3



(a) aligned data

(b) aligned data filtered by maximum amplitude

(c) offset after 60 seconds

Figure 5. Figure shows an aligned Result (a) full plotted, (b) reduced to the peaks and (c) the offset in this experiment after 60 seconds.

While the distance between Motorola and Samsung is growing, the other distances are jittering. Over all the measurements, there is an error between these devices. Table III provides an overview of the distances between the devices. For this purpose, the mean value and their standard deviation is given. The results in this table vary very strongly and become stable only over longer periods of time when the standard deviation can be neglected. To reduce the errors, these experiments must be repeated for synchronization of the devices. One reason for this may be the jittering, that is also mentioned by Lazik et al. [17]. In order to evaluate the skew of the devices, we also set up a GPS timing within our experiments. The Figure 6 shows the time on device compared to the GPS time.

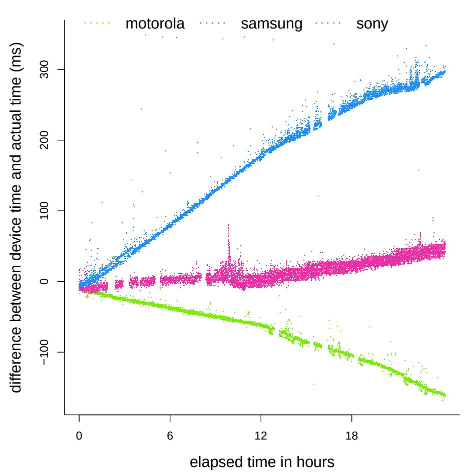


Figure 6. The clocks skew of the devices within 24 hours compared to the timestamp given by GPS.

So we can confirm the above measured distances between these three devices. Furthermore, we can calculate an average offset for this (Table IV). This offset can be used to interpolate / extrapolate the time on device. Thus, the data of the synchronized devices can be aligned. One example of such an alignment can be seen in Figure 7, where the data of two different devices (positioned at 1/6 of length and 1/2 of width of the above mentioned bridge) is aligned by the synchronized time.

TABLE IV. THE AVERAGE OFFSET PER SECOND ON DEVICE IN NANOSECONDS.

Motorola		Samsung		Sony	
mean	std	mean	std	mean	std
23.59	366.33	0.73	7.35	15.79	356.34

V. CONCLUSION

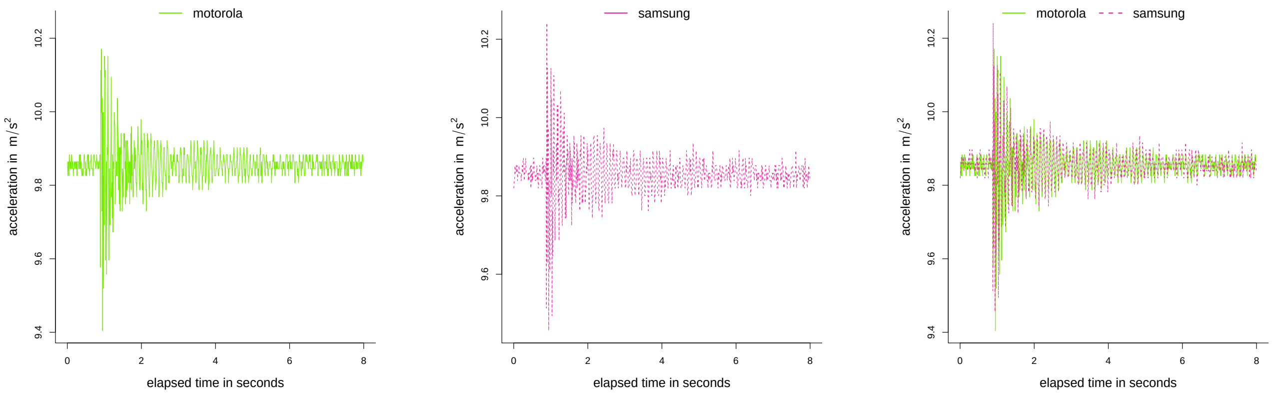
In this article, we introduced an approach that synchronizes the time between mobile phones. Therefore, the microphone sensors were utilized. The devices were synchronized by an external sound event. This event can be recognized by observing the amplitude of the built in microphones and the corresponding timestamp. Besides the synchronization of time, the individual clock skew and drift can be derived. So we can achieve an accuracy of up to 5 ms. One reason for this value is the strong jittering on these devices. Theoretically, the achievable resolution is up to the order of microseconds, since the microphones work with a sample rate of 44,100 Hz. Thus, this approach would provide a more accurate result than the synchronization via NTP but would still be behind than the synchronization using GPS. Therefore, further investigations of the jittering is necessary, which will follow.

One advantage of this approach is that network, data or GPS connections are not necessary. The only limitation is the distance between the sound source and the devices. This means that this approach is highly suitable for areas where the above mentioned services are not available.

REFERENCES

[1] M. Messtechnik., "Frequenzanalyse durch fft," 2009, (retrieved: 31-May-2016). [Online]. Available: http://www.beitzer.de/products/FFT.html

[2] L. Özdamar, E. Ekinci, and B. Küçükyazici, "Emergency logistics planning in natural disasters," Annals of operations research, vol. 129, no. 1-4, 2004, pp. 217–245.



(a) vibrations of motorola (b) vibrations of samsung (c) vibrations of both devices combined

Figure 7. This figure shows same vibrations as Figure 2, recorded by the smartphones. Figures (a) shows the vibrations recorded by motorola, (b) recorded by samsung and (c) the data of both devices aligned after time synchronization.

[3] K. Sun, P. Ning, and C. Wang, "Tinysersync: secure and resilient time synchronization in wireless sensor networks," in Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 264–277.

[4] B. Kusy et al., "Elapsed time on arrival: a simple and versatile primitive for canonical time synchronisation services," International Journal of Ad Hoc and Ubiquitous Computing, vol. 1, no. 4, 2006, pp. 239–251.

[5] J. Postel, "Rfc 768 - user datagram protocol," Isi, 1980.

[6] D. L. Mills, "Network time protocol (ntp)," Network, 1985.

[7] ——, "Internet time synchronization: the network time protocol," Communications, IEEE Transactions on, vol. 39, no. 10, 1991, pp. 1482–1493.

[8] J. Levine, "Timing in telecommunications networks," Metrologia, vol. 48, no. 4, 2011, p. S203.

[9] A. N. Novick and M. A. Lombardi, "Practical limitations of ntp time transfer," in Frequency Control Symposium & the European Frequency and Time Forum (FCS), 2015 Joint Conference of the IEEE International. IEEE, 2015, pp. 570–574.

[10] K. Zhao, A. Zhang, and K. Liang, "Research on the uncertainty evaluation of network time service system," in European Frequency and Time Forum (EFTF), 2012. IEEE, 2012, pp. 122–125.

[11] S. Ping, "Delay measurement time synchronization for wireless sensor networks," Intel Research Berkeley Lab, 2003.

[12] D. C. Mazur, R. A. Entzminger, J. A. Kay, and P. A. Morell, "Time synchronization mechanisms for the industrial marketplace," in Industrial & Commercial Power Systems Technical Conference (I&CPS), 2015 IEEE/IAS 51st. IEEE, 2015, pp. 1–7.

[13] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," Network, IEEE, vol. 18, no. 4, 2004, pp. 45–50.

[14] J. Aron, "Flickering lights could help smartphones keep time," New Scientist, vol. 215, no. 2874, 2012, p. 22.

[15] Z. Li et al., "Flight: Clock calibration using fluorescent lighting," in Proceedings of the 18th annual international conference on Mobile computing and networking. ACM, 2012, pp. 329–340.

[16] T. Haenselmann, Wireless Sensor Networks: Design Principles for Scattered Systems. Oldenbourg Verlag, 2011.

[17] P. Lazik, N. Rajagopal, B. Sinopoli, and A. Rowe, "Ultrasonic time synchronization and ranging on smartphones," in Real-Time and Embedded Technology and Applications Symposium (RTAS), 2015 IEEE. IEEE, 2015, pp. 108–118.

[18] A. Shipilv, "Nanotrusting the nanotime," 2014, [retrieved: 2016-05-16]. [Online]. Available: http://shipilev.net/blog/2014/nanotrusting-nanotime/

[19] S. Kobylansky, "What is behind system.nanotime()?" 2012, (retrieved: 16-May-2016). [Online]. Available: http://stas-blogspot.blogspot.de/2012/02/what-is-behind-systemnanotime.html

[20] M. Kuperberg, M. Krogmann, and R. Reussner, "Timermeter: Quantifying properties of software timers for system analysis," in Quantitative Evaluation of Systems, 2009. QEST'09. Sixth International Conference on the. IEEE, 2009, pp. 85–94.