An Analysis of Time Drift in Hand-Held Recording Devices

Mario Guggenberger, Mathias Lux, and Laszlo Böszörmenyi

Institute of Information Technology,
Alpen-Adria-Universität Klagenfurt,
9020 Klagenfurt am Wörthersee, Austria
{mg,mlux,lb}@itec.aau.at

Abstract. Automatic synchronization of audio and video recordings from events like music concerts, sports, or speeches, gathered from heterogeneous sources like smartphones and digital cameras, is an interesting topic with lots of promising use-cases. There are already many published methods, unfortunately none of them takes time drift into account. Time drift is inherent in every recording device, resulting from random and systematic errors in oscillators. This effect leads to audio and video sampling rates deviating from their nominal rates, effectively leading to different playback speeds of parallel recordings, with deltas measured up to 60 ms/min. In this paper, we present experiments and measurements showing that time drift is an existing problem that cannot be ignored when good quality results are demanded. Therefore, it needs to be taken care of in future synchronization methods and algorithms.

Keywords: Audio, video, multimedia, crowd, events, synchronization, clock drift, time drift

1 Introduction

Digital cameras, smartphones and tablets are ubiquitous devices that many people carry with them all day. They make it incredibly easy to record good quality audio or video at all kinds of events, e.g. concerts, sports, or speeches. Synchronizing all those recordings from an event opens up various interesting use-cases like detecting key moments by looking at the frequency of concurrent recordings, temporal stitching of clips to get a complete and continuous coverage of a whole event, creating vivid videos by switching between different perspectives or showing different shots side-by-side, improving presentation quality by picking the best audio and video tracks from concurrent recordings, or reconstructing 3D scenes from recordings of different angles. Many approaches for automatic synchronization have already been proposed, and a recent overview of methods is presented in [2]. Unfortunately, all devices have an inherent error of time, which results in slightly different recording speeds across devices, effectively separating them into different time scales. We first ran into this problem in private amateur projects, e.g. recording concerts with multiple consumer cameras from different

perspectives or recording talks with separate audio and video recording devices, later during our work on an automatic audio synchronization software [5], and most recently while trying to synchronize the Jiku Mobile Video Dataset [12], a dataset with crowd-sourced smartphone video recordings. We were not able to get satisfactory results, neither with an automatic fingerprint-based method, nor by doing it manually. Parallel recordings were often out of sync, resulting in audio echoes, whose intensity varied between hardly noticeable to totally unacceptable. This problem, called drift, has also been identified in [2]. Drift by itself is not a new phenomenon, but it has been mostly ignored in the multimedia community. It has been covered in other areas, e.g. in network delay measurements [11] and for the identification of physical network devices through fingerprinting [14].

Drift in electronic devices is an error in the oscillators that drive, coordinate and synchronize the digital circuitries. Due to random and systematic influences, no oscillator runs at its specified nominal frequency. This frequency is specified in hertz (Hz), an SI unit based upon the second as defined by the Coordinated Universal Time (UTC) world time scale [3]. Since oscillators are also integral parts of clocks, this means that no clock, even when perfectly set exactly to UTC time at some point, can keep the correct time infinitely long. Even more important, oscillators are responsible to accurately time audio sampling rates and video frame rates during recording and playback. They also drive the system clock of a device which is additionally used to e.g. stamp recordings with the capture time. In case of recordings, this boils down to system and user errors in timestamps denoting the moment *when* a recording was captured, and system errors in timing of *how long* the capture process was running. With accurate clocks, synchronization would be as easy as reading the timestamps and aligning the recordings on a common timeline. All of the currently published synchronization methods handle only the unreliable timestamps and ignore the drift altogether. This concerns mainly recordings from uncontrolled environments, respectively consumer and prosumer equipment used by amateurs; professional environments usually avoid the problem by feeding all devices with a common master clock signal at recording time.

With this paper, we want to highlight a problem to the community that, in our opinion, it is not aware of, but that we think it should consider. This paper is not intended to provide concrete solutions, albeit we discuss suggestions on how the problem can be tackled. In the remainder of this paper, we continue to give a more detailed introduction into the issue. We present a method to measure drift with a precision that is sufficient for multimedia use cases and to underline our claim. We present experiments that provide proof that drift is a recent and pressing problem in current devices that needs to be handled by synchronization methods to achieve good quality results. We then continue with suggestions on how to detect and how to remove or compensate drift, and finish with our conclusion.

1.1 Oscillators

Oscillators exhibit a number of frequency/period instabilities and are usually specified in terms of short term, long term and environmental frequency stability [1]. Short term instabilities are known as jitter and degrade audio quality, but are not of our concern. Responsible for the drift problem are the long term and environmental instabilities that are a result of the initial error in crystal manufacturing, aging, and dependencies from temperature, vibration, and power supply [1, 13]. Temperature usually has the biggest impact. The error ϵ in accuracy is measured in parts per million (ppm) and called *drift*. One ppm can be regarded as one microsecond per second, meaning that a clock with an oscillator specified at -10 ppm looses $10\,\mu s/s$ from UTC time, amounting to almost one second per day. According to various service manuals of smartphones and tablets, commonly used oscillators are either crystal oscillators (XO), temperature controlled XOs (TCXO), or voltage controlled TCXOs (VCTCXO). XOs are usually specified with an accuracy of ± 10 ppm to ± 100 ppm, TCXOs are more accurate at ± 1 ppm which is why they are often used to drive audio components. In comparison, atomic oscillators are accurate to at least 0.001 ppm, UTC is accurate to ≈ 0.00000001 ppm [15] and even more precise oscillators exist.

1.2 Time Drift

Drift can be regarded as a deviation of a value in a time series from an ideal time series. In our case, the ideal time series is a timebase itself, e.g. UTC time, and the clock drift is a deviation from it, where the drift factor is another function over time which the drift accumulates by. Clock drift leads to deviations in the audio playback and recording sample rates in devices, that result in pitch shifts in, and bandwidth changes of the transmitted signals, but also impact the runtime of played or recorded files. It also leads to changes in video frame rates. This change of runtime is what we call the time drift. We specify it in ppm, but for better understanding in the multimedia domain, it is sometimes more intuitive to specify it in milliseconds per minute (ms/min) or per hour (ms/h).

1.3 Synchronization

Synchronization of two or more audio/video recordings is the act of mapping each moment in each recording to a common timeline such that all captured moments, which simultaneously happened at recording time, are mapped to the same time instant on the timeline. This mapping can essentially be divided into two steps: (i) compensating the drift in all recordings, and (ii) positioning the recordings on a timeline. Drift compensation is the process of removing or altering the drift inherent in all recordings to establish a mapping to the common time over their whole runtime. When the common synchronization time is UTC, we call it absolute drift compensation and all recordings need to be fit to that time. An alternative is relative drift compensation, where the inherent time in

recordings of one device is taken as the common time and only recordings from other devices need to be compensated. This is still a simplification as it assumes that the drift of a recording device is constant, which we will later show is wrong, but the variance in the drift is a much smaller problem than the existence of the drift itself. The result of such a mapping is a timeline of synchronized recordings.

2 Measurements

To achieve high precision and absolute measurements, we chose to use GPS as the reference time source, since it is the cheapest and easiest way to obtain a high precision time signal. While GPS has been shown to be accurate up to ≈ 3 ns [9], the datasheet of our particular Garmin GPS receiver [4] specifies an accuracy of 1 µs, which is sufficient for our measurements. The GPS receiver emits a one pulse per second (PPS) signal that we fed into one channel of an audio capture interface connected to a computer workstation. The PPS signal was adjusted to audio line-level through a simple voltage divider circuit. The second channel of the audio device was fed with a 440 Hz sine wave test signal played back on the device to be measured. To obtain absolute measurement results, we used the audio signal analysis software Spectrum Lab[1] which measures the computer workstation's drift by analyzing the GPS signal and removing it from the analysis result of the input signal from the measured device. With this setup, we could measure the playback drift factor $d_p = 440/f_p$, where f_p is the measured frequency output from the measured playback device. Since the hardware audio codec of a device C derives the playback sample rate from the same clock signal as the recording sample rate, we can assume that both sample rates are equal, thus both drift factors are equal and $d_p^C \equiv d_r^C$. The drift factor can be converted to ppm by calculating $\epsilon = (d-1) \times 10^6$. For quick estimates of inherent drift in multimedia devices, we have published an Android app that is capable of on-the-fly measurements built upon this method [6].

2.1 Inter-Device Drift

To investigate the drift between different devices, we measured 16 devices in an isolation booth of a recording studio at constant room temperature by playing the test signal from a laptop computer and recording it in parallel by each device for 90 minutes. We then filtered the recordings with a band-pass filter of 40 Hz width around the center at 440 Hz to remove environmental noise, and deduced their recording drift d_r from the average frequency over the whole length. We additionally measured the absolute drift of the laptop and removed it from the device measurements to convert them to absolute drifts from UTC. It is important to note that the drifts cannot be added but must be multiplied to get correct results. To calculate the absolute drift d_t^D of device D, the playback drift of the notebook d_p^N must be inserted into the formula $d_t^D = d_p^N \times d_r^D$. The

[1] `http://www.qsl.net/dl4yhf/spectra1.html`

Table 1. Drift measurements (ppm) of various hand-held recording devices (Tablet, Smartphone, MP3 Player, Video Camera, Audio Recorder).

Device	Type	audio	video
Samsung Galaxy Note 10.1	T	17.03	13.70
Samsung Galaxy SII	S	273.93	272.46
Samsung Galaxy Spica	S	-15.13	n/a
iRiver H120	M	93.98	n/a
iRiver H320	M	57.96	n/a
Canon HF10	V	n/a	6.45
Acer Iconia A200	T	13.39	-554.19
Apple iPad 2 Wi-Fi	T	13.73	11.96
Apple iPod touch 4G	M	416.76	413.64
M-Audio Microtrack 24/96	A	-40.42	n/a
LG Nexus 4 (rev. 10)	S	6.74	3.39
LG Nexus 4 (rev. 11)	S	4.12	1.88
Asus Nexus 7 2012 Wi-Fi	T	2.92	2.15
Sony PCM-M10	A	8.61	n/a
Editor UA-5	A	-3.21	n/a
Zoom R16	A	23.46	n/a

results are listed in Table 1. It is interesting to note that two devices, marketed as exceptionally high-end, suffer from huge drift making compensation utterly important, even for short clips. Another interesting observation is that almost all devices run too fast and just a small fraction too slow. We repeated the measurement where possible, this time recording the test signal on video instead of pure audio. The results in Table 1 show us that the drifts generally decreased, which is typical for the increased temperature of the oscillators resulting from higher computational demands and lit screens. The most interesting point, however, is the immensely different drift of the Acer tablet, leading to the assumption that this particular device does use different time sources for audio and video recordings, against the usual practice of timing video frames with the audio clock. This leads to the conclusion that video drift cannot be assumed to be the same as audio drift and needs to be determined separately. Experiments on selected devices have also shown that the drift of the user-visible clocks of devices, taken offline to avoid time synchronization over a network, equals the drift measured in the audio signal.

2.2 Intra-Device Drift

The next series of measurement answers the question how big the drift variance between devices from the same make, model and production batch are. For this measurement, we took five Nexus 5 smartphones and eight Nexus 7 tablets

bought at the same time in the same store. We therefore assume each type coming from the same production batch. We measured all of them on our measurement workstation by playing back the test signal, and again conducted a second run recording the signal on video for 20 minutes with the same method as described in the previous subsection. Results are listed in Table 2, and again show a similar decrease of video drift as before. They also show a variance of a few ppm between devices. The most interesting point is that their drift is generally pretty low and uniform, making these devices good choices for parallel recordings and bypassing the need of drift compensation if the individual recordings are kept reasonably short.

Table 2. Drift measurements of 5 LG Nexus 5 smartphones and 8 Asus Nexus 7 (2013) tablets from the same production batches with their means (λ) and standard deviations (σ).

		audio drift		video drift	
Device	No	ppm	ms/h	ppm	ms/h
Nexus 5	1	6.89	25	5.17	19
	2	4.51	16	1.41	5
	3	7.36	26	4.30	15
	4	7.82	28	5.01	18
	5	5.61	20	4.25	15
	λ	6.44	23	4.03	15
	σ	1.36	5	1.52	5
Nexus 7	1	5.11	18	0.88	3
	2	8.80	32	2.66	10
	3	8.88	32	3.66	13
	4	6.98	25	2.30	8
	5	8.77	32	3.82	14
	6	8.48	31	5.04	18
	7	5.89	21	1.21	4
	8	6.62	24	4.14	15
	λ	7.44	27	2.97	11
	σ	1.49	5	1.46	5

2.3 Temperature Influence

We already indicated that the temperature is the major influence on oscillator drift rates. Figure 1 shows a plot of the room temperature in our office and the drift of our measurement workstation as calculated from GPS time, recorded

over five winter days. It clearly shows that these two variables have a strong inverse correlation and that temperature is our primary concern. The two sudden drops in temperature and corresponding spikes in drift result from venting our office by opening the windows at an outside temperature of about 0 °C. Plotting the 5-minute moving average drift over time from the devices measured in the previous subsection as shown in Figure 2 demonstrates the warm-up phase of the devices during which the drift is not linear. It takes about 10 minutes for the devices to reach their working temperature and for the drift to change into a linear progression. This implicates that recordings of short clips with cool-down pauses in between will have a higher inherent average drift rate than long running recordings. Finally, we measured a few devices under extreme conditions to discover how much impact the temperature is expected to have in the worst case. We put them into a freezer and cooled them down to −20 °C, then moved them onto a heater with an air temperature of +50 °C and waited until the maximum drift was reached. We think this covers almost all situations in which recordings with consumer hand-held devices are made, and it is also the range in which crystal oscillators have an almost linear dependency between drift and temperature [13]. The results are shown in Table 3.

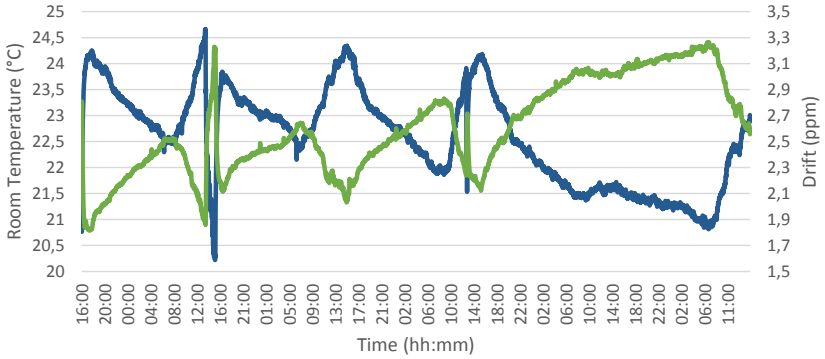

Fig. 1. Impact of room temperature (blue) on the computer workstation's drift (green) over five days.

Table 3. The influence of extreme temperature on device drift rates.

	−20 °C		+50 °C	
Device	ppm	ms/h	ppm	ms/h
iRiver H120	71.27	257	94.37	340
Asus Nexus 7 (2012)	0.49	2	4.95	18
Samsung Nexus S	11.04	40	20.54	74

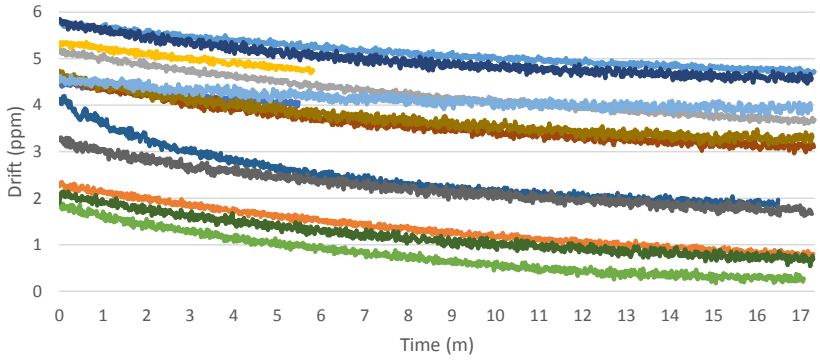

Fig. 2. 5-minute moving averages of drifts in devices recording HD video. The warm-up phase is clearly visible during the first minutes until they reach their working temperature and the drift stabilizes.

3 Drift Compensation

With the experiments and measurements outlined above, we have shown that time drift is a current and existent problem. Examples like the Acer Android tablet and the Apple iPod touch, whose video recordings drift apart by 60 ms/min (amounting to almost 3.5 s/h), clearly demonstrate that it is essential to remove drift to keep recordings in sync over time. This removal, drift compensation, is basically a contraction or expansion of the duration of each affected audio and/or video signal, leading to a mapping of all streams onto the same time scale.

The first step is always to determine the drift that needs to be compensated. Except for directly measuring recording devices as shown in Section 2, it is also possible to detect and measure drift if only the recordings are given. The most obvious method is manual measurement, which is easy for a pair of recordings, but gets tedious the more recordings are involved. It can be done by manually establishing synchronization points at the start and the end of an overlapping interval of two recordings, and calculating the difference of the lengths of the corresponding intervals in both recordings. Automatic synchronization methods can also be changed to incorporate drift handling. Approaches based on the correlation of feature series, e.g. [2], can consider using dynamic time warping [10] instead of cross-correlation, which we have already shown to work in a demo application [5]. Approaches that yield multiple discrete synchronization points between pairs of recordings, e.g. fingerprinting methods like in [7], can employ a method similar to the manual approach above, by applying linear regression to the relative time offsets between the synchronization points and calculating the slope of the fitted line.

The next two subsections cover suggestions for drift compensation of audio and video streams, we are however not going into detail as this is out of scope of this paper. To compensate drift in multimedia audio/video streams, we suggest

to split them into their elementary streams and process them separately with the appropriate method.

3.1 Audio

Audio drift can be easily removed by resampling. Changing the duration of an audio signal by resampling usually has the effect of a pitch change, which is often unwanted. To avoid the effect, elaborate pitch-preserving time stretching algorithms have been proposed [16]. In the case of drift, however, a pitch change is already introduced by the drifted recording process and inherent in the signal, but it is usually too small to be noticed by the human ear. Relative drift compensation by resampling would again shift the pitch of the signal and not necessarily remove or decrease it. Absolute compensation by resampling completely removes the pitch shift from the recording, reconstructs the originally recorded signal, and is therefore the optimal choice.

3.2 Video

Video resampling is a much more complex problem. In comparison to audio, a video stream has less temporal samples (frames) but the complexity comes from their twodimensionality which requires handling of motion between consecutive frames. Simple video resampling methods like frame averaging or frame repetition/skipping incur highly visible video quality degradations. Methods based on motion compensation produce better results but are much more complex and computationally expensive [8], and artifacts still remain visible to the trained eye. We propose that video streams should be left untouched whenever possible to preserve quality. Due to the low number of frames per second, videos can be allowed a little bit of drift which goes completely undetected if it stays below the frame presentation interval. This could be compensated by shifting the complexity from image processing to drift minimization. Examples are skipping or repeating frames in particular sections of low movement or uniform colors, e.g. black frames at a nighttime concert when the lights go off, or by intelligently cutting between different recordings to lose or catch up drifted time.

4 Conclusion

In this paper we have shown that time drift is a real problem that needs to be taken care of in the domain of multimedia synchronization. This especially concerns recordings that are crowd-sourced and/or crawled from Internet sites or social networks, where control over the recording devices being used is not possible. For coordinated amateur productions, where there is control over the devices, there is at least the possibility to select devices with minimized relative drift to avoid post-processing.

Drift is inherent in recording devices, and therefore inherent in all audio and video recordings, and makes the exact synchronization of recordings more complex than current synchronization methods suggest. Depending on the signal's

content and the consumer's experience, an audio drift of 10 ms may or may not be noticeable, but a drift of 300 ms after five minutes is unacceptable, even for video frames. It not only ruins the experience of a human consumer, but also makes any kind of post-processing error-prone.

We have suggested ways of detecting and dealing with drift, but questions are still open on how much drift is acceptable, and how much it negatively impacts the performance of content-based synchronization methods. We know that synchronization methods that synchronize recording upon a single point in time do not yield satisfying results for drifted recordings as the synchronization gets lost with increasing distance to the synchronization point. We hypothesize that synchronization methods that either directly incorporate drift handling, or are used on recordings where drift has been compensated as a pre-processing step, yield better results, and leave that also open for further discussion.

Acknowledgments. This work was supported by Lakeside Labs GmbH, Klagenfurt, Austria, and funding from the European Regional Development Fund (ERDF) and the Carinthian Economic Promotion Fund (KWF) under grant 20214/22573/33955. We also thank Wolfgang Büscher for his great work on the software Spectrum Lab and offering it for free.

References

1. Cardinal Components Inc. *Clock Oscillator Stability (No. A.N. 1006).*
2. A. L. Casanovas and A. Cavallaro. Audio-visual events for multi-camera synchronization. *Multimedia Tools and Applications*, March 2014.
3. C. C. H. David W. Allen, Neil Ashby. *Agilent AN 1289 The Science of Timekeeping.*
4. Garmin International, Inc., Olathe, Kansas. *GPS 18x Technical Specifications*, Oct 2011.
5. M. Guggenberger, M. Lux, and L. Böszörmenyi. Audioalign - synchronization of a/v-streams based on audio data. In *Multimedia (ISM), 2012 IEEE International Symposium on*, pages 382–383, Dec 2012.
6. M. Guggenberger, M. Lux, and L. Böszörmenyi. Clockdrift: A mobile application for measuring drift in multimedia devices. In *Proceedings of the 22st ACM International Conference on Multimedia*, MM '14, New York, NY, USA, 2014. ACM.
7. L. Kennedy and M. Naaman. Less talk, more rock: Automated organization of community-contributed collections of concert videos. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 311–320, New York, NY, USA, 2009. ACM.
8. C.-H. Kuo, L.-C. Chang, Z.-W. Liu, and B.-D. Liu. System level design of a spatio-temporal video resampling architecture. In *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, pages 2797–2800, May 2008.
9. M. A. Lombardi. The use of gps disciplined oscillators as primary frequency standards for calibration and metrology laboratories. *Measure: The Journal of Measurement Science*, 3(3):56–65, 2008.
10. M. Müller. *Information Retrieval for Music and Motion.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

11. S. Moon, P. Skelly, and D. Towsley. Estimation and removal of clock skew from network delay measurements. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 227–234 vol.1, Mar 1999.
12. M. Saini, S. P. Venkatagiri, W. T. Ooi, and M. C. Chan. The jiku mobile video dataset. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys '13, pages 108–113, New York, NY, USA, 2013. ACM.
13. Semtech. *Improving the Accuracy of a Crystal Oscillator (AN1200.07)*, Jan. 2009.
14. S. Sharma, A. Hussain, and H. Saran. Experience with heterogenous clock-skew based device fingerprinting. In *Proceedings of the 2012 Workshop on Learning from Authoritative Security Experiment Results*, LASER '12, pages 9–18, New York, NY, USA, 2012. ACM.
15. C. Thomas. Stability and accuracy of international atomic time tai. In *European Frequency and Time Forum, 1996. EFTF 96., Tenth (IEE Conf. Publ. 418)*, pages 520–527, Mar 1996.
16. U. Zoelzer, editor. *Dafx: Digital Audio Effects*. John Wiley & Sons, Inc., New York, NY, USA, 2002.