Potentials and Limitations of WIFI-Positioning Using Time-of-Flight

Lorenz Schauer, Florian Dorfmeister and Marco Maier
Mobile and Distributed Systems Group
Ludwig Maximilian University of Munich
Munich, Germany
Email: {firstname.lastname}@ifi.lmu.de

***Abstract*—Many existing approaches for WIFI-based indoor positioning systems use the received signal strength indicator (RSSI) of all the access points in reach to estimate the current location of a mobile device. Those systems' accuracy, however, is strongly influenced by external interferences and suffers both from short-term and long-term changes in the respective environments. Time-of-Flight-based (TOF) approaches bypass these problems by relying on the relation between distance and the time it takes a radio signal to travel that distance. Furthermore and in contrast to RSSI-based fingerprinting methods, they require neither a time-consuming calibration phase nor an extensive database. In this paper, we assess the advantages and limitations of TOF-based positioning techniques and compare different approaches in literature in terms of communication flow, hardware components, method of time measurement and positioning accuracy. Additionally, we present a novel approach using NULL-ACK-sequences, off-the-shelf hardware components and the CPU's time stamp counter offering a nanosecond resolution. An association with access points is hence not required and there is no need for modifications of client or infrastructure WIFI-components. We evaluate our system in various settings. Our results indicate that the accuracy of such TOF-based approaches depends on both the used hardware and the characteristics of the given environment. We find that time fluctuations caused by varying delays of the interrupt service routine as well as multipath effects render precise distance estimations based on a single measurement infeasible. In order to obtain stable ranging results we try to minimize these effects by utilizing a high amount of NULL-ACK-sequences. We investigate several filters and statistical estimations and compare them within our system settings. Using a band-pass filter and taking the average of a series of measurements, we are able to achieve a ranging accuracy with a mean absolute error of less than 1.33 meters in an ideal environment.**

I. Introduction

In the middle of October 2011 the first two operative satellites of *Galileo*, the new European global navigation satellite system (GNSS), were launched into orbit. With *Galileo*, the European Union hopes both to achieve huge economical benefits and to become independent of other existing GNSS such as the American *NAVSTAR-GPS* or the Russian *GLONASS*. *Galileo* promises a higher accuracy and reliability, while at the same time causing costs for development until the start of the first satellite of up to 1.1 billion Euro. The operation costs will be 800 million Euro a year. The Europeans' willingness to make this high investment shows that an adequate positioning system is absolutely important and necessary for today's mobile world.

However, while GNSSs work well outdoors, indoor positioning is hard to reach by these, given the fact that satellite signals are highly damped by roofs, walls and other obstacles. Thus, other ways have to be investigated to realize an adequate means for indoor positioning which is required for location-based services within buildings. Generally, every existing radio communication technique like RFID, Bluetooth or WIFI can be used for that purpose. Most of today's research approaches towards indoor positioning, however, focus on WIFI due to the fact that it is standardized by IEEE 802.11 and hence, is widely used all over the world. This makes WIFI-based positioning applicable in many indoor and outdoor environments. Furthermore, WIFI-hardware components are quite cheap and already integrated into state-of-the-art mobile devices like laptops, tablets and smartphones.

However, up to and including the development of the currently used IEEE 802.11n standard, WIFI has not been designed for positioning purposes. E.g., there are no extra protocols or adequate frame types which allow for on-the-fly positioning. That is the reason why WIFI-positioning has become a huge research field and many different approaches have been developed. Most of them are based on the received signal strength indicator (RSSI) of all the access points in reach in order to estimate a mobile device's location. A very common RSSI-based technique is *WIFI fingerprinting*, which is able to achieve acceptable results in terms of accuracy but requires a time-consuming training and calibration phase both for initialization as well as each time the environment's conditions change.

Time-of-Flight (TOF) based approaches bypass these problems, as they are capable to estimate the position of a mobile target on the fly based on trilateration. They are built on the fact that the distance d between transmitter and receiver is directly proportional to the signal propagation delay t and the radio signals' (nearly) constant propagation velocity given by the speed of light c. Thus, d can be estimated by the simple formula:

$$d = c \cdot t \tag{1}$$

Due to this relation, TOF techniques are more robust against external interferences than RSSI methods [1] [2] and they neither suffer from access points implementing mechanisms for energy-efficiency nor changes in the environment. The exact signal propagation delay, however, is hard to measure with today's standard WIFI-components due to a missing high-precision timer. Furthermore, varying delays in time measurement process caused by hardware, software and multipath

978-1-4799-4043-1/13/$31.00 ©2013 IEEE

effects degrade the ranging accuracy. Apart from these limitations, we believe that TOF-techniques show high potentials for indoor-localization. We therefore present and implement our own TOF-based approach and evaluate its positioning performance in various settings.

The remainder of this paper is structured as follows: In the next Section, we compare different TOF-based approaches in literature in terms of communication flow, hardware components, method of time measurement and positioning accuracy. Section III describes the basic design and conceptual details of our TOF-based approach. Implementation issues will be presented in Section IV. We evaluate our system on different client hardware as well as in different environments and describe our results in Section V. This paper closes with Section VI, where we discuss the obtained results and conclude.

II. Related Work

Different TOF-based approaches can be found in literature. Hybrid methods that combine advantages of RSS and TOF approaches are presented by [3] and [4]. Wong et al. [5] use the angle-of-arrival technique (AOA) with MIMO-compatible access points to determine a mobile device's position, achieving an accuracy of two meters. Niculescu and Nath [6] use this technique in ad-hoc networks.

Li et al. [7] and Yamasaki et al. [8] investigate the time difference of arrival (TDOA) approach in WIFI infrastructures. The latter reach a mean accuracy of 2.4 meters with their system. However, TDOA is hard to realize in commercial WIFI infrastructures, because highly precise time synchronization is required.

Most research works in this field focus on the round-trip-time-of-flight (RTOF) technique, which avoids complex time synchronization between the access points and mobile devices. This technique measures the time it takes for a WIFI-signal to travel from sender to receiver and back, hence a terminal-based relative time measurement is possible and ranging can be performed on the fly. The distance d between a mobile device and the access point in reach can be computed from the round trip time (RTT) by the formula

$$d = \frac{\mathrm{RTT} - t_{PROC\_AP}}{2} c \qquad (2)$$

with t_{PROC_AP} being the processing time of the signal at the access point. Various RTOF-based methods can be found in literature, which differ from each other in terms of communication flow, hardware components, method of time measurement and positioning accuracy. Guenther and Hoene [2] use ICMP-ping requests in commercial WIFI-infrastructures and the standard IEEE 802.11 time synchronizing function (TSF), achieving a mean ranging error of about eight meters. Haustein [9] also uses the TSF, yet in combination with IEEE 802.11 *Null Function Frame - Acknowledgement* (NULL-ACK) sequences for communication. His approach reaches an accuracy of up to 0.9 meters with 5,000 measurements in ad-hoc mode. Both [2] and [9], however, require an additional monitor node to capture the IEEE 802.11 control frames.

Ciurana et al. [10] use a *Data - Acknowledgment* (DATA-ACK) communication based on ICMP-ping requests and measure the time by a special 44 MHz-WLAN clock in order to avoid any software delays. Thus, this approach is not based on pure, commercially available WIFI-components like the works presented before. With this method, the authors reach a mean ranging error of 0.81 meters over all measured distances.

Another software-only method is presented by [1], which uses Intel's *time stamp counter* (TSC) as a high resolution software timer. Outgoing DATA and incoming ACK frames are acknowledged with a timestamp in the WIFI card driver, thereby allowing for the computation of the RTT. 1,000 ICMP-requests are performed at each distance, resulting in a mean absolute error of 1.7 meters for this approach.

Casacuberta and Ramirez [11] present and compare three different methods to obtain the TOF of a ICMP-ping communication by software on a Siemens WIFI access point. The first method uses the standard TSF, the second one is based on the TSC and the third approach calculates the RTT using the TSF which is called directly by the corresponding WIFI card function. The authors evaluate each method by performing 3,000 sequences at each position and reach a mean error of 2.8, 1.5 and 4.4 meters, respectively. This shows that the TSC is able to provide the best ranging results. However, it should be noticed here, that none of the presented works has successfully used the TSC in combination with a NULL - ACK communication. Yet with the latter seeming to be a promising and advantageous combination, we base our implementation of a TOF-based positioning system on this idea.

III. Estimating RTOF with NULL-ACK-Sequences

In order to discuss the advantages and limitations of TOF-based WIFI-positioning, we implement and evaluate our own RTOF approach. We use commercial off-the-shelf WIFI components without any modification in firmware or hardware, thereby rendering our method easily employable in any IEEE 802.11 infrastructure. As mentioned before, by using RTOF technique we are able to start the ranging process on the fly without the need for a complex offline phase or time synchronization between access points and our mobile device. Our approach divides into three main parts:
First, measuring RTT' which contains the real RTT and additional time delays. Second, processing of measured RTT' values and third, distance estimation.

A. Measuring RTT'

1) Communication: In order to obtain a signal's RTT, some kind of communication between mobile device and access point has to take place where a transmitted frame is answered in a deterministic time by the receiver and sent back to the sender. Fortunately, any IEEE 802.11 data frame satisfies this requirement, given that all frames of this type are being acknowledged by the access point by sending out an IEEE 802.11 ACK frame after a clearly defined time interval, the so-called *short interframe space* (SIFS).

NULL frames represent a special type of IEEE 802.11 data frames, because they merely carry a power management bit in their frame control block while the data field is being left empty. This has the advantage that NULL frames are smaller than other data packets and hence, are less likely to collide with other packets on the air interface. Usually they are only sent by a station in order to inform an access point about its particular power status. The `ToDS` bit is always set to zero, indicating that the distribution system is not addressed and hence, will not be stressed by `NULL` function frames at any time. Furthermore, they do not require encryption, which means that their propagation time only depends on the used modulation process and bitrate [9].

Due to the fact that these `NULL` frames belong to the group of data frames they have to be acknowledged after the time of one SIFS by the receiving entity. One handy difference to other types of data frames, however, is that a station sending a `NULL` frame to an access point does not have to be associated with the latter. Thus, the RTT can be measured in all IEEE 802.11 compliant infrastructures on the fly without a time-consuming offline-phase or the mobile device having to associate with a particular access point in reach. In combination, these facts offer great potentials for indoor-positioning systems.

2) Time Measurement: The most delicate part of any TOF-based positioning approach is the process of measuring the time as accurately as possible. E.g., a deviation of the measured signal's traveling time from the real signal propagation time of only one nanosecond causes a ranging error of 0.3 meter, based on the speed of light the radio signals travel with.

In order to obtain the RTT, we perform a relative time measurement on the mobile device by capturing the timestamps of outgoing `NULL` and incoming `ACK` frames. The difference of both timestamps leads to a measured RTT', which contains not only the real RTT, but unfortunately also various delays caused by software and hardware. To avoid any additional processing time caused by the mobile device's operating system, it is essential to capture the timestamps at the closest point to the WIFI hardware still accessible by software. This means that the timestamps have to be set in the WIFI card's driver, more precisely in its interrupt handler functions which are called by interrupt requests of the WIFI card every time a packet is being sent or received.

Modern PCs come with various mechanisms for time measurement [12]. For our purpose, however, only fast-accessible and highly precise software timers readable from the driver's code and offering a nanosecond resolution can be taken into consideration.

The only existing timer which meets all of these requirements is Intel's *time stamp counter* (TSC). Due to its special properties, it is said that "the TSC is, by far, the finest grained, widest, and most convenient timer device to access" [13]. Moreover, it gives the best results in terms of accuracy and costs of all the timer investigated in [12].

The TSC is a 64-bit model-specific register containing a counter which is incremented at each clock tick of the CPU, starting at the last reset. Hence, the TSC's frequency depends directly on the CPU's clock rate, i.e., a 2.0 GHz processor comes with a TSC-resolution of 0.5ns. To obtain the time in seconds out of the TSC-value, the number of clock cycles n has to be divided by the CPU's frequency f in Hz:

$$t = \frac{n}{f} \tag{3}$$

Conveniently, the TSC is directly accessible on the driver level. The instruction *rdtsc* loads the higher 32 bit in the EDX register and the lower 32 bit in the EAX register. Due to the fact that two 32-bit registers cannot be read at once on a 32-bit-system, the lower 32 bit alone can be loaded in the EAX register by the instruction *rdtscl*. Given a 2.0 GHz processor this register overflows every 2.1 seconds, which is sufficient for RTT measurements.

However, there are some aspects negatively influencing the TSC and thereby capable of decreasing its accuracy. State-of-the-art energy management functions like *Power Now* or *Quiet'n'Cool* change the CPU's frequency depending on the actual system workload, which might lead to absolutely impractical time measurements. For this reason, Intel offers a constant TSC in modern processors which always relies on the maximum CPU speed. If no constant TSC is available, it is also possible to deactivate the energy management functions in the system's BIOS to get a constant CPU speed.

Another problem is posed on the TSC by multiprocessor architectures: In this case, it cannot be guaranteed that the counters of the different CPUs are synchronized, which means that they can drift away from each other during one session. This can lead to the situation that a process reads two timestamps from different TSCs and returns a wrong number of clock cycles, leading to a severe problem for TSC-based time measurements. However, once again the constant TSC functionality reduces this problem by keeping the frequency constant in all TSCs. Thus, the EAX registers should contain the same number of clock cycles at any time.

[14] describes further influences on the TSC like the *out-of-order* execution. This is a state-of-the-art mechanism of modern processor architectures in order to use the full capacity of the CPU. However, this leads to the problem that instructions might not always be executed exactly as in the program's order. Hence, the *rdtscl* instruction can be executed at a wrong point of time which leads to an incorrect time measurement. To prevent this problem, a serialization command such as *cpuid* must be set before the *rdtscl* instruction. Thereby, it is guaranteed that all previous commands are executed before the TSC will be read.

Another influence on the TSC's accuracy are varying latencies caused by the data or instruction cache. These effects lead to varying TSC-differences within one program sequence. Those deviations are quite common in modern systems and can only be balanced by mean values derived from a higher amount of TSC measurements.

Additionally, any number of external interrupts, e.g., caused by hardware components might occur which have to be handled by the CPU immediately. This disrupts the normal program flow and leads to a delayed handling of the corresponding program sequence and hence unpredictably distorted TSC timestamps, respectively.

The biggest problem, however, is caused by the system's normal mode of operation itself: All hardware interrupts are handled by the operating system's *interrupt service routine* (ISR), which can itself be disrupted by higher priority interrupts. Hence, the time between the appearance of the interrupt and its handling differs greatly on a case to case basis. Even in real time operating systems, which meet special requirements in order to be able to handle such requests within a guaranteed maximum response time, the ISR shows a time-variance of a

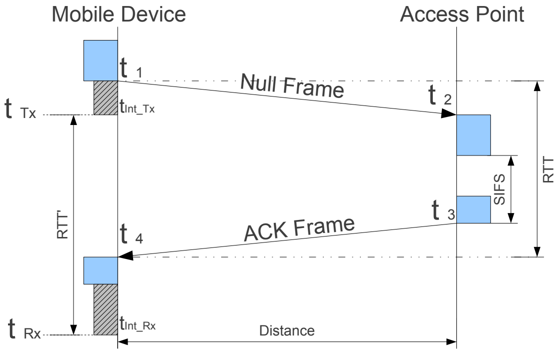


Fig. 1: Illustration of taking a single RTT' measurement utilizing a NULL-ACK sequence between a mobile device and an access point. Notice the difference between RTT' (indicated on the left) and the actual RTT (right), which is mainly caused by software delays.

few microseconds. These fluctuations lead to the fact that TSC-timestamps can always only be set with varying, unpredictable delays adding up to the corresponding actual TOF. Hence, even under constant environmental conditions one always retrieves different values for RTT'. This problem can be observed as the biggest limitation of the presented RTOF-based approach in terms of our system's positioning accuracy. However, as shown in our evaluation a suitable RTT estimation based on a high amount of RTT' values and adequate statistical estimators is still reachable.

The whole sequence of a NULL - ACK communication and the presented time measurement approach is summarized and illustrated in figure 1. The shaded areas represent the different interrupt delays which cause varying timestamps t_{Tx} and t_{Rx}. Notice that apart from the presented limitations and the described problems of the TSC, there can also be additional, non-software delays occurring during the signal transmission itself. This happens, e.g., when radio signals are deflected from their direct path due to reflections on obstacles or walls. The so-called multipath propagation causes longer times for the signal to travel between sender and receiver. However, these delays are in nanosecond range. Thus, the fluctuations relying on the ISR have a considerably higher influence on the RTT' measurement than these additional propagation delays.

Due to these findings, a stable and accurate RTT estimation can only be determined in a further processing phase. In that stage, the RTT will be estimated from a sufficiently large set of RTT' measurements by means of statistical analysis.

B. Processing RTT'

We will now briefly describe the processing of the raw RTT' measurements. The first step in the processing phase is the detection and elimination of all outliers in the set of RTT' measurements with adequate filters. Subsequently, we try to estimate the true RTT by analyzing the remaining set's statistical characteristics.

1) Filtering: [15] summarizes different methods for outlier detections which can be categorized as parametric and non-parametric outlier detection algorithms. The first ones are based on the knowledge about the distribution underlying the collected data. Due to the fact that the distribution of the measured RTT' values cannot be known in advance, however, we have to use a non-parametric method such as the outlier detection by Walsh: Using a predefined level of significance α, we decide whether a certain number of values at the border of a sorted amount of RTT' measurements belong to a group of outliers or not. This test seems promising for our case considering its ability to analyze huge amounts of data without requiring the analyzed data to be normally distributed.

According to [1] this might as well be realized using a bandpass filter letting only values from a certain interval of the original RTT' histogram pass. All values lying outside the the filter's boundaries will simply be ignored. Hence, similar to Walsh's outlier detection, a certain amount of values at the lower and upper end of the histogram will be excluded from further analysis. The remaining question, however, is how to choose the actual values for the filter's lower and upper cut-off frequencies for a specific case. This is to be answered during our evaluation.

2) Statistical estimation: From the set of filtered RTT' values, we now try to estimate the actual roundtrip time RTT by means of statistical analysis. To this end and in accordance with [1] we utilize both the data's arithmetic mean μ and its mode. In addition, we take into consideration the distribution's median as well as the estimator function $(\mu - \sigma/3)$ given that these were successfully employed in [10]. Which of these estimators is able to offer the most stable positioning results will be part of our evaluation in Section V.

C. Distance estimation

As soon as our algorithm has calculated an estimate for the actual RTT, this value has to be transformed into a physical distance in a meter scale. Therefore, in a first step the number of *clock cycles* has to be divided by the CPU's frequency (cf. Eq. 3), in order to calculate the RTT in seconds. Consequently, we can calculate the distance d by using Eq. 2 and considering the delays depicted in Figure 1 as follows:

$$d = \frac{\mathrm{RTT} - t_{PPDU\_Null} - t_{SIFS} - \mathrm{const}}{2} c \qquad (4)$$

The constant *const* serves as a correcting parameter here, which is necessary for compensating the average systematic delays occurring during the measurement process. With the different interrupt requests always being fired after the sending and receiving of a packet, respectively, and a NULL frame being bigger in size than the corresponding ACK, it follows that $const = t_{PPDU\_Ack} - t_{PPDU\_Null}$ is negative if $t_{Int\_Tx} = t_{Int\_Rx}$. However, in practice the two interrupt delays t_{Int_Tx} and t_{Int_Rx} are usually not the same. Furthermore, additional delays might occur in the system, resulting in the fact that *const* has to be larger than simply taking the difference of the two frames' lengths. Unfortunately, this value cannot simply be calculated statically, but has to be estimated by utilizing a high number of RTT measurements. This is realized in accordance to [1] by setting *const* at a known distance between our mobile device and an access point in a way the average distance calculation exactly matches the current distance. This way, the distance estimation function can be calibrated for compensating the system's constant delays. Dynamic delays,

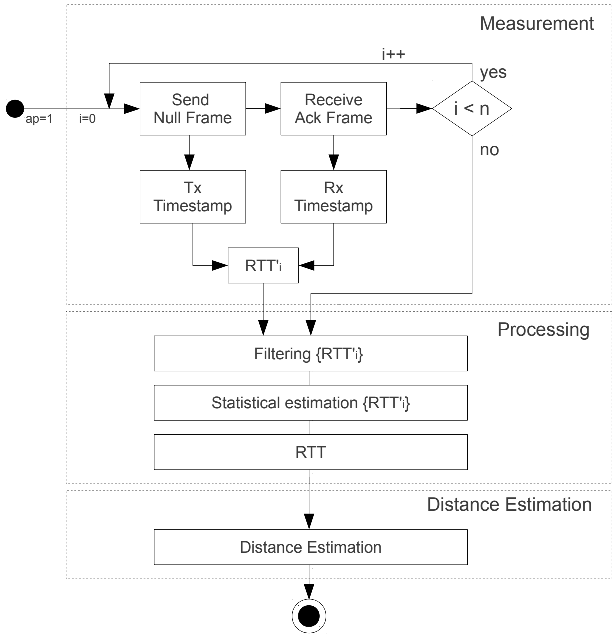


Fig. 2: Activity diagram of ranging process

however, are still able to distort our distance estimations to a certain amount.

Figure 2 shows the complete sequence of steps in the ranging process as an activity diagram. If we are able to retrieve at least three distance estimations to three different access points with the ranging process just described, we can easily determine the position of the mobile device by means of trilateration. However, this extra step is not within the scope of this paper.

IV. Implementation

We have implemented our approach on a Linux Ubuntu 11.04 system with kernel version 3.0.0-19 generic. We use two different WIFI drivers working on two different network interface cards. The first setup is using the madWifi Driver on an IEEE 802.11 b/g PCMCIA-card with Atheros chipset AR5212/5213 provided by Netgear. The second setup is based on the open source rt2800pci driver included in Linux's compat-drivers package working on an onboard IEEE 802.11n PCIe card with RT3090 chip set provided by Ralink. Both drivers support the monitor/injection mode which is a precondition for sending data frames and capturing `ACK` frames all on the same WIFI card.

The particular aspects of our ranging process as described above are technically implemented as follows:

A. Generating `NULL` Frames

The first challenge when implementing our approach is that Linux as an operating system does not provide a function to generate and inject `NULL` Frames on the IEEE 802.11 hardware. However, external libraries like *libpcap* allow to infiltrate IEEE 802.11 packages via the Linux Kernel by user programs. We use *nullinject* from [9] to generate `NULL` frames with the libpcap library and to send them to a specific access point. Additionally, the program offers some useful options, e.g., to define the bit rate, modulation method, or the number of `NULL` frames to be sent. Furthermore, we are able to set a delay between the `NULL-ACK` sequences, which is essential for accurately isolating consecutive `NULL-ACK` pairs. Preliminary tests with different parameters have shown that the best results can be obtained with a 6 MBit/s OFDM modulation and a delay of 0.5 milliseconds between sending `NULL` frames.

B. Generating Timestamps

As mentioned before, the timestamps are set in the interrupt handler function of the WIFI-card's driver with the *rdtscl* command and are passed to the syslogd-process with the *printk* command. We log timestamps in the event of two different hardware interrupts: The one which is triggered when a frame is received (*rx_done*), and the one which is triggered when a frame is sent (*tx_done*). Both are handled in *rt2800pci_interrupt* for rt2800pci and in *ath_intr()* for madWifi. To associate the *rx_done* or *tx_done* interrupt with a particular frame, we extract the frame type, the source address and the destination address from the corresponding frame header in the interrupt context handler. This is done with the *printk* command in the *rx_done* or *tx_done* method when using the rt2800pci driver, and in the *ath_rx_capture* or *ath_tx_capture* method when using the madWifi driver.

Finally, we get a log-file containing the TSC-timestamps of all incoming `ACK` frames and outgoing `NULL` frames from one measurement process. The `NULL-ACK` sequences can be distinguished by their source or destination address, thus, an allocation to the used access point is possible.

C. Processing Log-file

The log-file may contain defective `NULL-ACK` sequences due to errors or delays in data transmission or network noise. Only those sequences will be processed which have correct properties, i.e., the source address of an outgoing `NULL` frame must be the same as the destination address of the ensuing incoming `ACK` frame. Due to the fact that a `NULL-ACK` sequence must not be interrupted by any other packets when complying with IEEE 802.11, no frames should be received between a `NULL` and an `ACK` frame. Otherwise the sequence is invalid. After eliminating corrupt `NULL-ACK` sequences, RTT' is calculated from each remaining sequence by taking the difference between the *rx_done* and the corresponding *tx_done* timestamps. In case the difference is negative because of a overflow in the EAX register the corresponding RTT' is invalid and will be eliminated. Finally, we get a list of valid RTT' measurements which are further processed as described in section III-B.

V. Evaluation

We have evaluated our RTOF approach on two Laptops which differ in terms of WIFI hardware and software:

1) The first is a Samsung X65 with 2.2 GHz Intel Core2 Centrino Duo processor containing the described IEEE 802.11 b/g PCMCIA-card from Netgear with the madWifi driver using the net80211 subsystem.

2) The second is a HP ProBook 4520s with Intel's i3 2.4 GHz processor containing the described IEEE 802.11n PCIe card from Ralink with the rt2800pci driver using the mac80211 subsystem.

Both devices work on the same operating system which is Linux Ubuntu 11.04 with kernel version 3.0.0-19 generic. Using two different hardware setups allows us to at least get a basic feeling for our system's dependency on particular hardware and drivers, and their influence on ranging accuracy.

A. Analyzing RTT' measurements

In a first step, we analyze a large number of RTT' measurements to evaluate the occurrent time fluctuations in the measuring process. For this purpose we send 100,000 NULL frames with each laptop to an access point which is arranged in a shared and obstacle-free fresnel zone to avoid multipath effects. The actual distance between access point and laptop is exactly one meter. The test is performed in an interference-free environment where no WIFI-signals are received except the ones from the access point we communicate with. With this setup we minimize influences on the signal propagation itself, thus we can investigate the time fluctuations caused by our timestamping approach. Our results as shown in figure 3 indicate that the RTT' measurements differ extremely within the same settings.

On the HP ProBook we get 99,905 valid NULL-ACK sequences with a maximum ranging difference of 595,968 clock cycles while on the Samsung X65 we obtain 93,183 valid sequences and a maximum difference of even 2,125,354 clock cycles. This demonstrates that the RTT' has to be treated like a random number at this point, and a single measurement is not sufficient to determine the distance. The high deviation can be seen in figure 3a which shows all RTT' values in ascending order.

It is noticeable that the boundary values on both sides differ extremely from the ones in the middle. This can also be observed in the histogram 3b where we get a high frequency for the mean and only small frequencies for other measurements. This effect is mainly caused by varying interrupt latencies. In most cases the ISR needs about the same time to respond to a *tx_done* as to a *rx_done* interrupt, thus the RTT' measurements are more stable. However, in rare cases the response time to both interrupts can be completely different. If the *tx_done* response takes longer while the *rx_done* interrupt is quick, we get lower measurements, and the other way round we observe higher RTT' values. These abnormal measurements have to be filtered in the processing phase because they lead to an inaccurate RTT distance estimation.

Besides a higher standard deviation of 21,390 clock cycles, we also observe extreme fluctuations for the Samsung X65 on both sides which cannot only be caused by different interrupt latencies, given that the difference between the lowest and the highest value is too large. Thus, there also have to be other factors which influence the measurement procedure, resulting in the ranging process on the Samsung laptop being more error-prone than that on the HP laptop. This observation is supported by a lower number of valid NULL-ACK sequences on the Samsung laptop and is suspected to be caused by the corresponding hardware, because the test setup and parameters are the same for both devices.

Generally, the differences with regard to the fluctuations and errors of the RTT' measurements on the two devices indicate that our approach is plattform-dependent to some extent.

B. Analyzing Filters and estimators

In the next step we analyze different filters and statistical estimators to obtain a stable RTT from the measured RTT' values. In order to find the best filter and statistical estimator, we send again a high amount of n NULL-ACK sequences on both systems under the same circumstances as before.

We send $n =$10,000 NULL frames since preliminary tests have shown that this is a good compromise between stable ranging results and the number of sent packets. In comparison to other approaches in literature like [11] this is a relatively high value for n. The reason is that we work on an off-the-shelf laptop which is not optimized for WIFI frame injections. Depending on the workload of the system we need up to 10 seconds to exchange this amount of NULL-ACK sequences.

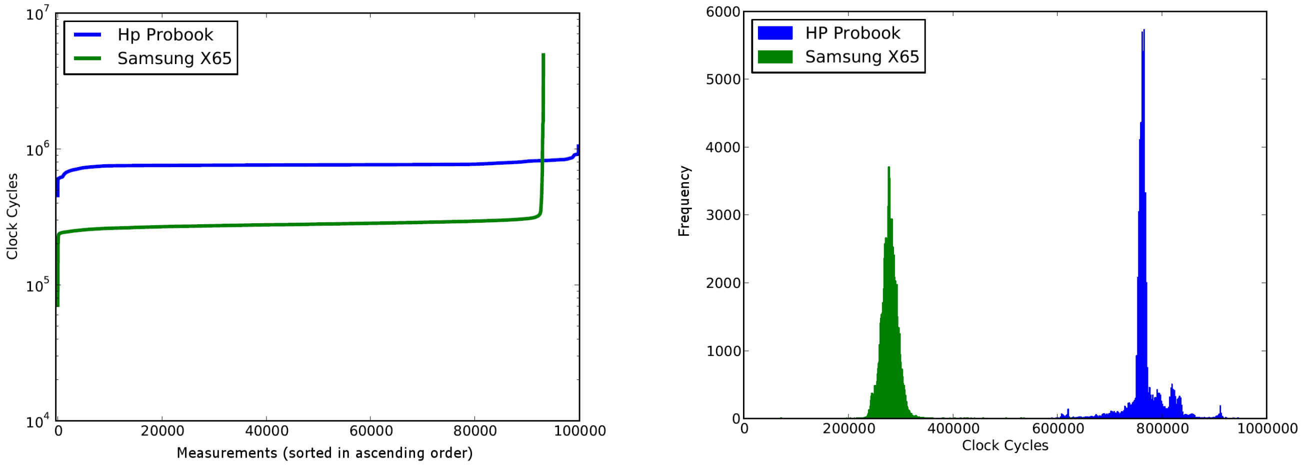
After applying the different filters to the 10,000 RTT' measurements, the RTT is calculated with the help of the already mentioned statistical estimators: average, median, mode and $(\mu - \sigma/3)$. The results can be summed up in a matrix consisting of the used filters and estimators which allows us to determine the best filter parameters using a certain estimator. We repeat this test m times to determine mean, standard deviation and mean absolute error for each entry of the matrix. After several tests with different values of m, we found $m = 30$ to be an optimal compromise between the number of iterations and a low standard deviation.

The results after sending 30 times 10,000 NULL-ACK sequences are shown in table I for the HP Probook and in table II for the Samsung X65. Each field of the table contains the standard deviation in clock cycles over 30 RTTs computed with the corresponding filter and statistical estimator. We find that a specific band pass-filter combined with the mean results in the lowest standard deviation on both systems. However, the Samsung X65 again shows much higher values in this field. This indicates a lower accuracy for the ranging process which confirms the assumption made before.

The optimal parameters of the band pass-filter have been obtained experimentally by trying all band pass-filter combinations and comparing the resulting standard deviations. The smallest variance on both laptops can be achieved if 10 % of the lowest and 35 % of the highest RTT' values are eliminated. Thus, we consider 55% of all valid measurements in the ranging process. With these findings we perform several distance estimations in different environments on both systems and evaluate the results.

C. Distance estimation

We measure the RTT between each test device and one access point at the following distances: 1, 5, 10 and 20 meters. To analyze the mean ranging results and the variance we repeat this process 30 times for each distance. The one-meter distance is used for system-calibration. For this purpose we perform the



(a) RTT' measurement with 100,000 `NULL-ACK` sequences sorted in ascending order (HP and Samsung)

(b) Histogram of RTT' frequencies with 100,000 `NULL-ACK` sequences (HP and Samsung)

Fig. 3: Analyzing raw data on both test devices: 100,000 unfiltered RTT' measurements at a real distance of one meter

TABLE I: Standard deviation of 30 RTT values computed with different filters and estimators on HP ProBook

Filter	Median	Mode	μ	$(\mu - \sigma/3)$
None	24602	2458	747	800
Walsh	627	1646	542	560
Low-Pass	5542	6258	5234	5303
Band-Pass (0.1,0.35)	510	2463	157	173

TABLE II: Standard deviation of 30 RTT values computed with different filters and estimators on Samsung X65

Filter	Median	Mode	μ	$(\mu - \sigma/3)$
None	715	1014	1481	2771
Walsh	892	755	727	767
Low-Pass	2058	1814	1279	1047
Band-Pass (0.1,0.35)	815	1093	693	695

TABLE III: Ranging results of 30 trials at each distance with HP Probook

	1 m	5 m	10 m	15 m	20 m
Ø	1.00	1.54	7.05	15.17	18.72
Max.	24.26	20.17	21.99	34.32	46.23
Min.	-25.90	-19.59	-9.76	-6.62	-7.67
ϱ	13.62	7.81	6.27	9.82	12.91
ØError	0.00	-3.46	-2.95	0.17	-1.28

TABLE IV: Ranging results of 30 trials at each distance with Samsung X65

	1 m	5 m	10 m	15 m	20 m
Ø	1.00	-27.20	11.24	77.54	32.16
Max.	46.84	11.02	73.42	116.93	82.33
Min.	-28.27	-59.77	-35.63	48.62	-14.01
ϱ	10.63	13.54	24.98	14.61	23.55
ØError	0.00	-32.29	1.24	62.54	12.16

ranging process m times and compute the mean distance out of m estimated RTTs. The mean distance is then calibrated to be exactly one meter by setting the $const$ parameter in formula 4 accordingly. The calibration is necessary due to the fact that the system contains constant software and hardware delays which can only be determined empirically. Once the $const$ parameter is set it is used to adjust all other ranging results. It has to be mentioned that because of the $const$ value also negative RTT estimations are possible. We have performed the described distance estimation in two environments, namely an *optimal environment* and an *office environment*.

1) Optimal environment: In this case the experiment is performed in the same interference-free environment as described in section V-A. The system is set up in an obstacle-free fresnel zone with an access point which is the only one in reach, thus multipath effects are minimized. The results of the distance estimation are shown in table III for the HP laptop and in table IV for the Samsung laptop.

Considering the mean absolute deviation over all estimated distances on the HP laptop (i.e., $\varrho = 10.09$ meters), it can be seen that a single distance estimation is not stable enough to accurately determine the actual distance. However, in our case we perform 30 ranging processes and take the mean of all estimated distances at each actual distance. These mean estimated distances show a close correlation to the real distances. We are able to achieve a mean absolute error of 1.33 meters over all distances (with the first being left out because it is used for calibration). This is a reasonable result for a ranging process within buildings.

The correlation between the estimated distances and the corresponding real distance is shown in figure 4. It is well observable that the mean of all estimated distances on HP ProBook increases with ascending distance to the access point.

Considering the results on the Samsung laptop, we achieve a mean absolute deviation over all estimated distances of $\varrho = 17.46$ meters and a mean absolute error of 27.06 meters.

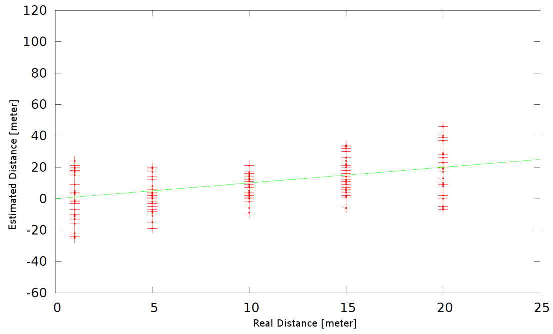


Fig. 4: Correlation between estimated and real distances on HP ProBook

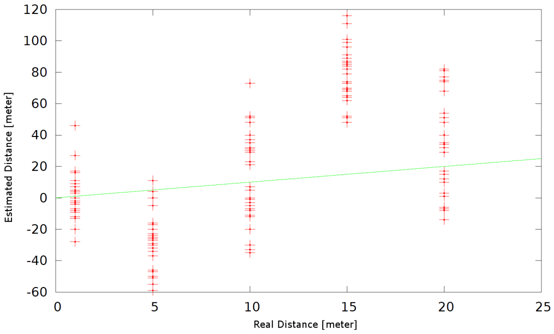


Fig. 5: Correlation between estimated and real distances on Samsung X65

Unfortunately, this is very high for a valid and adequate distance estimation. It is also shown in figure 5 that the mean estimated distances are not usable, and a clear correlation between estimated and real distances is missing.

Comparing both systems we see that only on the HP laptop we achieve suitable ranging results. With regard to the findings made in previous experiments this is an expected observation. Hence, the different hardware components used on the test devices has an influence on our RTOF approach. It has to be investigated in future research which of the used components decreases the ranging accuracy.

2) Office environment: We have repeated the previous experiment within an office environment where more WIFI signals from different access points are received, and the line of sight between the test devices and the access point is sometimes interrupted by moving persons. In this setup we are not able to achieve any suitable ranging results on the Samsung laptop. The mean absolute error is at 275 meters and no correlation between estimated and real distances can be found. A trilateration relying on these results is not feasible, rendering indoor-positioning impossible under these circumstances.

However, we still achieve acceptable ranging results on the HP laptop. The mean absolute deviation of all estimated distances is at $\varrho = 7.96$, which is even lower than before. Besides the fact that the mean absolute error is 4.24 meters higher than in the previous setup, a correlation between estimated and real distance can still be observed. This is illustrated in figure 6.

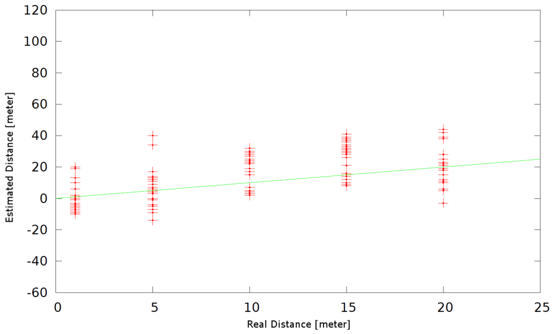


Fig. 6: Correlation between estimated and real distances on HP ProBook within an office environment

It is shown that also in a realistic environment the HP achieves feasible ranging results for the mean while our approach is not suitable on Samsung.

VI. Conclusion

In this paper we have presented the potentials of TOF-based WIFI-Positioning systems. Such systems have the ability to determine a mobile devices position on-the-fly without requiring a complex offline-phase. Furthermore, they are more robust against external interferences than RSSI based approaches due to the relation between distance and the propagation delay. If a `NULL-ACK` communication is used no association with the access point is required.

We have shown that the accuracy of such systems is limited due to missing precise and adequate hardware timers. We have implemented our own RTOF approach using TSC and an exchange of a large number of `NULL-ACK` sequences to overcome this limitation. Unfortunately, the accuracy of our approach suffers from time fluctuations caused by varying delays of the TSC and the ISR. To reduce these fluctuations, we have investigated different filters and statistical estimators.

Our evaluation shows that a band-pass filter and the mean of a large number of RTT' measurements return the best ranging results while the accuracy depends on both the used hardware and the given environment. In the best case we achieve a mean absolute ranging error of 1.33 meters for all measured distances. Compared to other RTOF-based approaches found in literature, this is a good result.

However, due to the fact that the mean deviation of all measured distances is never less than 6.27 meters, the presented ranging approach does not reach the accuracy required in indoor environments. We therefore conclude that TOF-based WIFI-positioning is not yet feasible on off-the-shelf devices. Further development with regard to more precise WIFI-clocks is necessary. A great opportunity for a more accurate TOF-based WIFI-positioning system is the new IEEE 802.11v standard which includes new time measurement functions. Hence, further research in terms of ranging accuracy relying on this new WIFI standard seems promising.

References

[1] M. Ciurana, D. López, and F. Barceló-Arroyo, "Softoa: Software ranging for toa-based positioning of wlan terminals," *Location and Context Awareness*, pp. 207–221, 2009.

[2] A. Günther and C. Hoene, "Measuring round trip times to determine the distance between wlan nodes," *NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*, pp. 303–319, 2005.

[3] D. Giustiniano and S. Mangold, "Caesar: carrier sense-based ranging in off-the-shelf 802.11 wireless lan," in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*. ACM, 2011, p. 10.

[4] M. Abusubaih, B. Rathke, and A. Wolisz, "A dual distance measurement scheme for indoor ieee 802.11 wireless local area networks," in *Mobile Wireless Communications Networks, 2007 9th IFIP International Conference on*. IEEE, 2007, pp. 121–125.

[5] C. Wong, R. Klukas, and G. Messier, "Using wlan infrastructure for angle-of-arrival indoor user location," in *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th*. IEEE, 2008, pp. 1–5.

[6] D. Niculescu and B. Nath, "Ad hoc positioning system (aps) using aoa," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3. IEEE, 2003, pp. 1734–1743.

[7] X. Li, K. Pahlavan, M. Latva-aho, and M. Ylianttila, "Comparison of indoor geolocation methods in dsss and ofdm wireless lan systems," in *Vehicular Technology Conference, 2000. IEEE VTS-Fall VTC 2000. 52nd*, vol. 6. IEEE, 2000, pp. 3015–3020.

[8] R. Yamasaki, A. Ogino, T. Tamaki, T. Uta, N. Matsuzawa, and T. Kato, "Tdoa location system for ieee 802.11 b wlan," in *Wireless Communications and Networking Conference, 2005 IEEE*, vol. 4. IEEE, 2005, pp. 2338–2343.

[9] M. Haustein and M. Werner, "Lokalisierung durch die messung von signallaufzeiten mittels handlesblicher wireless lan adapter," in *Ortsbezogene Anwendungen und Dienste*. GI/KuVS, 2011, pp. 101–113.

[10] M. Ciurana, F. Barcelo-Arroyo, and F. Izquierdo, "A ranging system with ieee 802.11 data frames," in *Radio and Wireless Symposium, 2007 IEEE*. IEEE, 2007, pp. 133–136.

[11] I. Casacuberta and A. Ramirez, "Time-of-flight positioning using the existing wireless local area network infrastructure," in *2012 International Conference on Indoor Positioning and Indoor Navigation*, 2012.

[12] 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.

[13] Vmware, "Timekeeping in vmware virtual machines," http://www.vmware.com/files/pdf/Timekeeping-In-VirtualMachines.pdf, 2010.

[14] I. Coorporation, "Using the rdtsc instruction for performance monitoring," tech. rep., Intel Coorporation, Tech. Rep., 1997.

[15] V. Chandola, A. Banerjee, and V. Kumar, "Outlier detection: A survey," *ACM Computing Surveys, to appear*, 2007.