Timing Guide for Tobii Eye Trackers and Eye Tracking Software
Timing Guide for Tobii Eye Trackers and Eye Tracking Software

February 23, 2010
Tobii Technology AB

The Tobii Eye Tracking systems consist of hardware and software for gaze point estimation and communication with various applications. The gaze data calculations either occur in a processor located in a dedicated computer embedded in the eye tracker (Tobii T & X Series Eye Trackers) or in an external computer connected to the eye tracker (Tobii -50 Series Eye Trackers). The data is then streamed to an interaction or analysis software for further processing, for example by using Tobii Studio or by using custom interaction and analysis applications and the Tobii Software Development Kit (Tobii SDK). The objective of this document is to provide a description of the different components and processes that are responsible for the timing accuracy of Tobii Eye trackers and the Tobii Studio Analysis Software.
# Contents

1 Brief introduction to computer timing in Windows ................................................................. 3  
1.1 Windows time acquisition APIs .................................................................................. 3  

2 Tobii Eye Trackers ............................................................................................................. 4  
2.1 Eye tracker latency ........................................................................................................ 5  
2.2 Gaze data sampling rate .............................................................................................. 6  
2.3 Gaze data timestamp .................................................................................................... 7  
2.4 Synchronization between the TET server and an external system .................................. 8  
2.5 Summary ...................................................................................................................... 9  
2.6 Recommendations ...................................................................................................... 10  

3 Tobii Studio Gaze Analysis Software ................................................................................. 11  
3.1 Tobii Studio recording data sources ............................................................................. 11  
3.2 Stimulus display and time stamping of stimuli display related events ............................... 12  
3.2.1 Tobii Studio stimulus display offset test results ....................................................... 13  
3.3 User Camera Video, External video, Scene camera, Sound & Screen recording ..................... 15  
3.3.1 General information on digital video media files ...................................................... 15  
3.3.2 Tobii Studio video and sound capture ...................................................................... 16  
3.3.3 Tobii Studio video replay ....................................................................................... 17  
3.4 Summary ..................................................................................................................... 19  
3.5 Recommendations ...................................................................................................... 20  

# Appendix

Appendix I - How to test the synchronization between video/image recording and actual screen display time using a high speed camera
1 Brief introduction to computer timing in Windows

Every computer possesses dedicated control and timing circuits that ensure that the different operations/commands are being deployed at specific times. A pulse generator of a type determined by computer design provides the main timing signals for any given type of computer. These pulse generators are commonly termed master or hardware clocks. They usually operate at a frequency or pulse repetition rate determined by the maximum rate at which the computer handles data (CPU speed). The master clock is the key to the timing circuits in the computer. It will set in motion the computer’s main timing circuits. From the main timing circuits, other circuits responsible for various other areas (arithmetic, memory, and I/O) can be enabled or disabled. The clock will produce electrical pulses with extreme regularity. Modern computers can achieve quite precise timing when performing different tasks, for example, on a 3GHz processor, this timer can measure events with an accuracy of +/- 0.333 nanoseconds. However current operating systems are designed to work in a multi-task environment where different processes run simultaneously. As a consequence the operating system shares the processor timing and resources between different processes (for example, allocation of memory, retrieving data from external sources and automatic memory dump processes). All these operations do not necessarily perform in a predictable time lapse, making it hard to produce or register an event at any precise instant (e.g. an input from an external device).

1.1 Windows time acquisition APIs

There are several application interfaces (APIs) in Windows that deal with time acquisition; however in practice there are only two underlying methods available for extracting higher-frequency timing data. The first method relies on the processor timestamp counter located on the computer’s motherboard. The timestamp counter is a 64-bit register present on all x86 processors (since the launch of the Intel Pentium processor) and registers the number of clock ticks the processor has experienced since it was last reset. The primary time functions (the RDTSC API) used to access the information from this counter are of high accuracy and low precision, i.e., the average value of a set of timestamps will be relatively accurate, but each timestamp per se can lack accuracy, examples of functions that fall into this first category are: GetTickCount, TimeGetTime and GetSystemTimeAsFileTime. Even though these functions are able to return values with nanoseconds fraction precision and have a small drift against real time, they are inadequate to perform a comparison between the occurrences of two events: 1. they are strongly affected by changes in the computer processor frequency, current laptops and high end computers possess power management technologies that change the computer’s CPU frequency, the number of clock ticks that occurred during this time may be accurate, but the elapsed time will be an unknown; 2. it assumes that the program thread is run on a single processor, in multiprocessors or dual-core computer systems a thread can be run by a different processor each time it is executed, the cycle counters in different processors are not synchronized (this is enhanced by the power management settings that can change the frequency of one processor relative independently from the other), resulting in timing values with large deltas, negative deltas or halted timing; 3. It locks the timing information to the processor timestamp counter (frequency), as newer motherboards are now including dedicated timing devices that provide higher time resolution.

The second method relies on the use of the Windows API functions QueryPerformanceCounter and QueryPerformanceFrequency. These APIs may make use of the CPU timestamp counter, but can also access other timing devices on the motherboard or other system services that provide high-quality high-resolution timing information. The benefit of using these functions is the microsecond precision and the possibility to adjust the timing thread to be run on a single processor in multi and dual-core environment. However these functions are also affected by changes in processor frequency if the system is set to hibernation or sleep mode, and it has a known tendency to occasionally make a counter jump on some buggy hardware platforms. This counter also suffers from a high hardware dependent drift against both the system clock (counter) and the real
time (i.e. the "current time" or clock wall time, values for the year, month, date, hours, minutes, and seconds) clock. No matter what source of time is chosen to timestamp events in the computer, there will be two synchronization issues; first the one between software and real time and second the one between applications running on different hosts.

2 Tobii Eye Trackers

Tobii Eye Trackers are a version of the traditional Pupil Centre Corneal Reflection (PCCR) remote eye tracking technology. Near infrared illumination is used to create reflection patterns on the cornea and pupils of the eye of a user. An image sensor then captures images of the eyes and respective reflection patterns, and sends them to the eye tracker firmware - TET Server (or Tobii Eye Tracking server). The position of the eye and the point of gaze are then derived with high accuracy from image processing algorithms and a mathematical model of the eye. The TET Server also time stamps each image when it arrives to the server using a time function dll – TTime.dll. This function extracts timing information from a high resolution counter located on the computer or eye tracker processor. The TET Server then recalculates this timestamp in order to show the time when the image was captured by the sensor.

In the Tobii -50 Series Eye Trackers the TET Server is installed and run from an external computer that communicates with the eye tracker through USB and Firewire connections. The images from the sensor are sent through the firewire connection to the computer, while the TET Server controls the eye tracker systems (IR illuminators) through the USB cable. The images are processed into gaze data in the TET Server on the external computer. The TET Server is then accessed by other applications (e.g. Tobii Studio Software) that extract and use the raw data to analyze eye gaze points and/or produce eye tracking visualizations. These applications can either run on the same or a different computer than the TET Server. The method chosen for producing the timestamps for the gaze data is to use the processor high resolution counter located on the processor of the computer where the TET Server is installed, together with QueryPerformanceCounter and QueryPerformanceFrequency functions.

Figure 1 Tobii -50Series Eye Tracker gaze sample processing. The gaze data is processed externally from the eye tracker.

In the Tobii T/X Series Eye Trackers the TET Server runs on a computer integrated in the eye tracker hardware. As a result the images from the sensor are processed in the eye tracker and the resulting gaze data is then sent
to the network via a TCP/IP Lan connection. Comparative to the Tobii -50 Series Eye Trackers, the T/X Series allows a simpler and more efficient integration between the eye tracker and external applications such as the Tobii Studio Analysis Software. Since the TET server runs on a dedicated separate computer it reduces the impact of high processor load on image processing and gaze data computation.

![Diagram](image)

**Figure 2** Tobii T/X Series Eye Tracker gaze sample processing. The gaze data is processed in a processor located in the eye tracker.

In real-time applications latency is one of the key aspects, as the interaction between the application and the eye tracker is dependent on how fast the gaze data is delivered to the application. For example, in gaze-contingency research paradigms and eye based interaction applications, the test procedure often requires that the test participant gazes at a certain area of the screen for a specific time, in order for it to trigger a response. However when it comes to post-recording analysis the eye tracker sensor data rate, the time stamping process of the gaze data and its synchronization with the application used to present and record the stimulus, are crucial to ensure good timing information in the data used in the analysis (for example slide show and web studies).

### 2.1 Eye tracker latency

The latency of an eye tracker corresponds to the interval measured from the time when an image is captured by the sensor to when valid gaze data is outputted by the the TET Server to the network or an eye tracking application. This delay is the sum of the time taken for camera exposure, transfer, calculation and delays in the system. For the Tobii X50 Series the typical latency is between 25-35 ms, when used in normal conditions and limited head-motion on a 2.4 Ghz Pentium 4 Computer. The Tobii TX Series shows a typical latency between 27- 33 ms.
In all Tobii Eye Trackers the timestamps of the gaze data are calculated based on the time of image exposure at the sensor (i.e. when the sensor collects an image of the eye) and thus are independent of the latency. However the response of real-time applications, such as gaze contingency, will inherently possess a maximum of 33 or 35 ms (depending on the eye tracker series) delay response due to the eye tracker latency.

2.2 Gaze data sampling rate

The data rate of an eye tracker denotes the number of gaze data points collected by the system per second. All new Tobii system sensors run at a constant data rate of 60 or 120 Hz. The Tobii T60 an X60 Eye Trackers data sampling rate show a standard deviation of the mean of 0.002 Hz, while the Tobii T120 & X120 Eye Trackers have a standard deviation of 0.003 Hz. The nominal frame rates are 60.04 Hz and 120.08 Hz for long term frequency measurement. For any measurement device there is always a theoretical limit in the measurement accuracy and in several cases it has been seen that the frame rate variation is close to this limit thus indicating that the above numbers may depend on the amount of time used for the measurement itself. Current tests show that in a short timescale (1-5 seconds) the data rate frequency for the Tobii T60/X60/T60XL Eye Trackers vary between 59.7-60.15 Hz while the T120/X120 Eye Trackers 119.5-121.3 Hz, but these variations fall within the uncertainty in the timestamp itself and average out in an longer timescale to the nominal data rates specified mentioned average.

For older Tobii systems, such as Tobii X-50 and Tobii 1750 the frame rates are about 50 Hz but may drop over time. Such drops are caused by specific situations:

- when the subject moves the head rapidly, so the system has difficulty finding the subject’s eyes.
- when the subject turns away completely from the system so the eyes are lost.
- when the system has difficulties finding the subject’s eyes for other reasons (i.e. under conditions when the track status meter is flickering or showing a red light).
when other applications are running on the same computer as the eye tracker server and consume a lot of resources. However, as soon as stable tracking is once again resumed, the frame rate goes up to full speed again.

For most analysis and interaction applications, the varying data rate will not be an issue (it occurs mostly in conditions where the subject is not looking properly at the screen anyway). However, if timing is absolutely critical, it is recommended to compare the timestamp of the recorded data to verify a constant frame rate. Since other applications running on the eye tracking computer may cause highly varying frame rates for older eye trackers, a double computer configuration is recommended for time critical studies, in particular if these are combined with other resource-demanding software.

2.3 Gaze data timestamp

The position of the eye is estimated based on an image taken by the sensor. The sensor is set to a fixed exposure time and the timestamp for each gaze data is set to indicate the middle of exposure time. This mid exposure time is the best theoretical estimation of when the image was taken. The TET Server calculates the Timestamp of each gaze point by taking the time of image arrival at the server (in Server Time) and subtracting it to a known latency value that translates the time it takes for the sensor to capture the image (mid exposure value) and send it to the server. This latency is calculated according to the specifications of the sensor supplier (latency of image delivery algorithm).

![Tobii Eye Tracker gaze data time stamping process.](image)

The Gaze data timestamp accuracy is +/- 3 ms in the Tobii -50 Series Eye Trackers (under normal computer load i.e. when the system is working at full speed) while for the Tobii T/X Series Eye Trackers the uncertainty in the timestamp has a nominal value of 1 ms. Odd timestamps (meaning more than 1 ms away from correct
nominal value) can occur but they are related to sequences of bad data when the eye-tracker has lost the position of the eyes and enters in search mode, for example:

- when the subject moves the head rapidly, so the system has difficulty finding the subject’s eyes.
- when the subject turns away completely from the system so the eyes are lost.
- when the system has difficulties finding the subject’s eyes for other reasons.

Recent tests show that as long the eyes are detected by the sensor the timestamps retain a 1 ms variation in uncertainty and no odd timestamps are generated.

The TET Server provides validity codes for each eye with every gaze data point. The validity code is a measure of the system’s certainty that it has recorded the correct data. The validity code ranges from 0 to 4, with the following interpretations for each value:

- 0 The system is certain that it has recorded all relevant data for the particular eye, and that the data recorded belongs to the particular eye (no risk of confusing left eye with right eye by the system).
- 1 The system has only recorded one eye, and has made some assumptions and estimations regarding if the recorded eye is left or right. However, it is still highly probable that the estimations done are correct. The validity code on the other eye is in this case always set to 3.
- 2 The system has only recorded one eye, and has no way of determining if this is the left or the right eye.
- 3 The system is confident that the actual gaze data is actually incorrect or corrupted. The other eye will always have validity code 1.
- 4 The actual gaze data is missing or definitely incorrect. A couple of gaze data with validity code 4 on both eyes, followed by a number of gaze data with validity code 0 on both eyes, are usually a sure sign of a blink.

During search mode each sample that does not contain information on eye position is marked as a non valid data point. It is recommended that the validity codes are always used for data filtering, to remove data points which are obviously incorrect. For most studies, we recommend removing all data points with a validity code of 2 or higher. Tobii Studio fixation filters use validity codes associated to each gaze data sample to filter out invalid data automatically (validity code = 2 or higher).

2.4 Synchronization between the TET server and an external system (e.g. Tobii Studio running on a separate computer)

At startup Tobii Studio sends several commands to the TET Server, through the computer network formed by the Eye Tracker and the Tobii Studio computer, requesting time information. During this phase the response time from the TET Server may vary depending on the bandwidth and traffic on the network, and as a consequence the synchronization between Server Time and Client Time will be vulnerable to network related time offsets. To minimize these offsets the TTime function takes two readings of the client clock when it sends the time request command to the TET Server: t0 – the time when the command is sent to the network; and t1 – the time when the server response arrives to the computer. The program then assumes that the time for the command to arrive at the TET Server is the same as the time it takes for the TET Server’s reply to arrive to the Tobii Studio Computer. In other words the Server Time sent by the TET Server will be equivalent to the total time it took for the command to be sent and the response to arrive to the Tobii Studio computer divided by two: \( t_1 - \left[\frac{t_1 - t_0}{2}\right] = TTime\) timestamp. In order to further increase the accuracy of the synchronization, and since at startup Tobii Studio sends several Server Time request commands, the time reading that has a lower t1-t0 score is then chosen as a sync reference for the two clocks.
Figure 5 Data timestamp synchronization process between a Tobii TX Series Eye Tracker and .Tobii Studio.

Although the two hardware clocks are synchronized at start-up, they may run at slightly different rates and may gain or lose time relative to each other while the eye tracker and the computer are running. In actual operation, most hardware clocks gain or lose time at a rate of about 1 to 15 seconds per day, with 5 or 6 seconds per day being typical. In order to avoid potential synchronization drifts the TTime function requests time information from the TET Server hardware clock regularly (every 2 minutes) and updates the time values for the timestamps accordingly.

However, if the TET Server and the application are running on the same computer in Tobii -50 Series setups, no synchronization between them is required, as the timing information is obtained from the same counter located in the computer processor.

2.5 Summary

The accuracy of the gaze data timestamps is dependent on 3 main features of the eye trackers, the data sampling rate, the time stamping process of each gaze data sample and the synchronization between the eye tracker and the computer clock counters, when an application running on an external computer interacts with the eye tracker. Table 1 summarizes the different timing values for each one of these processes.

<table>
<thead>
<tr>
<th>Eye tracker</th>
<th>Data rate (Hz)</th>
<th>Gaze data timestamp (ms)</th>
<th>TET Server synch (ms)</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tobii TX Series</td>
<td>60±0.002 (16.67 ms)</td>
<td>± 1</td>
<td>&lt; 1</td>
<td>30±3</td>
</tr>
<tr>
<td></td>
<td>120±0.003 (8.33 ms)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tobii -50 Series</td>
<td>50 (20 ms)</td>
<td>± 3*</td>
<td>&lt; 1**</td>
<td>30±5</td>
</tr>
</tbody>
</table>

* = ½ a “network ping” time (http://www.wikihow.com/Ping-an-IP-Address)  ** under normal computer load

The data rate determines how frequent the eye position is sampled, thus it will influence how fast a change in eye position can be measured by the eye tracker. For example, if we want to measure how fast the eye reacts...
to a rapid change in the position of a stimulus, we need to take in consideration how quick the eye tracker is able to sample the next eye position, e.g. if the eyes changes the position immediately after a sample is collected by the eye tracker, then it will take approximately 16 or 8 ms for the eye tracker to register the change (next sample point), depending whether we are using a 60 or 120Hz data sampling rate.

The Tobii Eye Trackers gaze data timestamps reflect the time at which the eye was exposed to the sensor. The accuracy of the gaze data timestamp together with the synchronization process between the eye trackers and the software used for analysis are responsible for the general accuracy of the gaze data in the software gaze data recording. In the TX Series Eye Trackers these calculations are performed in a dedicated computer (processor) located in the eye tracker hardware, while in the -50 Series Eye Trackers they are performed in an external computer.

For synchronization with other applications and systems, the eye trackers use a Tobii proprietary communication protocol implemented that runs as a background thread taking care of the offset error and compensates for time drift between hosts – the TTime.dll. The synchronization may be shared among all processes and threads on a host (computer where the TET Server is installed) and several clients (software or computer that is interacting with the TET Server) that are connected to each other.

In applications that require a real time response, for example, in gaze-contingency research paradigms and eye based interaction applications, latency is one of the key aspects, as the interaction between the application and the eye tracker is dependent on how fast the gaze data is delivered to the application. Latency is defined as the time taken from when the actual eye position is recorded until data reaches the application. Thus the latency of gaze data is the sum of the time taken for camera exposure, transfer, calculation and delays in the system.

**2.6 Recommendations**

Below you can find a list of recommendations to achieve good timing with your eye tracker:

- if a resource-intensive application is used in parallel with a Tobii -50 Series Eye Tracker, we recommend using a double computer configuration. For example studies when using high frame-rate video captures, conducting web studies or if studies demand high timing accuracy. One computer will be running the TET server firmware and calculating the gaze data, while the second computer will be running the application.
3 Tobii Studio Gaze Analysis Software

A Tobii Studio recording is composed of different data and media elements. Each one of these elements contains temporal information (timestamps) that is initially derived from different hardware clocks (i.e. the eye tracker processor, the computer running Tobii Studio or an external device). In order for Tobii Studio to record and replay these elements in the correct sequence, it uses a set of time synchronization routines (TTime.dll) that are deployed during Tobii Studio start up and at runtime (i.e. during a recording and playback) as a background thread (read the previous chapter for more information). These routines essentially ensure that the different timestamps are converted into a unique timestamp source - the computer’s processor high resolution counter (located in the hardware clock).

3.1 Tobii Studio recording data sources

The two basic data elements found in any Tobii Studio recording are the stimulus display events and gaze data elements. The stimulus display events contains information on the start and stop times of the different media used during the test, as well as key presses and mouse clicks, while the gaze data contains information related to the position of the eyes, distance to the sensor, pupil size and gaze point location on the screen for each eye (eye tracker data output). The time stamping of the stimulus display events occurs in the computer running Tobii Studio whereas the gaze data timestamps are obtained from the Tobii Eye Tracker.

During replay Tobii Studio uses the time information in the stimulus display events data to playback the stimulus and synchronize with the display of the gaze data. Additional to these data sources Tobii Studio will also store media elements such as user video and sound, captured from a camera and microphone connected to the computer, screen recordings, images, videos, pdf and web pages with timing information. Once again Tobii studio uses the TTime function to time stamp the different media files with the same time counter as the other data sources and enabling the synchronization during playback.

The type of media stored in a Tobii Studio recording depends on the type of stimulus used during the test, Table 2 shows a list of Tobii Studio stimulus and the respective media files that are saved together with the two basic data elements during a recording.

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Bitmap</th>
<th>Screen recording</th>
<th>Web page bitmap</th>
<th>Camera recording</th>
<th>Video capture card recording</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image</td>
<td></td>
<td>Image files are recalled during replay and analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Movie</td>
<td></td>
<td>Movie files are recalled during replay and analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen Rec</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Video</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Scene Camera</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Questionnaire</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDF Element</td>
<td></td>
<td>PDF pages are recalled during replay and analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In the next sections of this chapter we shall look at how Tobii Studio time stamps and synchronizes the different data sources that are part of an eye tracking recording.

### 3.2 Stimulus display and time stamping of stimuli display related events

Tobii Studio initiates the stimulus presentation by sending a command to the computer’s graphics card to write the stimulus data to its memory. The speed at which the stimulus is written on the graphics card is affected by different factors, such as, the type and size of the stimulus; the computer’s processor usage and speed; the RAM memory allocated to this command; and the graphics card speed and available memory. When this process is finished the stimulus is sent to the monitor via a VGA or DVI connector.

![Computer and eye tracker hardware involved in stimulus display.](image)

In the monitor the stimulus is first stored and buffered by the screen controller card (typically 1-3 frames) and then sent to the display following the vertical screen refresh rate. Refresh rates are measured in Hz and represent the number of times per second the monitor refreshes the pixels on its display. We can calculate the time interval between two screen refresh cycles by dividing 1000 ms (=1 second) by the refresh rate, for example, a 60 Hz refresh rate is equivalent to an image update every 16.6 ms. If an image is sent to the screen while it is refreshing, the offset for its display will depend only on how fast each pixel on the screen changes, i.e. the response time of the display. However if the image is sent to the display between two refresh cycles, then the offset will vary depending on where in the cycle the image becomes available. For example, if the image is available in the middle of a 60 Hz refresh cycle, it will take 8.34 ms extra to be shown on the screen, as it has to wait for the next refresh event to be displayed. The response time of the display is the amount of time that it takes for a pixel to change color and/or brightness, when the image arrives at the display. Typically this interval varies with the degree and order of the change. For example, the time it takes for a pixel to change from white to black is different compared to when it changes from black to white.

In order to obtain accurate timestamps associated to the events related to the stimulus presentation (e.g. image start and stop times), the command used to create the timestamp in Tobii Studio is sent to the TTime...
function simultaneously with the last command that initiates the process of displaying the stimulus on the screen. The TTime function reads the counter in the computer hardware clock and sends the timestamp information to Tobii Studio. This timestamp will then be used to synchronize the image presentation and the gaze data in the Tobii Studio recordings. The speed at which these different commands are executed depends on the number of processes running on the computer and the ability of the Tobii Studio software to interrupt other processes to gain access to the graphics card memory and the hardware clock.


For movies: The timestamp is taken during the call “IDirect3DDevice9::Present” in Direct3D.

3.2.1 Tobii Studio stimulus display offset test results
All tests in this section were performed using Tobii recommended hardware and computer components, and were specifically design to test if the start stimulus display timestamp information in Tobii Studio, produced an offset below 50 ms.

A sensor was attached to the Eye tracker screen in order to determine the precise moment each stimulus becomes available on the display. The sensor detects light emission transitions and is connected to a data acquisition device that samples data at an extremely high rate, enabling it to register changes on the screen faster than the screen refresh rate. By editing the stimulus to contain areas with black-white-black transitions and analyzing the data output of the sensor during stimulus presentation by Tobii Studio we can detect and timestamp the precise instants (frames) when changes occur on the screen. These timestamps can then be compared with the equivalent stimulus presentation timestamps in the Tobii Studio recording data file (Text export file), and the offsets calculated. Since the sensor and Tobii Studio use different counters (systems) to time stamp the events we also need a way to synchronize the data from the two devices. One way to achieve this is to provide a common event that can be measured accurately by both devices. The timestamp created by this event will then act as a reference point between the two data sets. If we then calculate the difference between the timestamps associated to the stimulus presentation and the reference point for each data set, we obtain a value that can be directly compared and used to estimate the time offset between the two data sets.

Figure 8 Shows the different computer and eye tracker hardware components that produce the stimulus display offset.
A simulated eye blink was used as a reference event which is recorded shortly after the stimulus is displayed on the screen. Both the eye tracker and the sensor are able to detect and record this type of event with a high timing accuracy. During a blink, the shutter obstructs the pupil of the eye model causing a momentarily loss of eye position data that is represented in the Tobii Studio Text export data file a missing value (-1), while the data acquisition device connected to the eye-eyelid model registers the time when the shutter was closed.

![Figure 9 Stimulus offset test setup.](image)

In the first appendix of this document an alternative setup is described that can be used to measure the same offset error as the setup described above.

The following components were tested in different configurations (see Tables 5a and 5b, in page 19 and 20, for an overview of the configurations and summary of the test results):

- Tobii T120, T60 and T60XL Eyetrackers
- Dell precision T5400: Intel Xeon Processor E5420 @ 2.5 GHz, 4GB RAM; with ATI FireGL V3300 128MB or NVidia Quadro FX570 256MB graphics card.
- Dell M6400 laptop: Core2Duo @ 2,67GHz, 4GB RAM, NVidia Quadro graphics adapter. Windows Vista business set to Classic view.
- Dell M6300 laptop: Core2Duo @ 2,4GHz, 2GB RAM, NVidia Quadro graphics adapter. Windows XP Professional set to classic view.
- VGA vs. DVI cables
- Windows XP vs. Vista
- Single image: BMP and JPEG formats, 1280x1024 pixels resolution (T60 and T120 native screen resolution) and 1920x1200 pixels resolution (T60XL native screen resolution).
- Multiple stimuli: 4 image sequence and 3 videos sequence.
- Single videos: techsmith1.avi, dviX.avi, MV1.avi and XviD codec; 1280x1024 (T60 and T120 native screen resolution) and 1920x1200 (T60XL native screen resolution) pixels resolution; max. 25fps.

The results of the tests confirm that all tested configurations show a timestamp offset error between 12 and 50 ms.
3.3 User Camera Video, External video, Scene camera, Sound & Screen recording

During a Tobii Studio recording, the different media and data elements contained in the test will be displayed to the participant following the order determined by the test design. During this process each element’s start and stop timestamps are registered in the events data element of the recording. Tobii Studio plays back each of the media elements according to the information contained in the stimulus events data element. The playback synchronization with the gaze data is then determined by the accuracy of the timestamps registered on the events data element. In some cases additional synchronization issues may arise due to media load time to the graphics card, however these offsets are more common when the computer has insufficient CPU or memory resources.

![Figure 10 External capture device recording delay.](image)

3.3.1 General information on digital video media files

A video media file generally consists of a video data package containing a video header block and a video data block - this type of file format is often called container or wrapper (e.g. AVI is a commonly used container for video media in Windows). The video data block contains video images or frames, while each header block contains timestamp information, identity of the format of the video data, type of compression, image size, display order, and other global parameters of the respective frames. The frames are first captured by the sensor on an external device (e.g. user camera or video capture card) and then processed via a codec into a media file. The codec is a software or device capable of encoding and decoding video data, in Tobii Studio this process is done using Microsoft Windows’ media framework DirectShow. DirectShow is a multimedia framework and API responsible for performing various operations with media files or streams in a Windows environment, for example it includes filters for decoding some common media file formats.

Both the camera, codec and the file can have internal buffers that wait for a certain amount of information to be accumulated before processing it and sending it to the next stage. A frame is given a timestamp in the device driver or when the DirectShow framework is producing the container file. The timestamp is taken from the clock specified in the framework (for example, in a Tobii Studio recording this is done by calling the TTime function to read the Hardware counter of the computer).
Besides the video image data and information blocks the media container files may also include audio packets. In turn each audio data packet includes an audio data block that has any number of blocks or frames of audio data, and a header block that contains information about the format of the audio data as well as instructions related to how the audio data is to be decoded and processed (the latter includes timestamp information). Similarly to the video packages the audio header blocks also contain timestamp values that are obtained using the TTime function and the computer hardware counter.

The decoding of the video data normally requires decompression, conversion of partial frames into full frames and the recognition of full frames. Simultaneously with the decoding process, the frames of audio and video data are being output, i.e. played back to the user, and playback must be synchronized such that the frames of audio and video present a coordinated and coherent presentation. This activity generally requires some resources from the computer’s processor, RAM memory and graphics card and is greatly affected by the video properties such as resolution and size of each frame, frame rate and codec type.

3.3.2 Tobii Studio video and sound capture

During a Tobii Studio recording the audio and video signals are captured and stored in separate media files. In the Tobii T series (T60, T120 & T60XL) the video source is provided by either the user camera located in the eye tracker, from an external video camera connected to the computer running Tobii Studio, or from the graphics card or video capture card in the Tobii Studio computer. In order to produce a coordinated and coherent replay of the audio and video recordings the playback must be synchronized. This is done via the TTime function during the recording stage, Tobii Studio uses this function to time stamp both user video and sound at the start and end of the recording. This start time will then be used during the playback of the recording.

![Diagram](image)

**Figure 11** Tobii Studio video and sound capture. Each element start time is time stamped in Tobii Studio event data.

To synchronize different video and audio streams two methods are used in different versions of Tobii Studio:

- During the recording Tobii Studio stores the start time of the video stream as accurately as possible (i.e. time stamps the first frame with accuracy), this is the method used in Tobii Studio 2.0 and earlier versions.
• It uses a container format that stores timestamps for each frame (according to a recording start offset) so that the file plays exactly as it was recorded, even if it drops frames. This is the new method used by Tobii Studio 2.1 and future subsequent releases.

In order to store the start time accurately Tobii Studio inserts a timestamp module between the video device and codec, that collects and stores TTime timestamps in the media container. Simultaneously it also stores the timestamps of the video driver on the replay database. Although the video/audio device may have internal buffers this is the most general and accurate way to synchronize streams of different types. In the gaze Replay tool of Tobii Studio a playlist is built using the timestamps stored in the database. As long as no new synchronization issues are introduced in the replay framework the media streams will be synchronized thanks to the start timestamps stored in the database and the media container format that contain timestamps for each frame.

In the new version of Tobii Studio (version 2.1) it is possible to select the container format you wish to use to perform the recording, whereas older versions of Tobii Studio (versions 2.0 and older) uses only one container type. The two options are: AVI, this option was present in the old versions of Tobii Studio and ASF new container available in Tobii Studio 2.1. AVI, is a multimedia container format introduced by Microsoft in November 1992 as part of its Video for Windows technology. AVI files can contain both audio and video data in a file container that allows synchronous audio-video playback. The first frame in each recording is time stamped and the remaining frames are recorded without timestamp information. AVI containers often suffer frame losses during the coding and processing of video data, the absence of times tamped frames can lead to a loss of synchrony between the audio and video streams during playback. To cope with frame losses Tobii Studio’s replay tool stretches the video tracks (by repeating some frames) to the full duration of the playback that is stored in the database. As long as the video device doesn’t drop frames during the replay this approach works very well. Advanced Systems Format (formerly Advanced Streaming Format or Active Streaming Format) is Microsoft's proprietary digital audio/digital video container format, especially meant for streaming media. ASF is part of the Windows Media framework, this format contains a timestamp for each media frame facilitating media synchronization, even in situations when frames have been dropped during the recording. In this case synchronization will still be maintained as by the possibility to match each frame timestamp with the data timestamp, the frames that are dropped a replaced by the previous frame during the playback of the recording.

Preliminary tests show that recording and replay works well using both containers, however sound and video synchronization becomes worse with time in a recording when using the AVI container. No synchronization deterioration was detected when using the ASF container. In three different setups, lip sync has been reached in two when using ASF and none of the setups reached lip sync when using AVI. The three user cam setups that have been tested are Logitech Pro 9000, Logitech Sphere and the future Tobii User cam coupled with a Logitech USB microphone.

<table>
<thead>
<tr>
<th>User Camera</th>
<th>Tobii Studio 2.0 or earlier</th>
<th>Tobii Studio 2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logitech 9000 Pro + own microphone</td>
<td>-110±35</td>
<td>150±130</td>
</tr>
<tr>
<td>Logitech Sphere + own microphone</td>
<td>15±40</td>
<td>160±40</td>
</tr>
<tr>
<td>T60/120 user camera + Logitech microphone</td>
<td>-20±65</td>
<td>40 ± 20</td>
</tr>
</tbody>
</table>

When using the external video media, no difference in synchronization could be seen when using AVI or ASF. The external video setup tests were made with two different framegrabber cards: Black magic Intensity Pro and Datapath UFG-03 and one setup using the Logitech Sphere and Pro 900 web cameras.

3.3.3 Tobii Studio video replay
In Tobii Studio 2.0 and earlier versions, the start and stop times of a recording is set to the start and stop times of the longest media in the test. For example, when a user camera is used to record the test subject, the
camera video feed will be started first and ended last during the test recording. The start and stop times of the user camera recording is extracted from the moment the first and last video frame reaches Tobii Studio timestamp module, and not from timestamps produced by the camera. Since the user camera recording element is the longest one in the recording playlist Tobii Studio sets the start and stop time of the replay to the start and stop timing information of the user camera. In most cases this process guarantees a good synchronization between the user camera video playback and the remaining elements in the recording. However since the timestamp is determined when the frames arrive to Tobii Studio and not during the image exposure to the sensor of the user camera there might be a small delay between the user camera video and the gaze data.

Figure 12 Tobii Studio 2.0 (and earlier versions) recording replay. The start time of the replay is synchronized with the longest element in the recording, the video element in this example.

In Tobii Studio 2.1 the start and stop time of a recording is always set to when the eye tracker is started and stopped. This is the very first and last thing that happens when a test is run so in cases when it takes time to initiate the user camera capture there will be “gaps” in the beginning and the end of the recording while other elements and capture devices are starting up and closing down.

Figure 13 Tobii Studio 2.1 recording replay. The gaze data collection is always initiated first during the recording and the start time of the replay is synchronized with the gaze data element in the recording.
If a recording made with a previous version of Tobii Studio is opened in Tobii Studio 2.1 it will be synchronized using the new synchronization technique introduced in 2.1. In most cases you should not be able to notice any difference but some cases the synchronization between video and audio streams will appear worse than they did in 2.0. If that is the case it’s a result of the limited synchronization in the recorder in earlier versions of Tobii Studio. The replay appearance in Tobii Studio 2.1 is what we actually captured when recording. If a new recording is made in the same test it will be recorded using the new and improved synchronization capabilities.

### 3.4 Summary
Tobii Studio computing operations are subject to the operating system allocation of memory and processor resources, as a consequence time offsets may occur when recording an eye tracking session. These offsets may vary according to the eye tracker display hardware (Table 4) and type of computer used in the test (laptop vs. desktop, processor speed, RAM memory, graphics card), the video capture devices (capture cards and cameras), third party software installed and running on the computer, type and stimuli format used in the test (image, video, codecs).

<table>
<thead>
<tr>
<th>Eye Tracker</th>
<th>Native resolution (pixels)</th>
<th>Controller refresh rate (Hz)</th>
<th>*TFT Response time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1750</td>
<td>1280×1024</td>
<td>60-75</td>
<td>5-25</td>
</tr>
<tr>
<td>T120 &amp; T60</td>
<td>1280×1024</td>
<td>60-75</td>
<td>5-16</td>
</tr>
<tr>
<td>T60XL</td>
<td>1920×1200</td>
<td>60</td>
<td>16</td>
</tr>
</tbody>
</table>

*These values represent typical values for a full transition of a pixel from black-white-black.

Recent tests show that when using Tobii Studio with Tobii recommended hardware configurations and devices these offsets can be maintained at a low value. Tables 5a and 5b summarize the different configurations of computer hardware and stimulus that have been used to test the stimulus display timing offset. These tests measure the offset between the Tobii Studio recording timestamps and the stimulus display on a screen. All setups show that the offset between Tobii Studio timestamps and image or video display is below 50 ms.

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Format</th>
<th>Codec</th>
<th>Eye tracker</th>
<th>Windows OS</th>
<th>Offset (ms), means±s.d.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single image</td>
<td>JPEG</td>
<td>-</td>
<td>T120</td>
<td>XP</td>
<td>20±6</td>
</tr>
<tr>
<td></td>
<td>BMP</td>
<td>-</td>
<td>T120</td>
<td>XP</td>
<td>17±5</td>
</tr>
<tr>
<td></td>
<td>JPEG</td>
<td>-</td>
<td>T60</td>
<td>Vista Classic***</td>
<td>43±6**</td>
</tr>
<tr>
<td></td>
<td>JPEG</td>
<td>-</td>
<td>T60 XL</td>
<td>XP</td>
<td>33±7*</td>
</tr>
<tr>
<td>4 images sequence</td>
<td>JPEG + BMP</td>
<td>-</td>
<td>T120</td>
<td>XP</td>
<td>18±5</td>
</tr>
<tr>
<td>Single video</td>
<td>AVI</td>
<td>techsmith</td>
<td>T120</td>
<td>XP</td>
<td>28±6</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>dviX</td>
<td>T120</td>
<td>XP</td>
<td>32±5</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>MV1</td>
<td>T120</td>
<td>XP</td>
<td>32±4</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>XviD</td>
<td>T120</td>
<td>XP</td>
<td>31±5</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>techsmith</td>
<td>T120</td>
<td>Vista Classic***</td>
<td>21±7</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>MV1</td>
<td>T120</td>
<td>Vista Classic***</td>
<td>18±5</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>XviD</td>
<td>T60</td>
<td>XP</td>
<td>20±4*</td>
</tr>
<tr>
<td></td>
<td>AVI</td>
<td>XviD</td>
<td>T60XL</td>
<td>Vista Classic***</td>
<td>36±7**</td>
</tr>
<tr>
<td>3 videos sequence</td>
<td>AVI</td>
<td>Techsmith, MV1, dviX</td>
<td>T120</td>
<td>XP</td>
<td>31±5</td>
</tr>
</tbody>
</table>

*This test was performed using Dell Precision M6300 laptop computer.
** This test was performed using Dell Precision M6400 laptop computer.
*** Windows Vista appearance settings optimized for better performance.
Table 5b. Test results for the offset of the start time of the stimulus display – display connector, graphics card, and type of computer (Dell Precision T5400 desktop, M6300 and M6400 laptop)

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Display connector</th>
<th>Card*</th>
<th>Computer</th>
<th>Eye tracker</th>
<th>Windows OS</th>
<th>Offset (ms), mean±s.d.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single image</td>
<td>DVI</td>
<td>Nvidia</td>
<td>T5400</td>
<td>T120</td>
<td>XP</td>
<td>19±5</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
<td>ATI</td>
<td>T5400</td>
<td>T120</td>
<td>XP</td>
<td>15±5</td>
</tr>
<tr>
<td></td>
<td>VGA</td>
<td>Nvidia</td>
<td>M6300</td>
<td>T60</td>
<td>XP</td>
<td>43±7</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
<td>Nvidia</td>
<td>M6300</td>
<td>T60</td>
<td>XP</td>
<td>39±4</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
<td>Nvidia</td>
<td>M6400</td>
<td>T60</td>
<td>Vista Classic**</td>
<td>43±6</td>
</tr>
<tr>
<td>Single video</td>
<td>DVI</td>
<td>Nvidia</td>
<td>T5400</td>
<td>T120</td>
<td>XP</td>
<td>25±5</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
<td>ATI</td>
<td>T5400</td>
<td>T120</td>
<td>XP</td>
<td>30±5</td>
</tr>
<tr>
<td></td>
<td>VGA</td>
<td>Nvidia</td>
<td>M6300</td>
<td>T60</td>
<td>XP</td>
<td>33±7</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
<td>Nvidia</td>
<td>M6300</td>
<td>T60</td>
<td>XP</td>
<td>35±9</td>
</tr>
</tbody>
</table>


** Windows Vista appearance settings optimized for better performance

Additional to the stimuli display offsets one can also encounter video and sound capture offsets, that are caused by the latency between image recording or capture (e.g. at image sensor exposure when using a camera) and the Tobii Studio timestamps.

### 3.5 Recommendations

During the design of an experiment it is always good practice to test both experimental procedure as well as the equipment you will be using to perform the tests. If you are conducting Eye Tracking experiments that require high timing precision we strongly recommend testing your eye tracking setup with the stimuli you will be presenting as well as conduct a pilot to access the quality of the data.

Tobii Studio timing offsets can be reduced by following these recommendations:

- Optimize your computer hardware by using Tobii recommend components. For an overview of the configurations tested by Tobii Technology read Table 5 on the next page, for a more detailed list of Tobii Eye Tracking Recommended components read the System Recommendations for Tobii Studio and TX-series Eye Trackers.
- The computer running Tobii Studio should have the least number of processes running in the background, this means that you should disable or uninstall any unnecessary programs that might be running in the background (Anti-virus software, Firewall software, Anti-Spyware software, Skype or other chat/communication programs) – ideally use the computer only to run Tobii Studio tests and analysis.
- It may help to disable or disconnect Bluetooth, WLAN/Wifi and network connectors and to unload CD or DVD disk drives.
- Use a direct connection between your computer and the Tobii Eye Tracker. For more information on how to connect the computer and the Tobii Eye Trackers read the section entitled “Connecting Tobii Eye Tracker” in your Eye Tracker User Manual.
- Use images and movies as stimuli (using the recommended formats and codecs - Table 6 below).
- Don’t run your study at excessive screen resolutions, choose something reasonable for your purpose. Higher resolutions take up precious video RAM graphics memory and increase the time for clearing the frame buffer, drawing your stimuli and for post-processing operations.
- Make sure that the energy saving options of your machine do not slow down the processor and other subsystems to save power or battery life. Especially important for laptops on battery power.
- Running two displays simultaneously requires more video RAM memory and processor resources in your graphics card, plus some additional management overhead in the graphics driver. This can cause lower graphics performance and therefore lead to missed stimulus onset deadlines when running complex stimuli on high resolution, high refresh rate displays, especially when using older mobile graphics chips with a low amount of local video RAM.
- Test the timing of your setup by following the test procedures described in the Tobii Studio Timing Testing Guidelines (Appendix I).

| Table 5. Hardware recommended configurations (includes Operative system and studio versions) |
|---------------------------------|--------|--------|---------------|-----------------|-----------------|
| Computers                      | CPU    | RAM    | Video RAM    | Graphic card   | Operating System | Tobii Studio versions |
| **Laptop**                     |        |        |              |                 |                 |                   |
| Dell Precision M6300           | Core 2Duo, 2.4GHz | 2 GB 4 GB | 256 MB 512 MB | NVIDIA Quadro FX1600M NVIDIA Quadro FX2700M | Windows XP Professional SP3 Windows Vista Business SP2, Classic view | 1.6 and later 1.6 and later |
| Dell Precision M6400           | Core 2Duo, 2.67GHz | 4 GB | 512 MB | NVIDIA Quadro FX2700M | Windows Vista Business SP2, Classic view | 1.6 and later |
| **Desktop**                    |        |        |              |                 |                 |                   |
| Dell Precision T5400           | Core 2Duo, 2.5GHz | 4 GB 4 GB | 256 MB 128 MB | NVIDIA Quadro FX570 ATI FireGL V3300 | Windows XP Professional SP3 Windows XP Professional SP3 | 1.6 and later 1.6 and later |

| Table 6. Recommended stimulus |
|--------------------------------|--------|--------|---------------|-----------------|-----------------|
| **Stimulus**                   | Format | Codec | Size          | Frame rate      |                 |
| Image                          | JPEG, BMP | - | 1280x1024; 1920x1200 (T60 XL) | - |                 |
| Video                          | AVI    | Techsmith1, dvix, MV1, XviD | 1280x1024; 1920x1200 (T60 XL) | maximum 25 fps |                 |
Appendix I -
How to test the synchronization between video/image recording and actual screen display time using a high speed camera
Tobii Studio Timing Testing:
How to test the synchronization between video/image recording and actual screen display time using a high speed camera

September 21, 2009
Tobii Technology AB
## TOBII STUDIO TIMING TESTING – SYNCHRONIZATION BETWEEN VIDEO/IMAGE RECORDING AND ACTUAL SCREEN DISPLAY TIME

### CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tobii Studio Stimulus Display</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Introduction to the test setup</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Equipment needed</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Procedure</td>
<td>5</td>
</tr>
<tr>
<td>4.1</td>
<td>Step 1</td>
<td>5</td>
</tr>
<tr>
<td>4.2</td>
<td>Step 2</td>
<td>5</td>
</tr>
<tr>
<td>4.3</td>
<td>Step 3</td>
<td>6</td>
</tr>
<tr>
<td>4.4</td>
<td>Step 4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>Data Analysis</td>
<td>7</td>
</tr>
<tr>
<td>5.1</td>
<td>Video recording</td>
<td>7</td>
</tr>
<tr>
<td>5.2</td>
<td>Tobii Studio Recording</td>
<td>7</td>
</tr>
<tr>
<td>5.3</td>
<td>Final offset calculation</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>Potential sources of error during measurements</td>
<td>9</td>
</tr>
<tr>
<td>6.1</td>
<td>Tobii Studio data</td>
<td>9</td>
</tr>
<tr>
<td>6.2</td>
<td>Measurements using the high speed camera recording</td>
<td>9</td>
</tr>
</tbody>
</table>
1 Tobii Studio Stimulus Display

Tobii Studio initiates the stimulus presentation by sending a command to the computer’s graphics card to write the stimulus data to its memory. The speed at which the stimulus is written to the graphics card is affected by different factors: the type and size of the stimulus; the computer’s processor usage and speed; the RAM memory allocated to this command; and the graphics card speed and available memory. When this process is finished the stimulus is sent to the monitor via a VGA or DVI connector. In the monitor the stimulus is first stored and buffered by the screen controller card (typically 1-3 frames) and then sent to the display following the vertical screen refresh rate. Refresh rates are measured in Hz and represent the number of times per second the monitor refreshes the pixels on its display. We can calculate the time interval between two screen refresh cycles by dividing 1000 ms (=1 second) by the refresh rate, for example, a 60 Hz refresh rate is equivalent to an image update every 16.6 ms. If an image is sent to the screen while it is refreshing, the offset for its display will depend only on how fast each pixel on the screen changes, i.e. the response time of the display. However if the image is sent to the display between two refresh cycles, then the offset will vary depending on where in the cycle the image becomes available. For example, if the image is available in the middle of a 60 Hz refresh cycle, it will take 8.34 ms extra to be shown on the screen, as it has to wait for the next refresh event to be displayed. The response time of the display is the amount of time that it takes for a pixel to change color and/or brightness, when the image arrives at the display. Typically this interval varies with the degree and order of the change. For example, the time it takes for a pixel to change from white to black is different compared to when it changes from black to white.

Figure 1. Overview of the stimulus display process during a Tobii Studio recording.

In order to obtain accurate timestamps associated to the events related to the stimulus presentation (e.g. image start and stop times), the command used to create the timestamp in Tobii Studio is sent to the TTime function simultaneously with the last command that initiates the process of displaying the stimulus on the screen. The TTime function reads the counter in the computer hardware clock and sends the timestamp information to Tobii Studio. This timestamp will then be used to synchronize the image presentation and the gaze data in the Tobii Studio recordings. The speed at which these different commands are executed depends
on the number of processes running on the computer and the ability of the Tobii Studio software to interrupt other processes to gain access to the graphics card memory and the hardware clock.

The aim of this document is to provide a brief description of a setup that can be used to test Tobii Studio timing offsets related to stimulus display. This setup will allow you to measure the time interval from when Tobii Studio takes a timestamp to mark the start of image presentation, to when the image is actually displayed on the screen (figure 2).

![Diagram showing the concept of timing offsets between Tobii Studio and the display]

Figure 2. Offset measured by the present setup.

2 Introduction to the test setup

In order to determine the offsets related to stimulus presentation, we need to determine the precise moment the stimulus becomes available on the display. High speed video cameras are commonly used to measure very fast visual events as they are capable of producing recordings at high frame rates (i.e. number of frames per second). By analysing a high speed video recording of a stimulus presentation by Tobii Studio, we can detect and timestamp the precise instants (frames) when changes occur on the screen. These timestamps can then be compared with the equivalent stimulus presentation timestamps in the Tobii Studio recording data file (Text export file), and the offsets calculated. Since the video camera and Tobii Studio use different counters (system clocks) to timestamp the events, we also need a way to synchronize the data from the two devices. One way to achieve this is to provide a common event that can be measured accurately by both devices. The timestamp created by this event will then act as a reference point between the two data sets. If we then calculate the difference between the timestamps associated to the stimulus presentation and the reference point for each data set, we obtain a value that can be directly compared and used to estimate the time offset between the two data sets.
In the current test setup a voluntary eye blink is used as a reference event, which is recorded shortly after the stimulus is displayed on the screen. Both the eye tracker and the video camera are able to detect and record this type of event with a high timing accuracy. During a blink, our eyelids will obstruct the pupil causing a momentarily loss of eye position data that is represented in the Tobii Studio Text export data file as a missing value (-1), while the video camera visually records the act. In order to simultaneously record the blink and the changes on the display, the video camera has to be positioned facing the test participant and a mirror has to be placed close to the test participant’s head so that the camera is also able to capture the reflection of the screen on the mirror - figures 3 and 4.

Figure 3. Frame from a High Speed Camera recording using the test setup. The camera is facing the participant that has a mirror beside him. The screen where the image is being displayed is filmed through the reflection on the mirror. This frame pictures the moment when the image appears on the eye tracker screen (right).

3 Equipment needed
- High speed video camera (minimum 1000 fps)
- Mirror (suggested size 40x40cm)
- Video editing software (e.g. VirtualDub – freeware software http://www.virtualdub.org/; if you are recording in a different format than .avi you additionally need an application to convert the file e.g. MPEG StreamClip – freeware software http://www.squared5.com/svideo/mpeg-streamclip-win.html)
- Spreadsheet application or equivalent (e.g. Microsoft Office Excel)
- Test computer with Tobii Studio software
- Tobii T-Series Eye Tracker (or Tobii X-Series Eye Tracker and a computer display)

4 Procedure

4.1 Step 1
Setup the eye tracker and connect it to the Tobii Studio computer. Start Tobii Studio and create a Tobii Studio Project and Test. Choose the type of stimulus you wish to test by dragging it to the timeline and set the stimulus presentation time to at least 30 seconds.

4.2 Step 2
Position the test participant in front of the eye tracker and position the mirror at the same height as the test participant’s eyes/head. Ask the participant to cover the eye opposite to the mirror (see figure 3). Use the Tobii Studio’s Track Status to verify that the eye tracker is able to detect the participant’s eye and that the participant is seated at the correct distance (63-65 cm) from the eye tracker.

Position the camera next to the monitor facing the participant, filming both his eyes and the mirror. Re-orient the mirror and adjust the position of the computer so that the screen where the stimulus will be presented becomes visible in the reflection (see figures 3 and 4). If necessary, adjust the light in the room so that the eyes are clearly visible in the video footage.
Figure 4. Test setup – the participant is positioned in front of the eye tracker with a mirror placed next to his head at eye level. The camera is placed facing the participant enabling it to simultaneously record the participant’s eye and the reflection of the screen in the mirror.

4.3 Step 3
Start the Tobii Studio recording and calibrate the participant. After the calibration, start the video camera recording, and press Start Recording in the Start Recording dialogue in Tobii Studio in order to start the stimulus presentation. Ask the participant to blink once a few seconds after the stimulus has appeared on the screen.

For statistical purposes, repeat this step at least 20 times to obtain 20 Tobii Studio recordings and thus 20 measurements of the offset.

Note: To avoid doing a calibration every time you start a recording use the same participant and uncheck the Always calibrate before recording starts checkbox on the Select Participant dialog.

4.4 Step 4
Video Camera
Upload the video recording to the computer. If recorded in another format than AVI, convert the file into an AVI format (make sure to keep the original frame rate).

Tobii Studio
After you have made all the recordings, you need to export the raw data using Tobii Studio’s Text Export feature: Go to the Replay view, press the Export button on the Replay tab and click on Text Export. On the Text Export dialog, choose the folder you want to export the data to. Select All recordings in the test in the Export scope field and check the All Data check box in the Preset Configurations tab. Press Export when you are ready to export the recordings.
5 Data Analysis

5.1 Video recording

We will first use the frame counter in the video editing software to determine the number of frames between the two events. Open the video recording using Virtual Dub. Replay the recording until you detect the frame in the recording corresponding to the moment when the image/or 1st frame of the video is fully displayed on the screen, then note the frame count. Continue viewing the recording until you reach the frame showing the eyelid covering half the participant’s eye (figure 5) when the person starts to blink.

![Video recording analysis](image)

Figure 5. Test setup – Video recording analysis. The frame showing the eyelid covering half the participant’s eye is frame number 21470 (the count is displayed on the toolbar located below the image).

We calculate the time between the two events, by subtracting the two frame count values, and converting the resultant value to milliseconds using the frame rate of the recording.

For example, a recording was done using a frame rate of 1.200 fps. During the analysis we detected that the image appeared on the screen at frame 9871 and the blink was recorded at frame 10513, the number of frames between the two events will be equal to 10513-9871= 642. From the frame rate we know that each frame is taken every 1000/1200 (=0.83ms). We then multiply this value by the number of frames that occurred between the two events to calculate the time between the two events i.e. 642x0.83= 532.86 ms.

5.2 Tobii Studio Recording

In order to get the same information from the Tobii Studio recordings, we need to extract the following information from the files generated by the Text export: the timestamp associated to the ImageStart (or MovieStart) event, and the timestamp associated to the first row where the gaze data columns show invalid gaze data, i.e. either -1 or negative full screen resolution values, (figures 6a and 6b respectively). The files can be opened using Microsoft Excel and the timestamps can be extracted consulting the Timestamp, Event and gaze position related (e.g. CamXLeft, DistanceLeft) columns. Finally we can calculate the duration of the interval between the two events by subtracting the two timestamps.
Figure 6a. Tobii Studio Text export data file. The image was presented at 0 milliseconds (ImageStart event).

Figure 6b. Tobii Studio Text export data file. The blink was recorded 553 milliseconds after the image was presented (missing values -1 appear in the columns related to the gaze position data).
For example, in images 6a and 6b we can observe that Tobii Studio has registered that the image was presented at the start of the test, at time 0, whereas the blink occurred at 553 ms. Thus the duration of the interval between the two events was 553 ms.

5.3 Final offset calculation
To obtain the final offset value we subtract the two values extracted from the two data sources. In the previous examples we would subtract 553 (Tobii Studio) - 532.86 (High Speed Camera) = 20.14 ms

6 Potential sources of error during measurements

6.1 Tobii Studio data
Eye tracker sampling rate – can introduce an error to the Tobii Studio data regarding the eye blink. In a worst case scenario it introduces a maximum error of 16.67 ms at 60 Hz or 8.33 ms at 120 Hz.

6.2 Measurements using the high speed camera recording
Recording frame rate – By using a high frame rate ≥ 1000 fps we ensure that the error introduced by the camera sampling rate is small.

   Image start frame – Manual measurement set to when the image is fully displayed on the screen. Can be affected by intra and inter observer reliability factors i.e. the ability to identify on which frame of the recording the stimulus is shown in full.

   Eye blink – This is a manual measurement set to the frame where the eyelid covers half of the eye. Can also be affected by intra and inter observer measurement reliability.