Deliverable Number

D3.4

EMPOWER deliverables

Deliverable name	**Eye-Tracking Module**
Type	**DEM – Demonstrator, pilot, prototype.**
Dissemination level	**PU - Public**
Date	**Month 12**

System components needed to get eye-tracking data from devices to be used as an input for the platform games, as well as interfaces with WP4.

Description

WP.3

Work Package. 3

Lead Beneficiary – (UV)

IRTIC Universitat de València

Eye-tracking Module

Description of the Eye-tracking Module

System components are needed to get eye-tracking data from devices to be used as input for the platform games and interfaces with WP4.

Date	Version	Description	Authors
29.09.2023	1.0	First version	Attila Kovari, Jozsef Katona, Are Daehlen, Ilona Heldal, Jerry Chun-Wei Lin, Abdul Rehman Javed
17.01.2024	1.1	Updated version	Attila Kovari

Table of Content

#1. Introduction

The Eye-tracking (ET) Module will be used for considering the children's eye gaze data and visual focus during games-based testing to examine the children's cognitive behaviour. The focus is that the gaze data provided by the ET Module allows connecting the eye gaze parameters with emotion-based parameters and analysing these data (WP4).

#2. ET Module preliminaries and specification

Before starting the development, we took the following aspects into account:

- The most used eye-tracking devices manufactured by Tobii for both research and other purposes.
- In our experience, the Tobii eye-tracking devices are the best if the head moves during the test, which a Tobii expert also confirmed at a presentation.
- In most cases, 5-point calibration is used for gaze calibration.
- Based on the preliminary literature review, there is no unique detailed specification for the eye-tracking system for disabled children.

ET MODULE SPECIFICATION

ET device: Tobii Pro Nano eye-tracker hardware

ET device placement: the holder provided by the manufacturer for the ET device is used (under screen position). The order of the ET device above the screen must also be examined because this placement is preferable for a touch screen (the hand may cover the device).

Eye and head position calibration: the commonly used solutions were selected for these calibrations. The head position calibration screen can adjust the head to the correct position. The head is visualised using head/eye icons and indicates that actual position is right or wrong. For the eye gaze position calibration, the standard 5-point calibration is used.

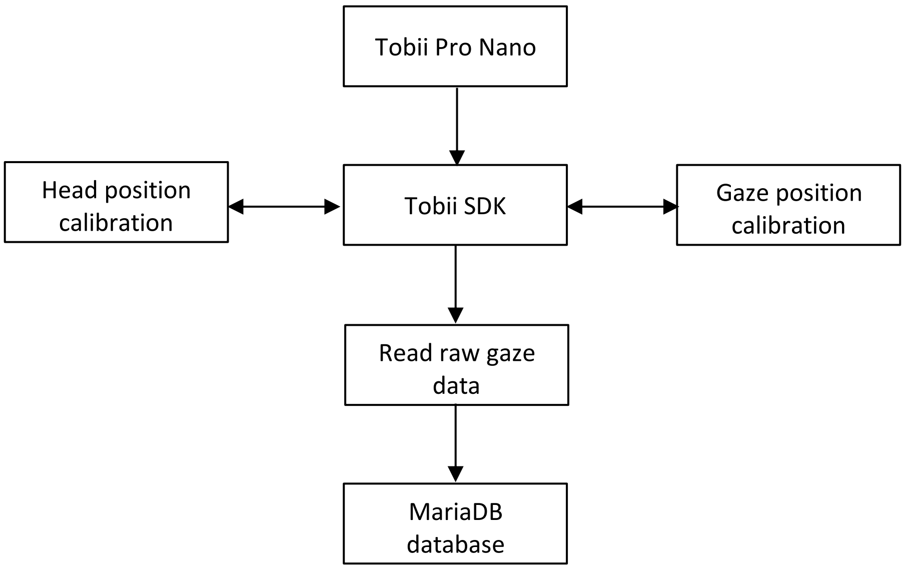
Interfaces: The ET Module provides head and eye gaze calibrations, which need two user interfaces: head position and gaze point calibration screens.

Provided data: left and right gaze position on the screen, left and suitable pupil diameter, and head position. Using these raw data, the additional feature extraction of general eye movement parameters (like fixations and saccades) is implemented as part of the WP4 platform algorithms.

#2. ET Module demonstration

If the ET data recording is started, it runs in the background and collects the data, and at the end of the test, the recorded raw data is stored in the MariaDB database. The platform game starts and stops the ET data recording and has access to real-time data. The central element of the system is the Tobii Software Development Kit (SDK), which manages the Tobii Pro Nano device and calibration and provides access to the raw gaze data. The system architecture is shown in Figure 1.

Figure 1 System architecture



The ET Module has two interfaces' head and eye gaze calibrations. These interfaces are presented in the following.

During the head calibration, by moving the position of the head, the head/eye icon on the screen also moves, and the appropriate place is indicated by a green background (Figure 2). While the position is imperfect, the background changes red depending on the degree of deviation from the correct position (Figure 3).

Figure 2 Head calibration screen - acceptable

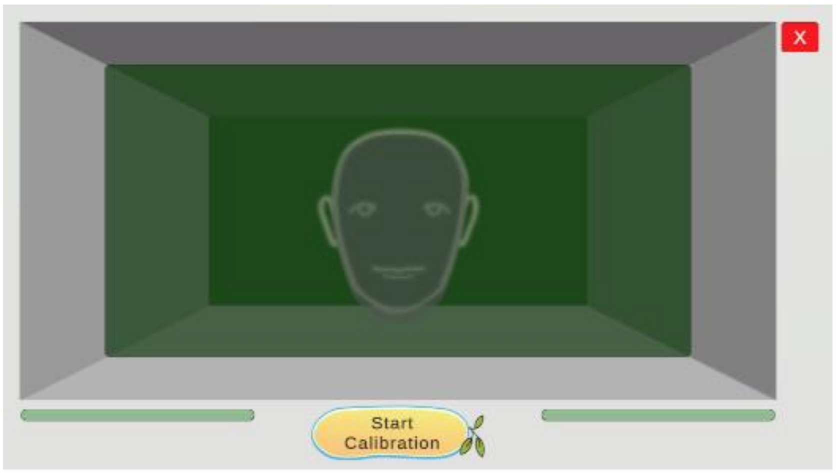
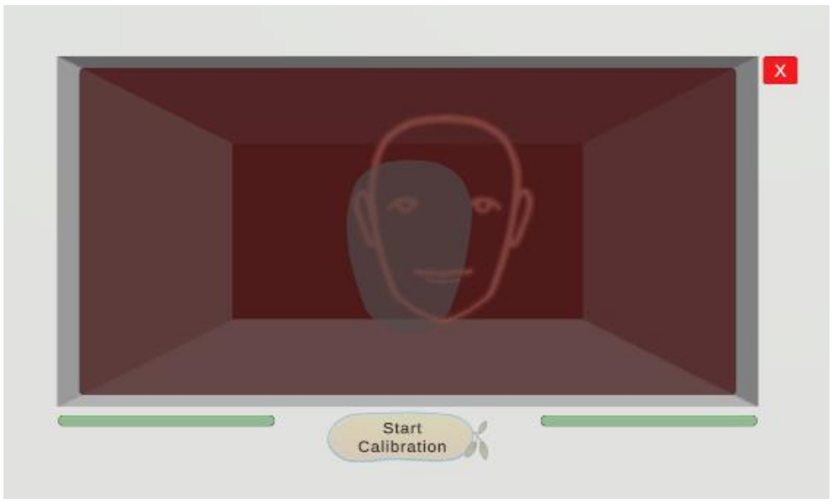
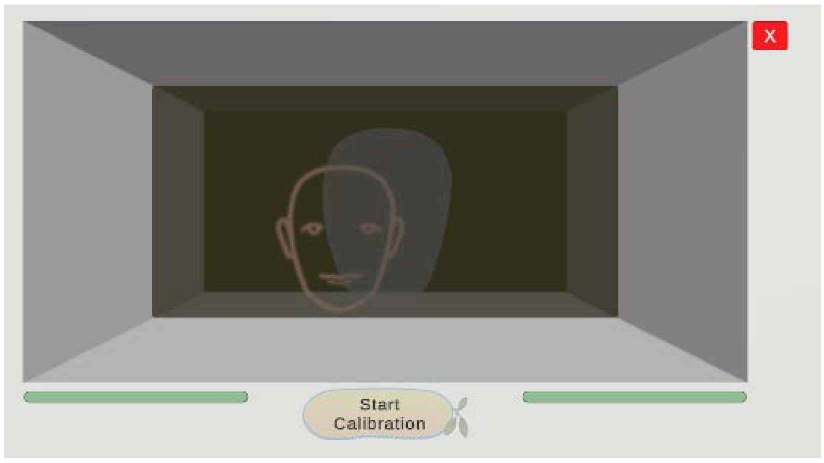


Figure 3 Head calibration screen - inadequate





Usually, the standard calibration process shows 5 (or 9) points to get a helpful calibration that works in most areas of the screen (Figure 4). This was also implemented in the ET Module (Figure 5).

Figure 4 5-point gaze position calibration

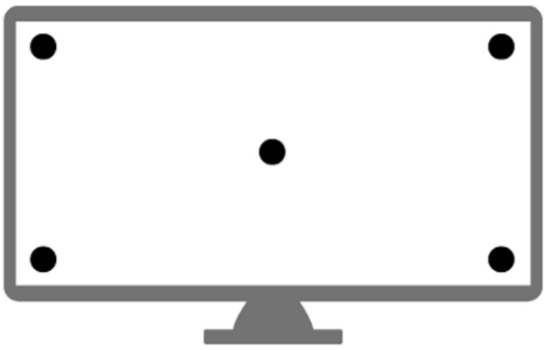
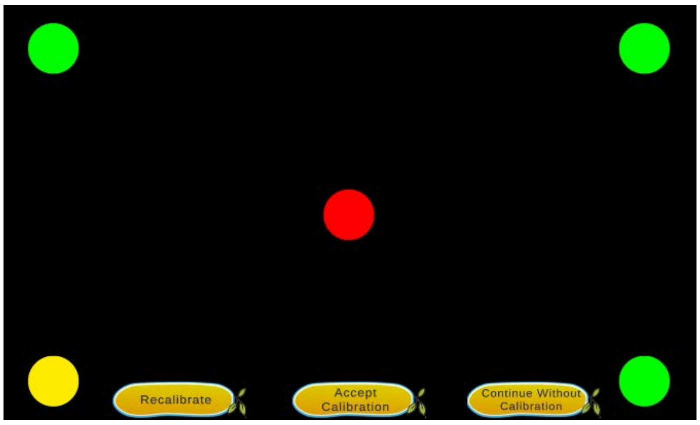
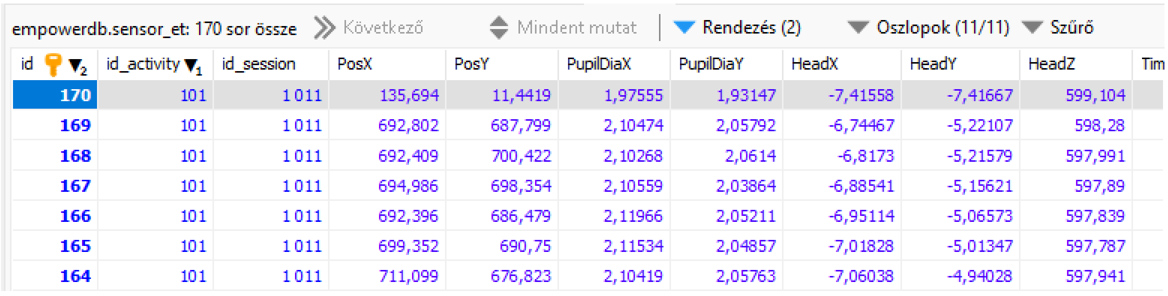


Figure 5 Implemented 5-point calibration (green: good, yellow: acceptable, red: wrong)



The recorded raw eye-tracking data is stored in the MariaDB database (Figure 6).

Figure 6 Raw gaze data stored in the MariaDB database.



empowerdb.sensor_et: 170 sor össze

id	id_activity	id_session	PosX	PosY	PupilDiaX	PupilDiaY	HeadX	HeadY	HeadZ	Tim
170	101	1 011	135,694	11,4419	1,97555	1,93147	-7,41558	-7,41667	599,104	
169	101	1 011	692,802	687,799	2,10474	2,05792	-6,74467	-5,22107	598,28	
168	101	1 011	692,409	700,422	2,10268	2,0614	-6,8173	-5,21579	597,991	
167	101	1 011	694,986	698,354	2,10559	2,03864	-6,88541	-5,15621	597,89	
166	101	1 011	692,396	686,479	2,11966	2,05211	-6,95114	-5,06573	597,839	
165	101	1 011	699,352	690,75	2,11534	2,04857	-7,01828	-5,01347	597,787	
164	101	1 011	711,099	676,823	2,10419	2,05763	-7,06038	-4,94028	597,941	

#3. Pilot Feedback and Interventions on ET Module

Two pilots were conducted in Romania and Portugal in May and June of 2023.

ROMANIAN PILOT

Place: Constanța

1. Date: May 22nd - May 26th

Feedbacks:

- 1, Hand positioning of users occludes the eye-tracker. The problem relates to the touch-based interaction method and the ET device positioned at the bottom of the screen.
- 2. The calibration experience was boring for most children. During the process, they sometimes look away from the screen or get distracted by something else. This negatively affects the collected eye-tracking data.
- 3. The standard 5-point calibration was too fast for most children. 2-3 rounds of recalibration were needed to get satisfactory calibration results.
- 4. The calibration process was repeated after every level of the game. This resulted in some children being bored/annoyed with how many times they had to calibrate, leading to worse calibration results further into the tests.
- 5. The database solution needed to split sessions for each participant adequately. This required creating a new database for each user to be able to separate the data.
- 6, Head positioning guidelines only provided changes in the background colour to assist in creating proper head positioning (greed = good, red = bad). It would also be ideal to have text guidance to help teachers understand what is wrong and what is needed for proper head positioning.
- 7, Distance requirements for the eye-tracker were challenging for some children, as their arms were shorter and could not reach the screen.
- 8, Calibration required the detection of both eyes to begin. One participant could not open one of their eyes, forcing us to skip calibration and ET data collection.

Modifications:

- 1. Placement of the ET device above the screen gives uncertain operation and is not recommended by the manufacturer. As a result, this placement is not applicable. Unique Tobii holders may be too expensive.
- 2, Additional animation during the eye gaze calibration: the calibration point pulses, the calibration is in two stages at each point, the more attention-grabbing yellow colour is used.
- 3. The speed of calibration has been slowed down.
- 4. No ET Module problem.
- 5, ET Module database integration issues were solved. This includes a fix to the database session issue.
- 6. We have modified the calibration screen and used a head pictogram to visualise the actual head position.
- 7, Hardware limitations. Special Tobii holders may solve this problem, but this is too expensive.
- 8. If it's important, then we will investigate.

PORTUGAL PILOT

Place: Lisbon

Date: June 5-16, 2023

Feedbacks:

- In the case of the Portuguese pilot, a computer mouse was used during the attention game, where eye tracking was used. As a result, the problem of covering the eye movement tracking device did not occur.
- During the head and eye calibration, no significant problems appeared; presumably, the children tested were of a different nature. Recalibration was only necessary in a few cases.
- The technical problems that arose during the pilot in Romania were resolved, so they no longer occurred.

Modifications:

- No specific intervention

Post evaluation of recorded data: summarise the raw data and the feature extracted eye parameters. The review confirmed the ET Module's functionality (number of data, eye parameters, heatmaps). The recorded data can be found in the WP3-Deliverables\Prototype Application\ Portugal_ETdata_evaluation.xlsx file.

#4. ET Module prototype

The ET Module unit package prototype can be found in:

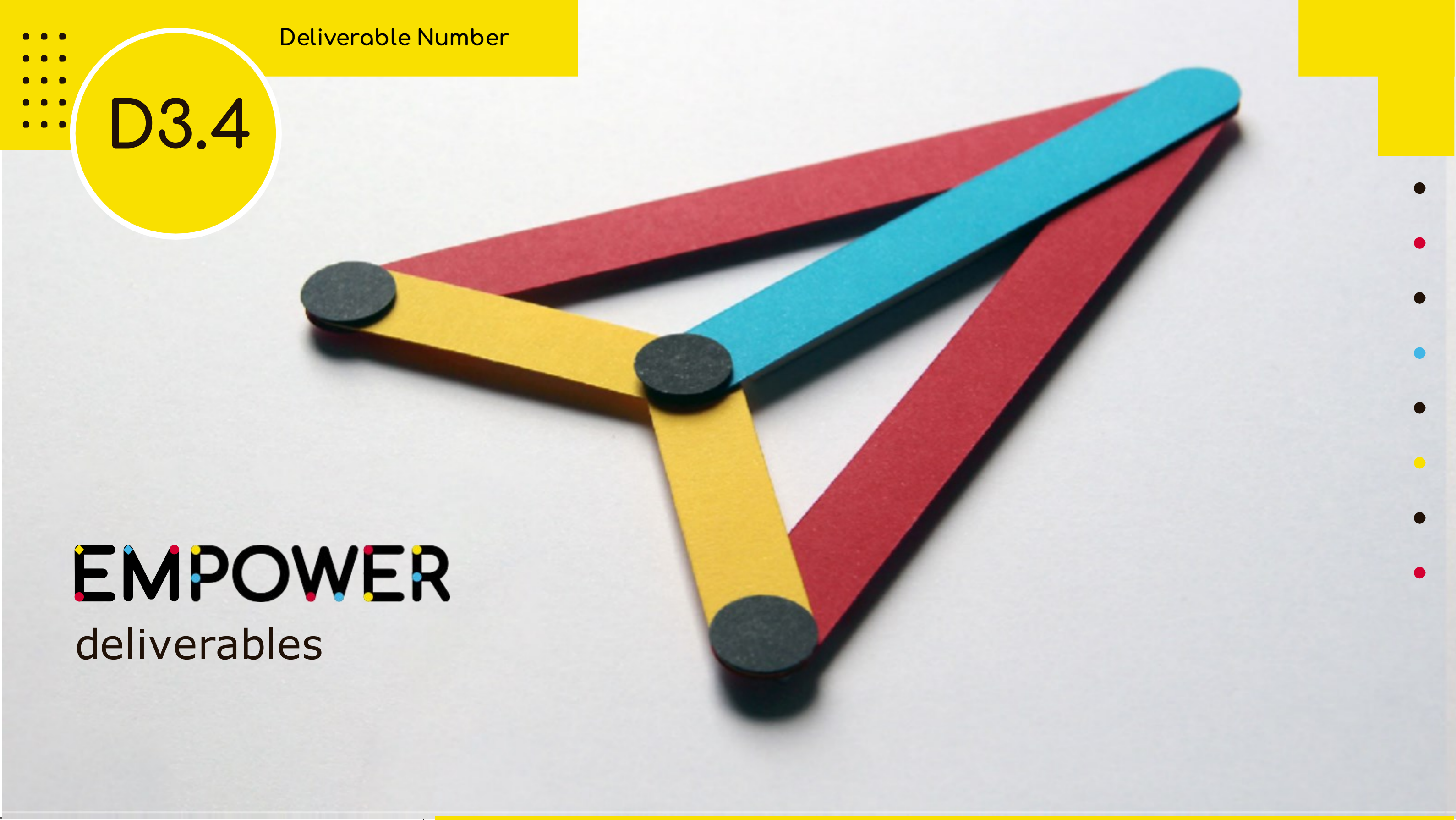
https://drive.google.com/drive/folders/1mowbgZFquFwJnY7AQ8McwPmRuPlaml8P

Tobii Pro Nano eye-tracker hardware, Windows 10 or 11 operating system, and Tobii Eye Tracker Manager must be installed and set up to use eye movement tracking during games.

ET documentation 2023.09.22.docx contains detailed information about the use of the Module.

Deliverable Number

D3.4



Deliverable name	**EMPOWER Annex to Deliverable 3.4**
Type	DEM – Demonstrator, pilot, prototype.
Dissemination level	PU - Public
Date	Month 12

System components needed to get eye-tracking data from devices to be used as an input for the platform games, as well as interfaces with WP4.

Description

WP.3

Work Package. 3

Lead Beneficiary – (UV)

IRTIC Universitat de València

EMPOWER – Unity Package documentation

The **Empower Eye tracking Tools** is a collection of features that facilitate the integration of the Tobii Pro eye tracker into Unity projects. These features include calibration of the eye tracker, recalibration, data collection, and data saving. The package is designed to be user-friendly and customizable, even for developers who are new to eye-tracking technology, even to Unity.

#1. Recommended system requirements

To run Empower Unity development package with no issues or limits, you must complete following system requirements:

- Microsoft Windows 10 Pro.
- X64 architecture with SSE2 instruction set support.
- DirectX 10, DirectX 11 capable GPUs.
- USB 2.0 Type A.
- 16:9 19" - 24" display.
- 10 GB free disk space.

#2 Download Unity and Tobii Pro Eye Tracker Manager

To run Empower Unity development package with no issues or limits, you must complete following system requirements:

I. Tobii Pro Eye Tracker Manager

1. Visit Tobii Connect center and download Tobii Pro Eye Tracker Manager for Windows 64-bit application.
2. After success installation a new window will be appear titled as Tobii Pro Eye Tracker Manager. Click on the Install new Eye Tracker button.
3. Select your Tobii Pro device from the list. We will use Tobii Pro nano in the documentation.
4. Click on Install button and grant administration permissions for the driver installer.

II. Unity Hub

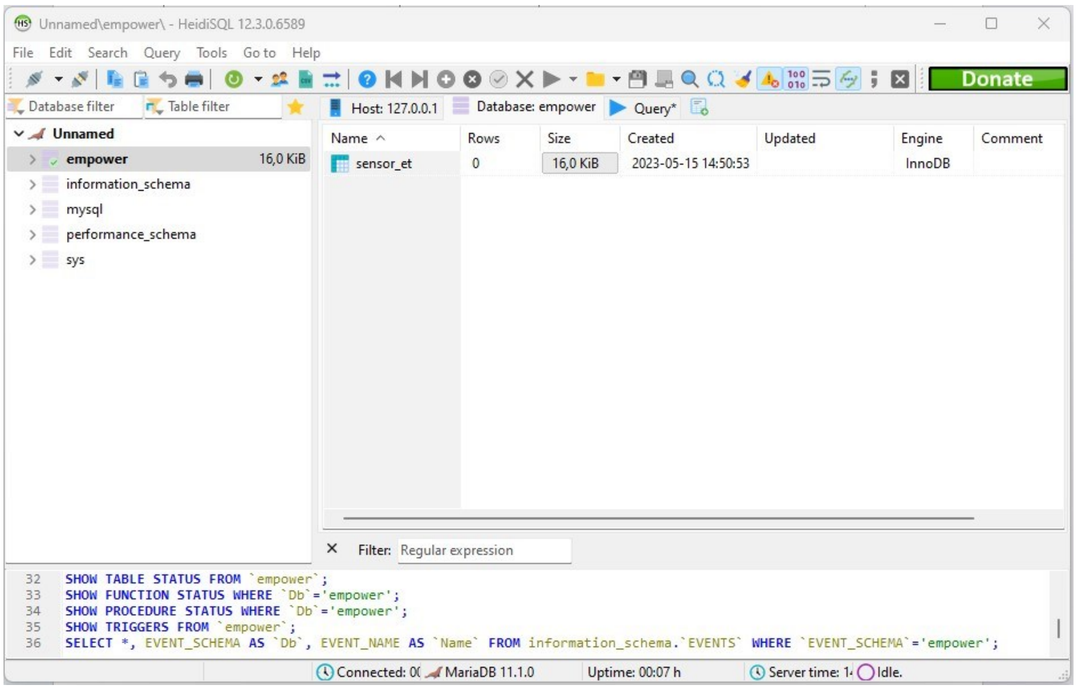
1. Register a new Unity account or login to an existing one at the Unity website.
2. Download the official Unity Hub installer from https://unity.com/download.
3. Run Unity Hub Installer, and open Unity Hub launcher after.
4. In the Unity Hub select the newest Unity Version and install it.

#3 Setup MariaDB Database

This package uses a remote SQL server to save gaze data properties. In case you must run a SQL server. In this document we will use Maria DB.
Follow these steps to install a local MariaDB database:

1. Visit MariaDB's official website and select MariaDB Server 10.11.3 version or later. Select your server's operation system, we will choose Windows for now. Make sure architecture is x86_64 and we recommend using the msi installer. Clink on the blue Download button.
2. In the installer follow the guide until the Default instance properties view. There make sure the "Enable access from remote machines for 'root' user" is checked if you are going to use local host. We also recommend check the UTF8 as default server's character set. After typing password click on Next twice, and then click Install.

The setup will also install **HeidiSQL** as well automatically. It is an easy tool to browser among your SQL data, and even modify it. Open this software and click on new button left bottom side. Give a custom name in the list. On the right-side type fill the details (hostname, username, password and port). Click on Open. If you do not have an active database, right click to the left panel, hover Create new and select Database. Give a name for your database then Click OK. The package will create necessary tables automatically.



Name	Rows	Size	Created	Updated	Engine	Comment
sensor_et	0	16,0 KiB	2023-05-15 14:50:53		InnoDB	

```
SHOW TABLE STATUS FROM `empower`;
SHOW FUNCTION STATUS WHERE `Db`='empower';
SHOW PROCEDURE STATUS WHERE `Db`='empower';
SHOW TRIGGERS FROM `empower`;
SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.`EVENTS` WHERE `EVENT_SCHEMA`='empower';
```

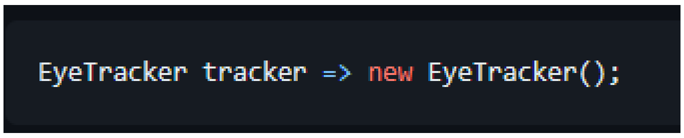
Connected: 0(MariaDB 11.1.0 Uptime: 00:07 h Server time: 1 Idle.

In the Unity script you must run tracker.CreateConnection like this:

```
//Create database connection
tracker.CreateConnection("127.0.0.1","teszt1",3306,"root","root");
```

#4 Getting started with the package

Follow the instructions provided below.
After adding the *".unitypackage"* file, create a new behavior script, or you can call the functions included in the package by creating an instance of the *EyeTracker* object using the following code in an existing behavior script attached to a game object:



```
EyeTracker tracker => new EyeTracker();
```

We will include a sample script file, named as *"ET_Test.cs"* in your Assets folder. This script includes all necessary functions such as head positioner, calibration runner, start and stop recording, change activity instance, and the focus point follower. By default, these functions are binded to hotkeys, but you can use them whenever you want.

4.1 MAIN FUNCTIONS

- **Connect() :** Use this function to connect to the eye tracker. It returns "Returns" enum. OK when connection is success else Returns.FAILED.
- **CreateConnection(host, database, port, username, password) :** Use this function to connect to your SQL Server database. It returns Returns.OK when connection is success else Returns.FAILED.
- **Disconnect() :** Use this function to disconnect from the eye tracker. **NOTE:** Only use this function when you quit from your Unity game. It returns Returns.OK when disconnect is success else Returns.FAILED.
- **StartGazeDataCollection() :** This function starts the gaze data collection. It returns Returns enum. OK when the package is collecting data successfully, else Returns.FAILED
- **StopGazeDataCollection() :** This function stops the gaze data collection. It returns Returns enum. OK when package is not collecting data anymore, else Returns.FAILED.
- **SetActivity(UserID, ActivityID, SessionID) :** Calling this function you can set the current user id, game id and session id. It Returns Returns enum. OK when activity change is success, else Returns FAILED
- **GetCurrentActivityID() :** This function returns current activity ID.
- **GetLatestGazeData() :** You can view the latest gaze data. Returns a Data object.

- **GetLatestFixationData() :** You can view the latest fixation data. Returns the Fixation frequency, Average fixation time, and Total fixation time.
- **GetHeadPosValidity() :** This function returns back is the Tobii seeing the subject's eyes and head correctly.
- **GetPreviousInvalidGazeDataNumber() :** This function returns back how many gaze data is invalid in last 30 seconds.
- **Save() :** This function will save all gaze data that have been collected during collection. It returns Returns enum. OK when all data has been written to the database, else Returns.FAILED.

NOTE: The directory path is "**Empower-ET"** in user's document folder. If you get any FAILED return, you can check eyetracker.log to check the failure reason.

4.2 MAIN PROPERTIES

5 **RecalibrationOrApply.CalibrationAccepted:** Boolean, it saves whether user accepted the calibration.
6 **GazeTrailFollwer.GazeTrailState:** Boolean, it saves is Focus Point follower enabled.
7 **EyeTracker.IsSavingRunning** : Boolean, it saves when gaze data is writing out to csv and into database.
8 **tracker.State** : It returns back is eye tracker device connected to the Unity. See EyeTrackerState enum.
9 **tracker.SightSate** : This is a SightState enumerator which saves which part of the screen the subject watching. Values are: SightState.UP or SightState.DOWN else user is not watching the display.
10 **tracker**.RecordingState = This is a GazeDataProcessorState enumerator which returns back whether recording is started and collecting the gaze data.

4.3 ENUM RETURNS

HeadPosValidity:
4.4 NOT_VALID: The Eye Tracker device can't see the subject's head and eyes at all.
4.5 MARGIN: The device is can see his eyes and head but he is in near to margin.
4.6 VALID: The subject is sitting in correct place.
SightState:
4.7 UP: The subject is watching the upper part of the screen (height/2>)
4.8 DOWN: The subject is watching the bottom part of the screen (height/2<)
4.9 AWAY: The subject is not watching the display or his eye is closed.
EyeTrackerState:
4.10 LISTENING: Eye Tracker connected and listening.
4.11 DISCONNECTED: Disconnected or there is no eye tracker device.
4.12 ABORTED: Eye Tracker operation is aborted.

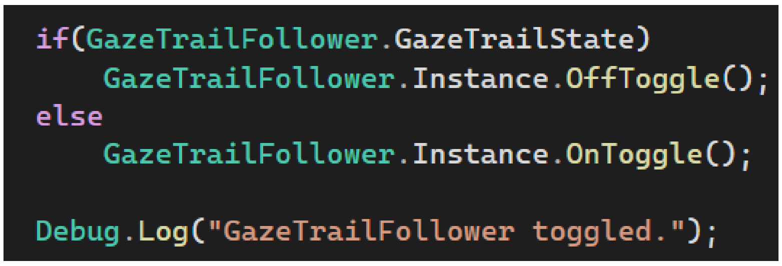
4.13 IN_CALIBRATION: Eye Tracker is in active calibration.
4.14 OPERATION_CANCELLED_BY_USER: Eye Tracker operation is cancelled.

Returns:

4.15 OK: The function completed without any error.
4.16 FAILED: The function could not finish the operation due to failure.
4.17 CANCELLED: Function cancelled before it was executed.
4.18 NO_USER_SET: No test subject selected to execute current function.
4.19 USER_SET_PROCESS. Function that connected with subject is done.
4.20 USER_SET_ERROR: There was an error in function that connected with subject.

4.4 GAZE TRAIL FOLLOWER

You can visualize the current gaze point on any object which have a mesh collider. To toggle on/off this feature you can modify in Unity settings at GazeTrail prefab, default hotkey is **"P"**. Besides you can enable or disable the Focus Point follower inside the code. Implementation:

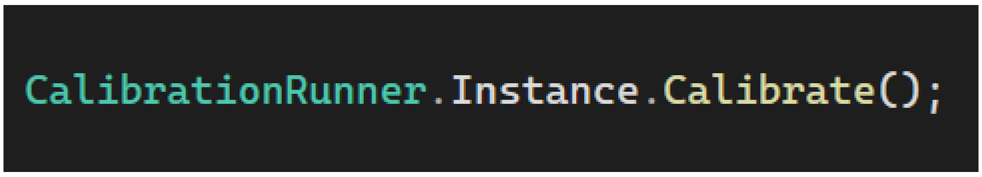


```
if(GazeTrailFollower.GazeTrailState)
    GazeTrailFollower.Instance.OffToggle();
else
    GazeTrailFollower.Instance.OnToggle();

Debug.Log("GazeTrailFollower toggled.");
```

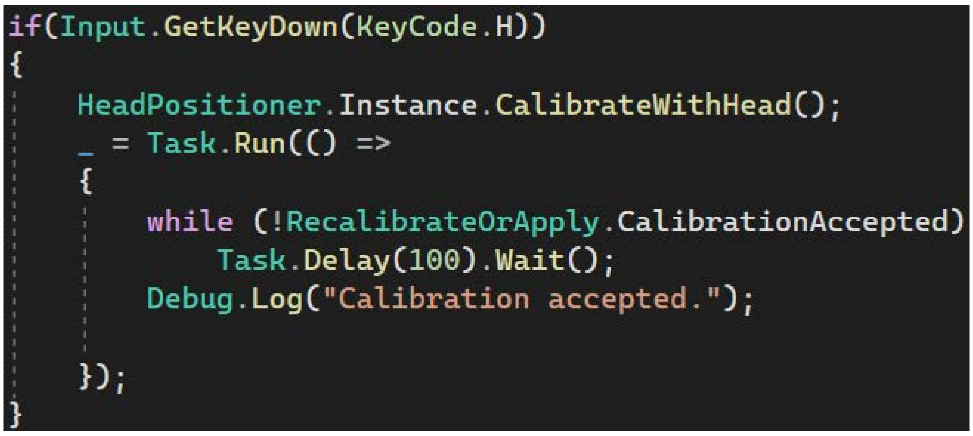
4.5 CALIBRATION

You can calibrate the Tobii Pro device manually using the **"C"** key on your keyboard, but you can implement your own solutions. (**Note:** Using the **"C"** key only calibrates, not shows the HeadPosition prefab).
You can also implement only the calibration (without the HeadPositioner), like this:



```
CalibrationRunner.Instance.Calibrate();
```

This method will show the HeadPositioner where you can position your head, then you can click on Start Calibration button to proceed:



```
if(Input.GetKeyDown(KeyCode.H))
{
    HeadPositioner.Instance.CalibrateWithHead();
    _ = Task.Run(() =>
    {
        while (!RecalibrateOrApply.CalibrationAccepted)
            Task.Delay(100).Wait();
        Debug.Log("Calibration accepted.");

    });
}
```

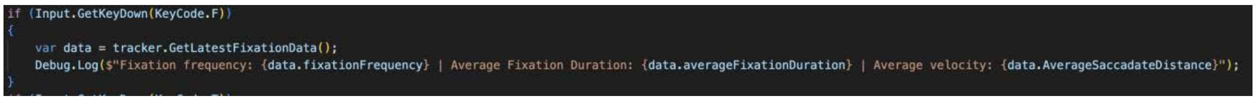
Reminder: **“H”** key will show Head positioner and calibration runner as well (if you click on calibrate button), **“C”** only runs the calibration.

4.6 REAL TIME GAZE STATISTICS

With **GetLatestStatistics()** method from **Tracker** class you can request some statistics about Subject’s gaze movement during recording. It calculates from gaza data which are collected in previous 30 seconds. This is a tuple, returning these variables:

- **fixationCount int:** It returns how much fixation collected in that time.
- **averageFixationDuration double:** it returns the average of the collected fixation durations.
- **fixationFrequency double:** It returns how much fixation creates per second in average.
- **averageSaccadeDistance double:** It returns the average of the saccades between two fixation points.
- **Timestamp long:** This is the unix time (in UTC zone) when the statistics created.

Here is an example:



```
if (Input.GetKeyDown(KeyCode.F))
{
    var data = tracker.GetLatestFixationData();
    Debug.Log($"Fixation frequency: {data.fixationFrequency} | Average Fixation Duration: {data.averageFixationDuration} | Average velocity: {data.AverageSaccadateDistance}");
}
```

NOTE: In default you can request these variables by pressing **F** button, but you can modify it any time in the **ET_TEST** class or in your script file.

4.7 PROCESS

1. Tracker.Connect();
2. Tracker.CreateConnection();

3. Tracker.SetActivity();
4. Tracker.StartGazeDataCollection();
5. Tracker.StopGazeDataCollection();
6. Tracker.Save();
7. Go to 3. in case of new activity, new session, new user otherwise quit the game.

PS: You can call the SetActivity between 4. and 5. to change the current activity.

#5 Models

5.1 DATA :

Properties

Type	Name	Description
double	X	The X-coordinate in pixel of the current gazepoint.
double	Y	The Y-coordinate in pixel of the current gazepoint.
int	SubjectID	Game User ID. Set this with SetActivity(UserID, ActivityID, SessionID)
int	SessionID	Game Session ID. Set this with SetActivity(UserID, ActivityID, SessionID)
int	GameID	Game Activity ID. Set this with SetActivity(UserID, ActivityID, SessionID)
float	LeftPupilDia	The left pupil diameter in pixels.
float	RightPupilDia	The right pupil diameter in pixels.
Int64	Timestamp	The time when the gazepoint sampled.
double	HeadPosX	The X position of the head in the real world.
double	HeadPosY	The Y position of the head in the real world.
double	HeadPosZ	The Z position of the head in the real world.

5.2 MARIADB TABLE

Table

Name	Type	Schema	Description
Sensor _ET		DROP TABLE IF EXISTS sensor_et; CREATE TABLE sensor_et(id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, id_activity INT, id_session INT NOT NULL, PosX FLOAT, PosY FLOAT, PupilDiaX FLOAT, PupilDiaY FLOAT, HeadX FLOAT, HeadY FLOAT, HeadZ FLOAT, FOREIGN KEY (id_activity) REFERENCES activity(id) ON UPDATE NO ACTION ON DELETE SET NULL, FOREIGN KEY (id_session) REFERENCES session(id) ON UPDATE CASCADE ON DELETE CASCADE, Timestamp BIGINT)	sql command to create sensor_et table
id	INTEGER	"id" INTEGER NOT NULL AUTO_INCREMENT	Autoincrement primary key
id_activity	INT	"id_activity" INT FOREIGN KEY (id_activity) REFERENCES activity(id) ON UPDATE NO ACTION ON DELETE SET NULL	Activity ID from game
id_session	INT	" id_session" INT FOREIGN KEY (id_session) REFERENCES session(id) ON UPDATE CASCADE ON DELETE CASCADE	Session ID from game (same for one user)

PosX	FLOAT	"PosX" FLOAT		Gaze x position ont he screen
PosY	FLOAT	"PosY" FLOAT		Gaze y position ont he screen
PupilDiaL	FLOAT	"PupilDiaX"	FLOAT	Left pupil diamater in pixels
PupilDiaR	FLOAT	"PupilDiaY"	FLOAT	Rigth pupil diameter in pixels
HeadX	FLOAT	"HeadX"	FLOAT	Head x position in mm (horizontal)
HeadY	FLOAT	"HeadY"	FLOAT	Head y position in mm (vertical)
HeadZ	FLOAT	"HeadZ"	FLOAT	Head z position in mm (distance from the screen)
Timestamp	BIGINT	"Timestamp"	BIGINT	UNIX UTC time in milliseconds