Design Specification

Elias William
Rasmus Olofsson
Malva Eveborn
Oskar Ramsberg
Mohammad Alasmi
Adam Roos

December 12, 2023

Version 0.5

Status

Reviewed	Malva Eveborn	2023-10-13
Approved		

Project Identity

Group E-mail: eliwi204@student.liu.se

Homepage: http://www.isy.liu.se/tsrt10/group

Orderer: Martin Skoglund, ISY/Eriksholm Research Centre
Phone: +46 13 28 18 90
E-mail: martin.skoglund@liu.se

Customer: Sergi Rotger Griful, Oticon
Phone: -
E-mail: segr@eriksholm.com

Supervisor: Johanna Wilroth, ISY
Phone: +46 13-28 28 05
E-mail: johanna.wilroth@liu.se

Course Responsible: Daniel Axehill & Gustaf Hendeby
Phone: +46 013-28 40 42 & +46 13-28 58 15
E-mail: daniel.axehill@liu.se & gustaf.hendeby@liu.se

Participants of the group

Name	**Responsible**	**E-mail**
Elias William	Project Leader (PL)	eliwi204@student.liu.se
Rasmus Olofsson	Responsible for software (SW)	rosol028@student.liu.se
Malva Eveborn	Responsible for documentation (DOC)	malev556@student.liu.se
Oskar Ramsberg	Responsible for design (DES)	oskra262@student.liu.se
Mohammad Alasmi	Responsible for hardware (HW)	mohal555@student.liu.se
Adam Roos	Responsible for testing (TEST)	adaro336@student.liu.se

CONTENTS

Oticon
eliwi204@student.liu.se

DOCUMENT HISTORY

Version	Date	Changes made	Sign	Reviewer
0.1	2023-09-20	First draft.	All	EW
0.2	2023-10-04	Second draft.	All	EW
0.3	2023-10-09	Third draft.	All	EW
0.4	2023-10-13	Fourth draft.	All	ME
0.5	2023-11-09	Fifth draft.	EW, OR	MA

1 INTRODUCTION

Communicating in noisy environments like restaurants, festivals and exhibitions remain troublesome for people suffering of impaired hearing. Traditionally, hearing aid devices have amplified all surrounding sounds of the user, making the users avoid these types of rowdy environments. One way to improve the experience of hearing aid users would be to understand the environment and the intent of the user, enabling focused sound amplification combined with noise reduction.

This project aims to investigate and develop methods for using information about the environment to be able to extract useful sound to the user. This will be done with the assistance of a pair of hearing aids and a pair of wearable glasses providing sensors like camera and eye-tracker.

In collaboration with the hearing aid company Oticon and Linköpings University, we aim on this years' CDIO-project to further develop the research field to help people suffering from hearing problems with the focus on real-time implementation. The project will be carried out using the LIPS-model and this document will describe the design of the product.

1.1 Definition of terms

Table 1: Definition of terms.

Term	Meaning
G3	Tobii Pro Glasses 3.
IMU	Inertial Measurement Unit.
SNR	Signal to Noise Ratio.
CDIO	Conceive Design Implement Operate.
GUI	Graphical User Interface.
LIPS	Project model.
DP	Decision Point, a decided time when specific implementation has to be made.
TDOA	Time Difference of Arrival.
DOA	Direction of Arrival.
SLAM	Simultaneous Localization and Mapping.
HA	Hearing Aid.
HRTF	Head-Related Transfer Function.
HRIR	Head-Related Impulse Response.

2 SYSTEM OVERVIEW

In this section a general overview of the system and all its parts will be presented. Figure 1 shows the flow chart of the complete system.

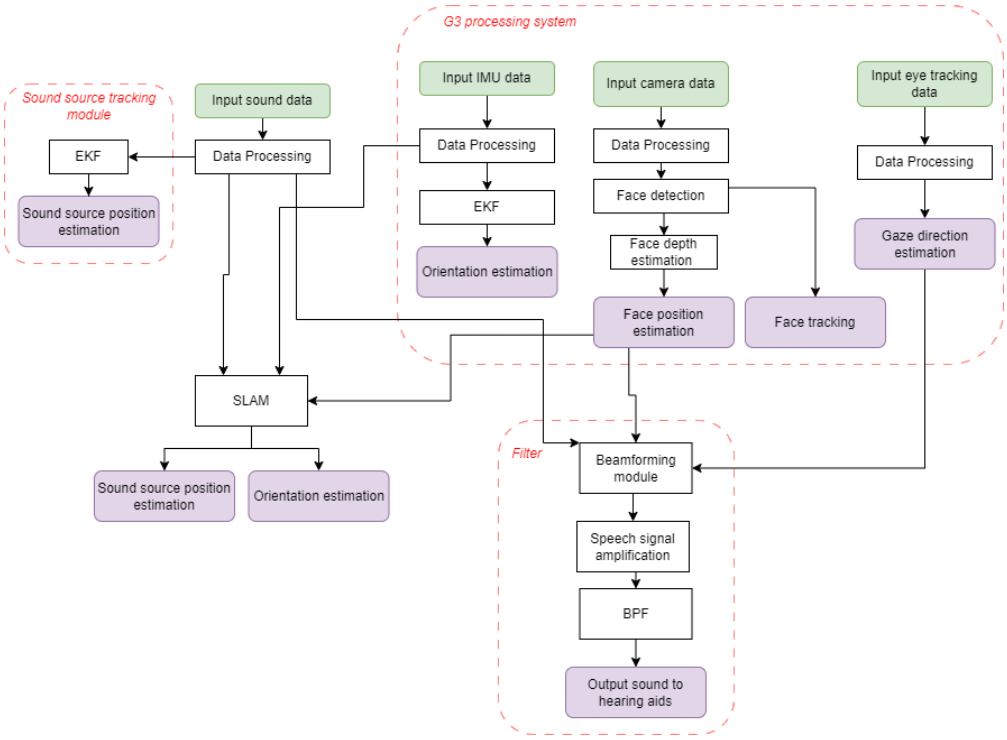

Figure 1: The flowchart of the entire system.

2.1 Hardware

In this section an brief explanation of all the hardware that is used will be presented. The diagram of the battery circuit for the hearing aids can be seen in Figure 2.

2.1.1 *Tobii Pro Glasses G3*

The latest generation of Tobii Pro Glasses is used for collecting data of the surrounding environment of the user and can be seen in Figure 3. Sensors such as front facing camera, eye-tracker, microphones, IMU and magnetometer can provide additional information for interpreting the estimation of the user's surrounding environment. This new generation improves from the previous with all around better and more precise sensors and also extends the sensor suite with a magnetometer. The glasses are powered through a battery pack that also has a slot for a SD-card for recording data. The package can be connected by WIFI or internet cable to a computer [1].

2.1.2 *Prototype hearing aid devices*

The hearing aid devices can be seen in Figure 4. They consist of two microphones each and one in-ear earbud. The microphones connect to an amplifier with an RCA-connector (red and white). The earbud receives audio from a sound-card (black connector).

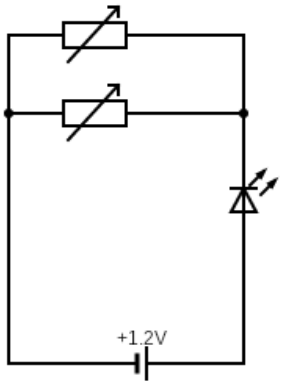

Figure 2: Hearing aids battery circuit diagram.

Figure 3: Tobii Pro Glasses G3. Image reproduced by permission from Tobii AB [2].

2.1.3 *Amplifier*

A phono pre-amplifier is going to be used to amplify the sound from the hearing aid device microphones. The pre-amplifiers are of model type "NANO-LP1". They consist av 2 input ports and 2 output ports. There are 2 pieces pre-amplifiers used each one for each earpiece.

2.1.4 *Sound-card*

An audio interference is going to be used in the project to record the incoming sound from the microphones. The audio card is of the model "Maya44" which has 4-inputs and 4-outputs. The sound card has a USB connection to connect to a computer device where the input signals can be processed and measured.

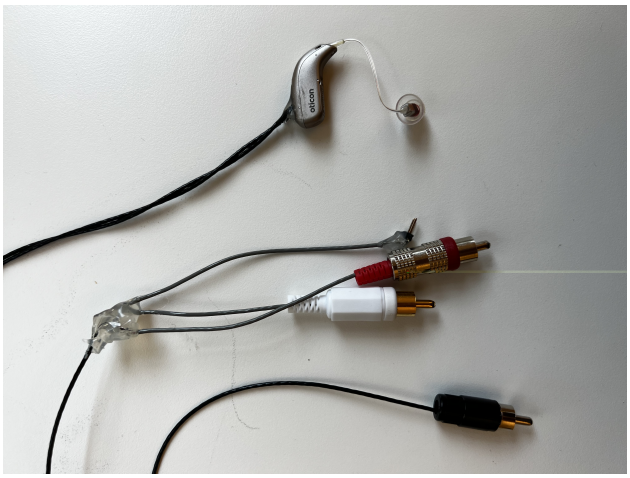

Figure 4: Hearing aid prototype device.

3 SOUND SOURCE TRACKING MODULE

This section will focus on the methods for tracking the sound source from the hearing aid prototypes. It will assume that the HA is stationary. The parameters that will be estimated for sound source i is $x^{(i)} = [p^{(i)}, v^{(i)}]$, where $p^{(i)}$ is the position och $v^{(i)}$ is the velocity in the global 3D Cartesian coordinate frame.

3.1 Localization

This section will present the methods used to localize a sound source, using data only received by the hearing aids microphones. In order to localize the sound source with the hearing aids, a sensor model is needed to be implemented. The sensor models that will be used in this project is time difference of arrival (TDOA) and direction of arrival (DOA). A motion model and a static model will also be presented in this section. A localization algorithm using head-related transfer functions is also presented.

3.1.1 *Estimating time delay*

The cross-correlation between two continuous signals $f(t)$ and $g(t)$ are defined as

$$(f \star g)(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} g(t+\tau) dt, \tag{1}$$

where τ is called lag. Similarly, for discrete signals $f[m]$ and $g[m]$, the cross-correlation is defined as

$$(f \star g)[n] = \sum_{m=-\infty}^{\infty} \overline{f[m]} g[m+n], \tag{2}$$

where n is the lag. The parameter τ will be used for the lag in the rest of the document for clarity. The cross-correlation is a measure on how similar two signals are and the lag τ is the time difference of arrival of the two signals f and g.

3.1.2 *Sensor Model TDOA*

To calculate the direction of a sound source relative to the HA, the TDOA is estimated. To do this it is assumed that the sound source is close enough to the user so when the sound is detected in microphone M2 with a delay of τ compare to M1, see Figure 5. For ease of use the TDOA is returned as a distance instead of of a time difference. This is done with following equation

$$TDOA = \tau \cdot c \tag{3}$$

where τ is the time difference between when the sound hits M1 and M2 and c is the speed of sound $\approx 343 m/s$. See Section 3.1.1 for how τ is estimated. M1 and M2 include two microphones each that are assumed to be located at the same position. This results in four parings between M1 and M2 resulting in four independent measurement's of the TDOA.

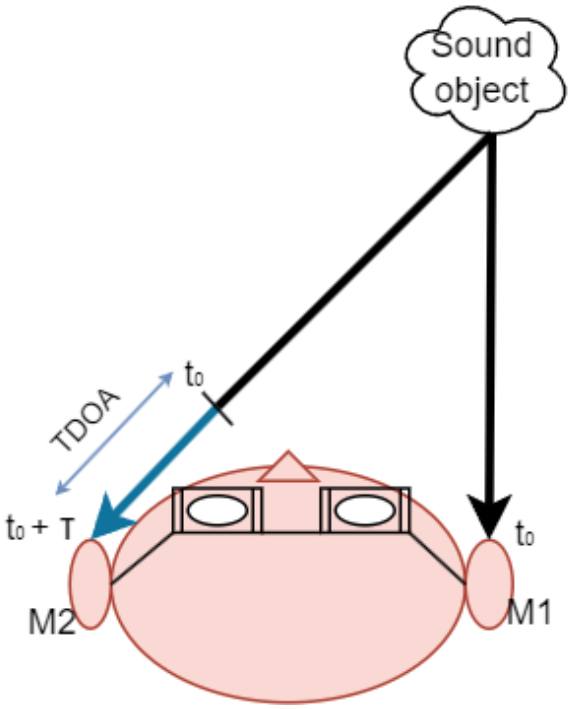

Figure 5: Microphone network to illustrate TDOA.

The TDOA model can be expressed using the equation

$$y_{tdoa}^{(i)} = h_{tdoa}^{(i)}(x_1, x_2) + e_{tdoa}^{(i)}, \tag{4}$$

where

$$\begin{aligned} h_{tdoa}^{(i)}(x_1, x_2) = &\sqrt{(x_1 - m_1^{(i)})^2 + (x_2 - m_2^{(i)})^2} \\ &- \sqrt{(x_1 - m_1^{(i-1)})^2 + (x_2 - m_2^{(i-1)})^2}, \end{aligned} \tag{5}$$

eliwi204@student.liu.se

x_1, x_2 is the position of the sound source, $m_1^{(i)}$, $m_2^{(i)}$, $m_1^{(i-1)}$, $m_2^{(i-1)}$ is the positions of the microphones i and $i-1$, where $i = 1$ and 2. The positions are in Cartesian coordinates.

$$e_{tdoa}^{(i)} \sim \mathcal{N}(0,\, R_{tdoa}^{(i)})$$

is the measurement error. The covariance matrix $R_{tdoa}^{(i)}$ for TDOA is calculated by taking the covariance of $e_{toa}^{(i)} - e_{toa}^{(i-1)}$, which was measured in section 3.1.4. The total covariance matrix R_{tdoa} for $i = 1$ and 2 can be described using the matrix

$$R_{tdoa} = \begin{pmatrix} R_{toa}^{(0)} + R_{toa}^{(1)} & -R_{toa}^{(1)} & 0 \\ -R_{toa}^{(1)} & R_{toa}^{(1)} + R_{toa}^{(2)} & -R_{toa}^{(2)} \\ 0 & -R_{toa}^{(2)} & R_{toa}^{(2)} + R_{toa}^{(3)} \end{pmatrix} \tag{6}$$

where $R_{toa}^{(i)}$ is the covariance or variance of the noise $e_{toa}^{(i)}$ in microphone i for the TOA (time of arrival) measurements.

3.1.3 *Sensor Model DOA*

To find the DOA a far field assumption is made. This ensures that the DOA for each microphone is the same and so the angle ϕ can be calculated with following equation

$$\phi = \overset{+}{(-)} \arccos \frac{TDOA}{d}, \tag{7}$$

where ϕ is assumed to be the DOA and $TDOA$ is the difference in distance between the microphones and the sound source and is calculated using equation 3. See figure 6 for a more detailed illustration. Note that the angle is not explicit. The frontal or backward direction will be determined by the two microphones on the same hearing aid.

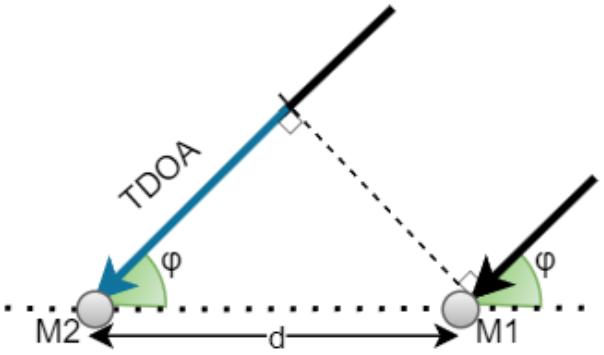

Figure 6: Microphone network to illustrate DOA, assuming far field.

The DOA model can be expressed using the equation

$$\phi^{(i)} = \alpha + arctan2(x_1 - m_1^{(i)}, x_2 - m_2^{(i)}) + e_{doa}^{(i)}, \tag{8}$$

where α is the yaw angle of the glasses in relation to the 2D Cartesian coordinate plane, x_1, x_2 is the position of the sound source and $m_1^{(i)}$, $m_2^{(i)}$ is the postion of the microphones i, where $i = 1$ and 2. The positions are in Cartesian coordinates in the horizontal plane.

$$e_{tdoa}^{(i)} \sim \mathcal{N}(0, R_{doa}^{(i)})$$

is the measurement error of DOA, which is assumed to be zero-mean Gaussian noise with the total covariance matrix R_{doa} that can be described as

$$R_{doa} = \sigma^2 I. \tag{9}$$

3.1.4 *TDOA Sensor Calibration*

To estimate variance and counteract bias of the microphones, a few measurements and sensor calibrations is needed. The bias can for example be error in the sound card. To do this a calibration data set is first recorded where all microphones are equally spaced from the known sound source. The values from the calibration data set is then used to find the measurement errors $e^{(i)}$ for TOA for a given microphone i. By first multiplying the measured difference in time with the speed of sound, (340 m/s), an estimate of the distance between the sound source and the microphone is therefor used instead. The mean value for all microphones at instance k was then calculated as

$$y_{\mu,k} = \frac{1}{N} \sum_{i=0}^{N} y_k^{(i)}, \tag{10}$$

where y_μ is the mean distance value of all measurements $y_k^{(i)}$, measured from microphone i, at instance k and N is the number of microphones. The measurement error for a given microphone i for all time instants k was thereafter calculated by subtracting the mean value from all the values according to

$$e_k^{(i)} = y_k^{(i)} - y_{\mu,k}, \tag{11}$$

where $y_k^{(i)}$ is the measured distance value for microphone i and $e_k^{(i)}$ is the measurement errors for microphone i. The mean and variance for all errors $e^{(i)}$ was thereafter calculated. The measurement bias for all microphones are thereafter removed from the measurements to calibrate the sensor by $y^{(i)} - E[e^{(i)}]$, for each new distance measurement $y^{(i)}$.

3.1.5 *Static Motion Model*

When the sound source is assumed to be static, the constant position motion model will be used. The model can be described as

$$x_{k+1} = x_k + T v_k, \tag{12}$$

where v_k is some unknown additive noise. The state $x_k = [p]^T$, where p is the position in 2D of the sound source. T is the sample time.

3.1.6 *CV Motion Model*

The constant velocity (CV) motion model will be used when the sound source is assumed to be moving. The model can be described as

$$x_{k+1} = \begin{pmatrix} I_3 & TI_3 \\ 0_3 & I_3 \end{pmatrix} x_k + \begin{pmatrix} \frac{T^2}{2} I_3 \\ TI_3 \end{pmatrix} v_k, \tag{13}$$

where v_k is some unknown additive noise. The state $x_k = [p, v]^T$, where p is the position in 2D and v is the velocity in 2D. T is the sample time.

3.1.7 *EKF*

The extended Kalman filter will be used to update the state vector in order to track the sound source. The measurement model will be used in combination with either the CV model or the static model.

3.2 Head-Related Transfer Functions

As sound waves propagates through the human body they are effected and change their characteristics depending on which body part they pass though. Head-Related Transfer Functions (HRTF) are mathematical models that specify the relation between sound pressures the listener can hear at their two ear drums. The HRTF model can be used to determine from which direction a sound reaches a listeners' ear and therefore help to localize the origin of the sound.

An array of m microphones has a center at a point P. For microphones mounted at the ears, P would be a point in the center of the head. A sound originates coming from the azimuth angle θ and elevation angle ϕ relative to P will then have some different characteristics. A digital recording I_j, where j is the numbering of the microphone, can be described with

$$I_j = O * F_j^{(\theta,\phi)}, \tag{14}$$

where O is the sound that would arrive at P if there was no head present, $*$ is the convolution operator and $F_j^{(\theta,\phi)}$ is the head-related impulse response (HRIR) for microphone j when the sound originates from the direction (θ,ϕ). The HRIR and the HRTF are the same thing but represent the characteristics in a slightly different way. The HRIR is in the time domain and HRTF is in the frequency domain.

In order to find (θ,ϕ), the input from each microphone is convolved with the HRIR associated with the opposite microphone. Note that this requires a placement of the microphones in a symmetric way, which comes naturally with the use of hearing aids. If only two microphones are used, one at each ear then,

$$I_1 * F_2^{(\theta,\phi)} = O * F_1^{(\theta,\phi)} * F_2^{(\theta,\phi)} \tag{15}$$

and

$$I_2 * F_1^{(\theta,\phi)} = O * F_2^{(\theta,\phi)} * F_1^{(\theta,\phi)} = O * F_1^{(\theta,\phi)} * F_2^{(\theta,\phi)}. \tag{16}$$

If the correct location the operation will give the correct result for both microphones inputs. The sound source origin is therefore chosen to be the (θ,ϕ) that minimizes the difference between the two equations 15 and 16. [3]

4 TOBII PRO GLASSES G3

4.1 Orientation

The global frame G is the earth's fix right-hand coordinates where the z-axis is parallel to the direction of gravitational acceleration. The body frame B is the local coordinates of the G3 glasses in the testing environment.

In order for the coordinate system to be rotated from the global to the local coordinates. The global frame is defined with unit quaternion and the orientation with a vector $\mathrm{q} = [q_0\ q_1\ q_2\ q_3]^T$. Where q_0 is scalar and $q_{\mathbf{1,2,3}}$ are complex units. The rotational matrix R^{GB} is computed as such:

$$R^{GB}(q) = \begin{bmatrix} 2q_0^2 - 1 + 2q_1^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & 2q_0^2 - 1 + 2q_2^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & 2q_0^2 - 1 + 2q_3^2 \end{bmatrix} \tag{17}$$

The derivative is calculated as follows to take the quaternion (q) as a state in the motion model. The S matrix can either be defined with respect to ω or quaternion q. The angular velocities which are pitch, roll and yaw are defined as ω_x, ω_y and ω_z:

$$\dot{q} = \frac{1}{2}S(\omega)q = \begin{pmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{pmatrix} \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} \tag{18}$$

$$\dot{q} = \frac{1}{2}\bar{S}(q)\omega = \begin{pmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \tag{19}$$

Discretizing of quaternion and normalising as follows:

$$q_{k+1} = q_k + \frac{T_s}{2}\bar{S}(q_k)\omega_k \tag{20}$$

$$q_{k+1} = \frac{q_{k+1}}{||q_{k+1}||} \tag{21}$$

In time discrete the motion model for the IMU is as such:

$$\underbrace{\begin{bmatrix} q_{k+1} \\ \omega_{k+1} \\ b^{acc}_{k+1} \\ b^{gyro}_{k+1} \\ b^{mag}_{k+1} \end{bmatrix}}_{x_{k+1}} = \underbrace{\begin{bmatrix} 1 & \frac{T_s}{2}\bar{S}(q_k) & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix}}_{F} \underbrace{\begin{bmatrix} q_k \\ \omega_k \\ b^{acc}_k \\ b^{gyro}_k \\ b^{mag}_k \end{bmatrix}}_{x_k} + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{T_s^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} w^q_k \\ w^\omega_k \\ w^{bias,acc}_k \\ w^{bias,gyro}_k \\ w^{bias,mag}_k \end{bmatrix}}_{w} \tag{22}$$

4.1.1 *Accelerometer*

The accelerometer bias is calculated by taking the mean of the measurement when the IMU is in at still position to ensure that the calibration is plausible. The bias will thus almost be equal to the local gravity vector as follows:

Bias:

$$b^{acc} = \begin{bmatrix} b^{acc}_x \\ b^{acc}_y \\ b^{acc}_z \end{bmatrix} = \begin{bmatrix} \mu^{acc}_x \\ \mu^{acc}_y \\ \mu^{acc}_z \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ -9.82 \end{bmatrix} \tag{23}$$

The sensor model for the accelerometer is computed by taking the acceleration as a state.

Sensor model:

$$y_t^{acc} = R^{GB}(q) + b^{acc} + e^{acc}, e^{acc} \sim N(0, R^{acc}), R^{acc} = \begin{bmatrix} (\sigma_x^{acc})^2 & 0 & 0 \\ 0 & (\sigma_y^{acc})^2 & 0 \\ 0 & 0 & (\sigma_z^{acc})^2 \end{bmatrix} \tag{24}$$

4.1.2 *Gyroscope*

The gyroscope bias is calculated by taking the mean of the measurement when the IMU is in at still position to ensure that the calibration is plausible.

Calculating Bias:

$$b^{gryo} = \begin{bmatrix} b_x^{gyro} \\ b_y^{gyro} \\ b_z^{gyro} \end{bmatrix} = \begin{bmatrix} \mu_x^{gyro} \\ \mu_y^{gyro} \\ \mu_z^{gyro} \end{bmatrix} \tag{25}$$

The sensor model for the gyroscope is computed by taking the angular velocity as a state. Sensor model:

$$y_t^{gyro} = R^{GB}\omega + b^{gyro} + e^{gyro}, e^{gyro} \sim N(0, R^{gyro}), R^{gyro} = \begin{bmatrix} (\sigma_x^{gryo})^2 & 0 & 0 \\ 0 & (\sigma_y^{gyro})^2 & 0 \\ 0 & 0 & (\sigma_z^{gyro})^2 \end{bmatrix} \tag{26}$$

4.1.3 *Magnetometer*

The magnetometer bias is calculated by taking the mean of the measurement when the IMU is in at still position to ensure that the calibration is plausible. While deducting the magnetic field in Linköping from the x and y biases. The Navigation toolbox in Matlab is used to do calibration.

Calculating Bias:

$$B_r^0 = 49\mu T \tag{27}$$

$$B_\theta^0 = 16\mu T \tag{28}$$

Outlier:

$$(\sqrt{(B_r^0)^2 + (B_\theta^0)^2} - 10) < norm\left[\mu_x^{mag}\mu_y^{mag}\mu_z^{mag}\right] < (\sqrt{(B_r^0)^2 + (B_\theta^0)^2} + 10) \tag{29}$$

The sensor model for the magnetometer is computed by taking the magnetic field and the acceleration as states. Sensor model:

$$y_t^{mag} = R^{GB}m^G + b^{mag} + e^{mag}, e^{mag} \sim N(0, R^{mag}), R^{mag} = \begin{bmatrix} (\sigma_x^{mag})^2 & 0 & 0 \\ 0 & (\sigma_y^{mag})^2 & 0 \\ 0 & 0 & (\sigma_z^{mag})^2 \end{bmatrix} \tag{30}$$

where the local magnetic field vector is defined as

$$m^G = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} \tag{31}$$

where each component in the vector represent the local magnetic field in respectively direction.

4.2 Face detection

The faces in a frame will be detected using an algorithm called Haar cascades. This algorithm is already implemented with pre-trained weights in the library open-cv for python. The Haar cascades algorithm is a classifier which extracts so called Haar features on different parts of the image and classifies if it is a face in this region or not. An illustration of the filters used for the feature extraction can be seen in Figure 7.

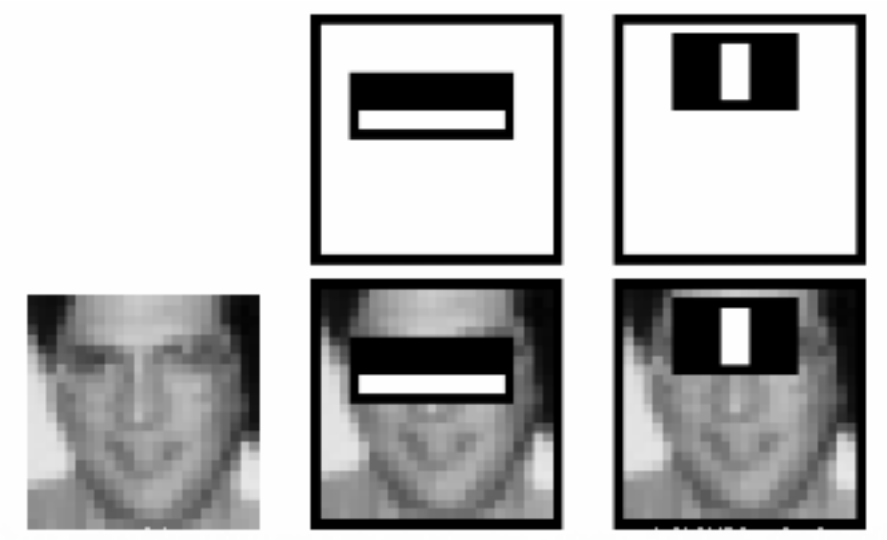

Figure 7: Haar-cascade algorithm [4].

This will result in a set of detections with the format below:

$$d_i^t = (u^{face}, v^{face}, w_p^{face}, h_p^{face}) \tag{32}$$

where t is the frame number, i is the detection number (in frame t), u^{face} and v^{face} is the center in pixels of the detection bounding box in the image plane , w_p^{face} and h_p^{face} is the width and height of the bounding box in pixels.

4.3 Face tracking

Varied approaches to face tracking will be used, but some commonalities apply to all. The method used in this project will be a so called multi-object tracking by detection method [5]. This method divides the problem in two parts, the first one detection and the second one tracking. The detection of the faces has already been explained in Section 4.2 and the tracking part is left. The output of this module will be a set of tracklets. A tracklet T_j is a set of detections across different frames [6].

$$Y_{track} = \{T_1, T_2, T_3, ...T_N\} \tag{33}$$

Where Y_{track} is the output. An example of a tracklet can be seen below:

$$T_1 = \{d_1^{t-1}, d_4^t, d_7^{t+1}\} \tag{34}$$

eliwi204@student.liu.se

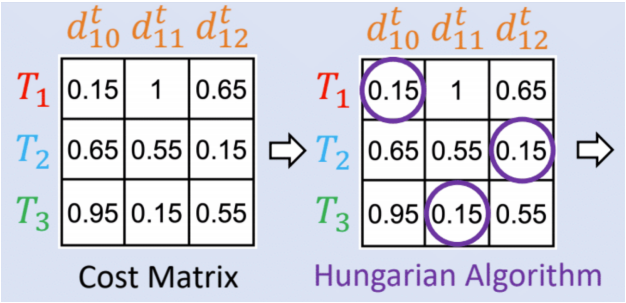

Figure 8: Cost matrix and hungarian algorithm.

When a new incoming frame is processed and new detections are found they should be associated with a tracklet. This will be done by using a cost matrix and the hungarian algorithm. An example of this can e seen in 8. New detections will be compared with tracklets to see which tracklet fits best.

The detection with the lowest cost will be associated with the tracklet, the hungarian algorithm ensures minimal total cost [7]. Listed below are some cases where special actions are needed.

- Unmatched tracklets: Kill if to old
- Unmatched detection: Start a new track

These cases will be identified with both trivial cases where we have more or less detections than tracks, but also a cost threshold which will not associate a detection to the track if the cost is to high. The question now is how to compute this cost between detection and tracklet. Note also that these tracklet entities does not need to be explicitly produced, each detection can instead be associated with an ID that marks which track they are associated with.

4.3.1 *Cost using constant velocity model*

The first method purposed is to use a constant velocity model to predict the position of the new detection from the previous ones. After this prediction it is possible to use intersection over union to calculate cost. The velocity between two consecutive detections can be estimated as below:

$$v_{t,t-1} = \frac{d^t(u,v) - d^{t-1}(u,v)}{\Delta t} \tag{35}$$

where $d^t(u,v)$ is the center of the detection at frame t and Δt is the time between two consecutive frames.
The predicted position of the new detection can now be estimated as below:

$$d^t_{pred}(u,v) = \frac{\Delta t}{N} \sum_{n=1}^{N} v_{t-n,t-n-1} + d^{t-1}_{meas}(u,v) \tag{36}$$

Which is an average over N previous frames using the detections in one tracklet.
Now the intersection over union can be calculated using the new detections as below:

$$IoU = \frac{|d^t_{pred}(u,v) \cap d^t_{meas}(u,v)|}{|d^t_{pred}(u,v) \cup d^t_{meas}(u,v)|} \tag{37}$$

Atlast the cost can be calculated as below:

$$Cost_{cv}(T_i, d^t_j) = 1 - IoU(d^t_{pred}, d^t_{meas}) \tag{38}$$

4.3.2 *Cost using Visual features*

Visual features can also be used to associate detections across frames. This will be done with a pre-trained image encoder network V and the feature vectors for each detection will be compared using the cosine distance as below:

$$Cost_{vf}(T_i, d_j) = \frac{1 + dist_{cosine}(V(T_i), V(d_j))}{2} \tag{39}$$

4.3.3 *Combining costs*

It is easy to combine the cost for the constant velocity model and the visual encoders cost by averaging the cost matrix they produce. The purpose of combining them is to both take into account the physical movement and position of the face but also the visual appearance. The weight of averaging can be explored to achieve a good combination. It is also important to know that both of the systems are working properly on their own. If the visual encoder does not perform well at creating similar feature vectors for the same face or the other way around creating different feature vectors for different faces.

4.4 Depth estimation

To enable future development and accurate localization a depth estimation module will be implemented. This module will estimate the depth using the face detections in each frame. The distance to each detection will be estimated by the size of the detection and a reference face size.

4.4.1 *Distance conversion*

The detections will be given with a certain height and width in pixels from the camera. These needs to be converted to actual distance for comparision with the reference face. This can be done by using the pinhole camera [8] model illustrated in Figure 9. The distance in meters of the object on the image plane can be calculated as:

$$x_i = f\frac{x_c}{z_c} \tag{40}$$

$$y_i = f\frac{y_c}{z_c} \tag{41}$$

Where x_i and y_i is the distances on the image plane, x_c and x_c is the real object size, z_c is the distance from the camera to the object and f is the focal length of the camera. Note that image plane actually refers to the virtual image plane in this case.

The relation between a distance of the object on the image plane and the pixel width of the detection is not linear in reality, but for our intents and purposes we can approximate it to be. Also, the pixels are as wide as they are tall so the same relationship holds for both x and y axis. We get the relationships below:

$$w_p = c_p x_i \tag{42}$$

$$h_p = c_p y_i \tag{43}$$

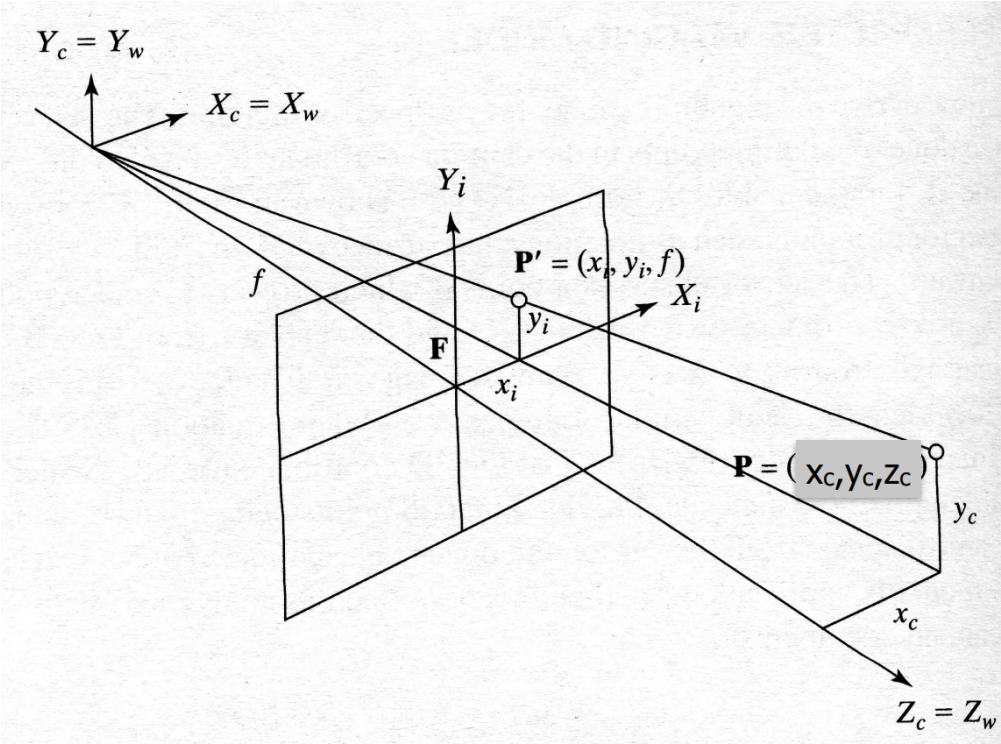

Figure 9: Illustration of pinhole camera model coordinate system.

where w_p and h_p are the pixel width and height of the detection and c_p is some constant.
Now by combining these equations we can create the expression for the distance to the object:

$$z_c = fc_p \frac{x_c}{w_p} \tag{44}$$

$$z_c = fc_p \frac{y_c}{h_p} \tag{45}$$

Both the focal length f and this constant for pixel to distance conversion c_p are unknown, but the separate values does not need to be known. Take a new parameter $C_{f,p} = fc_p$ and the equations become:

$$z_c = C_{f,p} \frac{x_c}{w_p} \tag{46}$$

$$z_c = C_{f,p} \frac{y_c}{h_p} \tag{47}$$

This constant C_f, p will be evaluated by experiment with an object of known size and distance.

4.4.2 *Estimating the distance to the face*

After converting the detection to distance, the distance to the face can be estimated according to the equation below:

$$z_{c,x} = C_{f,p} \frac{x_{ref}}{w_p} \tag{48}$$

$$z_{c,y} = C_{f,p} \frac{y_{ref}}{h_p} \tag{49}$$

$$z_f = \frac{z_{c,x} + z_{c,y}}{2} \tag{50}$$

where x_{ref} and y_{ref} is the reference width and height of a face, w_p and h_p is the detection size in pixels and z_f is the final estimated depth. z_f is calculated as the mean of the depth estimations for the x and y axis.

4.5 Eye-tracking

The direction of the gaze of the user is well developed in the python API from Tobii and will directly be used. The output data from the API includes a time stamp, gaze point in 2D, gaze point in 3D, gaze origin and gaze direction. The 2D gaze point is given in normalized video coordinates for the video output of the glasses. The 3D gaze point is a position vector in the coordinate system of the glasses, which is illustrated in Figure 10. The gaze origin is a position in the same coordinate system as the 3D position and represents the location of the center of the pupil of the eye. The gaze direction describes the direction of the gaze for each eye and has its origin in the gaze origin for the respective eye. There are thus two vectors of gaze origin and gaze direction, one for each eye. In order to get accurate values a calibration has to be performed since the head size and face attributes differs from user to user. [9]

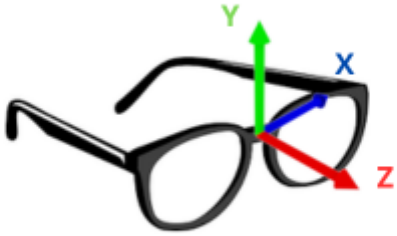

Figure 10: Coordinates system of Tobii G3 glasses.

5 SLAM

The purpose of this system is to fuse the input data from the sound source tracking module and the G3 processing system together. This will be done in order to get a better tracking and localization estimation of the sound source and the moving platform including G3 and hearing aids. SLAM is an algorithm that are used to estimate the location of landmarks and its own position and orientation according to those landmarks. One specific approach within SLAM is known as EKF-SLAM which are used in this system.

$$x_{k+1} = F_k x_k + G_k w_k, \qquad \text{Cov}(w_k) = Q_k \tag{51a}$$
$$m_{k+1} = m_k \tag{51b}$$
$$y_k = C_k^x x_k + C_k^m(c_k^{1:I_k}) m_k + e_k, \qquad \text{Cov}(e_k) = R_k \tag{51c}$$

The measurement, denoted as y_k depends on the presence of associations with specific landmarks, which are represented by the vector m_k. The measurements will change when landmarks are introduced or removed. To apply EKF to this model the state vector is extended and the map is included.

$$\hat{z}_{k|k-1} = \begin{pmatrix} \hat{x}_{k|k-1} \\ \hat{m}_{k|k-1} \end{pmatrix}, \qquad P_{k|k-1} = \begin{pmatrix} P^{xx}_{k|k-1} & P^{xm}_{k|k-1} \\ P^{mx}_{k|k-1} & P^{mm}_{k|k-1}. \end{pmatrix} \tag{52}$$

By using equations (51) and (52), the standard form of EKF can applied to obtain the measurement and time update. The equations for the measurement update are

$$S_k = C_k^x P_{k|k-1}^{xx} C_k^{xT} + C_k^m P_{k|k-1}^{mm} x_k^{mT} + C_k^m P_{k|k-1}^{mx} C_k^{xT} + C_k^x P_{k|k-1}^{xm} c_k^{mT} + R_k \tag{53a}$$

$$K_k^x = (C_k^x P_{k|k-1}^{xx} + C_k^m P_{k|k-1}^{mx})^T S_k^{-1} \tag{53b}$$

$$K_k^m = (C_k^x P_{k|k-1}^{xm} + C_k^m P_{k|k-1}^{mm})^T S_k^{-1} \tag{53c}$$

$$\varepsilon_k = y_k - C_k^x \hat{x}_{k|k-1} - C_k^m \hat{m}_{k|k-1} \tag{53d}$$

$$\hat{z}_{k|k} = \hat{z}_{k|k-1} + \begin{pmatrix} K_k^x \\ K_k^m \end{pmatrix} \varepsilon_k \tag{53e}$$

$$P_{k|k} = P_{k|k-1} - \begin{pmatrix} K_k^x \\ K_k^m \end{pmatrix} S_k \begin{pmatrix} K_k^x \\ K_k^m \end{pmatrix}^T \tag{53f}$$

and the time update equations are

$$\hat{z}_{k+1|k} = \begin{pmatrix} F_k & 0 \\ 0 & I \end{pmatrix} \hat{z}_{k|k} \tag{54a}$$

$$P_{k+1|k} = \begin{pmatrix} F_k P_{k|k}^{xx} F_k^T + G_k Q_k G_k^T & F_k P_{k|k}^{xm} \\ P_{k|k}^{mk} F_k^T & P_{k|k}^{mm} \end{pmatrix} \tag{54b}$$

6 SOUND PROCESSING CONTROL

The filter module will handle the beamforming algorithm in order to focus the transmitted signals to the specific direction the user is looking at. The module will also be able to detect speech signals and amplify the desired sound signal in a given frequency band. The beamforming algorithm will use data from the the G3 processing system. The beamforming module will have a direction as an input and will thereafter handle the amplification in that direction.

6.1 Beamforming module

6.1.1 *Signal model*

An acoustic signal from microphone m in the time domain can be expressed as

$$x_m(t) = s_t(t) * d_m(t, \theta_s) + s_q(t) * d_q(t, \theta_s) + v_m(t), \tag{55}$$

where $s_t(t)$ and $s_q(t)$ is the target signal and the interfering target signal at the reference microphone respectively. $d_m(t, \theta_s)$ is the relative transfer functions from the target to microphone m and $d_q(t, \theta_s)$ is the relative transfer functions from the interfering target to microphone m. The relative transfer function includes the acoustic transfer function from a far field source in the horizontal plane to each microphone, potentially shadowed by the head.

$\theta_s \in [-180°, 180°]$ is the DOA angle from the reference point to the target and the noise $v_m(t)$ is the sum of all undesired signals. The N-size fft is thereafter considered for each short segment in time (analysis window). The acoustic signal from microphone m in the time-frequency (TF) domain can therefore be expressed as

$$x_m(k,n) = s_t(k,n)d_m(k,n,\theta_s) + s_q(k,n)d_q(k,n,\theta_s) + v_m(k,n), \tag{56}$$

where k and n are the frequency bin index and the frame index respectively. The k and n indexes will be omitted in the rest of the document for brevity.

6.1.2 *Linear beamformer*

The goal of a beamformer is to amplify the target speech signal s_t and suppress surrounding noise. A linear beamformer can be expressed as

$$y = w^H x \tag{57}$$

, where w is the beamformer weights, H is the Hermitian transpose and y is the output signal in the TF domain. x is a vector that contains all the observed noisy signals from each microphone $m = 1, 2, 3$ and 4 in the TF domain. x can be expressed as $x = [x_1, x_2, x_3, x_4]^T$. There are many beamforming algorithms that can be used for estimating the beamformer weights. In this project the well-known beamformer algorithms minimum-variance distortion-less response (MVDR) and the multichannel Wiener filter (MWF) will be used [10].

6.1.3 *MVDR*

The beamformer weights for the MVDR solution in the TF domain can be expressed as

$$w_{MVDR} = \frac{\boldsymbol{C}_v^{-1}\boldsymbol{d}}{\boldsymbol{d}^H\boldsymbol{C}_v^{-1}\boldsymbol{d}}, \tag{58}$$

where $\boldsymbol{C}_v$ is the noise cross power spectral density (CPSD) matrix, $\boldsymbol{d} = [d_1(\theta_s), d_2(\theta_s), d_3(\theta_s), d_4(\theta_s)]^T$ is the relative transfer function (RTF). The CPSD matrix are defined as $\boldsymbol{C}_v = E[\boldsymbol{v}\boldsymbol{v}^H]$, where $\boldsymbol{v} = [v_1, v_2, v_3, v_4]^T$.

6.1.4 *MWF*

The solution for the Multichannel Wiener Filter in the TF domain is given by

$$w_{MWF} = w_{MVDR}g_{sc} = \frac{\boldsymbol{C}_v^{-1}\boldsymbol{d}}{\boldsymbol{d}^H\boldsymbol{C}_v^{-1}\boldsymbol{d}} \cdot \frac{\lambda_t}{\lambda_t + (\boldsymbol{d}^H\boldsymbol{C}_v^{-1}\boldsymbol{d})^{-1}}, \tag{59}$$

where w_{MVDR} is the MVDR beamformer and g_{sc} is known as the single-channel post Wiener filter. $\lambda_t = E[|s_t|^2]$ is the target power spectral density.

6.2 Personalized Hearing Amplification

An audiogram is a graph showing the results from a standardised test of a person's hearing. The tested frequencies are spanning from 125 to 8000 Hz. What is measured at preset frequencies is the lowest decibel hearing level (dB HL) at which the person can hear a pure sinusoidal tone. 0 dB HL is correspondent to the sound level at which a person without hearing loss can hear the tone. Due to normal variations in human ears the range -10 to 20 dB HL

is considered good hearing. The higher the dB HL for a specific frequency the worse hearing the person has at that specific frequency. The audiogram for a person can differ for each ear.

To be able to produce a personalized hearing amplification to the hearing aids the audiogram is used. The frequency band of the audiogram is split up into a number of smaller frequency bands called channels. Since the perception of hearing is not linear the channels do not need to be of the same size. There is also some overlap of the channels. The number of channels will affect the resolution of the fitting and are normally varying from 4 to 64. The higher the number of channels the more exact a fitting to an audiogram can be done but at a higher computational cost. The channels are then amplified according to the audiogram using a simple band pass filter. All channels are then combined to make make the sound output. [11]

A person might also have a more narrow frequency band in which the person can hear comfortable. A compression of the sound input can then be necessary. This can be done with dynamic range compression. The compression can be performed on the whole spectrum (single channel) or, for a better personalized hearing, on multiple channels.

REFERENCES

[1] "Tobii official website," https://www.tobii.com/, [Online; accessed September 13, 2023].

[2] T. AB, "Tobii pro glasses 3," https://corporate.tobii.com/sv/media/mediabank, [Online; accessed September 19, 2023].

[3] J. A. MacDonald, "A localization algorithm based on head-related transfer functions," *The Journal of the Acoustical Society of America*, p. 123 (6), March 2008.

[4] "Opencv documentation," https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html, [Online].

[5] "A guide to two-stage object detection: R-cnn, fpn, mask r-cnn," https://medium.com/augmented-startups/top-5-object-tracking-methods-92f1643f8435, [Online; accessed September 20, 2023].

[6] V. Sommers, "Lecture notes in deep learning for autonomous vehicles," February 2023.

[7] "Hungarian maximum matching algorithm," https://brilliant.org/wiki/hungarian-matching/, [Online; accessed September 20, 2023].

[8] P. Fua, "Lecture notes in computer vision cv442," February 2023.

[9] "Tobii pro glasses developer guide," https://go.tobii.com/tobii-pro-glasses-3-developer-guide, [Online; accessed September 26, 2023].

[10] Z.-H. T. S. M. I. J. J. Poul Hoang, Jan Mark de Haan, "Multichannel speech enhancement with own voice-based interfering speech suppression for hearing assistive devices," 2022.

[11] A. Goldin, "Digital signal processing for over-the-counter hearing aids," March 2023.