Blender Add-on for the Registration of Parametric Pinna Model for Personalized Binaural Audio

Bachelor Thesis from Aufnahmetechnik 1, SE

Yasen Velchev

Supervisor: Univ.Prof. DI Dr. Alois Sontacchi
Co-supervisors: Piotr Majdak, Florian Pausch, Felix Perfler

Graz, September 14, 2023

kunst uni graz

institut für elektronische musik und akustik

Abstract

Head-related transfer functions (HRTFs) are becoming a very prevalent part of 3D audio, virtual and augmented reality. The demand for a personalized set of HRTFs requires affordable ways of their acquisition. In order to overcome the high costs of measurement equipment for acoustic aquisition of HRTFs, it is possible to parametrize the human pinna, to acquire a 3D mesh and to numerically calculate the HRTFs. One method of parametrization, developed at the Austrian Academy of Sciences, the so-called parametric pinna model (PPM), is explored in this thesis. The PPM can be used to modify a perfect but generic pinna mesh to an individual but noisy 3D pinna mesh and the result of such registration can be evaluated in the geometric domain. In this thesis, I created a Blender add-on to improve the registration process by creating a better visualisation of the geometric evaluation. The results are shown using an example of registration showing that the registered model has very low geometric error (under one millimeter), as compared to the original target mesh and can make acquiring personalized HRTFs available to the wider audience.

Contents

1 Introduction - HRTF's and their acquisition

Head-related transfer functions (HRTFs) are the spatial filters of the acoustic field, coming from a sound source to the human ear. They describe the relationship between the sound field, filtered by the geometry of the human anatomy and the source signal. The HRTFs are direction dependent, as they depict the path of the acoustic field to the ear canal. They are used by the auditorial system for sound source localization trough aural cues [Bla97]. The two main cues are the interaural time difference (ITD) and the inter aural level difference (ILD). With ITDs a person can determine the location of a sound source based on the time difference of the arrival of the sound wave to the two ears. This is especially prominent in low frequencies (below 1.4 kHz). The damping of the sound through the head and torso is described by the ILDs which are prominent for frequencies between 1.5kHz and 3kHz around the wavelengths of the body and head geometries. ILDs and ITDs help with localization on the horizontal plane while spectral cues are responsible for the localization of different elevations. Spectral cues are the unique spectral filters of a person's pinna. They are mostly prominent above 3kHz, where the sound reflections from the pinna geometry can have the most effect. All of these cues are represented in the HRTFs and that makes the HRTFs unique to each person, like a fingerprint [AGE15].

The recent technology of binaural audio has a great use of HRTFs in 3D audio, virtual reality and the gaming industry. These industries use generic sets of HRTFs, which have to approximate the experience [MCD+12], but for a more immersive experience and more accurate localization, acquisition of a personalized set of HRTFs is required. There are various methods of acquiring a set of HRTFs both through acoustic measurement and through numerical calculation.

The acoustic measurement of HRTFs is a lengthy and costly process that requires a fully- or semi-anechoic chamber and a way of generating sound signals from multiple directions and capturing them from the human ear. The process is described in more detail in [LP20]. An alternative to that can be found in the numerical acquisition of HRTFs. In order to calculate the HRTFs a 3D scan of the person's head and pinna is needed. This is done by the means of magnetic resonance scan, photogrammetry, computer tomography or laser scan, the results from which may vary in accuracy and noise. [PMK22] The result of the scan is a 3D mesh, which is then used to virtually simulate the measurement experiment. By the means of the boundary element method (BEM) the HRTFs, which are similarly accurate to the measured ones, are calculated [ZRM13], [ZMK15]. Because the laser scan, MRI scan and the computer tomography are also costly and equipment dependent processes, alternative ways of acquiring a 3D mesh are needed. This can be done by expressing the pinna geometry using parameters.

Studies have shown that the rest of the human anatomy (head and torso) can be expressed using simple geometry in the context of numerical calculation of HRTFs [AAD01] [FV09]. Because of that, way more emphasis is put on the parametrization of the pinna geometry, as it is much harder to simplify it, because of its uniqueness for each person. Different ways of parametrization of the pinna geometry have been found. Algazi and Duda [ADTA01] produced the CIPIC database of HRTFs, but also created a parametrization of the body of each subject and analyzed different correlations between these param-

eters. Stitt and Katz [SK21] created a parametric pinna model (PPM-Paris) based on the CIPIC-parameters in the graphic software Blender and explored the effect of each parameter on the HRTFs. The PPM is a template pinna mesh, deformed using control points, the position of which is determined by the CIPIC-Parameters, see Figure 2a. By deforming the mesh with the control points a new mesh is created, the HRTFs are numerically calculated and then compared. The study shows that the areas affecting the HRTFs the most are the cavum conchae, cymba conchae, scapha and the fossa triangularis, all of which are located close to the ear canal (see Figure 1). The PPM from Stitt and Katz gives us important conclusions, but their model is limited on deformation capabilities and cannot be used for acquiring a 3D mesh of an arbitrary person's pinna. This is the goal of the parametric pinna model from Pollack and Majdak [PPM] [PM21], which can generate meshes with very low geometric errors from their target meshes (see Figure 2b). This is done by the state-of-art methods of skeletal animation and morph-target animation . Their model is evaluated in the geometric domain using Hausdorff distance and in the psychoacoustic domain using a sagittal-plane localization experiment with the Auditory Modeling Toolbox [SM13]. The HRTFs were numerically calculated using the Mesh2HRTF software [ZKM15]. The localization experiment in the psychoacoustic domain evaluates the HRTFs localization by the means of quadrant errors (QEs) and polar errors (PEs). The QEs are the rate (in %) by which signals from the front and back get confused and PEs are the rate (in degrees) by which signals from the same hemisphere get confused. The HRTFs, calculated from the adapted PPM are compared to acoustically measured HRTFs of these subjects. The results from the evaluation [PM21] show geometric errors under 1 mm and errors in the psychoacoustic domain within the typical range of listener specific HRTFs. [WAKW93]

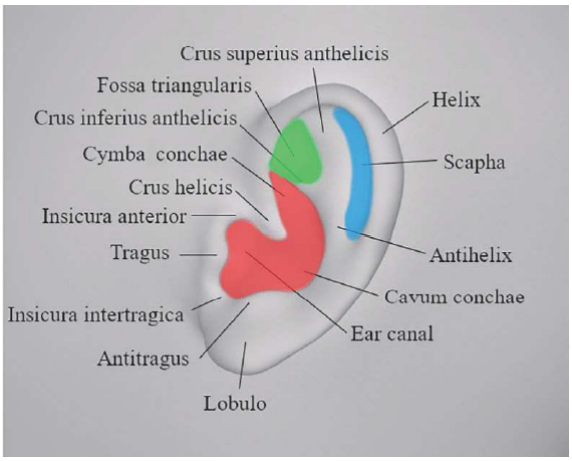


Figure 1: *Relevant concave areas of the pinna for the HRTFs: red - conchae, green - fossa triangularis, blue - scapha.*

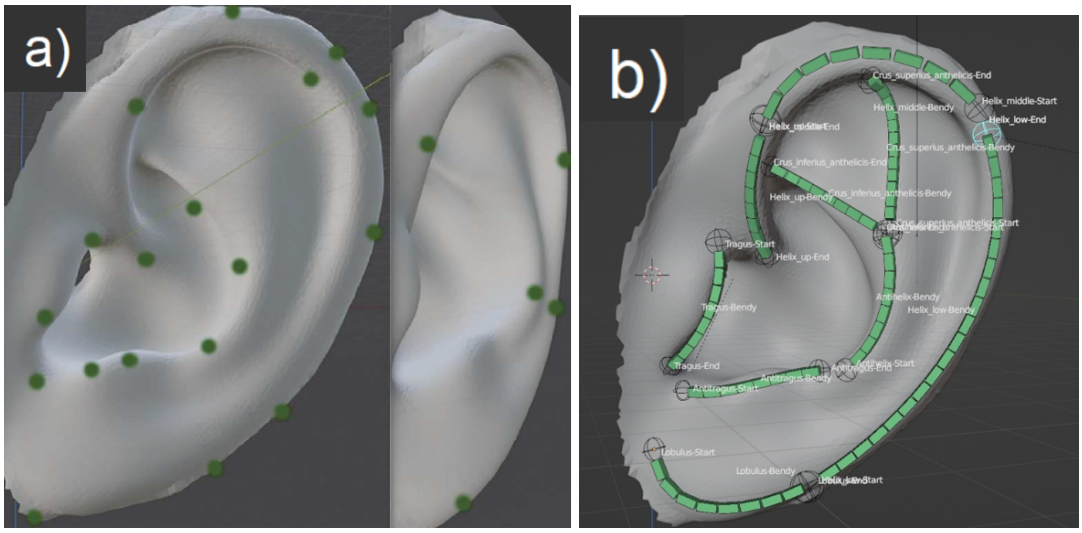


Figure 2a: *Parametric pinna model by Stitt and Katz. Template pinna (grey) deformed by 24 control points (green).* ***Figure 2b:*** *Parametric pinna model by Pollack and Majdak. The template pinna (grey) is deformed by 9 bendy bones (green Bezier curves), 18 control bones (black spheres) and 18 shape keys (not visible).*

The goal of this thesis is to explore the PPM from Majdak and Pollack (PPM-Vienna) by performing manual alignments of the PPM to two target meshes and evaluating them in the geometric domain (Section 2). In order to optimize the process, a tool for the visualization of the pointwise minimum distances is presented. (see Section 3) A discussion on what could be done in further research can be found in Section 4.

2 Exploring PPM-Vienna

2.1 Blender topologies

The PPM creates a 3D pinna mesh in the 3D graphic and animation software Blender, that can be manually aligned to a target left pinna with very low geometric errors. The model uses a template pinna mesh, taken from the WiDESPREaD database [GS20] as an average human pinna shape. The process of aligning the mesh to the target uses the state-of-art methods of skeletal animation and morph-target animation to deform the template pinna mesh. These methods use the Blender topologies, called “bendy bones” and “shape keys”, that are adapted to the template pinna mesh in order to deform it. The “bendy bones” are Bezier curves, that deform the mesh in a segmented manner to achieve the more global transformations of the ear shape. There are nine bendy bones, each of them is controlled by two control bones for a total of 27 Bones. All of them are “parented” to a “parent”-bone which enables control over the position, size and rotation of the whole mesh. The effect that the Bendy Bones have on the mesh is showed in Figure 3b, where the bone “Helix_Low_Bendy”, controlled by the “Helix_Low_Start” and “Helix_Low_End” control bones, is moved to deform the lower part of the helix. The effect of rotating the

parent bone “Size-Bendy” is showed in Figure 3c.

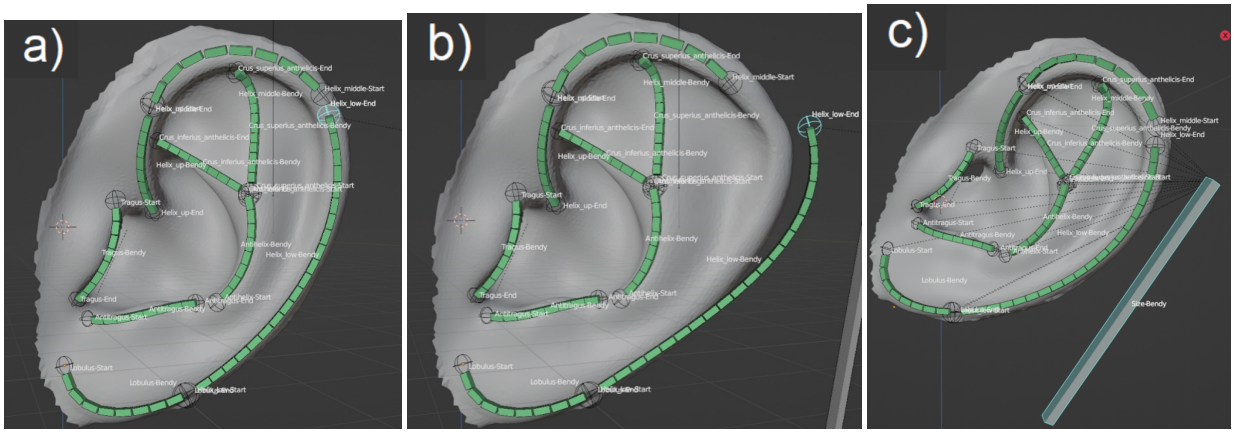


Figure 3a: *PPM in rest pose (no changes done to the mesh).* ***Figure 3b:*** *Example for deforming the mesh by moving the control bone “Helix_low_End”.* ***Figure 3c:****: Example for manipulating the mesh by rotating the parent bone “Size_Bendy”*

The other topologies, used to deform the template mesh are the shape keys. They are used to adjust details to the concave areas of the model. They can take a value between 0 and 1 or -1 and 1. There are 18 shape keys adapted to the model, which are responsible for the areas like cavum conchae, cymba conchae, ear canal diameter, the depth of the helix parts and others. The effect of adjusting the shape key “Fossa_Triangularis_Depth” from -1 to 1 is shown on Figures 4a and 4b.

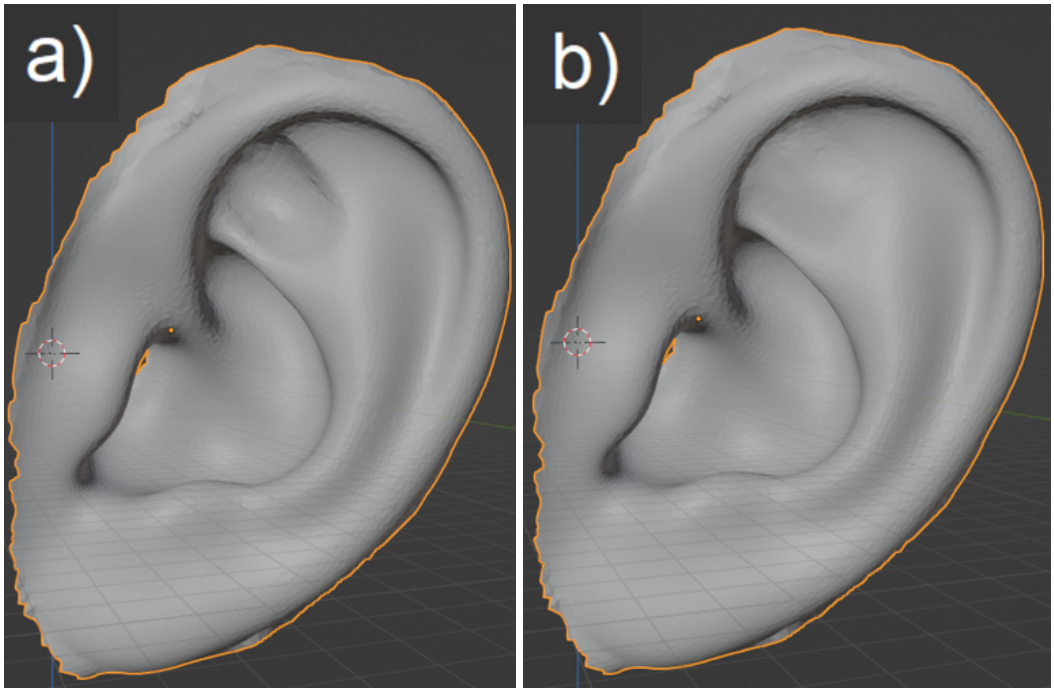


Figure 4: *Example for deforming the mesh using shape keys with the armature hidden:* ***a)*** *"Fossa_triangularis_depth" is set to -1,* ***b)*** *"Fossa_triangularis_depth" is set to 1*

In total the model can be described with Formula 1:

$$X = X_0 + \sum_{m=1}^{M} V_m \cdot a_m + \sum_{n=1}^{N} w_n \cdot b_n \tag{1}$$

X describes the deformed mesh, X_0 is the template pinna mesh, the first sum describes the deformation using the bendy bones a_m (M in total), multiplied by the weighing matrix V_m, describing the rigid transformations of the control bones and the size. The second sum describes the effect of the shape keys b_n (N in total) weighed by the respecting factor w_n (taking values between -1 and 1).

2.2 The Blender Python API and the Matlab interface

The software Blender stores the data of the objects (vertex coordinates, bendy bones deformations, shape keys weights), which could be accessed for further calculations. In order to access this data, one can use the Blender Python API. Using the programming language Python and importing the *bpy* module one can get the parameter values as well as export the target mesh and the aligned PPM as point clouds (See Figure 4). Then, using a Matlab interface [1], built by Majdak, Pollack and Pausch [PM21] and the data from the point clouds a geometric evaluation is performed (see Section 2.3). A more detailed look on how the Blender Python API accesses the parameters of the PPM is given in Figure 5.

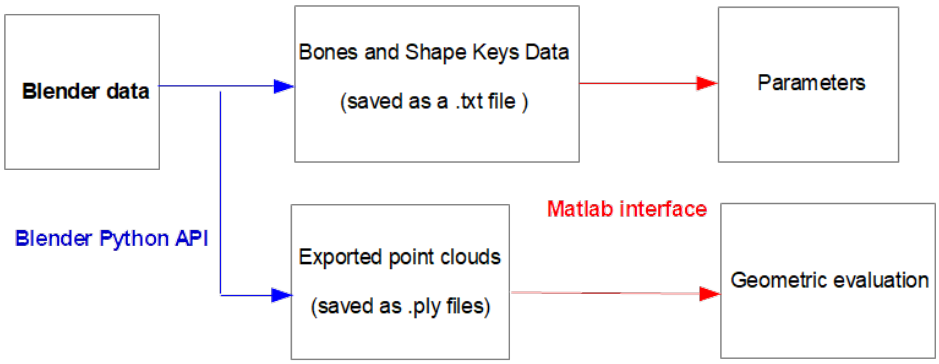


Figure 4: *Block diagram of which programming languages (modules) were used to access the Blender data*

1. https://github.com/Any2HRTF/PPM/tree/matlab

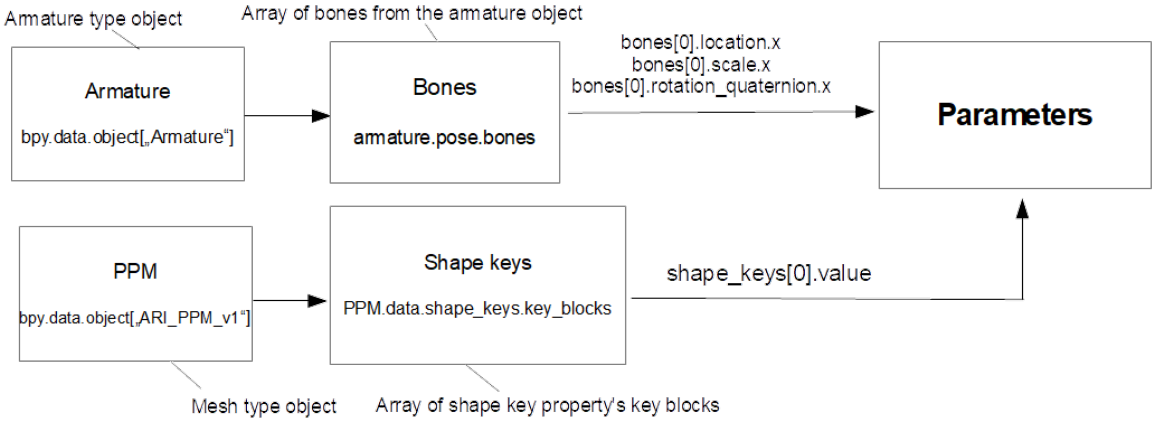


Figure 5: *Block diagram displaying how the Blender Python API accesses the PPM's parameters*

When the Matlab interface is run, a Python function is called to perform the extraction of the parameters as shown in Figure 5, and saves them as a *.txt* file from which the Matlab interface can store those parameters. The way the Python function works is that it looks for an object called *"Armature"*, which is the PPM's bendy bones configuration. From the armature object, the program can get the data for each bone of the *armature.pose.bones* array. The parameters themselves are the location, scaling and rotation in x,y or z direction of each bone. The function iterates over each bone of the bones array to get each of these parameters. The shape keys data is accessed from the same function by finding the *"ARI_PPM_v1"* object, which is the aligned pinna mesh and getting the object's *data.shape_keys.key_blocks* property. This property is an array of shape keys, whose value is accessed by the program through iteration over each shape key. The Blender Python API is the key to understanding the registration process of the parameters (Section 2.3), the registration results (Section 2.4), as well as the newly implemented Blender add-on for the geometric evaluation of the PPM (Section 3).

2.3 Registration Process

In order to get to know the workflow of the PPM, I performed two registrations. A registration is the process of manually aligning the PPM to a target pinna mesh, getting the parameters and evaluating the registration. In the following paragraphs I will describe the registration workflow. Firstly, I imported a 3D scan of a head and torso to the Blender project, containing the PPM. The mesh got cleaned up, so that only the left pinna remains, which became the target mesh. After that, I started aligning the PPM to the target mesh using only deformations of the bendy bones armature. The "Size_Bendy" parent bone was used to align the whole mesh and then the rest of the bendy bones were manipulated to match the forms of the two target meshes. After the overall shape was adapted, I concentrated on adapting the details in the concave areas using the shape keys. When the

aligning was done, I ran the Matlab interface of the PPM to get the parameters and evaluate the registration in the geometric domain. The geometric evaluation uses the metric called Hausdorff distance (Formula 2). It takes the maximum value from the bidirectional Hausdorff distances (Formula 3). A directed Hausdorff distance takes the maximum of the point-wise minimum distance (PMD) (Formula 4) from a set of coordinates P to a set of coordinates Q.

Hausdorff distance:

$$H(P,Q) = max(\vec{h}(P,Q), \vec{h}(Q,P)) \tag{2}$$

Directed Hausdorff distance:

$$\vec{h}(P,Q) = max(min(\|p_i - q_j\|)) \tag{3}$$

Pointwise minimum distance (PMD):

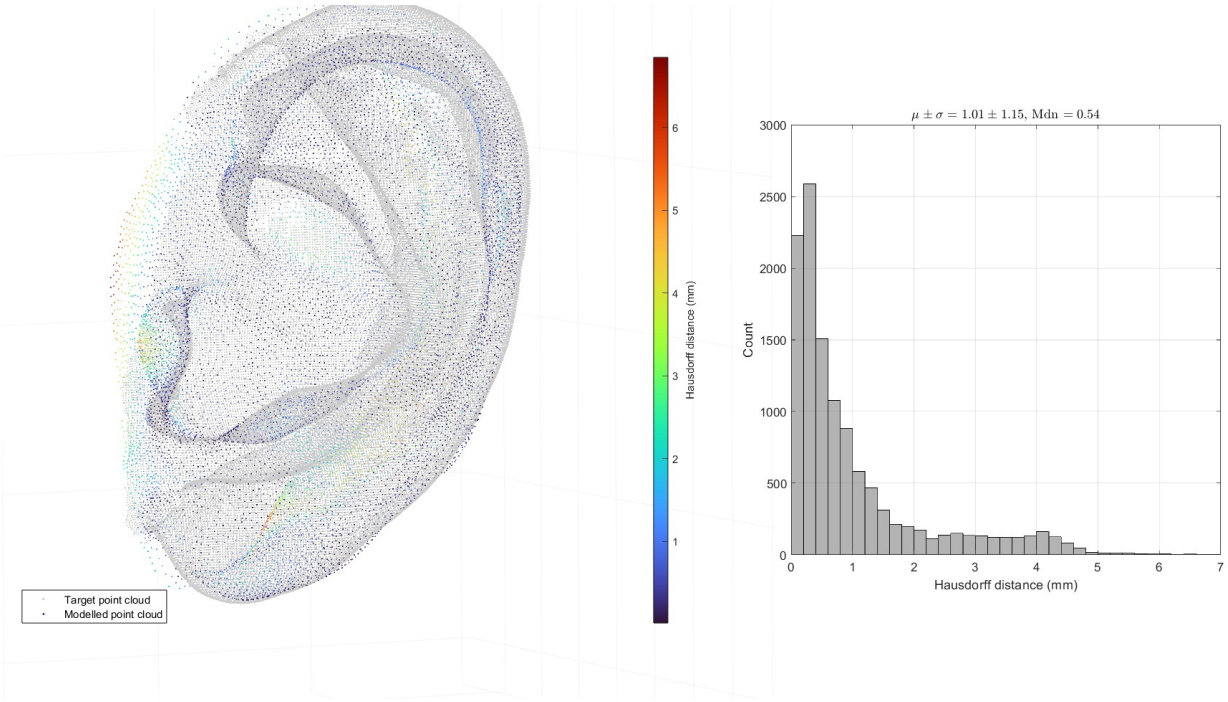
$$m_i = min(\|p_i - q_j\|) \tag{4}$$

In this notation P and Q are the sets of vertex coordinates of the aligned template pinna model and the target model respectfully, p_i and q_j are the elements of each set, from which the PMD and Hausdorff distances are calculated.

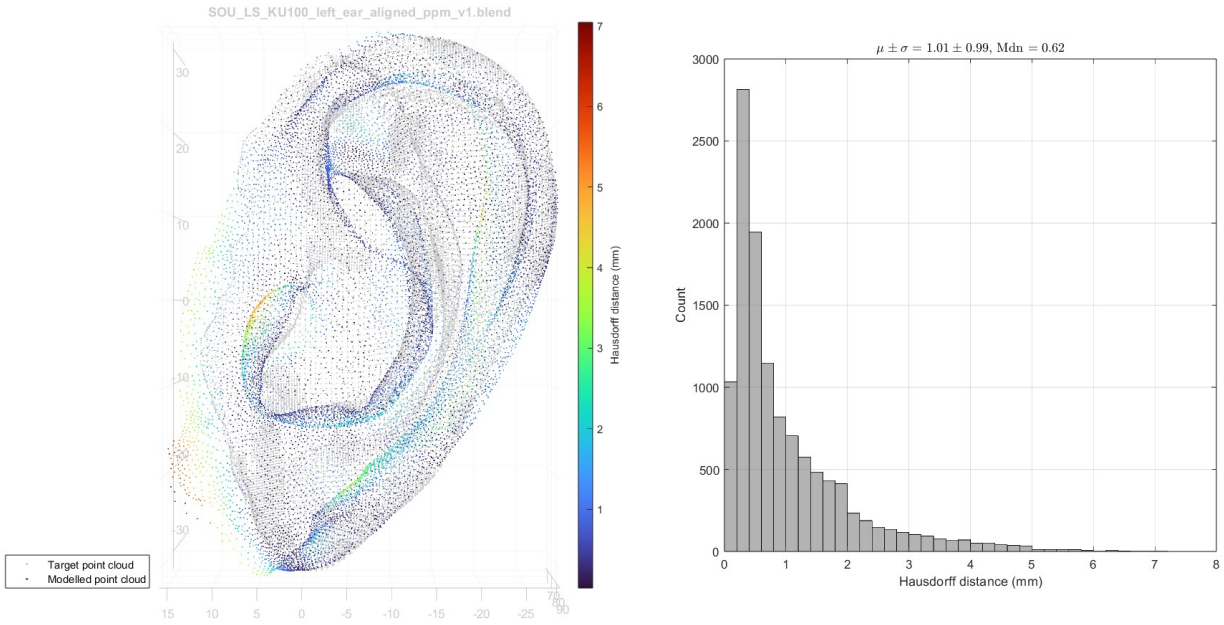
The evaluation outputs a histogram, which shows the PMD of each point of the aligned template mesh to the target mesh. It also outputs the mean and median distances. The evaluation also outputs a dynamic graph that visualizes the PMD. This is done, because in the context of HRTFs the most important parts of the pinna are the cavum concha, cymba concha and the fossa triangularis [SK21], as they affect the HRTFs the most and the visualization should provide information if the specific regions are well adapted. To show that, the diagram is color coded based on the PMD, where dark blue signifies a distance under 1 mm, green signifies a distance between 2 and 4 mm and red stands for distances over 4 mm. If the median distance is below 1 mm and the mean distance is around 1 mm, the registration is considered complete and the PPM can proceed to the numerical calculation of the HRTFs and the evaluation in the psychoacoustic domain. Because these processes, require a lot of computation power, I did the registration process only until the geometric evaluation.

2.4 Registration Results

The two registrations I did were of a 3D scan of an adult male head and torso and the KU100 Neumann Artificial Head's left pinna. A registration took between 15 minutes and one hour to do. The results, I achieved from the geometric evaluation using the Matlab interface, are shown in Figures 6a and 6b.



(a)



(b)

Figure 6a: *Registration of the human left pinna, showing a visualization of the target point cloud and the aligned point cloud (left) and a histogram with the values of the*

pointwise minimum distance (right),
Figure 6b: *Registration of the KU100 Artificial head's left pinna, showing the aligned point cloud (left) and the histogram with the values of the pointwise minimum distance (right)*

Judging by the mean and median values, both registrations were successful, the first one having 1,01 mm mean and 0,54 median distance and the second one having 1,01 mm mean and 0,62 median distance. From the 3D graph of the PMD it becomes clear, that there are no huge deviations in the important pinna areas for calculating the HRTFs. The vertices in the regions of the fossa triangularis and the conchae are all in the dark blue shades, indicating PMD under one millimeter. A better visualization of the point clouds could be done in Meshlab, but this will further lengthen the process, so another way of visualizing the results is needed.

3 Distance vizualisation add-on

A way of decreasing the time needed for a registration is to do the geometric evaluation directly in Blender. Using the Blender Python API, I have implemented an add-on [2] which calculates the pointwise minimum distance from a selected object to a selected reference object. This allows a better visualization using the Vertex Paint mode of Blender to create a clearer image of which areas deviate more from the reference image. The add-on also outputs the mean, median and Hausdorff distance (similar to the Matlab interface). This information is used to indicate if the object is well adapted and can proceed with the HRTFs calculation. The two functions can be accessed via a Panel in the Blender UI, see Figure 5.

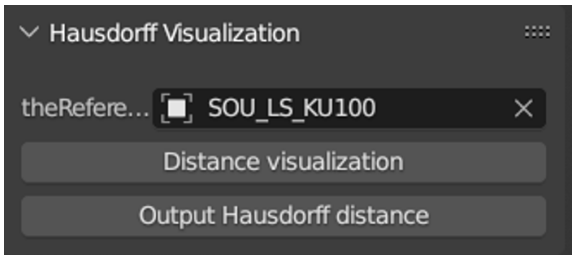


Figure 7: *The addon Panel with its two functions:* ***Distance visualization*** *and* ***Output Hausdorff distance****, as well as the object selection menu, where the reference object variable is selected*

Distance visualization. The first function is the *Distance visualisation* (See Figure 8). Its purpose is to create an image where each point is painted, depending on the calculated pointwise minimum distance. First, a reference object is selected in the panel. This

2. https://github.com/Any2HRTF/PPM

object is stored as a variable called The reference object, so that its properties (vertex coordinates) are accessible in the program. Then, the object, whose distance should be calculated is clicked and then Distance visualization can be run. The program checks if the two objects have an assigned material and color property and assigns them if needed. Otherwise, the object cannot be painted in vertex paint mode. Then the program calculates the directed pointwise minimum distance and stores it in an array, in which the index of the distance corresponds to the index of the vertex. Using this indexing the program iterates over the distance array and depending on the distance value it stores a color in a color map array. The color map is based on the distance in millimeters: if the distance is smaller than 1 mm, the vertex is colored blue, if it is smaller than 2 mm, it is colored light blue, if it is smaller than 4 mm it is colored green and over 4 mm is colored red. In a separate iteration the color gets assigned to the vertex and its neighbouring edges. After the coloring, the program outputs the mean, median, minimum and Hausdorff distances in one direction (P to Q). The outputted visualization is stored as a separate mesh object that can be moved but cannot be deformed. To create a new visualization, the previous one should be deleted. The program takes around 30 seconds for 30 000 data points to complete and the timing is to be improved in the future.

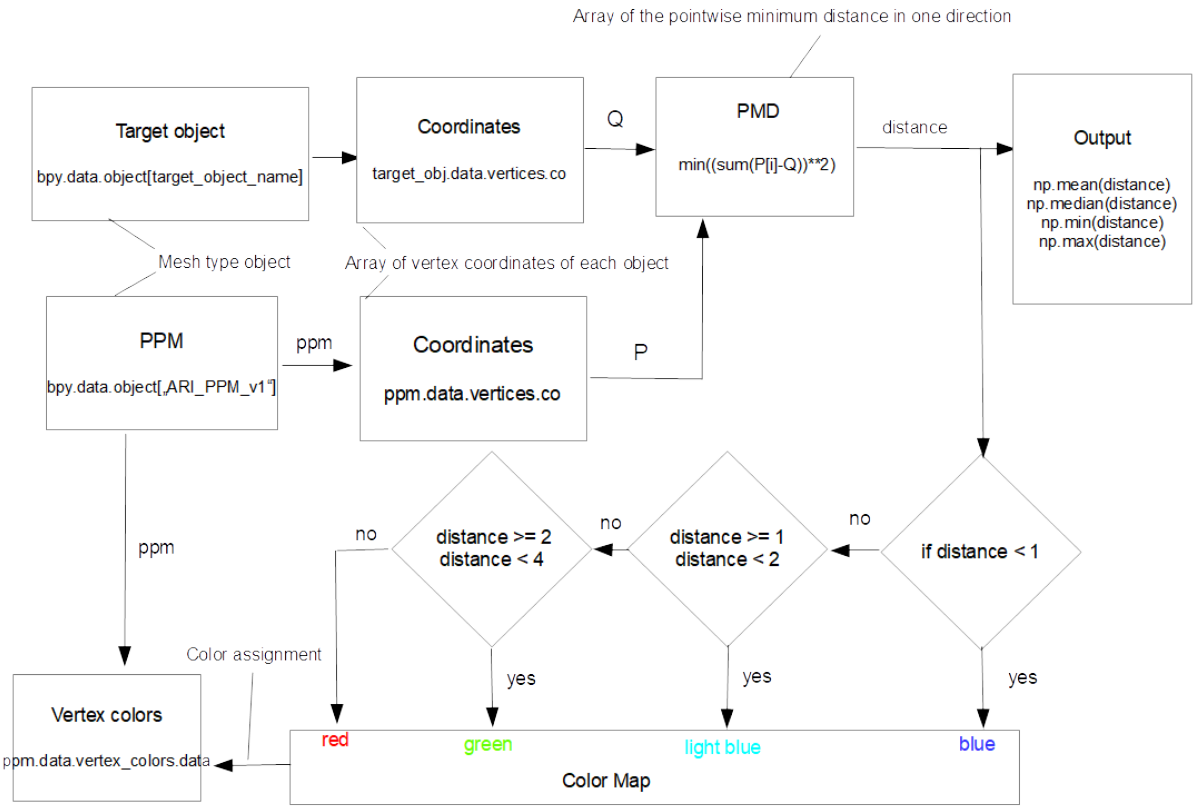


Figure 8: *Pseudo-code block diagram of the Distance visualization function with the calculation of the pointwise minimum distance and outputting its mean, median minimum and Hausdorff distances (upper right) and the color assignment, based on the distance (down)*

Output Hausdorff distance. The second function of the add-on is to output the bidirectional Hausdorff distance. (See Figure 9) The program takes the reference object and the selected object and calculates the pointwise minimum distance in both directions (P to Q and Q to P). The program then outputs the mean, median, minimum and Hausdorff distances of the combined two arrays. This gives a broader look at the worst-case PMD and Hausdorff distance, but does not create a visualization. The program takes around 1 minute for 30 000 data points to complete, because of the two directions. The slow time of the bidirectional calculation is part of the reason, why it was implemented as a separate function.

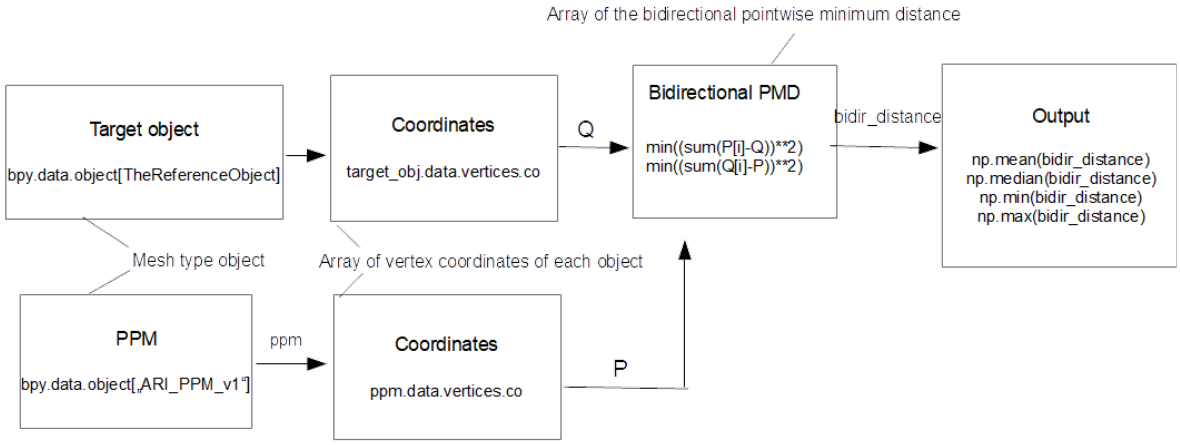
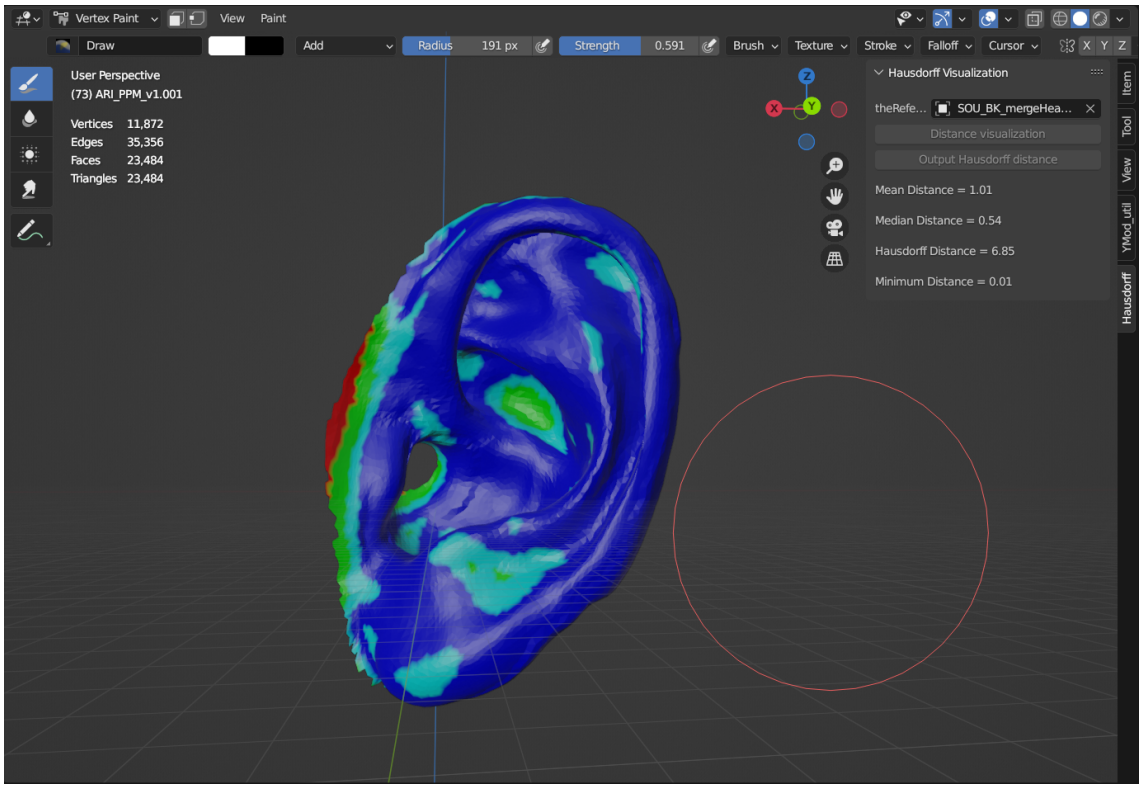
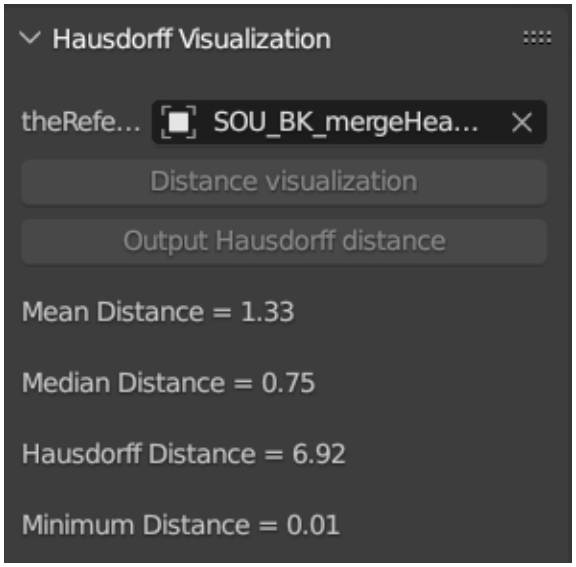


Figure 9: *Pseudo-code block diagram of the Output Hausdorf distance function. Function-wise very similar to Distance visualization (Figure 8), but the PMD gets calculated in both directions (from P to Q and from Q to P).*

Registration results using the addon. After running the *Distance visualization* and *Output Hausdorff distance* functions the output is very similar to the Matlab interface, see Figures 10a, 10b, 11a, 11b. This Blender add-on not only makes the visualization accessible during the manual alignment process, but also gives a clearer image of which areas need better alignment based on their PMD. Using the two functions, the add-on gives additional information about the bidirectional Hausdorff distance, which is important for the geometric evaluation of the model.



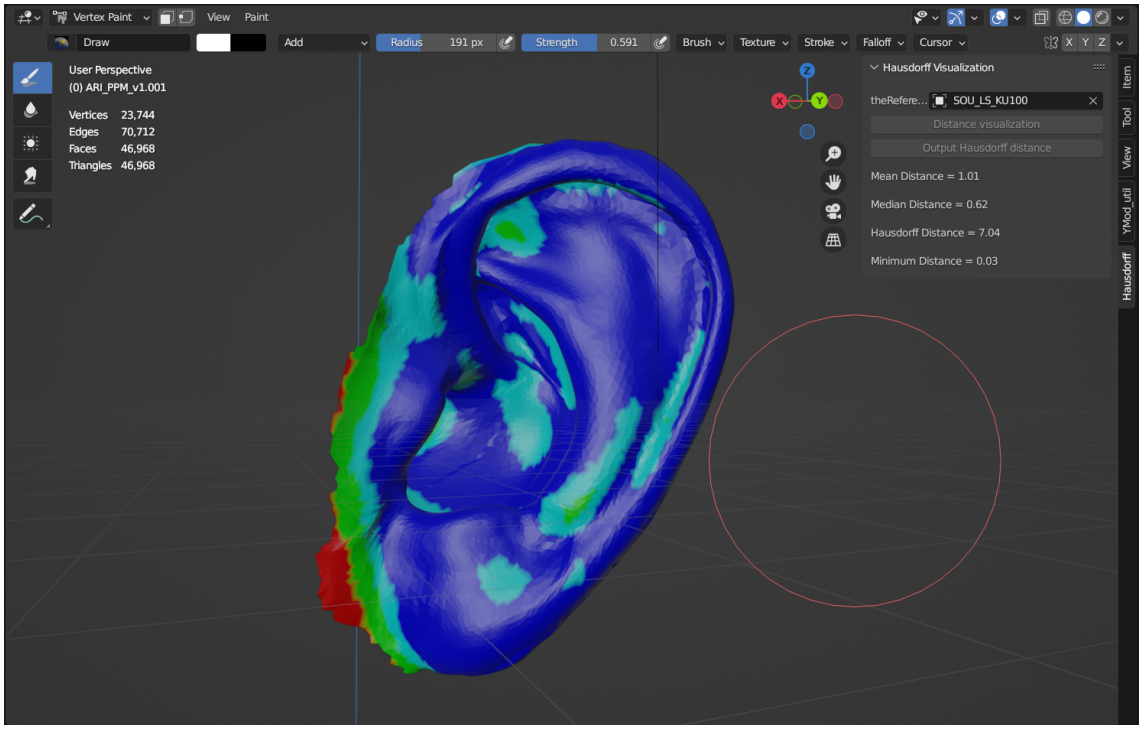
(a)



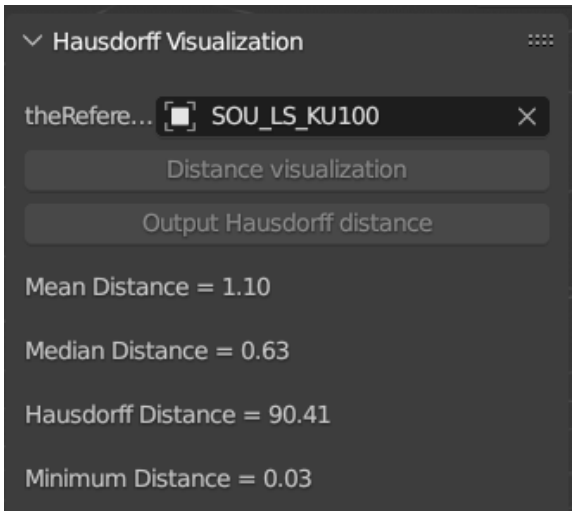
(b)

Figure 10: *Results from registration 1 using the distance visualization add-on:* ***(a)*** *Distance visualization outputs the colored mesh and the values of the directed pointwise minimum distance calculation,* ***(b)*** *Output Hausdorff distance function gives the bidirectional*

Hausdorff distance and the mean, median and minimum distances of the two directions (P to Q and Q to P).



(a)



(b)

Figure 11: *Results from registration 2 using the add-on:* ***(a)*** *Distance visualization outputs the colored mesh and the values of the directed minimum pointwise distance calculation,* ***(b)*** *Output Hausdorff distance function gives the bidirectional Hausdorff distance and the mean, median and minimum distances of the two directions (P to Q and Q to P).*

4 Conclusion and Outlook

Obtaining a 3D mesh using a parametric pinna model could change the way people get a personalized set of HRTFs. As projects like the software Mesh2HRTF [ZKM15] allow the numerical calculation of the HRTFs if a 3D mesh is provided, the PPM allows a simple way to generate one. Further research could achieve automation of the alignment process, as well as exploring the limits of the parameters, e.g. how far can the model be manipulated to achieve more unusual ear shapes. The distance visualization add-on is a tool that helps to speed up the registration process by providing the geometric evaluation directly in Blender. Shortening the time for registration could provide a large data base of parameters in the future, that could be used for example to automate the registration process. The add-on also limits the usage of external software like the Matlab interface, which now will only be used to get the parameters, which takes seconds to do. The add-on still has a lot to improve, mostly in time needed to execute and error handling. Newer versions of the add-on could include a function that extracts the parameters and could add more ways of geometric evaluation, like Chamfer distance or Jaccard index [DBM+23]. In conclusion, the PPM helps the acquisition of personalized HRTFs by providing a high quality 3D pinna mesh with very low geometric error compared to the original ear shape, which can make acquiring personalized HRTFs available to the wider audience.

References

[AAD01] V. R. Algazi, C. Avendano, and R. O. Duda, "Elevation localization and head-related transfer function analysis at low frequencies," *The Journal of the Acoustical Society of America*, vol. 109, no. 3, pp. 1110–1122, 2001.

[ADTA01] V. R. Algazi, R. O. Duda, D. M. Thompson, and C. Avendano, "The cipic hrtf database," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*. IEEE, 2001, pp. 99–102.

[AGE15] A. S. Anwar, K. K. A. Ghany, and H. Elmahdy, "Human ear recognition using geometrical features extraction," *Procedia Computer Science*, vol. 65, pp. 529–537, 2015.

[Bla97] J. Blauert, *Spatial hearing: the psychophysics of human sound localization*. MIT press, 1997.

[DBM+23] N. Dommaraju, M. Bujny, S. Menzel, M. Olhofer, and F. Duddeck, "Evaluation of geometric similarity metrics for structural clusters generated using topology optimization," *Applied Intelligence*, vol. 53, no. 1, pp. 904–929, 2023.

[FV09] J. Fels and M. Vorländer, "Anthropometric parameters influencing head-related transfer functions," *Acta Acustica united with Acustica*, vol. 95, no. 2, pp. 331–342, 2009.

[GS20] C. Guezenoc and R. Seguier, "A wide dataset of ear shapes and pinna-related transfer functions generated by random ear drawings," *The Journal of the Acoustical Society of America*, vol. 147, no. 6, pp. 4087–4096, 2020.

[LP20] S. Li and J. Peissig, "Measurement of head-related transfer functions: A review," *Applied Sciences*, vol. 10, no. 14, p. 5014, 2020.

[MCD+12] C. Mendonça, G. Campos, P. Dias, J. Vieira, J. P. Ferreira, and J. A. Santos, "On the improvement of localization accuracy with non-individualized hrtf-based sounds," *Journal of the Audio Engineering Society*, vol. 60, no. 10, pp. 821–830, 2012.

[PM21] K. Pollack and P. Majdak, "Evaluation of a parametric pinna model for the calculation of head-related transfer functions," 2021.

[PMK22] K. Pollack, P. Majdak, and W. Kreuzer, "Modern acquisition of personalised head-related transfer functions: An overview," *Advances in Fundamental and Applied Research on Spatial Audio*, 2022.

[PPM] K. POLLACK, F. PAUSCH, and P. MAJDAK, "Parametric pinna model for a realistic representation of listener-specific pinna geometry."

[SK21] P. Stitt and B. F. Katz, "Sensitivity analysis of pinna morphology on head-related transfer functions simulated via a parametric pinna model," *The Journal of the Acoustical Society of America*, vol. 149, no. 4, pp. 2559–2572, 2021.

[SM13] P. L. Søndergaard and P. Majdak, "The auditory modeling toolbox," *The technology of binaural listening*, pp. 33–56, 2013.

[WAKW93] E. M. Wenzel, M. Arruda, D. J. Kistler, and F. L. Wightman, "Localization using nonindividualized head-related transfer functions," *The Journal of the Acoustical Society of America*, vol. 94, no. 1, pp. 111–123, 1993.

[ZKM15] H. Ziegelwanger, W. Kreuzer, and P. Majdak, "Mesh2hrtf: Open-source software package for the numerical calculation of head-related transfer functions," in *22nd International Congress on Sound and Vibration*, 2015.

[ZMK15] H. Ziegelwanger, P. Majdak, and W. Kreuzer, "Numerical calculation of listener-specific head-related transfer functions and sound localization: Microphone model and mesh discretization," *The Journal of the Acoustical Society of America*, vol. 138, no. 1, pp. 208–222, 2015.

[ZRM13] H. Ziegelwanger, A. Reichinger, and P. Majdak, "Calculation of listener-specific head-related transfer functions: Effect of mesh quality," in *Proceedings of Meetings on Acoustics ICA2013*, vol. 19, no. 1. Acoustical Society of America, 2013, p. 050017.