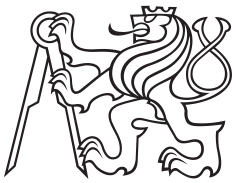
Master's Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Radioelectronics

Software for Visualization, Segmentation, and Sonification of Ultrasonic Vocalizations of Laboratory Rats

Bc. Antonín Gazda

Supervisor: prof. Ing. Roman Čmejla, CSc.
Field of study: Electronics and Communications
May 2025

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Gazda Antonín** Personal ID number: **499215**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Radioelectronics**

Study program: **Electronics and Communications**

Specialisation: **Audiovisual and Signal Processing**

II. Master’s thesis details

Master’s thesis title in English:

Software for Visualization, Segmentation and Sonification of Ultrasonic Vocalizations of Laboratory Rats

Master’s thesis title in Czech:

Software pro vizualizaci, segmentaci a sonifikaci ultrazvukových vokalizací laboratorních potkanů

Name and workplace of master’s thesis supervisor:

prof. Ing. Roman Čmejla, CSc. Department of Circuit Theory FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **05.02.2025** Deadline for master's thesis submission: _____

Assignment valid until: **20.09.2026**

doc. Ing. Stanislav Vítek, Ph.D.
Head of department’s signature

prof. Mgr. Petr Páta, Ph.D.
Vice-dean´s signature on behalf of the Dean

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work.
The student must produce his thesis without the assistance of others, with the exception of provided consultations.
Within the master’s thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt

Student’s signature

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Gazda Antonín** Personal ID number: **499215**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Radioelectronics**

Study program: **Electronics and Communications**

Specialisation: **Audiovisual and Signal Processing**

II. Master’s thesis details

Master’s thesis title in English:

Software for Visualization, Segmentation and Sonification of Ultrasonic Vocalizations of Laboratory Rats

Master’s thesis title in Czech:

Software pro vizualizaci, segmentaci a sonifikaci ultrazvukových vokalizací laboratorních potkanů

Guidelines:

Ultrasonic vocalizations of laboratory rats serve as a crucial communication tool in behavioral studies. This project focuses on developing software capable of analyzing recordings of ultrasonic communication and behavior in rats. The goal is to create a tool for segmenting and detecting these signals and enabling their sonification, supporting research into cooperative behavior and vocal communication in laboratory rats.
1. Conduct a literature review on existing approaches to analyzing ultrasonic vocalizations and evaluate software implementation possibilities. 2. Implement methods for segmenting and detecting ultrasonic vocalizations. 3. Evaluate the detection accuracy using available methods. 4. Implement sonification of ultrasonic signals from laboratory rats. 5. The software should allow easy signal editing and manipulation and be ready for the integration of potential recognition and classification modules.

Bibliography / sources:

[1] Pessoa, Diogo, et al. "Automatic segmentation and classification of mice ultrasonic vocalizations." The Journal of the Acoustical Society of America 152.1 (2022): 266-280.
[2] Premoli, Marika, et al. "Mouse and rat ultrasonic vocalizations in neuroscience and neuropharmacology: State of the art and future applications." European Journal of Neuroscience 57.12 (2023): 2062-2096.

FAKULTA ELEKTROTECHNICKÁ
FACULTY OF ELECTRICAL ENGINEERING
Technická 2
166 27 Praha 6

DECLARATION

I, the undersigned

Student's surname, given name(s): Gazda Antonín
Personal number: 499215
Programme name: Electronics and Communications

declare that I have elaborated the master's thesis entitled

Software for Visualization, Segmentation and Sonification of Ultrasonic Vocalizations of Laboratory Rats

independently, and have cited all information sources used in accordance with the Methodological Instruction on the Observance of Ethical Principles in the Preparation of University Theses and with the Framework Rules for the Use of Artificial Intelligence at CTU for Academic and Pedagogical Purposes in Bachelor's and Continuing Master's Programmes.

I declare that I used artificial intelligence tools during the preparation and writing of this thesis. I verified the generated content. I hereby confirm that I am aware of the fact that I am fully responsible for the contents of the thesis.

In Prague on 21.05.2025

Bc. Antonín Gazda
.....
student's signature

Acknowledgements

I would like to sincerely thank my supervisor, prof. Ing. Roman Čmejla, CSc., for his invaluable guidance, encouragement, and support throughout the development of this thesis.

I would also like to express my gratitude to RNDr. Tomáš Petrásek, Ph.D., for his helpful insights and consultations.

Abstract

This thesis presents *Squeak Peek Studio*, a user-friendly MATLAB application for analyzing ultrasonic vocalizations (USVs) in laboratory rats. Designed for intuitive use and integration into existing research workflows, the software combines segmentation, visualization, sonification, and evaluation in a unified interface. It implements multiple detection algorithms - including threshold-based and Bayesian methods - with tunable parameters and interactive editing. A phase vocoder-based sonification module and video synchronization enable deeper behavioral interpretation. The tool aims to streamline USV analysis and support both automated processing and human-in-the-loop refinement.

Keywords: ultrasonic vocalizations, rat communication, acoustic behavior analysis, MATLAB App Designer, signal segmentation, Bayesian changepoint detection, phase vocoder, sonification, automated segmentation, software tool development, bioacoustic research

Supervisor: prof. Ing. Roman Čmejla, CSc.
Czech Technical University in Prague,
Technická 1902/2,
166 27 Praha 6

Abstrakt

Tato práce představuje *Squeak Peek Studio*, uživatelsky přívětivou MATLAB aplikaci pro analýzu ultrazvukových vokalizací (USV) laboratorních potkanů. Software je navržen pro intuitivní použití a integraci do stávajících výzkumných pracovních postupů. Kombinuje segmentaci, vizualizaci, sonifikaci a vyhodnocení v jednotném rozhraní. Implementuje několik detekčních algoritmů - včetně prahovacích a bayesovských metod - s laditelnými parametry a možností interaktivní úpravy. Modul sonifikace založený na fázovém vokodéru a synchronizace videa umožňuje hlubší interpretaci chování. Cílem nástroje je zefektivnit analýzu USV a podpořit jak automatizované tak i manuální zpracování.

Klíčová slova: ultrazvukové vokalizace, komunikace potkanů, analýza akustického chování, MATLAB App Designer, segmentace signálu, bayesovská detekce, fázový vokodér, sonifikace, automatizovaná segmentace, vývoj softwaru, bioakustický výzkum

Překlad názvu: Software pro vizualizaci, segmentaci a sonifikaci ultrazvukových vokalizací laboratorních potkanů

Contents

Figures

Tables

Part I

Introduction

Chapter 1

Introduction

Ultrasonic vocalizations (USVs) are high-frequency acoustic signals produced by numerous animal species, with rodents being among the most studied. These vocal emissions serve as essential communicative cues in both social and affective contexts. In laboratory rats, USVs are particularly valuable for behavioral neuroscience, offering insight into emotional states, mating behavior, social hierarchy, and the physiological effects of drugs. Their study is instrumental in developing models of neurological and psychiatric disorders, including schizophrenia, anxiety, depression, and autism spectrum conditions.

However, the analysis of USVs presents substantial technical challenges. Since they occur far above the range of human hearing, these signals require specialized methods for recording, processing, and interpretation. Manual annotation remains a widely used approach but is both time-consuming and susceptible to human bias and error. Additionally, many existing software tools for USV analysis are either fragmented in functionality, overly complex for less technical users, or lack features.

This thesis presents *Squeak Peek Studio*, a MATLAB-based application developed to address these limitations by integrating key aspects of USV analysis into a cohesive and user-friendly platform. The application offers automated segmentation using multiple detection algorithms, dynamic visualization through synchronized waveform and spectrogram displays, and sonification that renders ultrasonic content audible through time-stretching and pitch-shifting. Furthermore, it includes a manual editing interface, detection performance evaluation tools, and a video synchronization module that allows for audiovisual synchronization of USVs with behavioral footage.

The software has been designed to be accessible to users without extensive programming expertise, while remaining extensible for future integration of advanced methods such as denoising or classification. By streamlining the USV analysis workflow - from data import to export-ready results - *Squeak Peek Studio* aims to improve the efficiency and reproducibility.

The following chapters provide an overview of the theoretical background, describe the software implementation in detail, present experimental results, and conclude with a summary and future outlook.

Part II

Theoretical Part

Chapter 2

Ultrasonic Vocalizations

2.1 Definition and Characteristics of USVs

Ultrasonic vocalizations (USVs) are high-frequency acoustic signals, typically ranging from 30 to 110 kHz, produced by rats in a wide range of social and affective contexts. These calls vary in spectral and temporal structure and serve as communicative signals that reflect emotional state, arousal, and motivational drives. [1]

Three major families of USVs have been characterized in rats, each with distinct acoustic properties and behavioral significance:

- **40-kHz USVs** are emitted primarily by pups between 30–70 kHz, typically in response to maternal separation. These vocalizations are considered markers of distress or negative affect and can also be triggered by handling or brief social contacts. [1]
- **22-kHz USVs** are long-duration, low-frequency calls (typically 20–33 kHz, >300 ms) produced by juveniles and adults in aversive situations such as exposure to predators, aggressive conspecifics, social defeat, or drug withdrawal. These vocalizations are widely regarded as indicators of negative arousal and emotional states, potentially analogous to human expressions of distress. [1]
- **50-kHz USVs** are short (typically <40 ms), high-frequency calls ranging from 35 to 80 kHz, often occurring in appetitive or rewarding contexts. These include social play, mating, anticipation of rewards, and administration of dopaminergic drugs. They may be flat or frequency-modulated (FM), with FM calls more strongly associated with positive affect. [1]

2.2 Behavioral and Neurological Relevance

Rats are highly social animals, and USVs serve as key behavioral markers in both social and non-social contexts. The emission of USVs reflects underlying arousal and emotional states, making them valuable in studies of affectivity, perception, and sociability.

The 22-kHz USVs are reliably evoked by aversive stimuli and mark negative emotional valence, such as anxiety or threat. Their emission is modulated by prior experience and can be elicited in both social encounters (e.g., aggression, unfamiliar humans) and non-social threats (e.g., air puffs, drug withdrawal). In contrast, 50-kHz USVs are associated with positively valenced states. They are enhanced by social play, rewarding interactions with familiar humans, and psychostimulant exposure - making them strong indicators of positive emotional arousal.

Sex, age, strain, and experimental history all influence the structure and frequency of USVs. Male and female rats show different vocal repertoires during mating or social exploration. Moreover, the quantity and acoustic features of USVs are sensitive to pharmacological modulation and are increasingly used as translational biomarkers in models of psychiatric and neurodevelopmental disorders. [1]

By encoding emotional and motivational states in their acoustic structure, USVs offer a non-invasive, temporally precise window into rodent behavior. This makes them an increasingly important tool in behavioral neuroscience, neuropharmacology, and affective research.

2.3 Applications in Preclinical Research

The sensitivity of ultrasonic vocalizations (USVs) to genetic, environmental, and pharmacological factors makes them valuable behavioral biomarkers in translational neuroscience. USVs are widely utilized in preclinical models of:

- **Autism Spectrum Disorder (ASD):** Models often exhibit reduced call rates during early development or social interactions. [2, 3]
- **Parkinson's Disease and Aging:** Changes in vocal motor control and call rates may reflect nigrostriatal degeneration. [4]
- **Anxiety and Depression:** Shifts in the balance of 22-kHz vs. 50-kHz call production reflect emotional valence and are modulated by anxiolytic or antidepressant treatments. [5]
- **Neurodevelopmental Delay:** Longitudinal USV recordings enable early detection of communication impairments that may not be evident in standard behavioral tests. [3]

As automated detection methods continue to improve, USVs provide a scalable and non-invasive approach to phenotype affective and social behaviors across time and experimental conditions. [4, 5]

2.4 Existing Software Tools for USV Analysis

The analysis of ultrasonic vocalizations (USVs) in rodents has been greatly facilitated by the development of various software tools. These tools employ a range of methodologies, from traditional signal processing to advanced machine learning techniques, to detect, classify, and analyze USVs. Below is an overview of prominent software tools used in USV research.

2.4.1 DeepSqueak

DeepSqueak is a MATLAB-based software that utilizes deep learning algorithms, specifically Faster Region-Based Convolutional Neural Networks (Faster-RCNN), for the detection and analysis of USVs. It offers both supervised and unsupervised classification methods, allowing for flexible analysis of vocalization data. DeepSqueak's graphical user interface (GUI) enables users to visualize spectrograms, manually review detections, and perform syntax analysis of vocal sequences. [6]

2.4.2 MUPET (Mouse Ultrasonic Profile ExTraction)

MUPET is an open-source MATLAB tool designed for the rapid and unsupervised analysis of mouse USVs. It employs signal processing techniques to extract and cluster syllable types from audio recordings. MUPET allows for the comparison of vocal repertoires across different experimental conditions and supports the export of data for further statistical analysis. Its unsupervised approach facilitates the discovery of novel vocalization patterns without prior labeling. [7]

2.4.3 TrackUSF

TrackUSF is an automated, high-throughput tool for analyzing USVs across various species, including mice, rats, and bats. Unlike traditional methods that focus on discrete syllable detection, TrackUSF analyzes continuous audio data by segmenting it into short fragments and extracting Mel-frequency cepstral coefficients (MFCCs) for each segment. This approach allows for the comparison of vocalizations between groups without prior assumptions about call structure, making it particularly useful for studies involving animal models of neurodevelopmental disorders. [8]

2.4.4 UltraVox XT

UltraVox XT, developed by Noldus Information Technology, is a commercial software package designed for the detection and analysis of rodent USVs. It provides real-time visualization of waveforms and spectrograms, batch processing capabilities, and integration with other behavioral analysis tools like EthoVision XT. UltraVox XT allows users to define call detection parameters

and categorize calls based on user-defined labels, facilitating the study of vocalizations in various behavioral contexts. [9]

2.4.5 USVSEG

USVSEG is a MATLAB-based tool focused on the robust segmentation of rodent USVs from continuous recordings, even in the presence of background noise. It utilizes a stable version of the sound spectrogram and additional signal processing techniques to enhance the separation of vocal signals. USVSEG provides precise time tracking of spectral peaks within each syllable and outputs data that can be used for further analysis of call features. [10]

2.4.6 VocalMat

VocalMat is a software tool that applies image-processing and differential geometry approaches to detect and classify mouse USVs from audio recordings. It eliminates the need for user-defined parameters by using adaptive thresholding and machine learning algorithms to categorize USVs into distinct types. VocalMat has demonstrated high accuracy in both detection and classification tasks, making it suitable for high-throughput analysis of vocal repertoires. [11]

2.4.7 ContourUSV

ContourUSV is a recently developed automated system for detecting USVs from audio recordings. It includes a pipeline that encompasses spectrogram generation, cleaning, pre-processing, contour detection, post-processing, and evaluation against manual annotations. ContourUSV has been shown to outperform other state-of-the-art systems in terms of precision, recall, F1 score, and specificity, while also achieving significant speed improvements. [12]

2.5 Goals of Automated USV Analysis

Automated USV analysis aims to process ultrasonic vocalization recordings in a standardized, scalable, and reproducible way. The primary goals of these tools include:

- **Segmentation**: Identifying the onset and offset of individual USV syllables from continuous audio recordings. Accurate segmentation is essential for meaningful downstream analysis.
- **Classification**: Grouping detected vocalizations into distinct types based on their acoustic features (e.g., flat, frequency-modulated, trill). This facilitates behavioral interpretation and phenotype comparisons.

- **Evaluation**: Computing metrics such as call rate, call duration, inter-call intervals, and frequency distribution. These are used as behavioral biomarkers in preclinical models.
- **Visualization and Playback**: Providing human-interpretable views of detected calls in time–frequency space and enabling sonification for auditory inspection.
- **Batch Processing and High-Throughput Capabilities**: Supporting the analysis of large datasets efficiently, which is essential for longitudinal or population studies.

By reducing the need for manual annotation, automated tools increase throughput and objectivity, making them indispensable in modern USV-based behavioral neuroscience.

2.6 Common Analysis Techniques Used

Existing tools use a variety of signal processing and machine learning approaches to achieve segmentation and classification. These include:

- **Spectrogram Thresholding**: Basic tools like USVSEG detect USVs by applying energy thresholds on spectrogram representations [10]. This approach is fast but can suffer from poor performance in noisy environments.
- **Template Matching**: Tools such as XBAT rely on matching detected spectrogram fragments to predefined templates [13]. While effective for structured calls, it lacks generalizability to varied datasets.
- **Convolutional Neural Networks (CNNs)**: DeepSqueak uses CNNs trained on labeled spectrogram segments to detect and classify USVs with high accuracy [6]. This method excels in noisy or variable data but requires large annotated datasets and substantial computational resources.
- **Clustering Algorithms**: MUPET and TrackUSF use unsupervised clustering (e.g., k-means or t-SNE) based on acoustic features to group vocalizations into syllable types [7, 8]. These methods do not require prior labels and are suited for exploratory analysis.
- **Image-Based Techniques**: VocalMat treats spectrograms as images and applies computer vision and geometric techniques for detection and classification [11]. This approach supports high-throughput analysis and parameter-free operation.

Each method offers trade-offs in speed, accuracy, and flexibility. Many tools now combine multiple techniques to maximize performance across diverse datasets.

Chapter 3

Visualization and Sonification of USVs

The analysis of ultrasonic vocalizations (USVs) benefits significantly from both visual and auditory representations. While automated detectors provide scalability, visualization and sonification serve as essential tools for exploratory analysis, intuitive validation, and interaction with detected acoustic events.

3.1 Spectrogram Visualization

Spectrograms are the primary method for visualizing USVs. They display frequency over time, with amplitude encoded by color intensity. This representation makes it possible to detect patterns such as frequency modulation, duration, and call sequences, which are critical for behavioral interpretation.

Spectrograms are computed using the short-time Fourier transform (STFT), with overlapping windows providing a balance between time and frequency resolution. High-resolution spectrograms allow precise inspection of transient events, especially in dense or noisy recordings.

3.2 Visualization in Existing Tools

Graphical interfaces in tools like DeepSqueak, MUPET, and TrackUSF provide synchronized waveform and spectrogram views, overlaid with USV detections and annotations. These overlays include bounding boxes, frequency contours, and color-coded class labels. Some tools also support video synchronization or dimensionality reduction for vocal cluster analysis.

Such interactive tools are valuable for reviewing detection output, correcting errors, and correlating vocal events with behavioral observations. They play a crucial role in ensuring reproducibility and interpretability in USV research pipelines.

3.3 Sonification of Ultrasonic Vocalizations

Because rat vocalizations lie well above the range of human hearing, typically (30–110 kHz) [1], sonification is necessary to render them audible. Sonification

is defined as "the transformation of data relations into perceived relations in an acoustic signal for the purposes of facilitating communication or interpretation" [14]. It is increasingly recognized as a perceptual and analytical tool, not just a playback convenience.

The simplest sonification method is time-stretching by lowering the playback sampling frequency to bring the USVs into audible range. However, this method couples pitch and time, which may not always be desirable. More sophisticated methods, such as the *phase vocoder*, address this limitation by independently controlling pitch and time.

3.3.1 Phase Vocoder for Time-Stretching and Pitch-Shifting

The phase vocoder is a signal processing technique that enables independent pitch shifting and time stretching of audio while preserving the signal's spectral characteristics. It is particularly well-suited for USV sonification, as it renders ultrasonic signals audible without compromising their frequency modulations or temporal detail.

Both pitch and time scaling are precisely controllable - pitch can be adjusted in musical units such as semitones (ST), while playback speed can be stretched without affecting pitch - making the method highly flexible.

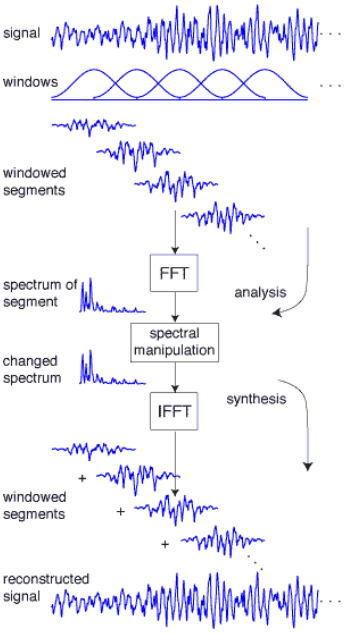


Figure 3.1: Phase vocoder functionality [15].

The phase vocoder operates as follows:

1. The signal is windowed and divided into overlapping frames.
2. Each frame is transformed via FFT into spectral data.
3. Time/pitch modifications are applied to the spectrum.
4. The result is transformed back with IFFT.
5. The time-domain signal is reconstructed using overlap-add.

3.4 Sonification as a Perceptual Tool

The human auditory system excels at detecting fine-grained variations in frequency, rhythm, and texture, making it especially well-suited for interpreting

dynamic acoustic phenomena such as ultrasonic vocalizations (USVs). [14]

In this thesis, sonification is achieved via a phase vocoder, which transposes USVs into the audible range while retaining their temporal and spectral structure. This enables researchers to audibly inspect the USV features like tremor, repetition, or variability in call patterns. This may be overlooked in spectrograms or lost in automated detection. Within *Squeak Peek Studio*, auditory playback complements visual representations, making it a more intuitive, perceptually rich, and human-centered approach to USV analysis.

Chapter 4

Segmentation Methods and Theoretical Approaches

4.1 Importance of Segmentation in the USV Pipeline

Segmentation is the essential first step in ultrasonic vocalization (USV) analysis. Its purpose is to isolate individual vocalizations in continuous audio recordings. Accurate segmentation directly affects the reliability of all subsequent analyses, including classification, behavior mapping, and statistical evaluation of call features.

Manual segmentation is time-consuming, varies based on the researcher, and is impractical for large datasets. Automated segmentation methods are essential for improving reproducibility, analytical consistency, and processing efficiency.

Methods used for automatic segmentation in this thesis fall into two categories: threshold-based signal processing and statistical changepoint detection.

4.2 Threshold-Based Approaches

4.2.1 Power Spectral Density (PSD) Detection

A common approach to segmentation is based on power spectral density (PSD) estimation. This method assumes that USVs have higher spectral energy than background noise in the ultrasonic range. The signal is divided into overlapping windows using short-time Fourier transform (STFT), and the spectral power in each frame is evaluated against a fixed or adaptive threshold.

If a frame's energy exceeds the threshold in the target frequency band (e.g., 30 - 110 kHz), it is labeled as a potential vocalization. Post-processing is applied to merge close detections and remove short false positives.

This approach is computationally efficient and easy to implement, but highly sensitive to environmental noise and may miss low-energy calls or

detect noise artifacts as vocalizations.

4.3 Bayesian Changepoint Detection

Changepoint detection methods aim to identify points in a signal where its statistical properties undergo abrupt transitions. In the context of ultrasonic vocalizations (USVs), these points typically correspond to vocal onset and offset boundaries.

4.3.1 Bayesian Autoregressive Changepoint Detection (BACD)

This thesis utilizes a Bayesian autoregressive changepoint detection method (BACD), originally proposed by Čmejla et al. [16], and adapted for the segmentation of ultrasonic vocalizations. The method assumes that a signal can be represented as a sequence of piecewise stationary autoregressive (AR) processes. At each potential changepoint, it evaluates the difference between adjacent AR models to compute the likelihood of a boundary.

The algorithm performs the following steps, is described in Figure 4.1:

- A sliding window moves across the signal, centered on the candidate point.
- For each position, the signal is modeled by two AR processes: one before and one after the candidate changepoint.
- Bayesian inference is used to estimate a posterior probability of a changepoint occurring at that position. This involves marginalizing over the AR parameters and normalizing by Bayesian evidence.
- The resulting posterior probability curve is filtered and thresholded to extract likely segment boundaries.

The model makes no assumptions about noise variance or specific signal features, making it well-suited for complex, real-world acoustic data such as USVs. By normalizing the posterior probability using Bayesian evidence, the method enables comparison across different signal regions and avoids bias from window size or signal energy. The formulation is inspired by the general linear model, with model fit assessed via log-likelihood ratios between hypotheses of change vs. no change. [16]

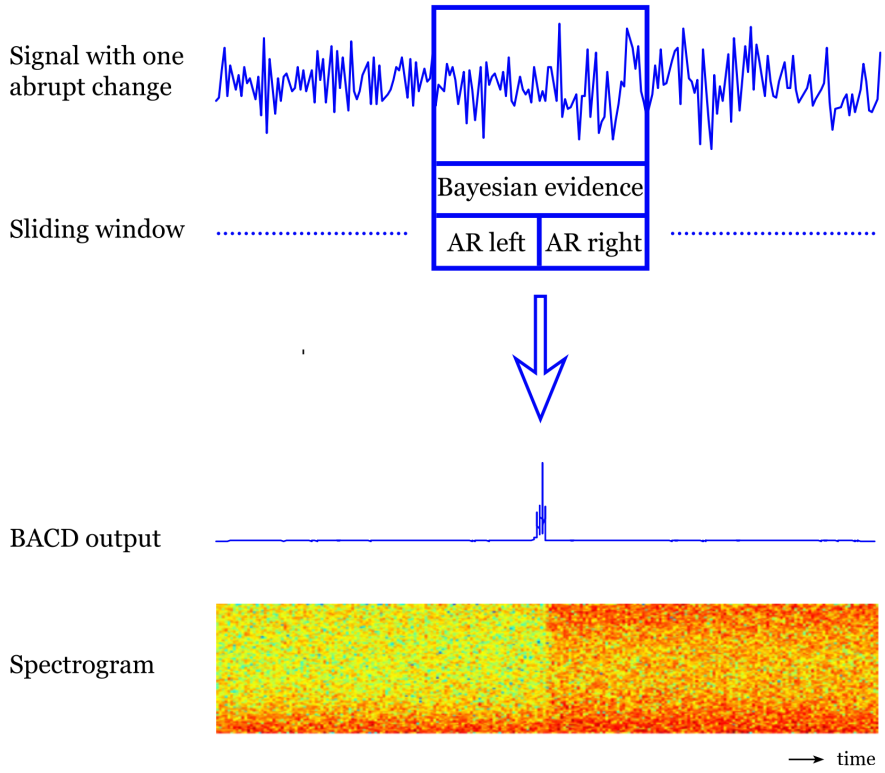


Figure 4.1: Principle of the BACD [16]

4.3.2 Recursive Bayesian Detection (RBD)

Recursive Bayesian Detection (RBD) is an extended formulation of the Bayesian autoregressive changepoint detector that aims to improve segmentation smoothness and reliability through repeated Bayesian evaluation across a sliding window. Unlike traditional BACD, which evaluates each candidate point independently, RBD accumulates evidence recursively by computing posterior probabilities for multiple possible changepoints within the window at each step.

This formulation increases robustness in variable or noisy signal conditions and can produce a smoother changepoint probability curve. However, it comes at the cost of increased computational load: RBD requires multiple full Bayesian evaluations per window and does not benefit from the incremental efficiency typical of other recursive filters.

As a result, RBD is significantly slower than BACD in practice, especially for long recordings.

Chapter 5

Theoretical Summary and Research Motivation

5.1 Motivation

Recent tools such as DeepSqueak [6], MUPET [7], and TrackUSF [8] have advanced the field of ultrasonic vocalization (USV) analysis, offering powerful frameworks for automated detection, classification, and visualization. However, despite these contributions, several key challenges remain:

- **Fragmented Functionality:** Many tools focus on isolated parts of the analysis pipeline - such as segmentation, clustering, or visualization - lacking end-to-end integration. Researchers must manually transfer data between tools or write custom scripts to complete the workflow.
- **Limited Interactivity:** User interfaces are often static or batch-oriented, offering little real-time feedback or flexibility for manual refinement of detections and classification.
- **Underdeveloped Sonification Capabilities:** While visual representations are common, real-time auditory inspection - especially is rarely implemented, limiting perceptual access to acoustic nuance.
- **High Technical Barrier:** Tools with advanced performance, particularly deep learning-based systems, frequently require programming expertise, environment setup, and curated training data, limiting accessibility.

These limitations highlight the need for a modular, user-friendly, and extensible platform that balances analytical depth with accessibility. This thesis introduces *Squeak Peek Studio*, a MATLAB-based application that unifies the complete USV analysis workflow in a single, interactive environment.

Part III

Practical Part

Chapter 6

Implementation and Evaluation

Ultrasonic vocalizations were recorded at the National Institute of Mental Health (NIMH) in Czechia by RNDr. Tomáš Petrásek, Ph.D., and his research team. These recordings were provided to serve as the primary dataset used for developing, testing, and validating visualization, segmentation, detection and sonification tools in this thesis.

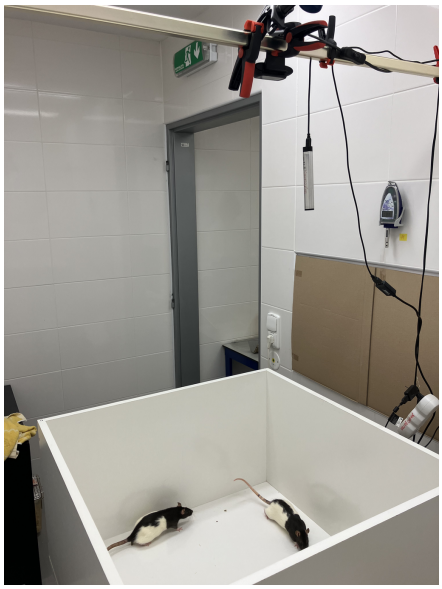
A total of five recordings, each approximately 10 minutes long, were provided for analysis, 6176 labeled ultrasonic vocalization segments in total. Each recording was accompanied by a set of manually annotated label files specifying the start and end times, frequency bounds, and classification of individual ultrasonic vocalizations. These labels served as ground truth for evaluating segmentation accuracy and supporting further classification analysis.

Additionally, a corresponding video recording was provided for each USV recording. As part of this thesis, a dedicated tool was developed to facilitate the sonification of USV data and to synchronize it with the associated video footage. Further details on the recording setup, labeling format, video structure, and implementation of the synchronization tool are presented in the following sections.

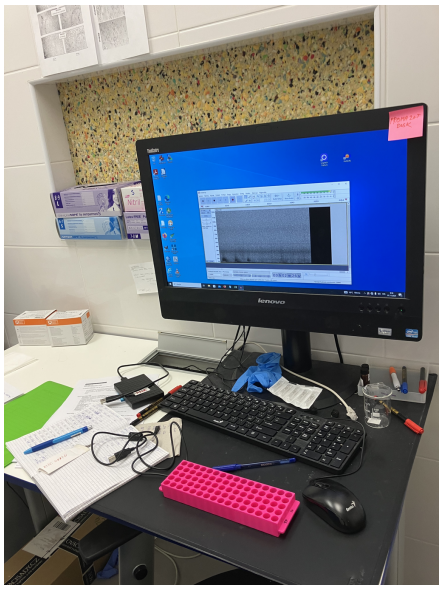
6.1 USV Recording

Figure 6.1 shows the measurement setup used for acquiring the data. An off-the-shelf ultrasonic microphone was suspended above a box containing laboratory rats. The microphone was connected to a Windows computer via USB and recordings were captured using Audacity, a widely used open-source audio recording and editing program.

All recordings were acquired using the Dodotronic Ultramic UM250K [17], a high-frequency USB microphone specifically designed for capturing ultrasonic animal vocalizations. Key technical specifications are listed in Table 6.1. The device's sampling frequency of 250 kHz enables accurate recording of signals up to 125 kHz, which fully encompasses the typical frequency range of rodent ultrasonic vocalizations. [1]



(a) : Subject recording enclosure



(b) : Recording station

Figure 6.1: Recording setup used at NIMH for capturing ultrasonic vocalizations from laboratory rats.

Sampling Frequency	250 kHz
Resolution	16-bit
PC Interface	Full-speed USB 2.0
Dimensions	130 × 20 mm

Table 6.1: Technical specifications of the Dodotronic Ultramic UM250K. [17]

6.2 Label Annotations

The researchers at the National Institute of Mental Health (NIMH) provided annotated label files indicating the presence of ultrasonic vocalizations (USVs) within the raw audio recordings. These labels were created in Audacity using the *Extended format with frequency ranges* [18], which allows each label to store the start time, end time, start frequency, end frequency and the vocalization classification. This format facilitates accurate alignment of vocalizations in time–frequency representations and supports subsequent classification and analysis.

An example of a labeled USV entry is shown below:

```
58.572189     58.602656     5w
\      57717.675781      63324.539063
```

The first line indicates the start and end times (in seconds) and the classification label of the vocalization. The second line specifies the corresponding start and end frequencies (in Hz), enabling precise localization in a spectrogram.

This labeling format has been previously used in studies conducted at NIMH, which allowed seamless reuse of their annotated data in this project. Moreover, since the format is defined by an open-source application, it ensures long-term compatibility. The software developed in this thesis maintains this format for its outputs, allowing generated labels to be directly reused in Audacity or external analysis pipelines.

An automatic classification methodology is currently being researched and implemented by Ing. Karolína Bendová, a PhD student at CTU FEE and researcher at NIMH. The classification scheme used by the annotators includes five distinct label types, summarized in Table 6.2. These labels serve as ground truth for evaluating automated detection algorithms and as training input for future classification systems.

Label	Description
sk	Finger snap - used for synchronizing audio with video footage
5	Unspecified high-frequency vocalization
c5	Composite call - overlapping or structurally complex vocalization
5t	Trill - a rapid, rhythmic modulation in frequency
5w	Wavy - a sinusoidal or modulated pattern in frequency

Table 6.2: Classification labels and their corresponding interpretations.

To facilitate working with Audacity label files in MATLAB, two utility functions - `importLabels` and `exportLabels` - were developed as part of this thesis. These functions enable seamless conversion between Audacity's extended label format and structured MATLAB data, allowing researchers to easily work with vocalization labels inside MATLAB.

The `importLabels` function takes as input the path to a label file and the sampling frequency (`labelPath`, `fs`), and returns a structured array containing the following fields for each label:

- `StartTime` – start of the vocalization in seconds
- `EndTime` – end of the vocalization in seconds
- `Label` – classification tag (e.g., `5w`, `5t`)
- `StartFrequency` – lower frequency bound in Hz
- `EndFrequency` – upper frequency bound in Hz
- `StartIndex` – start of the vocalization as an index in the USV recording
- `StopIndex` – end of the vocalization as an index in the USV recording

The `exportLabels` function performs the inverse operation. It takes a structured label array and an output path (`labels`, `filePath`), and saves the labels back to a text file in Audacity-compatible format.

Both functions are included in the project's source code. They can be readily adapted for future research that involves importing or exporting annotations using Audacity's label format within MATLAB workflows.

The ability to import and export Audacity-compatible label files directly in MATLAB enabled the use of annotated data as ground truth for evaluation. With this foundation in place, segmentation algorithms were developed and systematically tested on the provided dataset.

The following section describes the implemented detection algorithms designed to identify ultrasonic vocalizations (USVs) within the audio recordings. These algorithms can be applied directly to raw audio or to preprocessed (denoised) signals, enabling flexible evaluation of different signal conditions.

6.3 Segmentation Algorithms

One of the core tasks of this thesis was the automatic segmentation of ultrasonic vocalizations (USVs) from provided audio recordings. In this context, segmentation refers to the process of detecting the boundaries of individual vocalizations within a noisy signal, without prior knowledge of their exact timing or duration.

Researchers at NIMH work with large volumes of such recordings, but the need for manual labeling or reliance on existing tools - often not optimized for their specific workflows - can be a bottleneck to the pace of their research.

To address this, three different segmentation algorithms were implemented and tested. They can work with both raw and denoised audio and are designed to detect vocalizations with start and end times that closely match the manual labels.

The three implemented methods are:

- **PSD-based Segmentation** - a thresholding approach based on Power Spectral Density estimates.
- **BACD-based Segmentation** - a Bayesian Autoregressive Changepoint Detection method.
- **RBD-based Segmentation** - a Recursive Bayesian Autoregressive Changepoint Detection algorithm.

6.3.1 PSD-based Segmentation

The first implemented segmentation method is based on Power Spectral Density (PSD) analysis. The algorithm identifies USV candidates as high-energy events within a specified frequency band by analyzing the spectro-temporal distribution of signal power.

The audio signal is first normalized and, optionally, restricted to a defined Region of Interest (ROI).

A zero-phase bandpass filter is applied to isolate the ultrasonic frequency range - typically between 30 kHz and 110 kHz. The filtered signal is then

transformed using the short-time Fourier transform (STFT) to compute its spectrogram. The power envelope is obtained by summing the squared magnitude of spectral bins within the frequency band of interest across time. This envelope is smoothed using a moving average filter and normalized to lie between 0 and 1.

To enhance transient activity, a local noise floor is estimated using a moving minimum filter, and an effective envelope is calculated by subtracting the noise estimate. A local signal-to-noise ratio (SNR), local mean, and local standard deviation are computed. These are combined into a dynamic threshold that adapts based on signal characteristics using the formula:

$$\text{Threshold}(t) = \frac{\mu(t) + k \cdot \sigma(t)}{1 + w \cdot \text{SNR}(t)}$$

where $\mu(t)$ and $\sigma(t)$ are the local mean and standard deviation of the effective envelope, and k and w are empirically tuned parameters.

Segments exceeding the threshold are identified as candidate USVs. For each detected segment, the start and end times are recorded along with corresponding sample indices. A final filtering step discards segments whose mean effective power falls below a predefined minimum.

The algorithm is implemented in MATLAB as the function `PSDDetector`, which supports several configurable parameters including frequency cutoffs, window size, overlap factor, smoothing window, and threshold sensitivity. It accepts a raw or denoised USV recording and sampling frequency as inputs and returns a structured array of detected vocalizations with timing and indexing information.

The performance of this method was evaluated on the full dataset, and results are presented in Section 6.5.

6.3.2 BACD-based Segmentation

The second implemented segmentation method is based on Bayesian Autoregressive Changepoint Detection (BACD). The algorithm identifies USV candidates as transitions in the energy structure of the signal, based on statistical evidence of changepoints within a moving analysis window.

The audio signal is first normalized and, optionally, restricted to a defined Region of Interest (ROI).

A zero-phase bandpass filter is applied to isolate the ultrasonic frequency range - typically between 30 kHz and 110 kHz. The filtered signal is squared to emphasize power variations, and the resulting waveform is passed to a custom changepoint detector implemented in the function `bacd.m`. [16] This function calculates a time-varying evidence signal representing the likelihood of structural change, based on Bayesian model comparison of linear models.

The BACD output is then smoothed using a moving average filter, producing a power envelope that emphasizes transitions corresponding to potential vocalizations. A static threshold - based on the mean value of the envelope - is applied to identify above-threshold regions, which are treated as candidate USVs.

The algorithm is implemented in MATLAB as the function `BACDDetector`, which supports parameters such as BACD window length, smoothing window size, minimum event duration, maximum merge gap, and optional plotting. It takes a path to a USV audio file as input and returns a structured array of detected vocalizations with start and end times.

The performance of this method was evaluated on the full dataset, and results are presented in Section 6.5.

6.3.3 RBD-based Segmentation

The final implemented segmentation algorithm is based on Recursive Bayesian Autoregressive Changepoint Detection (RBD). This method improves upon standard changepoint detection by modeling the local dynamics of the signal using autoregressive (AR) models and tracking changes in these dynamics over time.

The input audio signal is normalized and optionally restricted to a Region of Interest (ROI). A bandpass filter is typically applied beforehand, although this step can be skipped if denoised recordings are used. The core of the algorithm involves recursively computing the Bayesian evidence of a changepoint at each time sample using autoregressive modeling from both sides of a sliding analysis window.

This is performed in the function `RBD`, which computes a scalar score proportional to the likelihood of a changepoint. The algorithm uses three key parameters: the AR order on the left and right of the changepoint, the AR order used for Bayesian evidence estimation, and the window length.

The resulting RBD output is normalized and smoothed using a moving average filter. A dynamic threshold is then computed, scaled by a selected factor. Segments where the smoothed RBD exceeds both the dynamic threshold and a fixed amplitude threshold are marked as candidate USVs.

A final step extracts start and end times for each above-threshold region and removes spurious detections. These boundaries are written to disk in label format and can be used for playback or comparison with ground truth.

The segmentation logic is implemented in MATLAB as the function `RBDDetector`, which internally calls `RBD` to compute the detection envelope. Parameters include AR model orders, window size, smoothing factors, and amplitude thresholds.

This method tends to produce smoother detection scores and improved temporal consistency compared to BACD, particularly in variable acoustic environments. Evaluation results are presented in Section 6.5.

Bayesian Optimization of Detector Parameters

To improve segmentation performance and systematically tune detection parameters, a Bayesian optimization framework was implemented in MATLAB. While not exposed in the graphical user interface, this functionality is available in the source code and can be used programmatically by advanced users.

The optimization process is built around MATLAB's `bayesopt` function and is designed to maximize segmentation accuracy - quantified by the F1 score with known ground-truth labels. The evaluation is performed using the `compareLabels` method. Each detector (PSD, BACD, and RBD) has an associated objective function tailored to its specific parameter set.

This modular approach allows researchers to adapt detector configurations to varying acoustic conditions or dataset-specific properties. Instead of requiring extensive manual labeling, users can annotate only a small, representative subset of their data. These annotations serve as the optimization target for tuning the detector's parameters using Bayesian inference.

Once optimal parameters are identified, they can be applied across the entire dataset, enabling automated segmentation that is customized to the experimental setup. This integration of Bayesian optimization into the development process significantly streamlines detector calibration and improves consistency across large-scale USV analyses.

6.3.4 Post-processing: Merging and Filtering Labels

A common challenge in automated USV segmentation is the generation of fragmented or spurious detections. Some detectors tend to split a single vocalization into multiple adjacent segments, while others may produce numerous extremely short detections caused by noise fluctuations or brief signal modulations. These issues reduce the clarity and interpretability of the resulting annotations and can negatively impact downstream analysis.

To address this, two post-processing functions were implemented in MATLAB:

- **`mergeCloseLabels`** - This function scans through the detected label sequence and merges consecutive labels if the temporal gap between them is smaller than a user-defined threshold. This helps ensure that a continuous vocalization is not mistakenly segmented into multiple parts.

- **`removeShortLabels`** - This function removes any labels whose duration falls below a predefined minimum threshold. Such very short segments are unlikely to correspond to meaningful vocalizations and are typically the result of noise or transient detection artifacts.

These post-processing steps significantly improve the quality of the output by reducing over-segmentation and eliminating low-confidence detections. Importantly, they are integrated into the *Squeak Peek Studio* application as optional modules within the Detection panel, allowing users to easily apply them without modifying the underlying detection algorithms.

This design choice reflects common issues observed in other detection tools, where such refinements are either absent or require external scripting. By embedding post-processing into the detection workflow, this project provides a more robust and researcher-friendly solution for USV analysis.

6.4 Segmentation Evaluation

To assess the accuracy of each segmentation algorithm, a custom MATLAB function named `compareLabels` was implemented. This function evaluates how well the automatically detected vocalization segments match the manually annotated ground-truth labels using the Midpoint Criterion. It is included in the project source code and can be readily reused in future research involving the same label format.

6.4.1 Midpoint Criterion

The Midpoint Criterion considers a detected segment correct if the midpoint of a ground-truth label lies within its duration. This simple yet robust rule ensures that the central portion of the vocalization is captured, while remaining tolerant to small shifts in onset or offset boundaries.

6.4.2 Calculated Metrics

The evaluation is carried out using the `compareLabels` function, which takes as input the paths to the ground-truth and detected label files, along with the sampling rate. Both sets of labels are imported using the `importLabels` function. The function then iterates through each ground-truth label, computes its midpoint, and checks whether it falls within any of the detected segments. Each detected segment can only be matched once.

Based on these matches, the following performance metrics are computed:

- **True Positives (TP):** Ground-truth segments that are successfully matched by a detected segment.
- **False Positives (FP):** Detected segments that do not match any ground-truth segment.
- **False Negatives (FN):** Ground-truth segments that are not matched by any detection.

From these, the following standard metrics are calculated:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

All results reported in this thesis use this evaluation method with the Midpoint Criterion. The computed statistics for each detector were used to compare their performance across multiple recordings.

6.5 Segmentation Results

This section presents the detection results for the three implemented segmentation algorithms: PSD-based, BACD-based, and RBD-based. Each

algorithm was applied to a set of five labeled recordings, and their performance was evaluated using the Midpoint Criterion described earlier. The reported metrics include precision, recall, and the F1 score, calculated by comparing the detected segments with manually annotated ground-truth labels.

6.5.1 PSD-based Segmentation Results

Signal	Provided	Detected	Precision	Recall	F1
mH02-I04-USV	1451	1239	0.8967	0.7657	**0.8260**
mK01-L03-USV	1369	1125	0.9182	0.7546	**0.8284**
mL01-M02-USV	1426	996	0.8865	0.6192	**0.7291**
mO02-S03-USV	557	354	0.9096	0.5781	**0.7069**
mR03-T01-USV	1373	923	0.8223	0.5528	**0.6611**

Table 6.3: Performance of the PSD-based detector (Midpoint Criterion)

6.5.2 BACD-based Segmentation Results

The Bayesian Autoregressive Changepoint Detector (BACD) identifies signal transitions to segment USVs, allowing for higher sensitivity to subtle changes in the acoustic structure.

Signal	Provided	Detected	Precision	Recall	F1
mH02-I04-USV	1451	1791	0.7069	0.8725	**0.7810**
mK01-L03-USV	1369	1692	0.7323	0.9050	**0.8095**
mL01-M02-USV	1426	1655	0.7202	0.8359	**0.7738**
mO02-S03-USV	557	713	0.6438	0.8241	**0.7228**
mR03-T01-USV	1373	1662	0.6095	0.7378	**0.6675**

Table 6.4: Performance of the BACD-based detector (Midpoint Criterion)

6.5.3 RBD-based Segmentation Results

The Recursive Bayesian Autoregressive Changepoint Detector (RBD) extends the BACD method by modeling local signal dynamics, which can improve performance in more variable acoustic conditions.

Signal	Provided	Detected	Precision	Recall	F1
mH02-I04-USV	1451	1690	0.6769	0.7884	**0.7284**
mK01-L03-USV	1369	1576	0.6935	0.7984	**0.7423**
mL01-M02-USV	1426	1339	0.7483	0.7027	**0.7248**
mO02-S03-USV	557	521	0.6929	0.6481	**0.6698**
mR03-T01-USV	1373	1684	0.5416	0.6642	**0.5967**

Table 6.5: Performance of the RBD-based detector (Midpoint Criterion)

6.5.4 Comparison and Observations

Overall, the three segmentation algorithms demonstrate distinct trade-offs between precision and recall. The PSD-based detector consistently achieves the highest precision, indicating strong specificity and a low rate of false positives. However, this comes at the cost of recall, as it tends to miss a larger number of vocalizations.

In contrast, the BACD-based detector yields the highest recall across recordings, successfully capturing most annotated USVs, but with reduced precision due to a higher false positive rate.

The RBD-based detector offers a compromise between these extremes by incorporating autoregressive modeling, which slightly improves the balance between precision and recall. Its overall F1 scores generally fall between those of the PSD and BACD methods. These findings highlight the importance of choosing a segmentation algorithm based on the specific goals of the analysis - whether the priority is higher sensitivity or stronger selectivity.

Beyond detection accuracy, computational efficiency is also an important factor. On a tested machine (Apple MacBook Pro M2, 24 GB RAM), the PSD-based detector processed a 10-minute recording in approximately 1 minute. The BACD-based detector required around 5 minutes for the same task, while the RBD-based detector took roughly 30 minutes.

These durations can vary depending on system load, signal complexity, and parameter settings, but they provide a practical estimate of relative runtime. For time-sensitive or large-scale processing, the PSD-based method offers the greatest computational efficiency, whereas the RBD-based method, although significantly slower, may offer advantages in challenging acoustic conditions where segmentation robustness is critical.

6.6 Sonification Implementation

The phase vocoder described in the theoretical section was implemented in MATLAB and integrated into the Squeak Peek Studio application to enable interactive and accurate playback of ultrasonic vocalizations (USVs). The goal of the implementation was twofold: (1) to allow researchers to audibly inspect individual segments of ultrasonic signals directly within the app, and (2) to generate synchronized audiovisual outputs in which USVs can be heard alongside experimental video footage.

6.6.1 Interactive Audio Playback

Within the application, users can select a segment of the USV signal and trigger playback using a dedicated `Sonify` button. This functionality is implemented via the `sonifySegment` function, which extracts the selected portion of the audio, applies a bandpass filter to isolate the ultrasonic frequency range (typically 30–110 kHz), and then processes the signal using a phase vocoder.

The function allows for independent pitch shifting and time stretching through the specification of parameters such as:

- `ST` – pitch shift in semitones (e.g., –36),
- `slowdown` – time stretch factor (e.g., 4),
- `winLen` and `winHop` – window size and hop size used in FFT analysis.

This enables USVs to be mapped into the human auditory range while preserving their temporal structure. The resulting audio is normalized and returned as a time-domain waveform ready for playback at standard sample rates (e.g., 44.1 kHz).

6.6.2 Synchronized Video Generation

Beyond individual playback, the application also includes functionality for exporting synchronized video and sonification. This feature overlays the sonified USV signal onto the original video footage of the experiment, allowing for audiovisual interpretation of rat behavior in direct correspondence with their vocalizations. This is particularly useful for researchers seeking to analyze the relationship between social behavior and ultrasonic communication.

Chapter 7

Application

Developing a user-friendly yet functionally rich application was a central goal of this thesis. Ultrasonic vocalizations (USVs) of laboratory rats are a key focus in behavioral neuroscience, offering insight into social interactions, affective states, and cognitive processes. The aim of the software was to create a unified tool for USV analysis - enabling segmentation, detection, sonification, visualization, and manual editing of ultrasonic signals.

7.1 Design Requirements

The following key design requirements were defined at the beginning of the project to guide the development of the application:

1. **Visualization Tools:** The GUI should provide synchronized time-domain and spectrogram visualization, enabling detailed inspection of the signal.

2. **Accurate USV Detection:** The application should support multiple algorithms for automatic segmentation and detection of ultrasonic vocalizations, with configurable parameters and support for batch processing.

3. **Label Editing:** Users should be able to interactively modify existing labels, including adjusting segment boundaries, manually editing classification labels, and accepting or rejecting individual events. In addition, the application should allow users to add new labels manually by selecting regions of interest directly within the spectrogram display.

4. **Sonification for Auditory Feedback:** Vocalizations should be playable using high-quality sonification, including pitch shifting and time stretching via a phase vocoder. This feature must support interactive playback as well as audio synchronized with video recordings.

5. **Video Integration:** The application must allow synchronization of USV audio with corresponding experiment video, enabling users to generate audiovisual outputs where rat vocalizations are audible alongside behavioral footage.

6. **Settings and Preferences:** The application must support saving and loading of user settings to ensure consistent behavior across sessions and reproducibility in analysis.

7. **Ease of Use:** The user interface must be intuitive and accessible to researchers without programming experience. All key functions should be reachable through clearly labeled controls and organized into logical panels.

8. **System Compatibility and Future-Proofing:** The application should be compatible with the current data formats used by researchers at NIMH, including support for their labeling conventions and file structures. Additionally, the system must be designed to accommodate future integration of new detection algorithms and automatic classification methods with minimal modification to the core architecture.

These requirements shaped the structure and functionality of the resulting application, ***Squeak Peek Studio***, which combines signal processing capabilities with an accessible GUI to support research into ultrasonic communication and cooperative behavior in laboratory animals.

The application was developed using *MATLAB App Designer*, an integrated environment for building interactive apps. This was chosen, because of the extensive use of MATLAB in research and academia. App Designer provides a visual design interface for laying out the GUI, as well as a built-in code editor for defining application logic. It streamlines the development process by automatically generating the underlying object-oriented structure of the app, making it easier to maintain, extend, and scale.

The user interface was constructed in the *Design View* using components such as buttons, drop-down menus, tab panels, and plotting axes. These components were organized into logical sections corresponding to the app's functionality - data input, visualization, detection, editing, and export. The behavior of the application, including user interaction and signal processing routines, was programmed in the *Code View* using MATLAB's object-oriented programming model. This separation of design and logic allowed for rapid prototyping while ensuring modularity and readability in the final implementation.

The result is a fully functional and modular tool supporting tasks such as file loading, batch processing, segmentation control, label editing, sonification, and synchronized video playback.

Below is a walkthrough of the main panels in the current Squeak Peek Studio application, presented in the order they appear in the user interface.

7.2 Application GUI

7.2.1 Data Input

The **Data Input** panel (Figs. 7.1 and 7.2) is where users load the essential input files: the raw USV audio files, manually annotated reference labels, and optionally, pre-computed detection labels. The application supports two modes of operation: *single-file mode* and *batch mode*.

In single-file mode, users work with a single audio recording and its corresponding labels. This mode is typically used for visualization, manual inspection, and editing of USVs. Batch mode, on the other hand, allows users to select multiple files and process them automatically. In this mode, the app displays the number of files loaded for each category and provides dropdown menus for selecting the active file for further interaction.

After selecting the paths, pressing the "Load Files" button imports the data into the app and prepares it for subsequent operations such as segmentation, sonification, or visualization.

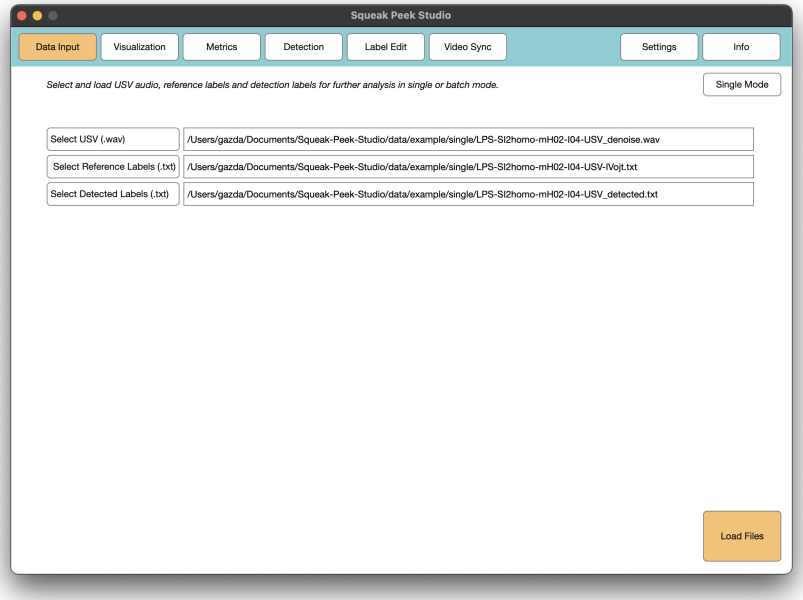


Figure 7.1: Data Input Panel – Single Mode

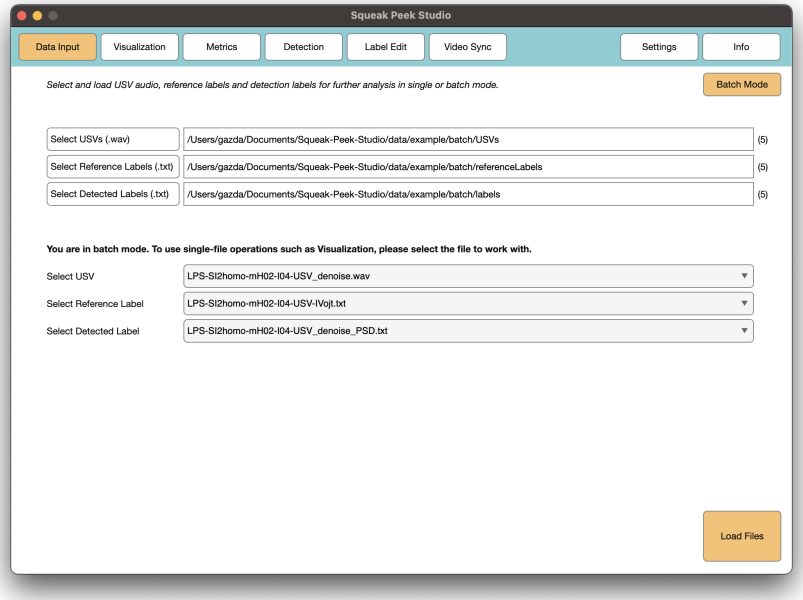


Figure 7.2: Data Input Panel – Batch Mode

7.2.2 Visualization

The **Visualization** panel (Fig. 7.3) displays the currently selected segment of the USV recording in both time-domain waveform and frequency-domain spectrogram. Users can specify the segment start time and length, navigate forwards/backwards through the recording, and click "Sonify" to play a human-audible version of the ultrasonic segment.

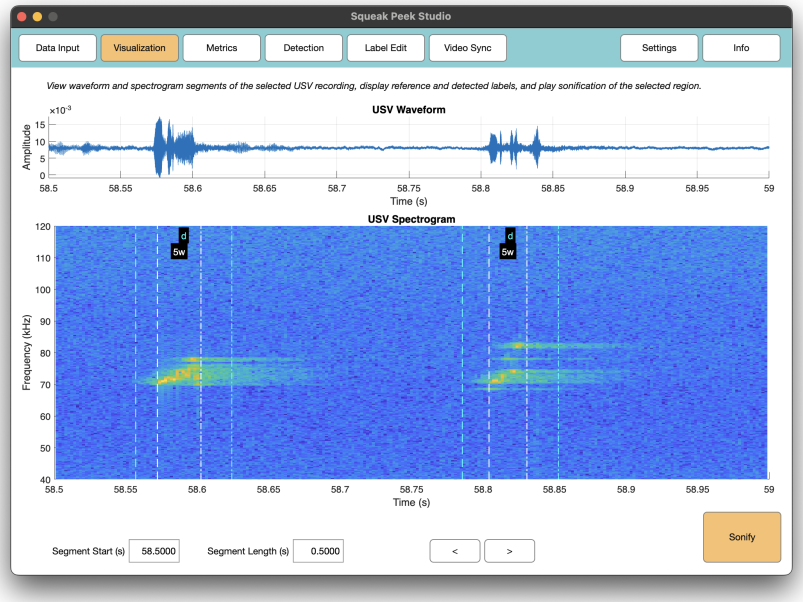


Figure 7.3: Visualization Panel

7.2.3 Metrics

In the **Metrics** panel (Fig. 7.4), the app computes standard detection performance measures by comparing detected labels against reference labels. It reports the total counts of detected and reference calls, true positives and false negatives, and derived precision, recall, and F1 score.

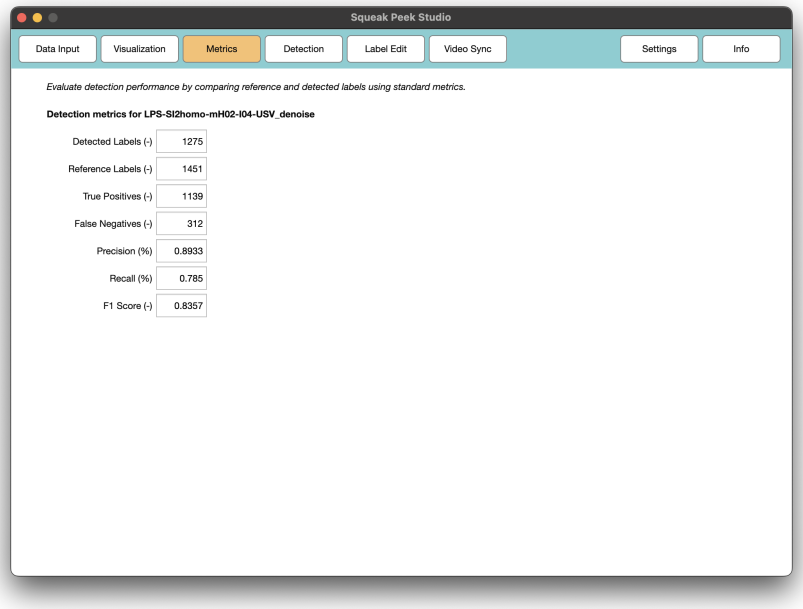


Figure 7.4: Metrics Panel

7.2.4 Detection

The **Detection** panel (Fig. 7.5) allows users to choose one or more detection algorithms (e.g. PSD, BACD, RBD) and optional post-processing steps (merging close labels, removing short labels). After selecting an export folder, clicking "Run Detectors" will batch-process all loaded audio files and save newly generated label files.

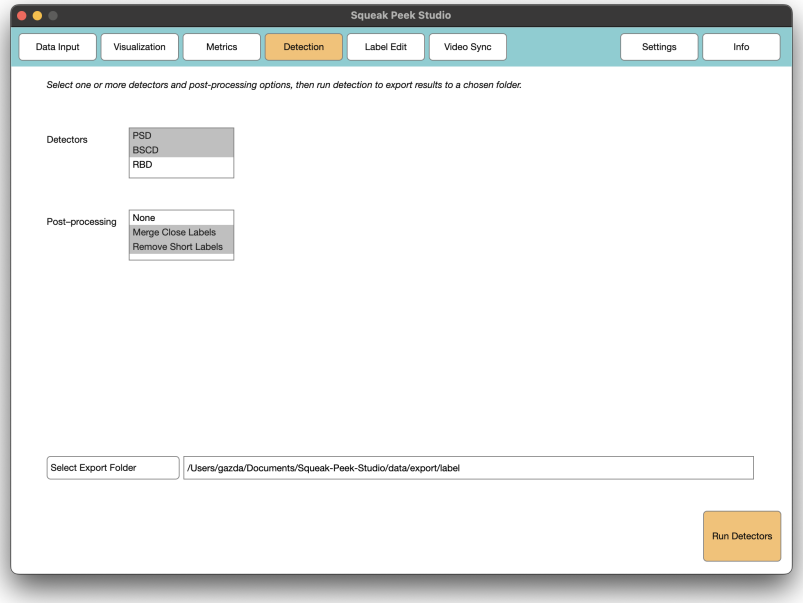


Figure 7.5: Detection Panel

7.2.5 Label Editing

The **Label Edit** panel (Fig. 7.6) provides manual curation tools. Users can drag the call-boundary handles on the spectrogram to fine-tune start/end times, accept or reject each detected call and its classification label, and step through calls one by one. Counters for accepted/rejected detections and classifications are shown, and "Export" saves the edited labels.

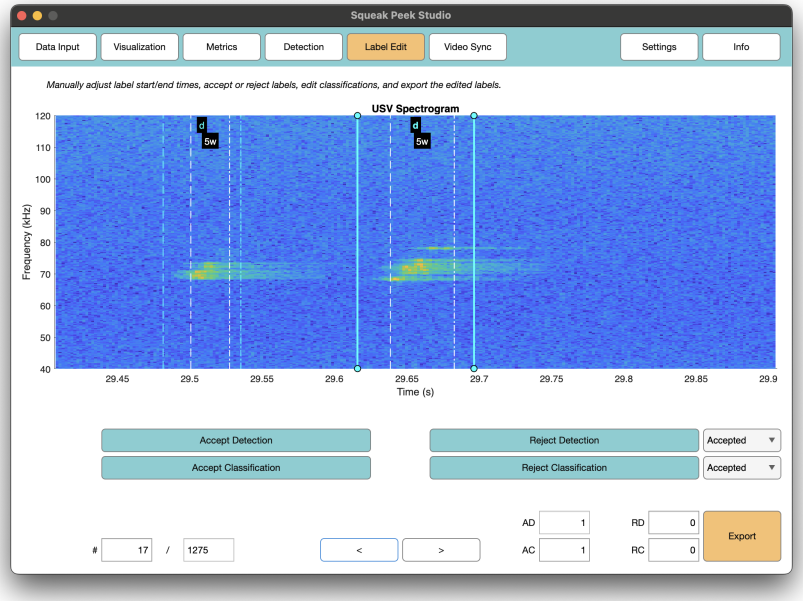


Figure 7.6: Label Edit Panel

7.2.6 Video Synchronization

The **Video Sync** panel (Fig. 7.7) synchronizes an external video recording with the USV audio. Users select the video file, enter the synchronization timestamp (when the first USV call occurred), choose whether to overlay the original or sonified audio, and click "Synchronize" to produce a trimmed, audio-replaced video.

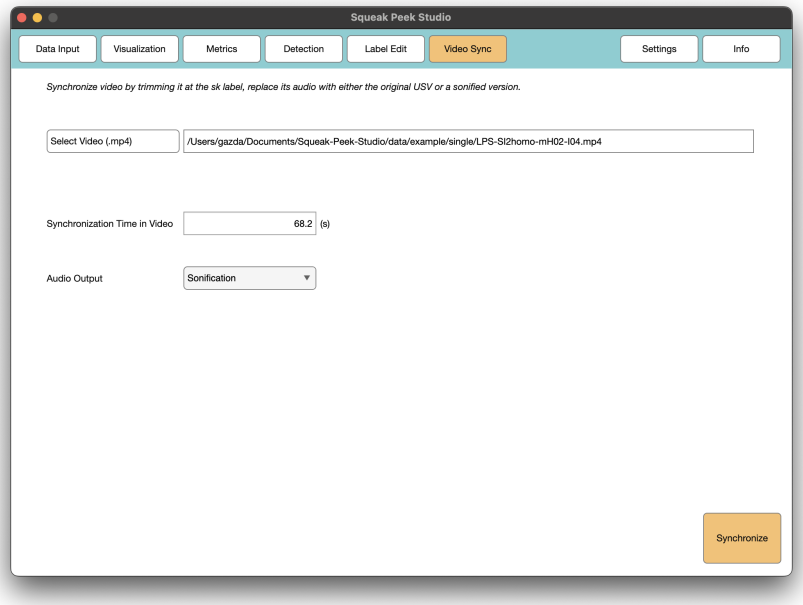


Figure 7.7: Video Sync Panel

7.2.7 Settings

The **Settings** panel provides centralized access to all configurable parameters within the application. It is context-sensitive. When accessed from a specific module (e.g., Data Input), it automatically opens the corresponding tab, streamlining navigation and improving usability.

All settings can be exported and imported as structured `.json` files, enabling reproducibility and easy sharing across users or sessions. These files can also be edited outside of MATLAB. On startup, the application automatically looks for a `default.json` file; if present, it loads this configuration, restoring the app to its exact previous state. This ensures that users return to the same working environment, facilitating continuity and personalized workflows.

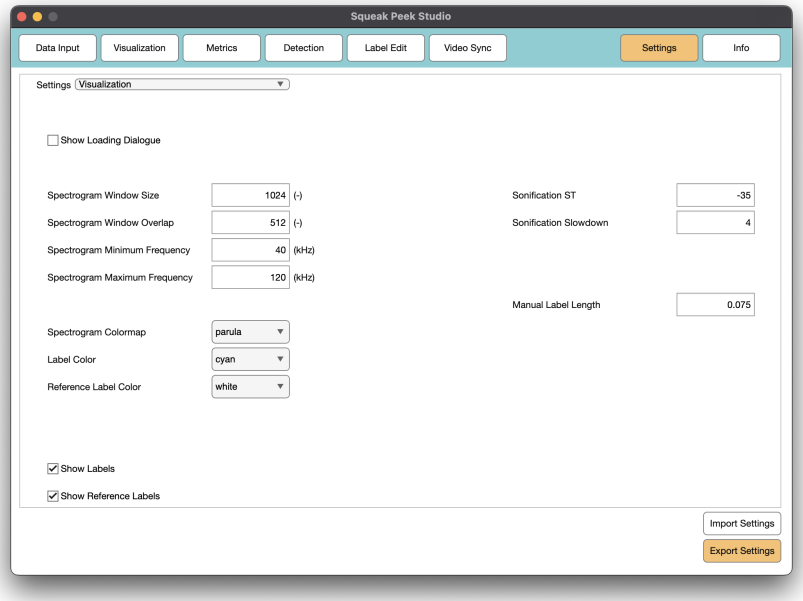


Figure 7.8: Settings Panel

7.2.8 Info

Finally, the **Info** panel (Fig. 7.9) displays application metadata: title, description, author and supervisor details, contact, academic year, MATLAB and app version numbers, and quick-link buttons to documentation, source code, and the full Master's thesis. Institutional logos are shown at the bottom.

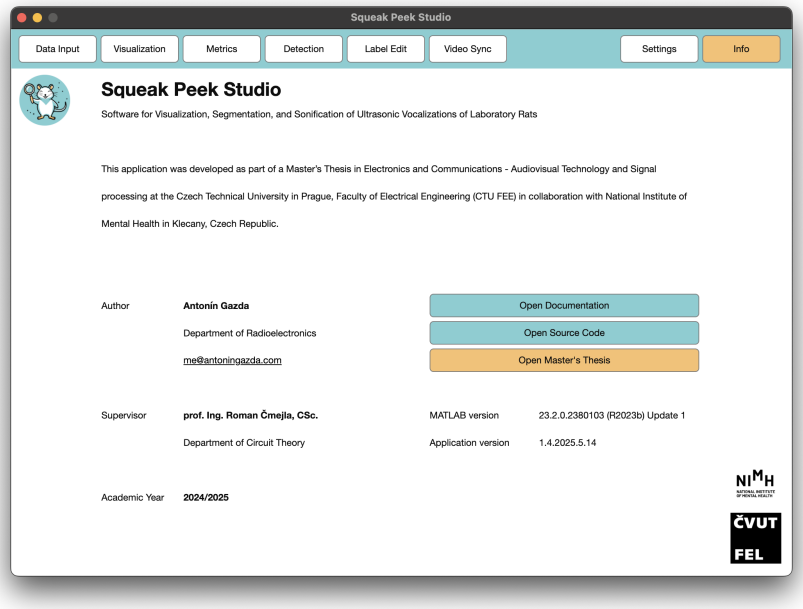


Figure 7.9: Info panel

7.3 Packaging

The ***Squeak Peek Studio*** application was packaged as a *MATLAB App*, allowing for straightforward distribution and installation within the environment. The application is provided as an `.mlappinstall` file, which can be installed by simply double-clicking it. Once installed, the app becomes accessible from the Apps gallery with a single click.

This packaging format is ideal for researchers already working within the ecosystem, as it ensures seamless integration, version control, and portability across different systems running MATLAB.

In addition to MATLAB App packaging, the application can also be compiled into a standalone desktop application using the *MATLAB Application Compiler*. This generates a platform-specific installer, which includes the MATLAB Runtime, enabling users to run the app on systems that do not have MATLAB installed.

Furthermore, with minimal modification, the application can be compiled and deployed as a web app using the *MATLAB Web App Server*. This allows the application to be accessed remotely through a web browser, providing a convenient interface for users who prefer not to install any software locally.

All packaging options are supported by MATLAB's built-in tools, making the app flexible to deploy and distribute depending on the user's environment and needs.

7.4 Installation and Updating

Squeak Peek Studio is distributed as a packaged MATLAB App Installer file (`.mlappinstall`) and can be easily installed or updated through the MATLAB interface.

Installation via `.mlappinstall`

To install the application:

1. Double-click the provided `Squeak Peek Studio.mlappinstall` file.
2. MATLAB will open a prompt asking whether you want to install the app (Figure 7.10).
3. Click **Install**. Once installed, the app appears under the **Apps** tab, ready to launch (Figure 7.11).

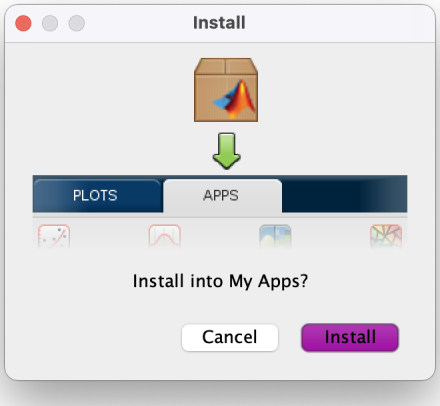


Figure 7.10: Installation prompt for the Squeak Peek Studio app.

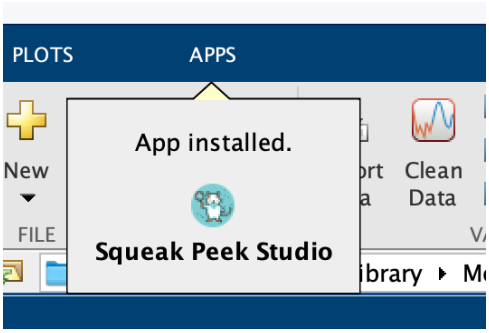


Figure 7.11: Squeak Peek Studio appears under the Apps tab after successful installation.

Updating the App

If an older version is already installed, MATLAB will prompt the user to confirm the update when a newer version is opened via `.mlappinstall` (Figure 7.12). Clicking **Update** replaces the older version while preserving its placement in the Apps tab.

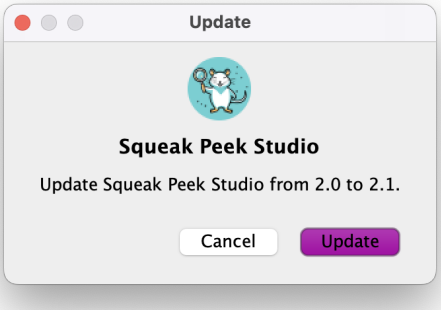


Figure 7.12: Update dialog for an existing installation of Squeak Peek Studio.

Development Mode via `runApp.m`

For users who wish to run the app in development mode (e.g., to access source code, debug features, or inspect runtime variables), the file `runApp.m` can be used. This script launches the application with all required paths initialized and retains access to the live session environment.

This method is recommended for developers or users interested in customizing detectors, modifying GUI elements, or extending functionality. Both `runApp.m` and the app package are listed in Appendix B.

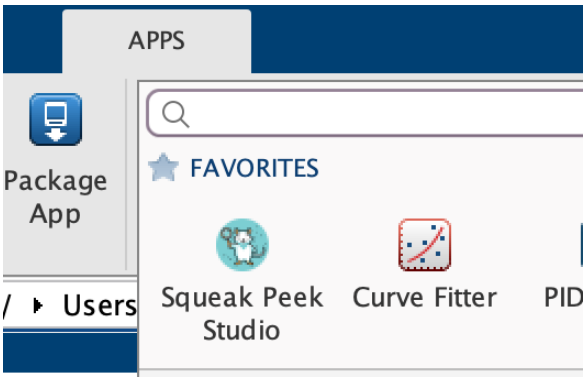


Figure 7.13: Squeak Peek Studio listed among other MATLAB Apps after installation.

Chapter 8

Conclusion

This thesis has presented the development of *Squeak Peek Studio*, a comprehensive application for the visualization, segmentation, sonification, and synchronization of ultrasonic vocalizations (USVs) in laboratory rats. Built using MATLAB App Designer, the software integrates the entire USV analysis pipeline into a single, interactive environment that combines automated processing with intuitive tools for manual refinement.

The application enables researchers to efficiently load single or batch recordings, inspect waveforms and spectrograms, and interactively explore and process vocalizations. It supports the import of reference labels and existing detection outputs, and implements several detection algorithms - power spectral density thresholding and Bayesian autoregressive changepoint detection with configurable post-processing options. A built-in metrics panel provides immediate feedback by comparing detected segments to reference annotations using standard metrics such as precision, recall, and F1 score.

Squeak Peek Studio includes a dedicated sonification pipeline that uses pitch-shifting and time-stretching via a custom phase vocoder to render ultrasonic signals audible. This improves validation and classification for researchers. Additionally, the system supports video synchronization, allowing audiovisual recordings to be exported with the original ultrasonic signal or a sonified version - enabling alignment of acoustic behavior with visual context.

A manual label editing interface allows users to refine detection results by adjusting segment boundaries, reassigning label types, and accepting or rejecting detections. The interface is designed for rapid, responsive interaction and supports keyboard shortcuts for annotation efficiency. Exported files preserve user annotations and maintain compatibility with external tools for downstream analysis.

In conclusion, *Squeak Peek Studio* provides a robust and extensible platform for USV analysis, already integrated into the research workflows of collaborators at the National Institute of Mental Health (NIMH). By combining segmentation, visualization, sonification, and evaluation into a single cohesive system, it supports both objective quantification and intuitive interpretation of ultrasonic vocal behavior. Its modular architecture enables users to adapt the software to diverse experimental designs and recording setups, while maintaining ease of use through an accessible graphical interface.

While the application meets a wide range of practical needs, several areas remain open for future improvement. The software is structurally prepared for integration of *automatic classification modules*, which could enable supervised or unsupervised labeling of USV types. Similarly, the current detectors, based on thresholding and statistical changepoint modeling, can be extended with *machine learning* approaches to improve generalization and adaptability across varied datasets. Although the application has been tested extensively at CTU and in collaboration with NIMH, additional evaluation across different environments, operating systems, and hardware configurations is necessary to ensure *full robustness* and portability. Future development should also focus on expanding post-processing capabilities, such as discarding detections with significant energy in the audible range, improving the flexibility of the graphical interface, and integrating advanced denoising techniques - ensuring the system continues to evolve toward more accurate, user-adaptable, and robust USV analysis.

Bibliography

[1] PREMOLI, M., PIETROPAOLO, S., WÖHR, M., SIMOLA, N., BONINI, S. A. Mouse and rat ultrasonic vocalizations in neuroscience and neuropharmacology: State of the art and future applications. *European Journal of Neuroscience*, 2023, 57(12), 2062–2096.

[2] PESSOA, D., PETRELLA, L., MARTINS, P., CASTELO-BRANCO, M., TEIXEIRA, C. Automatic segmentation and classification of mice ultrasonic vocalizations. *The Journal of the Acoustical Society of America*, 2022, 152(1), 266–280.

[3] HERDT, R., KINZEL, L., MAAß, J.G., WALTHER, M., FRÖHLICH, H., SCHUBERT, T., MAASS, P., SCHAAF, C.P. Enhancing the analysis of murine neonatal ultrasonic vocalizations: Development, evaluation, and application of different mathematical models. *arXiv preprint arXiv:2405.12957*, 2024. Available from: `https://arxiv.org/abs/2405.12957` [Accessed 10 May 2025].

[4] PREMOLI, M., SABANOVIC, M., CHABOUT, J. Neural and behavioral correlates of ultrasonic vocalizations in rats: A comprehensive review. *Neuroscience & Biobehavioral Reviews*, 2023, 144, 105048.

[5] ASHLEY, N. T., et al. A web-based automated ultrasonic vocalization scoring platform. *Frontiers in Behavioral Neuroscience*, 2021, 15, 702648.

[6] COFFEY, K. R., MARX, R. G., NEUMAIER, J. F. DeepSqueak: A deep learning-based system for detection and analysis of ultrasonic vocalizations. *Neuropsychopharmacology*, 2019, 44(5), 859–868.

[7] VAN SEGBROECK, M., KNOLL, A. T., LEVITT, P., NARAYANAN, S. MUPET—Mouse Ultrasonic Profile ExTraction: A Signal Processing Tool for Rapid and Unsupervised Analysis of Ultrasonic Vocalizations. *Neuron*, 2017, 94(3), 465–485.e5.

[8] NETSER, S., NAHARDIYA, G., WEISS-DICKER, G., DADUSH, R., GOUSSHA, Y., JOHN, S. R., TAUB, M., WERBER, Y., SAPIR, N., YOVEL, Y., HARONY-NICOLAS, H., BUXBAUM, J. D., COHEN, L., CRAMMER, K., WAGNER, S. TrackUSF, a novel tool for automated

ultrasonic vocalization analysis, reveals modified calls in a rat model of autism. *BMC Biology*, 2022, 20(1), 159.

[9] NOLDUS INFORMATION TECHNOLOGY. UltraVox XT: Ultrasonic vocalization detection made easy. *Noldus Information Technology*. Online. Available from: `https://noldus.com/ultravox-xt` [Accessed 10 May 2025].

[10] TACHIBANA, R. O., KANNO, K., OKABE, S., KOBAYASI, K. I., OKANOYA, K. USVSEG: A robust method for segmentation of ultrasonic vocalizations in rodents. *PLOS ONE*, 2020, 15(2), e0228907.

[11] SEGALIN, C., WILLIAMS, J., KARIGO, T., HUI, M., ZELIKOWSKY, M., SUN, J. J., PERONA, P., ANDERSON, D. J. Analysis of ultrasonic vocalizations from mice using computer vision and machine learning. *eLife*, 2020, 9, e59161.

[12] ANIS, S. S., KELLIS, D. M., KAIGLER, K. F., WILSON, M. A., O'REILLY, C. A reliable and efficient detection pipeline for rodent ultrasonic vocalizations. *arXiv preprint arXiv:2503.18928*, 2025.

[13] BARKER, D. J., et al. Automated detection of 50-kHz ultrasonic vocalizations using template matching in XBAT. *Journal of Neuroscience Methods*, 2014, 236, 68–75.

[14] HERMANN, T., HUNT, A., NEUHOFF, J. G. (eds.). *The Sonification Handbook*. Berlin: Logos Verlag, 2011. Available from: `https://sonification.de/handbook/` [Accessed 10 May 2025].

[15] SETHARNES, W. *A Phase Vocoder in Matlab*. Online. Available from: `https://sethares.engr.wisc.edu/htmlRT/vocoders/phasevocoder.html` [Accessed 10 May 2025].

[16] CMEJLA, R., RUSZ, J., BERGL, P., VOKRAL, J. Bayesian changepoint detection for the automatic assessment of fluency and articulatory disorders. *Speech Communication*, 2013, 55, 178–189.

[17] DODOTRONIC. *Ultramic UM250K | Dodotronic*. Online. Available from: `https://www.dodotronic.com/product/ultramic-um250k/` [Accessed 10 May 2025].

[18] AUDACITY. *Importing and Exporting Labels - Audacity Manual*. Online. Available from: `https://manual.audacityteam.org/man/importing_and_exporting_labels.html` [Accessed 10 May 2025].

[19] MATLAB. *MATLAB App Designer – MATLAB & Simulink*. Online. Available from: `https://www.mathworks.com/products/matlab/app-designer.html` [Accessed 10 May 2025].

Appendices

Appendix A

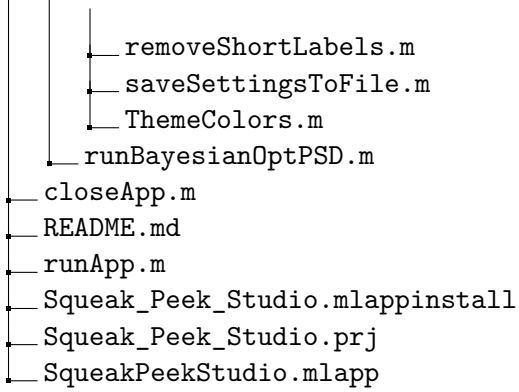
List of Abbreviations

Abbreviation	Meaning
AR	Autoregression
BACD	Bayesian Autoregressive Changepoint Detector
CNN	Convolutional Neural Network
CTU	Czech Technical University in Prague
FEE	Faculty of Electrical Engineering
FFT	Fast Fourier Transform
FP	False Positive
GUI	Graphical User Interface
IFFT	Inverse Fast Fourier Transform
MFCC	Mel-frequency cepstral coefficient
NIMH	National Institute of Mental Health
PC	Personal Computer
PSD	Power Spectral Density
RAM	Random Access Memory
RBD	Recursive Bayesian Autoregressive Changepoint Detector
ROI	Region of Interest
SNR	Signal-to-Noise Ratio
ST	Semitone
STFT	Short-Time Fourier Transform
TN	True Negative
TP	True Positive
USB	Universal Serial Bus
USV	Ultrasonic Vocalization

Appendix B

Included Files

```
Squeak-Peek-Studio/
├── assets/
│   └── logo/
│       ├── fee_logo.jpg
│       ├── nimh_logo.png
│       └── squeakpeak_logo.png
├── data/
│   └── example/
│       ├── batch/
│       │   ├── labels/
│       │   ├── referenceLabels/
│       │   └── USVs/
│       ├── single/
│       │   ├── Detected_Labels_Example.txt
│       │   ├── Reference_Labels_Example.txt
│       │   └── USV_Example_Short.wav
│       └── export/
├── settings/
├── Squeak_Peek_Studio_resources/
│   ├── icon_16.png
│   ├── icon_24.png
│   └── icon_48.png
└── src/
    └── functions/
        ├── bscd.m
        ├── BSCDDetector.m
        ├── compareLabels.m
        ├── exportLabels.m
        ├── exportLabelsDetector.m
        ├── importLabels.m
        ├── mergeCloseLabels.m
        ├── PSDDetector.m
        ├── RBD.m
        └── RBDDetector.m
```


removeShortLabels.m
saveSettingsToFile.m
ThemeColors.m
runBayesianOptPSD.m
closeApp.m
README.md
runApp.m
Squeak_Peek_Studio.mlappinstall
Squeak_Peek_Studio.prj
SqueakPeekStudio.mlapp