**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ГОРОДА МОСКВЫ  
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ**

**«ШКОЛА** №**709 ИМЕНИ ДВАЖДЫ ГЕРОЯ СОЦИАЛИСТИЧЕСКОГО ТРУДА**

**В.И. ДОЛГИХ»**

**МЕТОД ОБНАРУЖЕНИЯ БПЛА КВАДРОКОПТЕРНОГО ТИПА ПО ЗВУКОВОМУ СИГНАЛУ**

Участники:

ученики 11 «И» класса:

Королёва Валерия Александровна

Трунтаев Роман Витальевич

Исаев Роман Русланович

Руководитель:

Педагог ГБОУ школа №709

Петрова Марина Владимировна

Оглавление

[1. Введение 2](#_Toc190876414)

[2. Выбор программ 4](#_Toc190876415)

[3. Алгоритм записи звукового сигнала и построения его спектральных характеристик 7](#_Toc190876416)

[3.1. Блок схема алгоритма для записи звукового сигнала 7](#_Toc190876417)

[3.2. Визуализация потока данных с микрофона 9](#_Toc190876418)

[3.3. Запись звукового сигнала 9](#_Toc190876419)

[3.4. Построение спектральных характеристик 10](#_Toc190876420)

[3.4.1. Анализ и проверка работы системы 11](#_Toc190876421)

[4. Изучение спектральных характеристик квадрокоптера 14](#_Toc190876422)

[4.1. Метод выделения спектрального следа БПЛА в звуковом потоке 15](#_Toc190876423)

[4.2. Обновление индикатора 17](#_Toc190876424)

[5. Результаты работы и их проверка 19](#_Toc190876425)

[5.1. Эксперимент №1. В классе без шума 19](#_Toc190876426)

[5.2. Эксперимент №2. В классе со среднем шумом 20](#_Toc190876427)

[5.3. Эксперимент №3. На улице при высоком шуме и увеличенном расстоянии. 21](#_Toc190876428)

[6. Выводы 23](#_Toc190876429)

[7. Список литературы 24](#_Toc190876430)

# Введение

Развитие беспилотных летательных аппаратов (БПЛА), в том числе квадрокоптеров, открывает новые возможности в различных сферах, например: сельское хозяйство, логистика, доставка, но также несет в себе потенциальные риски. Рост популярности дронов, делающий их доступными для широкого круга пользователей, повышает риск их неправомерного использования, включая угрозы безопасности частной жизни, а также национальные конфликты и представляет собой глобальный вызов. Разработка эффективных методов обнаружения и идентификации квадрокоптеров является актуальной задачей.

Традиционные методы обнаружения БПЛА такие как радиолокационные системы и системы видеонаблюдения обладают определёнными ограничениями. Их эффективность существенно снижается в плохих погодных условиях (туман, дождь, ночь).

Актуальной задачей является разработка альтернативных и более эффективных методов обнаружения БПЛА. Звуковой сигнал квадрокоптера предлагается использовать за основу системы обнаружения. Данный подход позволяет создать компактные и мобильные системы, не требующие сложной и дорогостоящей инфраструктуры.

В данной работе предлагается разработка системы обнаружения квадрокоптеров по звуковому сигналу, как альтернативного метода в условиях ограниченной видимости.

**Цель:**

Разработка и создание эффективной системы обнаружения квадрокоптеров на основе анализа звукового сигнала

**Задачи:**

1. разработка алгоритмов для записи звукового сигнала, построения его спектральных характеристик:
2. изучить спектральные характеристик квадрокоптера
3. разработка алгоритма для выявления квадрокоптера по звуковому сигналу
4. проведение тестирования
5. анализ и оценка эффективности системы обнаружения

# Выбор программ

В процессе реализации проекта в качестве основной платформы разработки был выбран язык программирования Python. Выбор обусловлен рядом его преимуществ, критически важных для успешного выполнения проекта. Python, как современный язык программирования высокого уровня, характеризуется высокой степенью читаемости и структурированности кода, что существенно упрощает работу, поддержание и дальнейшее расширение системы. Его универсальность позволяет эффективно решать задачи различной сложности, от обработки аудиоданных до построения графического интерфейса пользователя. Наличие широкого спектра специализированных библиотек, предназначенных для научно-технических расчетов и обработки сигналов, значительно сократило время разработки и позволило сосредоточиться на решении ключевых задач проекта. Открытый исходный код библиотек обеспечивает прозрачность и дополнительные возможности для модификации и адаптации под специфические требования проекта.

Несмотря на несомненные преимущества, при использовании Python были учтены и его ограничения. Невысокая скорость выполнения кода по сравнению требовала оптимизации алгоритмов обработки звукового сигнала для обеспечения реального времени.

Для ускорения процесса разработки и обеспечения высокого качества кода была использована интегрированная среда разработки Visual Studio Code. Возможности по проверке совместимости версий интерпретатора и быстрому рефакторингу кода значительно упростили процесс разработки и поддержания чистоты кода.

Начальный этап работы над проектом включал освоение базовых конструкций языка Python и отработку алгоритмов графической визуализации данных.

Для реализации системы были использованы следующие библиотеки Python:

• NumPy: для работы с многомерными массивами, математических вычислений (включая FFT), и обработки цифровых сигналов. Яляется основой для многих научных вычислений в Python [1].

• PyAudio: библиотека для работы со звуковым вводом/выводом. Предоставляет интерфейс для доступа к аудиоустройствам компьютера, позволяя, записывать и воспроизводить звук [2].

• Matplotlib: библиотека для создания графиков и визуализации данных. Используется для построения спектрограммы в реальном времени [3].

• SciPy: научная библиотека, содержащая множество модулей для научных и инженерных расчетов. Используется модуль fftpack из SciPy для быстрого преобразования Фурье (FFT) [4].

• Tkinter: библиотека для создания графических интерфейсов (GUI). Используется для создания основного окна приложения и элементов управления (ползунки) [5].

• Time: библиотека Python для работы со временем, используется для временных задержек и измерений [6].

Установка библиотек выполняется с помощью менеджера пакетов pip

Для достижения поставленной цели необходимо разработать программу для записи звуковых сигналов с микрофона и построения спектра сигнала. Полученную программу необходимо протестировать, затем записать спектр звука, издаваемого БПЛА и разработать алгоритм идентификации БПЛА квадрокоптерного типа на фоне акустических шумов.

Использовано следующее оборудование: микрофон Fifane A8 и ноутбук Dell Precision 3520

Место и сроки: Работа выполнена в школе «№709 имени дважды героя социального труда В.И.Долгих.». С сентября 2024 года по январь 2025

# Алгоритм записи звукового сигнала и построения его спектральных характеристик

## Блок схема алгоритма для записи звукового сигнала

В данном разделе описывается процесс разработки программного обеспечения для захвата звукового сигнала с микрофона, его обработки и визуализации спектральных характеристик в реальном времени. На рисунке 1 представлена блок-схема.

Как видно из рисунка, алгоритм можно разбить на несколько этапов:

1. Инициализация библиотек и параметров:
   1. импорт библиотек: numpy, pyaudio, matplotlib, scipy, tkinter.
   2. начальные параметры аудиопотока: CHUNK (размер буфера) и RATE (частота дискретизации).
   3. создается объект pyaudio.PyAudio() и открывается аудиопоток
2. Запись аудио:
   1. считывание данных из аудиопотока с помощью библиотеки pyaudio и инициализации объекта
3. Преобразование Фурье (FFT):
   1. берется модуль комплексного результата для получения амплитуд.
   2. производится логарифмирование амплитуд для более удобной визуализации.
4. Обновление спектрограммы:
   1. изображение спектрограммы обновляется с использованием img.set\_data(spec\_data).
   2. временная ось спектрограммы обновляется для отслеживания динамики изменений.
5. Обновление графика амплитуды:
   1. вычисляются значения частот с помощью np.linspace(0, RATE // 2, n\_fft //2).
   2. обновляются данные графика зависимости амплитуды от частоты с помощью line.set\_data(freqs, y\_data).
6. Перерисовка графиков

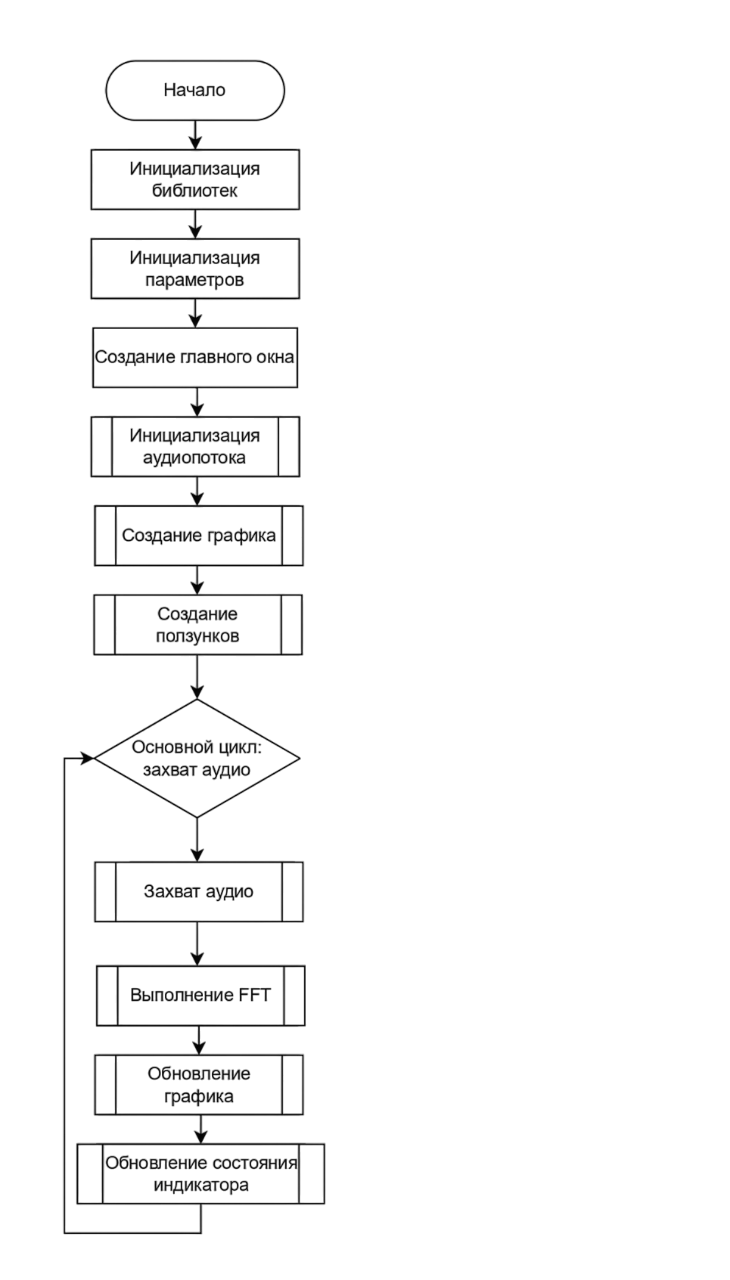


Рисунок 1. Блок-схема

## Визуализация потока данных с микрофона

Для наглядного представления звуковых данных в проекте используются два графика для визуализации системы

Спектрограмма. Визуализирует частоты звукового сигнала во времени. Ось Х – время, ось Y – частота, интенсивность цвета в каждой точке отображает амплитуду в данный момент времени на данной частоте. Позволяет отслеживать повторяющие паттерны, для помощи выявления квадрокоптера.

График зависимости амплитуды от частоты. Отображает мгновенный спектр звукового сигнала. Ось X – частота, ось Y – амплитуда. Позволяет определить доминирующие частоты в данный момент времени, а также предоставляет информацию о частотном составе сигнала

Оба графика обновляются в режиме реального времени, позволяющее оперативно реагировать на изменения в акустическом сигнале

Для построения графиков используется библиотека matplotlib. Особенностью предложенного решения является то, что пользователь наблюдает не статичные изображения, а изменения отслеживаемых параметров в режиме реального времени. Это достигается путем вызова данных функций

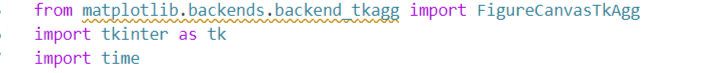


Рисунок 2. Библиотека для построения графиков

## Запись звукового сигнала

Код, представленный на рисунке 3, является частью программы, который анализирует звук в реальном времени.

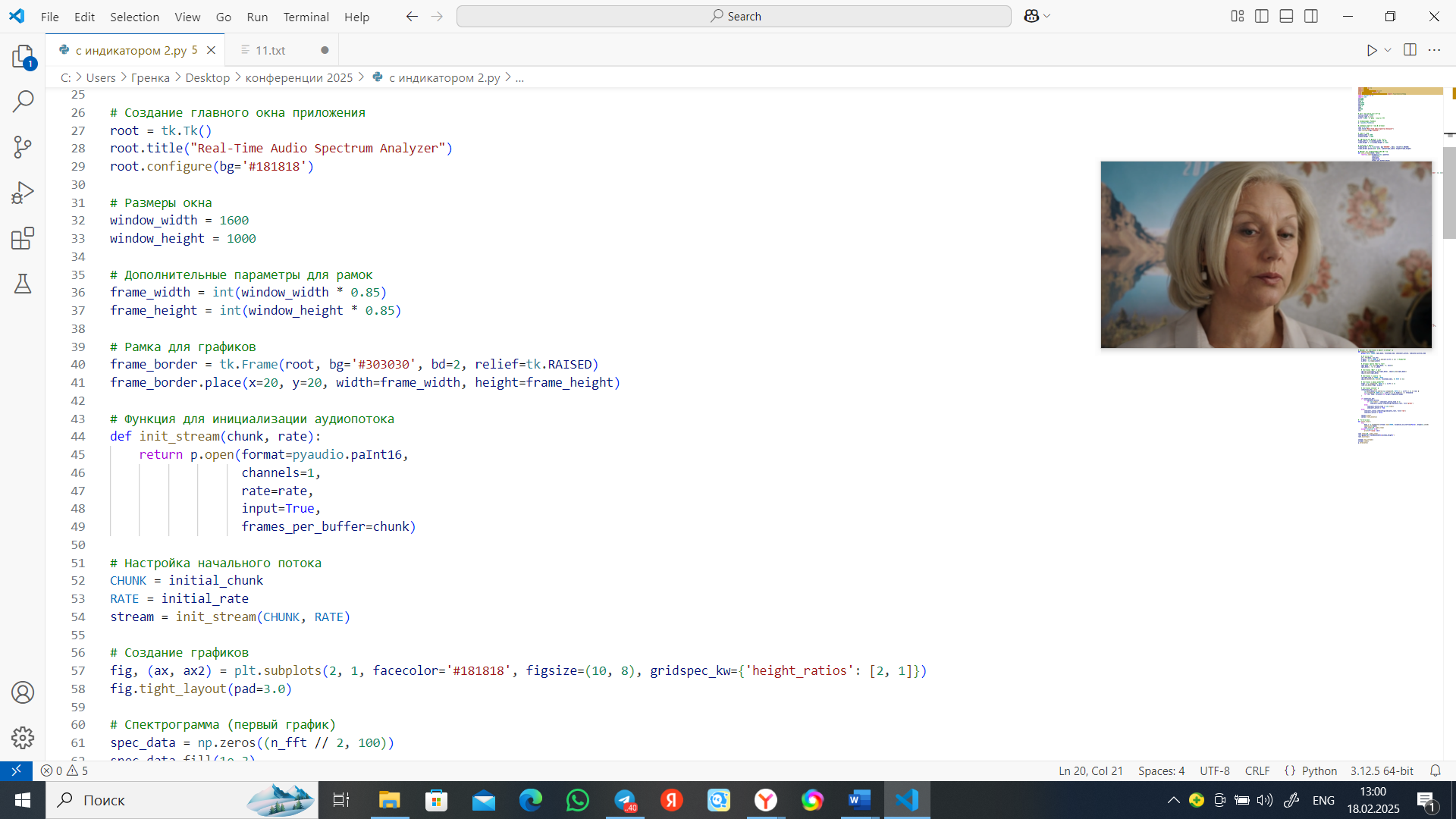


Рисунок 3. Инициализация аудиопотока

Для считывания аудиосигнала используется библиотека pyaudio. Происходит инициализация объекта «pyaudio. PyAudio». Он и предоставляет интерфейс для работы с аудиоустройствами. Далее открывается аудиопоток, в нём задаются следующие параметры:

1. format = pyaudio. paInt16: формат аудиоданных – 16-битное целое число.
2. channels = 1: моноканальная запись звука
3. rate: частота дискретизации звука
4. input = True: поток используется для записи аудио с микрофона
5. frames\_per\_buffer: размер буфера (количество отсчетов аудиосигнала, обрабатываемых за один раз).

Для улучшения работы, пользователем могут быть изменены размер буфера и частоты дискретизации в реальном времени с помощью ползунков в графическом интерфейсе.

## Построение спектральных характеристик

После получения массива данных, выполняется преобразование Фурье (FFT) для получения спектральных характеристик сигнала.

Чтобы визуализировать спектр во времени используется спектрограмма. Она показывает, как с течением времени меняется спектр сигнала. История спектральных данных хранится в переменной spec\_data и сдвигает ее влево при каждом новом фрейме.

Для этого используется функция fft из библиотеки scipy.fftpack, которая преобразует массив временных отсчетов в массив частотных отсчетов. Полученный массив делится на CHUNK для нормализации амплитуды. Так как fft дает симметричный спектр, то берется его половина. Работа программы представлена на рисунке 4

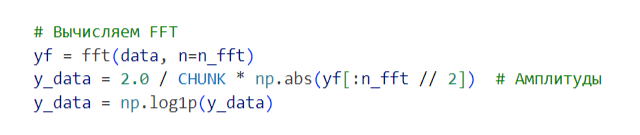


Рисунок 4. Вычисление FFT

Результатом является одномерный массив, где каждый элемент представляет амплитуду звукового сигнала на соответствующей частоте. Данные спектра хранятся и визуализируются как спектрограмма. Для этого создается двумерный массив spec\_data, в котором каждый столбец соответствует спектру сигнала, полученному в конкретный момент времени.

Для построения графика зависимости амплитуды от частоты используются полученные ранее значения амплитуд y\_data и соответствующие значения частот, вычисленные с помощью np.linspace(0, RATE // 2, n\_fft // 2). Эти данные затем используются для обновления графика line на графическом интерфейсе.

## Анализ и проверка работы системы

Для проверки работоспособности алгоритмов записи и обработки звукового сигнала был взят видеофайл, в котором частота звукового сигнала постепенно изменялась от 0 до 24000 Гц. Записанный аудиосигнал с видео анализировался в реальном времени с помощью разработанной программы. Результаты эксперимента представлены на скриншотах экрана системы (рисунки 5 и 6)

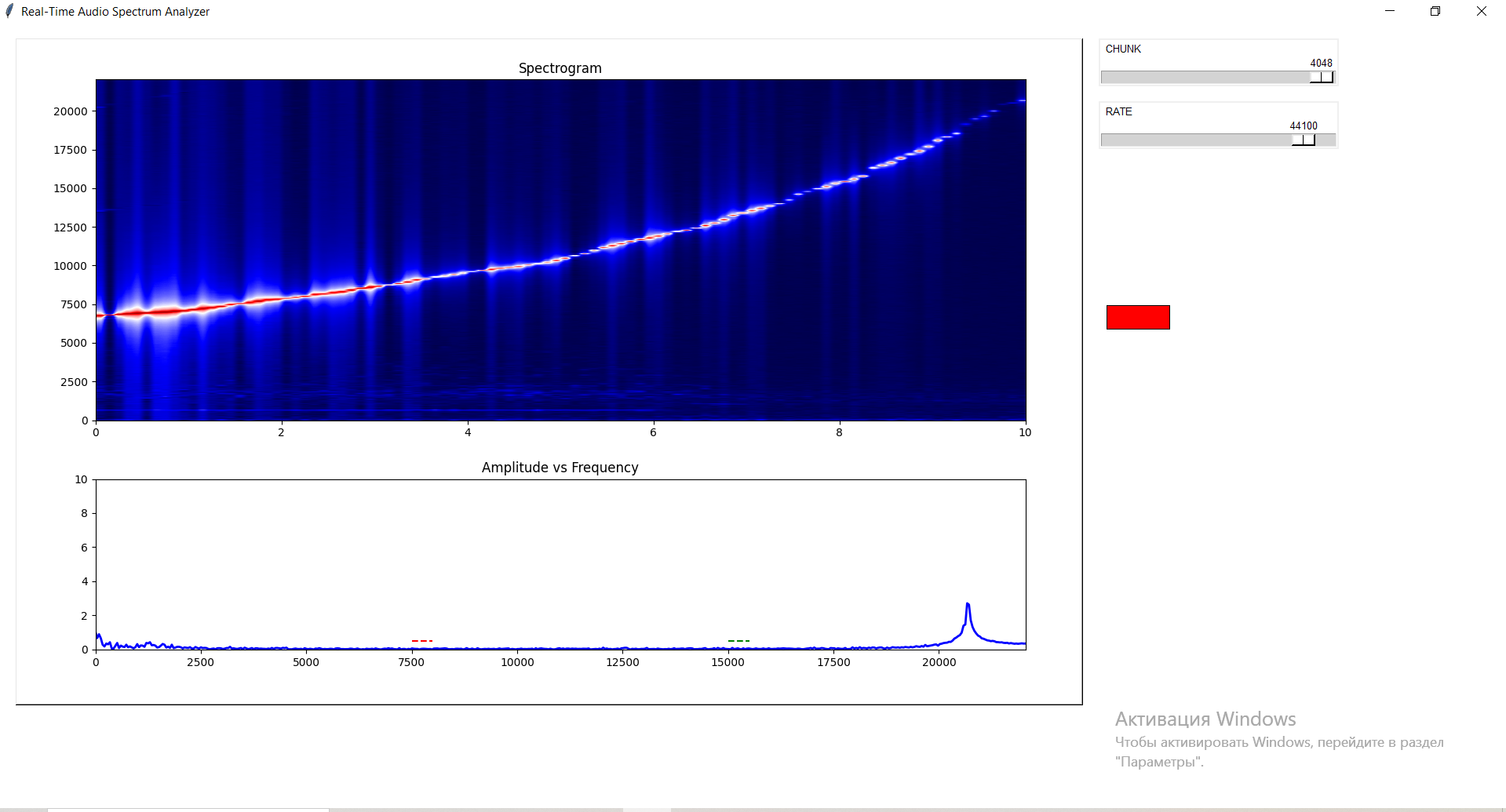


Рисунок 5. Повышение частоты от 7500 Гц до 24000 Гц

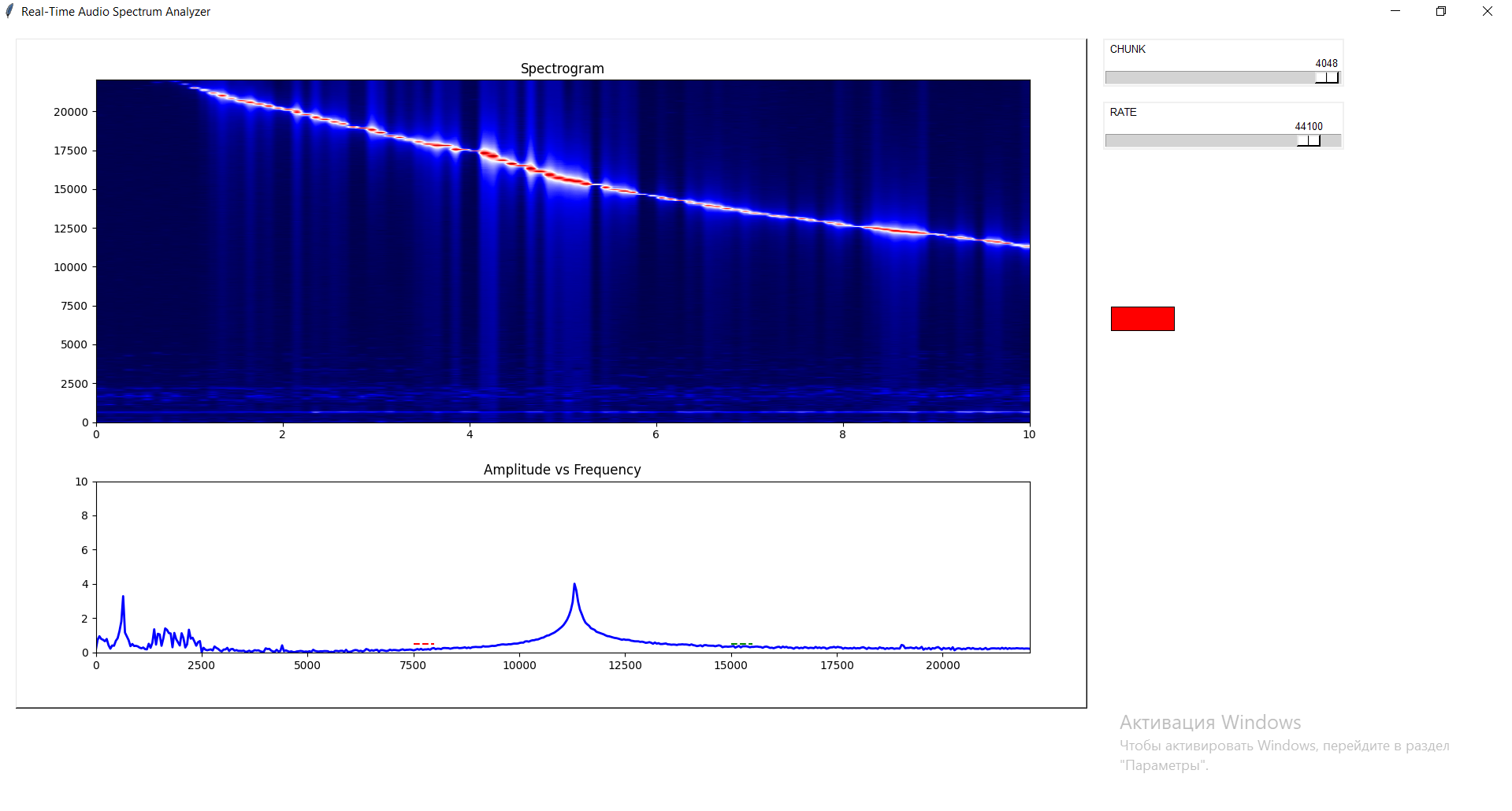


Рисунок 6. Понижение частоты от 24000 Гц до 11500 Гц

Для анализа на рисунке 5 частота плавно повышается и это отображается на графике, на рисунке 6 частота плавное понижается и это так же отображается на графике. Анализ полученных спектрограмм показал, что программа корректно отображает изменения частоты звукового сигнала. Это подтверждает работоспособность разработанного алгоритма.

Представленный алгоритм позволяет получать и анализировать звуковой сигнал в реальном времени и теперь может использоваться для дальнейшей работы с проектом.

# Изучение спектральных характеристик квадрокоптера

Для того, чтобы обнаружить квадрокоптер используется одна из важных спектральных характеристик – спектр звукового сигнала, выделяя ключевые частотные диапазоны: 7500-8000 Гц, 15000-15500 Гц и 0-500 Гц. Звук в данных диапазонах позволяет зафиксировать тона от вращающихся двигателей и винтов (высокие частоты) и другие вибрации (низкие частоты). Анализ сразу несколько диапазонов позволяет избежать ложных срабатываний и повысить надежность и точность системы. Важно, чтобы диапазоны работали одновременно. Это позволяет отфильтровать посторонние звуки, которые могут случайно попасть в один или два диапазона. Результаты исследования показаны на рисунке 7.

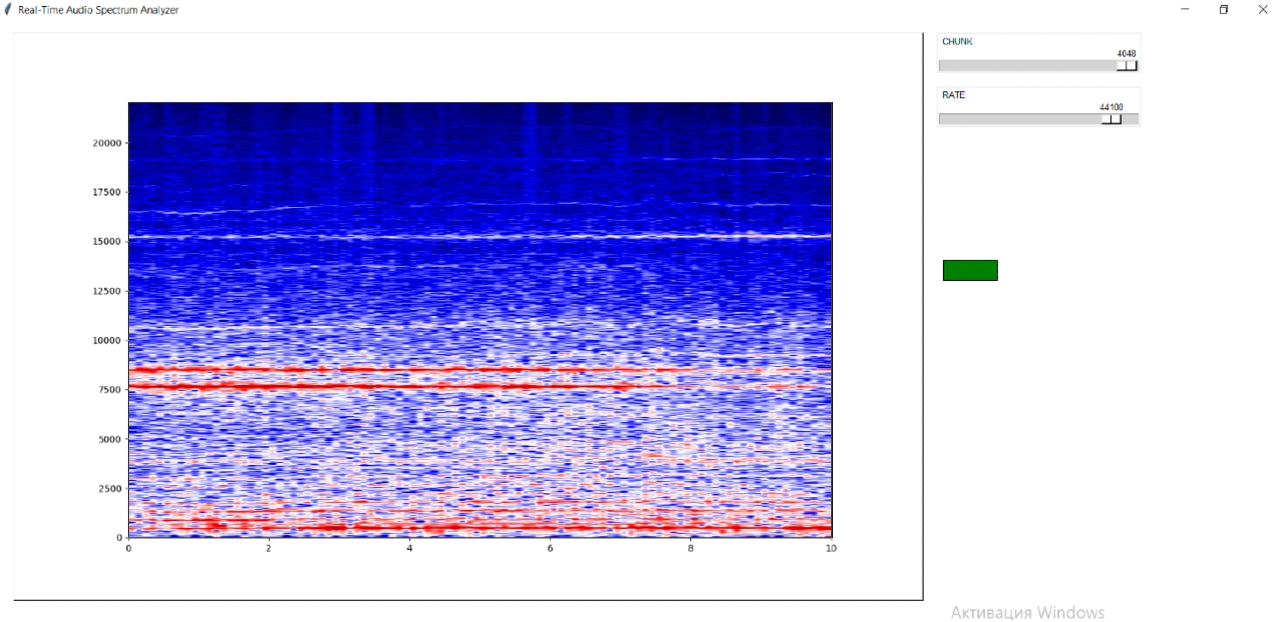


Рисунок 7. Пример нашего исследования

Проверяя работоспособность системы, были изучены некоторые зарубежные работы исследования в области развития методов акустической разведки[7]. Пример зарубежного исследования представлен на рисунке 8

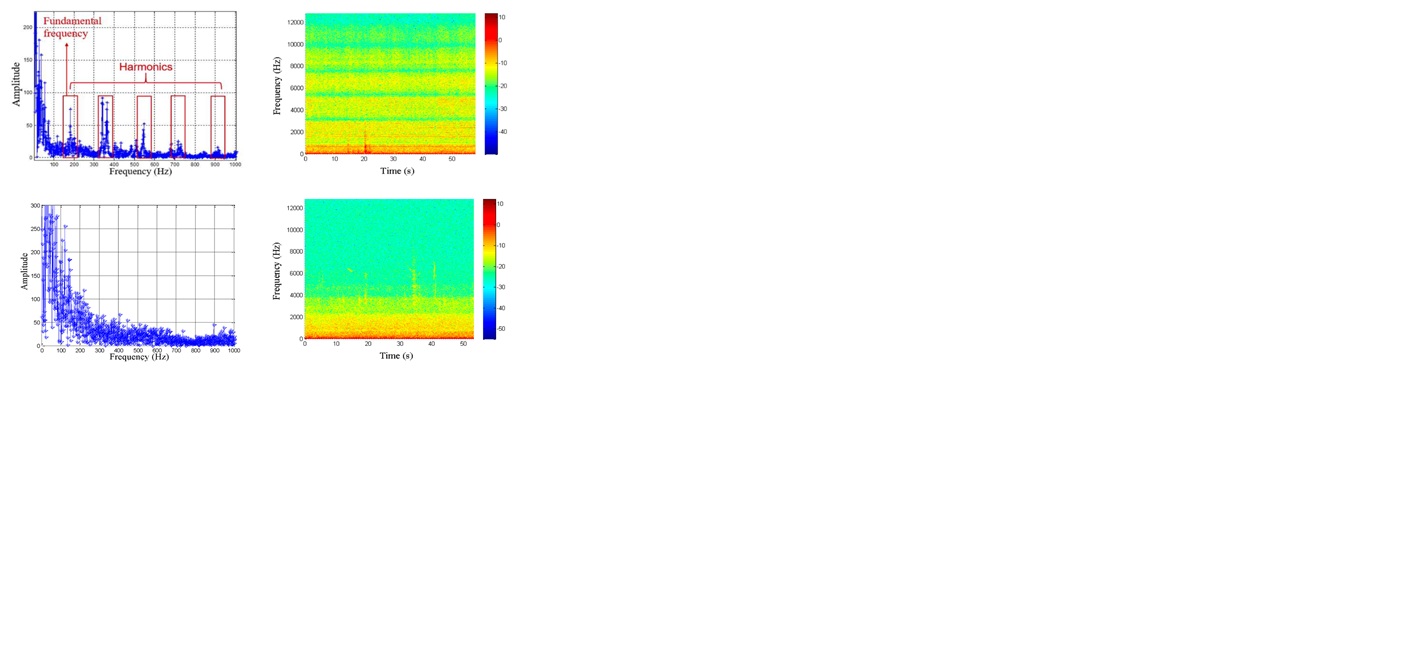


Рисунок 8. Пример зарубежного исследования

Было зафиксировано сходство результатов показателей частотных спектров с нашей программой. Это говорит о том, что система работает правильно и выявляет спектральные характеристики квадрокоптера.

## Метод выделения спектрального следа БПЛА в звуковом потоке

На рисунке 9 представлена часть кода, которая проверяет амплитуды в заданных диапазонах.

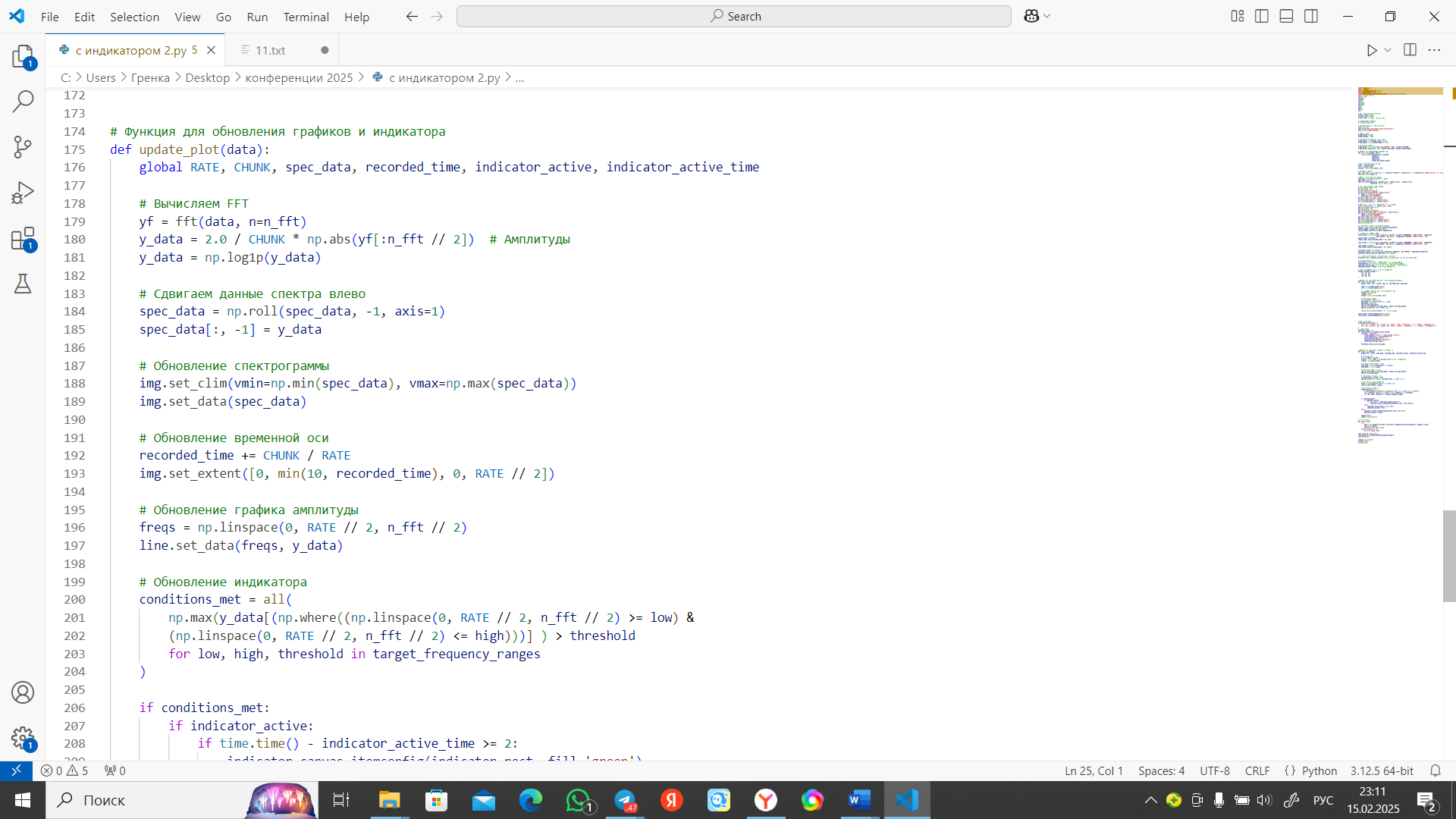


Рисунок 9. Проверка амплитуд

Переменные low, high и threshold. Они принимают крайние значения частотных диапазонов из target\_frequency\_ranges.

1. low. Переменная содержит нижнюю границу, то есть определяет минимальное значение частотного диапазона.
2. high. Переменная содержит верхнюю границу, то есть определяет максимальное значение частотного диапазона
3. threshold. Переменная определяет минимальное значение громкости в заданном частотном диапазоне, созданном low-high. Если амплитуда превышает значение threshold, то условие диапазона выполняется.

Чтобы обнаружить квадрокоптер проверяется наличие максимальных значений амплитуд, заданных переменными x1, y1, x2, y2 и x3, y3, диапазонах частот и пороговыми значениями амплитуд. Диапазон 1: x1=7500 Гц, y1=8000 Гц, z1=0,5. Диапазон 2: x2=15000Гц, y2=15500 Гц, z2=0,5. Диапазон 3: x3=0 Гц, y3=500 Гц, z3=1. Диапазоны хранятся в переменной target\_frequency\_ranges, и каждое значение представляет собой кортеж (нижняя граница, верхняя граница, порог) (рисунок 10).

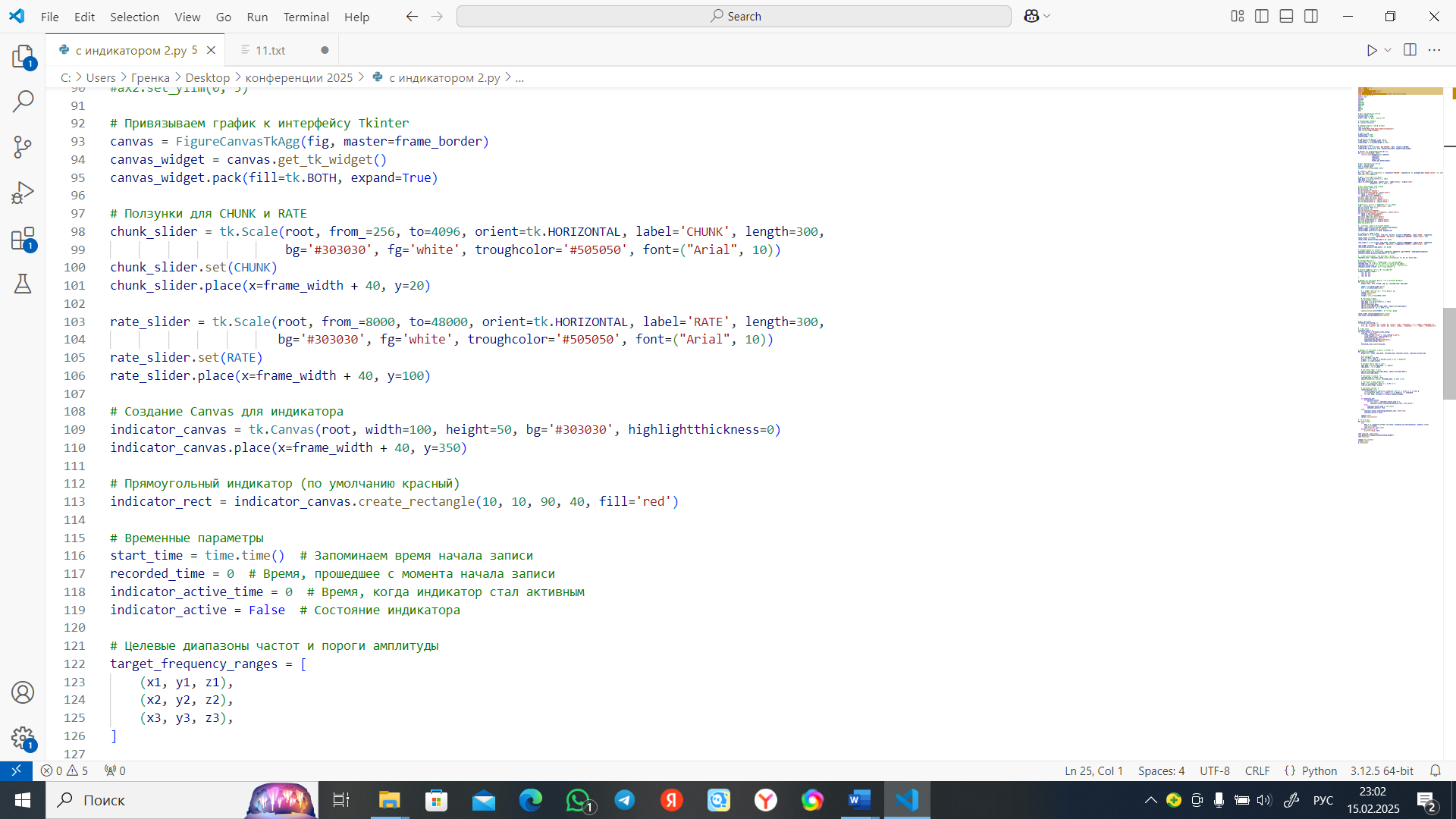


Рисунок 10. Порог амплитуды

Функция (np.linspace(0, RATE // 2, n\_fft // 2) >= low) & (np.linspace(0, RATE // 2, n\_fft // 2) <= high). Создает массив равномерно распределенных частот от 0 до половины частоты дискретизации (количество отсчетов в секунду). Массив определяет шкалу частот, использованных для анализа. (> = low) и (<= high). Все значения внутри массива будут True если частоты из шкалы попадают в диапазон.

Функция np.where принимает на вход значения массива и возвращает индексы элементов, которые равны True.(рисунок 9) Переменная y\_data содержит массив данных после обработки FFT. Здесь отбираются значения амплитуд только частот, входящих в диапазон. Функция np.max(...) находит максимальное значение амплитуды из предложенных массивом данных в определенном диапазоне частот. Функция ...> threshold сравнивает максимальное значение амплитуды с выбранным порогом threshold. Если максимальная амплитуда превышает порог, то результатом сравнения является True

Функция all принимает на вход результаты сравнения максимального значения амплитуды и порога. Если в каждом частотном диапазона значение амплитуды превышает порог, то переменная conditions\_met принимает значение True (вадрокоптер возможно присутвует), иначе – False

Если все заданные диапазоны частот превышают максимальные значения амплитуд, то квадрокоптер обнаружен и активируется индикатор.

## Обновление индикатора

Чтобы избежать случайных кратковременных активаций индикатора и ошибочного обнаружения квадрокоптера дополнительно проверяются следующие условия (рисунок 11)

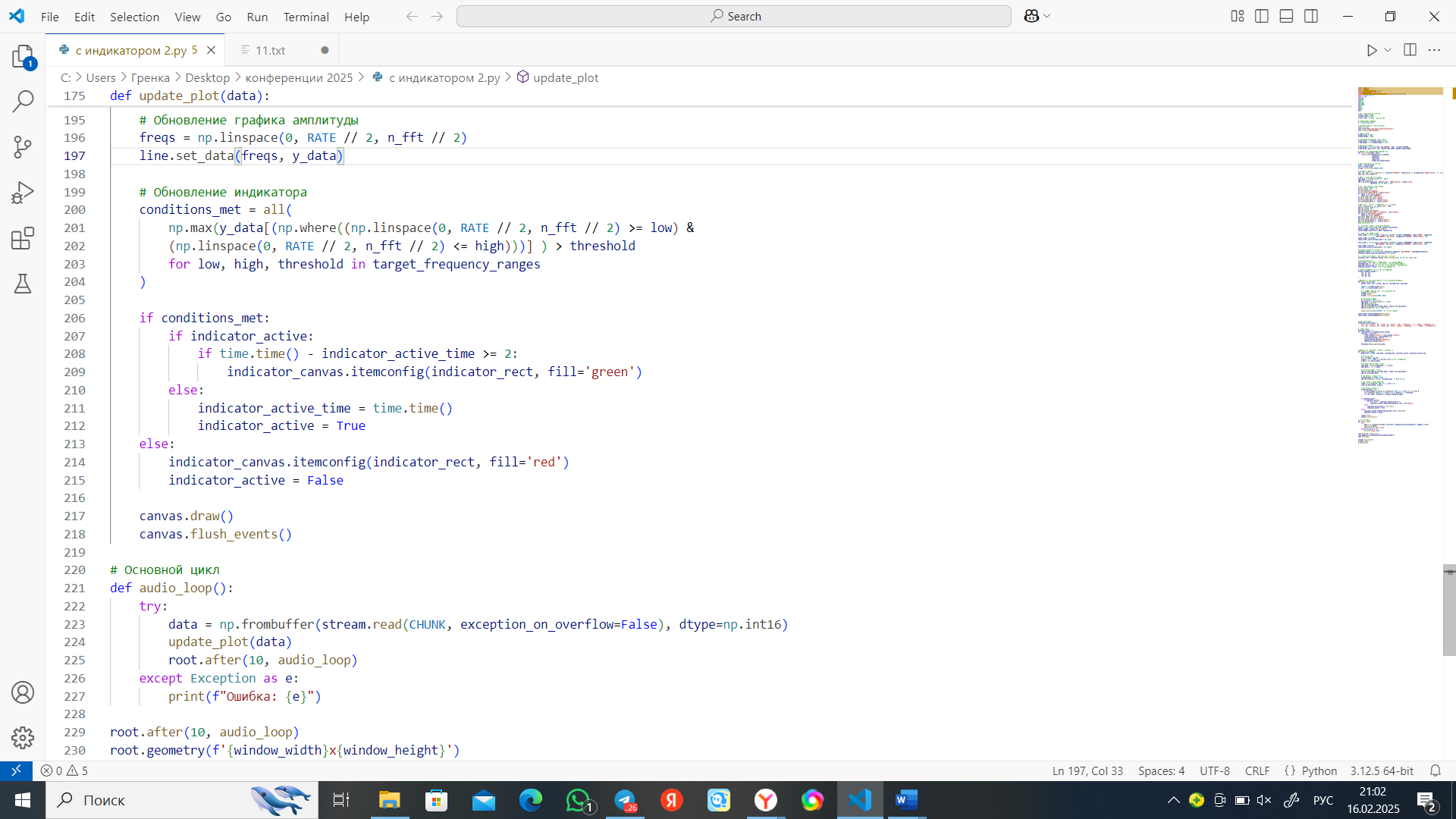


Рисунок 11.Обновление индикатора

В первом блоке условия проверяется значение переменной indicator\_active. В случае, если значение переменной ложно, то индикатор в окне приложения заполняется красным цветом и переменная indicator\_active остается в положении ложно (рисунок 13). В случае, когда значение переменной indicator\_active истинно, выполняется проверка следующих условий: выполнение условий по частотным диапазонам и если индикатор активен, а с момента его активации прошло не менее 2 секунд, то индикатор меняет цвет на зеленый (рисунок 12). Это значит, что квадрокоптер обнаружен. Если индикатор не активен, то программа запоминает время активации и индикатор становится активным

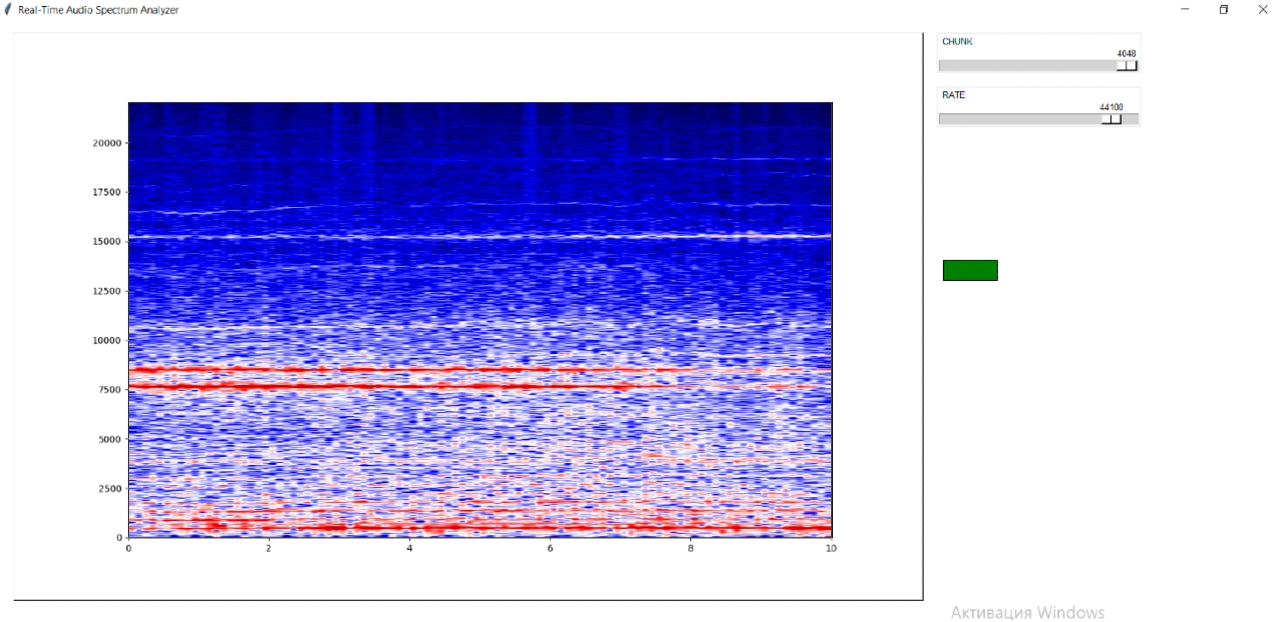


Рисунок 12. Квадрокоптер обнаружен

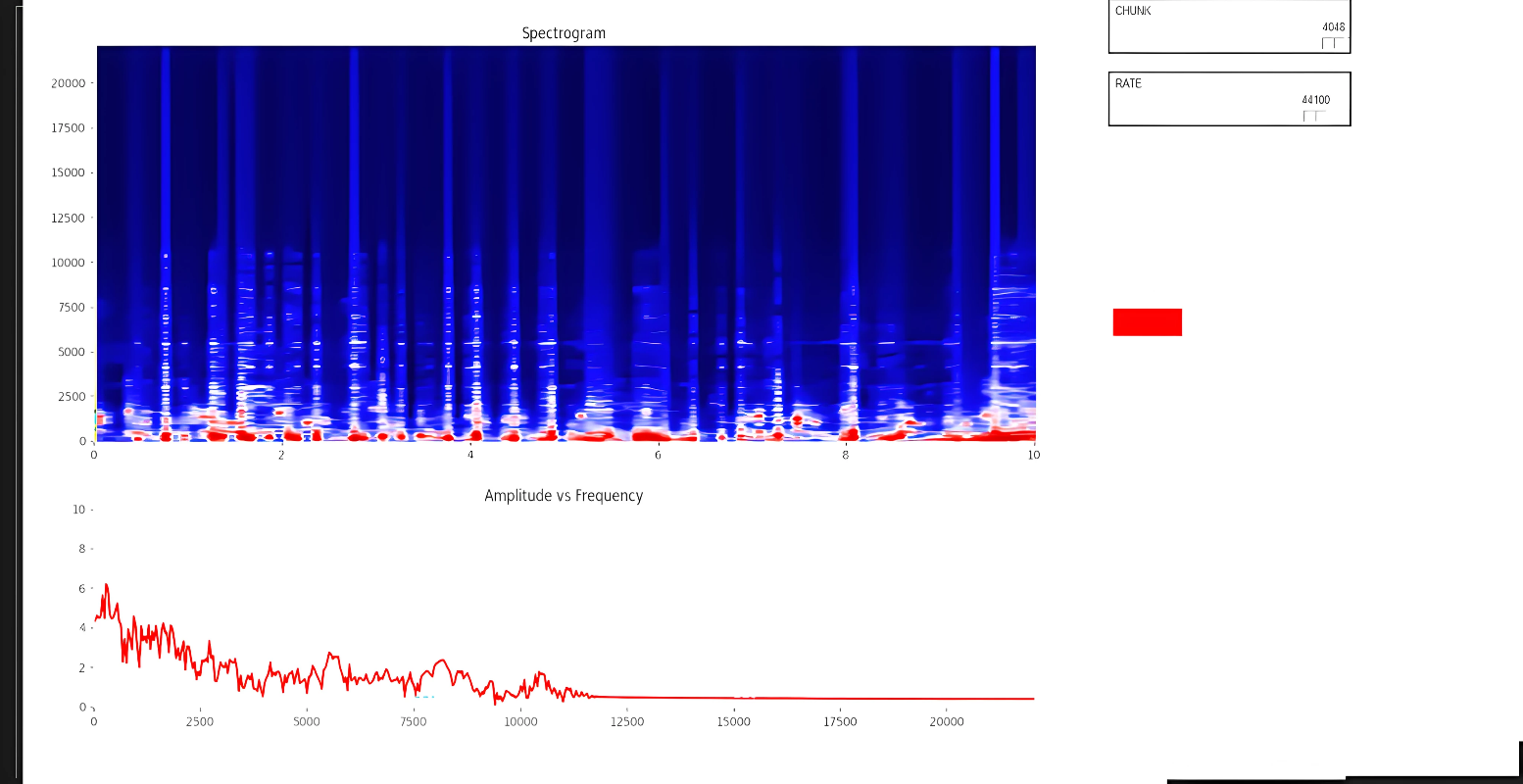


Рисунок 13. Квадрокоптер не обнаружен

# Результаты работы и их проверка

В данной работе были проведены 3 эксперимента для проверки работоспособности программы

## Эксперимент №1. В классе без шума

На рисунке 14 представлена схема эксперимента

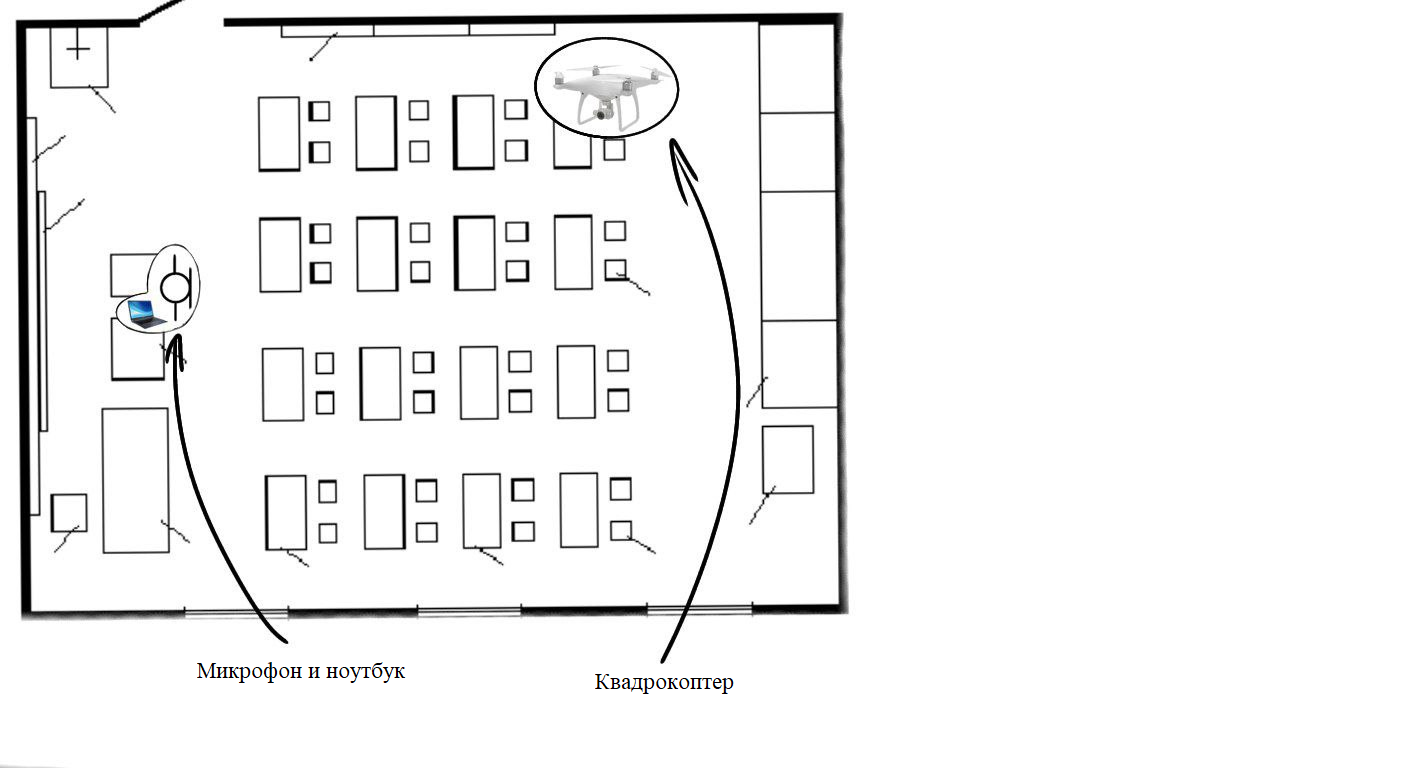


Рисунок 14. Схема эксперимента №1



Рисунок 15. Результат эксперимента №1

Расстояние от микрофона до квадрокоптера 6 метров, источник шума отсутствует. Эксперимент показал, что система обнаруживает квадрокоптер при полной тишине. (рисунок 15)

## Эксперимент №2. В классе со среднем шумом

На рисунке 16 представлена схема эксперимента

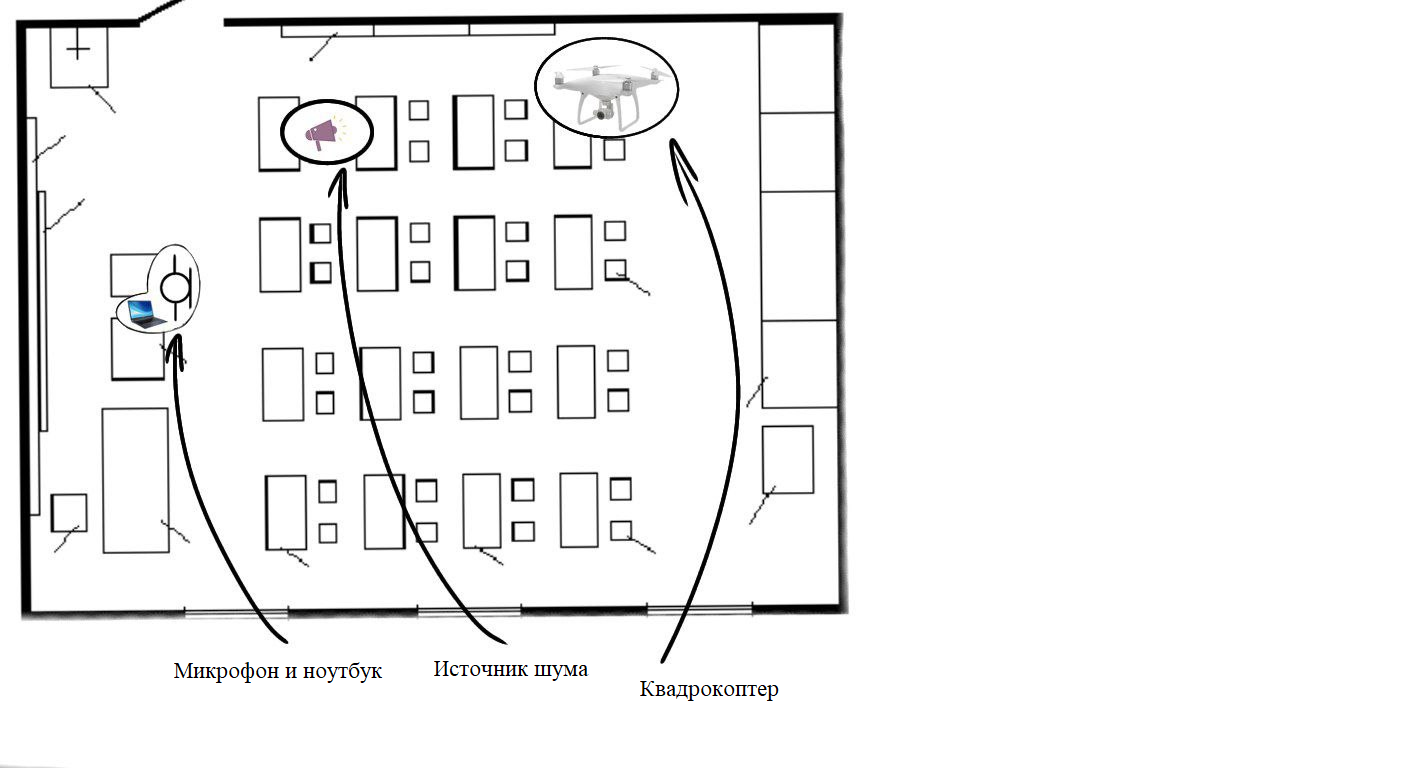


Рисунок 16. Схема эксперимента №2

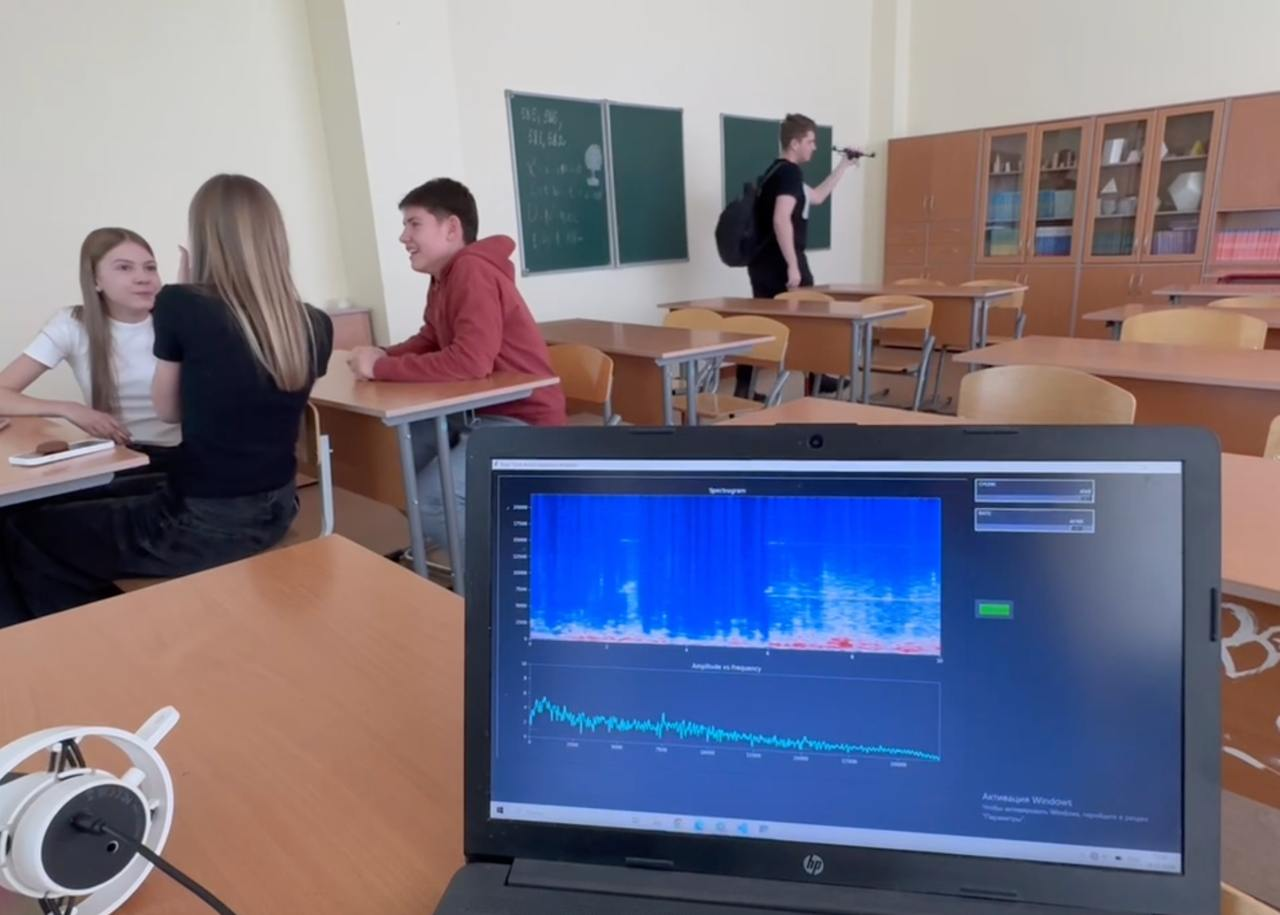


Рисунок 17. Результат эксперимента №2

Расстояние от микрофона до квадрокоптера 6 метров, источник шума – разговоры нескольких школьников. Эксперимент показал, что система обнаруживает квадрокоптер при среднем шуме (рисунок 17)

## Эксперимент №3. На улице при высоком шуме и увеличенном расстоянии.

На рисунке 18 представлена схема эксперимента

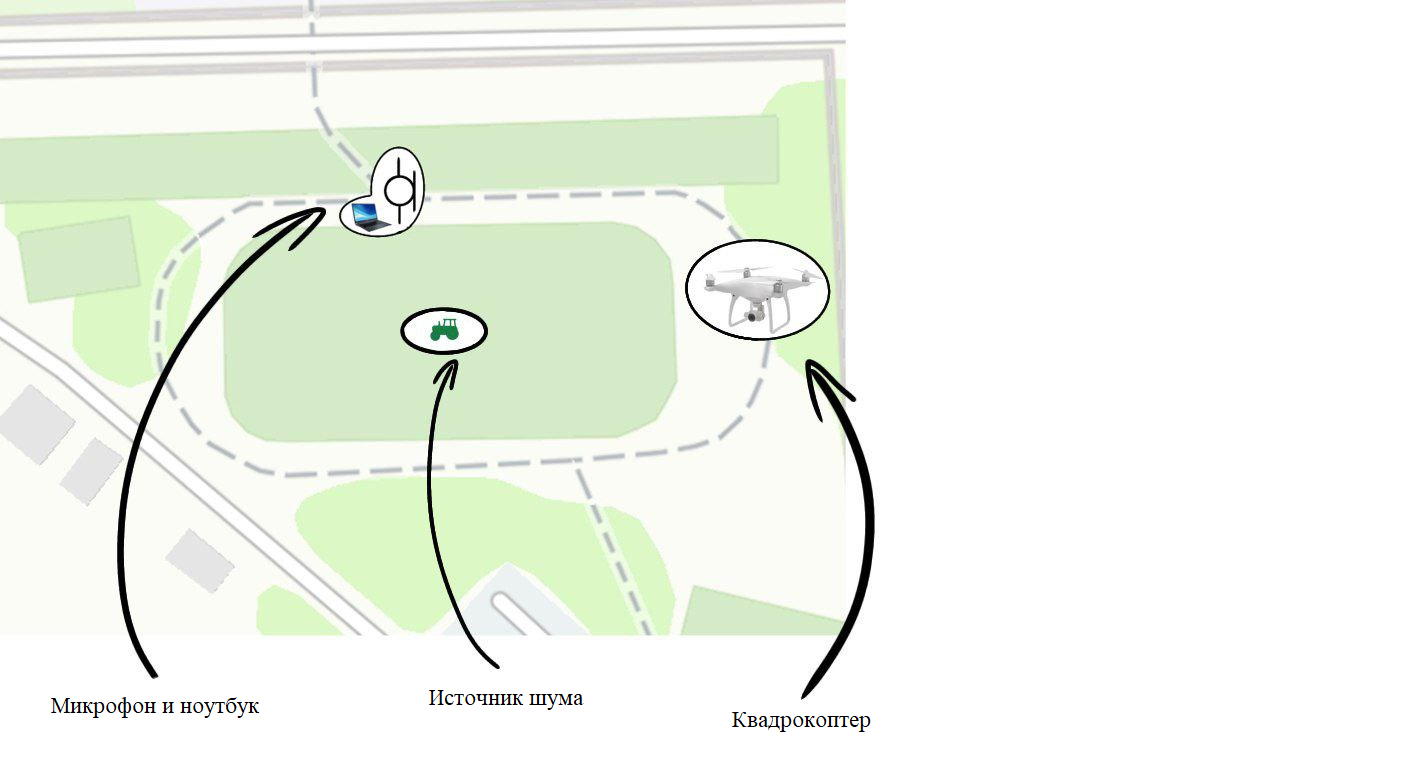


Рисунок 18. Схема эксперимента №3

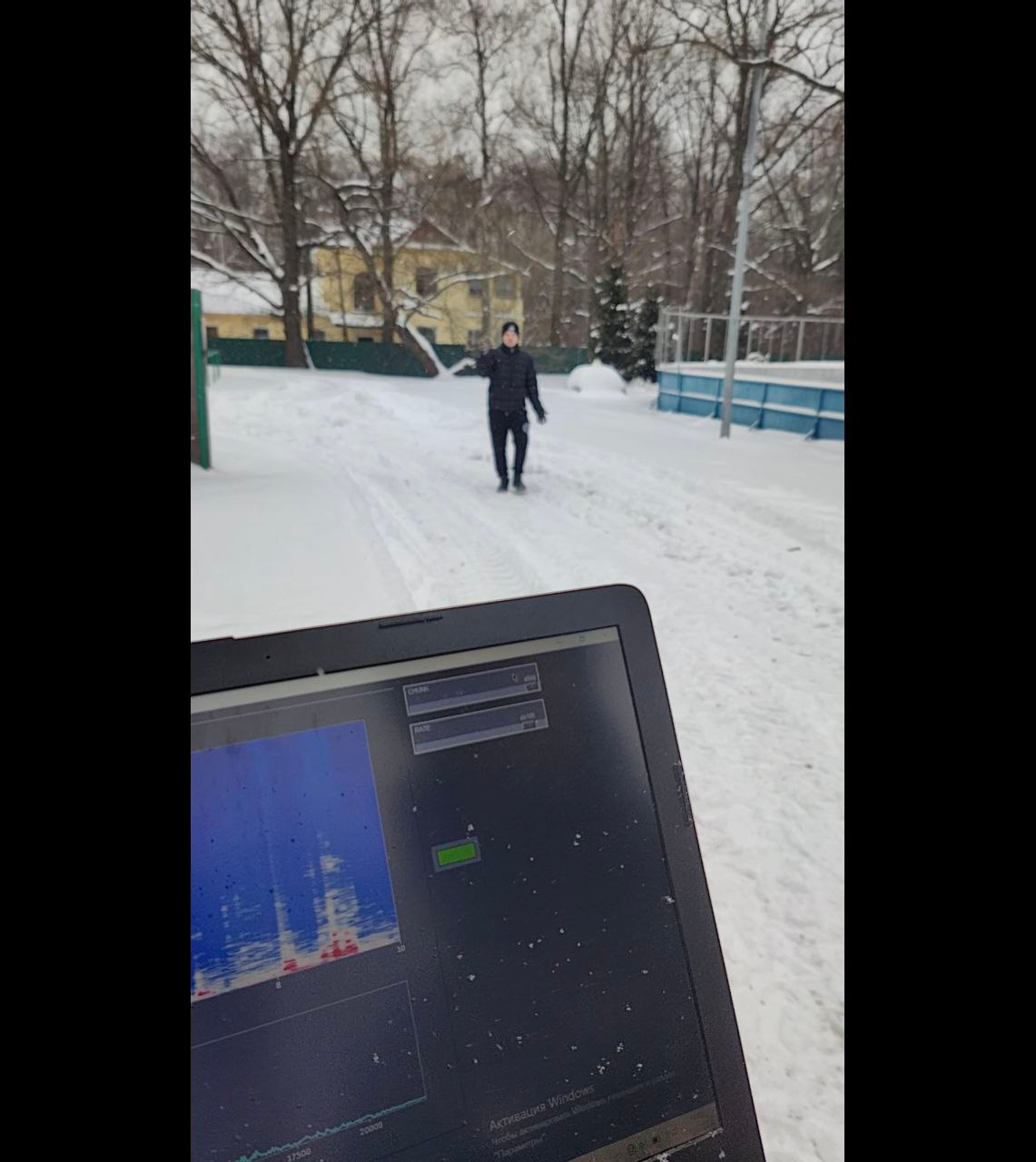


Рисунок 19. Результат эксперимента

Расстояние от микрофона до квадрокоптера – 15 метров, источник шума – трактор, работающий неподалеку. Эксперимент показал, что программа исправно работает про высокий уровень шума и большой дальности. (рисунок 19). Так же был проведен дополнительный тест на улице, который помог определить максимальную дальность работы системы – 30 метров.

Система способна обнаруживать квадрокоптер при высоком уровне шума, большой дальности. Исправно работает как в помещении, так и на улице в холод – 6 градусов.

# Выводы

Разработан алгоритм, который, позволяет обнаруживать квадрокоптер по его звуковому сигналу в режиме реального времени. Разработан алгоритм для обнаружения квадрокоптера, который построен на основе выявленных особенностей спектрального следа, используя быстрое преобразование Фурье (FFT). Изучены спектральные характеристики квадрокоптера, которые позволяют выявить характерный звуковой отпечаток, отличающий его от других источников шума. Проведен анализ звуковых сигналов квадрокоптера. Выявлены характерные частотные диапазоны и амплитудные соотношения, которые могут служить "спектральным следом" квадрокоптера в звуковой среде. Проведено три эксперимента по тестированию разработанной системы. Результаты тестирования показали, что система обнаруживает квадрокоптер в условиях легкого шума (разговоры детей), более высокого шума (шум трактора) и расстояния до 30 метров. Система сохранила свою работоспособность. Эти данные говорят о том, что алгоритм устойчив к сторонним шумам.

Разработанный алгоритм показывает хорошие результаты в установленных условиях, демонстрируя устойчивость системы к различному уровню шума. Система способна эффективно обнаруживать квадрокоптер в шумных условиях, но дальность обнаружения ограничена, максимальное значение 30 метров. Повышение дальности обнаружения является главной задачей для улучшения работы программы в дальнейшем усовершенствовании проекта.

Код системы выложен на сайте GitHub. Ссылка: <https://github.com/chikipuki2/finder>

# Список литературы

1. Python // URL: https://www.python.org/ (дата обращения: 18.02.2025).
2. PyAudio // URL: https://pypi.org/project/PyAudio/ (дата обращения: 18.02.2025).
3. Matplotlib // URL: https://pypi.org/project/matplotlib/ (дата обращения: 18.02.2025).
4. SciPy // URL: https://pypi.org/project/scipy/ (дата обращения: 18.02.2025).
5. Tkinter // URL: https://docs.python.org/3/library/tkinter.html (дата обращения: 18.02.2025).
6. Time // URL: https://docs.python.org/3/library/time.html (дата обращения: 18.02.2025).
7. An Acoustic-based Surveillance System for Amateur Drones Detection and Localization // 2020 IEEE International Conference on Robotics and Biomimetics (ROBIO) URL: https://ieeexplore.ieee.org/document/9266477 (дата обращения: 18.02.2025).