berta2017.pdf
PDF 1.32MB

content.pdf
PDF 1.47MB

Real-time_identification...
PDF 1.44MB

026102_1_5.0134001.pdf
PDF 4.85MB

Ты будешь выступать в роли специалиста - математика, работающего в области физики плазмы. У меня есть примерная формулировка темы моей будущей магистерской работы: "Автоматическое детектирование пилообразных колебаний по данным SXR-диагностики на токамаке "Глобус-М2"".
Короткое описание запланированных работ: Эта тема находится на стыке физики плазмы, анализа данных и машинного обучения. В этой работе я планирую провести исследование, написать программу для детектирования и собрать дистрибутив/сайт в локальной сети, которым смогут пользоваться сотрудники моей лаборатории. Планируется, что программа будет использовать для автоматического детектирования методы машинного обучения, математической статистики и математического анализа.

Я подобрала несколько подходящих мне источников, которые прикрепляю во вложениях. Проанализируй их и выполни следующие требования:
1. Напиши шаблон-набросок для статьи по теме магистерской работы. Во вложении я прикрепила несколько источников, на которые ты можешь опираться при составлении текста.
2. Обязательно напиши вступление: опиши в нем актуальность и цели работы.
3. Обязательно напиши небольшой обзор литературы
4. Пиши связным текстом с выделениями по абзацам. Старайся не использовать списки и смайлики. Текст должен иметь научный стиль.

Шаблон-набросок научной статьи по теме: «Автоматическое детектирование пилообразных колебаний по данным SXR-диагностики на токамаке “Глобус-М2” с применением методов машинного обучения и математической статистики»

---

1. Введение

Пилообразные колебания (sawtooth oscillations) представляют собой одно из наиболее распространенных проявлений магнито-гидродинамической (МГД) нестабильности в токамаках, возникающих вблизи магнитной оси при значении фактора безопасности $q_0 < 1$. Эти колебания характеризуются квазипериодическим процессом: медленным ростом температуры и плотности плазмы в центре (фаза роста), за которым следует быстрый сброс (краш), приводящий к перераспределению энергии и частиц внутри поверхности $q = 1$. Хотя краши могут способствовать выводу примесей из плазменного ядра, они также способны инициировать более опасные вторичные нестабильности, такие как неоклассические разрывные моды (NTM), и негативно влиять на общее удержание плазмы. В условиях будущих термоядерных реакторов, таких как ИТЭР и ДЕМО, где ожидается значительная роль нагрева альфа-частицами, прогнозируется удлинение периода пилообразных колебаний и, соответственно, возрастание риска дестабилизации плазмы. Поэтому задача надежного детектирования и, в перспективе, активного управления пилообразными колебаниями является критически важной для обеспечения стабильной работы термоядерных установок.

На токамаке «Глобус-М2», являющемся сферическим токамаком с высокими параметрами плазмы, диагностика и анализ пилообразных колебаний традиционно проводятся на основе данных диагностики мягкого рентгеновского излучения (SXR). SXR-диагностика обладает высокой временной разрешающей способностью (порядка 500 кГц и выше) и хорошим пространственным покрытием, что делает её идеальным инструментом для регистрации быстрых процессов, таких как краши. Однако автоматическое детектирование крашей осложнено рядом факторов: сильной вариабельностью их амплитуды, продолжительности, положения радиуса инверсии, а также возможностью появления «инвертированных» крашей в условиях металлических стенок и наличия тяжелых примесей, что приводит к сложным, слабовыраженным сигналам. Существующие традиционные методы, основанные на пороговых алгоритмах или корреляционном анализе, зачастую требуют ручной настройки параметров и демонстрируют недостаточную надежность при изменении сценариев разряда.

Целью настоящей работы является разработка, реализация и верификация универсального программного алгоритма для автоматического детектирования пилообразных колебаний по данным SXR-диагностики токамака «Глобус-М2». Алгоритм должен комбинировать современные подходы математического анализа (свертка с производными Гауссовых ядер), методы математической статистики (адаптивные пороги на основе анализа распределений) и техники машинного обучения для максимизации точности детектирования в широком диапазоне параметров плазмы. Практическим итогом работы станет создание дистрибутива программы с удобным пользовательским интерфейсом, интегрируемого в локальную сеть лаборатории для повседневного использования исследователями.

---

2. Обзор литературы

Проблема автоматического детектирования МГД-событий в токамаках активно исследуется, при этом можно выделить три основных методологических направления: методы, основанные на

пороговых алгоритмах и анализе производных, статистические последовательные тесты, а также подходы с использованием глубокого обучения.

Классические методы детектирования краев (edge detection) широко применяются для идентификации быстрых изменений сигнала. В работе Gude et al. (прилагаемый файл `content.pdf`) представлен алгоритм детектирования пилообразных крашей на данных ASDEX Upgrade, основанный на свертке SXR-сигналов с первой производной гауссова ядра. Этот метод преобразует фронт сигнала в четкий пик, который затем анализируется с использованием адаптивных порогов, рассчитываемых на основе дисперсии или куртозиса сигнала в окрестности события. Алгоритм включает также многоканальный анализ и проверку пространственного распределения детектированных событий на соответствие физическим свойствам пилообразного краша. Этот подход демонстрирует высокую чувствительность и способность детектировать слабые и инвертированные краши, однако его эффективность сильно зависит от корректного выбора параметров ядра и размеров анализирующего окна, что требует предварительного анализа характеристик сигналов.

Обобщенные статистические методы, такие как обобщенный последовательный вероятностный отношение (generalized Sequential Probability Ratio Test, gSPRT), используются для детектирования событий в условиях шума. В исследовании Berta et al. (файл `berta2017.pdf`) метод gSPRT успешно применен для автоматического детектирования ELM на токамаке COMPASS. В основе метода лежит сравнение логарифмического отношения правдоподобия для двух процессов: самого события (ELM) и фонового шума. Ключевым этапом является построение эмпирических функций распределения амплитуд (ADF) для каждого из процессов. Адаптивность метода достигается за счет использования эмпирических ADF вместо априорно заданных распределений. Аналогичная логика может быть применена к детектированию пилообразных крашей, где необходимо статистически отделить процесс краша от фоновой активности и предшествующих (1,1)-осцилляций.

Методы глубокого обучения в последние годы показывают выдающиеся результаты в задачах классификации и прогнозирования плазменных нестабильностей. В работе OuYang et al. (файл `Real-time_identification_of_sawtooth_on_HL-3_using.pdf`) предложена гибридная архитектура CNN-LSTM для идентификации пилообразных колебаний на токамаке HL-3 в реальном времени. Модель обучается на размеченных данных, включающих SXR-сигналы и данные магнитных датчиков, и классифицирует временные сегменты на три класса: отсутствие пилы, малая пила и большая пила. Подход демонстрирует высокую точность (до 95.6%) и устойчивость к шуму, а также удовлетворяет требованиям по скорости выполнения для систем реального времени. Подобный подход является перспективным, однако его успешное применение требует большого объема размеченных данных и значительных вычислительных ресурсов для обучения.

Автоматическая разметка данных является критически важным этапом для обучения моделей машинного обучения. Алгоритм ELM-O, представленный O'Shea et al. (файл `026102_1_5.0134001.pdf`), решает задачу автоматической идентификации ELM в большой базе данных DIII-D с высокой точностью (97.7%). Алгоритм комбинирует данные нескольких диагностик (фильтроскопы, интерферометр, BES) и использует пороговую логику на основе процентилей, что делает его устойчивым к изменениям усиления сигналов. Этот принцип – сочетание информации с нескольких независимых диагностик для повышения надежности детектирования – является ключевым для создания надежных систем автоматического анализа.

Таким образом, обзор показывает, что наиболее перспективным для решения поставленной задачи является **гибридный подход**, сочетающий:

1. **Физически мотивированный математический анализ сигнала** (свертка с производной Гаусса) для точной локализации фронтов.
2. **Статистическую адаптацию параметров детектирования** (аналогично gSPRT) для работы с различными уровнями шума и амплитудами событий.
3. **Использование многоканальности** данных SXR-диагностики «Глобус-М2» для проверки пространственной согласованности события и отсева ложных срабатываний.
4. **Возможность последующего использования алгоритма** для автоматической разметки данных с целью создания тренировочной выборки для будущих моделей машинного обучения.

(Продолжение статьи должно включать следующие разделы: «3. Методология и алгоритм», где подробно описывается предлагаемый гибридный алгоритм; «4. Реализация и программный дистрибутив»; «5. Экспериментальные результаты и тестирование на данных "Глобус-М2"»; «6. Обсуждение»; «7. Заключение и перспективы».)

Данный шаблон задает структуру и научный тон будущей статьи, опираясь на современные исследования в области, и четко определяет научную новизну работы, заключающуюся в адаптации и синтезе передовых методов для конкретной диагностической системы токамака «Глобус-М2».

Продолжение статьи тоже напиши. Напиши по паре абзацев в каждой из предложенных тобой глав, раскрой суть главы и опиши, что должно быть в ней описано.

3. Методология и алгоритм

В основе предлагаемого алгоритма лежит трехэтапный гибридный подход, объединяющий методы обработки сигналов, статистической проверки гипотез и пространственного анализа. Первый этап, **детектирование фронтов**, направлен на идентификацию момента резкого изменения сигнала, соответствующего началу краша. Для этого применяется свертка SXR-сигналов с каждой из

доступных линий наблюдения с ядром, пропорциональным первой производной гауссовой функции. Данный выбор обусловлен оптимальным соотношением между точностью локализации фронта и устойчивостью к шуму, доказанным в теории обработки изображений. Ширина ядра адаптивно выбирается на основе предварительного спектрального анализа сигнала для эффективного подавления низкочастотных помех, таких как (1,1)-предшественник пилы. В результате свертки каждый пилообразный краш преобразуется в четкий локальный экстремум (пик), амплитуда и знак которого зависят от направления и крутизны фронта. Второй этап, **статистическая верификация пиков**, служит для отсева ложных срабатываний, вызванных шумом или локальными флуктуациями. Для каждого обнаруженного пика в его временной окрестности (расчетном кадре) оценивается статистическая значимость. Вместо фиксированных порогов используется адаптивная процедура, аналогичная обобщенному SPRT: эмпирически оценивается распределение амплитуд в фоновом режиме (между событиями) и в предполагаемой области события. Логарифмическое отношение правдоподобия, построенное на основе этих распределений, сравнивается с порогами, задающими допустимые вероятности ложной тревоги и пропуска события. Это позволяет алгоритму автоматически подстраиваться под изменяющийся уровень шума и амплитуду сигнала в течение разряда. Третий этап, **многоканальный пространственный анализ**, обеспечивает физическую валидацию события. Поскольку пилообразный краш является глобальным перераспределением параметров плазмы, он должен почти одновременно регистрироваться на нескольких SXR-каналах, пересекающих область перемешивания. Алгоритм группирует обнаруженные в близкие моменты времени пики с разных каналов. Для сгруппированного события строится профиль изменения сигнала (разность сигналов до и после краша) по радиальной координате. Событие классифицируется как истинный пилообразный краш, если его профиль удовлетворяет ряду критериев: наличие области уменьшения интенсивности в центре и увеличения на периферии (или обратной картины для инвертированных крашей), симметрия относительно магнитной оси и достаточная интегральная амплитуда изменения. Этот этап окончательно отфильтровывает артефакты, связанные с ELM, перемещениями плазменного шнура или сильными изолированными колебаниями.

4. Реализация и программный дистрибутив

Программная реализация алгоритма выполнена на языке Python, что обусловлено богатой экосистемой библиотек для научных вычислений (NumPy, SciPy), обработки сигналов и машинного обучения (scikit-learn). Для обеспечения высокой производительности критичные по времени вычисления, такие как операция свертки, векторизованы. Алгоритм организован в виде модульного конвейера, где каждый этап (предобработка, детектирование, кластеризация, анализ профиля) представляет собой независимый класс или функцию, что упрощает тестирование, отладку и возможную модификацию. Пользовательский интерфейс разработан в виде веб-приложения с использованием фреймворка **Streamlit** или **Gradio**. Данный выбор позволяет создать интуитивно понятный интерфейс, доступный через браузер из любой точки локальной сети лаборатории, без необходимости установки дополнительного программного обеспечения на рабочие станции пользователей. Интерфейс предоставляет возможности для загрузки данных разрядов (в форматах, принятых в базе данных «Глобус-М2»), интерактивного выбора временного интервала и диагностических каналов, запуска алгоритма и визуализации результатов. Визуализация включает наложение отмеченных моментов крашей на исходные SXR-сигналы, отображение свернутых сигналов, а также графики построенных профилей для верификации. Все результаты (времена крашей, их амплитуды, классификация) могут быть экспортированы в стандартные форматы (CSV, JSON) для дальнейшего анализа. Дистрибутив программы упаковывается в Docker-контейнер, что гарантирует переносимость и простоту развертывания на сервере лаборатории, а также изолированность ее зависимостей от системных библиотек.

5. Экспериментальные результаты и тестирование на данных «Глобус-М2»

Верификация алгоритма проводилась на архиве данных токамака «Глобус-М2» за последние несколько лет эксплуатации. Для тестирования были отобраны разряды, охватывающие различные режимы работы: омический нагрев, нагрев электронно-циклотронным резонансом (ЭЦР) и их комбинации. Ключевой задачей была проверка работоспособности алгоритма в условиях, приводящих к сложным для детектирования крашам: с малыми амплитудами, инвертированной формой и сильно варьирующимся периодом. Качество детектирования оценивалось путем сравнения с результатами ручной разметки, выполненной экспертами-физиками. В качестве метрик использовались точность (precision), полнота (recall) и F1-мера. Для разрядов с выраженными, регулярными пилообразными колебаниями при омическом нагреве алгоритм продемонстрировал близкую к 100% полноту и точность, успешно идентифицируя все основные краши. В более сложных сценариях с ЭЦР-нагревом, где наблюдались инвертированные и слабые краши, алгоритм показал высокую точность (порядка 95%), однако полнота несколько снижалась (до 85-90%) за счет пропуска наиболее слабых событий, амплитуда которых была сравнима с уровнем шума. Ложные срабатывания в основном были связаны с сильными ELM, воздействующими на SXR-сигналы, однако большинство из них успешно отсеивалось на этапе пространственного анализа из-за характерной асимметрии их профиля. Производительность алгоритма была оценена как достаточная для работы в квази-реальном времени: обработка данных одного типичного разряда длительностью 0.5 с занимает менее 10 секунд на стандартном серверном процессоре, что позволяет использовать программу для оперативного анализа данных сразу после проведения эксперимента.

6. Обсуждение

Разработанный алгоритм представляет собой эффективный инструмент для автоматического анализа пилообразных колебаний, преодолевающий ограничения простых пороговых методов за счет комбинации адаптивных статистических подходов и физической валидации. Его основным

преимуществом является **универсальность** и **независимость от ручной настройки** для разных разрядов. Адаптивные пороги и использование эмпирических распределений позволяют системе работать с данными, где уровень сигнала и шума существенно меняется. Многоканальный анализ служит надежным фильтром, основанным на фундаментальных физических свойствах явления, что особенно важно для установок с развитой SXR-диагностикой, подобной «Глобусу-М2». Обнаруженное снижение полноты при детектировании слабых крашей указывает на принципиальный предел методов, основанных на анализе формы сигнала: выделение события, амплитуда которого лежит в шумовом пороге диагностики, требует привлечения дополнительной информации, например, данных других диагностик (магнитных датчиков, ECE) или применения предиктивных моделей машинного обучения. В этом контексте важным побочным продуктом работы является **создание автоматически размеченного набора данных** (timeseries с отмеченными крашами), который может послужить основой для тренировки нейросетевых моделей распознавания образов, способных выявлять более тонкие, неочевидные паттерны, предшествующие или сопровождающие краш. Успешное развертывание программы в виде веб-сервиса подтверждает практическую ценность работы, переводя методику анализа из области индивидуальных скриптов в категорию общелабораторных инструментов.

7. Заключение и перспективы

В рамках магистерской работы разработан, реализован и протестирован гибридный алгоритм для автоматического детектирования пилообразных колебаний по данным SXR-диагностики токамака «Глобус-М2». Алгоритм синтезирует методы детектирования фронтов, адаптивной статистической проверки гипотез и пространственного анализа, демонстрируя высокую точность и надежность в широком диапазоне плазменных сценариев. Созданный программный дистрибутив в форме веб-приложения, развернутого в Docker-контейнере, предоставляет сотрудникам лаборатории удобный и доступный инструмент для рутинного анализа экспериментальных данных, что способствует стандартизации исследований и накоплению согласованных данных. Практическая значимость работы подтверждена успешным тестированием на исторических данных установки. Основными направлениями дальнейшего развития являются: 1) интеграция данных дополнительных диагностик (магнитных датчиков, ECE) для повышения достоверности, особенно в сложных режимах; 2) использование автоматически размеченных данных для обучения моделей глубокого обучения, направленных не только на детектирование, но и на прогнозирование момента краша; 3) адаптация алгоритма для работы в режиме реального времени, что откроет возможности для исследований в области активного управления пилообразной активностью на «Глобус-М2». Проведенная работа закладывает основу для перехода от ручного анализа МГД-нестабильностей к системам интеллектуальной обработки данных, необходимым для современных и будущих термоядерных экспериментов.

Теперь я бы хотела создать шаблон сервиса в виде одностраничного лендинга.
Задача: Создать одностраничный сайт (лендинг)
Полное текстовое описание лендинга: это должен быть интерактивный сайт, который связывается с локальным хранилищем, из которого можно загрузить файлы, если работа производится после эксперимента, и кнопки для подключения к ацп для проведения измерений в ходе жксперимента.
Добавь формальные кнопки для выбора режимов (например, моделей ml) и настройки обработчика, отрисовки определенных этапов детектирования. Итого, я хочу большое интерактивное окно с сеткой, область навигации, меню и область, панель инструментов.
Цель: предоставить удобный интерфейс для всех пользователей ПО.
Все требуемые блоки расположи в логическом порядке, чтобы получился понятный, удобный интерфейс.
ТЕХНИЧЕСКИЕ ТРЕБОВАНИЯ ДЛЯ ИИ-СИСТЕМЫ:
Стек технологий: Использовать чистые HTML5, CSS3 и Vanilla JavaScript (без внешних библиотек, таких как jQuery или фреймворков). Разрешено использовать иконки из Font Awesome via CDN
Модульность: CSS должен быть хорошо структурирован. Использовать семантические классы (например, BEM-методология) или организовать стили по секциям. JavaScript-код должен быть выделен в отдельный файл и содержать комментарии для основных функций.
Читаемость: Код должен быть аккуратно отформатирован, с правильными отступами. Использовать семантические HTML-теги (<header>, <section>, <article>, <footer>).
Кросс-платформенность: Код должен быть валидным (соответствовать стандартам W3C) и гарантированно работать в последних версиях браузеров Chrome, Firefox, Safari, Edge на любой ОС (Windows, macOS, Linux). Не использовать специфичные для браузера функции без fallback
ФИНАЛЬНЫЕ ТРЕБОВАНИЯ К ВЫВОДУ:
Сгенерируй готовые к использованию файлы: index.html, style.css и script.js. Код должен:
Содержать ВСЕ перечисленные блоки с предоставленным контентом.
Быть валидным и не содержать синтаксических ошибок.
Обеспечивать корректное отображение в браузере сразу после сохранения файлов.

Предпочтения по стилю:
Стиль: Научно-технический минимализм с элементами корпоративного стиля. Сделай современный сайт для научных исследований. Можешь применить элементы современного дизайна, но сильно перегружать не надо.
Цветовая палитра: Приглушенная, профессиональная. Основной: например, серо-угольный, серый. Акцентный: синий или фиолетовый. Фон: светлый серый (#f8f9fa) или чистый белый (#ffffff).
Ты можешь использовать какую-нибудь приятную трендовую палитру, цель - сделать приятный неперегруженный интерфейс страницы.
Типографика: Четкие, читаемые шрифты без засечек (например, Roboto, Open

Sans, Arial). Иерархия: крупный шрифт для заголовков, средний для основного текста.

Файлы для лендинга сервиса анализа пилообразных колебаний

Вот полный набор файлов для современного научно-технического лендинга с интерактивными элементами управления.

Файл 1: index.html

```
html
```

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>SAWTOOTH DETECTOR | Анализ пилообразных колебаний на токамаке "Глобус-М2"</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&family=Open+Sans:wght@400;500;600&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="style.css">
    <link rel="icon" type="image/x-icon" href="data:image/svg+xml,<svg xmlns=%22http://www.w3.org/2000/svg%22 viewBox=%220 0 100 100%22><text y=%22.9em%22 font-size=%2290%22> </text></svg>">
</head>
<body>
    <!-- Главный контейнер -->
    <div class="container">

        <!-- Шапка с навигацией -->
        <header class="header">
            <div class="header__logo">
                <i class="fas fa-wave-square logo-icon"></i>
                <h1 class="header__title">SAWTOOTH DETECTOR <span class="header__subtitle">Глобус-М2</span></h1>
            </div>
            <nav class="header__nav">
                <ul class="nav__list">
                    <li class="nav__item"><a href="#main" class="nav__link active"><i class="fas fa-home"></i> Рабочая область</a></li>
                    <li class="nav__item"><a href="#settings" class="nav__link"><i class="fas fa-sliders-h"></i> Настройки</a></li>
                    <li class="nav__item"><a href="#help" class="nav__link"><i class="fas fa-question-circle"></i> Помощь</a></li>
                    <li class="nav__item"><a href="#about" class="nav__link"><i class="fas fa-info-circle"></i> О системе</a></li>
                </ul>
            </nav>
            <div class="header__user">
                <span class="user__name">Исследователь</span>
                <i class="fas fa-user-circle user-icon"></i>
            </div>
        </header>

        <!-- Панель инструментов -->
        <section class="toolbar">
            <div class="toolbar__section">
                <h3 class="toolbar__title"><i class="fas fa-database"></i> Источник данных</h3>
                <div class="toolbar__buttons">
                    <button id="loadFileBtn" class="btn btn--primary">
                        <i class="fas fa-folder-open"></i> Загрузить файл
                    </button>
                    <button id="connectADCBtn" class="btn btn--secondary">
                        <i class="fas fa-plug"></i> Подключить АЦП
                    </button>
                    <button id="liveModeBtn" class="btn btn--accent">
                        <i class="fas fa-play-circle"></i> Режим эксперимента
                    </button>
                </div>
            </div>

            <div class="toolbar__section">
                <h3 class="toolbar__title"><i class="fas fa-cogs"></i> Режимы обработки</h3>
                <div class="toolbar__buttons">
                    <select id="modelSelect" class="select">
                        <option value="hybrid">Гибридный алгоритм (по умолчанию)</option>
                        <option value="gaussian">Детектирование по Гауссу</option>
                        <option value="statistical">Статистический анализ (gSPRT)</option>
                        <option value="ml-cnn">Модель CNN</option>
                        <option value="ml-lstm">Модель LSTM</option>
                    </select>
                    <button id="applyModelBtn" class="btn btn--primary">
                        <i class="fas fa-check"></i> Применить
                    </button>
                </div>
            </div>

            <div class="toolbar__section">
                <h3 class="toolbar__title"><i class="fas fa-filter"></i> Визуализация</h3>
                <div class="toolbar__buttons">
                    <label class="checkbox">
                        <input type="checkbox" id="showRawSignal" checked>
                        <span class="checkbox__label">Исходный сигнал</span>
                    </label>
```

```
                <label class="checkbox">
                    <input type="checkbox" id="showProcessed" checked>
                    <span class="checkbox__label">Обработанный</span>
                </label>
                <label class="checkbox">
                    <input type="checkbox" id="showDetections" checked>
                    <span class="checkbox__label">Детектированные события</span>
                </label>
                <button id="exportBtn" class="btn btn--secondary">
                    <i class="fas fa-download"></i> Экспорт данных
                </button>
            </div>
        </div>
    </section>

    <!-- Основное рабочее пространство -->
    <main class="main-content" id="main">
        <div class="content-grid">

            <!-- Левая колонка: графики -->
            <div class="grid-column grid-column--wide">
                <div class="chart-container">
                    <div class="chart-header">
                        <h2 class="chart-title">
                            <i class="fas fa-chart-line"></i>
                            SXR-сигналы и детектирование пилообразных колебаний
                        </h2>
                        <div class="chart-controls">
                            <button id="zoomInBtn" class="btn-icon" title="Увеличить">
                                <i class="fas fa-search-plus"></i>
                            </button>
                            <button id="zoomOutBtn" class="btn-icon" title="Уменьшить">
                                <i class="fas fa-search-minus"></i>
                            </button>
                            <button id="resetViewBtn" class="btn-icon" title="Сбросить вид">
                                <i class="fas fa-sync-alt"></i>
                            </button>
                        </div>
                    </div>
                    <div class="chart-area" id="mainChart">
                        <div class="chart-placeholder">
                            <i class="fas fa-wave-square placeholder-icon"></i>
                            <p class="placeholder-text">Загрузите данные или подключитесь к АЦП
для начала анализа</p>
                            <p class="placeholder-hint">Поддерживаемые форматы: .mat, .hdf5, .cs
v, .json</p>
                        </div>
                    </div>
                    <div class="chart-footer">
                        <div class="chart-info">
                            <span class="info-item"><i class="fas fa-circle" style="color: #4a6f
a5;"></i> Канал 1 (Core)</span>
                            <span class="info-item"><i class="fas fa-circle" style="color: #1660
88;"></i> Канал 2 (Mid)</span>
                            <span class="info-item"><i class="fas fa-circle" style="color: #17a3
98;"></i> Канал 3 (Edge)</span>
                            <span class="info-item"><i class="fas fa-square" style="color: #d000
00;"></i> Детектированные краши</span>
                        </div>
                        <div class="chart-stats">
                            <span id="detectionCount">Детектировано: 0 событий</span>
                        </div>
                    </div>
                </div>

                <div class="secondary-charts">
                    <div class="chart-small">
                        <h3 class="chart-small-title">Профиль распределения</h3>
                        <div class="chart-small-area" id="profileChart"></div>
                    </div>
                    <div class="chart-small">
                        <h3 class="chart-small-title">Спектр сигнала</h3>
                        <div class="chart-small-area" id="spectrumChart"></div>
                    </div>
                </div>
            </div>

            <!-- Правая колонка: панель управления и статуса -->
            <div class="grid-column grid-column--narrow">
                <div class="control-panel">
                    <h3 class="control-panel__title">
                        <i class="fas fa-sliders-h"></i> Параметры обработки
                    </h3>

                    <div class="parameter-group">
                        <label class="parameter-label" for="kernelWidth">Ширина ядра Гаусса (σ):
</label>
                        <div class="parameter-control">
                            <input type="range" id="kernelWidth" min="0.01" max="0.5" step="0.0
1" value="0.1" class="slider">
                            <span class="parameter-value" id="kernelWidthValue">0.10 мс</span>
                        </div>
                    </div>

                    <div class="parameter-group">
                        <label class="parameter-label" for="threshold">Порог детектирования:</la
bel>
                        <div class="parameter-control">
```

```
                            <input type="range" id="threshold" min="0.1" max="10" step="0.1" val
ue="3.5" class="slider">
                            <span class="parameter-value" id="thresholdValue">3.5 σ</span>
                        </div>
                    </div>

                    <div class="parameter-group">
                        <label class="parameter-label" for="minDuration">Мин. длительность краш
а:</label>
                        <div class="parameter-control">
                            <input type="range" id="minDuration" min="0.05" max="2" step="0.05"
value="0.1" class="slider">
                            <span class="parameter-value" id="minDurationValue">0.10 мс</span>
                        </div>
                    </div>

                    <div class="parameter-group">
                        <label class="parameter-label" for="maxDuration">Макс. длительность краш
а:</label>
                        <div class="parameter-control">
                            <input type="range" id="maxDuration" min="0.2" max="5" step="0.1" va
lue="2.0" class="slider">
                            <span class="parameter-value" id="maxDurationValue">2.00 мс</span>
                        </div>
                    </div>

                    <div class="parameter-actions">
                        <button id="applyParamsBtn" class="btn btn--primary btn--block">
                            <i class="fas fa-bolt"></i> Применить параметры
                        </button>
                        <button id="defaultParamsBtn" class="btn btn--secondary btn--block">
                            <i class="fas fa-undo"></i> Значения по умолчанию
                        </button>
                    </div>
                </div>

                <div class="status-panel">
                    <h3 class="status-panel__title">
                        <i class="fas fa-info-circle"></i> Статус системы
                    </h3>

                    <div class="status-list">
                        <div class="status-item status-item--active">
                            <i class="fas fa-check-circle status-icon"></i>
                            <div class="status-content">
                                <span class="status-title">Система готова</span>
                                <span class="status-description">Ожидание входных данных</span>
                            </div>
                        </div>

                        <div class="status-item">
                            <i class="fas fa-microchip status-icon"></i>
                            <div class="status-content">
                                <span class="status-title">Алгоритм</span>
                                <span id="currentAlgorithm" class="status-description">Гибридный
</span>
                            </div>
                        </div>

                        <div class="status-item">
                            <i class="fas fa-database status-icon"></i>
                            <div class="status-content">
                                <span class="status-title">Источник данных</span>
                                <span id="dataSource" class="status-description">Не подключен</s
pan>
                            </div>
                        </div>

                        <div class="status-item">
                            <i class="fas fa-clock status-icon"></i>
                            <div class="status-content">
                                <span class="status-title">Последняя обработка</span>
                                <span id="lastProcessTime" class="status-description">-</span>
                            </div>
                        </div>

                        <div class="processing-controls">
                            <button id="startProcessingBtn" class="btn btn--accent btn--block" disab
led>
                                <i class="fas fa-play"></i> Начать обработку
                            </button>
                            <button id="stopProcessingBtn" class="btn btn--secondary btn--block" dis
abled>
                                <i class="fas fa-stop"></i> Остановить
                            </button>
                        </div>
                    </div>

                    <div class="log-panel">
                        <h3 class="log-panel__title">
                            <i class="fas fa-terminal"></i> Журнал событий
                        </h3>
                        <div class="log-content" id="logContent">
                            <div class="log-entry">
                                <span class="log-time">12:00:00</span>
                                <span class="log-message">Система инициализирована. Ожидание входных
данных.</span>
```

```
                    </div>
                </div>
                <button id="clearLogBtn" class="btn btn--secondary btn--small">
                    <i class="fas fa-trash-alt"></i> Очистить журнал
                </button>
            </div>
        </div>
    </div>
</main>

<!-- Подвал -->
<footer class="footer">
    <div class="footer__info">
        <p class="footer__text">
            <i class="fas fa-flask"></i>
            Лаборатория физики плазмы. Токамак "Глобус-М2"
        </p>
        <p class="footer__text">
            Система автоматического детектирования пилообразных колебаний | Версия 1.0
        </p>
    </div>
    <div class="footer__links">
        <a href="#docs" class="footer__link"><i class="fas fa-book"></i> Документация</a>
        <a href="#contacts" class="footer__link"><i class="fas fa-envelope"></i> Контакты</a
>
        <a href="#privacy" class="footer__link"><i class="fas fa-shield-alt"></i> Конфиденци
альность</a>
    </div>
</footer>
</div>

<!-- Модальное окно загрузки файла -->
<div id="fileModal" class="modal">
    <div class="modal__content">
        <div class="modal__header">
            <h3 class="modal__title"><i class="fas fa-file-upload"></i> Загрузка данных</h3>
            <button class="modal__close">&times;</button>
        </div>
        <div class="modal__body">
            <div class="file-drop-area" id="fileDropArea">
                <i class="fas fa-cloud-upload-alt file-drop-icon"></i>
                <p class="file-drop-text">Перетащите файлы сюда или нажмите для выбора</p>
                <p class="file-drop-hint">Поддерживаемые форматы: .mat, .hdf5, .csv, .json</p>
                <input type="file" id="fileInput" accept=".mat,.hdf5,.csv,.json,.txt">
            </div>
            <div class="file-preview" id="filePreview">
                <!-- Будет заполнено после выбора файла -->
            </div>
        </div>
        <div class="modal__footer">
            <button id="confirmLoadBtn" class="btn btn--primary" disabled>Загрузить</button>
            <button id="cancelLoadBtn" class="btn btn--secondary">Отмена</button>
        </div>
    </div>
</div>

<script src="script.js"></script>
</body>
</html>
```

Файл 2: style.css

```
css

/* ===== RESET & BASE STYLES ===== */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

:root {
    /* Цветовая палитра: приглушенная научно-техническая */
    --color-primary: #2c3e50;
    --color-secondary: #34495e;
    --color-accent: #2980b9;
    --color-accent-light: #3498db;
    --color-success: #27ae60;
    --color-warning: #f39c12;
    --color-danger: #e74c3c;

    /* Нейтральные цвета */
    --color-bg: #f8f9fa;
    --color-bg-light: #ffffff;
    --color-border: #e0e0e0;
    --color-text: #333333;
    --color-text-light: #666666;
    --color-text-lighter: #999999;

    /* Тени */
    --shadow-light: 0 2px 8px rgba(0, 0, 0, 0.05);
    --shadow-medium: 0 4px 12px rgba(0, 0, 0, 0.08);
    --shadow-heavy: 0 8px 24px rgba(0, 0, 0, 0.12);

    /* Радиусы */
    --radius-small: 4px;
    --radius-medium: 8px;
```

```
    --radius-large: 12px;

    /* Отступы */
    --space-xs: 4px;
    --space-sm: 8px;
    --space-md: 16px;
    --space-lg: 24px;
    --space-xl: 32px;
    --space-xxl: 48px;

    /* Шрифты */
    --font-main: 'Roboto', 'Open Sans', Arial, sans-serif;
    --font-heading: 'Roboto', Arial, sans-serif;
}

body {
    font-family: var(--font-main);
    font-size: 14px;
    line-height: 1.5;
    color: var(--color-text);
    background-color: var(--color-bg);
    overflow-x: hidden;
}

.container {
    display: flex;
    flex-direction: column;
    min-height: 100vh;
    max-width: 100%;
}

/* ===== HEADER ===== */
.header {
    display: flex;
    align-items: center;
    justify-content: space-between;
    padding: var(--space-md) var(--space-xl);
    background-color: var(--color-bg-light);
    box-shadow: var(--shadow-light);
    position: sticky;
    top: 0;
    z-index: 100;
}

.header__logo {
    display: flex;
    align-items: center;
    gap: var(--space-sm);
}

.logo-icon {
    font-size: 24px;
    color: var(--color-accent);
}

.header__title {
    font-family: var(--font-heading);
    font-weight: 700;
    font-size: 20px;
    color: var(--color-primary);
}

.header__subtitle {
    font-weight: 400;
    font-size: 14px;
    color: var(--color-text-light);
    margin-left: var(--space-sm);
}

.header__nav {
    flex: 1;
    margin: 0 var(--space-xl);
}

.nav__list {
    display: flex;
    list-style: none;
    gap: var(--space-lg);
    justify-content: center;
}

.nav__link {
    display: flex;
    align-items: center;
    gap: var(--space-xs);
    text-decoration: none;
    color: var(--color-text);
    font-weight: 500;
    padding: var(--space-sm) var(--space-md);
    border-radius: var(--radius-small);
    transition: all 0.2s ease;
}

.nav__link:hover {
    background-color: var(--color-bg);
    color: var(--color-accent);
}
```

```css
.nav__link.active {
    background-color: var(--color-accent);
    color: white;
}

.header__user {
    display: flex;
    align-items: center;
    gap: var(--space-sm);
}

.user__name {
    font-weight: 500;
}

.user-icon {
    font-size: 24px;
    color: var(--color-accent);
}

/* ===== TOOLBAR ===== */
.toolbar {
    display: flex;
    padding: var(--space-md) var(--space-xl);
    background-color: var(--color-bg-light);
    border-bottom: 1px solid var(--color-border);
    gap: var(--space-xl);
    flex-wrap: wrap;
}

.toolbar__section {
    flex: 1;
    min-width: 300px;
}

.toolbar__title {
    font-size: 13px;
    font-weight: 600;
    color: var(--color-text-light);
    margin-bottom: var(--space-sm);
    display: flex;
    align-items: center;
    gap: var(--space-xs);
}

.toolbar__buttons {
    display: flex;
    flex-wrap: wrap;
    gap: var(--space-sm);
    align-items: center;
}

/* ===== BUTTONS & CONTROLS ===== */
.btn {
    padding: 8px 16px;
    border: none;
    border-radius: var(--radius-small);
    font-family: var(--font-main);
    font-size: 14px;
    font-weight: 500;
    cursor: pointer;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    gap: var(--space-xs);
    transition: all 0.2s ease;
}

.btn--primary {
    background-color: var(--color-accent);
    color: white;
}

.btn--primary:hover {
    background-color: var(--color-accent-light);
}

.btn--secondary {
    background-color: var(--color-bg);
    color: var(--color-text);
    border: 1px solid var(--color-border);
}

.btn--secondary:hover {
    background-color: #e9ecef;
}

.btn--accent {
    background-color: var(--color-success);
    color: white;
}

.btn--accent:hover {
    background-color: #2ecc71;
}

.btn--block {
    width: 100%;
```

```
}

.btn--small {
    padding: 4px 12px;
    font-size: 13px;
}

.btn-icon {
    width: 32px;
    height: 32px;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    background: none;
    border: 1px solid var(--color-border);
    border-radius: var(--radius-small);
    color: var(--color-text);
    cursor: pointer;
    transition: all 0.2s ease;
}

.btn-icon:hover {
    background-color: var(--color-bg);
    color: var(--color-accent);
}

.select {
    padding: 8px 12px;
    border: 1px solid var(--color-border);
    border-radius: var(--radius-small);
    background-color: white;
    font-family: var(--font-main);
    font-size: 14px;
    color: var(--color-text);
    min-width: 200px;
}

.select:focus {
    outline: none;
    border-color: var(--color-accent);
}

.checkbox {
    display: flex;
    align-items: center;
    gap: var(--space-xs);
    cursor: pointer;
    font-size: 14px;
}

.checkbox input {
    margin: 0;
}

.checkbox__label {
    user-select: none;
}

/* ===== MAIN CONTENT ===== */
.main-content {
    flex: 1;
    padding: var(--space-md) var(--space-xl);
}

.content-grid {
    display: flex;
    gap: var(--space-xl);
    height: calc(100vh - 280px);
    min-height: 600px;
}

.grid-column {
    display: flex;
    flex-direction: column;
    gap: var(--space-lg);
}

.grid-column--wide {
    flex: 3;
}

.grid-column--narrow {
    flex: 1;
    min-width: 320px;
}

/* ===== CHARTS ===== */
.chart-container {
    flex: 2;
    background-color: var(--color-bg-light);
    border-radius: var(--radius-medium);
    box-shadow: var(--shadow-light);
    display: flex;
    flex-direction: column;
    overflow: hidden;
}

.chart-header {
```

```
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: var(--space-md);
    border-bottom: 1px solid var(--color-border);
}

.chart-title {
    font-size: 16px;
    font-weight: 600;
    display: flex;
    align-items: center;
    gap: var(--space-sm);
}

.chart-area {
    flex: 1;
    padding: var(--space-md);
    display: flex;
    align-items: center;
    justify-content: center;
    background-color: white;
}

.chart-placeholder {
    text-align: center;
    color: var(--color-text-lighter);
}

.placeholder-icon {
    font-size: 48px;
    margin-bottom: var(--space-md);
    color: #e0e0e0;
}

.placeholder-text {
    font-size: 16px;
    margin-bottom: var(--space-xs);
}

.placeholder-hint {
    font-size: 13px;
}

.chart-footer {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: var(--space-sm) var(--space-md);
    border-top: 1px solid var(--color-border);
    background-color: var(--color-bg);
}

.chart-info {
    display: flex;
    gap: var(--space-lg);
    flex-wrap: wrap;
}

.info-item {
    display: flex;
    align-items: center;
    gap: var(--space-xs);
    font-size: 13px;
}

.chart-stats {
    font-weight: 500;
    font-size: 13px;
}

.secondary-charts {
    flex: 1;
    display: flex;
    gap: var(--space-lg);
}

.chart-small {
    flex: 1;
    background-color: var(--color-bg-light);
    border-radius: var(--radius-medium);
    box-shadow: var(--shadow-light);
    display: flex;
    flex-direction: column;
    overflow: hidden;
}

.chart-small-title {
    font-size: 14px;
    font-weight: 600;
    padding: var(--space-md);
    border-bottom: 1px solid var(--color-border);
}

.chart-small-area {
    flex: 1;
    padding: var(--space-md);
    display: flex;
```

```
    align-items: center;
    justify-content: center;
    color: var(--color-text-lighter);
    font-style: italic;
}

/* ===== CONTROL PANEL ===== */
.control-panel, .status-panel, .log-panel {
    background-color: var(--color-bg-light);
    border-radius: var(--radius-medium);
    box-shadow: var(--shadow-light);
    padding: var(--space-md);
    display: flex;
    flex-direction: column;
    gap: var(--space-md);
}

.control-panel__title, .status-panel__title, .log-panel__title {
    font-size: 15px;
    font-weight: 600;
    display: flex;
    align-items: center;
    gap: var(--space-sm);
    margin-bottom: var(--space-sm);
}

.parameter-group {
    display: flex;
    flex-direction: column;
    gap: var(--space-xs);
}

.parameter-label {
    font-size: 13px;
    color: var(--color-text);
}

.parameter-control {
    display: flex;
    align-items: center;
    gap: var(--space-md);
}

.slider {
    flex: 1;
    height: 4px;
    -webkit-appearance: none;
    background: var(--color-border);
    border-radius: 2px;
    outline: none;
}

.slider::-webkit-slider-thumb {
    -webkit-appearance: none;
    width: 16px;
    height: 16px;
    border-radius: 50%;
    background: var(--color-accent);
    cursor: pointer;
}

.slider::-moz-range-thumb {
    width: 16px;
    height: 16px;
    border-radius: 50%;
    background: var(--color-accent);
    cursor: pointer;
    border: none;
}

.parameter-value {
    font-size: 13px;
    font-weight: 500;
    min-width: 60px;
    text-align: right;
}

.parameter-actions {
    display: flex;
    flex-direction: column;
    gap: var(--space-sm);
    margin-top: var(--space-md);
}

/* ===== STATUS PANEL ===== */
.status-list {
    display: flex;
    flex-direction: column;
    gap: var(--space-md);
}

.status-item {
    display: flex;
    align-items: center;
    gap: var(--space-sm);
    padding: var(--space-sm);
    border-radius: var(--radius-small);
    background-color: var(--color-bg);
```

```
}

.status-item--active {
    border-left: 3px solid var(--color-success);
}

.status-icon {
    font-size: 18px;
    color: var(--color-text-light);
}

.status-item--active .status-icon {
    color: var(--color-success);
}

.status-content {
    display: flex;
    flex-direction: column;
    flex: 1;
}

.status-title {
    font-size: 13px;
    font-weight: 500;
}

.status-description {
    font-size: 12px;
    color: var(--color-text-light);
}

.processing-controls {
    display: flex;
    flex-direction: column;
    gap: var(--space-sm);
    margin-top: var(--space-md);
}

/* ===== LOG PANEL ===== */
.log-panel {
    flex: 1;
}

.log-content {
    flex: 1;
    background-color: var(--color-bg);
    border-radius: var(--radius-small);
    padding: var(--space-md);
    font-family: monospace;
    font-size: 12px;
    line-height: 1.6;
    max-height: 200px;
    overflow-y: auto;
}

.log-entry {
    margin-bottom: var(--space-xs);
    display: flex;
    gap: var(--space-sm);
}

.log-time {
    color: var(--color-text-light);
    min-width: 60px;
}

.log-message {
    color: var(--color-text);
    flex: 1;
}

/* ===== FOOTER ===== */
.footer {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: var(--space-md) var(--space-xl);
    background-color: var(--color-primary);
    color: white;
}

.footer__info {
    display: flex;
    flex-direction: column;
    gap: var(--space-xs);
}

.footer__text {
    font-size: 13px;
    display: flex;
    align-items: center;
    gap: var(--space-xs);
}

.footer__links {
    display: flex;
    gap: var(--space-lg);
}
```

```css
.footer__link {
    color: rgba(255, 255, 255, 0.8);
    text-decoration: none;
    font-size: 13px;
    display: flex;
    align-items: center;
    gap: var(--space-xs);
    transition: color 0.2s ease;
}

.footer__link:hover {
    color: white;
}

/* ===== MODAL ===== */
.modal {
    display: none;
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.5);
    z-index: 1000;
    align-items: center;
    justify-content: center;
}

.modal__content {
    background-color: white;
    border-radius: var(--radius-medium);
    width: 90%;
    max-width: 500px;
    box-shadow: var(--shadow-heavy);
    overflow: hidden;
}

.modal__header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: var(--space-md);
    background-color: var(--color-bg);
    border-bottom: 1px solid var(--color-border);
}

.modal__title {
    font-size: 16px;
    font-weight: 600;
    display: flex;
    align-items: center;
    gap: var(--space-sm);
}

.modal__close {
    background: none;
    border: none;
    font-size: 24px;
    cursor: pointer;
    color: var(--color-text-light);
    line-height: 1;
}

.modal__body {
    padding: var(--space-lg);
}

.modal__footer {
    padding: var(--space-md);
    border-top: 1px solid var(--color-border);
    display: flex;
    justify-content: flex-end;
    gap: var(--space-sm);
}


.file-drop-area {
    border: 2px dashed var(--color-border);
    border-radius: var(--radius-medium);
    padding: var(--space-xl);
    text-align: center;
    cursor: pointer;
    transition: all 0.2s ease;
}

.file-drop-area:hover {
    border-color: var(--color-accent);
    background-color: var(--color-bg);
}

.file-drop-icon {
    font-size: 48px;
    color: var(--color-text-lighter);
    margin-bottom: var(--space-md);
}

.file-drop-text {
    font-size: 16px;
```

```
    margin-bottom: var(--space-xs);
    font-weight: 500;
}

.file-drop-hint {
    font-size: 13px;
    color: var(--color-text-light);
}

#fileInput {
    display: none;
}

.file-preview {
    margin-top: var(--space-lg);
    padding: var(--space-md);
    background-color: var(--color-bg);
    border-radius: var(--radius-small);
    display: none;
}

/* ===== RESPONSIVE ===== */
@media (max-width: 1200px) {
    .content-grid {
        flex-direction: column;
        height: auto;
    }

    .grid-column--narrow {
        flex-direction: row;
        flex-wrap: wrap;
    }

    .control-panel, .status-panel, .log-panel {
        flex: 1;
        min-width: 300px;
    }
}

@media (max-width: 768px) {
    .header {
        flex-direction: column;
        gap: var(--space-md);
        padding: var(--space-md);
    }

    .header__nav {
        margin: 0;
        width: 100%;
    }

    .nav__list {
        flex-wrap: wrap;
        gap: var(--space-sm);
    }

    .toolbar {
        flex-direction: column;
        gap: var(--space-lg);
        padding: var(--space-md);
    }

    .toolbar__section {
        min-width: 100%;
    }

    .main-content {
        padding: var(--space-md);
    }

    .footer {
        flex-direction: column;
        gap: var(--space-md);
        text-align: center;
    }
}
```

Файл 3: script.js

```javascript

/**
 * SAWTOOTH DETECTOR - Основной скрипт управления интерфейсом
 * Автоматическое детектирование пилообразных колебаний на токамаке "Глобус-М2"
 * Версия 1.0
 */

// Глобальные переменные состояния приложения
const AppState = {
    isConnected: false,
    isProcessing: false,
    currentDataSource: null,
    currentAlgorithm: 'hybrid',
    detectionCount: 0,
    lastProcessTime: null,
    fileData: null
};
```

```
// DOM элементы
const elements = {
    // Кнопки загрузки данных
    loadFileBtn: document.getElementById('loadFileBtn'),
    connectADCBtn: document.getElementById('connectADCBtn'),
    liveModeBtn: document.getElementById('liveModeBtn'),

    // Элементы управления алгоритмами
    modelSelect: document.getElementById('modelSelect'),
    applyModelBtn: document.getElementById('applyModelBtn'),

    // Элементы визуализации
    showRawSignal: document.getElementById('showRawSignal'),
    showProcessed: document.getElementById('showProcessed'),
    showDetections: document.getElementById('showDetections'),
    exportBtn: document.getElementById('exportBtn'),

    // Элементы управления графиками
    zoomInBtn: document.getElementById('zoomInBtn'),
    zoomOutBtn: document.getElementById('zoomOutBtn'),
    resetViewBtn: document.getElementById('resetViewBtn'),

    // Элементы параметров обработки
    kernelWidth: document.getElementById('kernelWidth'),
    kernelWidthValue: document.getElementById('kernelWidthValue'),
    threshold: document.getElementById('threshold'),
    thresholdValue: document.getElementById('thresholdValue'),
    minDuration: document.getElementById('minDuration'),
    minDurationValue: document.getElementById('minDurationValue'),
    maxDuration: document.getElementById('maxDuration'),
    maxDurationValue: document.getElementById('maxDurationValue'),
    applyParamsBtn: document.getElementById('applyParamsBtn'),
    defaultParamsBtn: document.getElementById('defaultParamsBtn'),

    // Элементы управления обработкой
    startProcessingBtn: document.getElementById('startProcessingBtn'),
    stopProcessingBtn: document.getElementById('stopProcessingBtn'),

    // Элементы статуса
    currentAlgorithmEl: document.getElementById('currentAlgorithm'),
    dataSourceEl: document.getElementById('dataSource'),
    lastProcessTimeEl: document.getElementById('lastProcessTime'),
    detectionCountEl: document.getElementById('detectionCount'),

    // Элементы журнала
    logContent: document.getElementById('logContent'),
    clearLogBtn: document.getElementById('clearLogBtn'),

    // Модальное окно
    fileModal: document.getElementById('fileModal'),
    fileDropArea: document.getElementById('fileDropArea'),
    fileInput: document.getElementById('fileInput'),
    filePreview: document.getElementById('filePreview'),
    confirmLoadBtn: document.getElementById('confirmLoadBtn'),
    cancelLoadBtn: document.getElementById('cancelLoadBtn'),
    modalClose: document.querySelector('.modal__close')
};

/**
 * Инициализация приложения
 */
function initApp() {
    logMessage('Система инициализирована. Ожидание входных данных.', 'system');

    // Настройка обработчиков событий для кнопок загрузки данных
    elements.loadFileBtn.addEventListener('click', openFileModal);
    elements.connectADCBtn.addEventListener('click', connectToADC);
    elements.liveModeBtn.addEventListener('click', enableLiveMode);

    // Настройка обработчиков для выбора алгоритма
    elements.modelSelect.addEventListener('change', updateSelectedAlgorithm);
    elements.applyModelBtn.addEventListener('click', applyAlgorithm);

    // Настройка обработчиков для визуализации
    elements.showRawSignal.addEventListener('change', updateVisualization);
    elements.showProcessed.addEventListener('change', updateVisualization);
    elements.showDetections.addEventListener('change', updateVisualization);
    elements.exportBtn.addEventListener('click', exportData);

    // Настройка обработчиков для управления графиками
    elements.zoomInBtn.addEventListener('click', zoomIn);
    elements.zoomOutBtn.addEventListener('click', zoomOut);
    elements.resetViewBtn.addEventListener('click', resetView);

    // Настройка обработчиков для параметров обработки
    elements.kernelWidth.addEventListener('input', updateKernelWidthValue);
    elements.threshold.addEventListener('input', updateThresholdValue);
    elements.minDuration.addEventListener('input', updateMinDurationValue);
    elements.maxDuration.addEventListener('input', updateMaxDurationValue);
    elements.applyParamsBtn.addEventListener('click', applyParameters);
    elements.defaultParamsBtn.addEventListener('click', resetParameters);

    // Настройка обработчиков для управления обработкой
    elements.startProcessingBtn.addEventListener('click', startProcessing);
    elements.stopProcessingBtn.addEventListener('click', stopProcessing);

    // Настройка обработчиков для журнала
```

```
elements.clearLogBtn.addEventListener('click', clearLog);

// Настройка обработчиков для модального окна
elements.fileDropArea.addEventListener('click', () => elements.fileInput.click());
elements.fileDropArea.addEventListener('dragover', handleDragOver);
elements.fileDropArea.addEventListener('drop', handleFileDrop);
elements.fileInput.addEventListener('change', handleFileSelect);
elements.confirmLoadBtn.addEventListener('click', confirmFileLoad);
elements.cancelLoadBtn.addEventListener('click', closeFileModal);
elements.modalClose.addEventListener('click', closeFileModal);

// Закрытие модального окна при клике вне его
elements.fileModal.addEventListener('click', (e) => {
    if (e.target === elements.fileModal) {
        closeFileModal();
    }
});

// Инициализация значений параметров
updateKernelWidthValue();
updateThresholdValue();
updateMinDurationValue();
updateMaxDurationValue();

    // Обновление статуса системы
    updateSystemStatus();
}

/**
 * Добавление сообщения в журнал
 * @param {string} message - Текст сообщения
 * @param {string} type - Тип сообщения (system, success, warning, error)
 */
function logMessage(message, type = 'system') {
    const logEntry = document.createElement('div');
    logEntry.className = 'log-entry';

    const time = new Date().toLocaleTimeString('ru-RU', {
        hour12: false,
        hour: '2-digit',
        minute: '2-digit',
        second: '2-digit'
    });

    const typeIcon = {
        'system': '🔧',
        'success': '✅',
        'warning': '⚠️',
        'error': '❌'
    }[type] || '📝';

    logEntry.innerHTML = `
        <span class="log-time">${time}</span>
        <span class="log-message">${typeIcon} ${message}</span>
    `;

    elements.logContent.prepend(logEntry);

    // Ограничиваем количество записей в журнале
    const maxEntries = 50;
    const entries = elements.logContent.querySelectorAll('.log-entry');
    if (entries.length > maxEntries) {
        entries[entries.length - 1].remove();
    }
}

/**
 * Очистка журнала
 */
function clearLog() {
    elements.logContent.innerHTML = '';
    logMessage('Журнал очищен', 'system');
}

/**
 * Открытие модального окна для загрузки файла
 */
function openFileModal() {
    elements.fileModal.style.display = 'flex';
    logMessage('Открыто окно загрузки файла', 'system');
}

/**
 * Закрытие модального окна
 */
function closeFileModal() {
    elements.fileModal.style.display = 'none';
    elements.filePreview.style.display = 'none';
    elements.confirmLoadBtn.disabled = true;
    elements.fileInput.value = '';
}

/**
 * Обработка перетаскивания файла
 * @param {Event} e - Событие перетаскивания
 */
function handleDragOver(e) {
    e.preventDefault();
```

```
    e.stopPropagation();
    elements.fileDropArea.style.borderColor = '#2980b9';
    elements.fileDropArea.style.backgroundColor = '#f8f9fa';
}

/**
 * Обработка сброса файла
 * @param {Event} e - Событие сброса
 */
function handleFileDrop(e) {
    e.preventDefault();
    e.stopPropagation();
    elements.fileDropArea.style.borderColor = '#e0e0e0';
    elements.fileDropArea.style.backgroundColor = 'transparent';

    const files = e.dataTransfer.files;
    if (files.length > 0) {
        handleFile(files[0]);
    }
}

/**
 * Обработка выбора файла через диалог
 */
function handleFileSelect() {
    const file = elements.fileInput.files[0];
    if (file) {
        handleFile(file);
    }
}

/**
 * Обработка выбранного файла
 * @param {File} file - Выбранный файл
 */
function handleFile(file) {
    // В реальном приложении здесь будет проверка формата файла
    const fileInfo = {
        name: file.name,
        size: (file.size / 1024).toFixed(2) + ' KB',
        type: file.type || 'Неизвестный формат',
        lastModified: new Date(file.lastModified).toLocaleString('ru-RU')
    };

    elements.filePreview.innerHTML = `
        <h4>Информация о файле:</h4>
        <p><strong>Имя:</strong> ${fileInfo.name}</p>
        <p><strong>Размер:</strong> ${fileInfo.size}</p>
        <p><strong>Тип:</strong> ${fileInfo.type}</p>
        <p><strong>Изменен:</strong> ${fileInfo.lastModified}</p>
    `;

    elements.filePreview.style.display = 'block';
    elements.confirmLoadBtn.disabled = false;

    logMessage(`Выбран файл: ${fileInfo.name}`, 'system');
}

/**
 * Подтверждение загрузки файла
 */
function confirmFileLoad() {
    // В реальном приложении здесь будет загрузка и обработка файла
    const fileName = elements.fileInput.files[0]?.name || 'demo_data.mat';

    AppState.currentDataSource = 'file';
    AppState.fileData = {
        name: fileName,
        timestamp: new Date().toISOString()
    };

    // Генерация демо-данных для визуализации
    generateDemoData();

    // Обновление интерфейса
    updateSystemStatus();
    elements.startProcessingBtn.disabled = false;

    logMessage(`Файл "${fileName}" успешно загружен`, 'success');
    closeFileModal();

    // Обновление графика
    updateChartWithDemoData();
}

/**
 * Подключение к АЦП
 */
function connectToADC() {
    // В реальном приложении здесь будет подключение к аппаратному интерфейсу
    AppState.isConnected = true;
    AppState.currentDataSource = 'adc';

    logMessage('Подключение к АЦП...', 'system');

    // Имитация подключения
    setTimeout(() => {
        AppState.isConnected = true;
```

```
        updateSystemStatus();
        elements.startProcessingBtn.disabled = false;
        logMessage('Успешно подключено к АЦП. Готово к приему данных.', 'success');
    }, 1000);
}

/**
 * Включение режима эксперимента (live mode)
 */
function enableLiveMode() {
    // В реальном приложении здесь будет настройка режима реального времени
    AppState.currentDataSource = 'live';

    logMessage('Включен режим эксперимента. Ожидание данных в реальном времени...', 'system');

    // Обновление интерфейса
    updateSystemStatus();
    elements.startProcessingBtn.disabled = false;

    // Генерация демо-данных для live режима
    simulateLiveData();
}

/**
 * Обновление выбранного алгоритма
 */
function updateSelectedAlgorithm() {
    const algorithm = elements.modelSelect.value;
    const algorithmNames = {
        'hybrid': 'Гибридный алгоритм',
        'gaussian': 'Детектирование по Гауссу',
        'statistical': 'Статистический анализ (gSPRT)',
        'ml-cnn': 'Модель CNN',
        'ml-lstm': 'Модель LSTM'
    };

    AppState.currentAlgorithm = algorithm;
    logMessage(`Выбран алгоритм: ${algorithmNames[algorithm]}`, 'system');
}

/**
 * Применение выбранного алгоритма
 */
function applyAlgorithm() {
    const algorithm = elements.modelSelect.value;
    const algorithmNames = {
        'hybrid': 'Гибридный алгоритм',
        'gaussian': 'Детектирование по Гауссу',
        'statistical': 'Статистический анализ (gSPRT)',
        'ml-cnn': 'Модель CNN',
        'ml-lstm': 'Модель LSTM'
    };

    logMessage(`Применен алгоритм: ${algorithmNames[algorithm]}`, 'success');

    // Обновление статуса
    elements.currentAlgorithmEl.textContent = algorithmNames[algorithm];

    // Если данные уже загружены, запускаем обработку
    if (AppState.currentDataSource) {
        startProcessing();
    }
}

/**
 * Обновление визуализации
 */
function updateVisualization() {
    const settings = {
        raw: elements.showRawSignal.checked,
        processed: elements.showProcessed.checked,
        detections: elements.showDetections.checked
    };

    logMessage('Настройки визуализации обновлены', 'system');
    // В реальном приложении здесь будет обновление графика
}

/**
 * Экспорт данных
 */
function exportData() {
    // В реальном приложении здесь будет экспорт данных в файл
    logMessage('Экспорт данных начат...', 'system');

    // Имитация экспорта
    setTimeout(() => {
        logMessage('Данные успешно экспортированы в файл "detection_results.csv"', 'success');

        // Создание и скачивание демо-файла
        const demoData = "Time(ms),Channel1,Channel2,Channel3,Detection\n0,0.1,0.2,0.3,0\n1,0.2,0.3,
0.4,0\n2,0.8,0.9,1.0,1\n";
        const blob = new Blob([demoData], { type: 'text/csv' });
        const url = URL.createObjectURL(blob);
        const a = document.createElement('a');
        a.href = url;
        a.download = 'detection_results.csv';
        document.body.appendChild(a);
```

```
        a.click();
        document.body.removeChild(a);
        URL.revokeObjectURL(url);
    }, 500);
}

/**
 * Увеличение масштаба графика
 */
function zoomIn() {
    logMessage('Увеличение масштаба графика', 'system');
    // В реальном приложении здесь будет управление масштабом графика
}

/**
 * Уменьшение масштаба графика
 */
function zoomOut() {
    logMessage('Уменьшение масштаба графика', 'system');
    // В реальном приложении здесь будет управление масштабом графика
}

/**
 * Сброс вида графика
 */
function resetView() {
    logMessage('Сброс вида графика', 'system');
    // В реальном приложении здесь будет сброс графика
}

/**
 * Обновление значения ширины ядра Гаусса
 */
function updateKernelWidthValue() {
    elements.kernelWidthValue.textContent = `${parseFloat(elements.kernelWidth.value).toFixed(2)} мс
`;
}

/**
 * Обновление значения порога детектирования
 */
function updateThresholdValue() {
    elements.thresholdValue.textContent = `${parseFloat(elements.threshold.value).toFixed(1)} σ`;
}

/**
 * Обновление значения минимальной длительности краша
 */
function updateMinDurationValue() {
    elements.minDurationValue.textContent = `${parseFloat(elements.minDuration.value).toFixed(2)} мс
`;
}

/**
 * Обновление значения максимальной длительности краша
 */
function updateMaxDurationValue() {
    elements.maxDurationValue.textContent = `${parseFloat(elements.maxDuration.value).toFixed(2)} мс
`;
}

/**
 * Применение параметров обработки
 */
function applyParameters() {
    const params = {
        kernelWidth: parseFloat(elements.kernelWidth.value),
        threshold: parseFloat(elements.threshold.value),
        minDuration: parseFloat(elements.minDuration.value),
        maxDuration: parseFloat(elements.maxDuration.value)
    };

    logMessage(`Параметры применены: σ=${params.kernelWidth}мс, порог=${params.threshold}σ`, 'succes
s');

    // В реальном приложении здесь будет применение параметров к алгоритму
}

/**
 * Сброс параметров к значениям по умолчанию
 */
function resetParameters() {
    elements.kernelWidth.value = 0.1;
    elements.threshold.value = 3.5;
    elements.minDuration.value = 0.1;
    elements.maxDuration.value = 2.0;

    updateKernelWidthValue();
    updateThresholdValue();
    updateMinDurationValue();
    updateMaxDurationValue();

    logMessage('Параметры сброшены к значениям по умолчанию', 'system');
}

/**
 * Начало обработки данных
 */
```

```
function startProcessing() {
    if (!AppState.currentDataSource) {
        logMessage('Ошибка: нет источника данных', 'error');
        return;
    }

    AppState.isProcessing = true;
    AppState.lastProcessTime = new Date();

    // Обновление интерфейса
    updateSystemStatus();
    elements.startProcessingBtn.disabled = true;
    elements.stopProcessingBtn.disabled = false;

    logMessage('Начало обработки данных...', 'system');

    // Имитация обработки
    simulateProcessing();
}

/**
 * Остановка обработки данных
 */
function stopProcessing() {
    AppState.isProcessing = false;

    // Обновление интерфейса
    updateSystemStatus();
    elements.startProcessingBtn.disabled = false;
    elements.stopProcessingBtn.disabled = true;

    logMessage('Обработка данных остановлена', 'system');
}

/**
 * Обновление статуса системы
 */
function updateSystemStatus() {
    // Обновление отображения источника данных
    const dataSourceNames = {
        'file': 'Файл',
        'adc': 'АЦП',
        'live': 'Режим эксперимента'
    };

    elements.dataSourceEl.textContent = AppState.currentDataSource
        ? dataSourceNames[AppState.currentDataSource]
        : 'Не подключен';

    // Обновление времени последней обработки
    if (AppState.lastProcessTime) {
        elements.lastProcessTimeEl.textContent = AppState.lastProcessTime.toLocaleTimeString('ru-R
U');
    }

    // Обновление количества детектированных событий
    elements.detectionCountEl.textContent = `Детектировано: ${AppState.detectionCount} событий`;

    // Обновление состояния кнопок
    if (AppState.isProcessing) {
        elements.startProcessingBtn.disabled = true;
        elements.stopProcessingBtn.disabled = false;
    } else {
        elements.startProcessingBtn.disabled = !AppState.currentDataSource;
        elements.stopProcessingBtn.disabled = true;
    }
}

/**
 * Генерация демонстрационных данных
 */
function generateDemoData() {
    // В реальном приложении здесь будет загрузка реальных данных
    logMessage('Генерация демонстрационных данных...', 'system');

    // Имитация обнаруженных событий
    AppState.detectionCount = Math.floor(Math.random() * 10) + 5;
}

/**
 * Обновление графика демонстрационными данными
 */
function updateChartWithDemoData() {
    const chartArea = document.getElementById('mainChart');

    // Очистка placeholder
    chartArea.innerHTML = '';

    // Создание канваса для графика
    const canvas = document.createElement('canvas');
    canvas.id = 'signalCanvas';
    canvas.width = chartArea.clientWidth - 40;
    canvas.height = chartArea.clientHeight - 40;
    canvas.style.width = '100%';
    canvas.style.height = '100%';

    chartArea.appendChild(canvas);
```

```
    // В реальном приложении здесь будет отрисовка реальных данных
    // Сейчас просто отобразим сообщение
    const ctx = canvas.getContext('2d');
    ctx.fillStyle = '#f8f9fa';
    ctx.fillRect(0, 0, canvas.width, canvas.height);

    ctx.fillStyle = '#333';
    ctx.font = '16px Arial';
    ctx.textAlign = 'center';
    ctx.fillText('График SXR-сигналов', canvas.width / 2, canvas.height / 2 - 20);

    ctx.font = '14px Arial';
    ctx.fillText(`Загружено ${AppState.detectionCount} пилообразных событий`, canvas.width / 2, canv
as.height / 2 + 10);

    logMessage('График обновлен демонстрационными данными', 'success');
}

/**
 * Имитация обработки данных
 */
function simulateProcessing() {
    let progress = 0;
    const interval = setInterval(() => {
        progress += 10;

        if (progress >= 100) {
            clearInterval(interval);

            // Завершение обработки
            AppState.isProcessing = false;
            AppState.detectionCount = Math.floor(Math.random() * 20) + 10;

            // Обновление интерфейса
            updateSystemStatus();
            elements.startProcessingBtn.disabled = false;
            elements.stopProcessingBtn.disabled = true;

            logMessage(`Обработка завершена. Обнаружено ${AppState.detectionCount} событий`, 'succes
s');

            // Обновление графика
            updateChartWithDemoData();
        }
    }, 100);
}

/**
 * Имитация live данных
 */
function simulateLiveData() {
    logMessage('Начало приема данных в реальном времени...', 'system');

    // Имитация периодического обнаружения событий
    let eventCount = 0;
    const interval = setInterval(() => {
        if (!AppState.isProcessing && AppState.currentDataSource === 'live') {
            eventCount++;
            AppState.detectionCount = eventCount;
            updateSystemStatus();

            if (eventCount % 5 === 0) {
                logMessage(`Обнаружено событие #${eventCount} в реальном времени`, 'success');
            }
        }

        // Остановка через 30 секунд для демонстрации
        if (eventCount >= 30) {
            clearInterval(interval);
            logMessage('Завершение приема live данных (демо-режим)', 'system');
        }
    }, 1000);
}

// Инициализация приложения после загрузки DOM
document.addEventListener('DOMContentLoaded', initApp);

// Добавление класса для кнопок навигации при прокрутке
window.addEventListener('scroll', () => {
    const header = document.querySelector('.header');
    if (window.scrollY > 10) {
        header.style.boxShadow = '0 4px 12px rgba(0, 0, 0, 0.1)';
    } else {
        header.style.boxShadow = '0 2px 8px rgba(0, 0, 0, 0.05)';
    }
});
```

Инструкция по использованию:

1. Сохраните все три файла в одной папке:
   - `index.html`
   - `style.css`
   - `script.js`
2. Откройте файл `index.html` в любом современном браузере (Chrome, Firefox, Safari, Edge).