

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ им. В.Г. ШУХОВА»**
(БГТУ им. В.Г. Шухова)

Институт магистратуры

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Специальность 09.04.01 Информатика и вычислительная техника

Специализация Проектирование интеллектуальных систем

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему:

Разработка и реализация алгоритма вычисления уровня топлива в баках сложной
конфигурации

Студент	Притчин Иван Сергеевич
Зав. кафедрой	канд. техн. наук, проф. Поляков В.М.
Руководитель	к.ф.-м.н., Зуев С.В.
Консультант	

К защите допустить

Зав. кафедрой _____ /Поляков В.М./
« _____ » _____ 2021 г.

Белгород 2021г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ им. В.Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)

Институт магистратуры

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Направление подготовки 09.04.01 Информатика и вычислительная техника

Направленность (профиль, специализация) образовательной программы: Проектирование интеллектуальных систем.

Утверждаю:

Зав. кафедрой _____

« _____ » _____ 20 ____ г.

ЗАДАНИЕ

на выпускную квалификационную работу студента

Притчина Ивана Сергеевича

(Фамилия Имя Отчество)

1. Вид выпускной квалификационной работы (ВКР) магистерская диссертация
 (бакалаврская работа, дипломный проект,
 дипломная работа, магистерская диссертация)

2. Тема ВКР: Разработка и реализация алгоритма вычисления уровня топлива в баках сложной конфигурации

утверждено приказом по университету от « _____ » 2021 г. № _____

3. Срок сдачи студентом законченной ВКР

4. Исходные данные: данные компании «Экспертком», методы обработки данных

5. Содержание ВКР: (перечень подлежащих разработке разделов) 1. Анализ предметной области. 2. Описание подхода для вычисления количества топлива в баках сложной конфигурации по нескольким датчикам уровня топлива. 3. Реализация алгоритма разбиения датчиков уровня топлива и построения зависимостей. 4. Написание пояснительной записки к ВКР.

6. Перечень графического материала:

56 рисунков;

12 таблиц;

22 слайдов презентации (титульный лист, актуальность, монтаж и тарифовка ДУТ, процесс вычислений показаний с ДУТ, цель и задачи, решение, этап 1: выделение зон, этап 2: получение виртуальных датчиков, этап 3: получение тарифовок ДУТ, этап 4: подавление экстраполяции, этап 5: получение формулы, используемые инструменты, результаты работы программы, структура проекта, тестирование, заключение)

Консультанты по работе с указанием относящихся к ним разделов

Раздел	Консультант	Задание выдал (подпись, дата)	Задание принял (подпись, дата)

Дата выдачи задания «_____» _____ 2021 г.

(подпись руководителя)

Задание принял к исполнению

(подпись студента)

КАЛЕНДАРНЫЙ ПЛАН

№ п/п	Наименование этапов работы	Срок выполнения этапов работы	Примечание
1	Изучение предметной области, сопутствующих проблем при определении уровня топлива в баках сложной конфигурации		Выполнено
2	Постановка задачи и формирование требований к разрабатываемым программным средствам.		Выполнено
3	Проектирование консольного приложения.		Выполнено
4	Разработка программной системы; её тестирование и отладка; написание пояснительной записки к ВКР		Выполнено
5	Подготовка презентационного материала и защитной речи		Выполнено

Студент

(подпись)

Притчин И.С.

(Ф.И.О.)

Руководитель

(подпись)

к.ф.-м.н.,

Зуев С.В.

(Ф.И.О.)

Содержание

Введение.....	7
1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ.....	9
Спутниковый мониторинг транспорта (СМТ)	9
Контроллеры и трекеры.....	10
Датчики уровня топлива.....	13
Программное обеспечение	15
Проблема вычисления уровня топлива.....	19
Цель и задачи выпускной квалификационной работы.....	23
2. ОПИСАНИЕ ПОДХОДА ДЛЯ ВЫЧИСЛЕНИЯ КОЛИЧЕСТВА ТОПЛИВА В БАКАХ СЛОЖНОЙ КОНФИГУРАЦИИ ПО НЕСКОЛЬКИМ ДАТЧИКАМ УРОВНЯ ТОПЛИВА	24
Баки – не сообщающиеся сосуды	24
Баки – сообщающиеся сосуды	30
Требования к программному обеспечению	35
Выбор инструментов для разработки.....	38
Выводы	39
3. РЕАЛИЗАЦИЯ АЛГОРИТМА ДЛЯ РАЗБИЕНИЯ ДАТЧИКОВ УРОВНЯ ТОПЛИВА И ПОСТРОЕНИЯ ЗАВИСИМОСТЕЙ.....	40
Описание алгоритма	40
Тестирование программного обеспечения	53

	6
Установка программного обеспечения	58
Вывод.....	70
ЗАКЛЮЧЕНИЕ	71
СПИСОК ЛИТЕРАТУРЫ.....	72

Введение

Заметный рост количества транспортных единиц, аппаратных и программных возможностей позволили лучше контролировать ресурсы организаций, использующих автотранспорт. Комплексы наблюдения за техникой состоят из трекера, подключаемого к нему оборудования, аппаратного обеспечения, программного обеспечения, которые участвуют в процессе обработки данных.

Появление отечественных и зарубежных систем спутникового мониторинга позволяет в довольно быстрые сроки осуществлять контроль в данном вопросе. Клиенты, использующие приложения, получают удобный интерфейс для решения своих задач, которого в большинстве случаев хватает, чтобы покрыть свои нужды.

Организации, осуществляющие контроль над количеством топлива в баках, столкнулись с необходимостью более точного определения данного показателя, который может иметь погрешность по ряду причин: наклон техники на местности [1], колебания при работе в тяжелых условиях. Логичным решением было добавление дополнительных датчиков уровня топлива в бак, что позволяло по получаемым данным вычислять среднее. Это привело к тому, что внешние факторы стали оказывать меньшее влияние на качество получаемой информации.

Однако проблема не была решена для баков сложной конфигурации. Сервер получает данные с некоторого множества датчиков и подставляет их в формулу, задаваемую пользователем системы спутникового мониторинга. Несмотря на ожидаемую, с нулевого приближения, простоту вычислений, наблюдаются определенные сложности в создании формулы для баков сложной конфигурации, по которым будет производиться расчёт.

В рамках магистерской диссертации рассматривается вопрос разработки и реализации алгоритма вычисления количества топлива в баках сложной

конфигурации с использованием нескольких датчиков уровня топлива, а также механизм адаптации данного подхода для систем спутникового мониторинга.

В первом разделе даны основные определения, которые касаются процессов установки, настройки и конфигурирования оборудования для контроля уровня топлива в транспортном средстве; описаны вычислительные проблемы при определении уровня топлива. В конце раздела поставлена цель выпускной квалификационной работы и сформулированы задачи.

Во втором разделе описан предлагаемый алгоритм (без детализации) для баков – не сообщающихся сосудов и баков – сообщающихся сосудов, а также сформулированы требования к программному обеспечению.

В третьем разделе представлена реализация алгоритма, приведены результаты тестирования, а также сведения по установке ПО, его эксплуатации, вкуче с системой спутникового мониторинга Wialon.

В заключении приведены выводы о проделанной работе.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Спутниковый мониторинг транспорта (СМТ)

К появлению систем спутникового мониторинга привела необходимость отслеживания единиц в реальном времени для получения координат. Со временем, потребности клиентов возрастали: возникла потребность получения и других данных. Трекер стал представлять собой более сложное устройство, с возможностью подключения множества датчиков, которые позволили получать данные более широкого спектра.

Перед современными системами спутникового мониторинга поставлен ряд задач, среди которых:

- Задача мониторинга (получение информации о транспортном средстве: координат, скорости движения, расходе топлива, подсчёте моточасов).
- Контроль над графиком движения (учет перемещения подвижных единиц, например, при доставке грузов).
- Сбор статистики и анализ (оценка результатов работы техники за промежутки времени в вопросе затраты ресурсов).
- Обеспечение безопасности.
- Идентификация персонала, работающего с транспортным средством.

Комплексы наблюдения за техникой состоят из следующих составляющих:

- Транспортное средство, оборудованное GPS (Global Positioning System) или ГЛОНАСС (Глобальная навигационная спутниковая система) контроллером или трекером, который получает данные со спутников и передаёт их на сервер при помощи GSM (Global System for Mobile Communications) или CDMA (Code Division Multiple Access). Можно встретить варианты, работающие за счёт спутниковой или УКВ связи в случае плохого GSM-покрытия.
- Множество датчиков, связываемых с трекером, для передачи дополнительных данных.

- Сервер с программным обеспечением для приёма, хранения, обработки и анализа получаемой информации.
- Компьютер диспетчера, производящий наблюдение за транспортом.

Рассмотрим составляющие более подробно.

Контроллеры и трекеры

Большинство видов оборудования данного класса имеют схожие функциональные возможности:

- Вычисление собственного местоположения и скорости движения на основании сигналов спутников GPS или ГЛОНАСС.
- Подключение датчиков через аналоговые или цифровые входы.
- Считывание данных с бортового оборудования, имеющего последовательный порт (интерфейс стандарта RS-232) или более специализированный интерфейс CAN (Controller Area Network). Контроль напряжения автомобильного и встроенного в трекер аккумуляторов.
- Хранение данных во внутренней энергонезависимой памяти устройства на случай отсутствия возможности их передачи на сервер.
- Передача данных на сервер согласно расписанию для дальнейшей обработки.

Рассмотрим принципы функционирования трекера на примере ГЛОНАСС-трекера SMART S-2433. Устройство осуществляет контроль:

- состояния подключенных датчиков
- напряжения основного источника питания и встроенного аккумулятора
- уровня сигнала GSM-модема
- работоспособности и показаний навигационного датчика (ГЛОНАСС/GPS) и т.д.

Устройство трекера представлено на рисунках 1.1-1.2 и представляет собой [2]:

1. Передняя крышка корпуса
2. Крепежное отверстие
3. Системный светодиодный индикатор
4. Светодиодный индикатор GSM
5. Светодиодный индикатор ГЛОНАСС/GPS
6. 14-контактный разъем
7. Разъем MiniUSB
8. Выталкиватель держателя SIM-карты
9. Держатель SIM-карты 1 (внешний)
10. Задняя крышка корпуса;
11. Крепежный винт задней крышки корпуса – 4 шт.

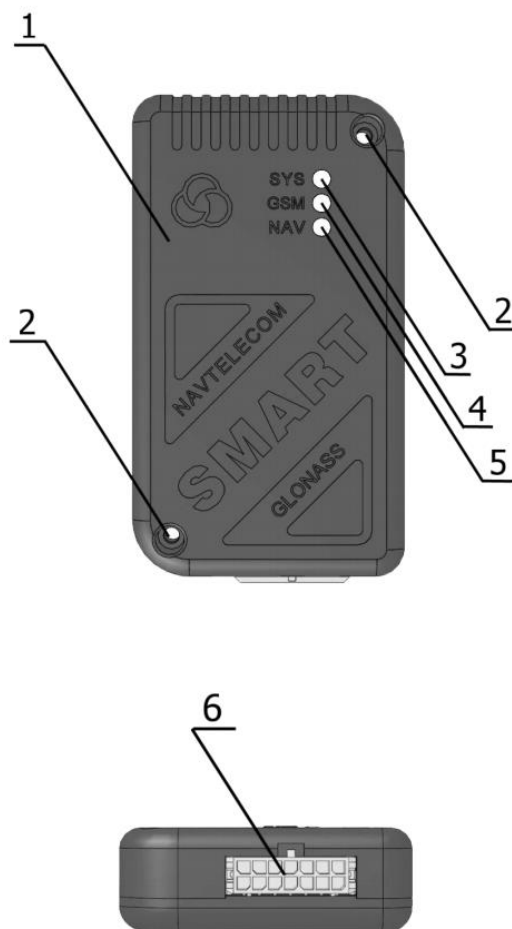


Рисунок 1.1 — ГЛОНАСС-трекер SMART S-2433

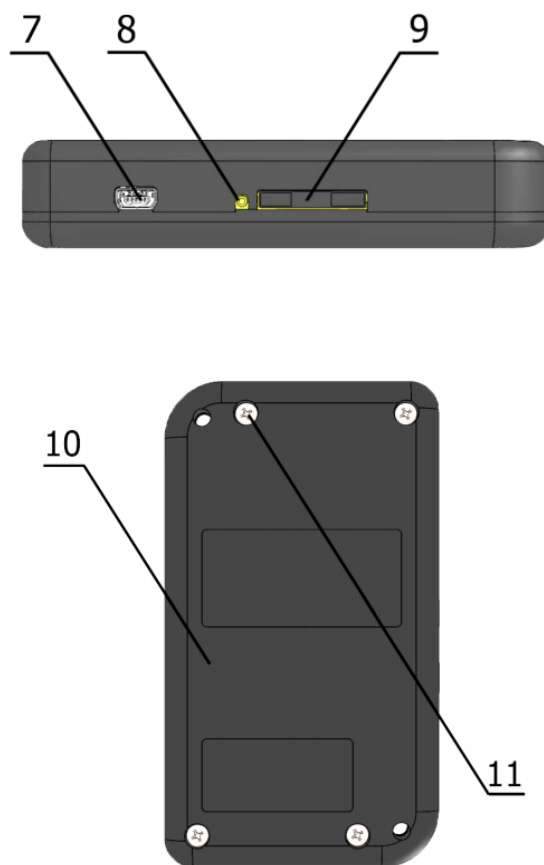


Рисунок 1.2 — ГЛОНАСС-трекер SMART S-2433

При подключении к датчику через компьютер посредством USB или за счёт основного питания происходит включение устройства. При отключении питания устройство работает от встроенной аккумуляторной батареи. При разряде встроенной АКБ до 3В происходит выключение устройства.

В процессе работы трекер формирует и отправляет сообщения, которые создаются при возникновении события, некоторые из них:

- Изменение курса
- Срабатывание таймера в движении или при стоянке
- Срабатывание датчика входной линии
- Изменение значения аналогового или цифрового датчика

Сообщения записываются в память и отправляются на сервер последовательно вместе с порядковым номером и кодом, определяющим причину формирования. Исключением в порядке отправки будут сообщения,

сформированные по «тревожным» событиям (например, срабатывание датчиков удара).

В зависимости от настроек прибора, происходит сохранение информации в энергонезависимую память. При невозможности отправить сообщения, данные будут выгружены по устранению проблемы.

При передаче информации трекер ждёт от сервера ответ о подтверждении получения. Если трекер не получил информацию об успешной операции он пробует отправить данный пакет снова.

Приведенный алгоритм, который заложен в протоколе передачи данных, позволяет обеспечить надежность отправки сообщений на сервер. Стоит так же отметить, что производители оборудования выпускают конфигураторы, которые дают возможность, относительно просто, настроить работу трекера под свои задачи.

Датчики уровня топлива

Ранее отмечалось, что операторы систем спутникового мониторинга хотят получать информацию не только о местоположении объекта, но и иные данные, которые могут быть получены при помощи внешних датчиков. Наиболее часто можно встретить датчики уровня топлива (ДУТ). ДУТы могут монтироваться как на стационарные (резервуары с горючим, автозаправочные станции), так и на подвижные объекты. Интеграция данного оборудования вкупе с программным обеспечением позволяет в режиме реального времени получать информацию о:

- фактическом объеме топлива в баке на текущий момент;
- расходе топлива за определенный промежуток времени;
- усредненном расходе топлива (на 100 км);
- наличии фактов сливов и заливок и их объемы.

По конструктивным особенностям выделяют следующие виды:

- потенциометрические (поплавковые);
- ёмкостные;
- ультразвуковые.

Одним из часто используемых датчиков является Omnicomm LLS5 [3] (Рисунок 1.3).



Рисунок 1.3 — Датчик Omnicomm LLS5

Датчик монтируется максимально приближенно к геометрическому центру бака (Рисунок 1.4) и подключается к трекеру:

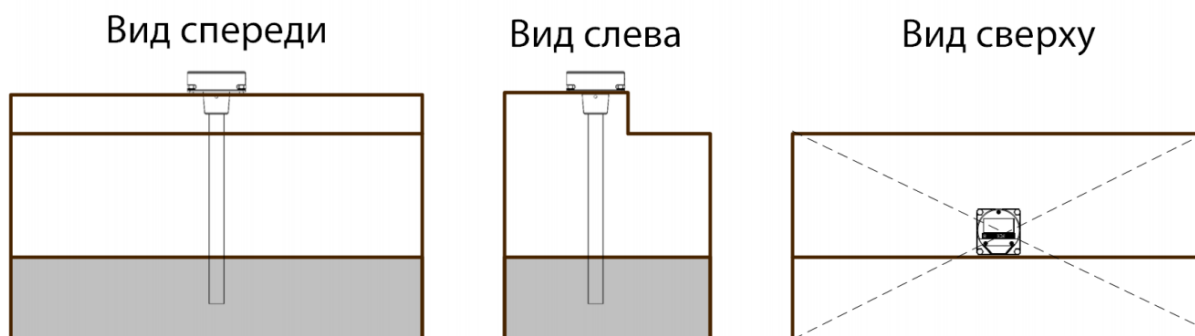


Рисунок 1.4 — Монтаж одного ДУТ

Установка нескольких датчиков уровня топлива позволяет значительно уменьшить зависимость уровня топлива от угла наклона транспортного средства. Например, в случае монтажа двух ДУТ они могут быть размещены следующим образом (Рисунок 1.5):

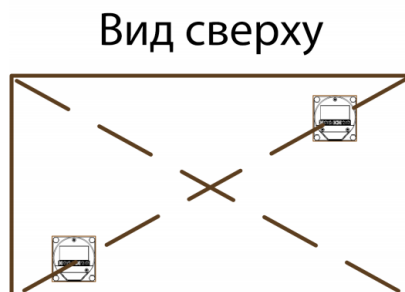


Рисунок 1.5 — Установка двух ДУТов в бак

Приём с использованием нескольких датчиков также часто используется при работе с баками сложной конфигурации. Подробнее об этом пойдёт речь во второй части работы.

Программное обеспечение

Трекеры отсылают данные на сервер, где они обрабатываются программным обеспечением. Информация предоставляется пользователю в удобном виде для решения практических задач (Рисунок 1.6):

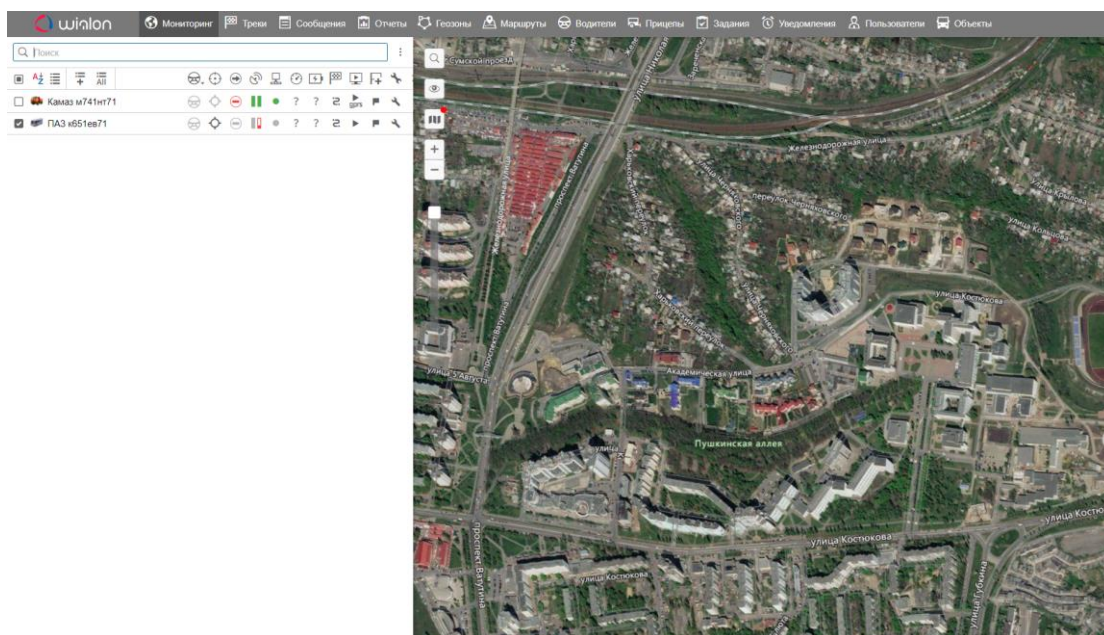


Рисунок 1.6 — Система спутникового мониторинга Wialon

Одним из центральных определений является объект. С физической точки зрения это некоторая точка, которая отслеживается трекером. В контексте выпускной квалификационной работы нас будет интересовать автотранспорт с ДУТ. Данные единицы характеризуются массой набором датчиков и их настроек. Говоря более строго, датчик (с точки зрения систем спутникового мониторинга) представляет собой некоторую функцию или суперпозицию функций. В качестве аргумента выступают данные, получаемые с трекера или значения с других датчиков, создаваемых пользователем. В качестве значения, выдаваемое датчиком, выступает некоторое число, реже - логическое значение.

Рассмотрим механизм работы на примере топливного бака с двумя ДУТ. Единица, которую мы будем разбирать далее, в необходимые моменты времени отправляет сообщения. Например, некоторые из параметров (таблица 1.1):

Таблица 1.1 — Фрагмент сообщения

Наименование параметра	Значение
Время	2021-04-01 09:40:37
Скорость	0
Координаты	50.7258678, 37.5213918
Количество спутников	4
Высота	187.3
pwr_ext	12.4
c1ls1	935
c1ls2	820

В данном случае, трекер отправляет информацию о напряжении сети автомобиля и информацию с двух датчиков уровня топлива. Информация, выдаваемая датчиком уровня топлива, представлена не в виде какого-то количества топлива в литрах, а имеет некоторый характер, нуждающийся в дополнительной обработке.

После монтажа ДУТов в бак происходит так называемый процесс тарировки, который заключается в постепенном (с некоторым шагом) заполнении бака топливом и снятием показаний с датчика при заданном

количестве литров. Данные заносятся в тарифовочную таблицу, которая содержит некоторое множество точек, задающих функцию:

$$fuel = f(dut)$$

где *fuel* – количество топлива, *dut* – показания с датчика уровня топлива. Значения между точками получаются посредством линейной интерполяции. В случае, если точка лежит за пределами – применяется алгоритм линейной экстраполяции.

Очевидно, что в случае с несколькими независимыми баками суммарный объем топлива будет вычисляться по формуле:

$$sum_fuel = fuel_1 + fuel_2 + \dots + fuel_n = \sum_{i=1}^n fuel_i$$

В системе спутникового мониторинга для объекта настраиваются нужные датчики. В нашем случае это:

- два ДУТа на *clls1*, *clls2*;
- два датчика *FUEL* для преобразования показаний с ДУТов в литры по тарифовочным таблицам
- датчик *SUMM_FUEL* для сохранения информации о суммарном количестве топлива
- датчик «Топливо в баке», который заносится с типом «Датчик уровня топлива» и используется в построении отчетов¹.

Заведенные датчики представлены на рисунке 1.7.

Механизм работы системы с точки зрения передачи и обработки данных представлен на рисунке 1.8.

¹ Как правило, у клиентов не имеется необходимости отслеживать баки отдельно: их интересует суммарный объем топлива

1. Специалистами сервисной службы производится процесс тарировки: порционно заливается топливо и фиксируются показания датчиков. Данные показания заносятся в тарировочную таблицу.

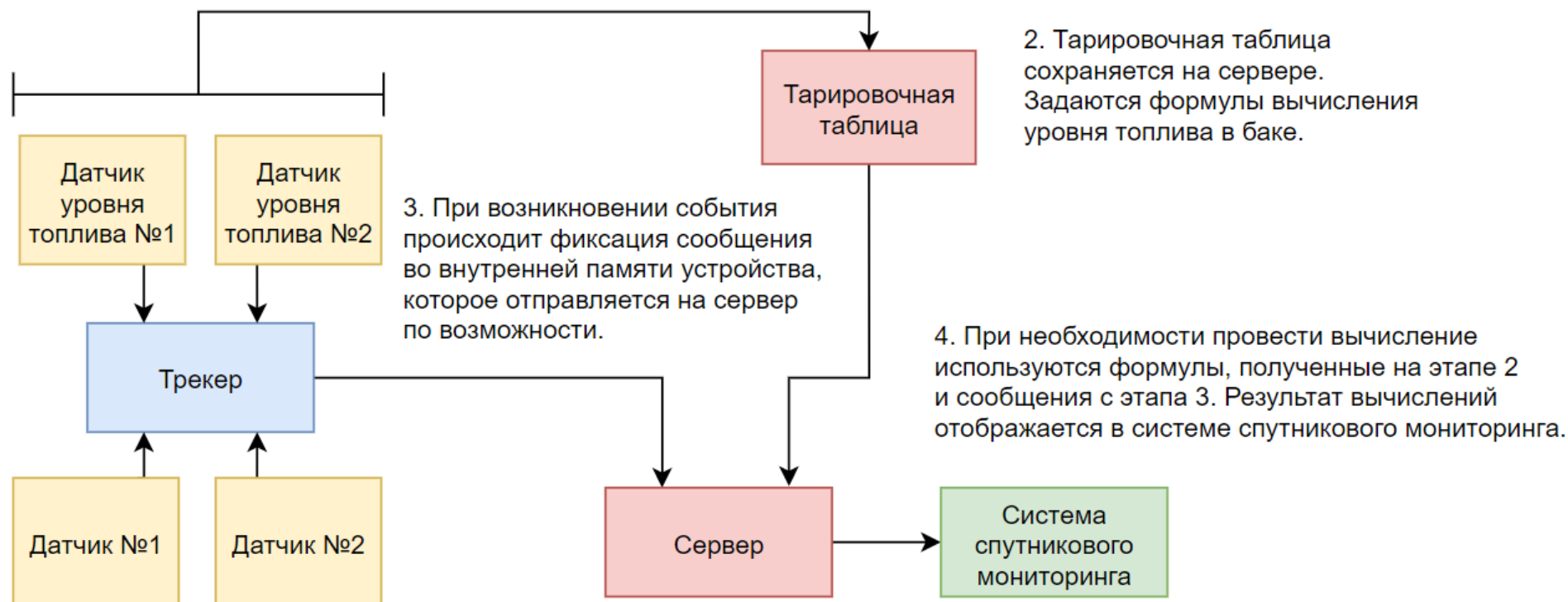


Рисунок 1.7 — Механизм работы системы

Основное

Доступ

Иконка

Дополнительно

Датчики

Произвольные поля

Группы

Команды

Качество вождения

Характеристики

Детектор поездок

Расход топлива

Техобслуживание

+

Создать

Мастер расхода по расчету

Имя	Тип	Ед. изм.	Параметр	Описание	Видимость	Время	
↑ FUEL_1	Произвольный датчик		[DUT_1]		<input type="checkbox"/>	<input type="checkbox"/>	
↑ Напряжение	Датчик напряжения	В	pwr_ext		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
↑ Топливо в баке	Датчик уровня топлива	л	[SUMM_FUE...]		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
↑ FUEL_2	Произвольный датчик		[DUT_2]		<input type="checkbox"/>	<input type="checkbox"/>	
↑ DUT_1	Произвольный датчик		clls1		<input type="checkbox"/>	<input type="checkbox"/>	
↑ DUT_2	Произвольный датчик		clls2		<input type="checkbox"/>	<input type="checkbox"/>	
↑ SUMM_FUEL	Произвольный датчик		[FUEL_1]+[F...]		<input type="checkbox"/>	<input type="checkbox"/>	

Восстановление свойств

Экспорт в файл

Отмена

ОК

Рисунок 1.8 — Заведенные датчики в системе спутникового мониторинга Wialon

Проблема вычисления уровня топлива

Большинство проанализированных работ направлены на создание тех или иных датчиков уровня топлива [4], [5], [6], [7]. Некоторые работы направлены на специализированную технику, такие как летательные аппараты (самолёты, ракеты [8]) или специфический вид топлива [9]. Данная же работа фокусируется не на аппаратных моментах, а на системах, которые обрабатывают показания с датчиков уровня топлива.

Рассмотрим проблему, которая возникает в баках сложной конфигурации. Ранее упоминалось, что для повышения точности измерений уровня топлива используется несколько ДУТов в одном баке. Они позволяют уменьшить влияние колебаний топлива в ходе работы техники. Также можно снять более точные показания в случае наклона единицы. В случае с прямоугольным баком мы могли бы наблюдать следующую картину (Рисунок 1.).

Реальное количество топлива в баке может быть определено как первым ДУТом, так и вторым. В данном примере не существует таких зон, в которой какой-то из ДУТов не может зафиксировать тот или иной объем топлива в баке.

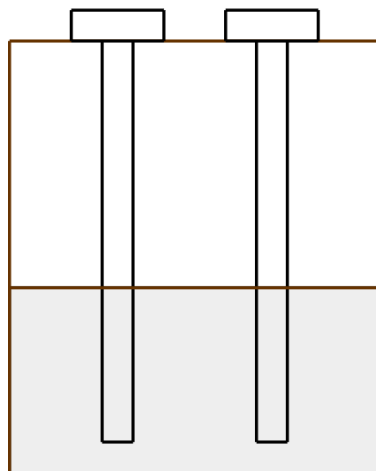


Рисунок 1.9 — Два ДУТа в одном баке

Пример тарифовочных таблиц:

Таблица 1.2 — Тарифовочная таблица к рисунку 1.9

$FUEL_1$	DUT_1	$FUEL_2$	DUT_2
0	30	0	30
10	491	10	409
20	614	20	982
30	1309	30	1227
40	1759	40	1554
50	2086	50	1841
60	2372	60	2331
70	2863	70	2904
80	3149	80	3231
90	3599	90	3681
100	4090	100	4093

Предположим, что в баке находится 50 литров топлива. Тогда первый ДУТ должен показывать около 2086, а второй ДУТ – 1841. Очевидно, что системы мониторинга производят обратный процесс, переводят показания ДУТов. Но в данном случае проблем не возникает ни при каких значениях DUT: если бы мы получили значения $dut_1 = 2372$, $dut_2 = 1554$ мы высчитали бы объем топлива посредством усреднения показаний на ДУТах:

$$sum_fuel = \frac{fuel_1 + fuel_2}{2} = \frac{f_1(dut_1) + f_2(dut_2)}{2} = \frac{60 + 40}{2} = 50$$

Таким образом, даже если техника была наклонена, то при наличии двух ДУТов можно получить более точные значения.

Рассмотрим другой случай: предположим, что бак имеет чуть более сложную форму, и мы монтируем в него два датчика уровня топлива. Первый из них покрывает весь диапазон значений литров в баке. Второй же – перекрывает часть (Рисунок 1.)

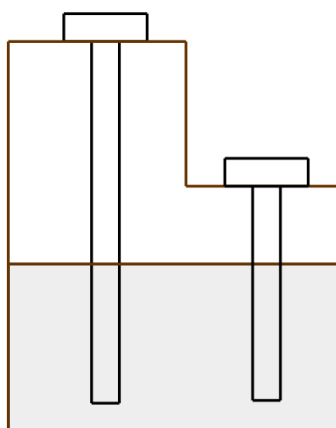


Рисунок 1.10 — Два ДУТа в одном баке (ситуация 2)

Тарировка представлена в таблице 1.3.

Таблица 1.3 — Тарировочная таблица к рисунку 1.10

$FUEL_1$	DUT_1	$FUEL_2$	DUT_2
0	30	0	30
10	245	10	532
20	695	20	1023
30	1145	30	1432
40	1841	40	1841
50	1841	50	1963
60	2413	60	2454
70	2945	70	2740
80	3395	80	2740
90	3763	90	2740
100	4090	100	2740

Для вычисления топлива в баке будут заведены два датчика, которые преобразуют значения, получаемые с ДУТов, в объем топлива в баке. Предположим, что в баке

находится 30 литров. Выдаваемые значения с ДУТов ожидаются в районе 1145 и 1432. И формула для вычисления могла бы выглядеть следующим образом:

$$\begin{aligned} sum_fuel &= \frac{fuel_1 + fuel_2}{2} = \frac{f_1(dut_1) + f_2(dut_2)}{2} = \\ &= \frac{f_1(1145) + f_2(1432)}{2} = \frac{30 + 30}{2} = 30 \end{aligned}$$

Вернёмся к тарифовочной таблице 1.3, которая задаёт функцию преобразования. Но из определения функции следует, что при одном и том же значении аргумента не могут выдаваться разные значения функции. Здравый смысл подсказывает, что значение 2740 говорит о том, что 70 литров в баке по показаниям второго ДУТа действительно имеется. И было бы ожидаемым считать, что

$$f(2740) = 70$$

Но представим, что баке оказалось большее количество топлива: он был полон. Формула вычисления суммы топлива в баке задана на сервере и не может изменяться в зависимости от каких-то условий. Была выбрана следующая:

$$sum_fuel = \frac{fuel_1 + fuel_2}{2} = \frac{f_1(dut_1) + f_2(dut_2)}{2}$$

Но если бак полон, тогда ДУТы показывают значения 4090 и 2740. Получаемый объем топлива:

$$sum_fuel = \frac{fuel_1 + fuel_2}{2} = \frac{f_1(4090) + f_2(2740)}{2} = \frac{100 + 70}{2} = 85$$

не соответствует действительности. Очевидно, что ошибка будет наблюдаться в верхней части бака, и можно было бы не заправлять бак объемами топлива больше 70 литров, что всё равно не является решением. Однако хотелось бы иметь метод, который выдавал бы верные показания при произвольном количестве топлива в баке.

Цель и задачи выпускной квалификационной работы

Целью данной выпускной квалификационной работы является улучшение точности измерения количества топлива в баках сложной конфигурации при использовании нескольких датчиков уровня топлива. В связи с этим были выделены следующие задачи:

- исследовать существующие подходы к решению задач определения уровня топлива в баках сложной конфигурации.
- определить требования к программному обеспечению.
- разработать и реализовать в программном обеспечении алгоритм адаптации данных, получаемых с датчиков уровня топлива для систем спутникового мониторинга.
- провести эксплуатационное приёмочное тестирование разработанного продукта.
- описать алгоритм для специалистов технической поддержки по внедрению метода вычислений в системы спутникового мониторинга.

Вывод

В разделе были даны основные определения, которые касаются процессов установки, настройки и конфигурирования оборудования для контроля уровня топлива. Описаны вычислительные проблемы при определении уровня топлива. В результате поставлена цель выпускной квалификационной работы и сформулированы задачи.

2. ОПИСАНИЕ ПОДХОДА ДЛЯ ВЫЧИСЛЕНИЯ КОЛИЧЕСТВА ТОПЛИВА В БАКАХ СЛОЖНОЙ КОНФИГУРАЦИИ ПО НЕСКОЛЬКИМ ДАТЧИКАМ УРОВНЯ ТОПЛИВА

Баки – не сообщающиеся сосуды

В первой части работы была показана проблема, возникающая при вычислении количества топлива в баках сложной конфигурации. Под последними же будем понимать произвольный бак, для контроля которого требуется использование двух или более ДУТов, и какой-то из датчиков, в силу расположения, шлет одно и то же показание при разном количестве топлива.

Со стороны сервера выдвигается следующее требование: необходимо составить такую формулу, которая бы не зависела от условий и корректно производила вычисления. Рассмотрим следующую конфигурацию топливного бака (рисунок 2.1):

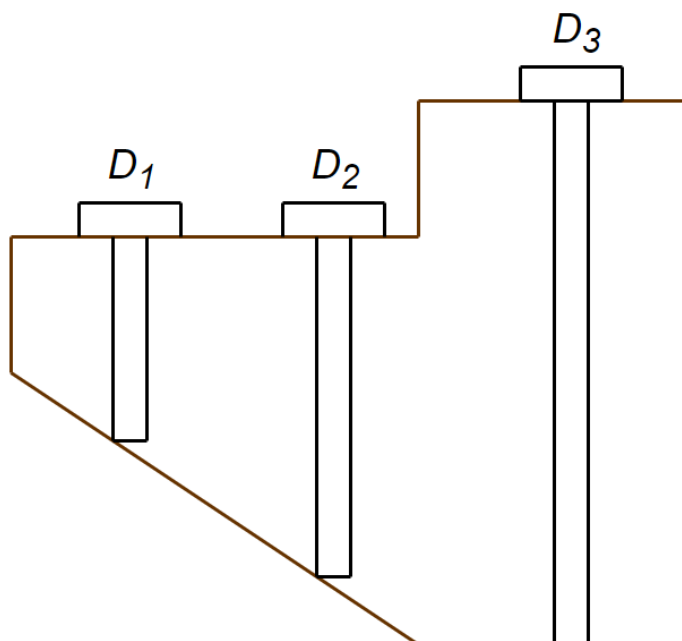


Рисунок 2.1 — Конфигурация топливного бака

с тремя датчиками уровня топлива D_1 , D_2 , D_3 . Поделим бак на зоны таким образом, чтобы в каждой из зон не было смены количества датчиков уровня топлива (рисунок 2.2):

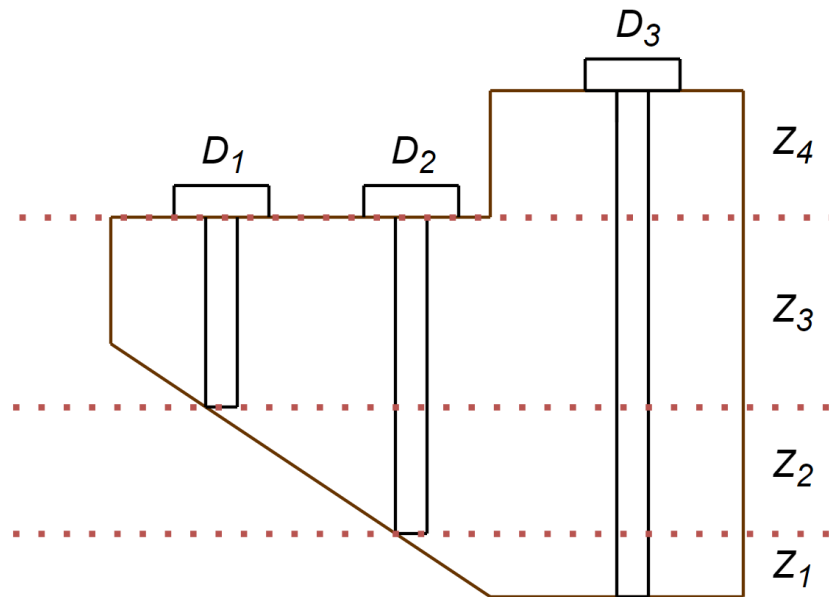


Рисунок 2.2 — Конфигурация топливного бака с выделенными зонами

В таком случае, в Z_1 и Z_4 вычисления производятся только под датчику D_3 , в Z_2 — по D_2 и D_3 , в Z_3 — по D_1 , D_2 , D_3 . При условии того, что техника стоит ровно, вычисления объема топлива могли бы осуществляться как среднее арифметические показателей датчиков для данной зоны:

$$sum_fuel_{Z_1} = f_3(D_3)$$

$$sum_fuel_{Z_2} = \frac{f_2(D_2) + f_3(D_3)}{2}$$

$$sum_fuel_{Z_3} = \frac{f_1(D_1) + f_2(D_2) + f_3(D_3)}{3}$$

$$sum_fuel_{Z_4} = f_3(D_3)$$

Однако реализовать такую разветвляющуюся структуру на стороне сервера невозможно. Как минимум, необходимо знать: какой уровень топлива наблюдается в контексте зон. Но чтобы ответить на этот вопрос, необходимо знать количество топлива в баке. Но ведь именно уровень топлива мы и хотим найти. Следовательно, должен существовать некоторый другой способ вычисления, в котором отсутствовали бы такие разветвления.

Предлагаемое решение сводится к разделению одного физического ДУТа на несколько виртуальных / логических (рисунок 2.3):

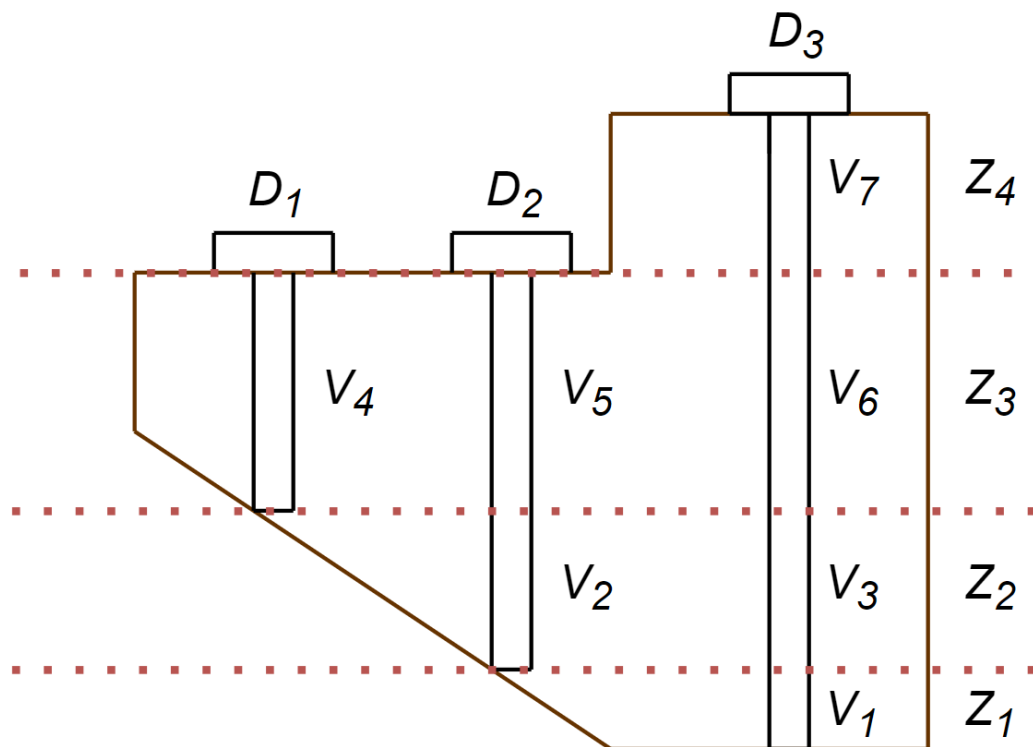


Рисунок 2.3 — Конфигурация топливного бака с выделенными виртуальными датчиками уровня топлива

И требуется задать некоторую функцию $sum_fuel(V_1, V_2, \dots, V_n)$ где N – количество виртуальных датчиков.

Значения с виртуальных датчиков должны вычисляться всегда и входить в итоговую формулу. Будем считать, что виртуальные датчики обладают следующими свойствами:

- Нижняя граница датчика соответствует уровню топлива 0.
- Значения $vdut_i$ берутся с соответствующего dut_j .

Рассмотрим на примере следующей тарифовочной таблицы (синим показаны зоны, по которым происходит изменения показаний на датчике уровне топлива при заполнении бака (таблица 2.3)):

Таблица 2.1 — Тарировочная таблица для рисунка 2.3

	L	D_1	D_2	D_3
Z_1	0	30	30	30
	10	30	30	100
Z_2	20	30	30	213
	30	30	112	345
	40	30	210	450
	50	30	360	567
Z_3	60	30	439	710
	70	120	560	842
	80	241	715	960
	90	365	840	1080
	100	480	965	1200
	110	600	1074	1315
Z_4	120	715	1205	1435
	130	715	1205	1553
	140	715	1205	1670
	150	715	1205	1800

Она соответствует рисунку 2.3. ДУТ D_3 разделяется на 4 виртуальных V_1, V_3, V_6, V_7 . Опишем процесс получения их тарировочных таблиц:

1. Разделить тарировочную таблицу ДУТа на множество таблиц меньшего размера в зависимости от зон. Точки, лежащие на границах зоны, должны оказаться в виртуальном ДУТе и первой и второй зоны. Например, значение 223 на D_3 должно попасть как в V_1 , так и в V_3 .
2. Произведём вычитание показаний литров для каждого уровня таким образом, чтобы нижняя граница соответствующего ДУТа соответствовала значению 0.

Таблица 2.2 с пошаговыми вычислениями для разбиения датчика D_3 представлена на следующей странице:

Таким образом, физический датчик был разбит на множество виртуальных, которые также задают функции. Рассмотрим виртуальный датчик V_6 чуть более подробно (таблица 2.3):

Таблица 2.3 — значения виртуального датчика V_6

L	D_3
0	710
10	842
20	960
30	1080
40	1200
50	1315
60	1435

Если с D_3 придёт значение меньше чем 710 – в системах спутникового мониторинга вычисление будет осуществляться за счёт экстраполяции, и мы получим отрицательное количество топлива по данному ДУТу. С этой целью для каждого датчика добавляются дополнительные значения, чтобы подавить экстраполяцию (таблица 2.4):

Таблица 2.4 — значения виртуального датчика V_6 с добавленными значениями для подавления экстраполяции

L	D_3
0	709
0	710
10	842
20	960
30	1080
40	1200
50	1315
60	1435
60	1436

И тогда формула для разбираемого случая может быть выражена следующим образом:

$$\begin{aligned}
 sum_fuel = f_{v1}(D_3) + \frac{f_{v2}(D_2) + f_{v3}(D_3)}{2} + \frac{f_{v4}(D_1) + f_{v5}(D_2) + f_{v6}(D_3)}{3} \\
 + f_{v7}(D_3)
 \end{aligned}$$

Ещё раз подчеркиваю, для каждого из виртуальных датчиков возможны следующие ситуации:

- Значение на физическом ДУТе меньше, чем покрываемый интервал виртуальным ДУТом – виртуальный ДУТ будет показывать 0 литров
- Значение на физическом ДУТе попадает в покрываемый интервал – виртуальный ДУТ будет показывать видимое им количество литров
- Значение на физическом ДУТе больше покрываемого интервала – виртуальный ДУТ будет показывать максимальное видимое значение на виртуальном ДУТе.

Какое бы значение не шло по физическому ДУТу, виртуальный ДУТ всегда может выдать значение уровня топлива.

Баки – сообщающиеся сосуды

Подход, описанный ранее, может быть применён и для баков – сообщающихся сосудов. Рассмотрим пример (рисунок 2.4):

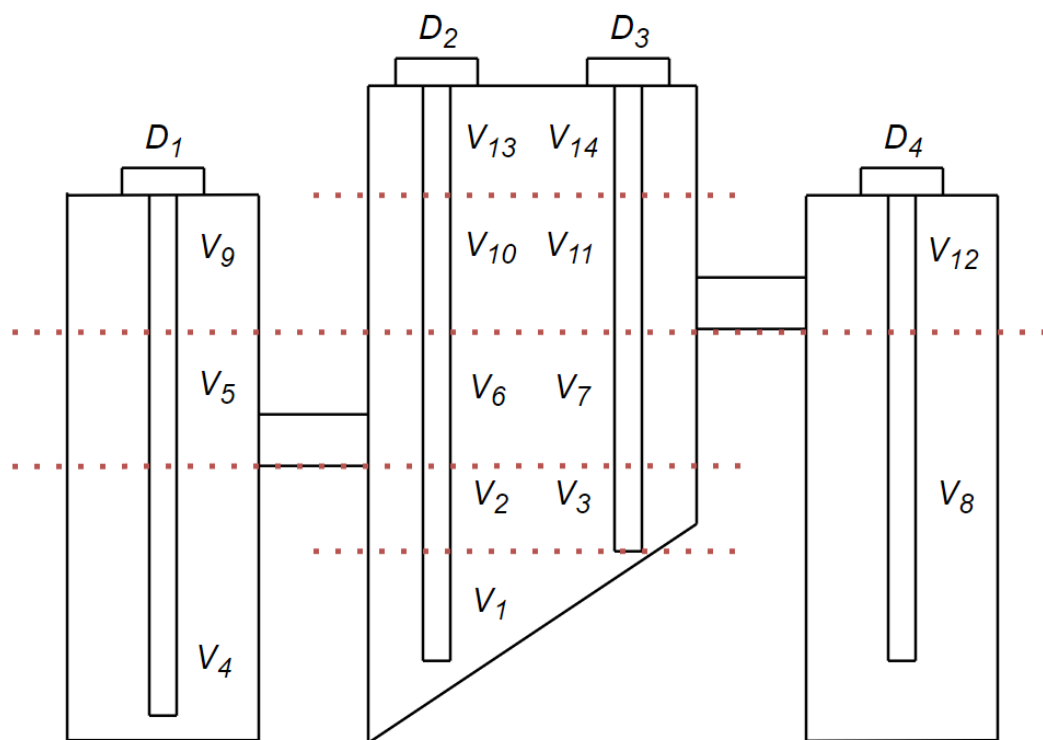


Рисунок 2.4 — Топливный бак в форме сообщающегося сосудов

который носит учебный характер, но позволяет лучше разобраться с последовательностью действий при работе с такой конфигурацией. Сгенерируем пример тарифовочной таблицы. В таблице 2.5 синим цветом выделены изменения показаний с ДУТов при заполнении бака топливом. Правая же часть таблицы показывает диапазон значений в литрах, за которую отвечает тот или иной виртуальный ДУТ:

Таблица 2.5 — Тарифовочная таблица для рисунка 2.4

L	D_1	D_2	D_3	D_4
0	30	30	30	30
10	30	100	30	30
20	30	160	30	30
30	30	220	100	30
40	30	280	160	30
50	30	350	220	30
60	100	350	220	30
70	160	350	220	30
90	220	350	220	30
100	280	350	220	30
110	340	440	280	30
120	400	500	347	30
130	460	560	400	30
140	460	560	400	100
150	460	560	400	160
160	460	560	400	220
170	460	560	400	280
180	460	560	400	340
190	460	560	400	400
200	460	560	400	460
210	520	620	460	520
220	580	680	520	580
230	600	740	580	600
240	664	800	620	664
250	664	850	680	664
260	664	900	730	664

V	$\min(L)$	$\max(L)$
V_1	0	20
V_2	20	50
V_3		
V_4	50	100
V_5	100	130
V_6		
V_7		
V_8	130	200
V_9	200	240
V_{10}		
V_{11}		
V_{12}		
V_{13}	240	260
V_{14}		

Прерывания синих столбцов говорят нам о том, что началась зона сообщения, и топливо начало переливаться в другой сосуд. При достижении уровня сообщения показания с ДУТов продолжают расти далее.

Тарировочную таблицу в представлении слева несколько трудно будет впоследствии разбивать на датчики. Рекомендуется выполнить дополнительное преобразование, заключающееся в дублировании некоторых строк исходной таблицы. После выполнения дублирования можно легко вычленить логические датчики (пример разбиения представлен в таблице 2.6).

После проведения преобразования выполним разделение таблицы на виртуальные датчики уровня топлива. Каждый из виртуальных датчиков уровня топлива будем характеризовать следующим набором свойств:

- *V_DUT_NUMBER* – номер виртуального ДУТа
- *PARENT_DUT_NAME* – имя родительского ДУТа
- *PARENT_LITERS* – показания литров родительского ДУТА на промежутке виртуального датчика уровня топлива
- *V_DUT_LITERS* - преобразованные показания литров по формуле:

$$V_DUT_LITERS_i = PARENT_LITERS_i - PARENT_LITERS_0$$

- *DUT_VALUES* – показания датчика уровня топлива на промежутке
- *ZONE_NUMBER* – номер зоны, в которой расположен виртуальный датчик уровня топлива.

При обработке таблицы программой вы должны получить программный объект со следующим содержимым:

<i>V_DUT_NUMBER</i>	1
<i>PARENT_DUT_NAME</i>	D2
<i>PARENT_LITERS</i>	[0, 10, 20]
<i>V_DUT_LITERS</i>	[0, 10, 20]
<i>DUT_VALUES</i>	[30, 100, 160]
<i>ZONE_NUMBER</i>	1
<i>V_DUT_NUMBER</i>	2
<i>PARENT_DUT_NAME</i>	D2
<i>PARENT_LITERS</i>	[20, 30, 40, 50]
<i>V_DUT_LITERS</i>	[0, 10, 20, 30]
<i>DUT_VALUES</i>	[160, 220, 280, 350]
<i>ZONE_NUMBER</i>	2

Таблица 2.6 — Преобразование таблицы

L	D_1	D_2	D_3	D_4		L	D_1	D_2	D_3	D_4		L	D_1	D_2	D_3	D_4	
0	30	30	30	30		0	30	30	30	30		0					Z_1
10	30	100	30	30		10	30	100	30	30		10		V_1			
20	30	160	30	30		20	30	160	30	30		20					
30	30	220	100	30		20	30	160	30	30		20					Z_2
40	30	280	160	30		30	30	220	100	30		30		V_2	V_3		
50	30	350	220	30		40	30	280	160	30		40					
60	100	350	220	30		50	30	350	220	30		50					
70	160	350	220	30		50	30	350	220	30		50					Z_3
90	220	350	220	30		60	100	350	220	30		60					
100	280	350	220	30		70	160	350	220	30		70	V_4				
110	340	440	280	30		90	220	350	220	30		90					
120	400	500	347	30		100	280	350	220	30		100					
130	460	560	400	30		100	280	350	220	30		100					Z_4
140	460	560	400	100		110	340	440	280	30		110	V_5	V_6	V_7		
150	460	560	400	160		120	400	500	347	30		120					
160	460	560	400	220		130	460	560	400	30		130					
170	460	560	400	280		130	460	560	400	30		130					Z_5
180	460	560	400	340		140	460	560	400	100		140					
190	460	560	400	400		150	460	560	400	160		150					
200	460	560	400	460		160	460	560	400	220		160					
210	520	620	460	520		170	460	560	400	280		170					
220	580	680	520	580		180	460	560	400	340		180					
230	600	740	580	600		190	460	560	400	400		190					
240	664	800	620	664		200	460	560	400	460		200					
250	664	850	680	664		200	460	560	400	460		200	V_9	V_{10}	V_{11}	V_{12}	Z_6
260	664	900	730	664		210	520	620	460	520		210					
						220	580	680	520	580		220					
						230	600	740	580	600		230					
						240	664	800	620	664		240					
						240	664	800	620	664		240					Z_7
						250	664	850	680	664		250		V_{13}	V_{14}		
						260	664	900	730	664		260					

V_DUT_NUMBER	3
PARENT_DUT_NAME	D3
PARENT_LITERS	[20, 30, 40, 50]
V_DUT_LITERS	[0, 10, 20, 30]
DUT_VALUES	[30, 100, 160, 220]
ZONE_NUMBER	2

V_DUT_NUMBER	4
PARENT_DUT_NAME	D1
PARENT_LITERS	[50, 60, 70, 90, 100]
V_DUT_LITERS	[0, 10, 20, 40, 50]
DUT_VALUES	[30, 100, 160, 220, 280]
ZONE_NUMBER	3

V_DUT_NUMBER	5
PARENT_DUT_NAME	D1
PARENT_LITERS	[100, 110, 120, 130]
V_DUT_LITERS	[0, 10, 20, 30]
DUT_VALUES	[280, 340, 400, 460]
ZONE_NUMBER	4

V_DUT_NUMBER	6
PARENT_DUT_NAME	D2
PARENT_LITERS	[100, 110, 120, 130]
V_DUT_LITERS	[0, 10, 20, 30]
DUT_VALUES	[350, 440, 500, 560]
ZONE_NUMBER	4

V_DUT_NUMBER	7
PARENT_DUT_NAME	D3
PARENT_LITERS	[100, 110, 120, 130]
V_DUT_LITERS	[0, 10, 20, 30]
DUT_VALUES	[220, 280, 347, 400]
ZONE_NUMBER	4

и так далее до:

V_DUT_NUMBER	12
PARENT_DUT_NAME	D4
PARENT_LITERS	[200, 210, 220, 230, 240]
V_DUT_LITERS	[0, 10, 20, 30, 40]
DUT_VALUES	[460, 520, 580, 600, 664]
ZONE_NUMBER	6

V_DUT_NUMBER	13
PARENT_DUT_NAME	D2
PARENT_LITERS	[240, 250, 260]
V_DUT_LITERS	[0, 10, 20]
DUT_VALUES	[800, 850, 900]
ZONE_NUMBER	7

V_DUT_NUMBER	14
PARENT_DUT_NAME	D3
PARENT_LITERS	[240, 250, 260]
V_DUT_LITERS	[0, 10, 20]
DUT_VALUES	[620, 680, 730]
ZONE_NUMBER	7

Полученные значения могут быть использованы в дальнейшем для вычисления формулы и последующего внедрения в системы спутникового мониторинга.

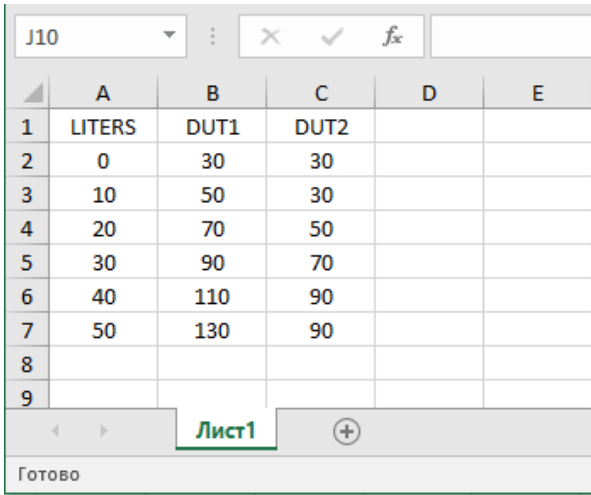
Аналогично ситуации без сообщающихся сосудов нужно добавить дополнительные значения для избегания некорректной экстраполяции в системах спутникового мониторинга (см. таблицы 2.3-2.4).

Описанный выше процесс является довольно трудоемким, и проведение его человеком может привести к вычислительным ошибкам. С другой стороны, хотелось бы избавить пользователя в принципе от проведения таких расчётов в принципе и возложить данный процесс на программное обеспечение. Именно решению данной проблемы будет отведена оставшаяся часть работы.

Требования к программному обеспечению

Данная выпускная квалификационная работа не ставит целью сбор исчерпывающего набора требований (с этим вопросом можно ознакомиться в [10], [11]). Ниже будут перечислены только высокоуровневые требования, касаемые аспектов поведения программы:

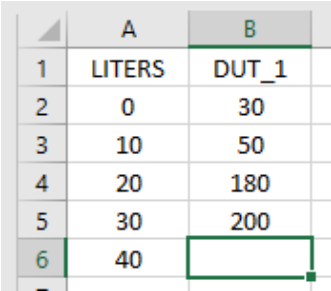
- Исходными данными в задаче является набор тарифовочных таблиц. Пример, созданный в MS Excel (рисунок 2.5):



	A	B	C	D	E
1	LITERS	DUT1	DUT2		
2	0	30	30		
3	10	50	30		
4	20	70	50		
5	30	90	70		
6	40	110	90		
7	50	130	90		
8					
9					

Рисунок 2.5 — Формат исходных данных

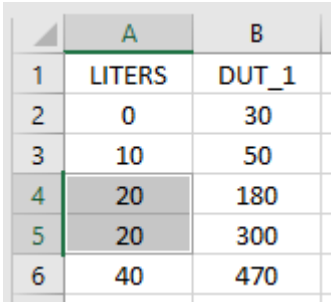
- Первый столбец обязан иметь название LITERS. Именовывать последующие столбцы разрешается произвольно (указанные значения будут использоваться в качестве имени родительского датчика уровня топлива).
- В случае наличия ошибок в исходных данных, программа информирует пользователя. Под ошибками считать:
 - Наличие пропусков в таблице в произвольном столбце тарифовочной таблицы (рисунок 2.6):



	A	B
1	LITERS	DUT_1
2	0	30
3	10	50
4	20	180
5	30	200
6	40	

Рисунок 2.6 — Таблица с пропущенным значением

- Колонка LITERS не является возрастающей (рисунок 2.7):



	A	B
1	LITERS	DUT_1
2	0	30
3	10	50
4	20	180
5	20	300
6	40	470

Рисунок 2.7 — Тарифовочная таблица с невозрастающими показаниями количества уровня топлива

- Наличие убывающих значений в показании ДУТов (все оставшиеся колонки, кроме LITERS) при увеличении количества топлива (рисунок 2.8):

	A	B	C
1	LITERS	DUT_1	DUT_2
2	0	30	30
3	10	50	50
4	20	180	180
5	30	170	200
6	40	470	470
7	50	500	460

Рисунок 2.8 — Убывающие значения LITERS

- Наличие двух последовательных рядов таблицы, в которых не наблюдается изменение показаний для ДУТов (рисунок 2.9):

	A	B	C
1	LITERS	DUT_1	DUT_2
2	0	30	30
3	10	30	30
4	20	79	50
5	30	250	200
6	40	470	470

Рисунок 2.9 — Строки с не отличающимися значениями по всем датчикам уровня топлива

- Поддерживаемые форматы файлов исходных данных: XLSX, XLS.
- При наличии нескольких баков требуется каждый отдельный бак завести на отдельном листе (рисунок 2.10). Имя листа будет использовано в качестве имени бака.

	A	B	C
1	LITERS	DUT_1	
2	0	30	
3	10	50	
4	20	180	
5	30	300	
6	40	470	
7			
8			

	A	B	C	D	E	F
1	LITERS	DUT_1	DUT_2			
2	0	30	50			
3	10	50	70			
4	20	180	120			
5						
6						
7						
8						

Рисунок 2.10 — Пример исходных данных при нескольких датчиках уровня топлива

- Нумерация создаваемых датчиков должна производиться на основании порядка заполнения топлива баком
- Программа получает на вход файл из некоторой директории. После его обработки в той же директории должна создаваться папка, которая содержит в себе следующие файлы:
 - Разбиение всех физических датчиков на логические (виртуальный датчики) и их тарифовочные таблицы.
 - Текстовый файл с формулой для вычисления уровня топлива бака
 - Графики:
 - $f(Liters, DUT_i)$
 - Расположение ДУТов относительно друг друга

Выбор инструментов для разработки

Для разработки приложения были выбраны следующие инструменты:

- Python 3.9 – высокоуровневый язык общего назначения, который ориентирован на быстрое написание прототипа, читаемости кода и его качества. Среда разработки: PyCharm Community. Предпочтение было отдано языку, так как он является бесплатным, и его сторонние библиотеки позволяют легко оперировать данными.
- Git – система контроля версий.

Библиотеки:

- Pandas. Библиотека, предоставляющая классы Series и DataFrame, которая позволяет удобно манипулировать многомерными массивами. Опирается на работы Numpy.
- Numpy. Библиотека предоставляет набор инструментов для работы с многомерными массивами. Такие встроенные типы данных как кортежи ведут к неоправданной затрате памяти. Данный же инструмент позволяет использовать более оптимизированные структуры данных для решения задач.
- Matplotlib. Библиотека разработана для визуализации данных. Инструмент не отличается дружелюбный интерфейсом к новичкам, но для опытных пользователей предоставляет широкий спектр низкоуровневых возможностей для работы с графикой.
- Openpyxl, xlrd. Библиотеки для работы с форматами.xlsx.

Выводы

Во втором разделе был описан предлагаемый алгоритм (без детализации) для баков – не сообщающихся сосудов и баков – сообщающихся сосудов, а также сформулированы требования к программному обеспечению.

3. РЕАЛИЗАЦИЯ АЛГОРИТМА ДЛЯ РАЗБИЕНИЯ ДАТЧИКОВ УРОВНЯ ТОПЛИВА И ПОСТРОЕНИЯ ЗАВИСИМОСТЕЙ

Описание алгоритма

Процесс разработки организуем «сверху вниз» (нисходящее программирование), которое начинается с определения целей решения проблемы, после чего будем детализировать каждый этап до тех пор, пока не будет разработана детальная программа.

Наибольшую ценность для пользователя представляют:

- Тарировочные таблицы в формате .csv для виртуальных датчиков уровня топлива.
- Текстовый файл с формулой вычисления уровня топлива по виртуальным датчикам уровня топлива.
- Графические зависимости.

Блок-схема в укрупненных блоках представлена на рисунке 3.1.

Опишем удобное представление для объекта `fuel_system_config`:

```
fuel_system_config = {
    "tank_name_1": {
        "calibration_table" : <pd.DataFrame>
        "virtual_duts_table": <pd.DataFrame>
    }
    "tank_name_2": {
        "calibration_table" : <pd.DataFrame>
        "virtual_duts_table": <pd.DataFrame>
    }
}
```

где `tank_name_i` – наименование бака, `calibration_table` – тарировочная таблица для бака, `virtual_duts_table` – таблица со столбцами: `V_DUT_NUMBER`, `PARENT_DUT_NAME`, `PARENT_LITERS`, `V_DUT_LITERS`, `DUT_VALUES`, `ZONE_NUMBER`.

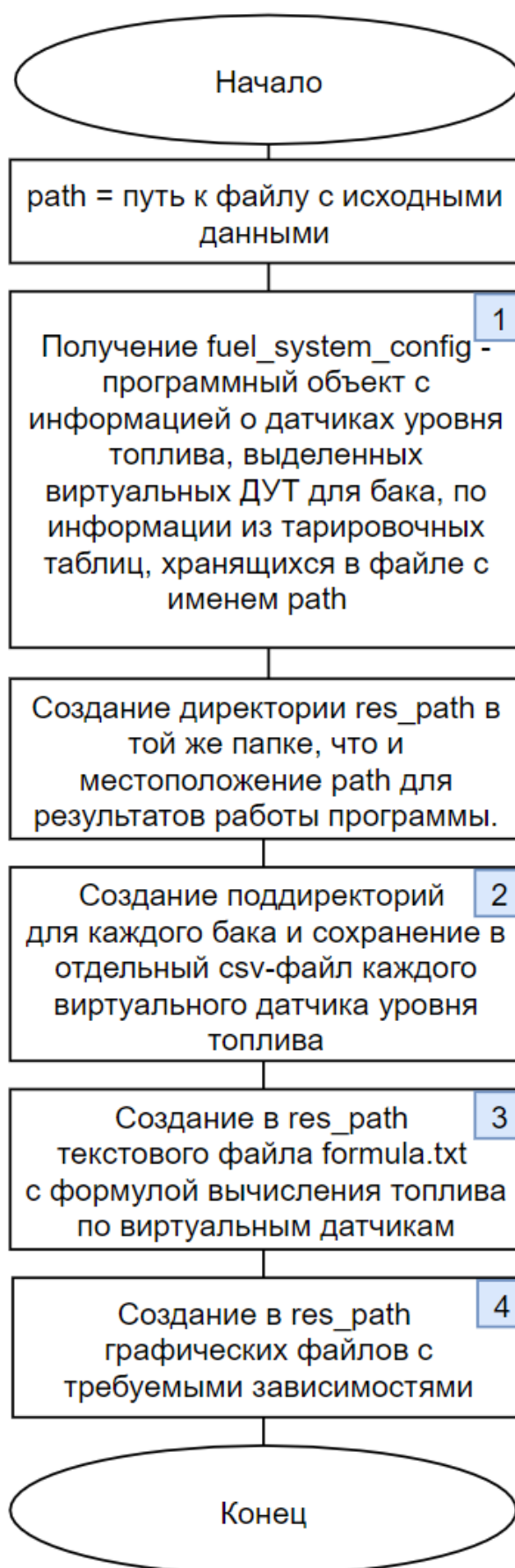


Рисунок 3.1 — Блок-схема в укрупненных блоках

Получение программного объекта `fuel_system_config` состоит из следующих этапов (рисунок 3.2; листинг 3.1):

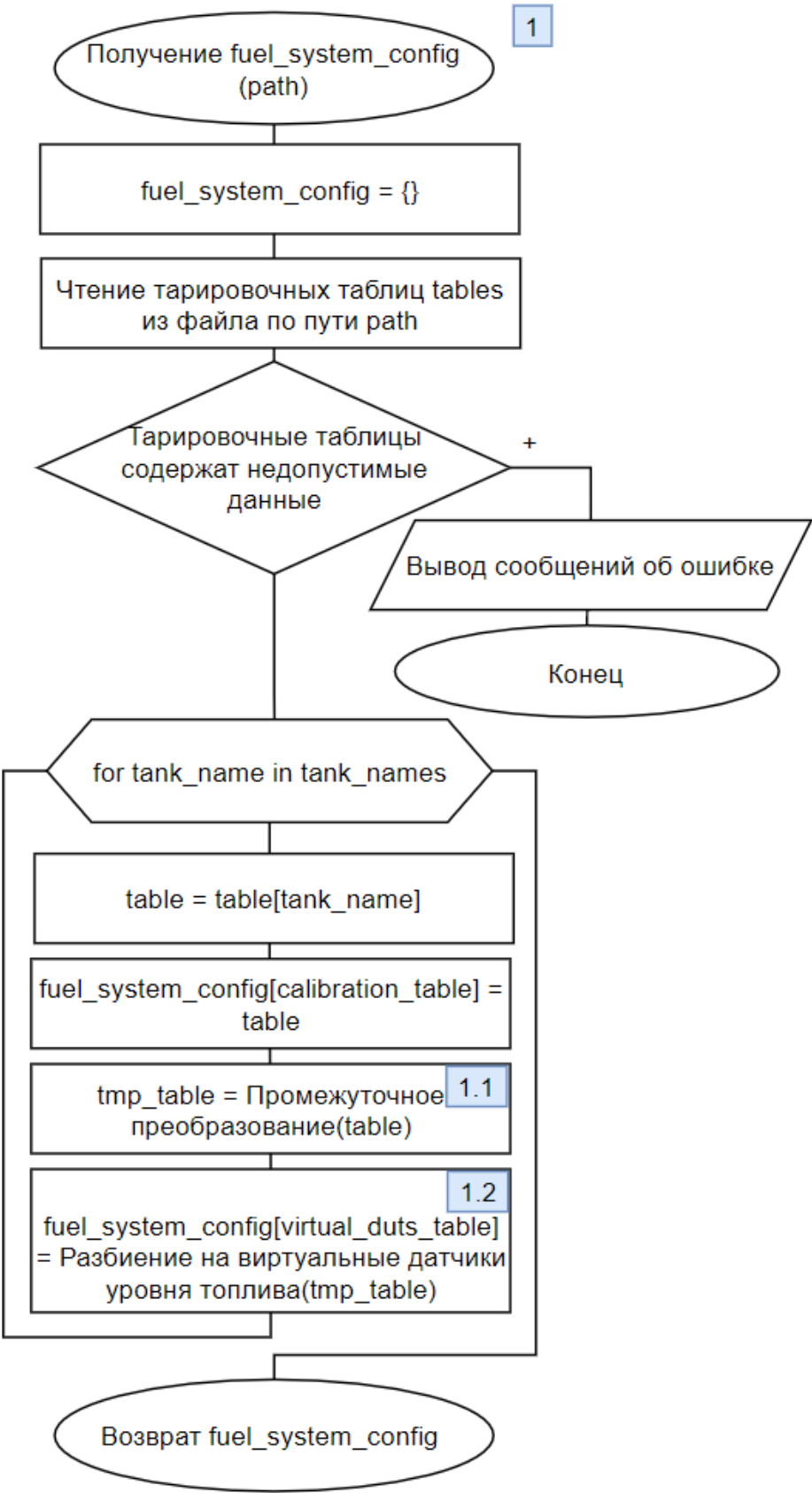


Рисунок 3.2 — Получение объекта `fuel_system_config`

```

import pandas as pd
import openpyxl

from DataChecker import DataChecker
from TableTransformer import TableTransformer
from TableSplitter import TableSplitter

class FuelSystemConfigCreator:

    def __init__(self, path):
        self.path = path
        self.fuel_system_config = {}
        self.open_file()
        self.get_virtual_dut_table()

    def open_file(self):
        try:
            wb = openpyxl.load_workbook(self.path)
        except Exception:
            print(f"open error file {self.path}.")
        n_sheets = len(wb.sheetnames)
        for sheet in range(n_sheets):
            self.fuel_system_config[sheet] = {}
            calibration_table = pd.read_excel(self.path, sheet_name=sheet, engine='openpyxl')
            DataChecker(calibration_table).check_table()
            self.fuel_system_config[sheet]["calibration_table"] = calibration_table

    def get_virtual_dut_table(self):
        for tank_name in self.fuel_system_config:
            calibration_table = self.fuel_system_config[tank_name]["calibration_table"]
            _ = TableTransformer(calibration_table).transform()
            self.fuel_system_config[tank_name]["virtual_duts_table"] = TableSplitter(_).split()

    def create(self):
        self.open_file()
        self.get_virtual_dut_table()
        return self.fuel_system_config

```

Листинг 3.1 – реализация алгоритма, описанного на рисунке 3.2

Описание алгоритмов проверки исходных данных на предмет ошибок производиться не будет, так как функции довольно просты.

После чтения исходных данных последовательно перебираются тарифовочные таблицы и выполняется создание промежуточной таблицы (описанной во 2 части работы), которая упрощает проведение дальнейших вычислений (блок-схема представлена на рисунке 3.4).

Преобразованная таблица разбивается на виртуальные датчики уровня топлива (блок схема представлена на рисунке 3.5). Детализация блока 1.2.1 посредством блок-схем описываться не будет, но реализация может быть позаимствована из с листинга 3.2.

При вычислении промежуточной таблицы и в процессе разбиения на виртуальные ДУТы используется функция, описанная на рисунке 3.6:

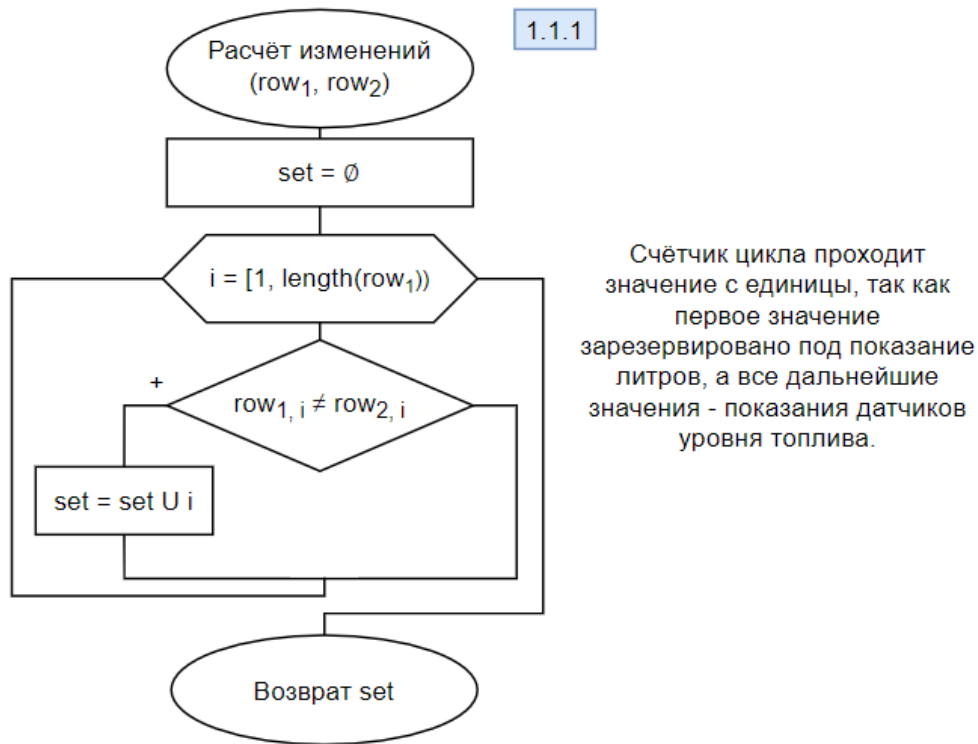


Рисунок 3.3 — Получение объекта fuel_system_config

которая возвращает множество номеров датчиков уровня топлива, зафиксировавших изменения между двумя строками тарифовочной таблицы.

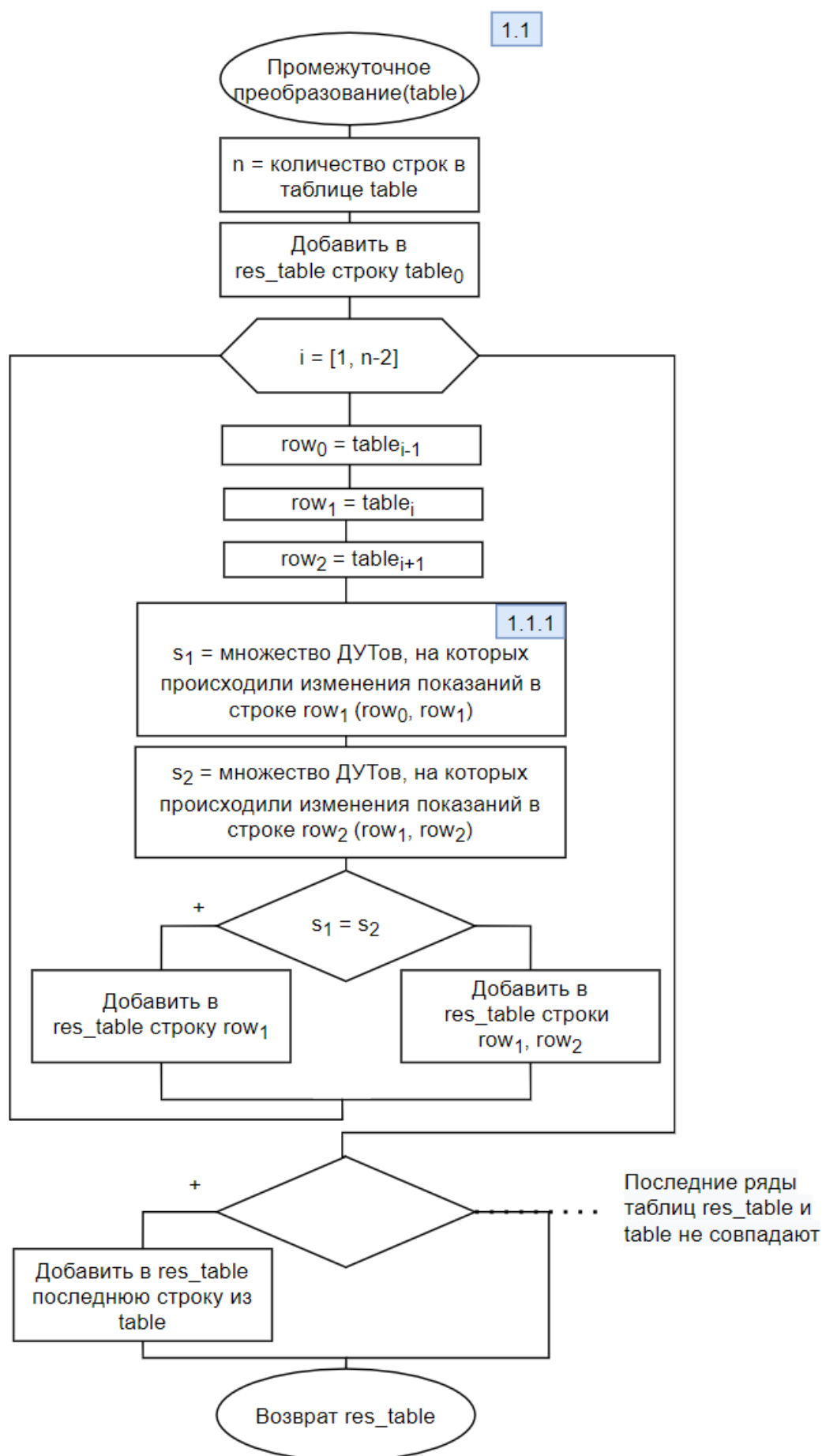


Рисунок 3.4 — Преобразование таблицы

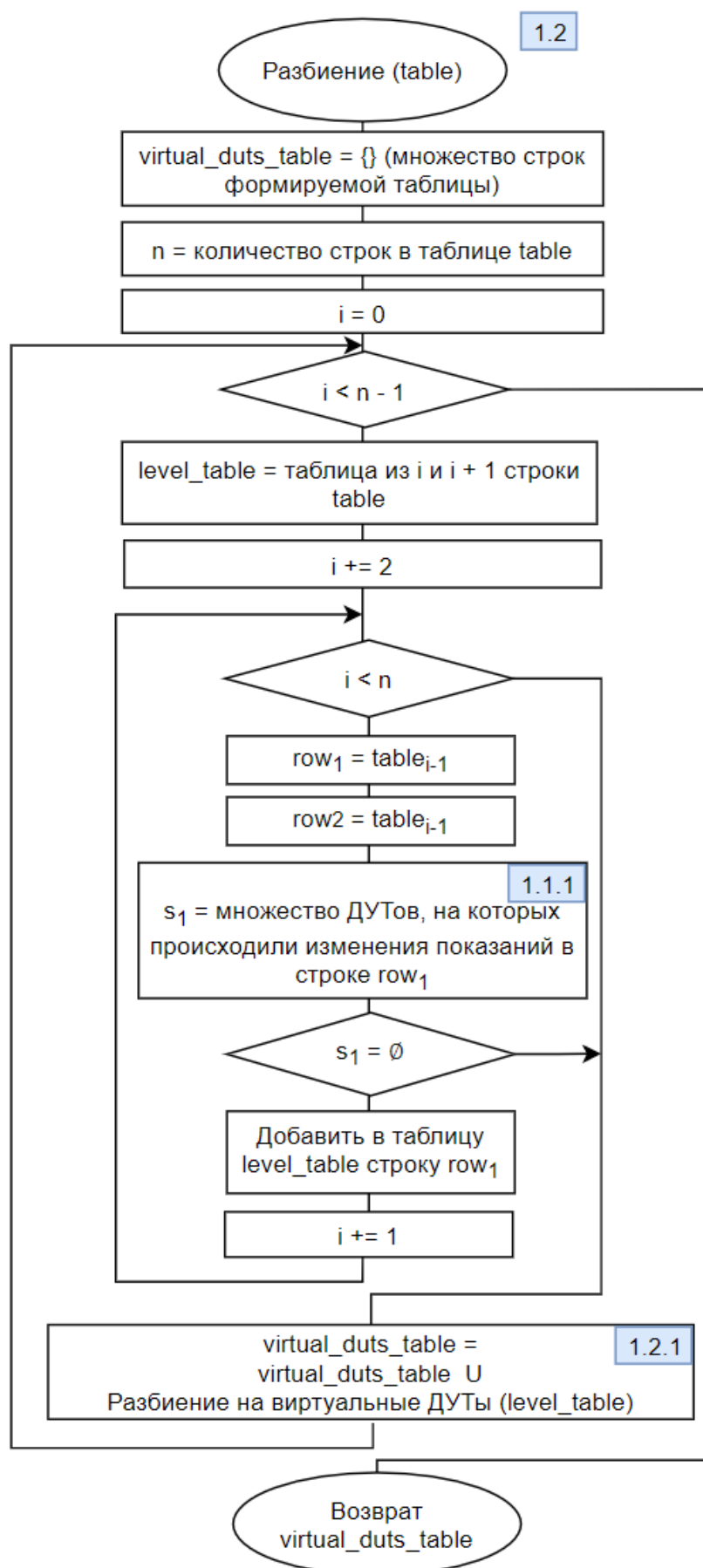


Рисунок 3.5 — Разбиение на виртуальные датчики уровня топлива

```

import pandas as pd

class TableTransformer:

    def __init__(self, pd_df: pd.DataFrame):
        self.table = pd_df

    def transform(self):
        res_table = [self.table.iloc[0]]
        for i in range(1, self.table.shape[0] - 1):
            row0 = self.table.iloc[i - 1]
            row1 = self.table.iloc[i]
            row2 = self.table.iloc[i + 1]
            s1 = [i for i in range(1, len(row1)) if row0[i] != row1[i]]
            s2 = [i for i in range(1, len(row2)) if row1[i] != row2[i]]
            if s1 == s2:
                res_table.append(self.table.iloc[i])
            else:
                res_table.append(self.table.iloc[i])
                res_table.append(self.table.iloc[i])
        last_row_res = res_table[-1]
        last_row_table = self.table.iloc[-1, :]
        if any(last_row_res != last_row_table):
            res_table.append(last_row_table)

        res_table = pd.DataFrame(res_table)
        res_table.index = range(res_table.shape[0])

        return res_table

```

Листинг 3.2 – реализация алгоритма, описанного на рисунке 3.4

```

import pandas as pd

class TableSplitter:

    def __init__(self, table: pd.DataFrame):
        self.table = table
        self.virtual_duts_table = []
        self.zone_number = 0
        self.__split()

    def __split(self):
        n = len(self.table)
        i = 0
        while i < n - 1:
            tmp = self.table.iloc[i:i+2][:]
            i += 2
            while i < n:
                row1 = self.table.iloc[i - 1][:]
                row2 = self.table.iloc[i][:]
                change = [i for i in range(1, len(row1)) if row1[i] != row2[i]]
                if not change:
                    break
                tmp = tmp.append(row2)
                i += 1
            self.__update(tmp)
        return self

```

```

def __update(self, tmp):
    self.zone_number += 1
    for dut_name in tmp.columns[1:]:
        if tmp[dut_name].iloc[0] != tmp[dut_name].iloc[-1]:
            d = pd.Series({
                "V_DUT_NUMBER": len(self.virtual_duts_table) + 1,
                "PARENT_DUT_NAME": dut_name,
                "PARENT_LITERS": tmp.LITERS.values,
                "V_DUT_LITERS": [x - tmp.LITERS.values[0]
                                for x in tmp.LITERS.values],
                "DUT_VALUES": tmp[dut_name].values,
                "ZONE_NUMBER": self.zone_number
            })
            self.virtual_duts_table.append(d)

def split(self):
    return self.virtual_duts_table

```

Листинг 3.3 – реализация алгоритма, описанного на рисунке 3.5

Создание графических файлов необходимо для контроля качества тарировки. Наличие специфических скачков позволяет обнаружить ошибки в процессе тарирования. Для контроля выводимой формулы хотелось бы также иметь представление о взаимном расположении датчиков уровня топлива в баке: существует необходимость понимать - за какой диапазон показаний отвечает тот или иной датчик.

Количество графиков должно создаваться в зависимости от конфигурации системы и определяется как:

$$n_{plots} = 2n_{tanks}$$

Для каждого из баков должны быть построены следующие зависимости:

- Количество топлива в баке от показаний на датчике уровня топлива
- Покрываемый диапазон топлива тем или иным датчиком уровня топлива.

Решение предлагается реализовать посредством библиотеки *matplotlib*, которая предоставляет низкоуровневый интерфейс для работы с графикой.

Код для генерации изображения довольно сложен и приведение его в тексте работы было бы избыточно: внимание хочется уделить получаемым результатам. Для тарифовочной таблицы 2.6, которая является отображением следующей конфигурации (рисунок 3.6):

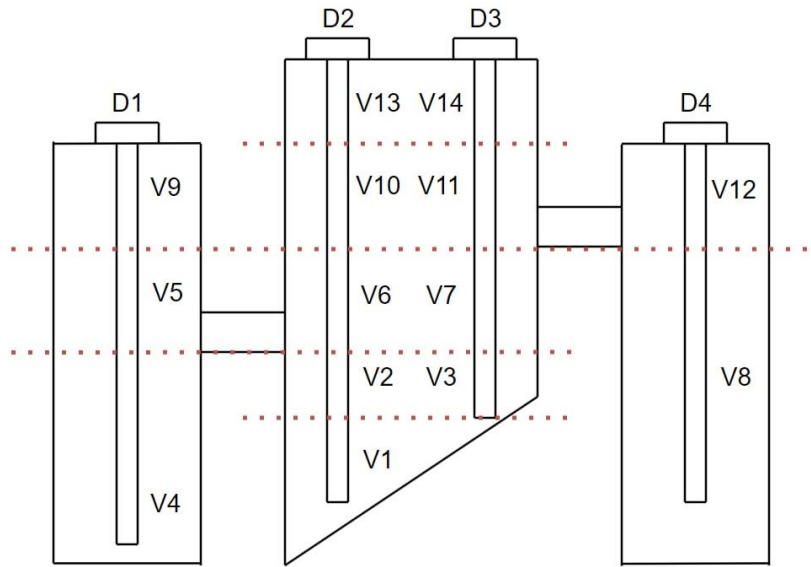


Рисунок 3.6 — Конфигурация бака

выполняется построение следующих графиков (рисунок 3.7):

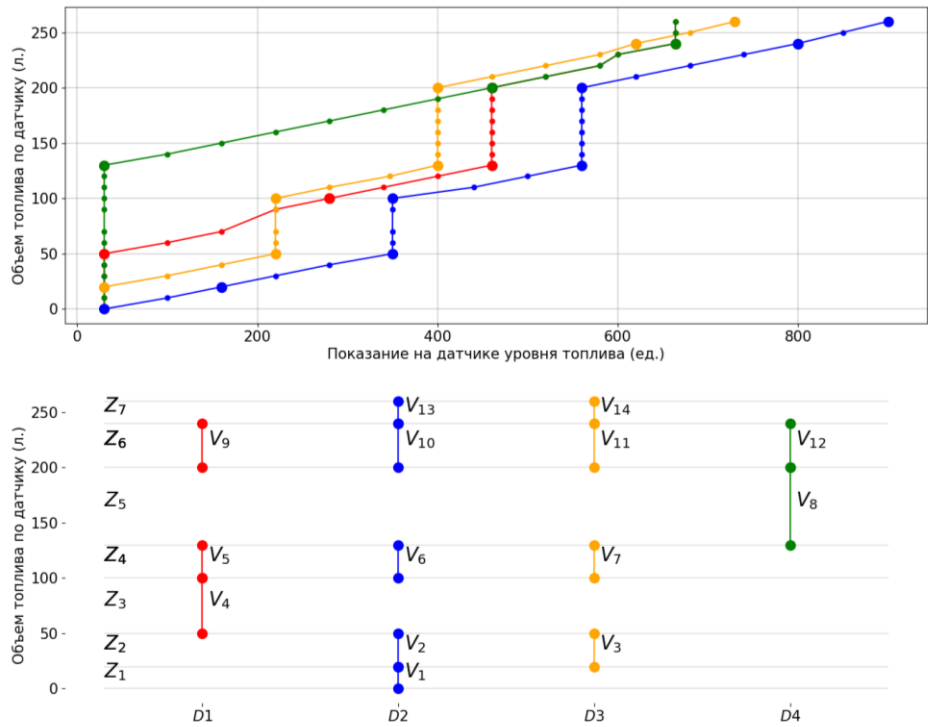


Рисунок 3.7 — Графики для конфигурации к рисунку 3.6

Сфокусируемся на выводимых данных. Реализованная программа выделила 7 зон. Область покрытия каждого из ДУТов отображена на нижнем графике. Видно, что на D_1 имеется разрыв между виртуальными датчиками V_5 и V_9 что является свидетельством того, что на данном промежутке началось переливание топлива к некоторому другому датчику. И им является датчик D_4 . Как только показания на V_8 достигнут максимума – начнут происходить изменения по всем датчикам уровня топлива.

Код для построения графиков может быть найден в приложении к данной работе.

Вычисление формулы при наличии разделения на виртуальные ДУТы не представляет сложности. Необходимо пройти по всем зонам Z_i , и тогда количество топлива может быть вычислено по следующей формуле:

$$sum_fuel = \sum_i \frac{f_1 + \dots + f_{n_{Z_i}}}{n_{Z_i}}$$

где $\frac{f_1 + \dots + f_{n_{Z_i}}}{n_{Z_i}}$ – среднее арифметическое показаний по виртуальным датчикам уровня топлива на уровне Z_i .

Файл, который является точкой входа в программу описан в листинге 3.4. В нём объявлена функция `process`, которая может быть модифицирована в зависимости от нужд. В данном случае создаётся директория с файлами-результатами работы программы там же, где и исходный файл (подробнее описано в требованиях к заданию).

Структура проекта представлена на рисунке 3.8. Исходный код программы поделен на 7 файлов:

- Класс `DataChecker` предназначен для проверки исходных данных на наличие ошибок;
- Класс `FormulaCreator` генерирует формулы для вычисления топлива в различном представлении для дальнейшего использования;

- Класс `FuelSystemConfigCreator` выполняет создание программного объекта `fuel_system_config`, содержание которого описано в разделе 2.
- Файл `main` – точка входа в программу.
- Класс `Plotter` выполняет построение зависимостей в виде графиков.
- Класс `TableSplitter` производит разбиение тарифовочной таблицы на виртуальные датчики уровня топлива.
- Класс `TableTransformer` выполняет подготовку исходных данных для удобного разбиения данных функцией `TableSplitter`.

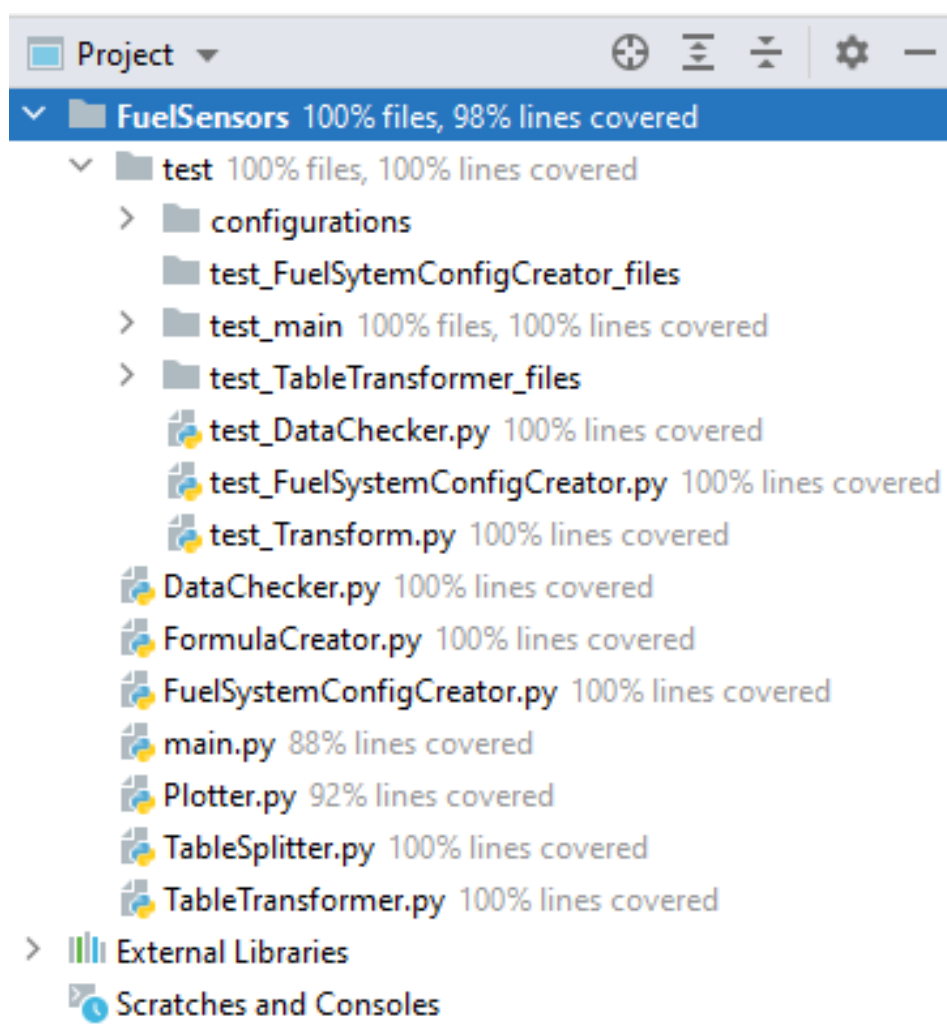


Рисунок 3.8 — Структура разработанного проекта

Основные аспекты, которые касаются рассмотрения алгоритмов были описаны. Остановимся подробнее на процессе установки приложения и интеграции результатов вычислений в систему спутникового мониторинга *wialon*.

```

import os
from tkinter import filedialog

import pandas as pd

from FuelSystemConfigCreator import FuelSystemConfigCreator
from Plotter import Plotter
from FormulaCreator import FormulaCreator

def process(path=None, save_path=None):
    if path is None:
        path = filedialog.askopenfilename(filetypes=[("Excel files", ".xlsx .xls")])
        save_path = ".".join(path.split(".")[:-1]) + "/"

    os.makedirs(save_path, exist_ok=True)
    fsc = FuelSystemConfigCreator(path).create()
    fc = FormulaCreator(fsc)

    for tank_name in fsc:
        os.makedirs(save_path + r"/" + tank_name, exist_ok=True)
        for virtual_dut in fsc[tank_name]["virtual_duts_table"]:
            liters = list(virtual_dut.V_DUT_LITERS)
            liters = [liters[0]] + liters + [liters[-1]]
            dut_values = list(virtual_dut.DUT_VALUES)
            dut_values = [dut_values[0] - 1] + dut_values + [dut_values[-1] + 1]
            df = pd.DataFrame({
                "values": dut_values,
                "liters": liters,
            })

            with open(save_path + r"/" + tank_name + r"/wialon formula.txt", "w") as f:
                f.write(fc.wialon_formula[tank_name])
            df.to_csv(save_path + r"/" + tank_name + f"//V{virtual_dut.V_DUT_NUMBER}.csv",
                      sep=";", header=False, index=False)

    Plotter().get_fx(fsc, fc.latex_formula_full, save_path=save_path)

if __name__ == "__main__":
    process()

```

Тестирование программного обеспечения

В случае некорректной настройки оборудования на стороне сервера возможно наблюдение ложных скачков уровня топлива. Возникновение таких ситуаций приведёт к тому, что специалисту сервисной службы придётся выезжать непосредственно к единице и проводить ряд мероприятий. Очень часто бывает, что техника находится в заметном удалении и ложный выезд обращается компании в убыток.

Таким образом, к корректности данного программного обеспечения предъявляются повышенные требования. В качестве тестовых данных послужили реальные тарифовочные таблицы, накопленные в ходе исследования, которые проверяют следующий набор ситуаций (в скобках укажем марку транспортного средства). Тестовую ситуацию опишем в подписи к рисунку.

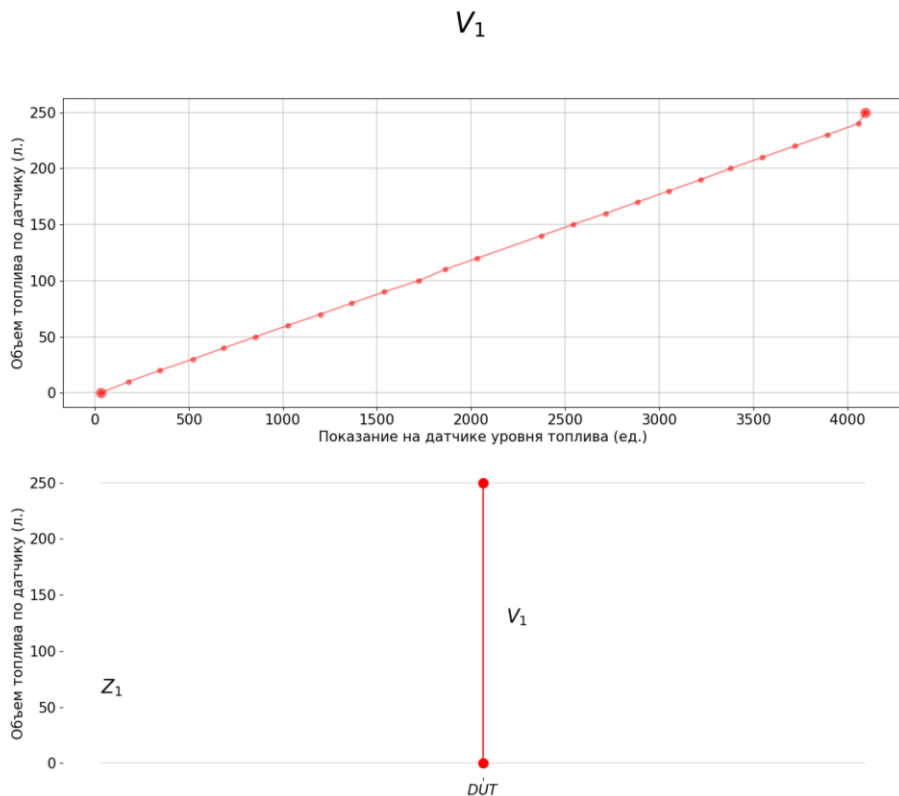


Рисунок 3.9 — Один датчик уровня топлива, один бак (Камаз)

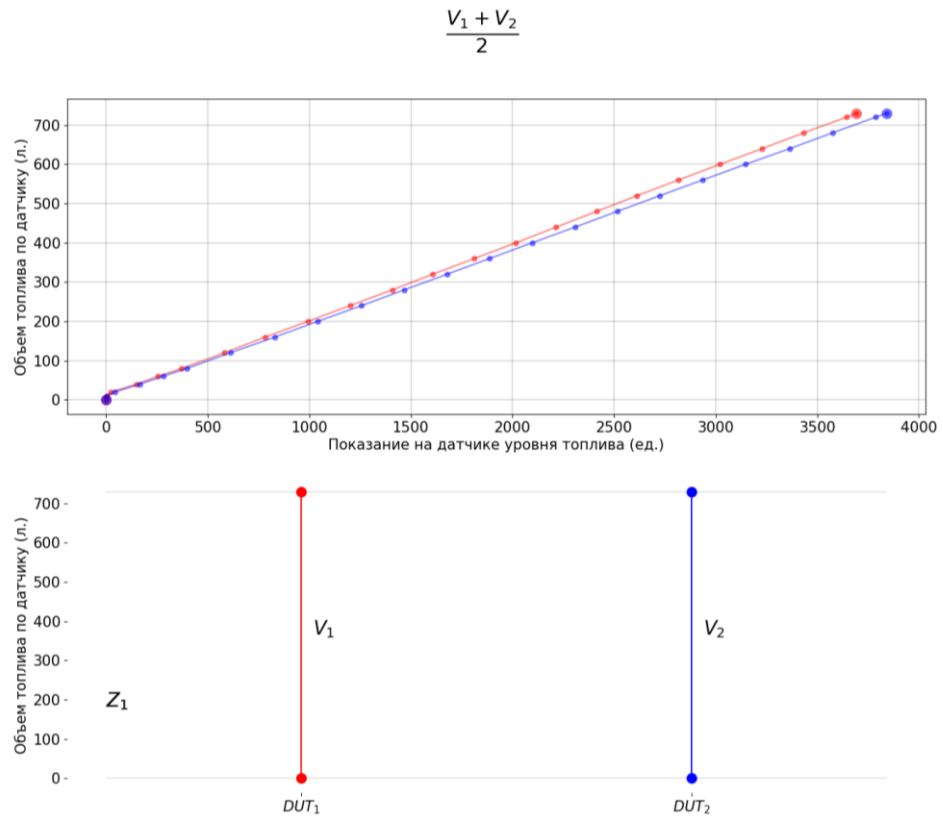


Рисунок 3.10 — Два датчика уровня топлива, установленных параллельно друг другу (MAN)

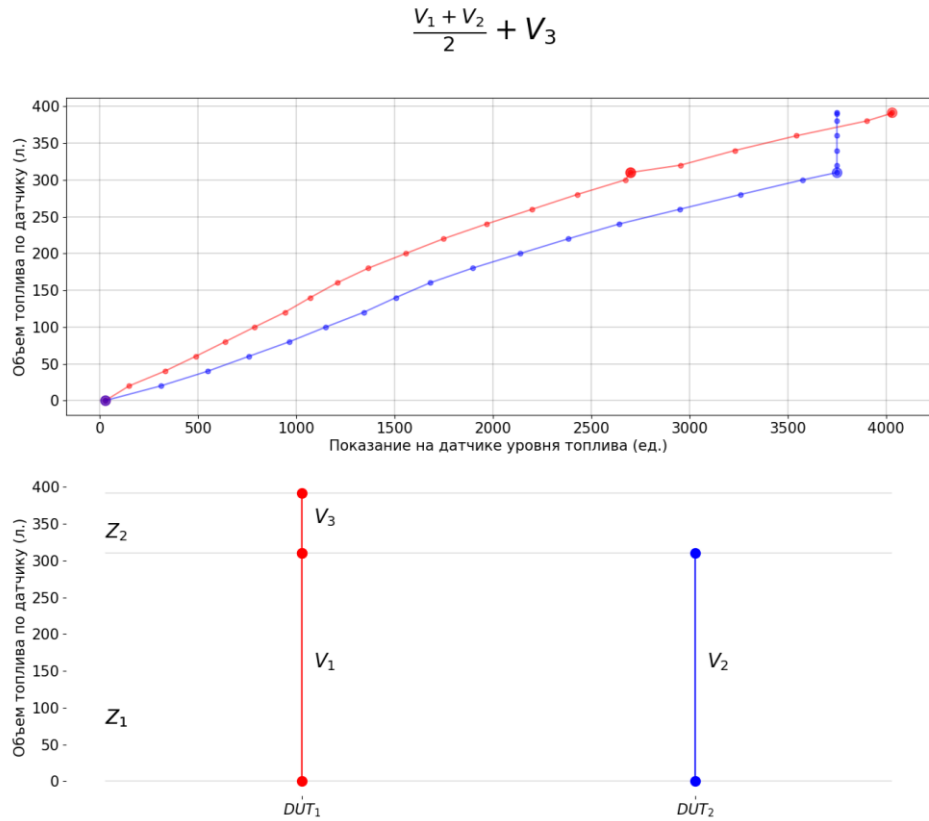


Рисунок 3.11 — Два датчика уровня топлива, один из которых не перекрывает верхнюю часть бака (Case)

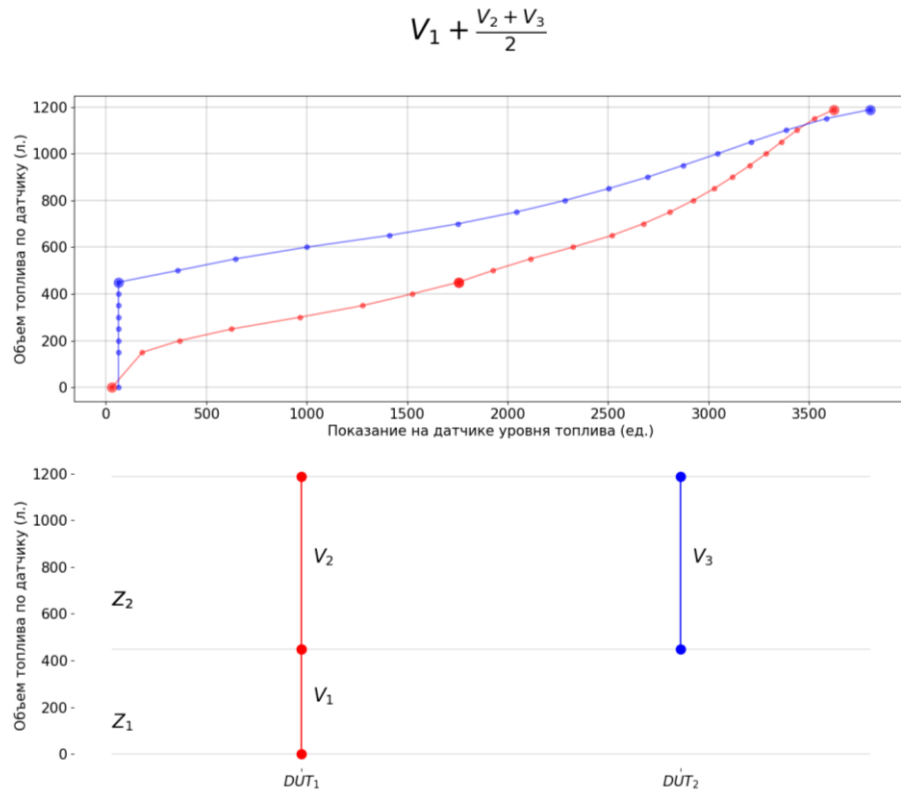


Рисунок 3.12 — Два датчика уровня топлива, один из которых не перекрывает нижнюю часть бака (Challenger)

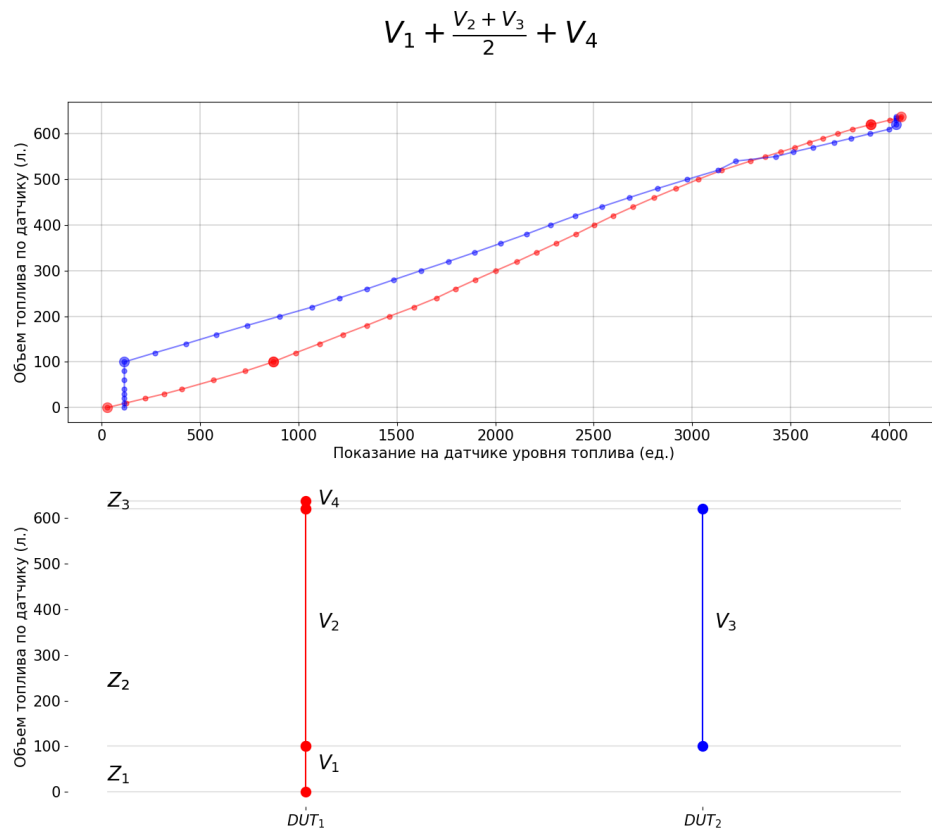


Рисунок 3.13 — Два датчика, один из которых покрывает всю высоту бака, второй – центральную часть бака

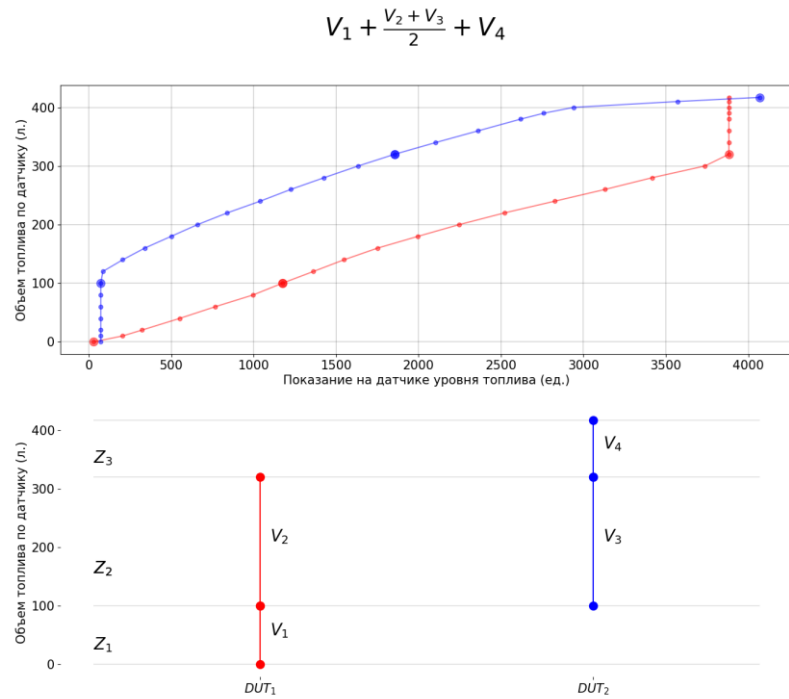


Рисунок 3.14 — Два датчика, один из которых покрывает нижнюю часть бака, но не покрывает верхнюю; второй датчик – перекрывает верхнюю часть, но не перекрывает нижнюю (New Holland)

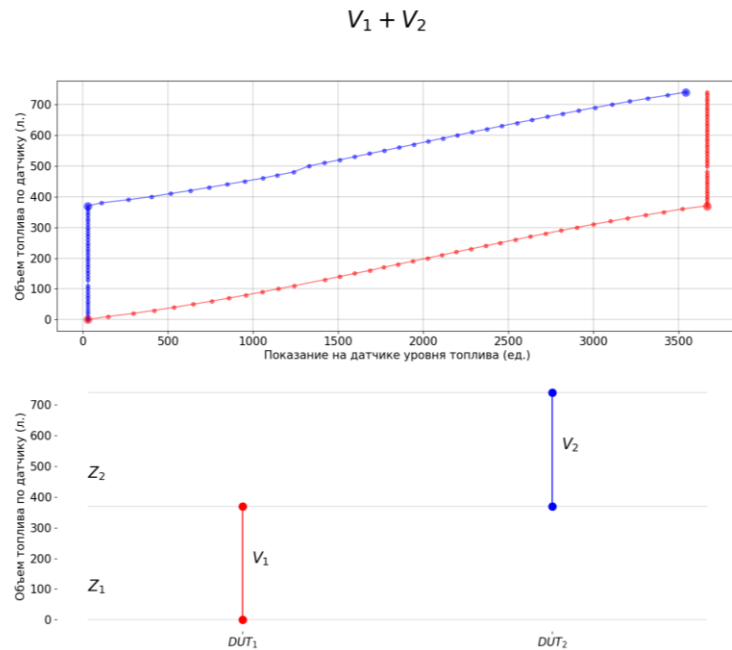


Рисунок 3.15 — Два датчика, показания второго начинаются по окончанию первого датчика²

² Данная конфигурация бака встречается крайне редко, на момент написания работы не представилось возможным найти реальные исторические данные по такому типу объектов.

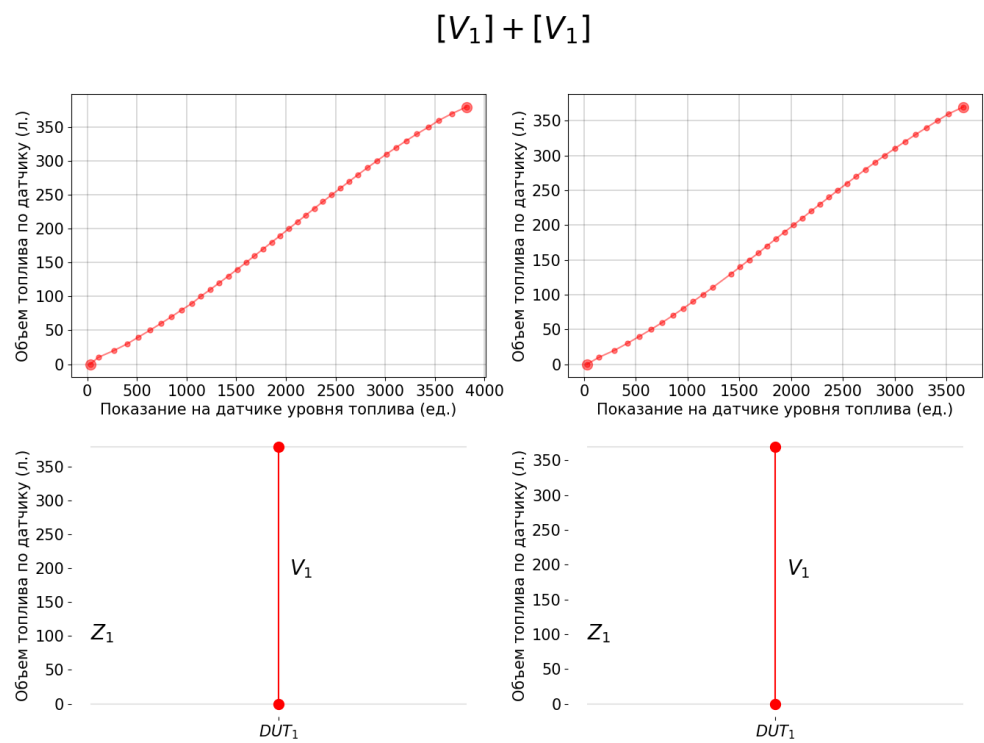


Рисунок 3.16 — Два датчика расположенных в разных баках (Freightliner)

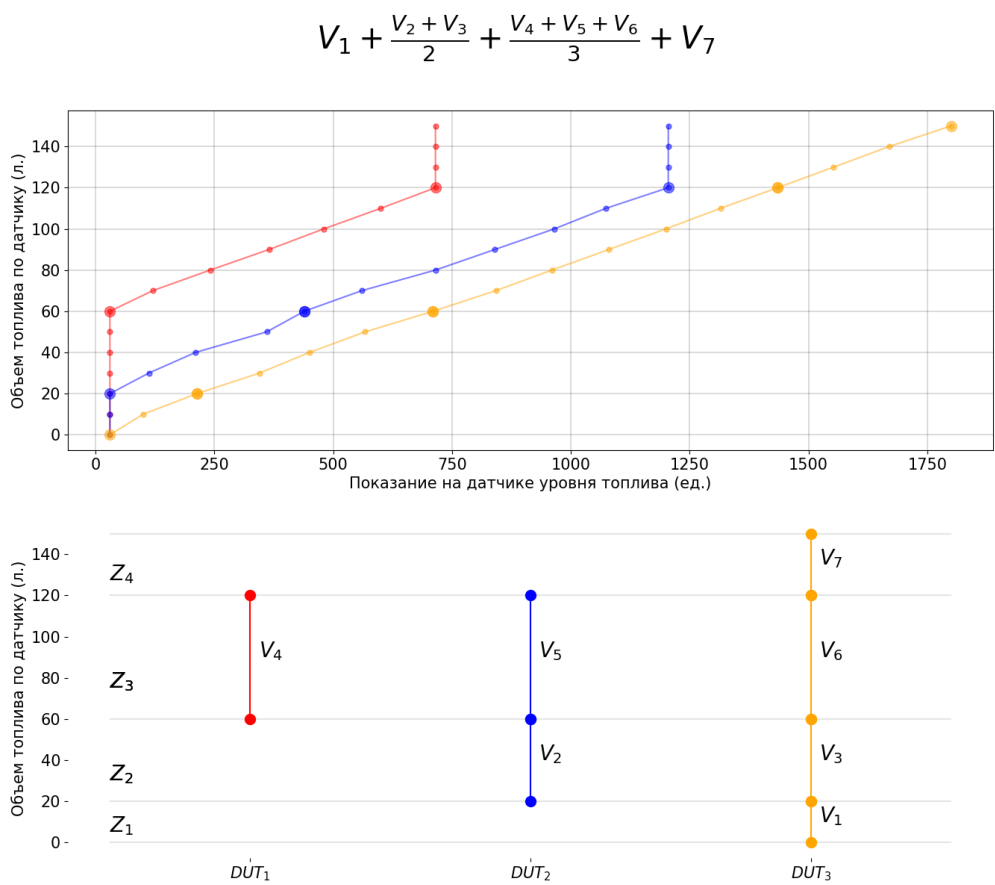


Рисунок 3.17 — Три датчика уровня топлива в одном баке (данные из таблицы 2.1)

Установка программного обеспечения

Установка приложения для операционной системы windows состоит из следующих этапов:

- Установка интерпретатора языка программирования Python
- Установка необходимых пакетов
- Создание bat-файла для удобного запуска приложения

Установка интерпретатора Python

Для установки интерпретатора Python необходимо перейти на официальный сайт Python: <https://www.python.org/downloads/> и скачать последнюю версию (рисунок 3.17):

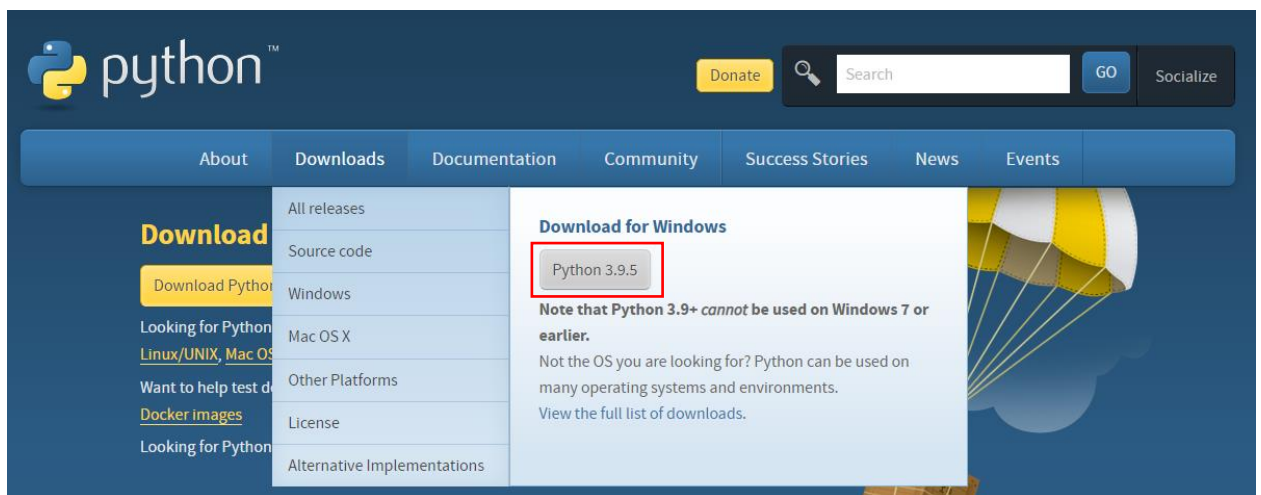


Рисунок 3.17 — Установка Python

Установка пакетов

Для установки пакетов необходимо выполнить следующие действия:

1. Переход в папку Scripts
 - a. Ввести в меню «Пуск» запрос Python;
 - b. Нажать правой кнопкой мыши по ярлыку и выбрать «Свойства» (рисунок 3.18).

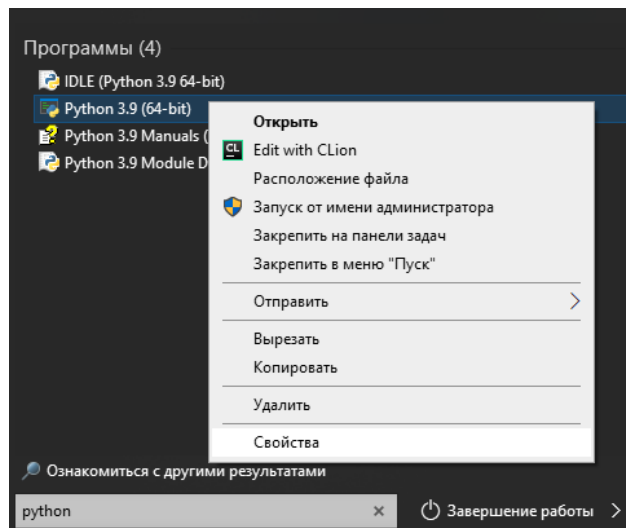


Рисунок 3.18 — Получение свойств

- с. В появившемся диалоговом окне путь может быть найден (рисунок 3.19):

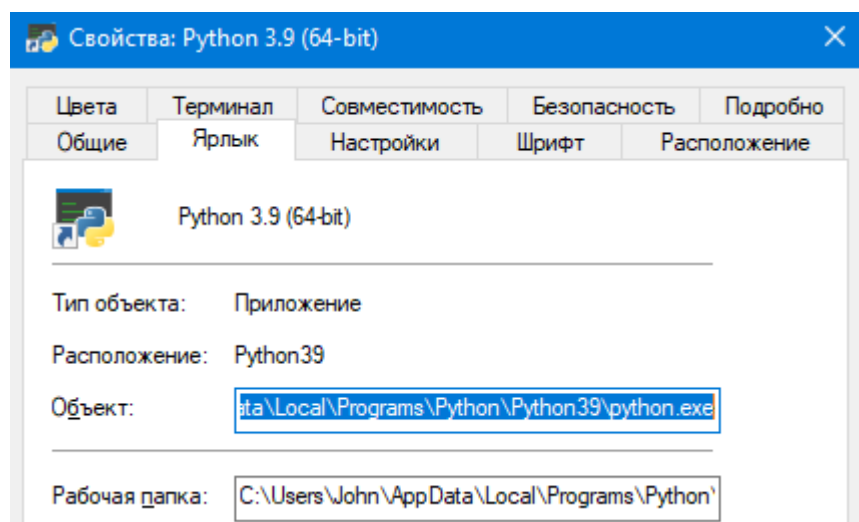


Рисунок 3.19 — Получение пути к файлу python.exe

- d. Скопируйте весь путь за исключением “python.exe”. В примере выше будет получен путь
 “C:\Users\John\AppData\Local\Programs\Python\Python39\”;
- е. Откройте командную строку от имени администратора:
- i. Нажмите комбинацию клавиш Win+R;
 - ii. Введите в окне cmd и нажмите ОК (рисунок 3.20);

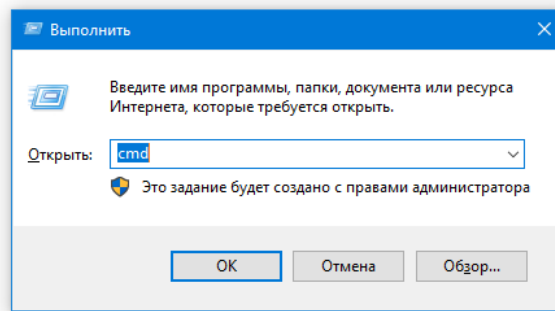


Рисунок 3.20 — Окно «Выполнить».

- f. В консоли введите команду `cd` и вставьте скопированный на этапе d путь и выполните переход в папку Scripts (рисунок 3.21).

Для установки пакета необходимо ввести команду «`pip install <имя пакета>`»:

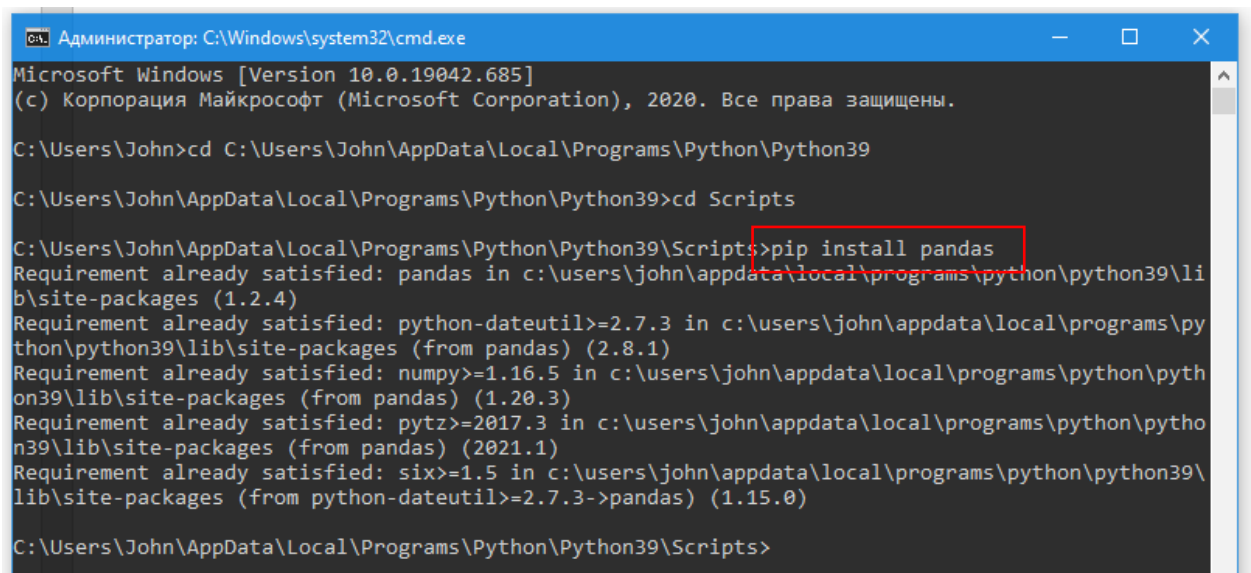


Рисунок 3.20 — Установка пакета pandas

Для работы приложения необходимо установить следующие пакеты:

- pandas [12]
- matplotlib [13]
- xlrd [14]
- openpyxl [15]

Создание bat-файла

Чтобы возможно было выполнять запуск приложения с ярлыка, необходимо:

- Создать в удобном месте файл с расширением bat (рисунок 3.21):

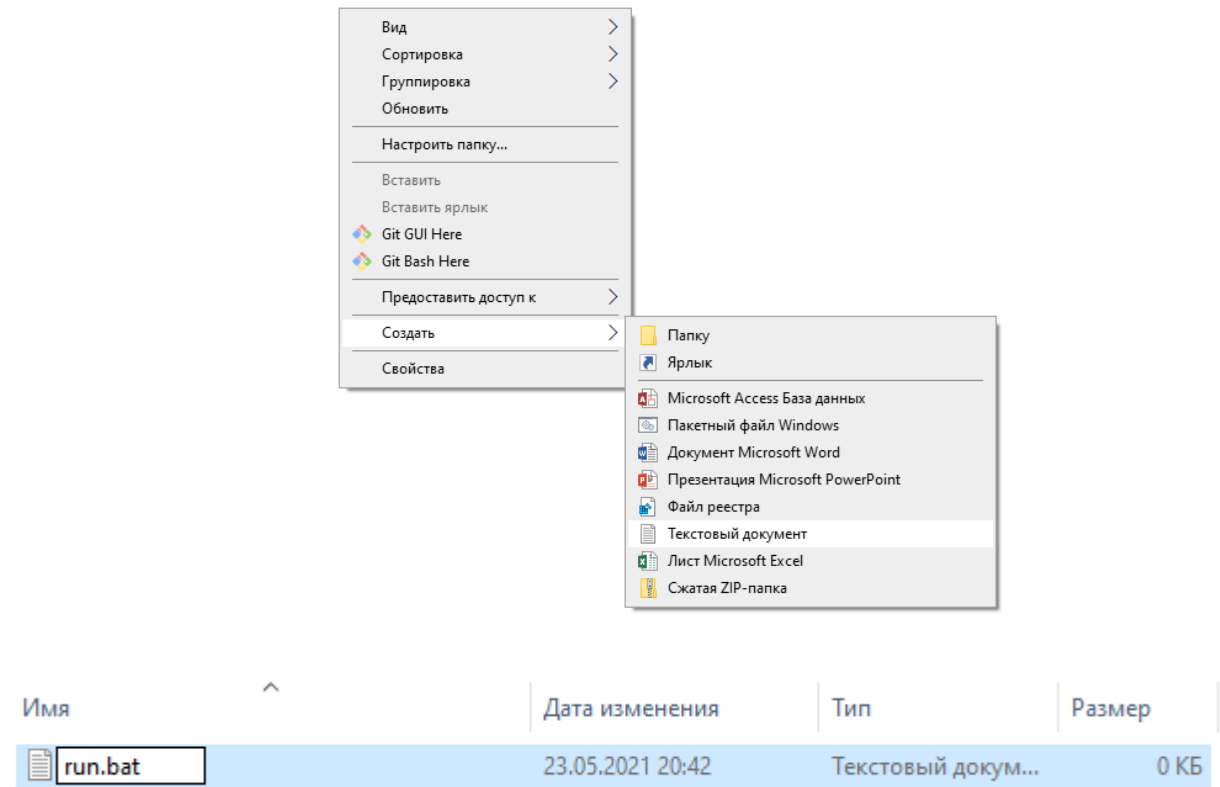


Рисунок 3.21 — Создание bat-файла

- Откройте файл для редактирования при помощи блокнота (рисунок 3.22):

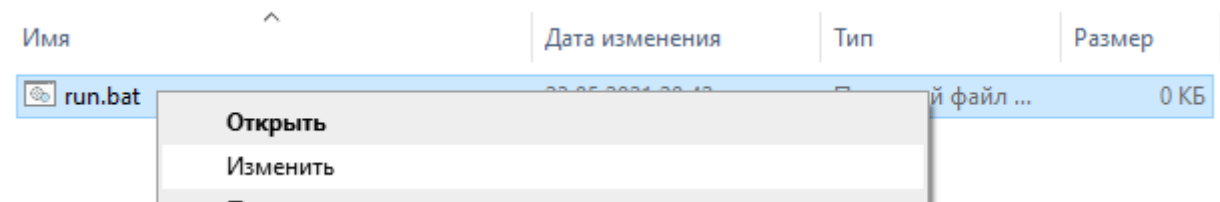


Рисунок 3.22 — Изменение bat-файла

- В качестве текста файла укажите два пути: первый путь – к интерпретатору Python `python.exe`, второй путь – путь к файлу `main` с архивом программы (рисунок 3.23):

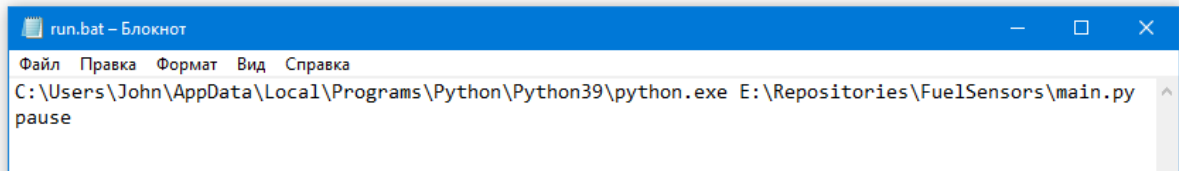


Рисунок 3.23 — Изменение bat-файла

После выполнения вышеперечисленных шагов, можно осуществлять запуск программы посредством двойного щелчка левой кнопки мыши по ярлыку.

24 мая 2021 года в лаборатории технической защиты информации было проведено инсталляционное тестирование на одной из предоставленных машин. Среди учетных записей пользователей не было найдено ни одной, которая бы не содержала русских букв. Пришлось осуществить запуск приложения из консоли указав необходимые пути.

Информация о машине, работе с консолью (установка пакетов и запуск приложения), результаты работы программы предоставлены на рисунке 3.24)

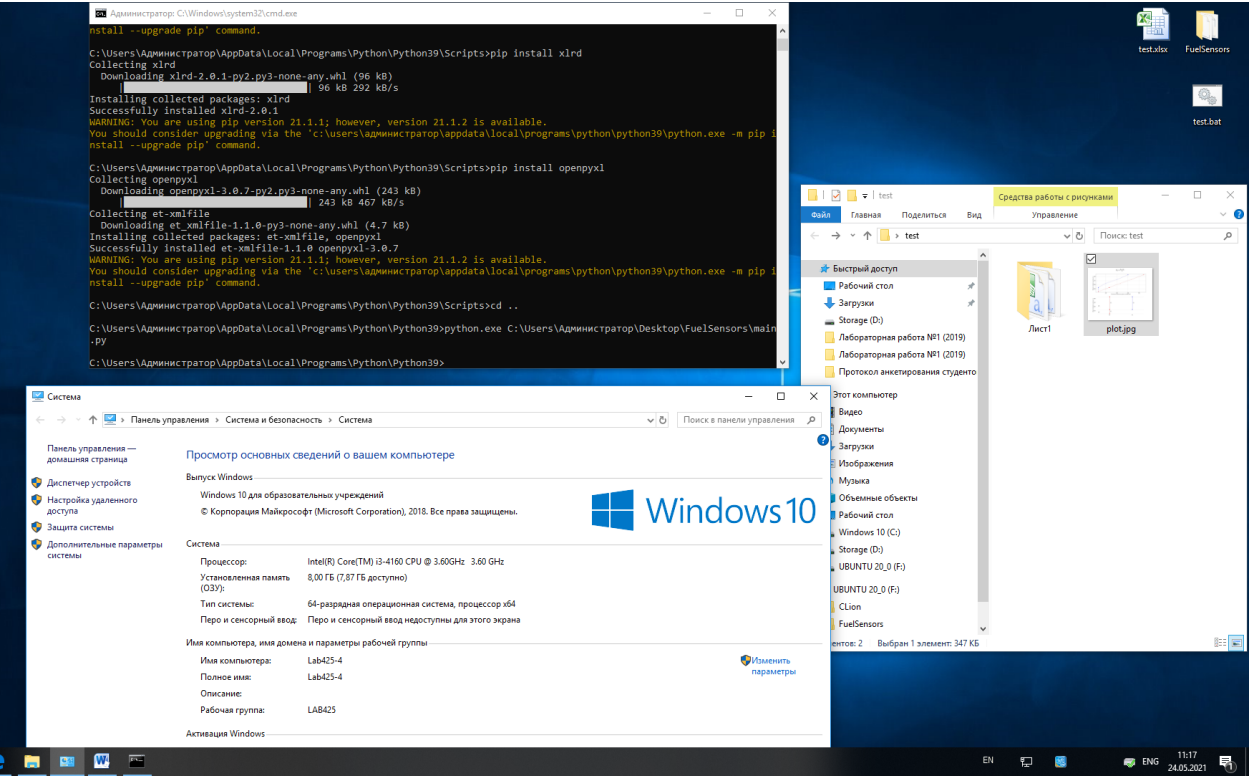


Рисунок 3.23 — Изменение bat-файла

Процедура работы с системой спутникового мониторинга

После того, как тарифовочная таблица была разбита на виртуальные датчики необходимо сделать следующее (на примере системы *wialon*):

- Выполнить поиск объекта в системе спутникового мониторинга и зайти в свойства объекта. И перейти вкладку «Датчики»

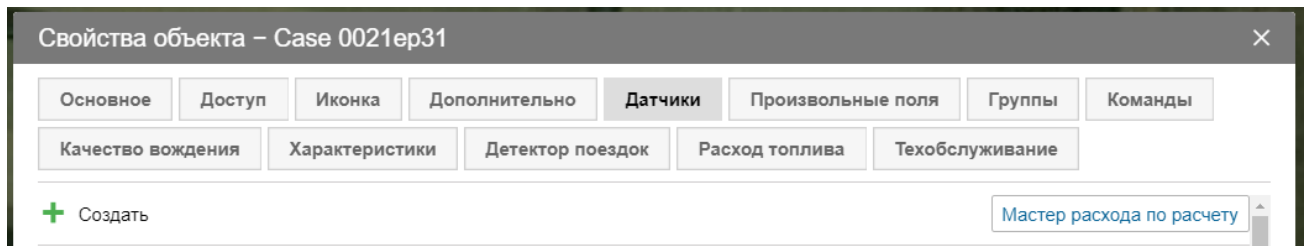


Рисунок 3.24 — Вкладка «Датчики» для объекта Case 0021ep31.

- В процессе установки ДУТов, тарифовки или непосредственно из сообщений можно узнать, какой параметр отвечает за тот или иной датчик. В нашем случае имеется два ДУТа: один подключен на cls1, другой – на cls2.
- Выполняется создание датчиков в системе. Для этого необходимо нажать на кнопку «Создать» и заполнить поля следующим образом:

Рисунок 3.25 — Создание датчика

В процессе создания датчика ему нужно задать имя. Рекомендуется использовать *DUT_1*, *DUT_2* и т. д. по количеству ДУТов. В примере имеем два датчика. Поэтому после выполнения указанных шагов должны получить следующее (рисунок 3.26):

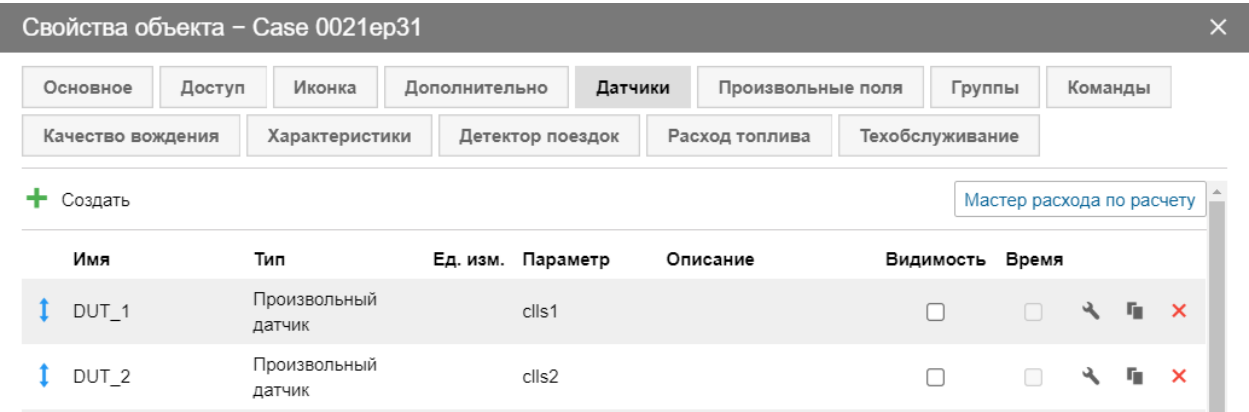


Рисунок 3.26 — Создание датчиков *DUT_1*, *DUT_2*

- Занести тарифовку в .xlsx-файл (рисунок 3.27):

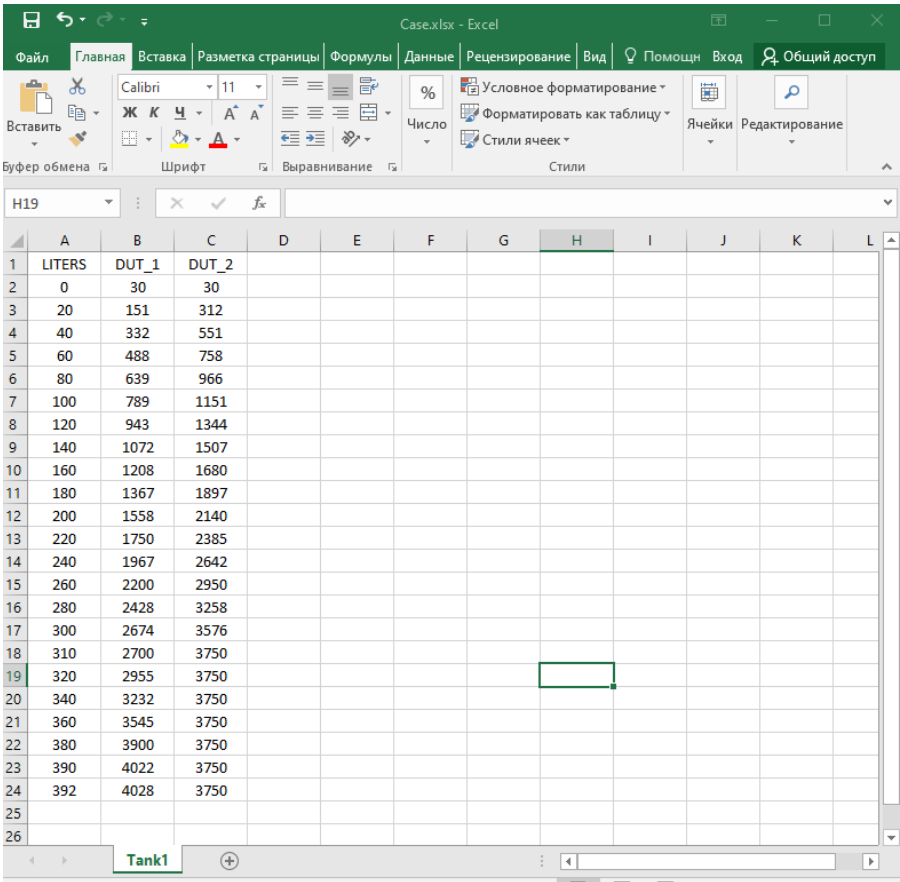


Рисунок 3.27 — Тарифовочная таблица для Case 0021ep31

- Выполнить запуск разработанного приложения, который выполнит запрос файла. Необходимо указать файл с тарифовочной таблицей.
- В той же директории, где и файл с тарифовкой должна появиться одноимённая папка со следующим содержимым (рисунок 3.28):

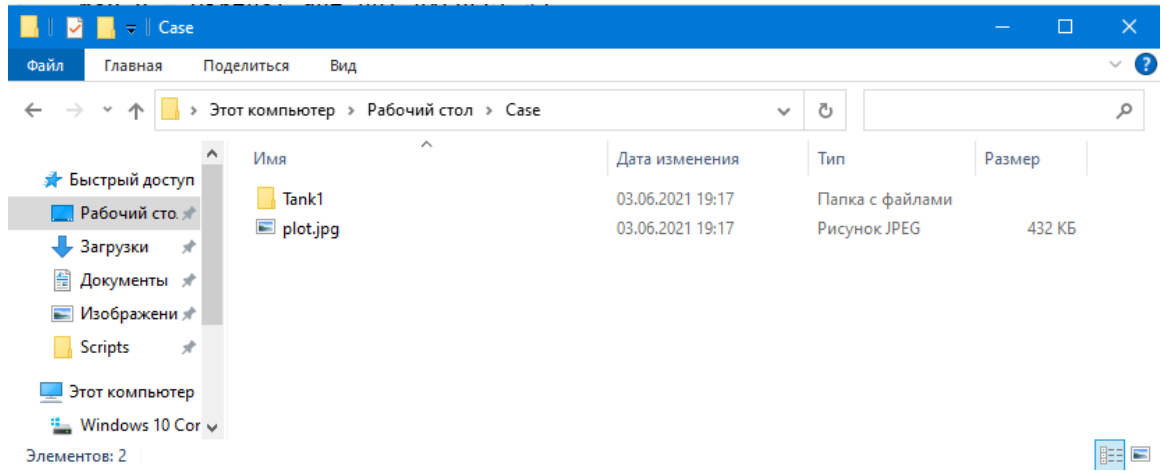


Рисунок 3.28 — Результат работы программы

При желании можно ознакомиться с получившимися функциональными зависимостями (рисунок 3.29):

$$\frac{V_1 + V_2}{2} + V_3$$

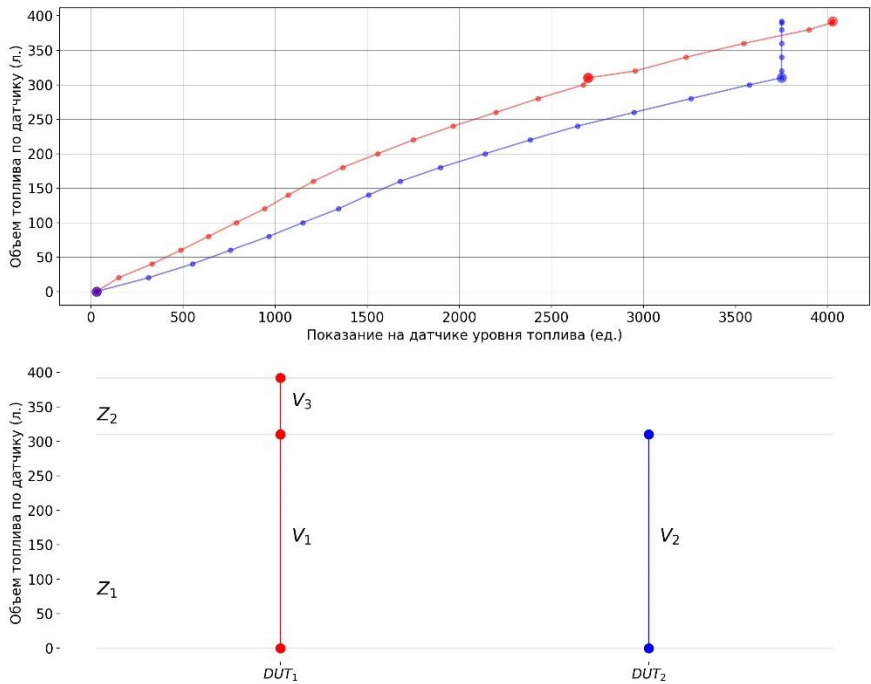


Рисунок 3.29 — Графики, построенные программой

- На следующем этапе необходимо создать виртуальные датчики. Исходя из графика, можно сказать, что имеется 3 виртуальных датчика. Опишем создание одного из них:

Свойства датчика — FUEL_1

Основное

Имя: * FUEL_1

Тип датчика: Произвольный датчик

Описание:

Параметр: * [DUT_1]

Система мер: Метрическая

Единица измерения:

Последнее сообщение: ☐

Таймаут, секунд: 0

Валидатор: Нет

Степень фильтрации (0...255): ☐

Текстовые параметры: ☐

Интервалы и цвета:

От Цвет Текст

+ Добавить интервал

Таблица расчета

Отмена ОК

Рисунок 3.30 — Создание датчика *FUEL_1*

Особое внимание следует уделить следующим моментам:

- Имя датчика: *FUEL_1* – имя датчика. Рекомендуется давать имена *FUEL_1, FUEL_2, ... FUEL_N*.
- В качестве параметра нужно указать имя физического ДУТа.
- На вкладке «Таблица расчета» необходимо выставить галочку в «Пары XY» а потом импортировать тарифовочную таблицу с виртуальным датчиком (рисунок 3.31).

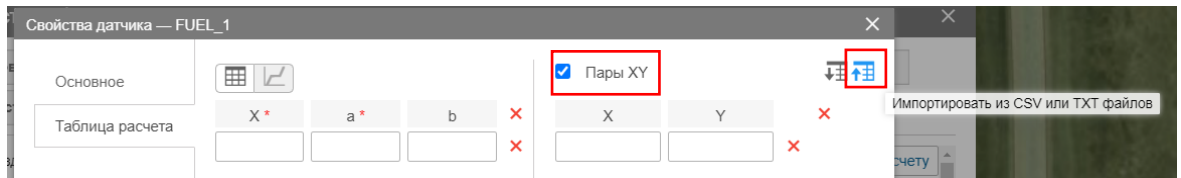


Рисунок 3.31 — Импортирование таблицы

- После импортирования необходимо нажать кнопку «Генерировать», после чего можно увидеть (рисунок 3.32):

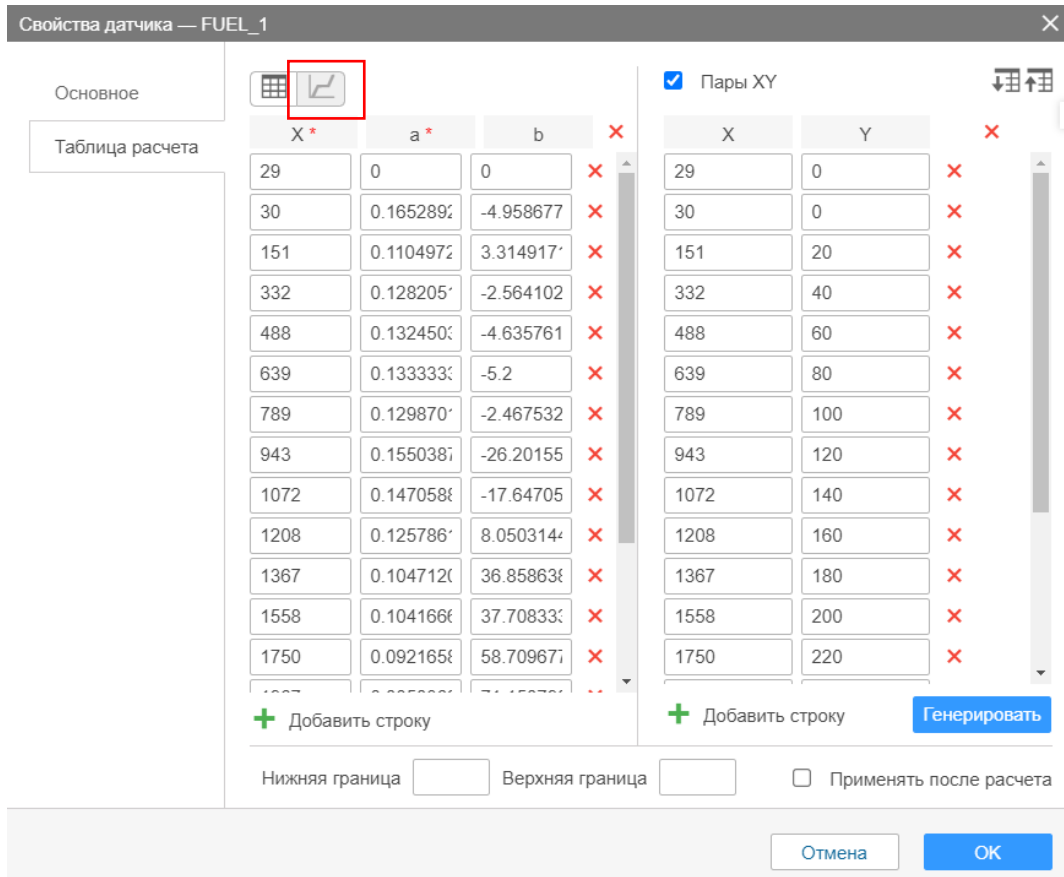


Рисунок 3.32 — Импортирование таблицы

- Если нажать на кнопку «Показать график» (рисунок 3.32), можно увидеть зависимость для виртуального ДУТа как и на рисунке 3.29, но до жирной красной точки. После неё идёт следующий виртуальный датчик уровня топлива.
- После занесения всех виртуальных датчиков, будем наблюдать следующую картину (рисунок 3.33):



Рисунок 3.33 — Заведенные виртуальные датчики уровня топлива

- На последнем этапе создаются ещё два датчика. Программа генерирует текстовый файл, из которого можно скопировать формулу для данной системы спутникового мониторинга и использовать в поле параметр (рисунок 3.34).

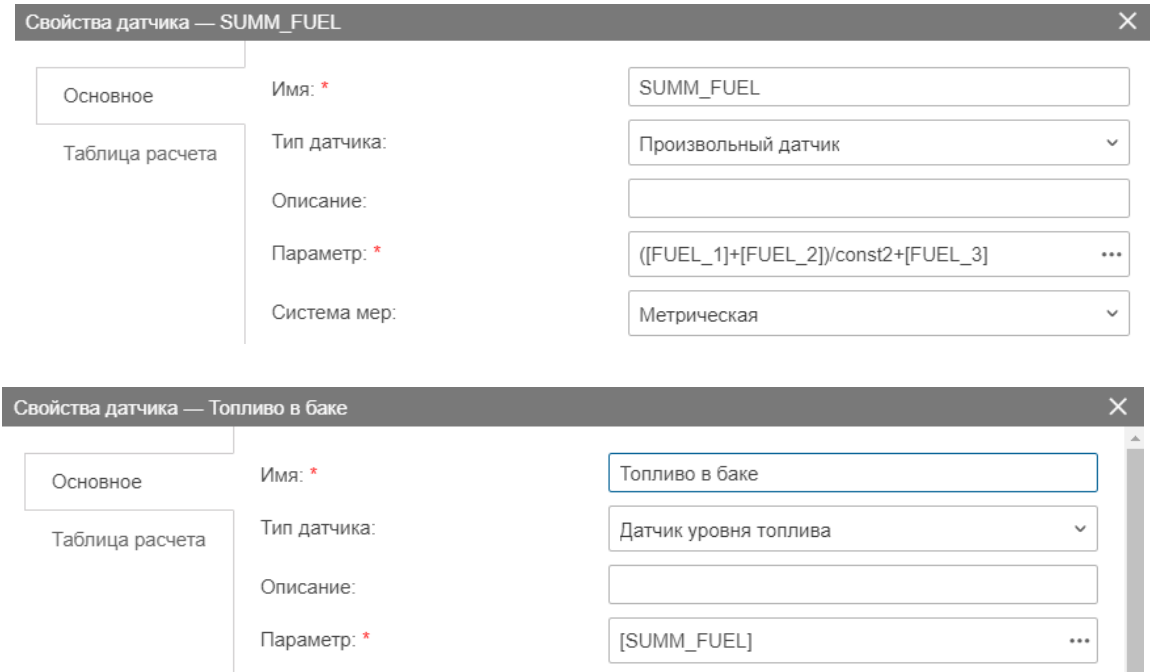


Рисунок 3.34 — Заведенные виртуальные датчики уровня топлива

Для датчика уровня топлива необходимо выставить также дополнительные настройки:

Общие настройки ^

Игнорировать сообщения после начала движения, секунд:

20

Рассчитывать расход топлива по датчику:

☒

Заменять ошибочные значения рассчитанными математически:

☐

Рассчитывать расход топлива по времени:

☒

Фильтровать значения датчиков уровня топлива:

☒

Степень фильтрации (0...255):

3

Рисунок 3.34 — Дополнительные настройки датчиков уровня топлива

Итоговая конфигурация представлена на рисунке 3.35:

Свойства объекта – Case 0021ер31

Основное

Доступ

Иконка

Дополнительно

Датчики

Произвольные поля

Группы

Команды

Качество вождения

Характеристики

Детектор поездок

Расход топлива

Техобслуживание

+ Создать

Мастер расхода по расчету

Имя	Тип	Ед. изм.	Параметр	Описание	Видимость	Время	
DUT_1	Произвольный датчик		clls1		<input type="checkbox"/>	<input type="checkbox"/>	
DUT_2	Произвольный датчик		clls2		<input type="checkbox"/>	<input type="checkbox"/>	
FUEL_1	Произвольный датчик		[DUT_1]		<input type="checkbox"/>	<input type="checkbox"/>	
FUEL_2	Произвольный датчик		[DUT_2]		<input type="checkbox"/>	<input type="checkbox"/>	
FUEL_3	Произвольный датчик		[DUT_1]		<input type="checkbox"/>	<input type="checkbox"/>	
SUMM_FUEL	Произвольный датчик		((FUEL_1)+[F...		<input type="checkbox"/>	<input type="checkbox"/>	

Рисунок 3.35 — Итоговая конфигурация

- После проведения настроек объекта рекомендуется выполнить построение отчёта по топливу с целью дополнительной проверки корректности настроек (фрагмент примера отчёта представлен на рисунке 3.36).



Рисунок 3.36 — Фрагмент отчёта

При правильно проведенной настройке объекта не должно наблюдаться резких скачков показаний уровня топлива, что мы и наблюдаем в данном случае.

Вывод

В третьем разделе представлена реализации алгоритма, приведены результаты тестирования, а также сведения по установке ПО, и эксплуатации результатов, вкупе с системой спутникового мониторинга Wialon.

Заключение

В рамках выпускной квалификационной работы по теме «Разработка и реализация алгоритма вычисления уровня топлива в баках сложной конфигурации» было выполнено исследование предметной области, описаны требования к алгоритму вычисления уровня топлива со стороны сервера.

В ходе работы был предложен алгоритм вычисления уровня топлива посредством разбиения физических датчиков уровня топлива на виртуальные (логические). Выработаны функциональные и нефункциональные требования к программному обеспечению, производящему разбиения, а также для построения графических зависимостей.

Представлена реализация алгоритма на языке программирования Python с использованием библиотек pandas, matplotlib, xlrd, openpyxl. Было проведено модульное, инсталляционное, эксплуатационное приёмочное тестирование разработанного продукта, которое показало корректность на различных исходных данных. Разработанное программное обеспечение внедрено в ООО «ЭкспертКом». Планируется дальнейшее использование разработанных алгоритмов и подходов для разработки станции для автоматизированного тарирования с передачей настроек на сервер.

Список литературы

Николаев Р.А., «Погрешность измерения уровня топлива, вызванная наклонами автомобиля,» *Информационные системы и измерительно-вычислительные комплексы. Сборник докладов студентов и аспирантов кафедры «Измерительно-вычислительные комплексы» на научно-технических конференциях.*, с. 44-48, 2010.

Навтелеком, *Руководство по эксплуатации SMART S-2433. Установка и подключение устройства*, Москва, 2020.

Omnicom, *Датчики уровня топлива Omnicom LLS 5 и LLS-Ex 5*, 2021.

Анашкин А.А., Акчурин Н.Г. и Чулючкин В.В., «Устройство для измерения уровня топлива». Патент RU 2445585 C1, 2012.

Мурашкина Т.И., Архипов А.В., Пивкин А.Г. и Серебряков Д.И., «Труды международного симпозиума "Надежность и качество",» в *Волоконно-оптическая система измерения уровня топлива*, Пенза, 2012.

Кузнецова Л.В., «Герконовый датчик для измерения уровня топлива в баке». Россия Патент RU 152949 U1, 2015.

Медведев А.Г., *Разработка и исследование поверхностных емкостных датчиков для измерения уровня топлива*, 2008.

Захаров Р.С., Скворцов Б.В. и Таипова Д.Р., «Физико-математическая модель емкостного метода измерения уровня топлива в баках ракет-носителей с учётом полей рассеивания».

Муравьёв С.А. и Новиков С.В., «7-я международная научно-техническая конференция "К. Э. Циолковский - 160 лет со дня рождения". Космонавтика. Радиоэлектроника. Геоинформатика.,» в *Повышение*

точности измерений уровня заправки жидкостных компонентов ракетного топлива.

Куликов С.С., Тестирование программного обеспечения, Минск: Четыре четверти, 2017.

Karl E. Wieggers, Software Requirements, Москва: Русская редакция, 2004.

Wes McKinney and the Pandas Development Team, *Pandas: powerful Python data analysis*, 2021.

John Hunter, Darren Dale, Eric Firing, Michael Droettboom и the matplotlib development team, *Matplotlib*, 2021.

Stephen John Machin, "Xlrd," [Online]. Available: <https://xlrd.readthedocs.io/en/latest/>. [Accessed 23 May 2021].

Eric Gazoni и Charlie Clark, «A Python library to read/write Excel 2010 xlsx/xlsm files,» 09 March 2021. [В Интернете]. Available: <https://openpyxl.readthedocs.io/en/stable/>. [Дата обращения: 23 May 2021].