МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Электронные вычислительные машины»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«___» _____ 2023 г.

Разработка программы для автоматизации измерения и расчета геометрических характеристик металлических трубок

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУРГУ-090301.2023.562 ПЗ ВКР

Руководитель работы,
к.пед.н., доцент каф. ЭВМ
_____ Ю.Г. Плаксина
«___» _____ 2023 г.

Автор работы,
студент группы КЭ-406
_____В.П. Карякин
«___» _____ 2023 г.

Нормоконтролёр,
ст. преп. каф. ЭВМ
_____С.В. Сяськов
«___» _____ 2023 г.

Консультант,
вед. инженер-конструктор
_____Е.А. Горюнов
«___» _____ 2023 г.

Челябинск-2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра «Электронные вычислительные машины»

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_____ Д.В. Топольский
«___» _____ 2023 г.

ЗАДАНИЕ

на выпускную квалификационную работу бакалавра

студенту группы КЭ-406
Карякину Владимиру Павловичу,
обучающемуся по направлению
09.03.01 «Информатика и вычислительная техника»

1. **Тема работы: «**Разработка программы для автоматизации измерения и расчета геометрических характеристик металлических трубок» утверждена приказом по университету №753-13/12 от 25 апреля 2023 г.

2. **Срок сдачи студентом законченной работы:** 01 июня 2023 г.

3. **Исходные данные к работе:**

3.1. Программа расчета геометрических характеристик трубок предназначена для сопровождения производственного цикла трубок массовых кориолисовых расходомеров ЭЛМЕТРО-Фломак.

3.2. Программа должно обеспечивать выполнение следующего спектра задач:

– Проведение оптического измерения трубок типоразмера S003 и сохранение полученных значений в базе данных.

– Проведение расчета геометрии трубок согласно данным оптического измерения.

3.3. Программа должна работать в операционной системе Windows 7 и выше.

3.4. Хранение данных должно быть осуществлено в базе данных.

3.5. Система управления базой данных MySQL версии 5.5.62.

3.6. Доступ к базе данных должен быть обеспечен с любого рабочего места.

3.7. Математический аппарат и классы, используемые для расчета характеристик трубок и работы с базой данных, должны быть выделены в отдельные модули или динамические библиотеки.

3.8. Классы данных и их обработки должны быть независимы от интерфейсной части и пригодны для использования в других проектах.

3.9. Архитектура программы должна быть оптимизирована для возможности глубокой модернизации:

– Изменения набора измеряемых параметров.

– Изменения последовательности потока данных.

– Изменения математической модели расчета.

3.10. Сценарии использования и архитектура программы должны быть согласованы с заказчиком предварительно, до непосредственной разработки программного кода.

4. Перечень подлежащих разработке вопросов:

– Аналитический обзор научно-технической, нормативной и методической литературы по тематике работы.

– Разработка архитектуры программы.

– Разработка набора алгоритмов для работы машинного зрения.

– Разработка алгоритма расчета геометрических характеристик металлических трубок.

– Реализация, тестирование и отладка программы.

5. Дата выдачи задания: 2 декабря 2022 г.

Руководитель работы _____/ *Ю.Г. Плаксина*

Студент _____/ В.П. Карякин

КАЛЕНДАРНЫЙ ПЛАН

Этап	Срок сдачи	Подпись руководителя
Введение и обзор литературы.	01.03.2023	
Разработка архитектуры программы.	24.03.2023	
Разработка набора алгоритмов для работы машинного зрения.	14.04.2023	
Разработка алгоритма расчета геометрических характеристик металлических трубок.	28.04.2023	
Реализация, тестирование и отладка программы.	12.05.2023	
Компоновка текста работы и сдача на нормоконтроль.	22.05.2023	
Подготовка презентации и доклада.	30.05.2023	

Руководитель работы _____/ *Ю.Г. Плаксина*

Студент _____/ В.П. Карякин

АННОТАЦИЯ

В.П. Карякин. Разработка программы для автоматизации измерения и расчета геометрических характеристик металлических трубок. – Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ВШ ЭКН; 2023, 62 с., 21 ил., библиогр. список - 26 наим.

В рамках выпускной квалификационной работы была разработана программа для автоматизированного измерения и расчета геометрических характеристик металлических трубок. Программа реализована на языке программирования C# и позволяет проводить автоматизированные измерения металлических трубок с помощью измерительной оснастки, электронного микроскопа и алгоритмов машинного зрения, а также преобразовывать полученные результаты измерений в необходимые Заказчику геометрические характеристики с выгрузкой в удаленную базу данных, расположенную на сервере Заказчика. Разработанная программа в полной мере отвечает требованиям заказчика.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Для производства качественного метрологического оборудования необходимо как можно точнее знать характеристики его составляющих частей, поэтому обеспечение надёжности процесса измерения, расчетов и хранения данных о результатах расчетов является важным условием для обеспечения качества выпускаемых приборов.

Существующий технологический процесс измерения и расчетов характеристик трубок на предприятии Заказчика выглядит следующим образом:

1. Измерение проводится на специальной оснастке для измерения. Вне зависимости от типа трубки, общий принцип измерения заключается в том, что трубка устанавливается в специальную оснастку для измерения, геометрия которой известна с достаточной степенью точности, и которая не влияет на геометрию трубки.

2. Поскольку геометрия оснастки выполнена с достаточной точностью, измерение трубки проводится при помощи измерения необходимого и достаточного количества расстояний от базовых объектов на оснастке до трубки. После этого по измеренным расстояниям рассчитываются геометрические параметры, которые в дальнейшем используются в других бизнес-процессах.

3. Пересчет размеров в настоящий момент ведется в приложении SolidWorks в силу особенностей механизма пересчета. Для этого необходимо подготовить Excel файл заданного формата, который отличается для каждого типоразмера. После обработки в SolidWorks в программу вводятся характерные размеры трубки, уникальные для каждого типоразмера. В программе строится специальная параметрическая модель пересчета, которая восстанавливает геометрию средней линии трубки по результатам измерений.

Существующий технологический процесс измерения и расчетов характеристик трубок на предприятии Заказчика не удовлетворяет необходимому уровню производительности и качества. Данная выпускная квалификационная работа посвящена устранению этой проблемы.

Целью работы является разработка программы для частичной автоматизации процессов по измерению и расчету геометрических характеристик металлических трубок с привлечением современных технологических решений, таких как машинное зрение.

В соответствии с целью были сформированы следующие задачи:

1. Рассмотреть существующие на данный момент аналоги, выполнить обзор литературы по теме работы.
2. Разработать архитектуру программы.
3. Просмотреть существующие алгоритмы работы машинного зрения, определить среди них необходимые для корректного проведения измерений трубок, объединить в единый алгоритм обработки.
4. Разработать алгоритм расчета геометрических характеристик трубок по результатам измерений.
5. Реализовать программу, выполнить её тестирование и отладку.

1 АНАЛИТИЧЕСКИЙ ОБЗОР НАУЧНО-ТЕХНИЧЕСКОЙ, НОРМАТИВНОЙ И МЕТОДИЧЕСКОЙ ЛИТЕРАТУРЫ ПО ТЕМАТИКЕ РАБОТЫ

1.1 Обзор аналогов

1.1.1 Автоматизированная система серии ГЕОМЕР

Автоматизированная система серии ГЕОМЕР [1] производства АктивТестГруп предназначена для автоматического измерения геометрических параметров прямошовных электросварных магистральных труб в условиях производства. Измерительный стенд системы представлен на рисунке 1.

Система размещается на рольганге с парой подъемно-поворотных роликов, использует бесконтактный, лазерный, триангуляционный метод измерения и реализует измерение большого количества параметров, описанные в таблице 1 [2].

Рисунок 1 – Измерительный стенд ГЕОМЕР

Таблица 1 - Основные характеристики установки ГЕОМЕР

Характеристика	Значение
Диаметр контролируемых труб, мм	По требованию Заказчика
Длина контролируемых труб, мм	
Время контроля одной трубы максимального диаметра, мин	Не более 5

Продолжение таблицы 1

Точность позиционирования трубы относительно манипулятора, мм	Не хуже 5
Точность позиционирования манипуляторов, мм	0,04
Точность измерения параметров	
— линейных (кроме длины трубы), мм	0,1
— угловых, град.	0,5
— длины трубы, мм	2
Перенастройка на другой типоразмер	Автоматическая
Давление в воздушной магистрали	До 4 МПа
Время перенастройки на другой типоразмер трубы, сек.	2
Потребляемая мощность установки, кВА	Не более 10
Габариты установки с кабиной затемнения, (Ш × Д × В) мм	3000×14200×4500
Вес установки, кг	Не более 3 000
Напряжение питания, В	220 / 380
Частота тока в сети, Гц	50
Диапазон рабочих температур	От +10 до +35 °С

Данное решение не удовлетворяет требованиям заказчика из-за следующих факторов:

1. Точность позиционирования манипуляторов.
2. Точность измерения параметров.
3. Габариты и вес установки.

1.1.2 Система для контроля труб TubeShaper

Система для контроля труб TubeShaper представляет собой набор инструментов, который позволяет контролировать геометрические характеристики труб любого назначения [3]. Внешний вид системы представлен на рисунке 2.

Основные возможности и характеристики системы в зависимости от конкретной модели представлены в таблице 2.

Система позволяет:

1. Определять длины труб, углы изгибов, измерить и ввести поправки на остаточные упругие деформации.

2. Контролировать правильность гибки трубы, сравнивая с ее компьютерной моделью либо с таблицей номинальных параметров.

3. Измерять и учитывать поправки на остаточные упругие деформации.

4. Экспортировать измеренные элементы в формате IGES.

Рисунок 2 – Система для контроля труб TubeShaper с Hexagon Absolute Arm [4]

Таблица 2 - Основные характеристики установки TubeShaper

Модель	7320	7325	7330	7335	7340	7345
Диапазон измерений	2.0 м	2.5 м	3.0 м	3.5 м	4.0 м	4.5 м
Повторяемость	0.030 мм	0.038 мм	0.059 мм	0.079 мм	0.099 мм	0.120 мм

Продолжение таблицы 2

Пространственная точность	± 0.042 мм.	± 0.051 мм	± 0.075 мм	± 0.100 мм	± 0.125 мм	0.150 мм
Вес	7.4 кг	7.7 кг	8.0 кг	8.3 кг	8.6 кг	8.9 кг

Данное решение не удовлетворяет требованиям заказчика из-за следующих факторов:

1. Отсутствует поддержка типоразмеров трубок на 2 мм.
2. Слишком высокая погрешность измерения (от 0,03 мм).
3. Отсутствие возможностей точного измерения на участках гиба трубок.

1.1.3 УСКТ-8 многоканальная система контроля тонких труб

Установка УСКТ-8 предназначена для неразрушающего ультразвукового контроля труб на наличие дефектов (обнаружение дефектов) типа нарушения сплошности и однородности металла, измерения толщины стенки, наружного диаметра на металлургических и машиностроительных предприятиях [5]. Внешний вид установки представлен на рисунке 3.

Установка содержит четыре канала дефектоскопии, два канала совместного измерения толщины стенки, наружного и внутреннего диаметра и один опорный канал для компенсации температурного изменения скорости распространения ультразвуковых колебаний (УЗК) в иммерсионной жидкости при измерении диаметра [6]. Подробные технические характеристики установки представлены в таблице 3.

Рисунок 3 – Измерительный стенд УСКТ-8

Таблица 3- Основные характеристики установки УСКТ-8

Характеристика	Значение
Наружный диаметр труб	5 - 35 мм
Толщина стенки	от 0,2 до 2,5 мм
Длина труб	от 1,5 до 6 м
Шероховатость поверхности Ra	не более 2,5
Количество каналов контроля	8
Усилитель	широкополосный 1,5-15 МГц
Допустимые колебания скорости трубы при контроле	5%
Время настройки и калибровки установки при смене диаметра	не более 30 мин
Длина неконтролируемых концов труб	не более 30мм
Абсолютная погрешность измерения наружного диаметра	±1 %

Продолжение таблицы 3

Абсолютная погрешность измерения толщины стенки	±1 %
Погрешность измерения координат по длине трубы	± 15 мм
Частота вращения трубы	50-1500 об/мин
Диапазон регулировки шага контроля	0-5 мм/об

Данное решение не удовлетворяет требованиям заказчика из-за следующих факторов:

1. Отсутствует поддержка типоразмеров трубок на 2 мм и, как соответствие, высокая абсолютная погрешность измерения диаметра.

2. Отсутствует возможность измерения участков сгиба.

3. Габариты и вес установки.

1.1.4 КИМ Faro Fusion Arm 8

Комплекс Faro Fusion Arm в комплекте с программным обеспечением TezetCAD [7] позволяет проводить:

1. Контроль геометрии трубопроводов различной пространственной формы, в том числе сборок трубопроводов, в составе которых могут быть компенсирующие и демпфирующие элементы.

2. Корректировку геометрии трубопровода с учетом фактических отклонений.

3. Автоматический расчет и корректировку управляющей программы для трубогибочного автомата.

Данное решение не удовлетворяет требованиям заказчика из-за следующих факторов:

1. Высокие требования к точности позиционирования манипулятора измерительной руки;

2. Недостаточное разрешение устройства захвата видеопотока, следствием чего является несоответствие требованиям точности.

1.1.5 Программное обеспечение «MCView»

Программа «MCView» предназначена для захвата видеопотока из устройства ввода (видеокамера, веб-камера, электронный микроскоп) и его дальнейшей визуальной обработки оператором с использованием различных встроенных инструментов [8]. Пользовательский интерфейс программы представлен на рисунке 4.

Данная программа используется для реализации существующей бизнес-логики на предприятии Заказчика в настоящий момент.

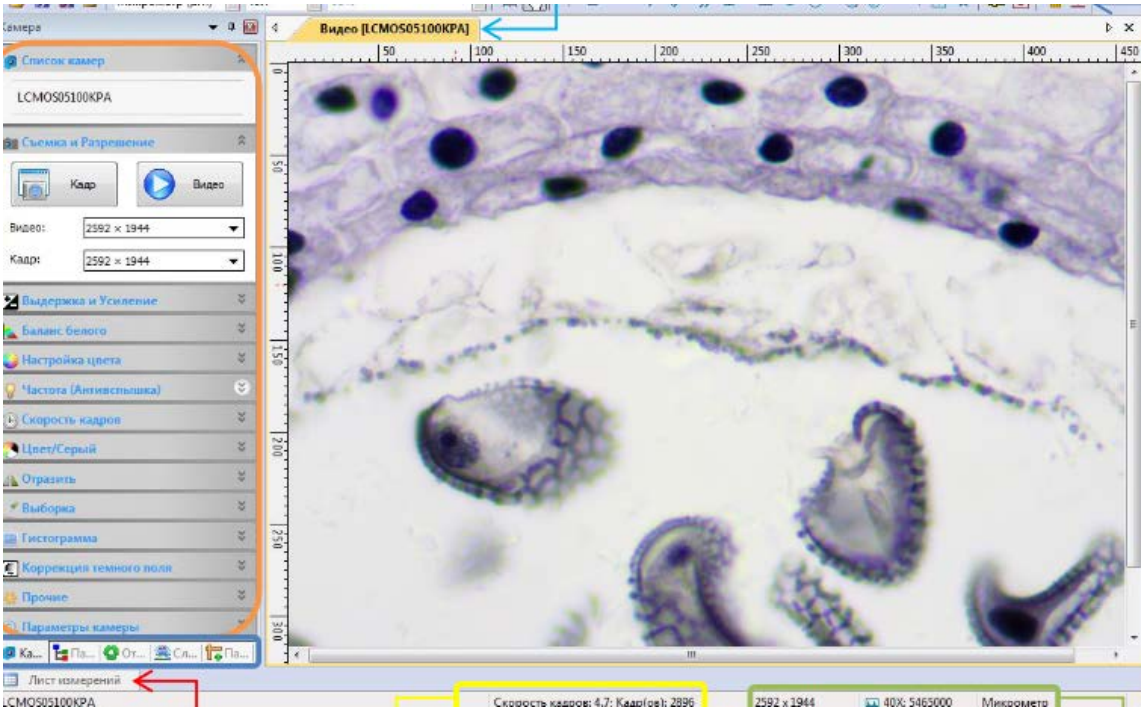

Рисунок 4 – Пользовательский интерфейс программы MCView

Недостатками данного программного обеспечения являются:

1. Высокие требования к производительности электронных вычислительных машин, на которые установлена данная программа для ее корректной работоспособности (в зависимости от количества данных, получаемых с устройства ввода).

2. Отсутствие возможности автоматической выгрузки измеренных данных в какие-либо базы данных.

3. Отсутствие возможности модернизации ПО для автоматизации бизнес-процессов.

1.1.6 Программное обеспечение «SOLIDWORKS»

SOLIDWORKS – программное обеспечение, позволяющее решить целый комплекс задач, возникающих на производстве. Обширный инструментарий SOLIDWORKS позволяет не только сделать общий дизайн модели, но и провести детальное тестирование в виртуальной среде, доработать отдельные узлы, после чего конвертировать модель в чертежи для создания реального прототипа [9].

Данное программное обеспечение используется для реализации существующей бизнес-логики на предприятии Заказчика в настоящий момент.

Недостатками данного программного обеспечения являются:

1. Низкая скорость начальной загрузки программы для обработки данных.
2. Высокие требования к производительности компьютера.
3. Высокая сложность разработки алгоритмов для расчета параметров.
4. Отсутствие возможности автоматической выгрузки рассчитанных параметров в базу данных.
5. Нахождение в списке программного обеспечения, подлежащего импортозамещению.

1.1.7 Программное обеспечение «Компас-3D»

Основная задача, решаемая системой КОМПАС-3D — моделирование изделий с целью существенного сокращения периода проектирования и скорейшего их запуска в производство [10]. Эти цели достигаются благодаря возможности:

1. Быстрого получения конструкторской и технологической документации, необходимой для выпуска изделий.
2. Передачи геометрии изделий в расчетные пакеты.
3. Передачи геометрии изделий в пакеты разработки управляющих программ для оборудования с ЧПУ.
4. Создания дополнительных изображений изделий (например, для составления каталогов, создания иллюстраций к технической документации и т.д.).

Недостатком данного программного обеспечения является отсутствие возможности автоматизации бизнес-процессов путем выгрузки результирующих данных в указанную базу данных в необходимом Заказчику формате [11].

1.2 Обзор основных технических решений

1.2.1 Анализ и выбор языка разработки

В качестве языка программирования для разработки был выбран язык C#, поскольку это является требованием Заказчика.

Для разработки приложения Windows с пользовательским на данный момент в основном используются такие технологии, как Windows Forms (WinForms) [12] и Windows Presentation Foundation (WPF) [13].

Для разработки программы была выбрана технология Windows Forms, так как она является популярным интерфейсом программирования приложений [14], созданным специально для написания программного обеспечения для Windows. Windows Forms — это технология интеллектуальных клиентов для .NET Framework. Она представляет собой набор управляемых библиотек, упрощающих выполнение стандартных задач, таких как чтение из файловой системы и запись в нее.

Технология WPF для применения в разработке не рассматривалась по причине сильной зависимости от стороннего программного обеспечения, такого как DirectX.

1.2.2 Выбор среды разработки

Для разработки программного обеспечения на языке C# в настоящее время используются такие программы, как Microsoft Visual Studio и Microsoft Visual Studio Code, которая является «облегченной» версией Microsoft Visual Studio и в своем базовом варианте не поддерживает компилирование кода.

В качестве среды разработки была выбрана IDE Microsoft Visual Studio 2022.

1.2.3 Выбор системы управления базой данных

В качестве СУБД была выбрана СУБД MySQL версии 5.5.62, т.к. это является требованием Заказчика.

1.2.4 Выбор библиотеки для работы с алгоритмами машинного зрения

В настоящее время для использования в проектах, разработанных на языке C#, наиболее распространенными и документированными библиотеками для работы с машинным зрением являются такие библиотеки, как AForge.NET [15],

OpenCVSharp [16], Accord.NET [17]. Данные библиотеки распространяются под лицензиями LGPL или Apache, что позволяет их коммерческое использование без дополнительных финансовых затрат на лицензирование.

Для разработки приложения была выбрана библиотека AForge.NET, поскольку при тестировании функционала остальных библиотек были выявлены недоработки, препятствующие их применению на производстве. Так, библиотека OpenCVSharp не позволяет обрабатывать видеопоток разрешением свыше 1280x720, что недопустимо в данной разработке. Библиотека Accord.NET по сути является ответвлением от библиотеки AForge.NET и привносит очень узкоспециализированный функционал.

Библиотека AForge.NET в настоящее время завершила разработку, что позволит избежать ошибок при дальнейшей модификации программы, связанных с изменением функционала библиотеки.

1.3 Выводы по разделу 1

Поскольку широко распространенные решения не удовлетворяют условиям, требуемым для автоматизации существующих бизнес-процессов, и/или не удовлетворяют требованиям к качеству выполняемой работы, Заказчиком было принято решение о создании собственной программы и внедрении в данную программу всех необходимых Заказчику функций с расчетом на её постепенную модернизацию в ходе развития логики бизнес-процессов. Для этого были проанализированы и определены требования к техническим решениям, применяемым при создании программы.

2 РАЗРАБОТКА АРХИТЕКТУРЫ ПРОГРАММЫ

2.1 Общие сведения

Программа для автоматизации расчета геометрических характеристик металлических трубок состоит из следующих частей:

1. Клиентская часть, представляющая собой приложение, обеспечивающее интерфейс для проведения измерений и расчета характеристик.

2. Серверная часть, представляющая собой таблицу в базе данных, хранящую результаты измерений и расчетных характеристик, а также набор сохраненных функций для обеспечения работы.

2.2 Клиентская часть

Клиентское приложение состоит из следующих частей:

1. Пользовательский интерфейс.

1.1. Главное окно приложения. Макет представлен на рисунке 5.

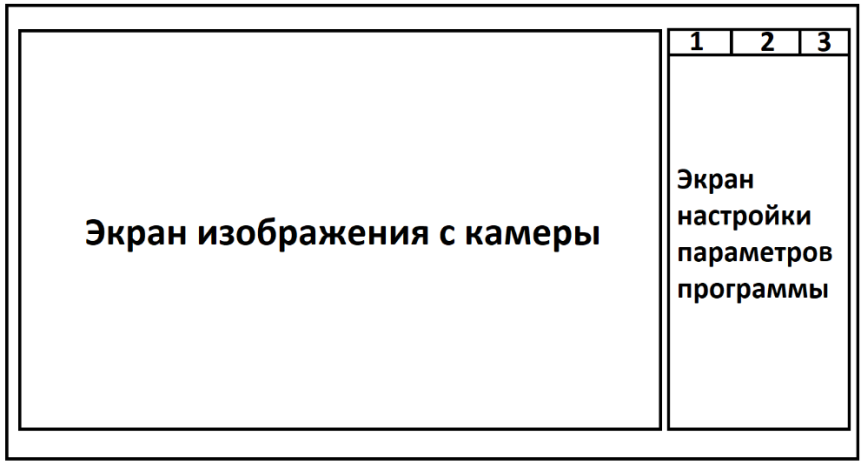

Рисунок 5 – Макет главного экрана программы

1.1.1. Окно с изображением, полученным с измерительного устройства (электронный микроскоп).

1.1.2. Панель инструментов с разделением по вкладкам:

1.1.2.1. Вкладка «Калибровка»:

1.1.2.1.1. Список подключенных устройств ввода видеопотока для выбора измерительного устройства.

1.1.2.1.2. Список всех доступных разрешений видеопотока для выбранного устройства с возможностью выбора разрешения.

1.1.2.1.3. Кнопка для начала калибровки устройства ввода видеопотока.

1.1.2.1.4. Поле с результатом калибровки.

1.1.2.1.5. Кнопка вызова интерфейса настройки параметров сервера для выгрузки результатов.

1.1.2.1.6. Кнопка для вызова окна просмотра измеренных данных.

1.1.2.1.7. Еще какие-либо элементы, которые в сессии работы приложения используются редко.

1.1.2.2. Вкладка «Измерение»:

1.1.2.2.1. Поле для просмотра и редактирования названия измеряемой трубки.

1.1.2.2.2. Поле для просмотра и редактирования текущей измеряемой точки данной трубки.

1.1.2.2.3. Еще какие-либо элементы, которые в сессии работы приложения используются часто.

1.1.2.3. Вкладка «Машинное зрение»:

1.1.2.3.1. Шкалы для настройки параметров распознавания элементов на изображении, полученным из видеопотока.

1.1.2.3.2. Еще какие-либо элементы, которые зависят от свойств выбранного алгоритма работы машинного зрения.

1.2. Интерфейс параметров сервера. Макет представлен на рисунке 6.

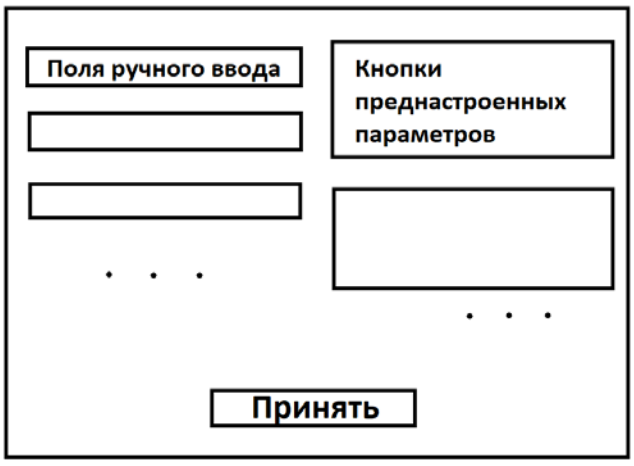

Рисунок 6 – Макет окна настройки параметров сервера

1.2.1. Поля для пользовательского ввода следующих параметров:

1.2.1.1. Адрес сервера.

1.2.1.2. Порт подключения.

1.2.1.3. Имя пользователя.

1.2.1.4. Пароль.

1.2.1.5. Название используемой базы данных.

1.2.2. Кнопки для использования предварительно настроенных параметров сервера, а также сохраненных пользователем конфигураций.

1.2.3. Кнопка для применения текущих параметров сервера.

1.2.4. Кнопка для сохранения введенной конфигурации.

1.3. Интерфейс просмотра введенных данных. Макет представлен на рисунке 7.

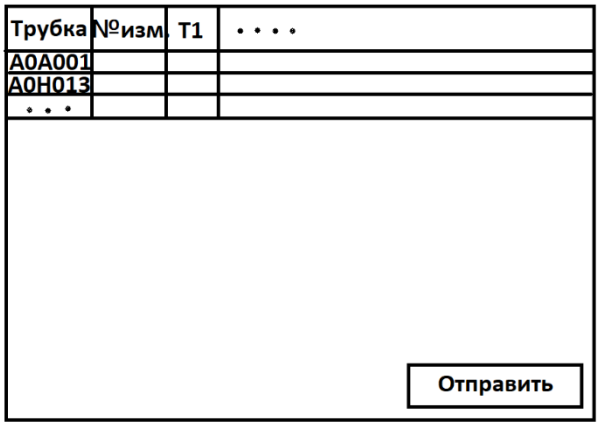

Рисунок 7 – Макет окна просмотра введенных данных

1.3.1. Таблица с результатами измерений.

1.3.2. Кнопка для начала расчета характеристик.

1.3.3. Таблица с результатами расчета.

1.3.4. Кнопка для выгрузки результатов измерений и расчета на сервер.

2. Набор функций для работы алгоритмов машинного зрения.

2.1. Функция для определения координат точки на оснастке, от которой должно проводится измерение.

2.2. Функция для определения уравнения прямой, принадлежащей измеряемой трубке, до которой должно производиться измерение от ранее определенной точки.

3. Набор функций для проведения расчета геометрических характеристик трубок на основе результатов измерения расстояния от определенных точек измерительной оснастки до трубки.

3.1. Функция для определения прямых базового профиля оснастки.

3.2. Функция для определения положения прямолинейных участков оси трубки.

3.3. Функция для определения радиусов гиба по осевой линии трубки.

3.4. Функция для определения биссектрисы.

3.5. Функция для определения точки пересечения двух прямых.

3.6. Функция для определения точки на биссектрисе, находящейся на расстоянии h от точки начала биссектрисы.

3.7. Функция для определения линии, соединяющей точки перемычек.

3.8. Функция для определения угла между двумя пересекающимися прямыми на плоскости.

3.9. Функция для определения расстояния между двумя точками.

3.10. Функция для определения расстояния от точки до прямой.

3.11. Функция проверки правильности измерения.

4. Набор функций для работы приложения с серверной частью.

4.1. Функция установления и проверки соединения с сервером.

4.2. Функция отправки данных на сервер.

2.3 Серверная часть

Серверная часть реализуется для сервера, работающего под управлением системы управления базами данных MySQL версии 5.5.62. Серверная часть состоит из следующих компонентов:

1. Таблица с результатами измерений и расчета характеристик, которая состоит из следующих компонентов:

1.1. Колонка с названием трубки.

1.2. Колонка с порядковым номером измерения для трубки.

1.3. Колонки для каждой точки трубки с результатами измерения расстояния от измерительной оснастки до трубки.

1.4. Колонки для каждого геометрического параметра трубки, полученного в результате расчета.

1.5. Колонка с датой и временем проведения измерения.

2. Сохраненные процедуры:

2.1. Процедура для записи результатов измерения и расчета характеристик.

2.2. Процедура для проверки наличия у трубки результатов измерений длины, массы и среднего диаметра из других таблиц базы данных.

2.4 Работа с программой

Работа оператора с программой следующим образом:

1. Открывается окно приложения.

2. Определяется устройство получения изображения (микроскоп), при этом автоматически выставляется максимально допустимое разрешение видеопотока.

3. Производится калибровка микроскопа.

4. Для каждой измеряемой трубки выполняются следующие этапы:

4.1. Пользователь вводит название измеряемой трубки.

4.2. Для каждой точки измерения:

4.2.1. Оператор направляет микроскоп на область измерения.

4.2.2. Программа автоматически определяет точку на измерительной оснастке, от которой должно производиться измерение расстояния, и прямую на трубке, до которой должно производиться измерение, и определяет расстояние от точки до прямой.

4.2.3. Оператор проверяет корректность определенных данных и нажимает кнопку подтверждения для перехода к следующей точке измерения.

5. По завершению всех измерений оператор открывает окно просмотра результатов измерений, проверяет наличие всех необходимых измерений, после чего нажимает кнопку начала расчета геометрических характеристик трубок.

6. По завершению расчета оператор нажимает кнопку выгрузки результатов расчета на сервер.

Визуализация программной архитектуры программы отображена на рисунке 8.

Рисунок 8 – Программная архитектура программы

2.5 Вывод по разделу 2

Результатом проведенной работы является архитектура программы для расчета геометрических характеристик металлических трубок. Разработана архитектура для клиентской и серверной частей, определены необходимые для реализации элементы пользовательского интерфейса программы, а также определен набор функций для соответствия работы программы бизнес-логике.

3 РАЗРАБОТКА НАБОРА АЛГОРИТМОВ ДЛЯ РАБОТЫ МАШИННОГО ЗРЕНИЯ

3.1 Общая информация

Использование алгоритмов машинного зрения в разрабатываемой программе позволит решать следующие задачи:

1. Производить определение координат точек, принадлежащих измерительной оснастке, от которой производится измерение. Для решения данной задачи были выбраны следующие алгоритмы:

1.1. Фильтрация изображения по цвету с целью определения области, в которой находится отметка точки начала измерения [18]. Для работы фильтра задаются пороговые значения для каждого из каналов R, G, B. Пример результата работы изображен на рисунке 9.

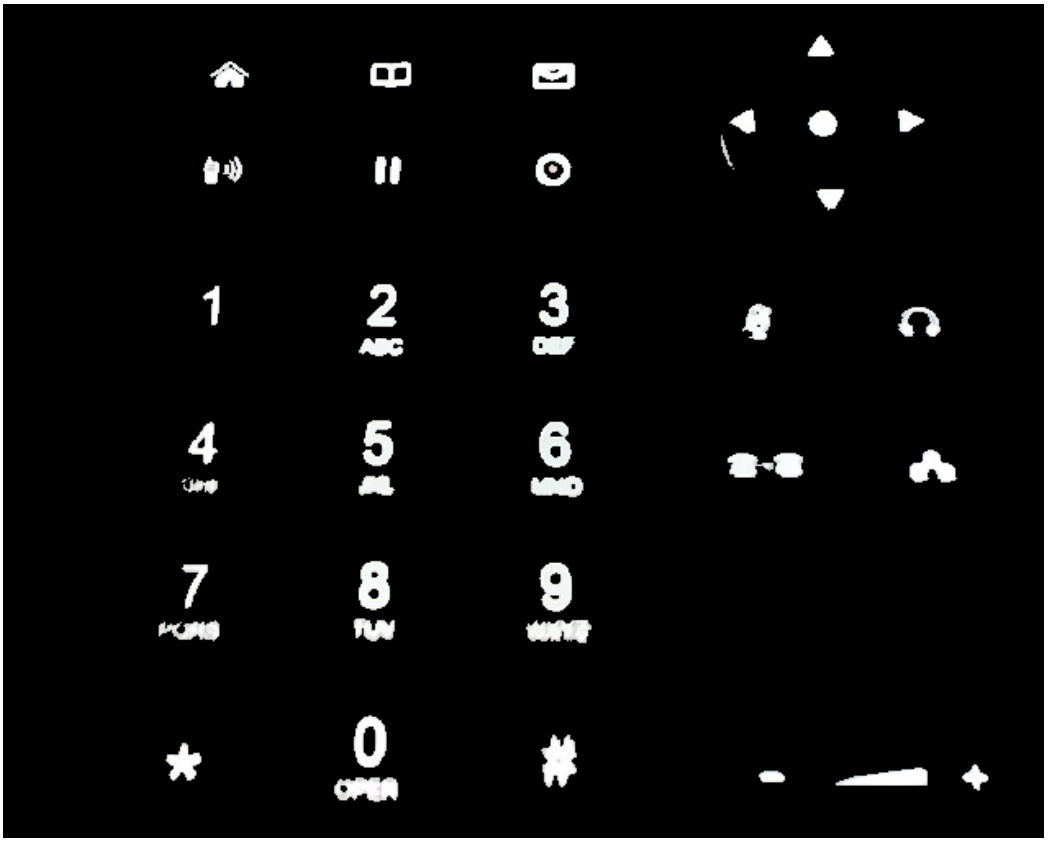

Рисунок 9 – Пример фильтрации изображения по цвету. При съемке лицевой панели стационарного телефона и проведении фильтрации по цвету, близкому к белому, получаем изображение только нанесенных белой краской значений

1.2. Определение центра масс некоторой области с целью определения координат точки, от которой проводится измерение [19]. Для этого на изображении определяются независимые области (blobs), которые далее отсеиваются по принципу минимально допустимого размера. Затем определяется наибольшая область из полученных с помощью фильтрации изображения по цвету областей. Согласно бизнес-логике, данная область будет являться меткой точки начала измерения. Далее определяется медианная точка по координатам точек области, координаты которой и будут координатами точки начала измерения.

2. Производить определение уравнения прямой, принадлежащей трубке, до которой производится измерение. Для решения этой задачи используются следующие алгоритмы:

2.1. Фильтрация изображения с целью выделения границ объектов с использованием алгоритма Кэнни [20]. Пример результата работы алгоритма представлен на рисунке 10. Для работы алгоритма необходимо преобразовать изображение в 8-битный формат (изображение в градациях серого). Алгоритм состоит из следующих элементов [21]:

2.1.1. Сглаживание. Размытие изображения для удаления шума.

2.1.2. Поиск градиентов. Границы отмечаются там, где градиент изображения приобретает максимальное значение.

2.1.3. Подавление не-максимумов. Только локальные максимумы отмечаются как границы.

2.1.4. Двойная пороговая фильтрация. Потенциальные границы определяются порогами.

2.1.5. Трассировка области неоднозначности. Итоговые границы определяются путём подавления всех краёв, несвязанных с определенными (сильными) границами.

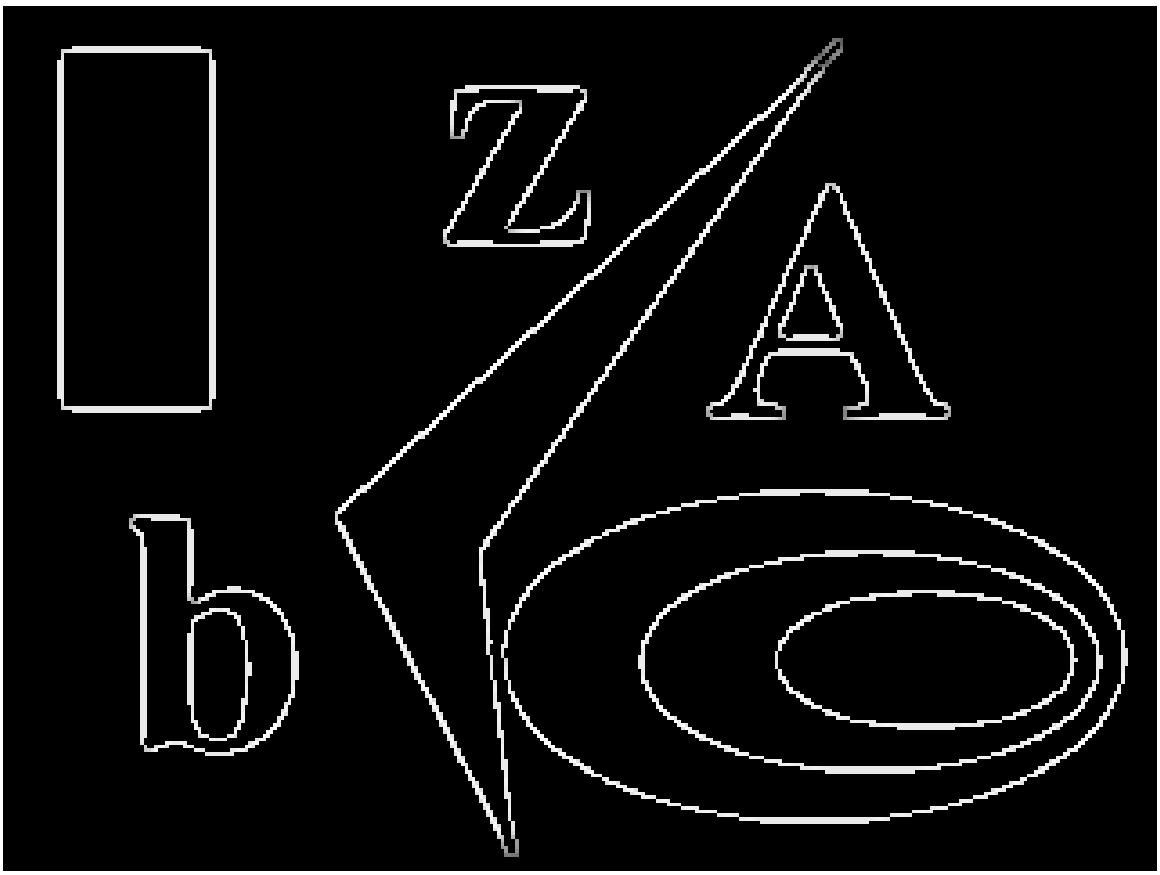

Рисунок 10 – Пример результата обработки изображения с использованием алгоритма Кэнни

2.2. Преобразования Хафа [22] с целью определения прямых линий на изображении. Результатом работы алгоритма является массив линий в полярных координатах [23], что отображено на рисунке 11. Линия с наибольшей интенсивностью (визуализируется толщиной) будет являться искомой, что гарантируется логикой бизнес-процесса.

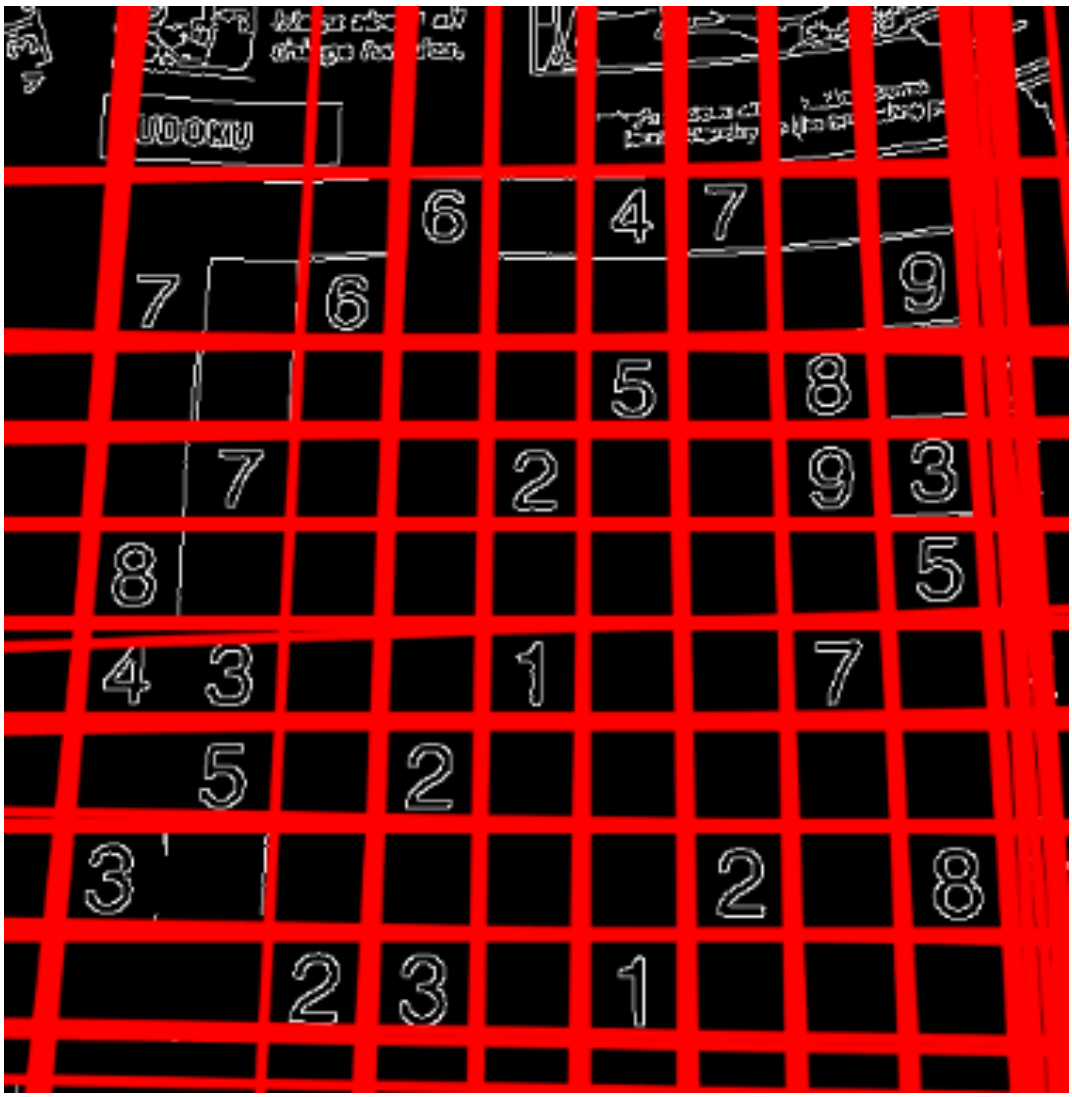

Рисунок 11 – Результат поиска прямых линий на изображении с использованием преобразования Хафа

3.2 Вывод по разделу 3

В результате проведенной работы были определен набор задач, предъявляемых к алгоритмам машинного зрения, и стандартные функции для работы с машинным зрением, которые реализованы во внешней библиотеке AForge.NET.

4 РАЗРАБОТКА АЛГОРИТМА РАСЧЕТА ГЕОМЕТРИЧЕСКИХ ХАРАКТЕРИСТИК МЕТАЛЛИЧЕСКИХ ТРУБОК

4.1 Исходные данные

В качестве исходных данных выступают параметры оснастки, которая изображена на рисунке 12, для измерения трубок, применяемых для производства расходомеров ЭлМетро-Фломак [24] и результаты измерений среднего диаметра трубок.

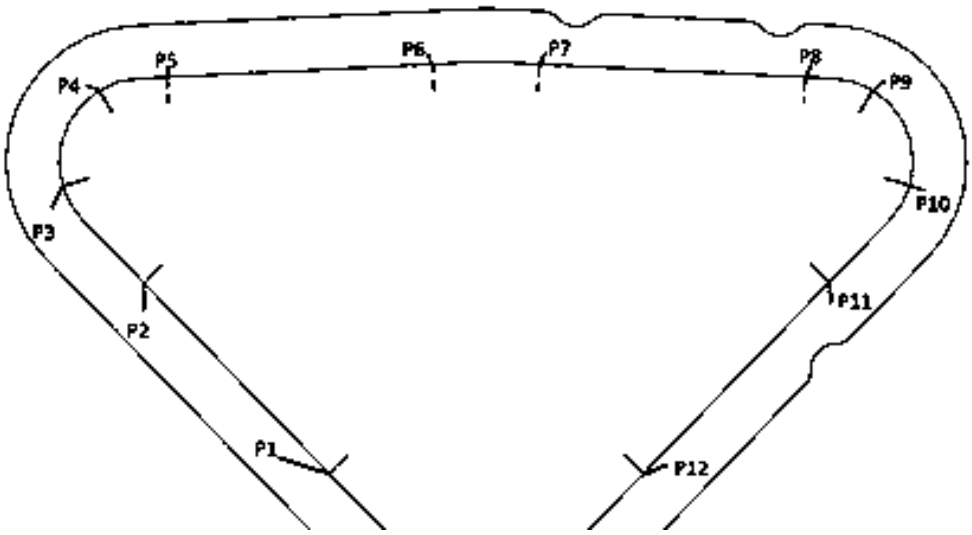

Рисунок 12 – Оснастка для измерения трубок; P_1-P_{12} - точки, относительно которых проводится измерения

Измерительная оснастка изготовлена из нержавеющей стали, температурный коэффициент линейного расширения которой позволяет полагать, что геометрия оснастки известна с достаточной точностью и характеризуется некоторыми константами [25].

В качестве данных об измерении используются 12 расстояний от базовых точек до измеряемой трубки. Целесообразность выбора точек объяснена в описании выполнения алгоритма.

Для корректной работы алгоритма в качестве исходных данных также следует принять ряд следующих допущений и начальных условий:

- расстоянием от базовой точки до образующей трубки будем считать длину отрезка, перпендикулярного к касательной к образующей трубки и соединяющего базовую точку и точку на образующей трубки;

- принимаем за направление обхода оснастки обход по часовой стрелки. То есть считаем, что для точек, указанных на рис. 1, образующих прямые линии, сначала идет начальная точка, а потом конечная. Например, для линии, образованной точками P_1 и P_2, начальной точкой считаем P_1, а конечной P_2;
- принимаем, что все прямые описываются общим уравнением вида $Ax + By + C = 0$. Это необходимо для дальнейшей целостности алгоритма.

В результате расчета должны быть получены следующие параметры:

- угол левый, угол правый, угол центральный, °;
- радиус левый, проверка радиуса левого, мм;
- радиус правый, проверка радиуса правого, мм;
- расстояние между перемычками. мм;
- расстояние симметрии, мм.

4.2 Алгоритм выполнения промежуточных вычислений.

Для проведения расчета выходных данных необходимо выполнить ряд промежуточных вычислений для математического описания некоторых характеристик объекта измерения, таких как расчет уравнений, определяющих прямые, принадлежащие трубке или оснастке, или определение координат точек.

4.2.1 Определение базовых прямых.

Под базовыми прямыми будем понимать прямые, образованные парами базовых точек на оснастке. Коэффициенты A, B, C общего уравнения прямой по двум точкам определяются следующим образом [26, с.60]:

$$\begin{cases} A = y_{\text{нач}} - y_{\text{кон}} \\ B = x_{\text{кон}} - x_{\text{нач}} \\ C = x_{\text{нач}} y_{\text{кон}} - x_{\text{кон}} y_{\text{нач}} \end{cases}, \quad (1)$$

где $y_{нач}$ – значение начальной точки по оси Y, $y_{кон}$ – значение конечной точки по оси Y, $x_{кон}$ – значение конечной точки по оси X, $x_{нач}$ – значение конечной точки по оси X.

Данным алгоритмом определяются коэффициенты прямых между точками P_1 и P_2, P_5 и P_6, P_7 и P_8, P_{11} и P_{12}.

4.2.2 Определение положения прямолинейных участков оси трубки.

Определим коэффициенты общего уравнения для прямых участков оси трубки. Для этого руководствуемся нижеуказанным алгоритмом.

Рассмотрим некоторый прямолинейный участок *AB*, принадлежащий оснастке и который схематично изображен на рисунке 13. Координаты начальной точки $A(x_{нач}, y_{нач})$ и конечной точки $B(x_{кон}, y_{кон})$ известны из начальных условий. Коэффициенты (*A*, *B*, *C*) линейной функции для данного участка определены в пункте «Определение базовых прямых».

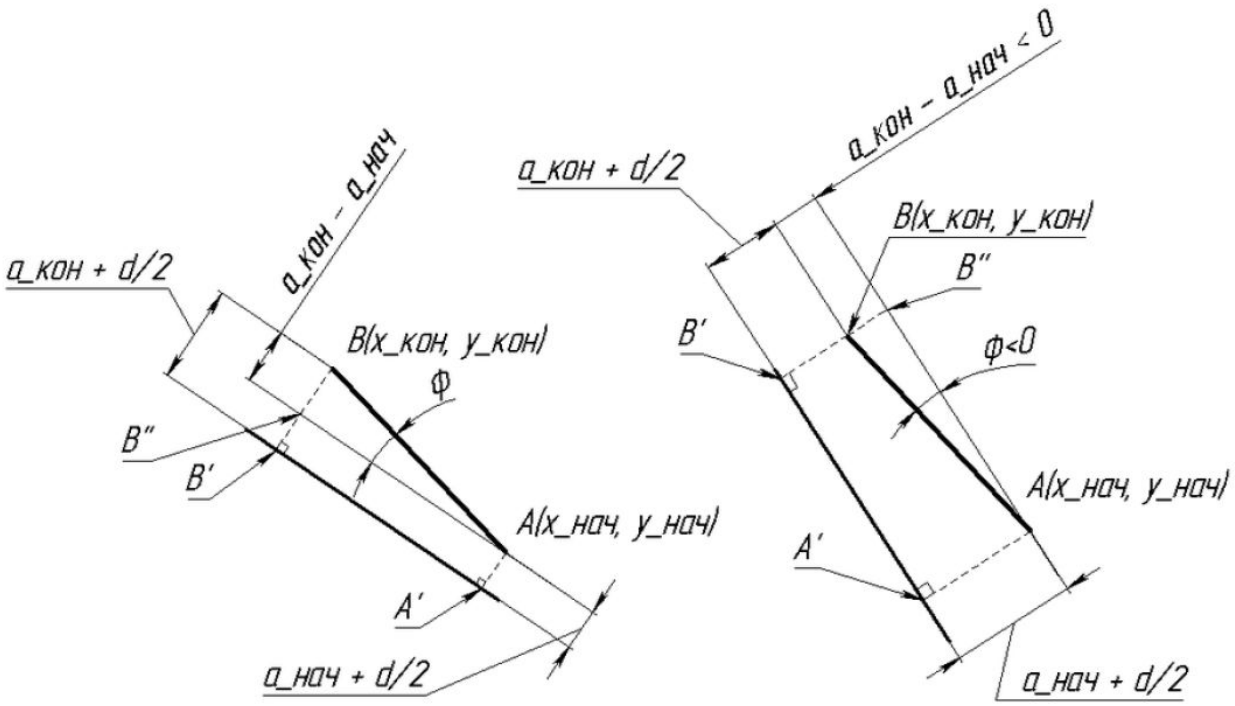

Рисунок 13 – Схема определения положения прямолинейного участка гибки

Требуется определить коэффициенты прямолинейного участка *A'B'* по расстояниям $a_{нач}$ и $a_{кон}$, которые определяются как перпендикуляры из точек *A* и *B* на искомую прямую *A'B'*, а также диаметр *d*.

Примем одну из точек базовой прямой за базовую. Пусть, например, это будет всегда точка с нечетным номером, для примера возьмём точку *A*. Построим прямую *AB''* так, чтобы она была параллельна прямой *A'B'* и проходила через точку *A*. Математически это эквивалентно следующему набору действий:

1. Перенести систему координат в точку *A*.

2. Повернуть прямую *AB* на угол φ [26, с.61]:

$$\varphi = \arcsin\frac{a_{\text{кон}} - a_{\text{нач}}}{\sqrt{(x_{\text{кон}} - x_{\text{нач}})^2 + (y_{\text{кон}} - y_{\text{нач}})^2}}. \quad (2)$$

Нужно заметить, что числитель ($a_{кон} - a_{нач}$) может получаться как положительным, так и отрицательным, и угол будет такого же знака. Положительное значение означает, что прямая *AB''* получена поворотом против часовой стрелки, отрицательное - по часовой.

3. Вернуть систему координат в исходную точку (0, 0).

В результате этих действий, которые можно описать в матричной форме, мы получим следующий набор коэффициентов для прямой *AB''*:

$$[A''B''C''] = [A\ B\ C]\begin{bmatrix}1 & 0 & x_{\text{нач}}\\ 0 & 1 & y_{\text{нач}}\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\cos\varphi & \sin\varphi & 0\\ -\sin\varphi & \cos\varphi & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -x_{\text{нач}}\\ 0 & 1 & -y_{\text{нач}}\\ 0 & 0 & 1\end{bmatrix}. \quad (3)$$

Получим параллельным переносом на расстояние ($a_{нач}$ + *d/2*) из прямой *AB''* прямую *A'B'*. Параллельный перенос для прямых, заданных уравнением общего вида, описывается следующим набором коэффициентов [26, с.63]:

$$\begin{cases} A' = A'' \\ B' = B'' \\ C' = C'' \pm \left(a_{\text{нач}} + \frac{d}{2}\right)\sqrt{A''^2 + B''^2} \end{cases}. \quad (4)$$

Знак «+» в формуле коэффициента C' означает, что прямая смещается в положительном направлении вдоль вектора нормали (*A''*, *B''*) исходной прямой, а знак «-» - что в отрицательном направлении.

Если соблюдать допущение об обходе по часовой стрелке, то для всех линий нужно использовать знак «-».

В результате проделанных операций получим коэффициенты прямых, описывающих оси прямолинейных участков трубки (4 набора коэффициентов)

4.2.3 Определение радиусов гиба по осевой линии трубки.

Для определения радиусов гиба по осевой линии необходимо решить следующую систему уравнений:

$$\begin{cases} \left(A_1C_1 + A_1B_1y_0 - {B_1}^2y_0\right)^2 - \\ \left({A_1}^2 + {B_1}^2\right)\left({C_1}^2 + 2B_1C_1y_0 + {B_1}^2({x_0}^2 + {y_0}^2 - R^2)\right) = 0; \\ \left(A_2C_2 + A_2B_2y_0 - {B_2}^2y_0\right)^2 - \\ \left({A_2}^2 + {B_2}^2\right)\left({C_2}^2 + 2B_2C_2y_0 + {B_2}^2({x_0}^2 + {y_0}^2 - R^2)\right) = 0; \\ (x_{\text{баз}} - x_0)^2 + (y_{\text{баз}} - y_0)^2 - \left(R - a - \frac{d}{2}\right)^2 = 0; \end{cases}, \quad (5)$$

где A_1, B_1, C_1 – коэффициенты, определяющие первую касательную, A_2, B_2, C_2 – коэффициенты, определяющие вторую касательную, a - измеренное расстояние относительно базовой точки оснастки с координатами ($x_{баз}$, $y_{баз}$), d - диаметр трубки, x_0, y_0, R - неизвестные координаты центра окружности и её радиус.

Система уравнений решается независимо для левого и правого участков. На выходе получается интересующий нас радиус и координаты центра окружности. Если точек на дуге две, система уравнений считается независимо для каждой из точек.

Данным алгоритмом определяются радиусы для точек P_3, P_4, P_9 и P_{10}.

4.2.4 Определение биссектрисы.

Биссектриса нужна для последующего определения расстояния между перемычками и симметрии. Биссектриса определяется для двух центральных участков, т.е. определяемых точками P_5 - P_8. Обозначим их как $P_5'P_6'$ (A'_{56}, B'_{56}, C'_{56}) и $P_7'P_8'$(A'_{78}, B'_{78}, C'_{78}). Возможны две биссектрисы, определяемые уравнениями [26, с.62]:

$$\frac{A'_{56}x + B'_{56}y + C'_{56}}{\sqrt{{A'_{56}}^2 + {B'_{56}}^2}} = \pm\frac{A'_{78}x + B'_{78}y + C'_{78}}{\sqrt{{A'_{78}}^2 + {B'_{78}}^2}}. \quad (6)$$

Знак «+» определяет биссектрису угла от линии $P'_5P'_6$ до линии $P'_7P'_8$ против часовой стрелки до первого совмещения; а знак «-» - биссектрису угла от линии $P'_7P'_8$ до линии $P'_5P'_6$ против часовой стрелки до первого совмещения.

Для вычисления геометрии необходимо использовать знак «-».

4.2.5 Определение точки начала биссектрисы.

Точка начала биссектрисы определяется как точка пересечения прямых $P_5'P_6'$ (A'_{56}, B'_{56}, C'_{56}) и $P_7'P_8'$ (A'_{78}, B'_{78}, C'_{78}). Координаты определяются следующими выражениями [26, с.62]:

$$x^{5678} = \frac{B'_{56}C'_{78} - B'_{78}C'_{56}}{A'_{56}B'_{78} - A'_{78}B'_{56}}, \quad (7)$$

$$y^{5678} = \frac{C'_{56}A'_{78} - C'_{78}A'_{56}}{A'_{56}B'_{78} - A'_{78}B'_{56}}, \quad (8)$$

где x_{5678} – координата точки начала биссектрисы по оси X, y_{5678} – координата точки начала биссектрисы по оси Y.

4.2.6 Определение точки на биссектрисе, находящейся на расстоянии h от точки начала биссектрисы.

Определяем координаты точки (x_h, y_h) на биссектрисе, находящейся на расстоянии h (см. исходные данные) от точки начала биссектрисы. Определять нужно из следующей системы уравнений:

$$\begin{cases} A_b x_h + B_b y_h + C_b = 0 \\ (x_h - x_{5678})^2 + (y_h - y_{5678})^2 - h^2 = 0 \end{cases}, \quad (9)$$

где A_b, B_b, C_b - коэффициенты, определяющие биссектрису.

Система будет иметь два решения. Нужно выбрать то, у которого y_h отрицательный (обеспечивается оснасткой). Если B_b равен 0, то $x_H = x_{5678}$, $y_H = y_{5678} - h$.

4.2.7 Определение линии, соединяющей точки перемычек.

Данная линия перпендикулярна к биссектрисе и проходит через точку (x_h, y_h).

Общее уравнение прямой будет иметь вид $A_H x + B_H y + C_H = 0$, где

$$\begin{cases} A_H = B_b \\ B_H = -A_b \\ C_H = A_b y_H - B_b x_H \end{cases}. \quad (10)$$

4.2.8 Определение координат точек перемычек.

Координаты точек перемычек (x_{per1}, y_{per1}) и (x_{per2}, y_{per2}). определяются по формулам:

$$\begin{cases} x_{per1} = \frac{B\prime_{12}C_H - B_H C\prime_{12}}{A\prime_{12}B_H - A_H B\prime_{12}} \\ y_{per1} = \frac{C\prime_{12}A_H - C_H A\prime_{12}}{A\prime_{12}B_H - A_H B\prime_{12}}, \end{cases} \quad (11)$$

$$\begin{cases} x_{per2} = \frac{B\prime_{1112}C_H - B_H C\prime_{1112}}{A\prime_{1112}B_H - A_H B\prime_{1112}} \\ y_{per2} = \frac{C\prime_{1112}A_H - C_H A\prime_{1112}}{A\prime_{1112}B_H - A_H B\prime_{1112}}, \end{cases} \quad (12)$$

где x_{per1}, y_{per1} - координаты первой перемычки; x_{per2}, y_{per2} - координаты второй перемычки; A_{12}, B_{12}, C_{12} - коэффициенты, определяющие базовую прямую между точками P_1', P_2'; A_{1112}, B_{1112}, C_{1112} - коэффициенты, определяющие базовую прямую между точками P_{11}', P_{12}'.

4.3 Расчет выходных данных.

Согласно имеющейся модели пересчёта геометрии, выходными данными для расчета являются:

1. Угол между двумя пересекающимися прямыми, определенными уравнениями общего вида, определяется следующим выражением [26, с.62]:

$$\tan \gamma_{12} = \frac{A_1 B_2 - A_2 B_1}{A_1 A_2 + B_1 B_2}. \quad (13)$$

При этом под углом γ_{12} следует понимать угол, на который нужно повернуть против часовой стрелки прямую 1 вокруг точки пересечения с прямой 2 до первого совмещения с прямой 2. В соответствии с этой логикой, нужно вычислять углы следующим образом:

- угол левый – это угол между прямой, определенной точками P'_1 и P'_2, и прямой, определенной точками P'_5 и P'_6;
- угол правый – это угол между прямой, определенной точками P'_7 и P'_8, и прямой, определенной точками P'_{11} и P'_{12};
- угол центральный — это угол между прямой, определенной точками P'_7 и P'_8, и прямой, определенной точками P'_5 и P'_6.

2. Радиусы определяются решением уравнений (см. п.4.2.3).

3. Расстояние между перемычками определяется выражением [23, с.57]:

$$L_{perem} = \sqrt{\left(x_{per1} - x_{per2}\right)^2 + (y_{per1} - y_{per2})^2}. \quad (14)$$

4. Расстояние симметрии определяется выражением:

$$L_{sym} = \sqrt{\left(x_{per1} - x_h\right)^2 + (y_{per1} - y_h)^2}. \quad (15)$$

Алгоритм расчета геометрических характеристик приведена на рисунке 14.

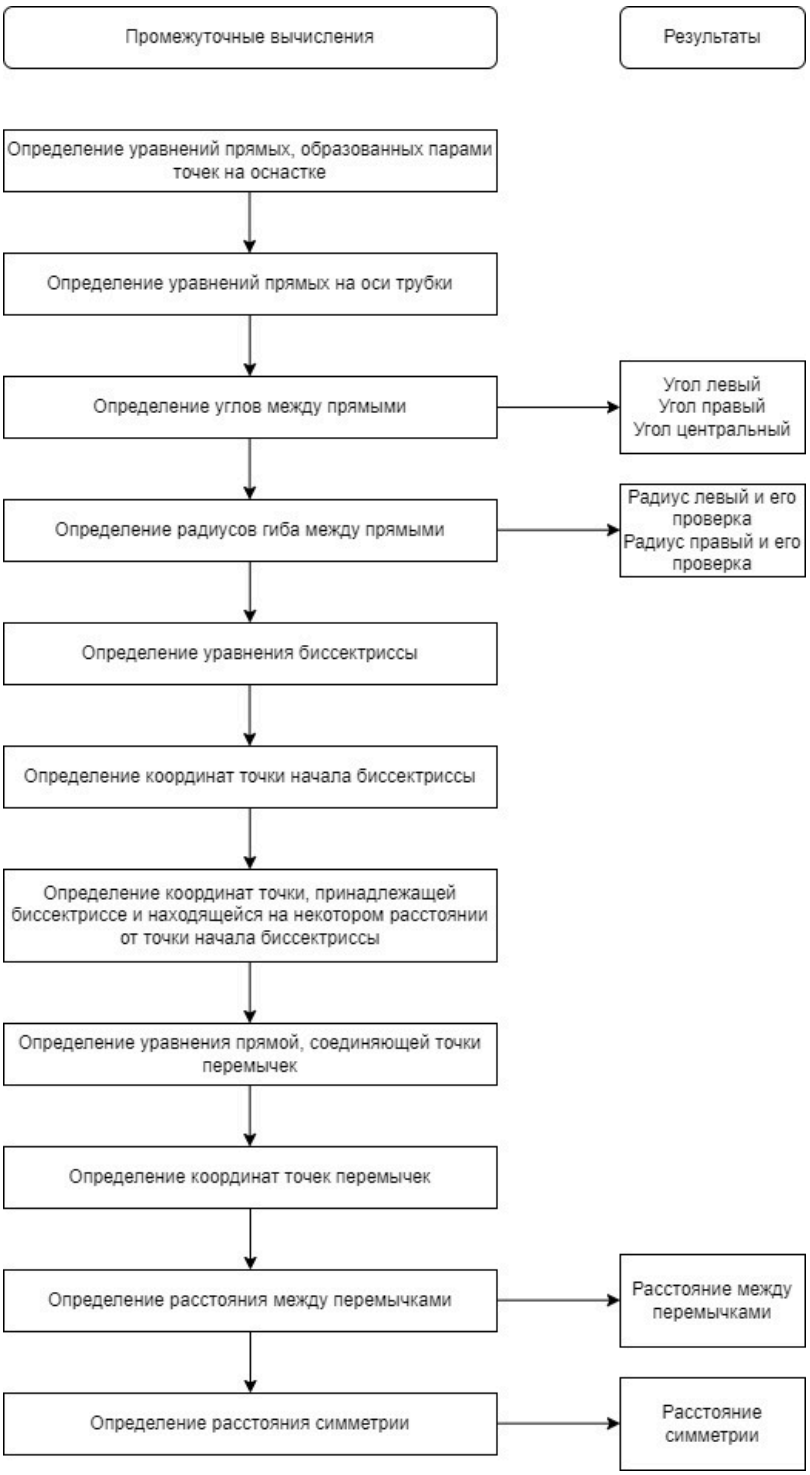

Рисунок 14 – Алгоритм расчета геометрических характеристик трубок

4.4 Вывод по разделу 4

Алгоритм, рассмотренный в данном исследовании, позволяет получить на основе результатов замеров гнутой трубки на измерительной оснастке ее геометрические характеристики – углы между линиями гиба, радиусы гибка, расстояние между перемычками и расстояние симметрии. Эти характеристики используются для дальнейшего сопровождения этапов производства, в которых участвуют измеренные трубки.

5 РЕАЛИЗАЦИЯ, ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММЫ

5.1 Реализация программы

Ниже представлен пользовательский интерфейс разработанной программы. На рисунке 15 изображено главное окно программы в процессе измерения трубок. Красной линией отображается граница объекта измерения. Желтым кругом выделяется область начала измерения – помеченная точка на измерительной оснастке. Желтая линия – расстояние от центра начала измерения до границы объекта измерения.

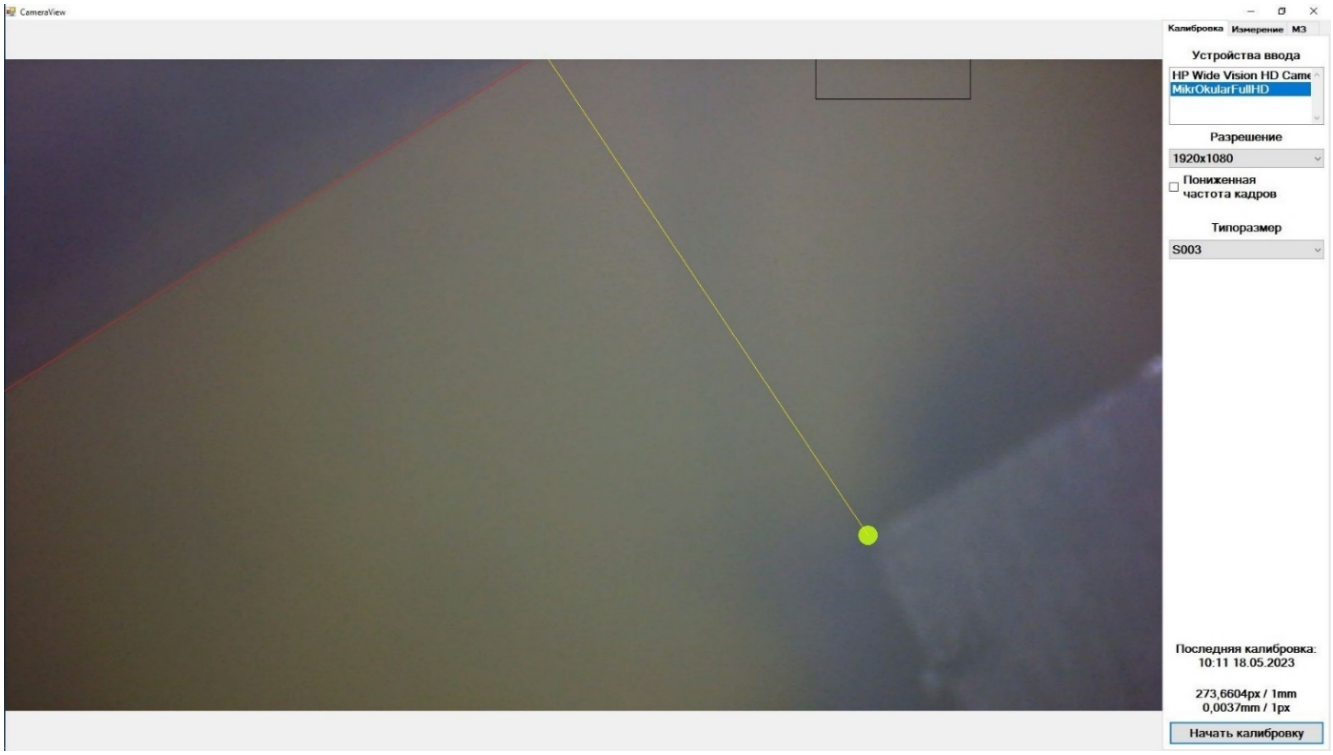

Рисунок 15 – Интерфейс процесса измерения

Исходный код для обработки получаемого с микроскопа изображения представлен в приложениях А и Б.

На рисунке 16 изображено главное окно программы в неактивном режиме. На боковой панели инструментов можно увидеть открытую вкладку «Измерение». Здесь находятся инструменты задания названия измеряемой трубки, порядкового номера измерения и измеряемой точки. Также присутствует элемент контроля качества измерения «Экранная лупа» для возможности ручного контроля.

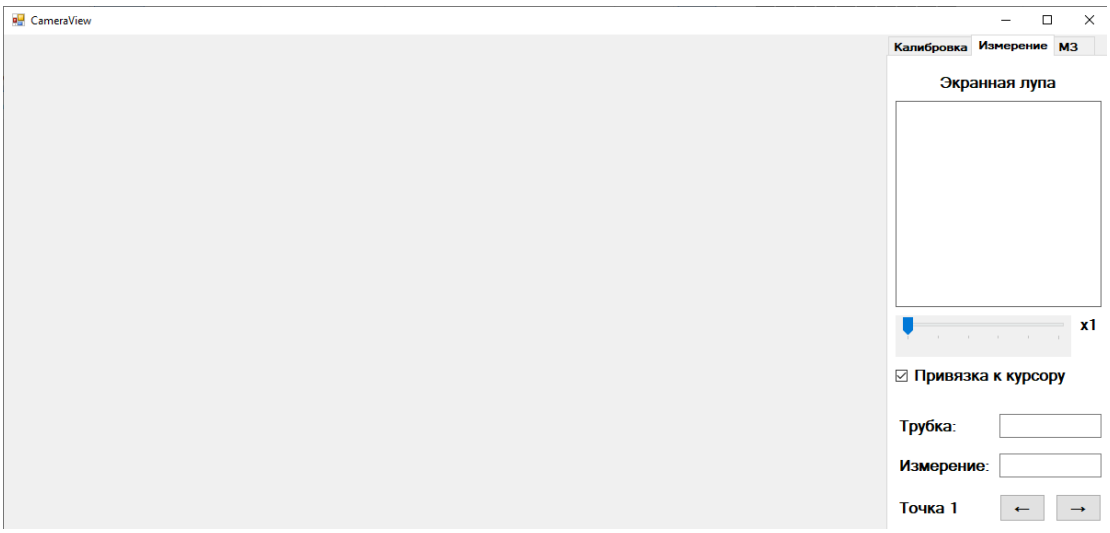

Рисунок 16 – Интерфейс ввода названия трубки, порядкового номера измерения и номера измеряемой точки, а также вспомогательные элементы контроля

На рисунке 17 изображен процесс калибровки электронного микроскопа. В процессе калибровки оператором измеряется расстояние от одного края калибровочной пластинки до другого края, и данное расстояние принимается за единицу измерения.

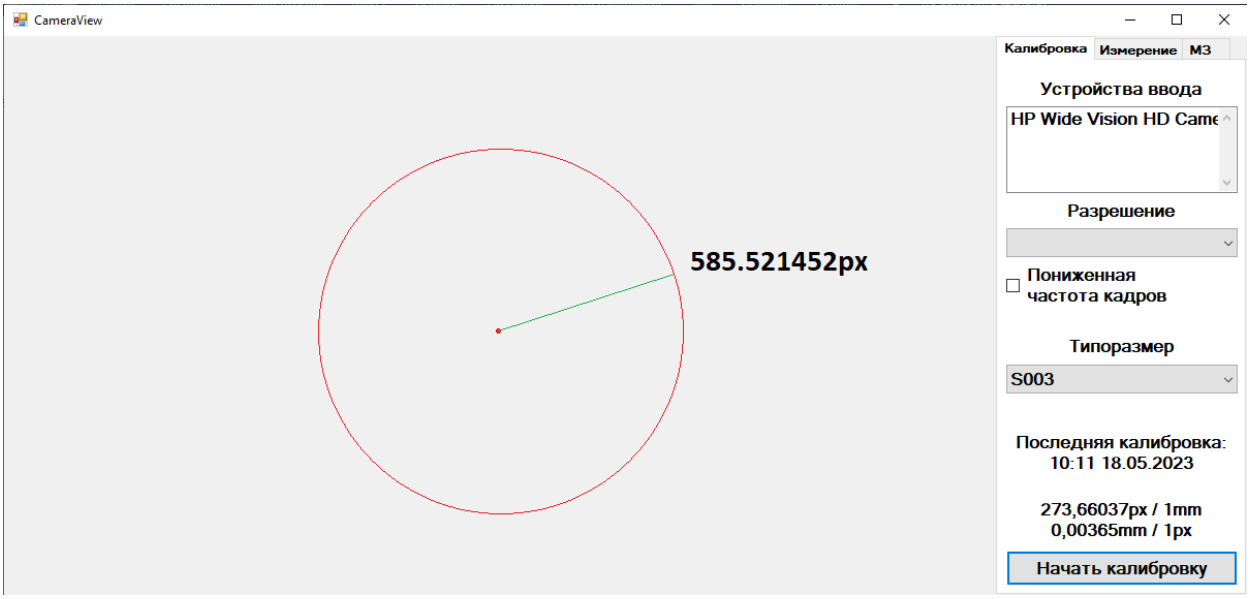

Рисунок 17 – Интерфейс калибровки микроскопа

На рисунке 18 изображены элементы вкладки «МЗ», отвечающие за настройку параметров для работы алгоритмов машинного зрения. Присутствует возможность настройки параметров для работы алгоритма Кэнни, для разбиения изображения на области и для фильтрации изображения по цвету.

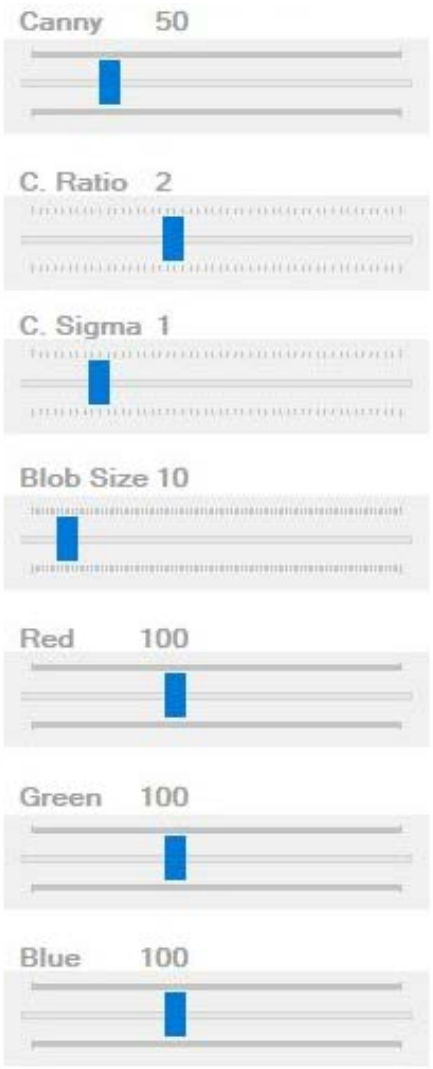

Рисунок 18 – Интерфейс калибровки алгоритмов машинного зрения

На рисунке 19 изображен интерфейс настройки параметров сервера. Присутствует возможность как ручной настройки адреса сервера и данных пользователя, так и предварительно заготовленные настройки.

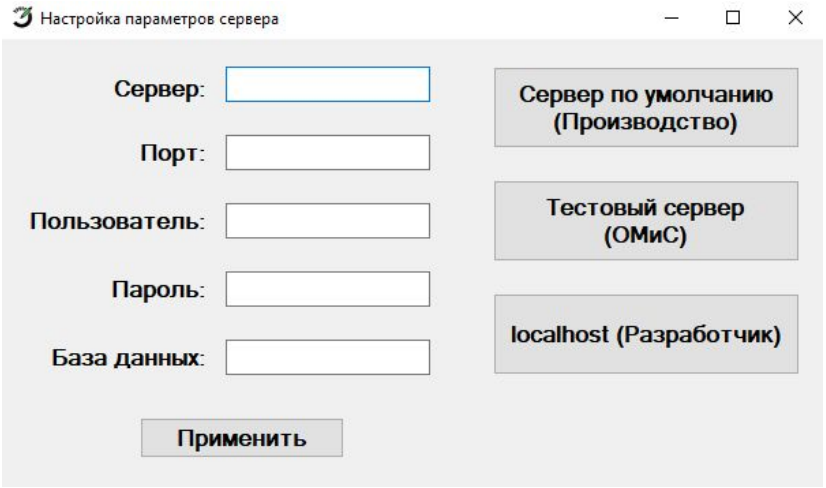

Рисунок 19 – Интерфейс настройки параметров сервера

На рисунке 20 изображен интерфейс просмотра данных. Есть возможность просмотра названия трубки, порядкового номера ее измерения и измеренных расстояний, а также кнопка инициализации расчета геометрических характеристик трубки и выгрузки всех результатов на сервер.

Рисунок 20 – Интерфейс просмотра данных

Исходный код для расчета геометрических характеристик на основе результатов измерений представлен в приложении В.

5.2 Тестирование и отладка программы.

Для проверки соответствия программы требованиям Заказчика было проведено функциональное тестирование блока измерения трубок и блока расчета геометрических характеристик.

Для тестирования программы применялись следующие методы:

1. Проверка результатов измерения. Для проверки измерения проводилось сравнение усредненных результатов нескольких измерений, полученных для одной и той же точки одной и той же трубки с помощью как программы, применяемой на производстве в настоящее время – MCView, так и разрабатываемой программы. После уточнения и корректной калибровки настроек алгоритмов машинного зрения результаты работы разрабатываемой программы были точнее результатов работы программы MCView, а достигались они быстрее.

2. Проверка результатов расчетов. Для проверки результатов расчетов разрабатываемой программы проводилось их сравнение с результатами расчетов геометрических характеристик трубок программой, применяемой в настоящее время – SOLIDWORKS. Каждой программе были выданы результаты измерений 3 партий трубок по 1000 трубок в каждой, 3 измерения для каждой трубки – всего 9000 измерений. Разница между соответствующими измерениями трубок никогда не превышала значения 10^{-8}, что было оценено Заказчиком как

удовлетворительный результат. Пример результатов тестирования приведен на рисунке 21.

Разница								
3,397E-09	-5,9E-10	-1,4E-10	4,66E-09	8,38E-09	-2,9E-09	2,51E-09	1,03E-08	-4,3E-08
3,397E-09	-5,9E-10	-1,4E-10	4,63E-09	8,22E-09	-2,9E-09	2,51E-09	1,03E-08	-4,3E-08
3,396E-09	-5,6E-10	-2,6E-10	4,99E-09	9E-09	-2,7E-09	2,52E-09	1,03E-08	-4,2E-08
3,395E-09	-5,6E-10	-2,8E-10	4,96E-09	9,03E-09	-2,8E-09	2,52E-09	1,03E-08	-4,1E-08
3,424E-09	-5,7E-10	-2,6E-10	4,9E-09	8,59E-09	-2,8E-09	2,52E-09	1,04E-08	-3,8E-08
3,423E-09	-5,7E-10	-2,8E-10	4,94E-09	8,71E-09	-2,8E-09	2,52E-09	1,04E-08	-3,7E-08
3,389E-09	-5,7E-10	-2,5E-10	4,8E-09	9,11E-09	-2,9E-09	2,51E-09	1,03E-08	-4,2E-08
3,39E-09	-5,7E-10	-2,7E-10	4,81E-09	9,14E-09	-2,8E-09	2,51E-09	1,03E-08	-4,2E-08
3,415E-09	-5,7E-10	-1,5E-10	4,98E-09	8,54E-09	-2,7E-09	2,52E-09	1,03E-08	-3,8E-08
3,415E-09	-5,7E-10	-1,6E-10	5,01E-09	8,51E-09	-2,8E-09	2,52E-09	1,03E-08	-3,8E-08
3,418E-09	-5,7E-10	-1,7E-10	4,86E-09	8,69E-09	-2,8E-09	2,52E-09	1,04E-08	-3,9E-08
3,418E-09	-5,7E-10	-1,8E-10	4,84E-09	8,71E-09	-2,7E-09	2,52E-09	1,04E-08	-3,9E-08

Рисунок 21 – Пример результатов тестирования расчета геометрических характеристик

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы была спроектирована и реализована программа для автоматизации измерения и расчетов геометрических характеристик металлических трубок.

Для достижения поставленной цели были решены следующие задачи:

1. Выполнен анализ предметной области.
2. Выполнена разработка архитектуры программы.
3. Разработан набор алгоритмов для
4. Реализован алгоритм расчета характеристик трубок на основе данных, полученных измерением.
5. Проведены реализация, тестирование и отладка программы.

Полученная программа полностью соответствует требованиям Заказчика.

Программа позволила автоматизировать некоторые из бизнес-процессов предприятия Заказчика, повысила качество измерений и надежность хранимых данных.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Системы для измерения геометрических параметров труб [Электронный ресурс]. - ООО «АктивТестГруп», 2010- . Режим доступа: http://activetest.ru/geometric_parameter_measurement_of_pipes/, свободный. Загл. с экрана. (Дата обращения 02.02.2023г.)

2 Автоматизированная установка для измерения геометрических параметров сварных труб [Электронный ресурс]. - ООО «АктивТестГруп», 2010- . Режим доступа: http://activetest.ru/media/uploads/geomer_activetestgroup.pdf, свободный. Загл. с экрана. (Дата обращения 02.02.2023г.)

3 Система для контроля труб TubeShaper с Hexagon Absolute Arm [Электронный ресурс]. - ЗАО «Квалитет». Режим доступа: https://qualitet.su/?id=20766, свободный. Загл. с экрана. (Дата обращения 03.02.2023г.)

4 Портативная измерительная рука ROMER ABSOLUTE ARM [Электронный ресурс]. - ЗАО «Квалитет». Режим доступа: https://qualitet.su/?id=10801, свободный. Загл. с экрана. (Дата обращения 03.02.2023г.)

5 РЭ-УСКТ-8 [Электронный ресурс]. - НПЦ «Кропус», 2023. Режим доступа: https://kropus.com/upload/iblock/200/%D0%A0%D0%AD%20%D0%A3%D0%A1%D0%9A%D0%A2-8.pdf, свободный. Загл. с экрана. (Дата обращения 03.02.2023г.)

6 УСКТ-8 многоканальная система контроля тонкостенных труб [Электронный ресурс]. - НПЦ «Кропус», 2023. Режим доступа: https://kropus.com/catalog/pribory_avtomatizatsii/uskt-8-mnogokanalnaya-sistema-kontrolya-tonkostennykh-trub/, свободный. Загл. с экрана. (Дата обращения 03.02.2023г.)

7 А. С. Подольская. Метрологическое обеспечение контроля трубопроводов сложной пространственной конфигурации / А. С. Подольская, В. П. Жереб // Актуальные проблемы авиации и космонавтики – 2019 г. – Том 2 – 627 с.

8 Программное обеспечение «MCView» [Электронный ресурс]. - ООО "ЛОМО-Микросистемы", 2005- . Режим доступа: https://www.lomo-

microsystems.ru/doc/po-ru-ms.pdf, свободный. Загл. с экрана. (Дата обращения 07.02.2023г.)

9 SOLIDWORKS Manage: Linking External Data Sources and Importing [Электронный ресурс]. - Bryce Hooper, 2021. Режим доступа: https://www.goengineer.com/blog/linking-external-data-sources-importing-into-solidworks-manage, свободный. Загл. с экрана. (Дата обращения 08.02.2023г.)

10 Официальный сайт САПР Компас [Электронный ресурс]. - Habr, 2006-2023. Режим доступа: https://kompas.ru/, свободный. Загл. с экрана. (Дата обращения 10.02.2023г.)

11 Импорт и экспорт [Электронный ресурс]. - ООО «АСКОН-Системы проектирования», 2022. Режим доступа: https://help.kompas.ru/1224_139_8_3_ehksport_v_formaty.html, свободный. Загл. с экрана. (Дата обращения 10.02.2023г.)

12 Overview – Windows Forms .NET Framework [Электронный ресурс]. – Microsoft, 2023. Режим доступа: https://learn.microsoft.com/en-us/dotnet/desktop/winforms/windows-forms-overview?view=netframeworkdesktop-4.8/, свободный. Загл. с экрана. (Дата обращения 13.02.2023г.)

13 What is Windows Presentation Foundation – WPF .NET [Электронный ресурс]. – Microsoft, 2023. Режим доступа: https://learn.microsoft.com/en-us/dotnet/desktop/wpf/overview/?view=netdesktop-7.0/, свободный. Загл. с экрана. (Дата обращения 13.02.2023г.)

14 What do you think of winforms? : csharp [Электронный ресурс]. – Reddit, 2023. Режим доступа: https://www.reddit.com/r/csharp/comments/kz9p41/what_do_you_think_of_winforms/, свободный. Загл. с экрана. (Дата обращения 17.03.2023г.)

15 AForge.NET :: Framework’s License [Электронный ресурс]. – Aforge.NET, 2008-2012. Режим доступа: http://www.aforgenet.com/framework/license.html/, свободный. Загл. с экрана. (Дата обращения 17.03.2023г.)

16 Opencvsharp/LICENCE at master [Электронный ресурс]. – Github, Inc., 2023. Режим доступа: https://github.com/shimat/opencvsharp/blob/master/LICENSE/, свободный. Загл. с экрана. (Дата обращения 13.02.2023г.)

17 License – Accord.NET Machine Learning in C# [Электронный ресурс]. – Cesar Souza, 2009-2017. Режим доступа: http://accord-framework.net/license.html/, свободный. Загл. с экрана. (Дата обращения 13.02.2023г.)

18 ColorFiltering Class [Электронный ресурс]. – AForge.NET, 2006-2013. Режим доступа: http://www.aforgenet.com/framework/docs/html/35bd90e3-4e35-8f5f-e255-26c5d8d4b927.htm/, свободный. Загл. с экрана. (Дата обращения 16.03.2023г.)

19 CenterOfGravity Property [Электронный ресурс]. – AForge.NET, 2006-2013. Режим доступа: http://www.aforgenet.com/framework/docs/html/6327728e-059e-9a46-e8a7-125a16e23bdb.htm/, свободный. Загл. с экрана. (Дата обращения 16.03.2023г.)

20 CannyEdgeDetector Class [Электронный ресурс]. – AForge.NET, 2006-2013. Режим доступа: http://www.aforgenet.com/framework/docs/html/e08cae30-7a37-db9f-cede-05cf6521343f.htm/, свободный. Загл. с экрана. (Дата обращения 16.03.2023г.)

21 Детектор границ Кэнни / Хабр [Электронный ресурс]. - Habr, 2006-2023. Режим доступа: https://habr.com/ru/articles/114589/, свободный. Загл. с экрана. (Дата обращения 17.03.2023г.)

22 HoughLineTransformation Class [Электронный ресурс]. – AForge.NET, 2006-2013. Режим доступа: http://www.aforgenet.com/framework/docs/html/aa234a78-bc23-c0a8-4a36-33aba5c75d9c.htm/, свободный. Загл. с экрана. (Дата обращения 16.03.2023г.)

23 Алгоритм Хафа для обнаружения произвольных кривых на изображении / Хабр [Электронный ресурс]. - Habr, 2006-2023. Режим доступа: https://habr.com/ru/articles/102948/, свободный. Загл. с экрана. (Дата обращения 17.03.2023г.)

24 Расходомеры и счетчики - ЭлМетро - производитель контрольно-измерительного и метрологического оборудования [Электронный ресурс]. - ООО «ЭлМетро-Групп», 2023. Режим доступа:

https://www.elmetro.ru/production/flowmeters/elmetro_flomac.html, свободный. Загл. с экрана. (Дата обращения 20.04.2023г.)

25 Башевская О.С. Исследование влияния температурных деформаций на точность линейных измерений. Измерительная техника. 2013;(9):34-36.

26 Корн Г. Справочник по математике для научных работников и инженеров Пер. с англ. И. Г. Арамановича, А. М. Березмана, И.А. Вайнштейна, Л. З. Румшинского, Л.Я. Цлафа под общей редакцией И. Г. Арамановича. – М.: Издательство "Наука", главная редакция физико-математической литературы, 1973. – 832 с.

ПРИЛОЖЕНИЯ

Приложение А

Исходный код обработки кадра изображения

```
private void CaptureDevice_NewFrame(object sender, NewFrameEventArgs eventArgs)
{
    // Опция обработки только 50% кадров, несколько повышает производительность.
    if (lowFrameRate) {
        if (!processNextFrame) {
            processNextFrame = true;
            return; }
        processNextFrame = false; }
    // Ширина/высота изображения для зума.
    int zoomWidth = pictureBoxZoom.Width / zoomFactor;
    int zoomHeight = pictureBoxZoom.Height / zoomFactor;
    int halfZoomWidth = zoomWidth / 2;
    int halfZoomHeight = zoomHeight / 2;
    // Фильтры для изменения размера изображения.
    var aspectRatio = (double)eventArgs.Frame.Width / (double)eventArgs.Frame.Height;
    var newSize = GetImageSize(aspectRatio);
    var filterCamera = new ResizeNearestNeighbor(newSize.Width, newSize.Height);
    var filterZoom = new ResizeNearestNeighbor(pictureBoxZoom.Width, pictureBoxZoom.Height);
    // Старые изображения для Dispose().
    var oldBitmapCamera = pictureBoxCamera.Image;
    var oldBitmapZoom = pictureBoxZoom.Image;
    // Отрисовываем данные с камеры на главный экран и изменяем его размер.
```

```
var bitmapCamera = eventArgs.Frame;
var bitmapCameraResized = filterCamera.Apply(bitmapCamera);
graphicsCamera = Graphics.FromImage(bitmapCameraResized);
graphicsCamera.InterpolationMode = InterpolationMode.HighQualityBicubic;
graphicsZoom = Graphics.FromImage(bitmapZoomResized);
graphicsZoom.InterpolationMode = InterpolationMode.HighQualityBicubic;
// ----------------------------------------------------------------------------
// МАШИННОЕ ЗРЕНИЕ НА ГЛАВНОМ ЭКРАНЕ.
var imageCopyColorFiltered = colorFilter.Apply(bitmapCameraResized);
blobFilter.ProcessImage(imageCopyColorFiltered);
var blobs = blobFilter.GetObjectsInformation();
var imageCopyGrayscale = bitmapCameraResized.Clone(new Rectangle(0, 0,
bitmapCameraResized.Width, bitmapCameraResized.Height),
PixelFormat.Format8bppIndexed);
cannyFilter.ApplyInPlace(imageCopyGrayscale);
lineDetector.ProcessImage(imageCopyGrayscale);
Bitmap houghLineImage = lineDetector.ToBitmap();
HoughLine[] lines = lineDetector.GetMostIntensiveLines(1);
if (blobs.Length > 0 && lines.Length > 0)
{
    var center = blobs[0].CenterOfGravity;
    var line = lines[0];
    int r = line.Radius;
    double t = line.Theta;
    // Если линия находится в нижней части изображения.
    if (r < 0) { t += 180; r = -r; }
    // Перевод в радианы
    t = (t / 180) * Math.PI;
    // Все координаты идут относительно центра изображения.
```

```
int w2 = imageCopyGrayscale.Width / 2;
int h2 = imageCopyGrayscale.Height / 2;
double x0 = 0, x1 = 0, y0 = 0, y1 = 0;
// Если линия не вертикальна
if (line.Theta != 0)
{
        x0 = -w2; // Левая точка
        x1 = w2;  // Правая точка
        y0 = (-Math.Cos(t) * x0 + r) / Math.Sin(t);
        y1 = (-Math.Cos(t) * x1 + r) / Math.Sin(t);
}
else
{
        // Вертикальная линия
        x0 = line.Radius;
        x1 = line.Radius;
        y0 = h2;
        y1 = -h2;
}
// Расчет уравнения прямой.
x0 = x0 + w2;
y0 = h2 - y0;
x1 = x1 + w2;
y1 = h2 - y1;
var A = y0 - y1;
var B = x1 - x0;
var C = x0 * y1 - x1 * y0;
var x2 = center.X;
var y2 = center.Y;
```

```
            var x3 = (B * (B * x2 - A * y2) - A * C) / (A * A + B * B);
            var y3 = (A * (A * y2 - B * x2) - B * C) / (A * A + B * B);
            // Расчет расстояния от точки до прямой.
            var distance = Math.Abs(A * x2 + B * y2 + C) / Math.Sqrt(A * A + B * B);
            graphicsCamera.DrawLine(redPen, (float)x0, (float)y0, (float)x1, (float)y1);
            graphicsCamera.DrawLine(greenPen, (float)x2, (float)y2, (float)x3,
(float)y3);
            graphicsCamera.DrawEllipse(
                  redPen,
                  new RectangleF(center.X - 5, center.Y - 5, 10, 10));
            graphicsCamera.DrawEllipse(
                  redPen,
                  new RectangleF((float)x3 - 5, (float)y3 - 5, 10, 10));
      }
      // ОТРИСОВКА НА ГЛАВНОМ ЭКРАНЕ.
      if (centerIsSet & currentRadius > 0)
      {
            // Отрисовка центра окружности.
            graphicsCamera.DrawEllipse(
                  redPen,
                  new RectangleF( pointCenter.X - 1, pointCenter.Y - 1, 2, 2));
            // Отрисовка окружности.
            graphicsCamera.DrawEllipse(
                  redPen,
                  new RectangleF(
                        pointCenter.X - currentRadius, pointCenter.Y - currentRadius,
                        currentRadius * 2, currentRadius * 2));
            // Отрисовка текущего радиуса в пикселях.
            graphicsCamera.DrawString(
```

```
                $"{currentRadius:0.000}px.",
                stringFont, greenBrush, pointCenter.X, pointCenter.Y - 15,
stringFormat);
            // Отрисовка текущего радиуса в миллиметрах (если откалибровано).
            if (calibrationValue > 0 && !isCalibrating)
                graphicsCamera.DrawString(
                    $"{currentRadius * calibrationValue:0.000}mm.",
                    stringFont, greenBrush, pointCenter.X, pointCenter.Y + 15,
stringFormat);
        }
        // Отрисовка области увеличения на главном экране.
        graphicsCamera.DrawRectangle(
            Pens.Black,
            new Rectangle(
                cursorPosition.X - halfZoomWidth, cursorPosition.Y -
halfZoomHeight, zoomWidth, zoomHeight));
        pictureBoxCamera.Image = bitmapCameraResized;
        imageCopyColorFiltered.Dispose();
        imageCopyGrayscale.Dispose();
        houghLineImage.Dispose();
        bitmapCamera.Dispose();
        bitmapZoom.Dispose();
        graphicsCamera.Dispose();
        graphicsZoom.Dispose();
        if (oldBitmapCamera != null)
            oldBitmapCamera.Dispose();
        if (oldBitmapZoom != null)
            oldBitmapZoom.Dispose();}
```

Приложение Б

Исходный код преобразования и записи сырых данных

```
private void ProcessData()
{
    var currentValue = currentRadius * calibrationValue;
    var currentImage = pictureBoxCamera.Image;
    var currentImageClone = (System.Drawing.Image)currentImage.Clone();
    var currentName = textBoxTubeName.Text;
    if (string.IsNullOrEmpty(currentName))
    {
        MessageBox.Show(this, "Введите название трубки.", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    var currentNumberCorrect = int.TryParse(textBoxMeasurementNumber.Text, out var currentNumber);
    if (!currentNumberCorrect)
    {
        MessageBox.Show(this, "Неправильный номер трубки.", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    var dataRow = Assist.CurrentDataTableStage2.NewRow();
    var newRow = true;
    foreach (DataRow row in Assist.CurrentDataTableStage2.Rows)
        if (row["Name"].Equals(textBoxTubeName.Text) && row["MeasurementNumber"].ToString().Equals(textBoxMeasurementNumber.Text))
        {
            dataRow = row;
```

```
                newRow = false;
                break;
            }
    dataRow["Name"] = currentName;
    dataRow["MeasurementNumber"] = currentNumber;
    dataRow[$"Point_{currentPointNumber}"] = currentValue;
    if (newRow)
        Assist.CurrentDataTableStage2.Rows.Add(dataRow);
    currentImageClone.Dispose();
}
```

Приложение В

Исходный код обработки результатов измерений

```
public static class S015
{
    protected class Line
    {
        public double X1 { get; }
        public double Y1 { get; }
        public double X2 { get; }
        public double Y2 { get; }
        public double A { get; }
        public double B { get; }
        public double C { get; }
        public Line(double x1, double y1, double x2, double y2)
        {
            this.X1 = x1;
            this.Y1 = y1;
            this.X2 = x2;
            this.Y2 = y2;

            this.A = Y1 - Y2;
            this.B = X2 - X1;
            this.C = X1 * Y2 - X2 * Y1;
        }
        public Line(double a, double b, double c)
        {
            this.A = a;
            this.B = b;
            this.C = c;
```

```
            }
        }
        private static Line CalculateShift(Line line, double A1, double A2, double D)
        {
            double angle = Math.Asin((A2 - A1) / (Math.Sqrt(line.A * line.A + line.B *
line.B)));
            double newA = line.A * Math.Cos(angle) - line.B * Math.Sin(angle);
            double newB = line.A * Math.Sin(angle) + line.B * Math.Cos(angle);
            double tempC = (line.A - newA) * line.X1 + (line.B - newB) * line.Y1 +
line.C;
            double newC = tempC - ((A1 + 0.5 * D) * Math.Sqrt(newA * newA + newB
* newB));
            return new Line(newA, newB, newC);
        }
        private static double CalculateAngle(Line line1, Line line2)
        {
            double tan = (line1.A * line2.B - line2.A * line1.B) / (line1.A * line2.A +
line1.B * line2.B);
            return Math.Atan(tan) * 180 / Math.PI;
        }
        private static double CalculateRad(Line line1, Line line2, double diameter, double
distance, double base_x, double base_y)
        {
            Func<Vector<double>, double>[] f =
            {
                x => (line1.A * line1.C + line1.A * line1.B * x[1] - line1.B * line1.B
* x[0]) * (line1.A * line1.C + line1.A * line1.B * x[1] - line1.B * line1.B * x[0])
                - ((line1.A * line1.A + line1.B * line1.B) * (line1.C * line1.C + 2 *
line1.B * line1.C * x[1] + line1.B * line1.B * (x[0] * x[0] + x[1] * x[1] - x[2] * x[2]))),
```

```
                x => (line2.A * line2.C + line2.A * line2.B * x[1] - line2.B * line2.B
* x[0]) * (line2.A * line2.C + line2.A * line2.B * x[1] - line2.B * line2.B * x[0])
                - ((line2.A * line2.A + line2.B * line2.B) * (line2.C * line2.C + 2 *
line2.B * line2.C * x[1] + line2.B * line2.B * (x[0] * x[0] + x[1] * x[1] - x[2] * x[2]))),
                x => (base_x - x[0]) * (base_x - x[0]) + (base_y - x[1]) * (base_y -
x[1]) - (x[2] - distance - (diameter / 2)) * (x[2] - distance - (diameter / 2))
            };
            Func<Vector<double>, Vector<double>, Vector<double>>[] df = { };
            var initialGuess = Vector.Create(42.0, 80.0, 80.0);
            DoglegSystemSolver dogleg = new DoglegSystemSolver(f, df, initialGuess)
{ TrustRegionRadius = 0.5, JacobianFunction = null };
            var solution = dogleg.Solve();
            return solution[2];
        }
        private static void CalculateBisector(Line line1, Line line2, double distance, out
Line bisector_out, out double[] point_out)
        {
            double n1 = Math.Sqrt(line1.A * line1.A + line1.B * line1.B);
            double n2 = Math.Sqrt(line2.A * line2.A + line2.B * line2.B);
            double A = (line1.A * n2 - line2.A * n1) / (n1 * n2);
            double B = (line1.B * n2 - line2.B * n1) / (n1 * n2);
            double C = (line1.C * n2 - line2.C * n1) / (n1 * n2);
            var bisector = new Line(A, B, C);
            double cross_x = (line1.B * line2.C - line2.B * line1.C) / (line1.A * line2.B
- line2.A * line1.B);
            double cross_y = (line1.C * line2.A - line2.C * line1.A) / (line1.A * line2.B
- line2.A * line1.B);
            Func<Vector<double>, double>[] f =
            {
```

```
                x => (bisector.A * x[0] + bisector.B * x[1] + bisector.C),
                x => (x[0] - cross_x) * (x[0] - cross_x) + (x[1] - cross_y) * (x[1] - cross_y) - distance * distance
            };
            Func<Vector<double>, Vector<double>, Vector<double>>[] df = { };
            var initialGuess = Vector.Create(0.0, 0.0);
            DoglegSystemSolver dogleg = new DoglegSystemSolver(f, df, initialGuess) { TrustRegionRadius = 0.5, JacobianFunction = null };
            var solution = dogleg.Solve();
            bisector_out = bisector;
            point_out = new double[] { solution[0], solution[1] };
        }
        public static void Solve(DataTable dataTable, string tubeType)
        {
            var dictionary = (JObject)Assist.Dictionary["Tubes"][tubeType]["Calculation"]["Geometry"];
            foreach (DataRow row in dataTable.Rows)
            {
                var lineP1P2_base = new Line((double)dictionary["P1"]["x"], (double)dictionary["P1"]["y"], (double)dictionary["P2"]["x"], (double)dictionary["P2"]["y"]);
                var lineP1P2 = CalculateShift( lineP1P2_base, (double)row["Point_1"], (double)row["Point_2"], (double)row["Diameter_Outer"]);
                var lineP3P4_base = new Line((double)dictionary["P3"]["x"], (double)dictionary["P3"]["y"], (double)dictionary["P4"]["x"], (double)dictionary["P4"]["y"]);
                var lineP3P4 = CalculateShift( lineP3P4_base, (double)row["Point_3"], (double)row["Point_4"], (double)row["Diameter_Outer"]);
```

```
var lineP5P6_base = new Line( (double)dictionary["P5"]["x"], (double)dictionary["P5"]["y"], (double)dictionary["P6"]["x"], (double)dictionary["P6"]["y"]);
var lineP5P6 = CalculateShift( lineP5P6_base,(double)row["Point_5"], (double)row["Point_6"], (double)row["Diameter_Outer"]);
var lineP7P8_base = new Line((double)dictionary["P7"]["x"], (double)dictionary["P7"]["y"], (double)dictionary["P8"]["x"], (double)dictionary["P8"]["y"]);
var lineP7P8 = CalculateShift( lineP7P8_base, double)row["Point_7"], (double)row["Point_8"], (double)row["Diameter_Outer"]);
var lineP9P10_base = new Line((double)dictionary["P9"]["x"],(double)dictionary["P9"]["y"],(double)dictionary["P10"]["x"],(double)dictionary["P10"]["y"]);
var lineP9P10 = CalculateShift(lineP9P10_base, (double)row["Point_9"], (double)row["Point_10"], (double)row["Diameter_Outer"]);
var lineP11P12_base = new Line((double)dictionary["P11"]["x"], (double)dictionary["P11"]["y"], (double)dictionary["P12"]["x"], (double)dictionary["P12"]["y"]);
var lineP11P12 = CalculateShift(lineP11P12_base, (double)row["Point_11"], (double)row["Point_12"], (double)row["Diameter_Outer"]);
var leftAngle = CalculateAngle(lineP1P2, lineP5P6);
var rightAngle = CalculateAngle(lineP7P8, lineP11P12);
var centralAngle = CalculateAngle(lineP7P8, lineP5P6);
var rad_3 = CalculateRad(lineP1P2, lineP5P6, double)row["Diameter_Outer"], (double)row["Point_3"], (double)dictionary["P3"]["x"], (double)dictionary["P3"]["y"]);
```

```
var rad_4 = CalculateRad(lineP1P2, lineP5P6, (double)row["Diameter_Outer"], (double)row["Point_4"], (double)dictionary["P4"]["x"], (double)dictionary["P4"]["y"]);
var rad_9 = CalculateRad(lineP7P8, lineP11P12, (double)row["Diameter_Outer"], (double)row["Point_9"], (double)dictionary["P9"]["x"], (double)dictionary["P9"]["y"]);
var rad_10 = CalculateRad(lineP7P8, lineP11P12, (double)row["Diameter_Outer"], (double)row["Point_10"], (double)dictionary["P10"]["x"], (double)dictionary["P10"]["y"]);
double[] point = new double[2];
CalculateBisector(lineP5P6, lineP7P8, (double)dictionary["h"], out Line bisector, out point);
var connector = new Line(bisector.B, -bisector.A, bisector.A * point[1] - bisector.B * point[0]);
double[] connector_1 = new double[]
{
    (lineP1P2.B * connector.C - connector.B * lineP1P2.C) / (lineP1P2.A * connector.B - connector.A * lineP1P2.B),
    (lineP1P2.C * connector.A - connector.C * lineP1P2.A) / (lineP1P2.A * connector.B - connector.A * lineP1P2.B)
};
double[] connector_2 = new double[]
{
    (lineP11P12.B * connector.C - connector.B * lineP11P12.C) / (lineP11P12.A * connector.B - connector.A * lineP11P12.B),
    (lineP11P12.C * connector.A - connector.C * lineP11P12.A) / (lineP11P12.A * connector.B - connector.A * lineP11P12.B)
};
double distance_between_connector_ends =
```

```
                    Math.Sqrt(
                        (connector_1[0] - connector_2[0]) * (connector_1[0] -
connector_2[0])
                        + (connector_1[1] - connector_2[1]) * (connector_1[1] -
connector_2[1]));
                double distance_symmetry =
                    Math.Sqrt(
                        (connector_1[0] - point[0]) * (connector_1[0] - point[0])
                      + (connector_1[1] - point[1]) * (connector_1[1] - point[1]));
                row["L"] = leftAngle;
                row["R"] = rightAngle;
                row["C1"] = centralAngle;
                row["LR"] = rad_3;
                row["LR_test"] = rad_4;
                row["RR"] = rad_10;
                row["RR_test"] = rad_9;
                row["A"] = distance_between_connector_ends;
                row["As"] = distance_symmetry;
            }
        }
    }
```