Министерство образования и молодежной политики
Свердловской области

Государственное автономное профессиональное образовательное
учреждение Свердловской области «Ирбитский политехникум»

КУРСОВАЯ РАБОТА

по ПМ.01 Разработка программных модулей программного обеспечения для компьютерных систем

МДК 01.01. Системное программирование

Тема: Термометрия, оптическая передача данных с термометра на языке Python

Выполнил:
студент группы ПКС-306
специальность 09.02.03
Программирование
в компьютерных системах
Останин Иван Васильевич

Проверил**:** преподаватель
Вишнякова Н.В.

Ирбит 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день язык программирования Python является одним из самых востребованных в мире. Область его применения охватывает множество сфер деятельности человека: обработка данных, искусственный интеллект, разработка приложений с интерфейсом, работа с видео и многое другое.

Также данный язык программирования обладает огромной базой пользовательских библиотек и расширений, созданных сообществом и находящихся в открытом доступе. Это расширяет возможности языка до невообразимых масштабов. Работа с сайтами, различные боты для социальных сетей, построение сложнейших приложений и кроссплатформенная разработка. Стоит отметить и то, что он обладает крайне простым синтаксисом и очень прост в освоении.

Основываясь на вышеперечисленное, было принято решение использовать Python для разработки инструмента оптической передачи данных с термометра.

Данный инструмент может пригодится в проведении различных химический и физических экспериментов для протоколирования температуры на определённом промежутке времени.

Цель курсовой работы – создание программного средства для оптической передачи данных с термометра на языке Python.

Задачи:

1. подобрать и систематизировать литературу по теме;
2. изучить принципы формирования изображения термометра ТР300;
3. рассмотреть особенности разработки программного продукта на языке Python;
4. разработать интерфейс взаимодействия пользователя с программой;
5. разработать алгоритм, реализовать его на языке программирования Python и провести тестирование и отладку программного средства;
6. оформить руководство оператора на программный продукт.

Объект исследования: Цифровой термометр.
Предмет исследования: Язык программирования Python.

Перейдём к рассмотрению языка программирования Python.

1. ОСОБЕННОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ И РАЗРАБОТКА МОДЕЛИ ПРОГРАММЫ.

1.1. Особенности языка программирования Python.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества. Язык является полностью объектно-ориентированным в том плане, что всё является объектами.

Достоинства языка программирования Python:

- Простой синтаксис;
- Кроссплатформенность;
- Открытый исходный код;
- Разнообразие библиотек;

Недостатки:

- Низкая скорость выполнения программ.

1.2. Используемые библиотеки и инструменты.

PyCharm

IDE PyCharm Community Edition 2020.2 – интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

Git

Git – распределённая система управления версиями. Она позволяет хранить всю историю изменений в проекте и удобно синхронизировать их между рабочими станциями.

OpenCV

OpenCV – библиотека для языка Python, предназначенная для работы с фото видео информацией и содержит огромное множество механизмов её обработки, включая поддержку библиотек компьютерного зрения и искусственного интеллекта.

В данной работе будут использованы только базовые механизмы этой библиотеки.

NumPy

NumPy – библиотека с открытым исходным кодом для языка Python для работы с многомерными массивами и высокоуровневыми математическими функциям, предназначенных для работы с ними.

Данная библиотека предоставляет функционал для обрезки, масштабирования и вращения изображений библиотеки OpenCV, которые и являются многомерными массивами.

MatplotLib

MatplotLib - библиотека на языке программирования Python для визуализации данных двумерной графикой, использовалась по прямому назначению.

ConfigParser

ConfigParser – встроенная библиотека python, дающая удобный доступ к конфигурационным файлам. Через эту библиотеку к программе был привязан файл конфигурации config.ini.

JSON

JSON - встроенная библиотека python для кодирования и декодирования данных JSON. Используется для экспорта данных в формат json.

Xlsxwriter

XlsxWriter – библиотека языка python предназначенная для записи файлов в формату Excel. Используется для экспорта данных в таблицу Excel.

1.3. Принципы формирования изображения на термометре.

В качестве объекта исследования стал цифровой погружной термометр ТР300. Он обладает большим диапазоном измеряемых температур (-50°С - +300°С), погрешность измерений не превышает ±1°С на всём диапазоне, точностью в 0.1°С, а также термометр оснащён щупом из нержавеющей стали.

Показания термометра отображаются на стандартном семисегментном дисплее. Для преобразования цифрового сигнала в вид, пригодный для отображения на индикаторе, используется микросхема-дешифратор К176ИД2. Она и выполняет всю работу по формированию изображения. На Рисунке 1 представлены изображения цифр, отображаемых на дисплее термометра.

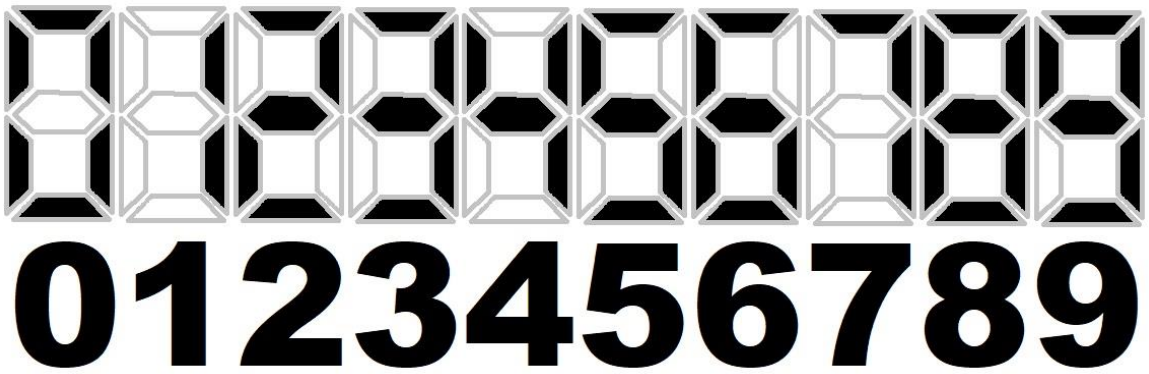


Рис. 1 Изображения цифр на дисплее термометра ТР300.

1.4. Интерфейс программы.

Работа с программной разделена на 5 этапов:

- Transforming – этап предварительной подготовки видео, в котором происходит настройка масштабирования и вращения;
- Placement – этап расстановки ключевых точек;
- Naming – этап присваивания каждой точке имени, положения её сегмента в семисегментном индикаторе и распределения их между различными цифрами индикатора;
- Scanning – этап сканирования в котором на основании всех ключевых точек собирается информация о состоянии каждого элемента индикатора, эта информация дешифруется в формат целого числа и экспортируется в удобном формате;
- Fixing – этап коррекции положения ключевых точек, на случай если в процессе исследования температуры камера и термометр сместились друг относительно друга и расположение сегментов на изображении изменилось.

Последовательность этапов представлена на схеме(Рис.2):

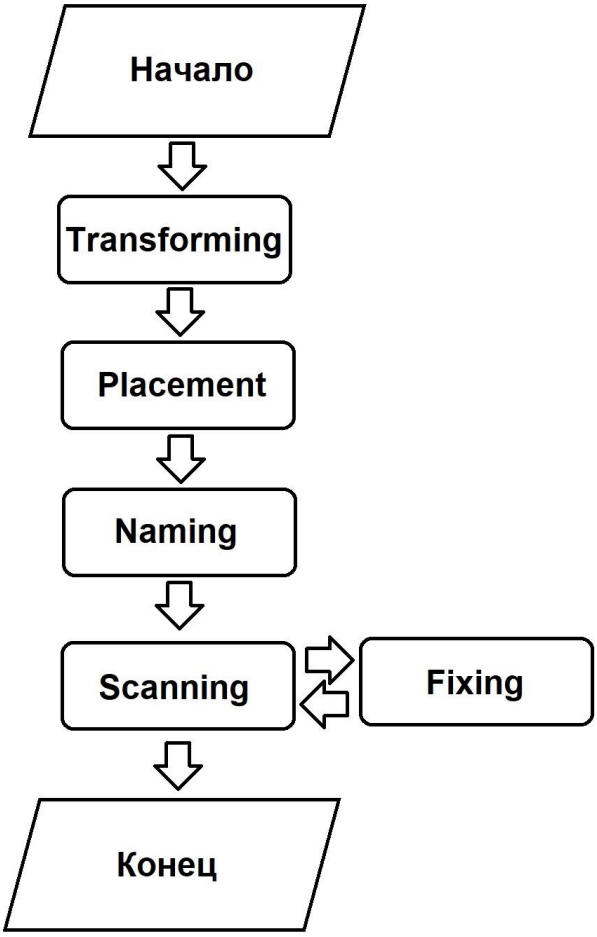


Рис. 2 Последовательность этапов работы программы.

Интерфейс программы состоит из 7 элементов(Рис.3):

1. Область просмотра исходного изображения;
2. Заголовок окна, в котором отображается текущий этап работы с программой;
3. Расставленные на исходном видео, ключевые точки. По цвету пикселя изображения определяется текущее значение на дисплее термометра;
4. Область предпросмотра текущей температуры, основанный на цвете ключевых точек;
5. Сегменты показателя;
6. Элементы управления окном.
7. Индикатор выполнения, представляющий собой прямоугольную область, которая «заполняется» областью зелёного цвета по мере сканирования.

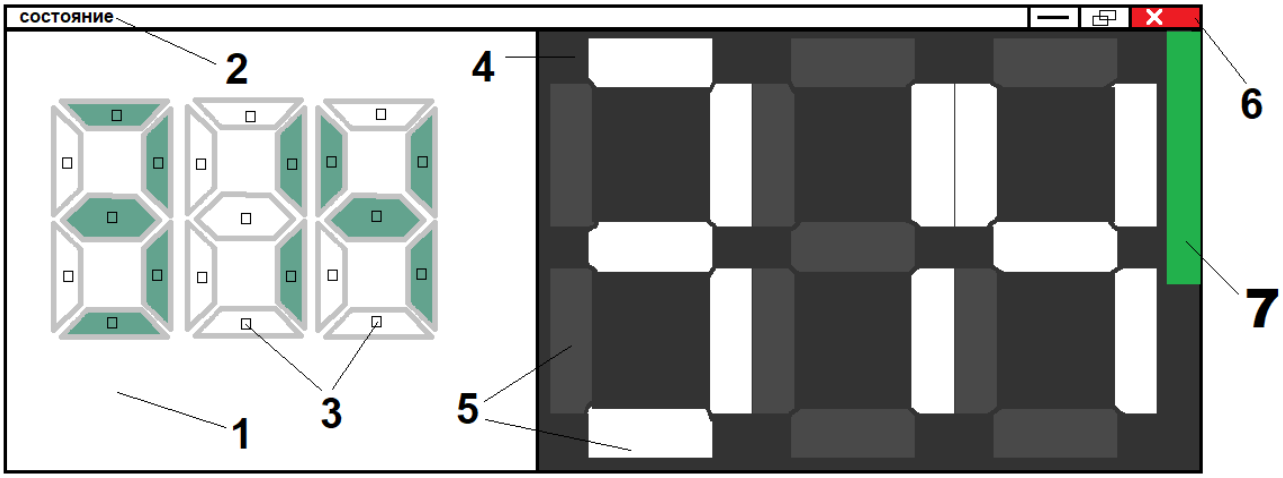


Рис. 3 Интерфейс программы.

В качестве механизма вывода полученных данных программа предоставляет 7 форматов экспорта:

1. RawTXT - текстовый файл, где все значения идут последовательно;
2. PythonList - текстовый файл, содержащий список всех значений, для исполнения в Python;
3. PythonDict - текстовый файл содержащий словарь всех значений с ключами, которые являются временем и пригодный для исполнения в Python;
4. JSON - текстовый файл JSON формата;
5. NumpyArray - бинарный файл массива библиотеки Numpy;
6. Excel – Excel таблица;
7. Graph - отображение графика на экране.

Программа настраивается через конфигурационный файл config.ini, который находится в одной директории с программой. Через него можно настроить следующие параметры:

1. videoPath – путь к видеофайлу;
2. startSec – секунда с которой начинается сканирование;
3. decimalPoint – число знаков после запятой в выходном файле;
4. exportFormat – формат экспорта выходных данных, возможные значения перечислены выше;
5. exportFileName – имя выходного файла.

Структура файла config.ini представлена на Рисунке 4.

```
[Video]

# Путь к видеофайлу
videoPath = Experiments/E-1/video.mp4

# Секунда начала сканирования
startSec = 5

# Число знаков после запятой
decimalPoint = 1

[Export]

# Возможные форматы экспорта:
#   RawTXT - текстовый файл, где все значения идут последовательно;
#   PythonList - текстовый файл со списком всех значений пригодный для исполне
#   PythonDict - текстовый файл со словарём всех значений с ключами, являющим
#   JSON - текстовый файл JSON формата;
#   NumpyArray - бинарный файл массива библиотеки numpy;
#   Excel - excel файл;
#   Graph - отобразить график на экране;

exportFormat = Excel

# имя файла экспорта (если такой должен быть) (расшенение не требуется)
exportFileName = data
```

Рис. 4 Структура файла config.ini.

В данном случае сканируемое видео находится по относительному пути `Experiments/E-1/video.mp4`, сканирование начинается с 5 секунды, и термометр имеет один знак после запятой.

Выходные данные экспортируются в формат "Excel" в файл с названием `data.xlsx`.

2. РАЗРАБОТКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ.

2.1. Разработка структуры программы.

2.1.1 Общая структура программы.

Программа разрабатывалась в методологии объектно-ориентированного программирования. Благодаря этому программу можно разделить на несколько независимых частей, каждая из которых играет определённую роль. Были созданы несколько классов. Два базовых класса:

- «VideoScaner» - Основной класс, который отвечает за обработку всей графической информации;
- «App» - класс приложения, отвечающий за работу с классом VideoScaner и экспорт полученных данных.

Также были созданы два класса, которые имеют физический аналог:

- «Segment» - объект, являющийся записью об определённом сегменте дисплея
- «Digit» - объект, являющийся записью об определённой цифре дисплея термометра и хранящий в себе массив объектов класса Segment.

Согласно руководству по написанию кода на python PEP 8, все слова в имени классов начинаются с большой буквы.

2.1.2 Структура класса VideoScaner.

2.1.2.1 Поля класса VideoScaner.

Класс VideoScaner содержит поля, представленные в таблице:

Таблица 1. Структура класса VideoScaner

Имя поля	**Тип**	**Описание**
`config`	ConfigParser	Парсер конфигурационного файла
`path`	str	Путь к видеофайлу
`capture`	cv2.VideoCapture	Объект захвата видео из видеофайла
`fps`	float	Количество кадров в секунду видео
`cropping`	((int, int), (int, int))	Координаты двух точек, прямоугольник между который и является отображается в результате обрезки
`croppingHistory`	list<((int, int), (int, int))>	История всех обрезок
`croppingArea`	list<((int, int), (int, int))>	Необработанные данные о обрезке
`state`	SetterState	Текущий этап работы программы
`scaleF`	float	Коэффициент масштабирования
`rotate`	int	Число от 0 до 3, для которого 90° * rotate, равен углу поворота изображения
`digits`	list<Digit>	Список всех цифр дисплея

noNamedSegments	list	Список ещё неназванных сегментов
segmentsHistory	list	Историй расстановки сегментов
nameHistory	list	Историй наименований сегментов
name_index	int	Подсчет номера сегмента
noNamedDigits	list<Digit>	Список ещё не названных цифр
error_count	int	Количество цифр, которые при сканировании выдали не точные данные
selection	list	Список выделенных сегментов на этапе коррекции
decimalPoint	int	Число знаков после запятой в итоговом числе, полученное из конфигурационного файла
totalFrameCount	int	Общее число кадров в видеофайле
global_scan_data	dict<int, float>	Словарь итоговых данных, где ключ - время видеофайле, значение - показания на дисплее
currentSecScan	int	Текущая секунда сканируемого файла
scan_data	list<dict< SE, bool>>	Список значений всех сегментов в своих цифрах, отправляемый на дешифровку

Далее рассмотрим механизмы данного класса.

2.1.2.2 Отображение изображения на экране.

За вывод изображения на экран отвечает метод showFrame. Его код представлен далее:

```
self.frame = self.source_img.copy()

self._cropping()
self._scale()
self._rotate()

self.sizeY, self.sizeX, _ = self.frame.shape

self._drawSegments()
self._drawBad()

cv2.imshow('Frame', self.frame)
```

Сначала в поле объекта frame мы записываем копию исходного кадра, полученного из видеофайла. Далее полученное изображение обрезается методом _cropping. Масштабируется для лучшего отображения методом _scale, и вращается методом _rotate. Ширина и высота трансформированного изображения записываются в поля sizeX и sizeY соответственно. Рисуются все сегменты и область предпросмотра, если такая необходима, используя методы _drawSegments и _drawPreview. И в конце с помощью функции imshow в окне с идентификатором “Frame” отображается полученное изображение.

Все вышеперечисленные методы не предназначены для доступа извне класса, поэтому они являются защищёнными и их идентификаторы начинаются с символа нижнего подчёркивания.

2.1.2.3 Реакция программы на события мыши.

Представлен код метода set.

```
def set(self):
    self.showFrame()
    cv2.setMouseCallback('Frame', self.onClick)

    self.transform()
    self.placement()
    self.naming()
```

В данном методе, отвечающем за вызов этапов конфигурации программы, после первого отображения кадра из видеофайла, вызывается функция модуля cv2 setMouseCallback, которая устанавливает для окна с идентификатором “Frame” метод onClick.

onClick - это функция обработки событий вызываемых мышью. В методе выполняется проверка текущего этапа работы программы из поля класс state, и, в зависимости от него, срабатывает различная логика. Описание логики работы программы рассматривается в следующих частях курсовой работы.

2.1.2.4 Общие принципы работы всех этапов.

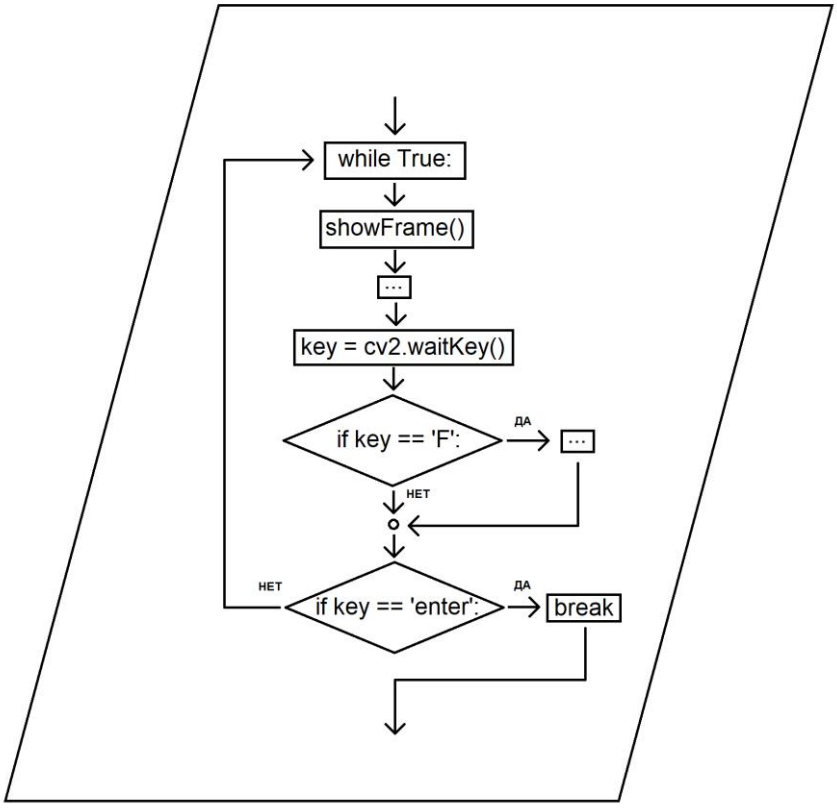


Рис. 5 Общая схема работы каждого этапа.

Все этапы являются методами основного класса и реализованы как бесконечный цикл. На каждой итерации этого цикла отрисовывается кадр изображения методом showFrame.

Программа ожидает нажатие клавиши на клавиатуре и записывает её код в переменную key. Программа анализирует код клавиши и вызывает соответствует соответствующую обработку из клавишного меню. В примере это клавиша “F”. Одновременно переменная key сравнивается с кодом клавиши “ENTER” для выхода программы из цикла и перехода к следующему этапу.

Во время ожидания нажатия клавиши программа всё также может получать события от мыши.

2.1.2.5 Этап «Transforming»

Этап «Transforming» необходим для предварительной настройки масштабирования и вращения кадра видео. По нажатию на клавишу ‘R’ (rotate, англ. - вращение) к полю «rotate» прибавляется единица и берётся остаток от деления на 4. После обновления, изображение на экране вращается на 90°.

В программе предусмотрена возможность "отката назад" сделанного действия. Для этого необходимо пользователю нажать клавишу «Backspace». Происходят следующие действия: из истории обрезки «croppingHistory» удаляется последний элемент, и становится текущим, путём сохранения в поле cropping. После обновления, изображение на экране возвращается к предыдущему.

Функция onMouse запоминает выделенную с помощью мыши область. Во время этапа «Transforming», по нажатию ЛКМ координаты курсора записываются в первый элемент списка в поле “croppingArea”, а при отжатии кнопки - во второй элемент того же списка.

Далее список подвергается обработке - выполняется проверка на размеры полученного прямоугольника. В случае успешного её прохождения, список преобразуется в неизменяемый картеж и записывается в поле cropping. Вызывается метод showFrame и изображение на экране обрезается.

2.1.2.6 Этап «Placement»

На этапе «Placement» пользователь должен расставить ключевые точки (Рис.6). В методе onMouse, если текущий этап «Placement», вызывается метод setSegment с передачей координат мыши в качестве аргумента. В методе создаётся экземпляр класса Segment по полученным координатам. Новый объект добавляется в список неназванных сегментов и в список истории сегментов. В случае нажатия клавиши ‘Backspace’, методом removeLast, удаляется последний элемент списка segmentHistory.

Рис. 6 Пример расстановки ключевых точек.

2.1.2.7 Этап «Naming»

Этап «Naming» точно присваивает каждой точке изображения положение её сегмента в семисегментном индикаторе. Точки распределяются между цифрами дисплея.

Присваивание положения сегмента внутри цифры идёт по порядку сверху вниз и слева направо. То есть сначала пользователь нажимает на верхний сегмент, за ним на левый верхний, правый верхний, средний, нижний левый, нижний правый, нижний. Наименование цифр идёт по порядку слева направо. Названные сегменты приобретают зелёную обводку, а сегменты, входящие в заполненную цифру, синюю (Рис.7)

Рис. 7 Пример названных ключевых точек.

В методе onMouse, если текущий этап «Naming», относительно позиции мыши выполняется поиск ближайшего сегмента из списка неназванных. Сегменту присваивается имя из списка всех имён по индексу “name_index”, далее к полю прибавляется единица и берётся остаток от деления на 7 - тем самым определяется номер позиции в семисегментном индикаторе.

Выход из данного этапа возможен только, если все сегменты имеют имя и список неназванных сегментов пуст, т.е все данные заполнены и четко определены.

2.1.2.8 Этап «Scanning»

Scanning – этап сканирования. На основании всех ключевых точек собирается информация о состоянии каждого элемента индикатора. Полученная информация интерпретируется классом Interrupt и сохраняется в словарь “global_scan_data”. Сегменты, входящие в цифру, которую удалось интерпретировать без проблем, имеют синюю обводку, а сегменты чья интерпретация завершилась с ошибкой жёлтую (Рис.8).

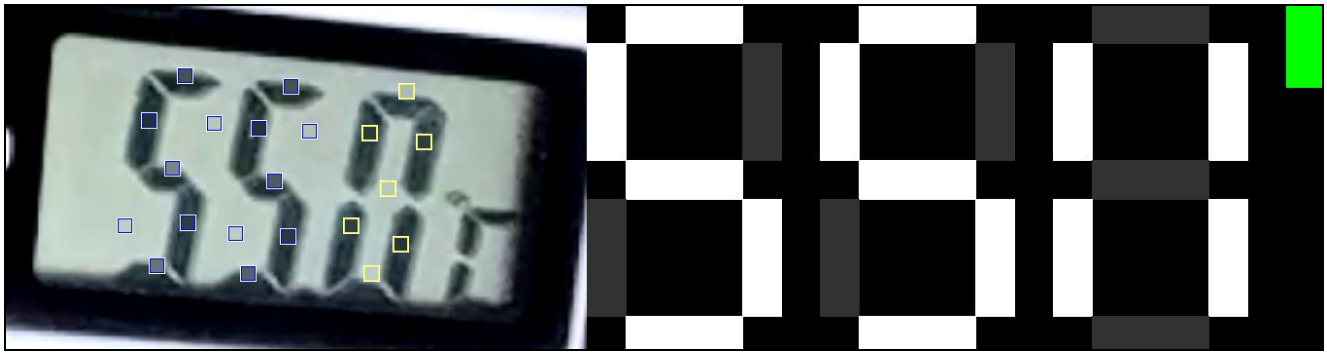


Рис. 8 Пример сканирования с ошибкой.

При нажатии клавиши ‘F’ программа перейдёт на этап «Fixing». По окончанию сканирования сырые данные возвращаются в класс App, который их экспортирует в нужный формат. Все события мыши на данном этапе игнорируются.

2.1.2.9 Этап «Fixing»

Fixing – этап коррекции положения ключевых точек. Используется в случае, если в процессе исследования температуры, камера и термометр сместились друг относительно друга и, расположение сегментов на изображении изменилось.

В Fixing реализована коррекция положения ключевых точек, с помощью суммирования координат сегментов, находящихся в списке выбранных с некоторым смещением. По нажатию клавиши на клавиатуре происходит сдвиг точек.

Нажатие ЛКМ приводит к очистке списка выбранных сегментов и записи в него ближайшего к курсору сегмента.

Нажатие ПКМ приводит к записи в список ближайшего невыбранного сегмента с сохранением всех уже выбранных.

По нажатию на клавишу ‘F’ на клавиатуре выделяются все сегменты.

По нажатию на ‘ENTER’ программа возвращается к сканированию с новыми позициями сегментов.

Сегменты, находящиеся в списке выбранных, приобретаю оранжевую обводку. (Рис. 9)

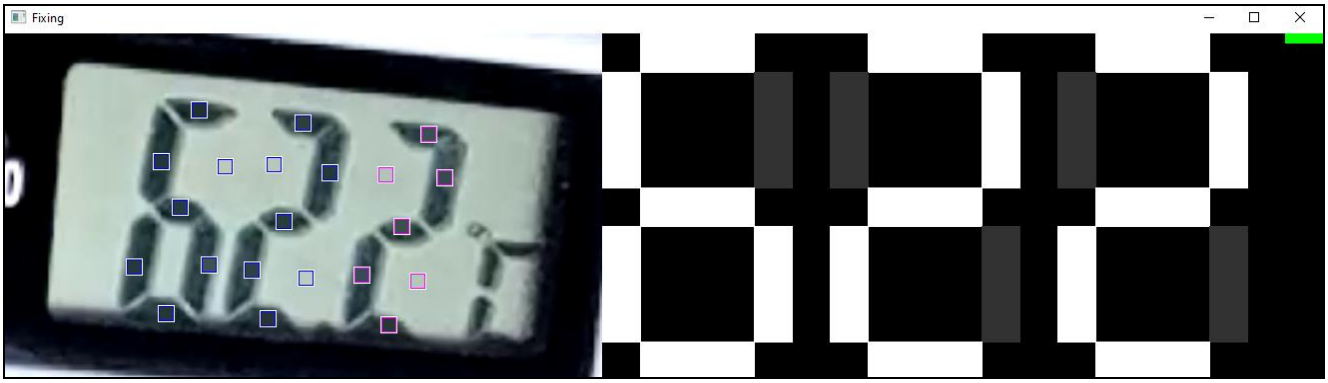


Рис. 9 Пример коррекции ошибки.

2.1.3 Структура класса «Digit».

Класс «Digit» хранит информацию об определённой цифре дисплея термометра. Поле segments класса представляет из себя массив объектов класса Segment. Класс необходим для управления сегментами: получения данных с сегментов, отправки полученных данных на интерпретацию и возвращения данных в класс «VideoScanner».

Для реализации вышеперечисленных функций у данного класса есть следующие методы:

- setSegment – запись сегмента в текущую цифру;
- removeLast – удаление последнего сегмента;
- sort – сортировка сегментов в списке данного объекта по порядку сверху вниз, слева направо;
- isFull – проверка количества сегментов у данной цифры: возвращает ИСТИНУ, если у текущего объекта 7 сегментов (т.е. все заполнено);
- isNamed – проверка, названий сегментов;
- scan – сканирование, интерпретация, возвращение полученных данных;
- interpret – статический метод по работе с классом «Interrupt», интерпретирует полученные от сегментов данные.

2.1.4 Структура класса «Segment».

Объекты класса «Segment» являются записью об определённом сегменте дисплея. Он содержит следующие поля:

- pos – поле типа картеж двух целых чисел, координаты на изображении где расположен сегмент;
- isSelected – поле логического типа, изначально принимает значение ЛОЖЬ и принимает значение ИСТИНА только в случае если данный сегмент выбран на этапе коррекции ошибок;
- name – поле, обозначающее расположение сегмента внутри своей цифры
- videoScaner – ссылка на основной объект сканера данной программы;
- digit – ссылка на цифру в которой находится данный сегмент.

Также данный класс содержит следующие методы:

- setDigit – установка для данного сегмента родительской цифры переданной как аргумент метода;
- select и deselect – методы изменяющие поле isSelected при выборе данного сегмента на этапе коррекции ошибок;
- getColor – метод возвращающий цвет пикселя на изображении по координатам из поля pos;
- draw – отрисовка сегмента на изображении;
- scan – анализ цвета на изображении по координатам сегмента;
- move – сдвиг координат сегмента на указанное смещение.

2.1.5 Структура класса «Interrupt»

Класс Interrupt необходим для дешифровки данных полученных с сегментов в целое число. Класс состоит из одного поля и одного статического метода. Таблицы dataSet, хранящей состояния всех сегментов для каждой цифры по образцу с рисунка 1. Метода find который ищет совпадение переданных в него данных и одной из строк в таблице dataSet.

2.2. Проблемы, возникшие во время разработки.

Суть проблемы

Так как класс Interrupt ищет точное совпадение полученных данных и одной из строк в таблице то если хотя бы в одном сегменте произойдёт ошибка при измерении состояния сегмента выходное значение будет непригодно для использования.

Решение

Реализация нечёткого поиска. Существует много алгоритмов нечёткого поиска, но в данном случае был реализован следующий. Если чёткого совпадения не было найдено создаётся список нулей длинной 10 элементов. И для каждой строки в таблице dataSet вычисляется значение какого количества сегментов не совпало с дешифруемыми значениями. Полученные вычисления сохраняются в список. После чего в данном списке находится минимальное значение, индекс которого и будет результатом нечёткого поиска.

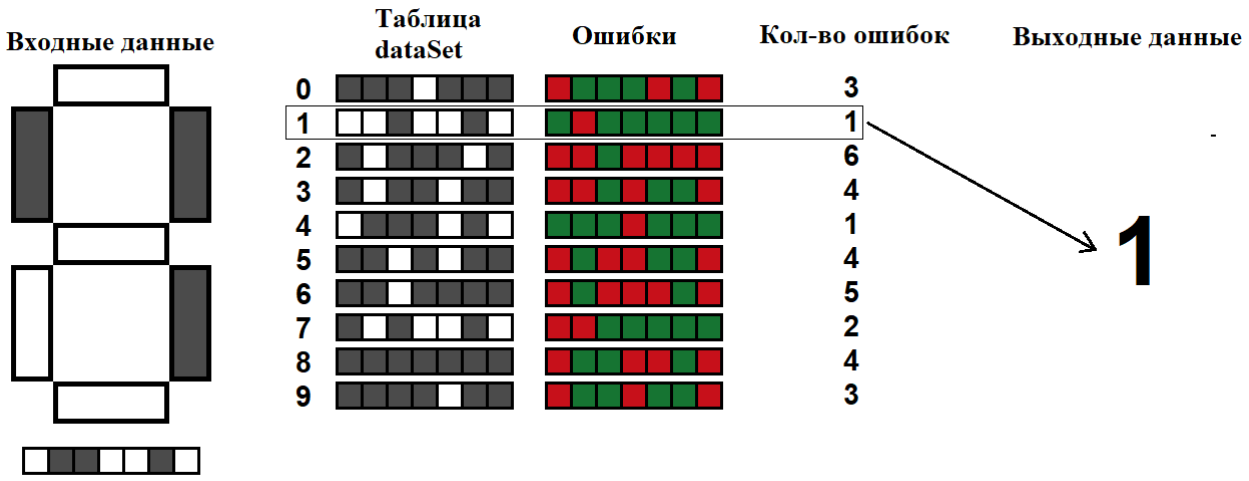


Рис. 10 Иллюстрация нечёткого поиска.

2.3. Тестирование программы.

2.3.1 Описание экспериментов.

Всего для тестирования было приготовлено 3 эксперимента:

- E-Test – чёрно-белое видео созданное с помощью компьютерной графики;
- E-ColorTest – цветное видео являющиеся копией видео эксперимента E-Test за исключением того что цвета для фона и сегментов максимально приближены к реальным, а также была добавлена ещё одна цифра повёрнутая на 90°;
- E-1 – первый реальный эксперимент.

2.3.2 Эксперименты E-Test и E-ColorTest.

Данные эксперименты проведены с целью проверки базовых механизмов программы. Оба видеоролика созданы с помощью компьютерной графики. Значения на экране идут последовательно от 0 до 9 четыре раза.

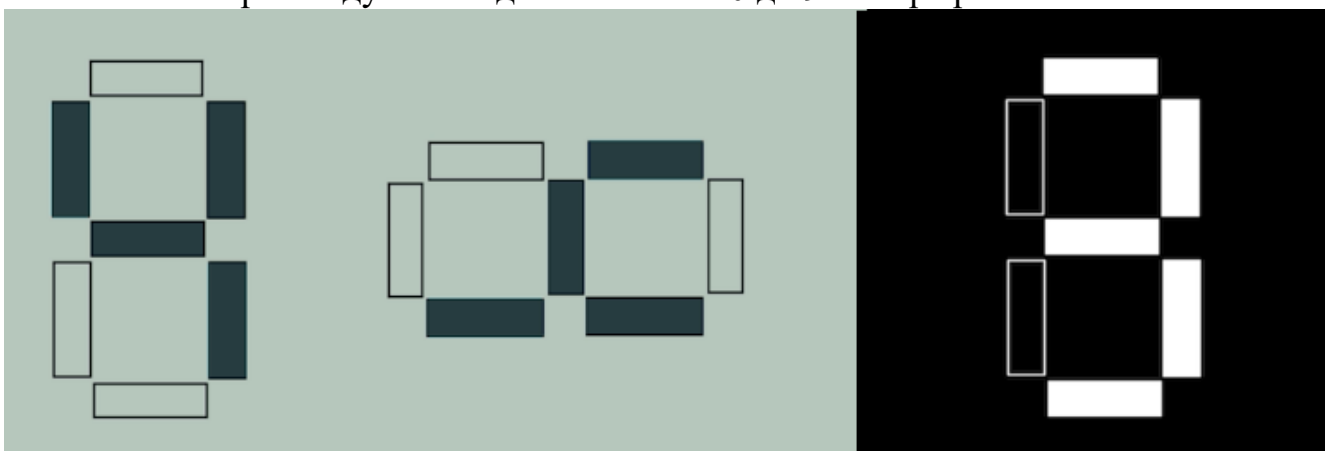


Рис. 11 E- ColorTest и E-Test.

Программа без проблем анализирует данные видеофайлы и в результате выводит следующий график (Рис.12).

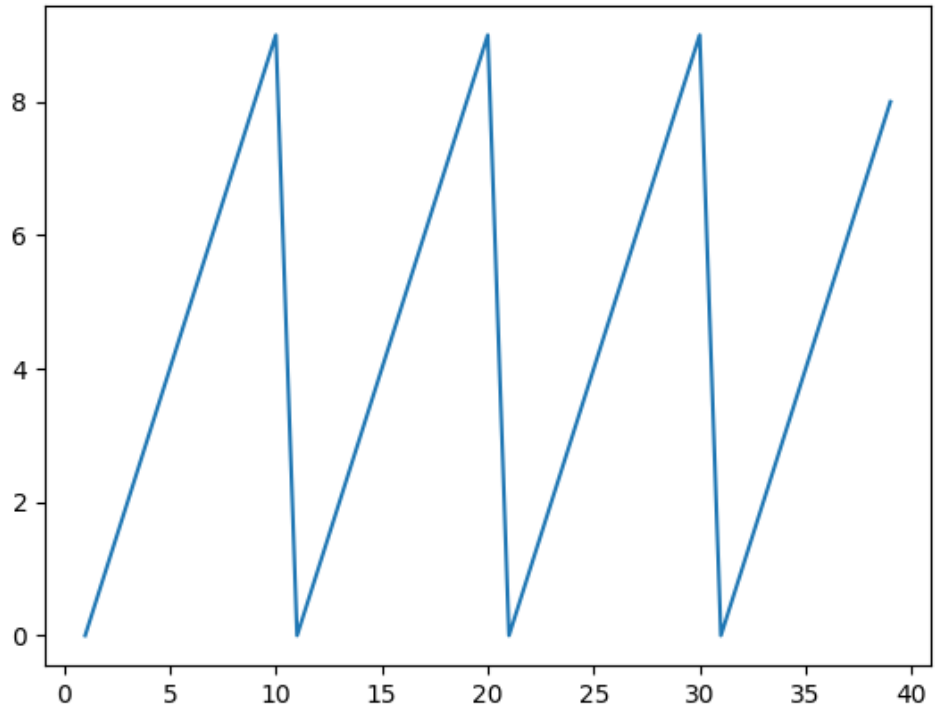


Рис 12. Результат сканирования E-ColorTest.

2.3.3 Эксперимент E-1.

Эксперимент E-1 проведён на реальном примере. Стакан кипячённой воды с начальной температурой 63.2°C постепенно остывает. Также на девятой и пятнадцатой минуте эксперимента в стакан было добавлено 10 мл холодной воды.

Рис. 13 Фрагмент из эксперимента E-1.

Результат сканирования (Рис. 14):

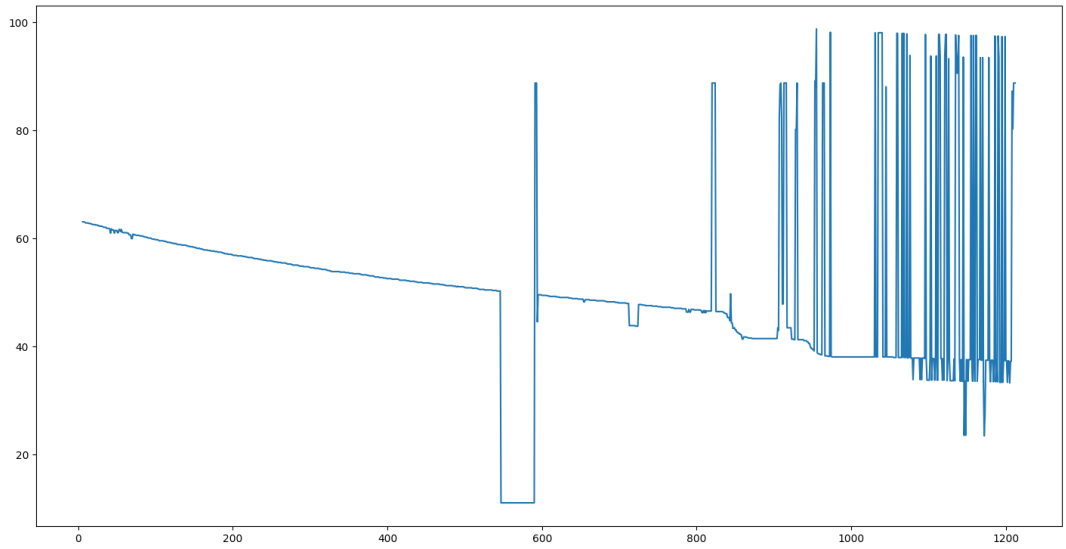


Рис. 14 Результат сканирования E-1.

Внешние обстоятельства внесли в исходное видео такие дефекты как: расфокус камеры, авто-выключение термометра на 550 секунде и изменение уровня освещения. Из-за всего вышеперечисленного удачное сканирование не было возможно изначально. Но в целом, считаю, результат удовлетворительным.

3. РУКОВОДСТВО ОПЕРАТОРА

Перед началом использования программы необходимо её сконфигурировать. Для этого откройте файл config.ini, находящийся в одной директории с программой. Пример файла config.ini:

```
[Video]

# Путь к видеофайлу
videoPath = Experiments/E-1/video.mp4

# Секунда начала сканирования
startSec = 5

# Число знаков после запятой
decimalPoint = 1

[Export]

# Возможные форматы экспорта:
#   RawTXT - текстовый файл, где все значения идут последовательно;
#   PythonList - текстовый файл со списком всех значений пригодный для исполнения в python;
#   PythonDict - текстовый файл со словарём всех значений с ключами, являющимися временем и пригодный для исполнения в python;
#   JSON - текстовый файл JSON формата;
#   NumpyArray - бинарный файл массива библиотеки numpy;
#   Excel - excel файл;
#   Graph - отобразить график на экране;

exportFormat = Graph

# имя файла экспорта (если такой должен быть) (расшенение не требуется)
exportFileName = data
```

Установите путь к видео в параметр videoPath. В параметр startSec введите секунду начала сканирования. Количество знаков после запятой на дисплее в параметр decimalPoint.

Выберите подходящий формат экспорта из представленных возможных и впишите его в параметр exportFormat. При необходимости, измените имя выходного файла в параметре exportFileName.

На этом настройка начальной конфигурации окончена.

Запускаем основной исполняемый файл MyScanner.exe.

Работа с программой разделена на 5 этапов:

1. Transforming (трансформация)
2. Placement (расстановка)
3. Naming (наименование)
4. Scanning (сканирование)
5. Fixing (исправление ошибок)

В открывшемся окне отображается кадр из исходного файла. На этапе Transforming необходимо обрезать всё лишнее и повернуть изображение для более удобной расстановки ключевых точек в будущем.

Для обрезки прямоугольной области необходимо встать курсором в первый угол прямоугольника, нажать и удерживать ЛКМ, перевести курсор к противоположному углу, отпустить ЛКМ.

По нажатию на клавишу «Backspace» обрезка вернётся к предыдущей. Нажатие на клавишу «R» повернёт изображение на 90°.

Пример результата:

По нажатию на клавишу «Enter» программа перейдёт на следующий этап.

На этапе «Placement» вам необходимо расставить ключевые точки. Они ставятся по нажатию на ЛКМ в позиции под курсором. Так нужно отметить все сегменты на дисплее. Пример результата:

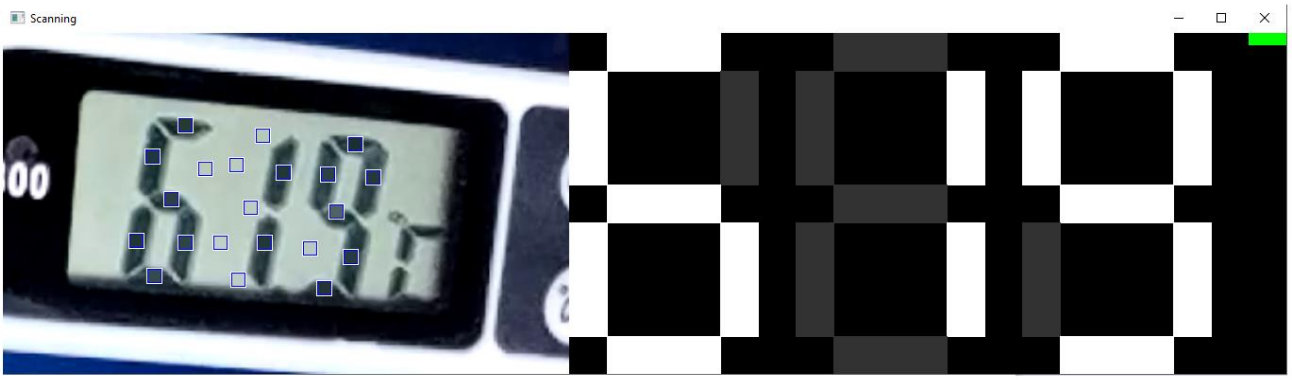
Нажмите «Enter» для перехода на этап «Naming».

На этом этапе нужно отметить все точки в определённой последовательности. Сначала точки первой цифры на дисплее, затем второй и так далее. Внутри каждой цифры их нужно отмечать в порядке сверху вниз, слева направо. То есть: верхняя, правая верхняя, средняя, нижняя левая, нижняя правая, нижняя.

Точки отмечаются нажатием на ЛКМ. Для отмены нажмите «Backspace». Отмеченные точки приобретают зелёную обводку, а точки, входящие в заполненную цифру, синюю. Пример заполнения:

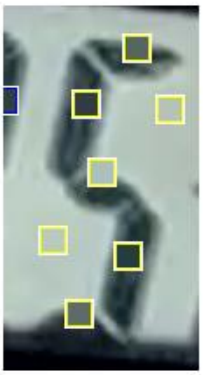
Переход на следующий этап также происходит по нажатию клавиши «Enter», но он возможен только в случае, если все точки поименованы и имеют синюю обводку.

На этапе «Scanning» начинается сам процесс сканирования.



На данном этапе вы должны следить за процессом сканирования и при необходимости исправлять возникающие ошибки. В правой части окна отображаются данные, полученные с ключевых точек, и индикатор выполнения, заполняющийся по мере сканирования.

Сегменты цифр, которые программа не может точно распознать, окрашиваются в жёлтый. Пример ошибки:

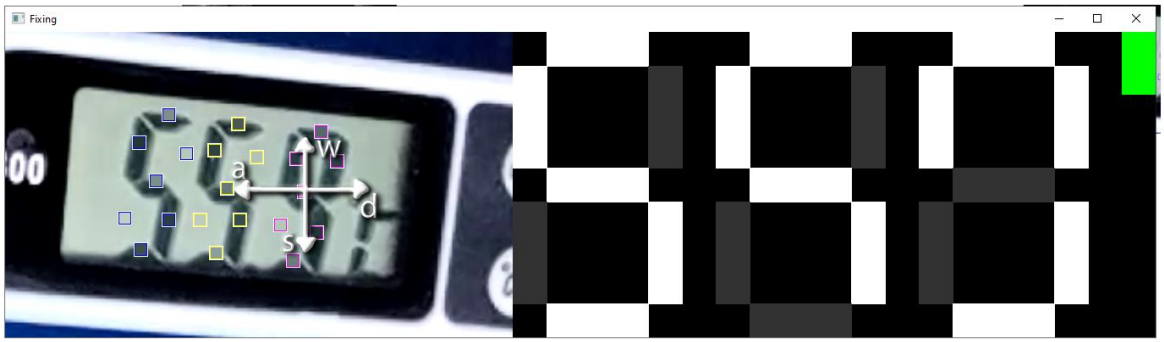


Для исправления возникших ошибок нажмите клавишу «F», после чего вы перейдёте на этап «Fixing» и сканирование приостановится.

На этапе коррекции ошибок выберете точки, которые необходимо сместить. Нажатие на ЛКМ выберет ближайшую точку, ПКМ выберет ближайшую точку с сохранением уже выбранных, клавиша «F» выбор всех.

Выбранные точки смещаются на 2 пикселя вверх по нажатию «W», влево «A», вправо «D», вниз «S».

После введённых исправлений возвращаемся к сканированию нажатием клавиши «Enter».



По окончанию сканирования полученные данные экспортируются в формат, указанный в файле confing.ini.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были изучены принципы формирования изображения термометра ТР300; рассмотрены особенности разработки программного продукта на языке Python; разработан интерфейс взаимодействия пользователя с программой; разработан алгоритм и реализован на языке программирования Python, проведено тестирование и отладка программного продукта; оформлено руководство оператора на программный продукт.

Результатом курсовой работы является разработанная программа для дешифровки данных с дисплеев электронных устройств. Данный продукт позволяет: экспортировать данные в различные форматы, проводить коррекцию ошибок на этапе сканирования. Одним из достоинств программы является простота в освоении и не требование от пользователя специальных знаний.

На данном этапе программа выполняет поставленные перед ней задачи, но её всё ещё можно модифицировать. Например, автоматически расставлять ключевые точки на основе шаблона их расположения, ввести систему автоматической коррекции ошибок, а также постобработку выходных данных.

Благодаря вышеперечисленному была выполнена цель курсовой работы, создано программного средства для оптической передачи данных с термометра на языке Python.

В процессе выполнения курсовой работы закреплены навыки объектно-ориентированного программирования, получен практический опыт разработки алгоритма поставленной задачи средствами языка программирования Python, выполнена отладка и тестирование программы на уровне модуля.

СПИСОК ЛИТЕРАТУРЫ

1. Python docs URL: https://docs.python.org/3/
2. Numpy docs URL: https://numpy.org/doc/
3. OpenCV docs for python URL: https://docs.opencv.org/4.x/d6/d00/tutorial.html
4. OpenCV docs URL: https://docs.opencv.org/
5. PEP8 for Python URL: https://www.python.org/dev/peps/pep-0008/
6. Вишнякова Н.В - Методические рекомендации для обучающихся по выполнению курсового проекта по ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» для специальности 09.02.03 «Программирование в компьютерных системах», 12 стр
7. Git docs URL:https://git-scm.com/docs
8. Pycharm docs URL: https://www.jetbrains.com/help/pycharm/quick-start-guide.html